Graduate AI
Lecture 2:
Search I

Teachers:
Zico Kolter
Ariel Procaccia (this time)
SEARCH PROBLEMS

• A search problem has:
  - States (configurations)
  - Start state and goal states
  - Successor function: maps states to (action, state, cost) triples
EXAMPLE: PANCAKES

© North-Holland Publishing Company

BOUNDS FOR SORTING BY PREFIX REVERSAL

William H. GATES
Microsoft, Albuquerque, New Mexico

Christos H. PAPADIMITRIOU*†
Department of Electrical Engineering, University of California, Berkeley, CA 94720, U.S.A.

Received 18 January 1978
Revised 28 August 1978

For a permutation $\sigma$ of the integers from 1 to $n$, let $f(\sigma)$ be the smallest number of prefix reversals that will transform $\sigma$ to the identity permutation, and let $f(n)$ be the largest such $f(\sigma)$ for all $\sigma$ in (the symmetric group) $S_n$. We show that $f(n) \leq (5n + 5)/3$, and that $f(n) \geq 17n/16$ for $n$ a multiple of 16. If, furthermore, each integer is required to participate in an even number of reversed prefixes, the corresponding function $g(n)$ is shown to obey $3n/2 - 1 \leq g(n) \leq 2n + 3$. 
Example: Pancakes
Example: 8-Puzzle
EXAMPLE: PATHFINDING
Tree Search

function TREE-SEARCH(problem, strategy)
set of frontier nodes contains the start state of problem
loop
• if there are no frontier nodes then return failure
• choose a frontier node for expansion using strategy
• if the node contains a goal then return the corresponding solution
• else expand the node and add the resulting nodes to the set of frontier nodes
Tree Search

- Tree search can expand many nodes corresponding to the same state
- In a rectangular grid:
  - Search tree of depth \( d \) has \( 4^d \) leaves
  - There are only \( 4d \) states at Manhattan distance exactly \( d \) from any given state
Algorithms that forget their history are doomed to repeat it!
**Graph Search**

function `GRAPH-SEARCH(problem, strategy)`

set of frontier nodes contains the start state of `problem`

loop

- **if** there are no unexpanded frontier nodes **then return** failure
- choose an unexpanded frontier node for expansion using `strategy`, and add it to the expanded set
- **if** the node contains a goal **then return** the corresponding solution
- **else** expand the node and add the resulting nodes to the set of frontier nodes, **only if not in the expanded set**
Uninformed
Can only generate successors and distinguish goals from non-goals

Informed
Strategies that know whether one non-goal is more promising than another
**Uniform Cost Search**

- **Strategy:** Expand by $g(x) =$ work done so far

![Diagram of a graph with nodes labeled and edges labeled with costs](image)

- Nodes: $s$, $a$, $b$, $c$, $d$, $e$, $t$
- Edges with costs: $1$, $2$, $3$, $5$, $1$
- Costs: $g(s) = 0$, $g(a) = 1$, $g(b) = 2$, $g(d) = 4$, $g(e) = 6$, $g(c) = 3$, $g(t) = 7$

#1, #2, #3, #5, #6, #4, #7
Uninformed vs. Informed

**Uninformed**
Can only generate successors and distinguish goals from non-goals

**Informed**
Strategies that know whether one non-goal is more promising than another
**Example: Heuristic**

<table>
<thead>
<tr>
<th>City</th>
<th>Aerial dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arad</td>
<td>366</td>
</tr>
<tr>
<td>Sibiu</td>
<td>253</td>
</tr>
<tr>
<td>Rimnicu Vilcea</td>
<td>193</td>
</tr>
<tr>
<td>Fagaras</td>
<td>176</td>
</tr>
<tr>
<td>Pitesti</td>
<td>100</td>
</tr>
</tbody>
</table>
**GREEDY SEARCH**

- **Strategy:** Expand by $h(x) = \text{heuristic evaluation of cost from } x \text{ to goal}$
A* Search

- Strategy: Expand by $f(x) = h(x) + g(x)$
- Poll 1: Which node is expanded fourth?

1. d
2. e
3. t
4. c
A* Search

• Should we stop when we discover a goal?

• No: Only stop when we expand a goal
**A* SEARCH**

- Is A* optimal?

- Good path has pessimistic estimate
- Circumvent this issue by being optimistic!

---

Slide adapted from Dan Klein
Admissible Heuristics

• $h$ is admissible if for all nodes $x$,
  $$h(x) \leq h^*(x),$$
  where $h^*$ is the cost of the optimal path to a goal

• Example: Aerial distance in the pathfinding example

• Example: $h \equiv 0$
**Optimality of A***

- **Theorem:** A* tree search with an admissible heuristic returns an optimal solution
- **Proof:**
  - Assume suboptimal goal \( t \) is expanded before optimal goal \( t^* \)
**Optimality of A**

- **Proof (cont.):**
  - There is a node $x$ on the optimal path to $t^*$ that has been discovered but not expanded
  - $f(x) = g(x) + h(x) \leq g(x) + h^*(x)$
    - $= g(t^*) < g(t) = f(t)$
  - $x$ should have been expanded before $t! \blacksquare$
8-puzzle Heuristics

• $h_1$: #tiles in wrong position
• $h_2$: sum of Manhattan distances of tiles from goal
• Poll 2: Which heuristic is admissible?
  1. Only $h_1$
  2. Only $h_2$
  3. Both $h_1$ and $h_2$
  4. Neither one
Heuristic for designing admissible heuristics: relax the problem!
8-puzzle Heuristics

- $h_1$: #tiles in wrong position
- $h_2$: sum of Manhattan distances of tiles from goal
- $h$ dominates $h'$ iff $\forall x, h(x) \geq h'(x)$
- Poll 3: What is the dominance relation between $h_1$ and $h_2$?
  1. $h_1$ dominates $h_2$
  2. $h_2$ dominates $h_1$
  3. $h_1$ and $h_2$ are incomparable
8-puzzle Heuristics

- The following table gives the number of nodes expanded by A* with the two heuristics, averaged over random 8-puzzles, for various solution lengths

<table>
<thead>
<tr>
<th>Length</th>
<th>$A^*(h_1)$</th>
<th>$A^*(h_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1301</td>
<td>211</td>
</tr>
<tr>
<td>18</td>
<td>3056</td>
<td>363</td>
</tr>
<tr>
<td>20</td>
<td>7276</td>
<td>676</td>
</tr>
<tr>
<td>22</td>
<td>18094</td>
<td>1219</td>
</tr>
<tr>
<td>24</td>
<td>39135</td>
<td>1641</td>
</tr>
</tbody>
</table>

- Moral: Good heuristics are crucial!
A* Graph Search

• Recall: Graph search is the same as tree search, but never expand a node twice
• Is optimality of A* under admissible heuristics preserved? No!
**Consistent Heuristics**

- \( c(x, y) = \) cost of cheapest path between \( x \) and \( y \)
- \( h \) is consistent if for every two nodes \( x, y \),
  \[ h(x) \leq c(x, y) + h(y) \]
- Assume \( h(t) = 0 \) for each goal \( t \)
- **Poll 4:** What is the relation between admissibility and consistency?
  1. Admissible \( \Rightarrow \) consistent
  2. Consistent \( \Rightarrow \) admissible
  3. They are equivalent
  4. They are incomparable
8-puzzle Heuristics, Revisited

- $h_1$: #tiles in wrong position
- $h_2$: sum of Manhattan distances of tiles from goal
- **Poll 5:** Which heuristic is consistent?
  1. Only $h_1$
  2. Only $h_2$
  3. Both $h_1$ and $h_2$
  4. Neither one
Heuristic for designing consistent heuristics: design an admissible heuristic!
Optimality of A*, Revisited

- Theorem: A* graph search with a consistent heuristic returns an optimal solution

- Proof sketch:*
  - Assume $h(x) \leq c(x, y) + h(y)$
  - Values of $f(x)$ on a path are nondecreasing: if $y$ is the successor of $x$, $f(x) = g(x) + h(x) \leq g(x) + c(x, y) + h(y) = g(y) + h(y) = f(y)$
  - When A* selects $x$ for expansion, the optimal path to $x$ has been found: otherwise there is a frontier node $y$ on optimal path to $x$ that should be expanded first
  - Nodes expanded in nondecreasing $f(x)$
  - First goal state that is expanded must be optimal ■

* Just for fun
SUMMARY

• Terminology and algorithms:
  o Search problems
  o Tree search, graph search, uniform cost search, greedy, A*
  o Admissible and consistent heuristics

• Theorems:
  o A* tree search is optimal with admissible $h$
  o A* graph search is optimal with consistent $h$

• Big ideas:
  o Don’t be too pessimistic!