CMU 15-781
Lecture 4:
Informed Search
Teacher:
Gianni A. Di Caro
**UNINFORMED VS. INFORMED**

*Uninformed*

Can only generate successors and distinguish goals from non-goals

*Informed*

Strategies that can distinguish whether one non-goal is more promising than another
Reminder: Tree Search

function TREE-SEARCH(problem, strategy)

set of frontier nodes contains the start state of problem

loop

if there are no frontier nodes

return failure

choose a frontier node for expansion using strategy

if the node contains a goal

return corresponding solution

else expand the node

add the resulting nodes to the set of frontier nodes
ReCAP:
Strategies for Uninformed Search

• BFS: Shallowest unexpanded node
• DFS / IDP: Deepest unexpanded node
• UCS: Lowest cost-to-come (from start)
• LS: Local highest-value successor node
ESTIMATE OF COST-TO-GO

I need a cost estimate on your project.

I have no idea, I haven't even gathered the user requirements.

Don't worry, I won't hold you to the estimate.

Yes you will, you will put it in the plan. Forget we had this conversation. And fire me when I go over budget.

Give me a number or I'll fire you right now.

Okay, it will cost ten million dollars.

That's too high. If you already know the cost, why are you asking me?

So you'll feel like you had input. Is input supposed to feel this bad?
STRATEGY: BEST-FIRST SEARCH

• General strategy template for informed search

• A node $n$ in the frontier set is selected for expansion based on an evaluation function $f(n)$, which is a cost estimate

• Backward (to-come) and (estimates of) Forward (to-go) costs

• The node with the lowest cost estimate is expanded first

• Data structure: Priority queue using $f(n)$ for ordering
UNIFORM COST SEARCH

• Strategy: Expand by $f(x) = g(x) = \text{cost-to-come}$
**UCS vs. Dijkstra’s Algorithm**

- All nodes are initially inserted into the PQ (*explicit graph description* is given as input)
- Initial distance \((g(s), \text{cost-to-come})\) from start:
  \[ d[s] = 0, \quad d[x \neq s] = \infty \]
- The node with the *minimal estimated distance* is selected at each step
- Shortest paths to all other nodes or to a single one
  - Explicit graph description is not given as input
  - Nodes are inserted to the PQ *lazily* during the search, based on node expansion choices
  - Can naturally handle the presence of *multiple goals*
**Dijkstra’s algorithm (1959)**

**Input:** Graph $G=(V,E)$
$\forall x \neq s \quad dist[x] = +\infty$
$dist[s] = 0$

$S = \emptyset$
$Q = V$  // Ordered by $dist[]$

while $Q \neq \emptyset$ do
  $u = extract\_min(Q)$
  $S = S \cup \{u\}$
  foreach vertex $v \in Adjacent(u)$ do
    $dist[v] = \min(dist[v], dist[u] + c(u,v))$  // “Relaxation”
  end do
end do
Problem description + Heuristic Knowledge

• So far, only problem description (successors, step costs) has been used to search the state space
• What about using (also) additional, heuristic knowledge, $h(x)$, to direct state expansion by looking forward?

Heuristics are rules of thumb, educated guesses, intuitive judgments or, simply, common sense

The term derives from the ancient Greek keuriskein, meaning serving to find out, or discover. Archimedes’ Eureka! means “I have found it!”
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 Best-First Search**

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

![Diagram of Greedy Best-First Search]

- **Nodes:**
  - **S**
  - **a**
  - **b**
  - **c**
  - **d**
  - **e**
  - **t**

- **Heuristic Values:**
  - $h(S) = 6$
  - $h(a) = 5$
  - $h(b) = 6$
  - $h(c) = 7$
  - $h(d) = 2$
  - $h(e) = 1$
  - $h(t) = 0$

- **Edges:**
  - **S to a:** $h = 6$
  - **a to b:** $h = 5$
  - **a to d:** $h = 3$
  - **b to d:** $h = 5$
  - **b to e:** $h = 1$
  - **d to t:** $h = 2$
  - **d to e:** $h = 1$

**Order of Expansion:**
1. $S$ (expanded first
2. $a$
3. $d$
4. $e$
5. $t$
A* SEARCH (1968)

• **Strategy:** Combine *cost-to-come* (past) and *heuristic estimate of cost-to-go* (future), expand by \( f(x) = h(x) + g(x) \)

• **Poll 1:** Which node is expanded fourth?

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

- Should we stop when we discover a goal?

- No: Only stop when we *expand* a goal! (same as in UCS)

Slide adapted from Dan Klein
A* SEARCH

• Is A* optimal?

• The good path has a 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.

  An admissible heuristic is a lower bound on real cost.

- Example: Aerial distance in the path finding example
- Example: $h \equiv 0$
- $\rightarrow$ The tighter the bound, the better
**Optimality of A***

- **Theorem:** A* tree search with an admissible heuristic returns an optimal solution

- **Proof (by contradiction):**
  - Assume that a suboptimal goal $t$ is expanded before the optimal goal $t^*$
OPTIMALITY OF A*

• Proof (cont.):
  o There is a node \( x \) in the frontier, on the optimal path to \( t^* \) that has been discovered but not expanded yet
  o \( f(x) = g(x) + h(x) \leq g(x) + h^*(x) \)
  o But since \( x \) is on the optimal path to \( t^* \):
    \[ = g(t^*) < g(t) = f(t) \quad (h(t)=0) \]
  o \( x \) should have been expanded before \( t \)!

Slide adapted from Dan Klein
8-puzzle Heuristics

- Defining a “good” heuristic it’s not a trivial task …
- $h_1$: #tiles in wrong position [$h_1(s) = 5$]
- $h_2$: sum of Manhattan distances of tiles from goal [$h_1(s) = 2 + 0 + 1 + 3 + 2 + 2 + 0 + 0 = 10$]
- 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!

Relaxation:
Remove functional / domain constraints
→ Add “forbidden” moves
DOMINANCE

- $h_1$: #tiles in wrong position
- $h_2$: sum of Manhattan distances of tiles from goal
- $h$ dominates $h'$ iff $\forall x, h(x) \geq h'(x)$
  ($h$ is consistently a tighter bound wrt $h'$)

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 search cost of 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!
**Graph Search**

- In tree search, expanding the same state multiple times can cause exponentially more work.

---

![Graph Search Diagram]

- Tree representation of a graph with repeated states.
  - Node 'a' connected to 'b' and 'c'.
  - Node 'b' connected to 'c'.
  - Node 'a' directly connected to 'b' and 'c' twice.

---
Graph Search

- Same as tree search, but never expand a node twice
- E.g., in BFS expanding the circled nodes is not necessary
- Set of already expanded nodes has to be stored in memory

Slide adapted from Dan Klein
GRAPH SEARCH AND OPTIMALITY

- Is optimality of A* under admissible heuristics preserved? No!

Search tree

Slide adapted from Dan Klein
CONSISTENT HEURISTICS

- \( c(x, y) = \) real cost of cheapest path between \( x \) and \( y \)
- \( h \) is consistent if for every two nodes \( x, y \),
  \[
  h(x) \leq c(x, y) + h(y)
  \]
- Triangle inequality
- Necessary for graph search optimality
**Consistent Heuristics**

- "Consistency": The estimated distance to the goal from $x$ cannot be reduced by moving to a different state $y$ and adding the estimate of the distance to the goal from $y$ to the cost of reaching $y$ from $x$

- $c(x, y) \geq h(x) - h(y) \rightarrow$ The real cost is higher than the cost implied by the heuristics

What was the problem?
**Consistent Heuristics**

- **Poll 4**: What is the relation between admissibility and consistency?
  1. Admissible $\implies$ consistent
  2. Consistent $\implies$ admissible
  3. They are equivalent
  4. They are incomparable

- $h$ is consistent if for every two nodes $x, y$,
  \[ h(x) \leq c(x, y) + h(y) \]
**Consistency → Monotonicity**

- **Lemma:** If $h(x)$ is consistent, then the values of the cost function $f(x)$ along any path are nondecreasing.

- **Proof:**
  - If $y$ is a successor of $x$: $g(y) = g(x) + c(x,y)$
  - By consistency: $f(y) = g(y) + h(y)$
    
    
    
    $$f(y) = g(x) + c(x,y) + h(y)$$
    
    $$
    
    
    
    
    \geq g(x) + h(x) = f(x) \blacksquare$$

In moving from a state to its neighbor, (a consistent) $h$ must not decrease more than the cost of the edge that connects them. Consistency is a property of $h(x)$, monotonicity is a property of $f(x)$.
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

Example state

Goal state
Heuristic for designing consistent heuristics: design an admissible heuristic!
ADMISSIBLE BUT INCONSISTENT HEURISTICS?

- Keep the LB property, but violate monotonicity
- Inconsistent for at least one pair of states

\[ f(p) = 10 \]
\[ f(c_1) = 8 \]

Manhattan distance for set \{1,2,3,4\}
Manhattan distance for set \{5,6,7,8\}

At each step, choose at random which set
What is the relation between heuristic estimates?
Theorem: A* graph search with a consistent heuristic returns an optimal solution

Proof:
- Whenever A* selects a state \( x \) for expansion, the optimal path to \( x \) has been found. Otherwise, there would be a frontier node \( y \) (separation property) on the optimal path from start to \( x \) that should be expanded first because \( f \) is non decreasing along any path (monotonicity).
- The first goal state \( x^* \) selected for expansion must be optimal, because \( f(x^*) \) is the true (optimal) cost for goal nodes \( (h(x^*) = 0) \), and any other later goal node would be at least as expensive because of \( f \) monotonicity.
**Contours in the State Space**

- \( f \)-costs are nondecreasing along any path from the start → Contours (isolines) in the state-space, like in topographic maps
- A* search fans out adding nodes in circoncentric bands of increasing \( f \)-cost
- The tight the LB bounds are, the more the bands will stretch toward the goal state
- Bands using UCS?
• If $C^*$ is the cost of the optimal path, $A^*$ expands all nodes with $f(x) < C^*$, and no nodes with $f(x) > C^*$ (automatic pruning)

• $A^*$ might expand some of the nodes on the goal contour, where $f(x) = C^*$, before selecting the goal node

• **Completeness?** Yes, if only a finitely many nodes with cost less or equal to $C^*$ are present ($b$ is finite and all step costs are $\varepsilon > 0$)
**A* IS OPTIMALLY EFFICIENT**

- **Theorem:** Any algorithm that returns the optimal solution given a consistent heuristic will expand all nodes **surely** expanded by A*

- But this is not the case when the heuristic is only admissible

Alg B: Conduct exhaustive search except for expanding a; then expand a only if it has the potential to sprout cheaper solution
SUMMARY

• Terminology:
  o Search problems
  o Algorithms: tree search, graph search, best-first search, uniform cost search, greedy, A*
  o Admissible and consistent heuristics

• Big ideas:
  o Properties of the heuristic ⇒ A* optimality
  o Don’t be too pessimistic!
  o Be consistent!
PROJECTS

• Proposals to be submitted by October 24
• 1-2 pages stating:
  o Motivation
  o Goals
  o Work plan
• 40 hours of work
• Can be done in pairs
• Poster presentation only, at the end of the semester