CSE 573: Artificial Intelligence

Problem Spaces & Search

Dan Weld

With slides from
Dan Klein, Stuart Russell, Andrew Moore, Luke Zettlemoyer, Dana Nau…
Search Problems

Uninformed Search Methods
- Depth-First Search
- Breadth-First Search
- Iterative Deepening Search
- Uniform-Cost Search

Heuristic Search Methods
- Uniform Cost
- Greedy
- A*
- IDA*

Heuristic Generation
Search thru a Problem Space (aka State Space)

- **Input:**
  - Set of states
  - Operators [and costs]
  - Start state
  - Goal state [or test]

- **Output:**
  - Path: start ⇒ a state satisfying goal test
    - [May require shortest path]
    - [Sometimes just need a state that passes test]

**Functions:** States → States
Aka “Successor Function”
N Queens Problem

Place N queens so they don’t attack each other (same row, same col, same diagonal)

- **States**
  - Chess board with 0 or more queens

- **Operators**
  - Add a queen

- **Initial**
  - No queens

- **Goal**
  - N queens
Getting a PhD In CSE

Input:
- Set of states
- Operators [costs]
- Start state
- Goal state (test)
Best-First Search

- Generalization of breadth-first search
- Fringe = *Priority* queue of nodes to be explored
- Ranking function $f(n)$ applied to each node

Add initial state to priority queue
While queue not empty
  Node = head(queue)
  If goal?(node) then return node
  Add new children of node to queue, sorted

“expanding the node”
Old Friends

- **Breadth First** =
  - Best First
  - with $f(n) = \text{depth}(n)$

- **Dijkstra’s Algorithm (Uniform cost)** =
  - Best First
  - with $f(n) = \text{the sum of edge costs from start to } n$
What is a *Heuristic*?

- An *estimate* of how close a state is to a goal
- Designed for a particular search problem

- Examples: Manhattan distance: 10+5 = 15
  - Euclidean distance: 11.2
Greedy Search

Expand the node that seems closest...

What can go wrong?
A* Search

Hart, Nilsson & Rafael 1968

Best first search with $f(n) = g(n) + h(n)$

- $g(n) = \text{sum of costs from start to } n$
- $h(n) = \text{estimate of lowest cost path } n \rightarrow \text{goal}$
  
  $h(\text{goal}) = 0$

Can view as cross-breed:

- $g(n) \sim \text{uniform cost search}$
- $h(n) \sim \text{greedy search}$

Best of both worlds…
A* Search

Hart, Nilsson & Rafael 1968

Best first search with $f(n) = g(n) + h(n)$

- $g(n) =$ sum of costs from start to $n$
- $h(n) =$ estimate of lowest cost path $n \rightarrow$ goal
  $h(\text{goal}) = 0$

If $h(n)$ is admissible and monotonic, then A* is optimal

Underestimates ($\leq$) cost of reaching goal from node

$f$ values never decrease From node to descendants (triangle inequality)
Is Manhattan distance admissible?

- Underestimate?
Is Manhattan distance **monotonic**?

- f values increase from node to children
- (triangle inequality)
Monotonicity (aka Consistency)

Defn monotonic:
\[ F(a) \geq F(b) \]
\[ G(a) + H(a) \geq G(b) + H(b) \]
\[ \geq G(a) + ab + H(b) \]
\[ H(a) \geq ab + H(b) \]
Admissible Heuristics

Slide credit: Travis Mandel
Monotonic/Consistent Heuristic

- Monotonic Heuristic:
  - True (optimal) cost remaining
  - $h(x)$ Heuristic-estimated cost remaining

- Not Monotonic (but admissible)

Slide credit: Travis Mandel
Monotonic/Consistent Heuristics

Monotonic

Not Monotonic (but admissible)

True (optimal) cost remaining

h(x) Heuristic-estimated cost remaining

f(x) Heuristic + cost so far
Optimality of A* (tree search)

Suppose some suboptimal goal $G_2$ has been generated and is in the queue. Let $n$ be an unexpanded node on a shortest path to an optimal goal $G_1$.

\[ f(G_2) = g(G_2) \quad \text{since } h(G_2) = 0 \]
\[ > g(G_1) \quad \text{since } G_2 \text{ is suboptimal} \]
\[ \geq f(n) \quad \text{since } h \text{ is admissible} \]

Since $f(G_2) > f(n)$, A* will never select $G_2$ for expansion.

Monotonicity required for proof in graph search version
Monotonicity Required to Ensure A* Graph Search is Optimal

Monotonicity needed to ensure optimality
Given GS optimization of queue
Suppose node(a>b>d) has been expanded but not node(a>c>d)
  It won’t be, because it’s state (d) is closed
Optimality of A*

- **Lemma 1**
  If \( h(n) \) is monotonic, then the values of \( f \) along any path are non-decreasing \(~ by defn.\)

- **Lemma 2**
  Whenever A* selects node \( n \) for expansion, the optimal path to that node has been found

  Assume not. Then \( \exists \) node \( m \) on frontier which is on a better path to \( n \), but by lemma 1, it would have been explored.

- **Lemma 3**
  A* expands nodes in order of increasing \( f \) value
Optimality Continued

A* gradually adds “f-contours” of nodes.
Contour i has all nodes with \( f = f_i \), where \( f_i < f_{i+1} \)
First goal expanded must have lowest f-value
\( \rightarrow \) Lowest cost, since \( h(\text{goal}) = 0 \)
A* Example

Arad
366 = 0 + 366
A* Example

- **Sibiu**: 393 = 140 + 253
- **Timisoara**: 447 = 118 + 329
- **Zerind**: 449 = 75 + 374
A* Example

- Arad
  - Fagaras: 415 = 239 + 176
  - Oradea: 671 = 291 + 380
  - Rimnicu Vilcea: 413 = 220 + 193
- Sibiu
- Timisoara: 447 = 118 + 329
- Zerind: 449 = 75 + 374
A* Example

Arad
646=280+366

Fagaras
415=239+176

Oradea
671=291+380

Sibiu

Rimnicu Vilea

Craiova
526=366+160

Pitesti
417=317+100

Sibiu
553=300+253

Timisoara
447=118+329

Zerind
449=75+374
A* Example
A* Example
European Example

Straight-line distance to Bucharest

- Arad: 366
- Bucharest: 0
- Craiova: 160
- Dobrota: 242
- Eforie: 161
- Fagaras: 178
- Giurgiu: 77
- Hirsova: 151
- Iasi: 226
- Lugoj: 244
- Mehadia: 241
- Neamt: 234
- Oradea: 380
- Pitești: 98
- Rimnicu Vilcea: 193
- Sibiu: 253
- Timisoara: 329
- Urziceni: 80
- Vaslui: 199
- Zerind: 374
A* Summary

- **Pros**
  - Produces optimal cost solution!
  - Does so quite quickly (focused)

- **Cons**
  - Maintains priority queue…
  - Which can get exponentially big 😞
Iterative-Deepening A*

- Like iterative-deepening depth-first, but...
- Depth bound modified to be an f-limit
  - Start with f-limit = h(start)
  - Perform depth-first search (using stack, no queue)
    - Prune any node if f(node) > f-limit
    - Next f-limit = min-cost of any node pruned
IDA* Analysis

- Complete & Optimal (ala A*)
- Space usage $\propto$ depth of solution
- Each iteration is DFS - no priority queue!
- # nodes expanded relative to A* ??
  - Depends on # unique values of heuristic function
  - In 8 puzzle: few values $\Rightarrow$ close to # A* expands
  - In traveling salesman: each f value is unique
    $\Rightarrow 1+2+\ldots+n = O(n^2)$ where $n=nodes$ A* expands
    if $n$ is too big for main memory, $n^2$ is too long to wait!
Forgetfulness

- A* used exponential memory
- How much does IDA* use?
  - During a run?
  - In between runs?
    - SMA*
Which Algorithm?

- Uniform cost search (UCS):
Which Algorithm?

- A*, Manhattan Heuristic:
Which Algorithm?

- Best First / Greedy, Manhattan Heuristic:
Demo

http://qiao.github.io/PathFinding.js/visual/

SUGGESTED BY Fernando Centurion