322
Jan 19

HEURISTIC SEARCH

Announcements:

- Assignment 1, in Connect, due Mon, Feb 2, 2pm
- Assignment 0 will be returned by Wed

Recap:

- IDS:
  - Re-compute elements of the frontier rather than saving them
  - Restart search at each level
  - Get the best of both worlds (DFS and BFS):

	Complete	Optimal	Time	Space
DFS	NO	NO	$O(b^m)$	O(bm)
BFS	YES	YES	$O(b^m)$	$O(b^m)$
IDS	YES	YES	$O(b^m)$	O(bm)

- Introduce the idea of costs (arcs): The cost of a path is the sum of the costs of its arcs
- Optimal criteria is now defined in terms of cost.
  - The best solution has the lowest path cost

Lowest-Cost-First Search (LCFS):

- Lowest-cost-first search finds the path with the lowest cost to a goal node
  - At each stage, it selects the path with the lowest cost on the frontier
  - The frontier is implemented as a priority queue ordered by path cost

Search Heuristic:

- A search heuristic h(n) is an estimate of the cost of the optimal (cheapest) path from node n to a goal node
- Arrows: we do not know about them, they are estimations about how far is the node from the goal
- Example - finding routes:
  - Possible h(n) is the straight-linedistance between source and goal node
  - Goal node: Bucharest
  - Straight distance is an estimation available if you have the coords

Best First Search (BestFS):

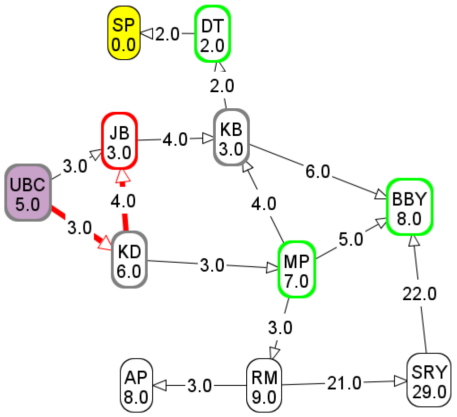
- Always choose the path on the frontier with the smallest h value
- Treats the frontier as a priority queue ordered by h
- Greedy approach: uses locally information that is estimated, i.e, chose the solution that is locally the best.
- AIspace:
    - Vancouver Neighborhood Graph
    - 5 expansions
    - Lead by values that are estimations of distance

- Not guaranteed to be completed
    - Misleading heuristics demo in AIspace
- No Optimal
    - Aispace, same example
    - Best of LCF is 42.8
    - Finds a solution but it is not optimal
    - Rely on statistics that are good at the moment but don't take in consideration the expanded view
- Time Complexity: $O(b^m)$
    - Worst case: depending on the heuristic can still have to explore all nodes
- Space Complexity: $O(b^m)$
- Why to use?
    - Faster for good heuristics
    - If you know you can give very good estimations
    - LCSF uses true information/values, but has to be more exhaustive so can take lot of time and space
- Lowest cost(p) + h(p): consider estimation and actual path

A* Search:

- Takes into account both:
    - the cost of the path to a node c(p)
    - the heuristic value of that path h(p).
- Let f(p) = c(p) + h(p).
    - f(p) is an estimate of the cost of a path from the start to a goal via p.
- Always chooses the path on the frontier with the lowest estimated distance from the start to a goal node constrained to go via that path.

- F-value of ubc -> kd -> jb is 10 (sum) :



- Complete
    - finds a solution, if one exists
- Optimal
    - finds the optimal path to a goal) if:
        - the branching factor is finite
        - arc costs are > 0
        - h(n) is admissible (explored in the next class while other 2 after it)

Admissibility of a heuristic

- Let c(n) denote the cost of the optimal path from node n to any goal node. A search heuristic h(n) is called admissible if h(n) ≤ c(n) for all nodes n, i.e. if for all nodes it is an underestimate of the cost to any goal
- The straight line distance is admissible
    - The shortest distance between two points is a line.
- Possible h(n) = sld(n, Bucharest) + cost(Bucharest, Urzineni) is admissible?
    - Cost of going from Vastul to Urzineni is shorter than this estimate
    - Counter example: Vastul is shorter than estimated
    - As long as I find a counter example is not admissible
- Example 2: grid world
    - It is admissible because ignores obstacles
    - Manhattan distance is the shortest path between any two tiles of the grid given the actions available and no walls. Including the walls will force the agent to take some extra steps to avoid them
- Example 3: Eight Puzzle
    - An admissible heuristics for the 8-puzzle is: Number of misplaced tiles
    - One needs at least that many moves to get the board in the goal state

How to Construct an Admissible Heuristic

- Identify relaxed version of the problem:
    - Where one or more constraints have been dropped
    - Problem with fewer restrictions on the actions
- You should identify constraints which, when dropped, make the problem easy to solve
- Robot: Optimal solution to this relaxed problem is Manhattan distance
    - Allowing the agent to move through walls
- Driver: Optimal solution to this relaxed problem is straight-line distance
    - Allowing the agent to move straight (instead of roads)
- 8puzzle: Optimal solution to this relaxed problem is number of misplaced tiles
    - Alowwing tiles to move anywhere

Learning goals:

- Apply basic properties of search algorithms
- Select the most appropriate search algorithms for specific problems.
    - Depth-First Search vs. Bredth-First Search vs. Iterative Deepening vs. Least-Cost-First Search, Best First Search, A*
- Define/read/write/trace/debug different search algorithms
- Construct heuristic functions and discuss their admissibility for specific search problems

TO DO:

- Reading: A*: Ch 6.1
- Practice Excercises 3C and 3D
- Start working on Assignment 1