Search: Advanced Topics

Jim Little

UBC CS 322 - Search 5
September 22, 2014

Textbook § 3.6
Lecture Overview

• Recap A*
• Branch & Bound
• A* tricks
• Other Pruning
A* advantages
Branch-and-Bound Search

• Biggest advantages of A*….

  Uses heuristics and is optimal

• What is the biggest problem with A*?

  Space

• Possible, preliminary Solution:

  DFS + heuristics
Branch-and-Bound Search Algorithm

• Follow exactly the same search path as depth-first search
  • treat the frontier as a stack: expand the most-recently added path first
• the order in which neighbors are expanded can be governed by some arbitrary node-ordering heuristic
Once this strategy has found a solution....
Branch-and-Bound Search Algorithm

- Keep track of a lower bound and upper bound on solution cost at each path
  - lower bound: $LB(p) = f(p) = cost(p) + h(p)$
  - upper bound: $UB =$ cost of the best solution found so far.
    - ✓ if no solution has been found yet, set the upper bound to $\infty$.
- When a path $p$ is selected for expansion:
  - if $LB(p) \geq UB$, remove $p$ from frontier without expanding it (pruning)
  - else expand $p$, adding all of its neighbors to the frontier
Branch-and-Bound Analysis

- Complete? 
  - yes
  - no
  - It depends

- Optimal? 
  - yes
  - no
  - It depends

- Space complexity? 
  - $O(b^m)$
  - $O(m^b)$
  - $O(bm)$
  - $O(b+m)$

- Time complexity?
Branch-and-Bound Analysis

- **Completeness**: no, for the same reasons that DFS isn't complete
  - however, for many problems of interest there are no infinite paths and no cycles
  - hence, for many problems B&B is complete
- **Time complexity**: $O(b^m)$
- **Space complexity**: ……
  - Branch & Bound has the same space complexity as DFS
  - this is a big improvement over A*!
- **Optimality**: ……..
Lecture Overview

• Recap A*
• Branch & Bound
• A* tricks
• Pruning Cycles and Repeated States
Other $A^*$ Enhancements

The main problem with $A^*$ is that it uses exponential space. Branch and bound was one way around this problem. Are there others?

- Iterative Deeping $A^* = IDA^*$
- Memory-bounded $A^*$
(Heuristic) Iterative Deepening – IDA*

B & B can still get stuck in infinite (extremely long) paths

• Search depth-first, but to a fixed depth
  • if you don't find a solution, increase the depth tolerance and try again
  • depth is measured in $f$

    at start node $f(\text{start}) = h(\text{start})$

• Then update with the lowest $f$ that passed the previous bound
Analysis of Iterative Deepening A* (IDA*)

- Complete and optimal: yes, no, It depends

- Space complexity:
  - \(O(b^m)\)
  - \(O(m^b)\)
  - \(O(bm)\)
  - \(O(b+m)\)

- Time complexity:
  - \(O(b^m)\)
  - \(O(m^b)\)
  - \(O(bm)\)
  - \(O(b+m)\)
(Heuristic) Iterative Deepening – IDA*

- Counter-intuitively, the asymptotic complexity is not changed, even though we visit paths multiple times (*go back to slides on uninformed ID*)

Just a factor of $b/(b-1)^2$
Memory-bounded $A^*$

- Iterative deepening $A^*$ and B & B use a tiny amount of memory
- what if we've got more memory to use?
- keep as much of the fringe in memory as we can
- if we have to delete something:
  - delete the worst paths with highest $f$
  - ``back them up'' to a common ancestor
MBA*: Compute New $h(p)$
Memory-bounded $A^*$

- Iterative deepening $A^*$ and B & B use a tiny amount of memory
- what if we've got more memory to use?
- keep as much of the fringe in memory as we can
- if we have to delete something:
  - delete the worst paths (with highest $f$)
  - ``back them up'' to a common ancestor
Lecture Overview

- Recap A*
- Branch & Bound
- A* tricks
- Pruning Cycles and Repeated States
You can prune a path that ends in a node already on the path. This pruning cannot remove an optimal solution.

- The time is linear in path length.
Repeated States / Multiple Paths

Failure to detect repeated states can turn a linear problem into an exponential one!
Multiple-Path Pruning

- You can prune a path to node $n$ that you have already found a path to
- (if the new path is longer – more costly).
Multiple-Path Pruning & Optimal Solutions

Problem: what if a subsequent path to $n$ is shorter than the first path to $n$?
• You can remove all paths from the frontier that use the longer path. (as these can’t be optimal)
Multiple-Path Pruning & Optimal Solutions

Problem: what if a subsequent path to $n$ is shorter than the first path to $n$?

- You can change the initial segment of the paths on the frontier to use the shorter path.
Learning Goals for today’s class

• Define/read/write/trace/debug different search algorithms
  • With / Without cost
  • Informed / Uninformed

• Pruning cycles and Repeated States
Next class: Wed

- Dynamic Programming
- Recap Search
- Start Constraint Satisfaction Problems (CSP)
- Chapter 4.

- Start working on assignment 1!