EEL5840: Elements of Machine Intelligence

Announcements

• Reading Assignment:
  > Nilsson chapters 9, 10
• Announcements:
  > Tentative 2nd Exam Date:
    – 12/03/15 (Thursday)
  > LISP Project due 12/01/15
• Today’s Handouts in WWW:
  > Outline Class 23
• Web Site
  > www.mil.ufl.edu/eel5840
  > Software and Notes

EEL5840: Elements of Machine Intelligence

Today’s Menu

• Finish up Heuristic Search (Chapter 9)
  ⇒IDA, IDA* & RBFS
  ⇒Search Efficiency
Heuristic Searches

Related Algorithms

• Bi-Directional Search
  > Breadth-First may expand less nodes bi-directionally than uni-directionally as shown in the Figure.
  > However, if the $h(n)$ used by the bidirectional process are slightly inaccurate, the search frontiers may not intersect.

• Staged Search: Prune the tree at some stages of computation in order to free up storage. At the end of each stage you keep only the most promising nodes (those with good $f$ values). Obviously, staged search carries no guarantee of optimality

• Limitation of Successors - to keep $\Gamma(n)$ small we also keep only the “best” nodes
  > We will need additional problem domain knowledge (i.e., a DEADEND)
  > We either modify $A^*$ or assign to some nodes very high $h(n)$ values
  > Consider evaluating $h(\Gamma(n))$ before evaluating the DBs themselves and only select those nodes with high promise
Heuristic Search

- **Iterative Deepening**
  - Enjoys the same linear memory requirements of DFS while guaranteeing that a goal node of minimal depth will be found - memory grows linearly with the depth of the goal.
  - Successive depth-first searches are conducted - each with a depth bound increasing by 1 until a goal node is found.

How many nodes are expanded by BFS?
Assuming uniform branching factor $b$, with a goal at depth $d$

$$1 + b + b^2 + \ldots + b^d = \frac{(b^{d+1} - 1)}{(b - 1)}$$

How many nodes are expanded by iterative deepening?
Down to level $j$ we have $N_{idj} = \frac{(b^{j+1} - 1)}{(b - 1)}$

At worst, we must conduct $d$ such searches for the goal at depth $d$

$$N_{id} = \sum_{j=0}^{d} \frac{(b^{j+1} - 1)}{(b - 1)} = \frac{(b^{d+2} - 2b^d + d + 1)}{(b - 1)^2}$$

For large $d$ this reduces to

$$N_{id}/N_{bf} \approx \frac{b}{(b - 1)}$$

For $b=10$ and large $d$ this is about 1.11% or ID expands about 11% more nodes than BFS.
Heuristic Search

- **IDA** (Korf, 1985)
  - Allows us to find minimal cost paths with memory that grows linearly with the depth of the goal.
  - Execute a series of depth-first searches. In the first search we establish a cost cutoff equal to \( f(n_0) = g(n_0) + h(n_0) \) with \( n_0 = s \).
  - Expand nodes in a DFS fashion—backtrack whenever the \( f \) value of a successor of an expanded node exceeds the cutoff value.
  - If this terminates at a goal node, then you have a minimal path, else the cost of an optimal path must exceed the cutoff value.
  - Use as your next cutoff value the value of a node visited but not expanded.
  - IDA* does have to repeat node expansions, but there are potential tradeoffs involving reduced memory requirements and ease of implementation.

Performance Issues

- **Penetrance**
  - The penetrance \( P \) of a search algorithm measures the extent to which the search focuses toward a goal and does not wander off.
  - \( P = \frac{L}{T} \) where \( L \) = length of a found path to the goal
  - \( T \) = total number of nodes (excluding \( s \))
  - \( P_{\text{max}} = 1 \)
  - \( P \) is a function (difficulty, efficiency of search)
  - \( T \) grows faster than \( L \) and thusly \( P \) is usually high for small \( L \) and small for large \( L \)
  - \( P \) measures when a tree is elongated vs bushy

- **Branching Factor**
  - The branching factor \( B \) is based on a tree having
  - depth equal to path length
  - a total number of nodes = to the nodes generated during search
  - \( B \) measures the constant number of successors of each node in such a tree
EEL5840: Elements of Machine Intelligence

Performance Issues

Branching Factor

\[ B = B^2 + B^3 + \ldots + B^L = N \]

L = total path length
N = total number of nodes

\[ \frac{[B^{L-1}]B}{B - 1} = N \]

if \( B = 1 \) we have a focused search

For the problem \( f(n) = d(n) + W(n) \) the \( B \) value is \( B \approx 1.3 \) For a depth of 18, \( N \approx 500 \) nodes and for a depth of 20, \( N \approx 1,000 \! \)