Lecture 5 CS6800 Artificial Intelligence:

- Breadth First Search
- Heuristic Graph Search
- Algorithm A
- Algorithm A*
- Admissibility of A*

Heuristic Search Procedures

Characteristics of uninformed search methods:

- Exhaustive
- Expands many nodes

Heuristic information may be used to order the exploration of the search graph through the use of an ***evaluation function*** that gives the measure of the "promise" of a node.

The choice of this function critically determines the search results. A function that underestimates the promise of a node may not find a minimum cost path, while one that overestimates the promise may expand too many nodes.

Algorithm A

We will now consider different evaluation functions and develop theoretical results about the performance of GRAPH-SEARCH under these functions.

Let's define the evaluation function f so that its value, $f(n)$, at any node n **estimates** the sum of the cost of the minimal cost path from the start node s to node n plus the cost of the minimal cost path from node n to a goal node.

$f(n)$ estimates the minimum cost of a path from the start node to a goal node, with the path constrained to go through node n.

The Evaluation Function $f(n)$

We will order the nodes on *OPEN* in such a way that the node with the least value of f is at the head, and therefore expanded next.

We define $h^*(n)$ to be the ***actual*** cost of a minimal cost path from node n to ***any*** of the goal nodes.

$g^*(n)$ is the ***actual*** cost of a minimal cost path from the start node to node n.

Then $f^*(n)$ may be defined as the ***actual*** cost of a minimal cost path from the start node to a goal node through node n.

The Evaluation Function $f(n)$

Clearly,

$$f^*(n) = g^*(n) + h^*(n)$$

Note that when $n = s$, then $g^*(s) = 0$, and $f^*(s) = h^*(s)$.

We desire f to be an estimate of f^*. We can express this estimate as:

$$f(n) = g(n) + h(n)$$

The Evaluation Function cont.

In the GRAPHSEARCH procedure, we will use for $g(n)$ the cost of the path from s to n in the ***search tree***.

Note that at a given instant, $g(n)$, may not equal $g^*(n)$. Why? However, the following relation holds:

$$g(n) \geq g^*(n)$$

For $h(n)$ we will use heuristic information about the problem domain. We call h, our estimate of $h^*(n)$, the ***heuristic function***.

Algorithm A

A GRAPHSEARCH procedure using an evaluation function *f(n)* as defined above, for ordering the nodes on *OPEN* is called ***Algorithm A***.

If we use an $h(n)$ that is a lower bound on $h^*(n)$, then the GRAPHSEARCH procedure is called ***Algorithm A****.

A function $h(n)$ is a lower bound on $h^*(n)$ if:

$$h(n) \leq h^*(n) \ \forall \text{ nodes } n.$$

Examples of $h(n)$

For example, $h \equiv 0$ is definitely a lower bound on $h^*(n)$. If $h(n) = 0$, and $g(n) = d(n)$, the depth of node n, we will obtain the breadth-first version of the algorithm.

Therefore, the breadth-first GRAPHSEARCH procedure is a special case of Algorithm A^*.

Can you think of a definition of f that would produce a depth-first search?

Admissibility

We will now show that procedures of type A^* are guaranteed to find a minimal cost path to a goal node, assuming a path exists.

An algorithm is ***admissible*** if, for any graph, it always terminates in an optimal path *s* to a goal node whenever such a path exists.

We must first show that an algorithm terminates whenever a goal node is accessible from the start node.

<u>RESULT 1</u>: GRAPHSEARCH always terminates for finite graphs.

Termination for infinite graphs

If a path to a goal node exists, we will now show that A^* will terminate even in infinite graphs.

Termination may be prevented only if new nodes are forever added to *OPEN*. If this were the case, however, the f values of the nodes would grow in an unbounded manner.

Infinite Graph Example

Another Example

We can express what we have learned from the above graphs more formally as follows. Let $d^*(n)$ be the ***length*** of the shortest path in the implicit graph being searched from s to any node n in the search tree produced by A*.

We now make use of the assumption that the cost of every arc is greater than some small number e. In particular:

$g^*(n) \geq d^*(n)e,$

$g(n) \geq g^*(n),$ therefore

$g(n) \geq d^*(n)e,$

Also, we know $f(n) \geq g(n)$ (Why?)

Therefore, $f(n) \geq d^*(n)e$

What does this say about nodes?

For each node n on *OPEN*, the corresponding value of f is at least as large as $d^*(n)e$. For an infinite subgraph, the associated f values will become arbitrarily large.

Now we must show that there are nodes whose costs do not become arbitrarily large. We must show that there are always nodes, leading to a solution, that have ***bounded*** costs.

More formally, before the termination of A^*, there is always a node n on *OPEN* which belongs to the solution path, and for which $f(n) \leq f^*(s)$.

Let the ordered sequence $(s = n_0, n_1, \ldots, n_k)$, where n_k is the goal node, be an optimal path from s to a goal node.

Let n' be the first node from the sequence that is on *OPEN*. One such node must be on *OPEN*. Why?

For this node n' we have:

$$f(n') = g(n') + h(n')$$

Now we also know that:

$$g(n') = g^*(n')$$

Why?

Therefore,

$$f(n') = g^*(n') + h(n')$$

Now since we are assuming the restriction on h, that for all n, $h(n) \le h^*(n)$,

$$f(n') \le g^*(n') + h^*(n')$$

But this implies:

$$f(n') \leq f^*(n')$$

But we know n' is on the solution path, so:

$$f^*(n') = f^*(s)$$

Thus we have derived:

$$f(n') \leq f^*(s)$$

Why does this matter? How can we make use of this result?

Final result about termination on infinite graphs

The unboundedness of the f values for nodes corresponding to the infinite subgraph and the boundedness of the f values for nodes that lie on the solution path proves that A^* will terminate.

Along the way we have established the following two important results:

RESULT 2: At any time before A^* terminates, there exists in *OPEN* a node n' that is on an optimum path from s to a goal node, with:

$$f(n') \leq f^*(s)$$

RESULT 3: If there is a path from s to a goal node, then A^* terminates.

In particular we have shown that if there is such a path that A^* will terminate by finding that path! What we still need to show is the path found by A^* is an ***optimal path***.

Optimality of Path by A^*

Suppose that A^* were to terminate at some goal node t without finding an optimal path, that is:

$$f(t) = g(t) > f^*(s)$$

By RESULT 2, there existed just **before** termination a node n' on *OPEN*, and on an optimal path with:

$$f(n') \leq f^*(s) < f(t)$$

Why is this a contradiction?

Optimality of Path by A^*

Finally we get:

RESULT 4: Algorithm A^* is admissible.

That is, if there is a path from s to a goal node, A^* terminates by finding an optimal path.

