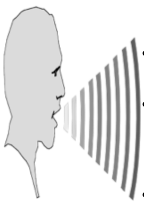
EEL5840: Elements of Machine Intelligence

Robobug

Announcements

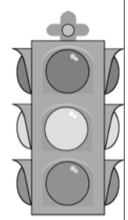


- Reading Assignment:
  - > Nilsson chapter 9
- Announcements:
  - > Tentative 2nd Exam Dates:
    - – 12/1/09 (Tuesday)
    - – 12/3/09 (Thursday)
- Today's Handouts in WWW:
  - > Outline Class 21
- Web Site
  - > www.mil.ufl.edu/eel5840
  - > Software and Notes

EEL5840: Elements of Machine Intelligence

Robobug

Today's Menu



- Heuristic Search (Chapter 9)
  - ⇒Algorithm A*
  - ⇒Admissibility of A*

EEL5840: Elements of Machine Intelligence

Robobug

Search Strategies

PROCEDURE GRAPH-SEARCH

1. Create a *search graph, G,* consisting solely of the start node, *s. Put s* on a list called *OPEN.*
2. Create a list called *CLOSED* that is initially empty.
3. LOOP: if *OPEN* is empty, exit with failure.
4. Select the first node on *OPEN*, remove it from *OPEN*, and put it on *CLOSED.* Call this node n.
5. If n is a goal node, exit successfully with the solution obtained by tracing a path along the pointers from *n to s in* G. (see step 7.)
6. Expand node n, generating the set, *M*, of its successors and install them as successors of n in *G*.

EEL5840: Elements of Machine Intelligence

Robobug

Search Strategies

7. Establish a pointer to n from those members of M that were not already in *G* (i.e., not already on either *OPEN or CLOSED).* Add these members of *M* to *OPEN*. For each member of M that was already on *OPEN or CLOSED*, decide whether or not to redirect its pointer to n. For each member of M already on *CLOSED*, decide for each of its descendants in G whether or not to redirect its pointer.
8. Reorder the list *OPEN*, either according to some arbitrary scheme or according to heuristic merit.
9. GO LOOP

EEL5840: Elements of Machine Intelligence

Robobug

Algorithm A

ALGORITHM A

Let $f(n) = g(n) + h(n)$ in step 8 of GRAPH-SEARCH where

$g(n)$: estimate of the cost of a minimum length path $s \rightarrow n$

$h(n)$: estimate of the cost of a minimum length path $n \rightarrow t$

also, step 7 guarantees that *$g(n)$ can never increase.*

{See slides 6-7 in class 21}

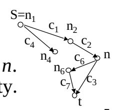
In the example:

$c(n_1,n_2) = \text{cost from } n_1 \rightarrow n_2$

$g(n) = c(n_1,n) = c(n_1,n_2) + c(n_2,n) = c_1 + c_2$

$h^*(n) = c(n,t) = c_3$ (the actual but unknown cost from $n \rightarrow t$)

$h(n)$ is an estimate of $h^*(n)$. When we are at node n we have not finished the problem and we do not yet have the real $c(n,t)$. We say that $f(n)$ is the cost of a minimal cost path constrained through node n. Here, $c_6 < c_3$ and $c_3 \leq c_6 + c_7$ by the triangle inequality.



EEL5840: Elements of Machine Intelligence

Robobug

Algorithm A

DEFINITIONS

- $f(n)$: estimate of the cost of a path $s \rightarrow t$ through node n
  $c(s,t) = c(s,n) + c(n,t)$
- $k(n_i,n_j)$: actual cost of a minimal-cost path $n_i \rightarrow n_j$
- $h^*(n)$: cost of a minimal cost path from node n to a goal, i.e., $\min(k(n,t_1), k(n,t_2), \ldots, k(n,t_i))$ and any node that achieves $h^*(n)$ is a node in the optimal path.
- $g^*(n)$: cost of an optimal path from $s \rightarrow n$ (the shortest path $s \rightarrow n$)
  $= k(s,n)\ \forall n$ accessible from s
- $f^*(n)$: (actual cost of an optimal path $s \rightarrow n$) + (cost of an optimal path $n \rightarrow t$) {the path $s \rightarrow t$ is constrained to go through n}
- $f(n) = g(n) + h(n)\ \forall n$ and $f^*(n) = g^*(n) + h^*(n)\ \forall n$

  Now when $n=s$;
- $f^*(s) = g^*(s) + h^*(s) = h^*(s)$ since by definition $g^*(s)=0=k(s,s)$
  $h^*(s)$: actual cost of an unconstrained path $s \rightarrow \{t_i\}$

EEL5840: Elements of Machine Intelligence

Robobug

Algorithm A

Let $g(n)$ be an estimate of $g^*(n)$ and $h(n)$ be an estimate of $h^*(n)$. Then $f(n)$ will be an estimate of $f^*(n)$ and further let

$E\{f(n) - f^*(n)\} = 0$ and

Variance$\{f(n)\}$ be non-increasing

We say that $f(n)$ is a consistent estimate of $f^*(n)$ and we expect algorithm A to yield good results.

Q: Are there any properties of f, g, or h that insure optimal results?

We also note :

if $h(n)=0$ and $g(n)=depth(n)$ algorithm A yields BFS

and if $g(n)=0$ algorithm A yields pure heuristic search (DFS)

$f(n) = \alpha g(n) + \beta h(n) = \alpha\{\text{breadth component}\} + \beta\{\text{depth component}\}$

and $\{\alpha, \beta\}=1$ in Algorithm A but it can be more general

EEL5840: Elements of Machine Intelligence

Robobug

Algorithm A*

ALGORITHM A*

Let $f(n) = g(n) + h(n)$ in step 8 of GRAPH-SEARCH where

$g(n)$: estimate of the cost of a minimum length path $s \rightarrow n$

$h(n)$: estimate of the cost of a minimum length path $n \rightarrow t$

and $0 \leq h(n) \leq h^*(n)$ and step 7 guarantees that $g^*(n) \leq g(n)$

- Algorithm A* always finds the optimal path from $s \rightarrow t$
- Notice that $h(n) \leq h^*(n)$ means that $h(n)$ underestimates the actual optimal cost $h^*(n)$ and thus, $h(n)$ is a *conservative* estimate of $h^*(n)$
- Since $h(n)=0$ is an underestimate of $h^*(n)$ then A* with $g(n)=depth(n)$ always finds the optimal path $s \rightarrow t$. That is, BFS, is optimal.
- On the other hand $g(n) = 0$, that is heuristic search (DFS), does not carry the same guarantee as we will see later.

Robobug

EEL5840: Elements of Machine Intelligence

The End!