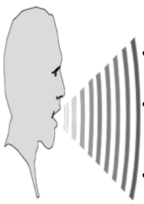
EEL5840: Elements of Machine Intelligence

Robobug

Announcements

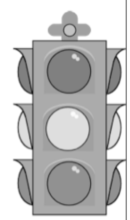


- Reading Assignment:
  > Nilsson chapters 8 & 9
- Announcements:
  > First Exam Date
    Thu. Oct. 8th in class
- Today's Handouts in WWW:
  > Outline Class 18
- Web Site
  > www.mil.ufl.edu/eel5840
  > Software and Notes
  > XLISP Documentation

EEL5840: Elements of Machine Intelligence

Robobug

Today's Menu



- Agents That Plan (Nilsson Chapter 7)
  > Searching Explicit State-Space Graphs
  > Feature-Based State Spaces
  > Graph Notation
- Uninformed Search (Chapter 8)
  ⇒Formulating the State Space
  ⇒Components of Implicit State Graphs

EEL5840: Elements of Machine Intelligence

Robobug

Agents That Plan

- State-Space Graphs
  > To find a set of actions that will achieve a specified goal, the robot needs simply to find a path in the graph from a node representing its initial world state (we will call this the *start node*) to a node representing the specified goal state (likewise called the *goal node*). The action that will achieve that goal can be read out as the labels on the arcs in this path.
  > It is easy (for us!) to find paths by visual inspection; however, computational agents will need to use various graph-searching processes in order to find such paths. For example to restack blocks ((A B C)) into ((C B A)) we would execute: {*move(A,floor), move(B,A),move(C,B)*}
  > The operators labeling the arcs along a path to a goal can be assembled into a sequence called a *plan*, and searching for that sequence in the graph is called *planning*.

EEL5840: Elements of Machine Intelligence

Robobug

Agents That Plan

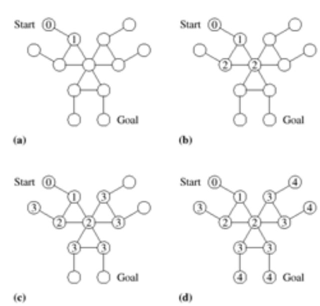


Figure 7.3 Stages of Search

Robobug

EEL5840: Elements of Machine Intelligence

Agents That Plan

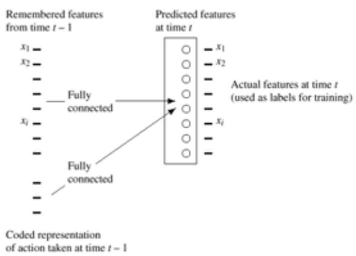
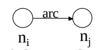


Figure 7.4 Predicting a Feature Vector with an ANN

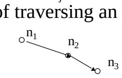
Robobug

EEL5840: Elements of Machine Intelligence

Graph Notation Review



- Graph: a finite set of nodes joined by {un}directed arcs
  - n_j is a successor (son) of n_i
  - n_i is a parent of n_j
- Tree: A kind of graph where each node has at most 1 parent
  - root node: a node with no parents (usually drawn near the top)
  - tip node: a node having no successors (usually near the bottom)
  - depth of a node: 1 + depth (parent) with depth(root) ≡ 0
- Path of length k ≡ $(n_{i1}, n_{i2}, \ldots, n_{ik})$ where each n_{ij} is a successor of n_{ij-1} j=2, 3, …, k
- Accessible: If there is a path from $n_i \rightarrow n_j$ then n_j is accessible from n_i
  - If accessible(n_j,n_i) then n_j is a descendant of n_i and n_i is an ancestor of n_j
- Cost of traversing an arc from n_i to n_j is given by $c(n_i,n_j) \geq e > 0$



$c(n_1,n_3) = c(n_1,n_2) + c(n_2,n_3)$

Robobug

EEL5840: Elements of Machine Intelligence

Graph Notation Review

- Start Node s: the initial DB in search problem or root node
- Goal Node t: the final DB in a search problem, usually a tip node
- Explicit Graph: When all the nodes in a search problem are specified by a table or a data structure
- Implicit Graph: The rules and the control strategy together specify a mathematical operation that allows us to obtain a node from a given node. That is $n_i = \Gamma(n_j)$. Γ is called the successor operator. $\Gamma(n_j)$ may yield multiple successors denoted by $\{n_i,n_{i+1}\}$
- Expanding a node: Apply Γ to any node.
- A Graph Search Control Strategy: Makes the implicit graph represented by Γ explicit (represented by a graph).
- The following algorithm, Graph_Search, is a reasonable algorithm

Robobug

EEL5840: Elements of Machine Intelligence

Graph and Tree Notation

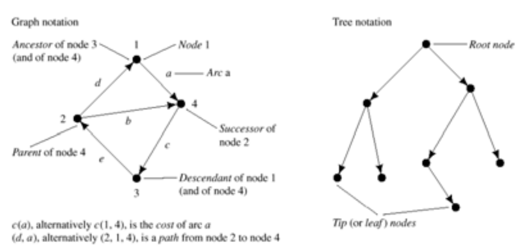


Figure 7.5 Graph and Tree Notation

Robobug

EEL5840: Elements of Machine Intelligence

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.*

Robobug

EEL5840: Elements of Machine Intelligence

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

Robobug

EEL5840: Elements of Machine Intelligence

Search Strategies

- The Representation Problem: dealing w/ large state space
  - Careful formulation is required - choose you representation, iconic model or data structure by taking inventory of the problem domain and any other problem-specific knowledge
  - Methods are required to represent large graphs implicitly
  - Efficient search methods are required

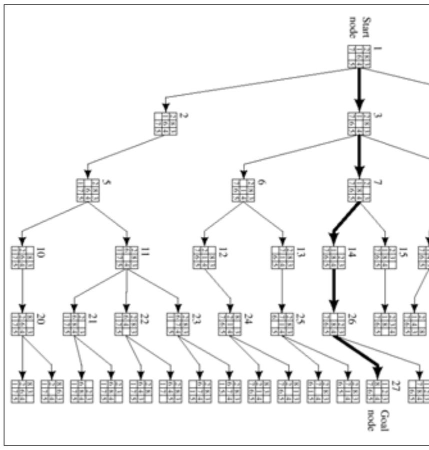
2	8	3
1	6	4
7		5

→

1	2	3
8		4
7	6	5

Eight Puzzle State Space
O(9!)=362,880 nodes

In this problem the obvious(?) iconic representation is a 3x3 array, with the blank tile moving {up, down, left, right}



Robobug

EEL5840: Elements of Machine Intelligence

The End!