Nets and Basic Search
—
Winston, Chapter 4

Michael Eisenberg and Gerhard Fischer
TA: Ann Eisenberg

AI Course, Fall 1997

Search — Finding Paths through Nets

examples of "blind" searches:

- depth-first search
- breath-first search
- nondeterministic search

searches guided by heuristic quality estimates:

- hill-climing
- beam search
- best-first search

questions about search methods:

- Is search the best way to solve the problem?
- Which search methods solve the problem?
- Which search method is most efficient for this problem?

Main Streets and Side Streets

- going from “start” to “goal” only once or very seldom ---> stay on main streets

- going from “start” to “goal” very often ---> explore side streets (find a better path with more work)

- example: learning high-functionality applications
  - physical efforts versus cognitive efficiency
  - forgetting seldom used operations
  - replace, replace all, query replace, temporary suspension of query replace

Search Trees

A search tree is a representation that is a semantic tree in which

- nodes denote paths
- branches connect paths to one-step path extensions

with writers that

- connect a path to a path description

with readers that

- produce a path's description

Search Trees — Concepts and Properties

a path: each child denotes a path that is a one-step extension of the path denoted by its parent

branching factor: the number of children which a node has

expanding a node: determining the children of a node

open node: until they are expanded

closed node: they are completely expanded

exponential growth of search trees: a tree with a branching factor b and depth d

---> total number of paths: $\mathbf{b^d}$

Depth-First Search

To conduct a depth-first search,

- Form a one-element queue consisting of a zero-length path that contains only the root node.
- Until the first path in the queue terminates at the goal node or the queue is empty,
  - Remove the first path from the queue; create new paths by extending the first path to all the neighbors of the terminal node.
  - Reject all new paths with loops.
  - Add the new paths, if any, to the **front** of the queue.
- If the goal node is found, announce success; otherwise, announce failure.

Breadth-First Search

To conduct a breadth-first search,

- Form a one-element queue consisting of a zero-length path that contains only the root node.
- Until the first path in the queue terminates at the goal node or the queue is empty,
    - Remove the first path from the queue; create new paths by extending the first path to all the neighbors of the terminal node.
    - Reject all new paths with loops.
    - Add the new paths, if any, to the **back** of the queue.
- If the goal node is found, announce success; otherwise, announce failure.

Nondeterministic Search

To conduct a nondeterministic search,

- Form a one-element queue consisting of a zero-length path that contains only the root node.
- Until the first path in the queue terminates at the goal node or the queue is empty,
  - Remove the first path from the queue; create new paths by extending the first path to all the neighbors of the terminal node.
  - Reject all new paths with loops.
  - Add the new paths at **random places** in the queue.
- If the goal node is found, announce success; otherwise, announce failure.

Hill-Climbing Search

To conduct a hill-climbing search,

- Form a one-element queue consisting of a zero-length path that contains only the root node.
- Until the first path in the queue terminates at the goal node or the queue is empty,
    - Remove the first path from the queue; create new paths by extending the first path to all the neighbors of the terminal node.
    - Reject all new paths with loops.
    - **Sort the new paths, if any, by the estimated distances between their terminal nodes and the goal.**
    - Add the new paths, if any, to the front of the queue.
- If the goal node is found, announce success; otherwise, announce failure.

Other Heuristically Informed Methods

beam search = breath-first search **+**
moves forward only through the best w nodes at each level

best first search = hill-climbing by changing

- from: move forward from the most recently created open node
- to: move forward from the best open node

Search Alternatives Form a Procedure Family with Different Advantages

- **Depth-first search** is good when unproductive partial paths are never too long.
- **Breadth-first search** is good when the branching factor is never too large.
- **Nondeterministic search** is good when you are not sure whether depth-first search or breadth-first search would be better.
- **Hill climbing** is good when there is a natural measure of distance from each place to the goal and a good path is likely to be among the partial paths that appear to be good at each choice point.
- **Beam search** is good when there is a natural measure of goal distance and a good path is likely to be among the partial paths that appear to be good at all levels.
- **Best-first search** is good when there is a natural measure of goal distance and a good partial path may look like a bad option before more promising partial paths are played out.

Summary

- **Depth-first search** dives into the search tree, extending one partial path at a time.
- **Breadth-first search** pushes uniformly into the search tree, extending many partial paths in parallel.
- **Nondeterministic search** moves randomly into the search tree, picking a partial path to extend at random.
- **Heuristic quality measurements**
    - turn depth-first search into hill climbing. Foothills, plateaus, and ridges make hills hard to climb.
    - are used in beam search and best-first search. Beam search expands a fixed number of partial paths in parallel and purges the rest. Best-first search expands the best partial path.
- More knowledge generally means less search.
- When you think you need a better search method, try to find another space to search instead.
- search is an important topic in many fields (e.g., operations research) — AI's specific contribution lies in the development of **heuristic methods**