Breadth-first Search; Search with Costs

CPSC 322 – Search 3

Textbook §3.5
Lecture Overview

1 Recap

2 Breadth-First Search
Graph Search Algorithm

**Input:** a graph, a set of start nodes, Boolean procedure \( \text{goal}(n) \) that tests if \( n \) is a goal node.

\[
\text{frontier} := \{ \langle s \rangle : s \text{ is a start node} \};
\]

**while** \( \text{frontier} \) is not empty:

- **select** and **remove** path \( \langle n_0, \ldots, n_k \rangle \) from \( \text{frontier} \);
- **if** \( \text{goal}(n_k) \)
  - **return** \( \langle n_0, \ldots, n_k \rangle \);
- **for every** neighbor \( n \) of \( n_k 
  - **add** \( \langle n_0, \ldots, n_k, n \rangle \) to \( \text{frontier} \);

**end while**

- After the algorithm returns, it can be asked for more answers and the procedure continues.
- Which value is selected from the frontier defines the search strategy.
- The *neighbor* relationship defines the graph.
- The *goal* function defines what is a solution.
Depth-first Search

- **Depth-first search** treats the frontier as a stack
  - It always selects one of the last elements added to the frontier.

- **Complete** when the graph has no cycles and is finite
- **Time complexity** is $O(b^m)$
- **Space complexity** is $O(bm)$
DFS Example

- http://aispace.org/search/
- “simple tree graph”
Using Depth-First Search

- When is DFS **appropriate**?
When is DFS appropriate?
- space is restricted
- solutions tend to occur at the same depth in the tree
- you know how to order nodes in the list of neighbours so that solutions will be found relatively quickly
Using Depth-First Search

- When is DFS appropriate?
  - space is restricted
  - solutions tend to occur at the same depth in the tree
  - you know how to order nodes in the list of neighbours so that solutions will be found relatively quickly

- When is DFS inappropriate?

Recap Breadth-First Search

Breadth-First Search; Search with Costs

CPSC 322 – Search 3, Slide 6
Using Depth-First Search

- When is DFS **appropriate**?
  - space is restricted
  - solutions tend to occur at the same depth in the tree
  - you know how to order nodes in the list of neighbours so that solutions will be found relatively quickly

- When is DFS **inappropriate**?
  - some paths have infinite length
  - the graph contains cycles
  - some solutions are very deep, while others are very shallow
Lecture Overview

1 Recap

2 Breadth-First Search
Breadth-first Search

- Breadth-first search treats the frontier as a **queue**
  - it always selects one of the earliest elements added to the frontier.

**Example:**
- the frontier is \([p_1, p_2, \ldots, p_r]\)
- neighbours of \(p_1\) are \(\{n_1, \ldots, n_k\}\)

**What happens?**
- \(p_1\) is selected, and tested for being a goal.
- Neighbours of \(p_1\) follow \(p_r\) at the end of the frontier.
- Thus, the frontier is now \([p_2, \ldots, p_r, (p_1, n_1), \ldots, (p_1, n_k)]\).
- \(p_2\) is selected next.
BFS Example

- http://aispace.org/search/
- “simple tree graph”
Analysis of Breadth-First Search

- Is BFS complete?
Is BFS **complete**?

- Yes (but it wouldn’t be if the branching factor for any node was infinite)
- In fact, BFS is guaranteed to find the path that involves the fewest arcs (why?)
Recap Breadth-First Search

Analysis of Breadth-First Search

- Is BFS complete?
  - Yes (but it wouldn’t be if the branching factor for any node was infinite)
  - In fact, BFS is guaranteed to find the path that involves the fewest arcs (why?)
- What is the time complexity, if the maximum path length is $m$ and the maximum branching factor is $b$?
Analysis of Breadth-First Search

- Is BFS **complete**?
  - Yes (but it wouldn’t be if the branching factor for any node was infinite)
  - In fact, BFS is guaranteed to find the path that involves the fewest arcs (why?)

- What is the time complexity, if the maximum path length is $m$ and the maximum branching factor is $b$?
  - The time complexity is $O(b^m)$: must examine every node in the tree.
  - The order in which we examine nodes (BFS or DFS) makes no difference to the worst case: search is unconstrained by the goal.
Analysis of Breadth-First Search

- **Is BFS complete?**
  - Yes (but it wouldn’t be if the branching factor for any node was infinite)
  - In fact, BFS is guaranteed to find the path that involves the fewest arcs (why?)

- **What is the time complexity**, if the maximum path length is $m$ and the maximum branching factor is $b$?
  - The time complexity is $O(b^m)$: must examine every node in the tree.
  - The order in which we examine nodes (BFS or DFS) makes no difference to the worst case: search is unconstrained by the goal.

- **What is the space complexity?**
Analysis of Breadth-First Search

- Is BFS **complete**?
  - Yes (but it wouldn’t be if the branching factor for any node was infinite)
  - In fact, BFS is guaranteed to find the path that involves the fewest arcs (why?)

- **What is the time complexity**, if the maximum path length is $m$ and the maximum branching factor is $b$?
  - The time complexity is $O(b^m)$: must examine every node in the tree.
  - The order in which we examine nodes (BFS or DFS) makes no difference to the worst case: search is unconstrained by the goal.

- **What is the space complexity**?
  - Space complexity is $O(b^m)$: we must store the whole frontier in memory
When is BFS appropriate?
Using Breadth-First Search

- When is BFS **appropriate**?
  - space is not a problem
  - it’s necessary to find the solution with the fewest arcs
  - although all solutions may not be shallow, at least some are
  - there may be infinite paths
Using Breadth-First Search

- When is BFS **appropriate**?
  - space is not a problem
  - it’s necessary to find the solution with the fewest arcs
  - although all solutions may not be shallow, at least some are
  - there may be infinite paths

- When is BFS **inappropriate**?
Using Breadth-First Search

- When is BFS **appropriate**?
  - space is not a problem
  - it’s necessary to find the solution with the fewest arcs
  - although all solutions may not be shallow, at least some are
  - there may be infinite paths

- When is BFS **inappropriate**?
  - space is limited
  - all solutions tend to be located deep in the tree
  - the branching factor is very large