Breadth-first Search

Zezhou Jing

Rev. September 21, 2016

Preliminaries

Breadth-first search (BFS) is a *search algorithm* to traverse trees or graphs in order of nodes' (or vertices') distance from the starting node. We may say that BFS searches the graph by surroundings, neighbors, or layers of s.

Two input parameters are a graph G and a starting vertex s. BFS algorithm returns all reachable vertices within the graph from starting vertex s while marking them as explored. BFS does not explore or mark any vertex twice. We use a *queue* data structure as the generic storage for nodes.

BFS enables us to compute shortest paths and connected components of an undirected graph.

Performance

Let m_s be the number of vertices $|V|$ and n_s be the number of edges $|E|$ reachable from s, both of which are nonnegative.

Time complexity:

- Linear time $\mathcal{O}(m_s + n_s)$ in the worst case (some may write it simply as $\mathcal{O}(|E|)$ given that $n_s \gg m_s$).

Space complexity:

- $\mathcal{O}(m_s)$ in the worst case since a queue is required to store nodes.

Implementation

- Start with a graph G and a starting vertex s. Initially all vertices are marked unexplored.
- Mark s as explored.
- Initialize a queue data structure Q. Enqueue s into Q.
- Explore all adjacent vertices as follows. While Q is not empty:
  - Dequeue the first vertex v from Q and explore all its adjacent vertices.
  - For each edge starting from v and leading to another vertex w, if w is unexplored, mark it as explored and enqueue w into Q.

Analysis

Shortest Paths

Connectivity and Connected Components