Principles of Distributed Computing
Roger Wattenhofer, Summer 2003

Chapter 3

Tree Algorithms

Section 3.1: Broadcast

Definition 3.1 [Broadcast]: A broadcast operation is initiated by a single processor, the source. The source wants to send a message to all other nodes in the system.

Definition 3.2 [Distance, Radius, Diameter]: The distance between two nodes u, v in an undirected graph is the number of hops of a minimum path between u and v. The radius of a node u in a graph is the maximum distance between u and any other node. The radius of a graph is the minimum radius of any node in the graph. The diameter of a graph is the maximum distance between two arbitrary nodes.

Remarks:
- The diameter is about twice the radius.
- Kevin Bacon, Paul Erdös, etc.

Theorem 3.3 [Lower Bound]: The message complexity of a broadcast is at least n-1. The radius of the graph is a lower bound for the time complexity.

Proof: Every node must receive the message.

Remarks:
- You can use a pre-computed spanning tree to do the broadcast with tight message complexity.
- If the spanning tree is a breadth-first spanning tree (for a given source), then also the time complexity is tight.

Definition 3.4 [Clean]: A graph (system/network) is clean if the nodes do not know the topology of the graph.

Theorem 3.5 [Clean Lower Bound]: For a clean network, the number of edges is a lower bound for the broadcast message complexity.

Proof: If you do not try every edge, you might miss a whole part of the graph behind it.

Algorithm 3.6 [Flooding]: The source sends the message to all neighbors. Each node receiving the message the first time forwards to all (other) neighbors.

Remarks:
- If node v receives the message first from node u, then node v calls node u "parent". This parent relation defines a spanning tree T. If the flooding algorithm is executed in a synchronous system, then T is a breadth-first spanning tree (with respect to the root).

- More interestingly, also in asynchronous systems the flooding algorithm terminates after r time units, where r is the radius of the source. (But note that the constructed spanning tree needs not be breadth-first.)

Section 3.2: Convergecast

(Broadcast: Termination detection)
(Broadcast with echo)
(Same as broadcast, just reverse)

Algorithm 3.7 [Echo]: Leaves send an ACK back to their parent. If a node has received ACKs from all children (all but the parent neighbor), it sends an ACK to the parent node.

Remarks:
- Message complexity of Echo is n-1, but together with flooding still O(|E|).
- Time complexity = radius (depth) of the spanning tree of the flooding algorithm.
- Very important remark: The flooding/echo (or broadcast/echo) algorithm can do much more than just collecting ACKS:
  - Example 1: Compute sum of values stored at nodes in the system.
  - Example 2: Find the maximum identifier for leader election. Root?!?
  - Example 3: Compute a route-disjoint matching.
- How does one compute a breadth-first tree in the asynchronous model?

Section 3.3: BFS Tree Construction

(Flooding was good solution for synchronous system)
(Two basic sequential algorithms: Dijkstra & Bellman-Ford)

(Dijkstra: Always add closest new node → develop BFS tree layer by layer)

Algorithm 3.8 [Dijkstra BFS tree]: The algorithm proceeds in phases. In phase p the nodes with distance p to the root are detected. T_p is the tree in phase p. We start with T_1 which is the root plus all direct neighbors of the root. Each phase is as follows:
- The root starts phase p by broadcasting "start p" within T_p.
- When receiving "start p" a "new leaf" node u of T_p ("new leaf" = a node that was newly discovered in the last phase) sends a "join p+1" message to all quiet neighbors. (A neighbor v is quiet if u has not yet received a message from v.)
- A node v receiving the first "join p+1" message replies with "ack" and becomes a leave of the tree T_{p+1}.
- A node v receiving any further "join" message replies with "nack".
- The leaves of T_p collect all the answers of their neighbors; then the leaves start the echo algorithm back to the root.
- When the echo is terminated at the root, the root starts phase p+1, unless there was no new node detected.

Theorem 3.9 [Analysis of Algorithm 3.8]: The time complexity of Algorithm 3.8 is $O(D^2)$, the message complexity is $O(|E|+nD)$, where D is the diameter of the graph.

Proof: The broadcast & echo algorithm in T_p needs at most time 2D. Finding new neighbors at the leaves costs time 2. Since the BFS tree height is bounded by the diameter we have D phases, giving a total time complexity of $O(D^2)$. Each node participating in broadcast & echo only receives (broadcast) at most 1 message and sends (echo) at most 1. Since there are D phases, the cost is bounded by O(nD). On each edge there are at most 2 "join" messages. Replies to a "join" request are answered by 1 "ack" or "nack", which means that we have at most 4 additional messages per edge. Therefore the message complexity is O(|E|+nD).

(Bellman-Ford: Simply flood the network with a number-of-hops counter in each message)

Algorithm 3.10 [Bellman-Ford BFS tree]: Use a variant of the flooding algorithm. Each node and each message store an integer which corresponds to the distance from the root. The root stores 0, every other node initially ∞. The root starts the flooding algorithm by sending a message "1" to all neighbors.

- A node u with integer x receives a message "y" from a neighbor v: if y < x then node u stores y (instead of x) and sends "y+1" to all neighbors (except v).

Theorem 3.11 [Analysis of Algorithm 3.10]: The time complexity of Algorithm 3.10 is O(D), the message complexity is O(n|E|), where D is the diameter of the graph.

Proof: We can prove the time complexity by induction. We claim that a node at distance d from the root has received a message "d" by time d. The root knows by time 0 that it is the root. A node v at distance d has a neighbor u at distance d-1. Node u by induction sends a message "d" to v at time d-1 or before, which is then received by v at time d or before. Message complexity is easier: A node can reduce its integer at most n-1 times; each of these times it sends a message to all it neighbors. If all nodes do this we have O(n|E|) messages.

Remarks:

- There are graphs and executions that produce O(n|E|) messages.
- How does the algorithm terminate?
- Algorithm 3.8 has the better message complexity; algorithm 3.10 has the better time complexity. The currently best known algorithm has message complexity $O(|E|+n \log^3 n)$ and time complexity $O(D \log^3 n)$.
- How do we find the root?!? Leader election in an arbitrary graph: FloodMax algorithm. Termination? Idea: Each node that believes to be the "max" builds a spanning tree… (More for example in Chapter 15 of Nancy Lynch "Distributed Algorithms")