Dijkstra–Scholten algorithm

From Wikipedia, the free encyclopedia
(Redirected from Dijkstra-Scholten algorithm)

The **Dijkstra–Scholten algorithm** (named after Edsger W. Dijkstra and Carel S. Scholten) is an algorithm for detecting termination in a distributed system.[1][2] The algorithm was proposed by Dijkstra and Scholten in 1980.[3]

First, let us consider the case of a simple process graph which is a tree. A distributed computation which is tree-structured is not uncommon. Such a process graph may arise when the computation is strictly a divide-and-conquer type. A node starts the computation and divides the problem in two (or more, usually a multiple of 2) roughly equal parts and distribute those parts to other processors. This process continues recursively until the problems are of sufficiently small size to solve in a single processor.

Contents [hide]

1 Algorithm
2 Dijkstra–Scholten algorithm for a tree
3 Dijkstra–Scholten algorithm for directed acyclic graphs
4 Dijkstra–Scholten algorithm for cyclic directed graphs
5 See also
6 References

Algorithm [edit]

The Dijkstra–Scholten algorithm is a tree-based algorithm which can be described by the following:

- The initiator of a computation is the root of the tree.
- Upon receiving a computational message:
  - If the receiving process is currently not in the computation: the process joins the tree by becoming a child of the sender of the message. (No acknowledgment message is sent at this point.)
  - If the receiving process is already in the computation: the process immediately sends an acknowledgment message to the sender of the message.
- When a process has no more children and has become idle, the process detaches itself from the tree by sending an acknowledgment message to its tree parent.
- Termination occurs when the initiator has no children and has become idle.

Dijkstra–Scholten algorithm for a tree [edit]

- For a tree, it is easy to detect termination. When a leaf process determines that it has terminated, it sends a signal to its parent. In general, a process waits for all its children to send signals and then it sends a signal to its parent.
- The program terminates when the root receives signals from all its children.

Dijkstra–Scholten algorithm for directed acyclic graphs [edit]

- The algorithm for a tree can be extended to acyclic directed graphs. We add an additional integer attribute Deficit to each edge.
- On an incoming edge, Deficit will denote the difference between the number of messages received and the number of signals sent in reply.
- When a node wishes to terminate, it waits until it has received signals from outgoing edges reducing their deficits to zero.
- Then it sends enough signals to ensure that the deficit is zero on each incoming edge.
- Since the graph is acyclic, some nodes will have no outgoing edges and these nodes will be the first to terminate after sending enough signals to their incoming edges. After that the nodes at higher levels will terminate level by level.

Dijkstra–Scholten algorithm for cyclic directed graphs [edit]

- If cycles are allowed, the previous algorithm does not work. This is because, there may not be any node with zero outgoing edges. So, potentially there is no node which can terminate without consulting other nodes.
- The Dijkstra–Scholten algorithm solves this problem by implicitly creating a spanning tree of the graph. A spanning-tree is a tree which includes each node of the underlying graph once and the edge-set is a subset of the original set of edges.
- The tree will be directed (i.e., the channels will be directed) with the source node (which initiates the computation) as the root.
- The spanning-tree is created in the following way. A variable *First_Edge* is added to each node. When a node receives a message for the first time, it initializes *First_Edge* with the edge through which it received the message. *First_Edge* is never changed afterwards. Note that, the spanning tree is not unique and it depends on the order of messages in the system.
- Termination is handled by each node in three steps :
    1. Send signals on all incoming edges except the first edge. (Each node will send signals which reduces the deficit on each incoming edge to zero.)
    2. Wait for signals from all outgoing edges. (The number of signals received on each outgoing edge should reduce each of their deficits to zero.)
    3. Send signals on *First_Edge*. (Once steps 1 and 2 are complete, a node informs its parent in the spanning tree about its intention of terminating.)

See also [edit]

- Huang's algorithm

References [edit]

1. ^ Ghosh, Sukumar (2010), "9.3.1 The Dijkstra–Scholten Algorithm", *Distributed Systems: An Algorithmic Approach*, CRC Press, pp. 140–143, ISBN 9781420010848.
2. ^ Fokkink, Wan (2013), "6.1 Dijkstra–Scholten algorithm", *Distributed Algorithms: An Intuitive Approach*, MIT Press, pp. 38–39, ISBN 9780262318952.
3. ^ Dijkstra, Edsger W.; Scholten, C. S. (1980), "Termination detection for diffusing computations" (PDF), *Information Processing Letters* **11** (1): 1–4, doi:10.1016/0020-0190(80)90021-6, MR 585394.

Categories: Graph algorithms | Termination algorithms