M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Distributed Systems
Course Code: CSE751/CSE20
Credits: 3:0:0

Term: Oct 2021 – Feb 2022

Faculty:
Sini Anna Alex

Huang’s Termination detection algorithm

Huang’s algorithm is an algorithm for detecting termination in a distributed system. The algorithm was proposed by **Shing-Tsaan Huang** in 1989 in the Journal of Computers.

Advantages of Huang’s Algorithm:

The algorithm detects every true termination in finite time.

Limitations of Huang’s Algorithm:

- The algorithm is unable to detect computation termination if a message is lost in transit.
- It also does not work when a process fails while in an active state.

Consider the following statements about termination detection (TD) algorithm

Statement 1: Execution of a termination detection algorithm cannot indefinitely delay the underlying computation.

Statement 2: The termination detection algorithm required addition of new communication channels between processes.

A. Statement 1 is true and statement 2 is false

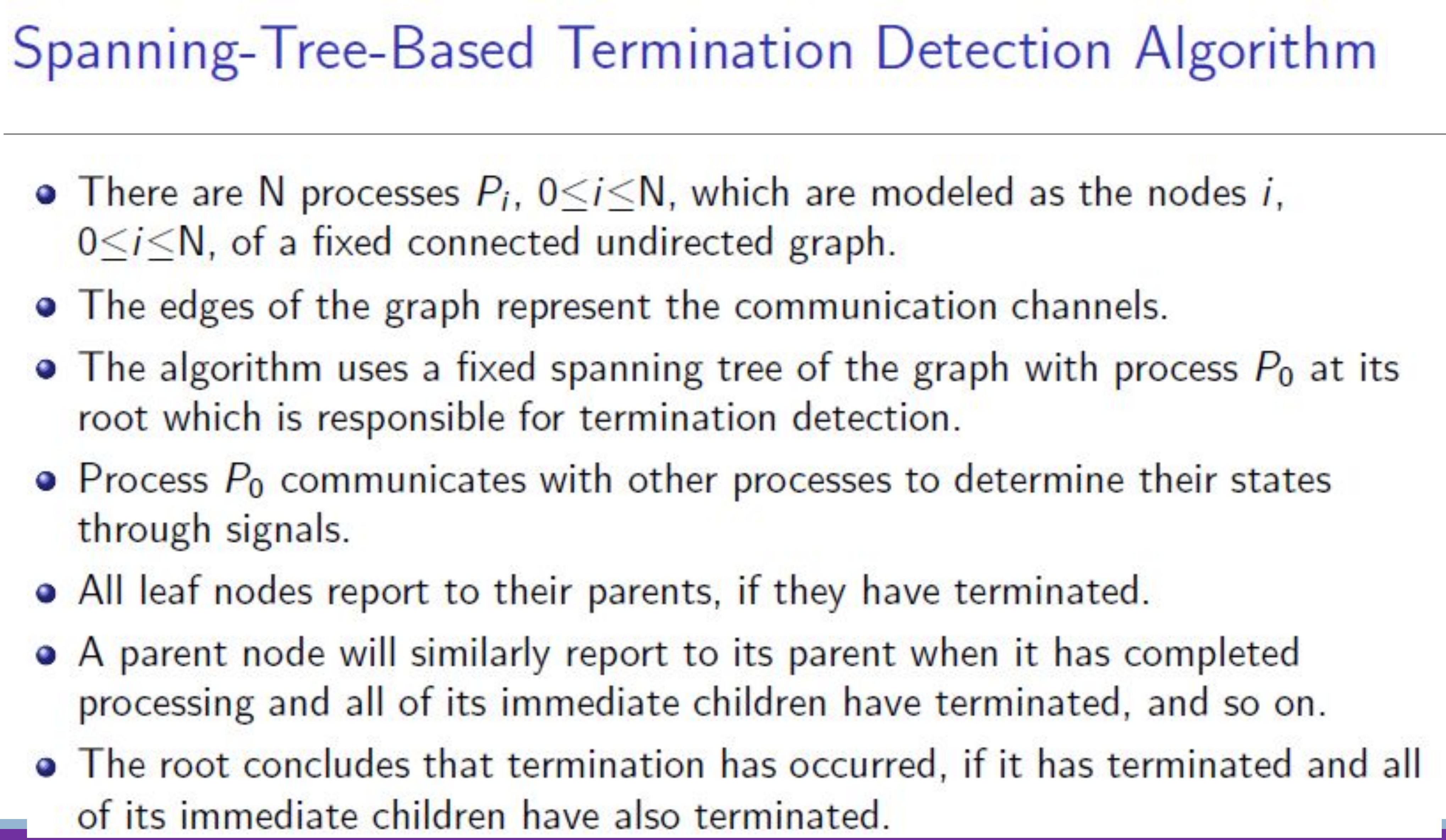
B. Statement 1 is false and statement 2 is true

C. Both statements are false

D. Both statements are true

A state in which a process has finished its computation and will not restart any action unless it receives a message is called as

A. Partially terminated state
B. Locally terminated state
C. Globally terminated state
D. Terminating state



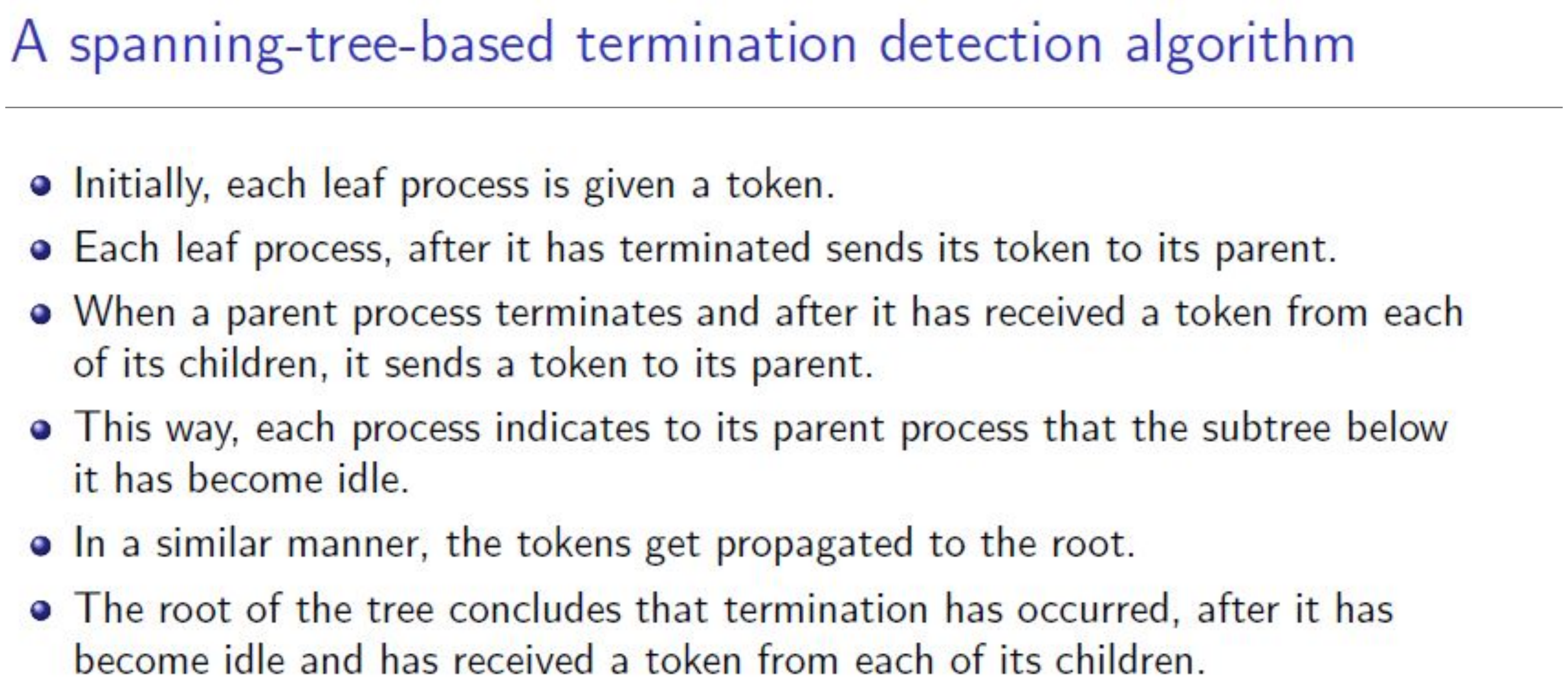
Spanning-Tree-Based Termination Detection Algorithm

- There are N processes P_i, $0 \leq i \leq N$, which are modeled as the nodes i, $0 \leq i \leq N$, of a fixed connected undirected graph.
- The edges of the graph represent the communication channels.
- The algorithm uses a fixed spanning tree of the graph with process P_0 at its root which is responsible for termination detection.
- Process P_0 communicates with other processes to determine their states through signals.
- All leaf nodes report to their parents, if they have terminated.
- A parent node will similarly report to its parent when it has completed processing and all of its immediate children have terminated, and so on.
- The root concludes that termination has occurred, if it has terminated and all of its immediate children have also terminated.

RAMAIAH
Institute of Technology

Spanning-Tree-Based Termination Detection Algorithm

- Two waves of signals generated one moving inward and other outward through the spanning tree.
- Initially, a contracting wave of signals, called *tokens*, moves inward from leaves to the root.
- If this token wave reaches the root without discovering that termination has occurred, the root initiates a second outward wave of *repeat* signals.
- As this repeat wave reaches leaves, the token wave gradually forms and starts moving inward again, this sequence of events is repeated until the termination is detected.



A spanning-tree-based termination detection algorithm

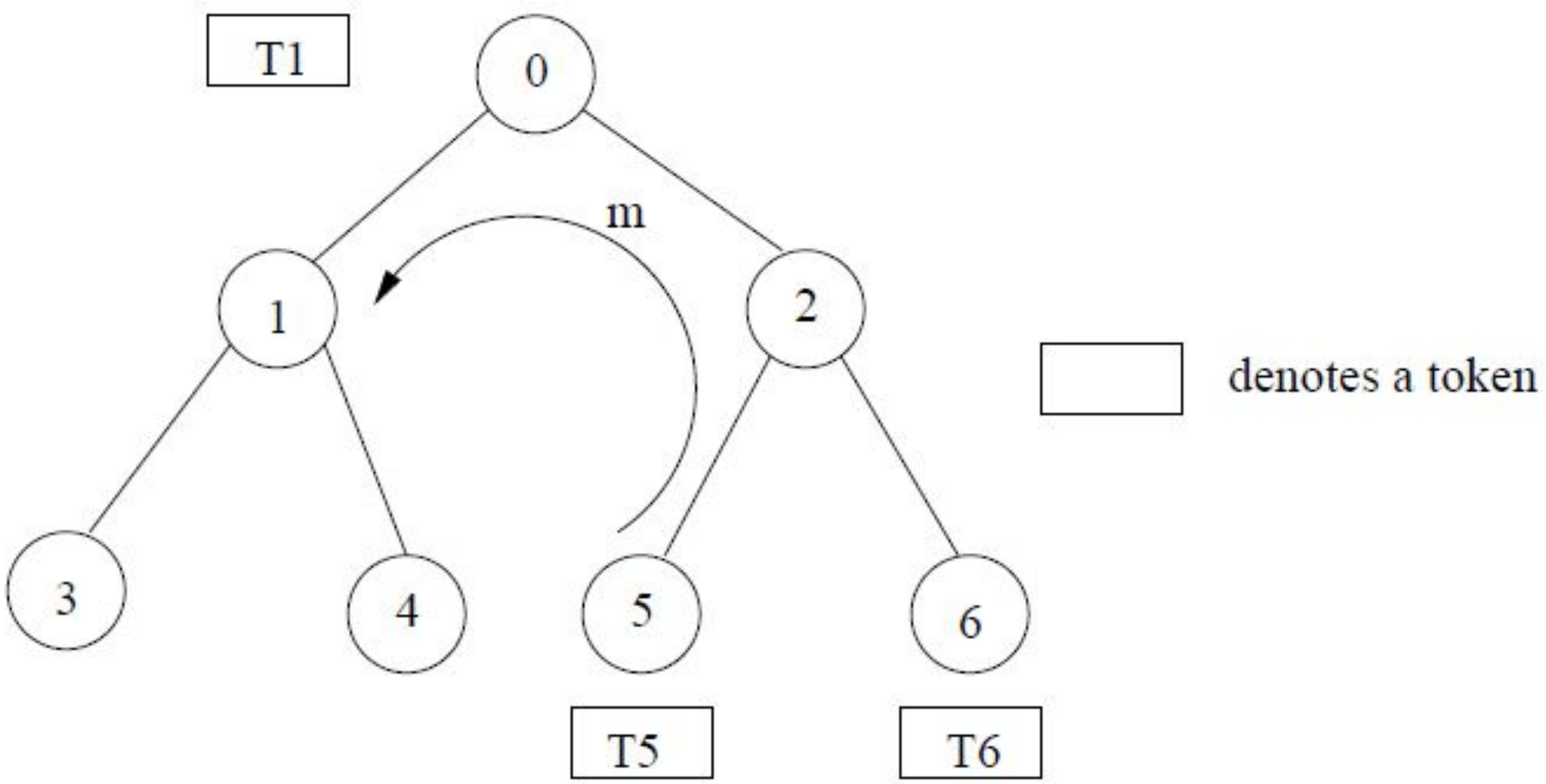
- Initially, each leaf process is given a token.
- Each leaf process, after it has terminated sends its token to its parent.
- When a parent process terminates and after it has received a token from each of its children, it sends a token to its parent.
- This way, each process indicates to its parent process that the subtree below it has become idle.
- In a similar manner, the tokens get propagated to the root.
- The root of the tree concludes that termination has occurred, after it has become idle and has received a token from each of its children.

Spanning-Tree-Based Termination Detection Algorithm

A Problem with the algorithm

- This simple algorithm fails under some circumstances, when a process after it has sent a token to its parent, receives a message from some other process, which could cause the process to again become active
- Hence the simple algorithm fails since the process that indicated to its parent that it has become idle, is now active because of the message it received from an active process.
- Hence, the root node just because it received a token from a child, can't conclude that all processes in the child's subtree have terminated.

A spanning-tree-based termination detection algorithm

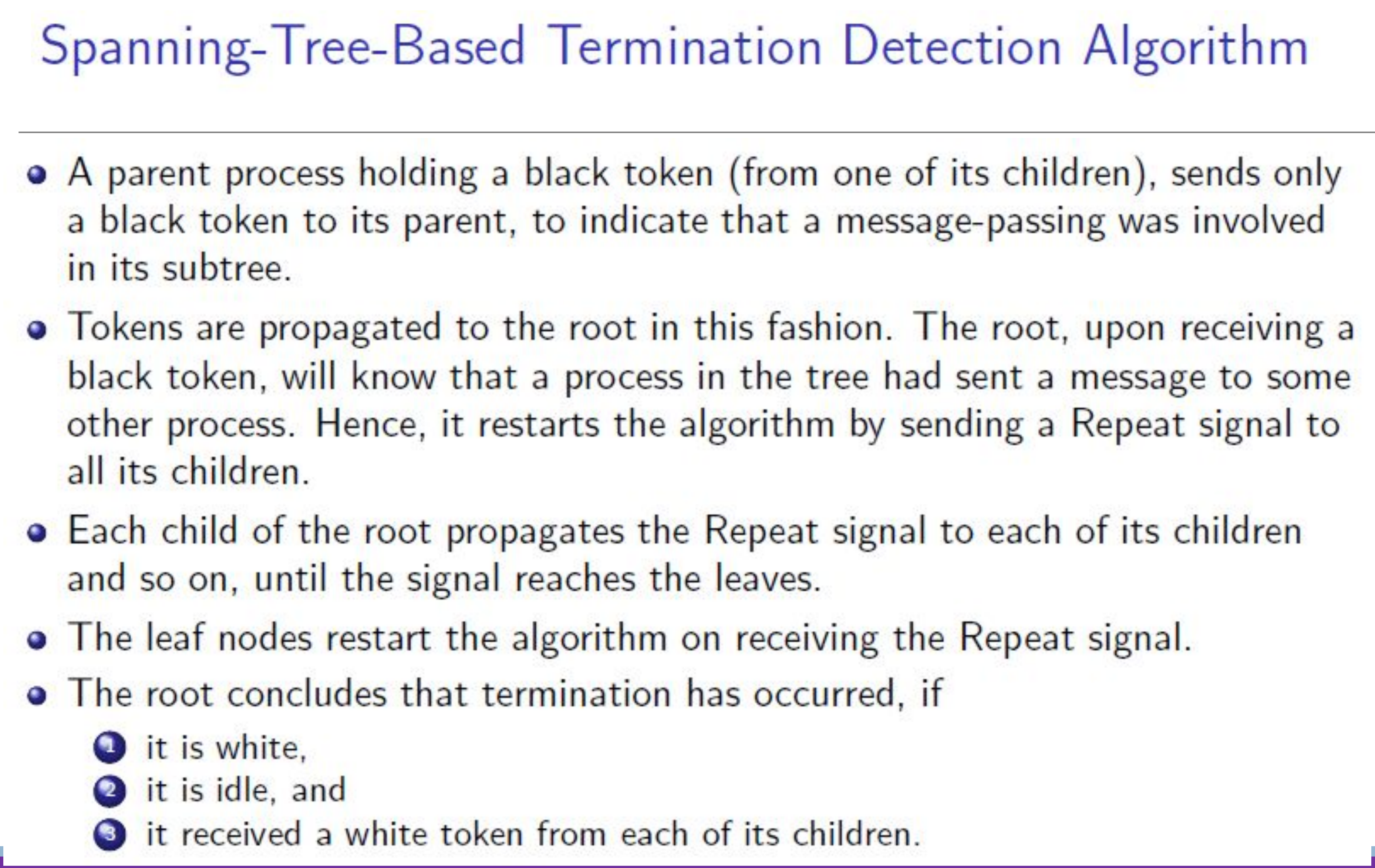


Spanning-Tree-Based Termination Detection Algorithm

- Main idea is to color the processes and tokens and change the color when messages such as in Figure 1 are involved.

The algorithm works as follows:

- Initially, each leaf process is provided with a token. The set S is used for book-keeping to know which processes have the token Hence S will be the set of all leaves in the tree.
- Initially, all processes and tokens are colored white.
- When a leaf node terminates, it sends the token it holds to its parent process.
- A parent process will collect the token sent by each of its children. After it has received a token from all of its children and after it has terminated, the parent process sends a token to its parent.
- A process turns black when it sends a message to some other process. When a process terminates, if its color is black, it sends a black token to its parent.
- A black process turns back to white, after it has sent a black token to its parent.

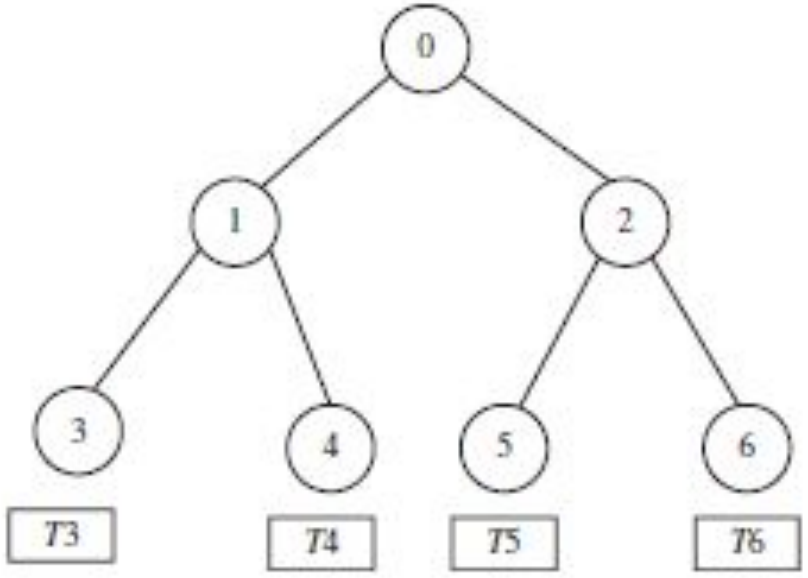


Spanning-Tree-Based Termination Detection Algorithm

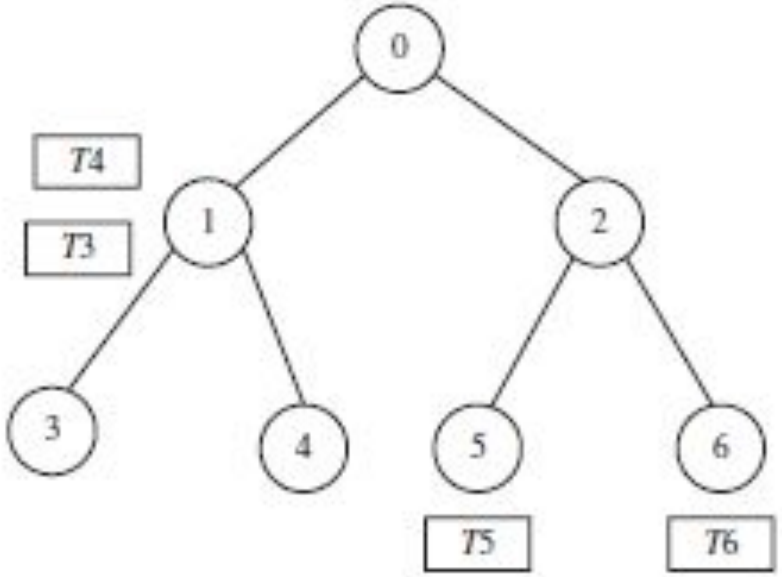
- A parent process holding a black token (from one of its children), sends only a black token to its parent, to indicate that a message-passing was involved in its subtree.
- Tokens are propagated to the root in this fashion. The root, upon receiving a black token, will know that a process in the tree had sent a message to some other process. Hence, it restarts the algorithm by sending a Repeat signal to all its children.
- Each child of the root propagates the Repeat signal to each of its children and so on, until the signal reaches the leaves.
- The leaf nodes restart the algorithm on receiving the Repeat signal.
- The root concludes that termination has occurred, if
  1. it is white,
  2. it is idle, and
  3. it received a white token from each of its children.

TD Spanning Tree Example

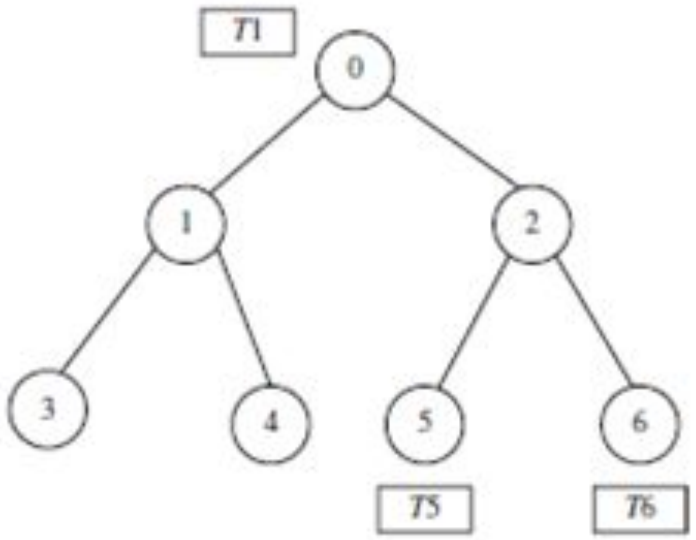
All leaf nodes have tokens. $S = \{3, 4, 5, 6\}$.



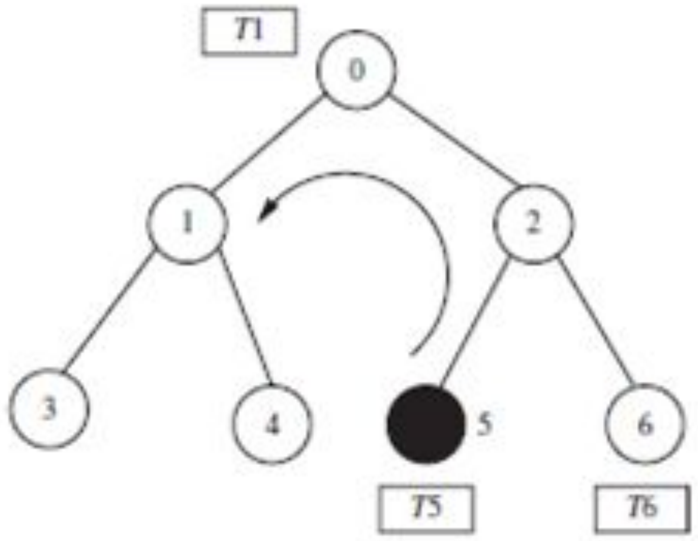
Nodes 3 and 4 become idle. $S = \{1, 5, 6\}$.



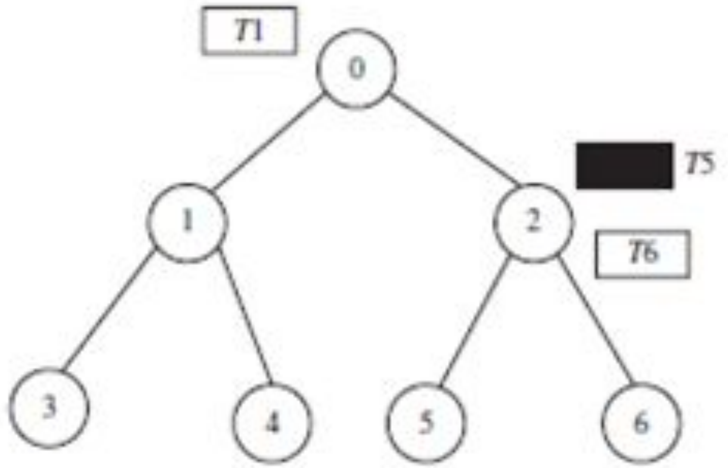
Node 1 becomes idle. $S = \{0, 5, 6\}$.



Node 5 sends a message to node 1.



Nodes 5 and 6 become idle. $S = \{0, 2\}$.



Spanning-Tree-Based Termination Detection Algorithm

Performance

- The best case message complexity of the algorithm is O(N), where N is the number of processes in the computation, which occurs when all nodes send all computation messages in the first round.
- The worst case complexity of the algorithm is O(N*M), where M is the number of computation messages exchanged, which occurs when only computation message is exchanged every time the algorithm is executed.

RAMAIAH
Institute of Technology

Thank you