Election Algorithms

Election Algorithms

- If we are using one process as a coordinator for a shared resource …
- …how do we select that one process?
- Often, there is no *owner* or *master* that is automatically considered as coordinator
  - E.g., Grapevine, there is no *owner* for a *Registry*
  - By contrast:–DNS has a master for every domain

Solution – an Election

- All nodes currently involved get together to *choose* a coordinator
- If the coordinator crashes or becomes isolated, elect a new coordinator
- If a previously crashed or isolated node, comes on line, a new election *may* have to be held.

Election Algorithms

- Wired systems
  - Bully algorithm
  - Ring algorithm
- Wireless systems
- Very large-scale systems

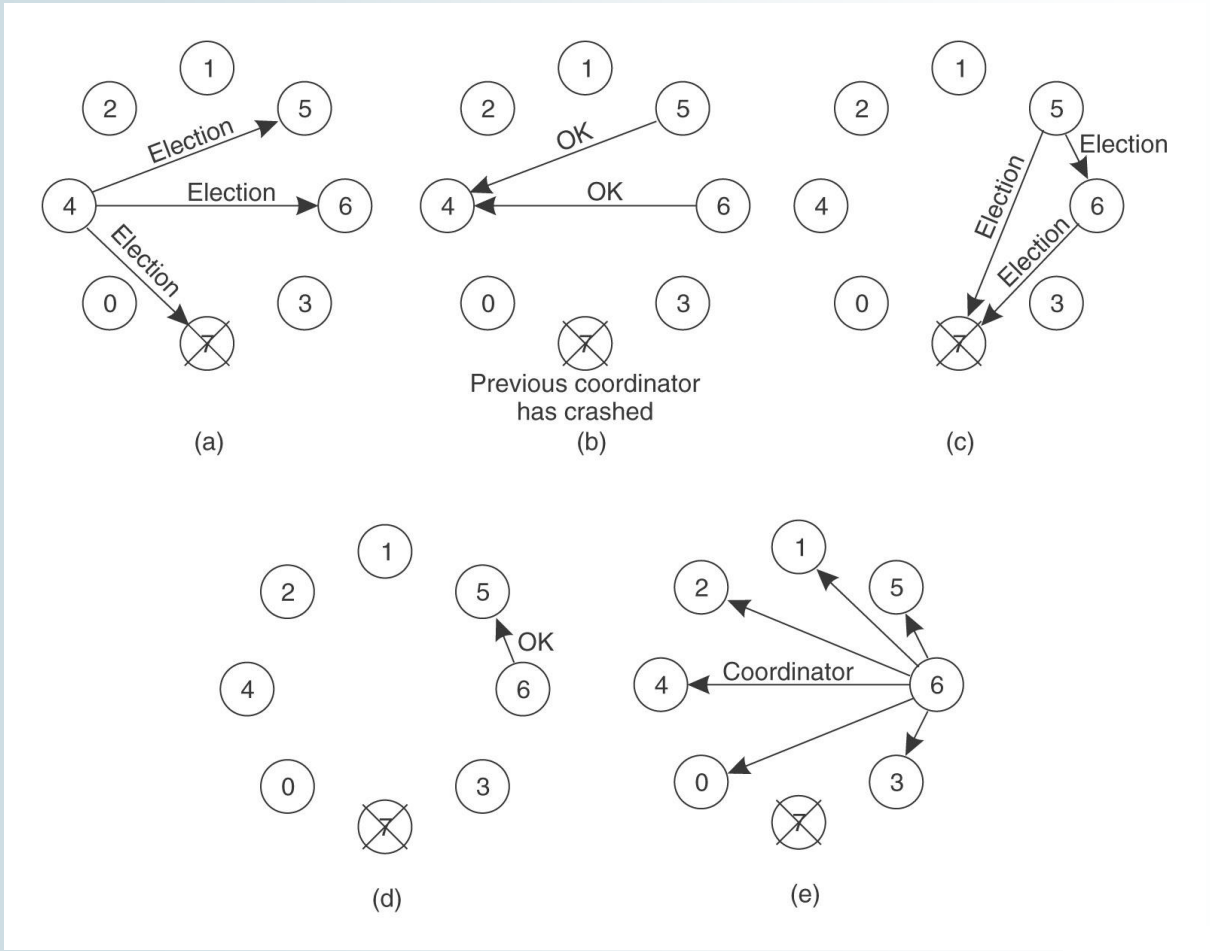
Bully Algorithm

- Assume
  - All processes know about each other
  - Processes numbered uniquely
- Suppose *P* notices no coordinator
  - Sends *election message* to all higher numbered processes
  - If none response, *P* takes over as coordinator
  - If any responds, *P* yields
- …

Bully Algorithm (continued)

- ...
- Suppose *Q* receives *election message*
  - Replies *OK* to sender, saying it will take over
  - Sends a new *election message* to higher numbered processes
- Repeat until only one process left standing
  - Announces victory by sending message saying that it is coordinator

Bully Algorithm (continued)



Bully Algorithm (continued)

- ...
- Suppose *R* comes back on line
  - Sends a new *election message* to higher numbered processes
- Repeat until only one process left standing
  - Announces victory by sending message saying that it is coordinator (if not already coordinator)
- Existing (lower numbered) coordinator yields
  - Hence the term “bully”

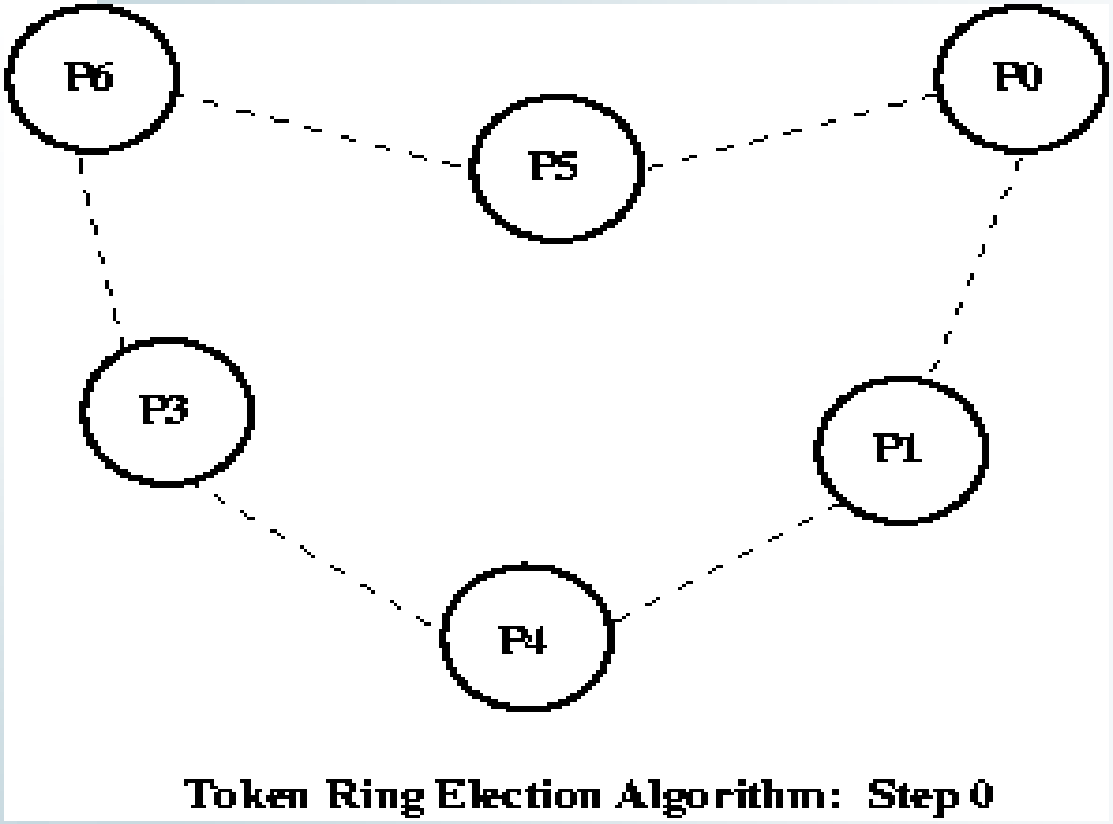
Alternative – Ring Algorithm

- All processed organized in ring
  - Independent of process number
- Suppose *P* notices no coordinator
  - Sends *election message* to successor with own process number in body of message
  - (If successor is down, skip to next process, etc.)
- Suppose *Q* receives an election message
  - Adds own process number to list in message body
- ...

Alternative – Ring Algorithm

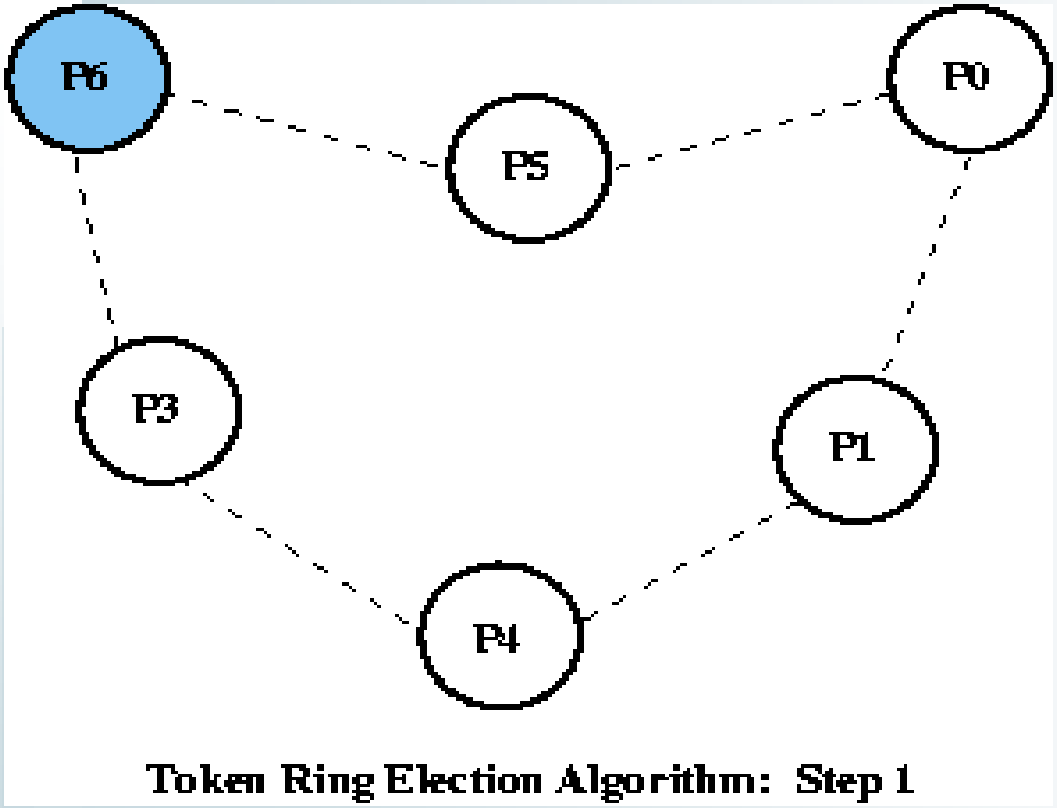
- Suppose *P* receives an election message with its own process number in body
  - Changes message to *coordinator* message, preserving body
  - All processes recognize *highest numbered process* as new coordinator
- If multiple messages circulate …
  - …they will all contain same list of processes (eventually)
- If process comes back on-line
  - Calls new election

Ring Algorithm (continued)



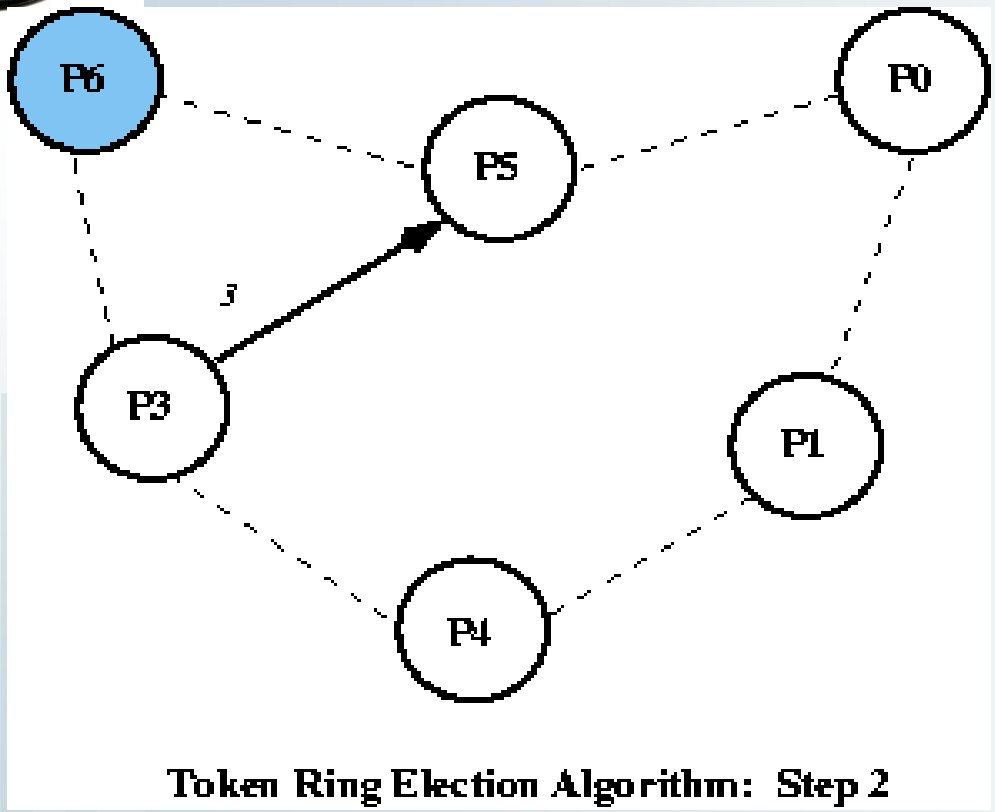
We start with 6 processes,
connected in a logical ring.
Process 6 is the leader,
as it has the highest number.

Ring Algorithm (continued)



Process 6 fails

Ring Algorithm (continued)

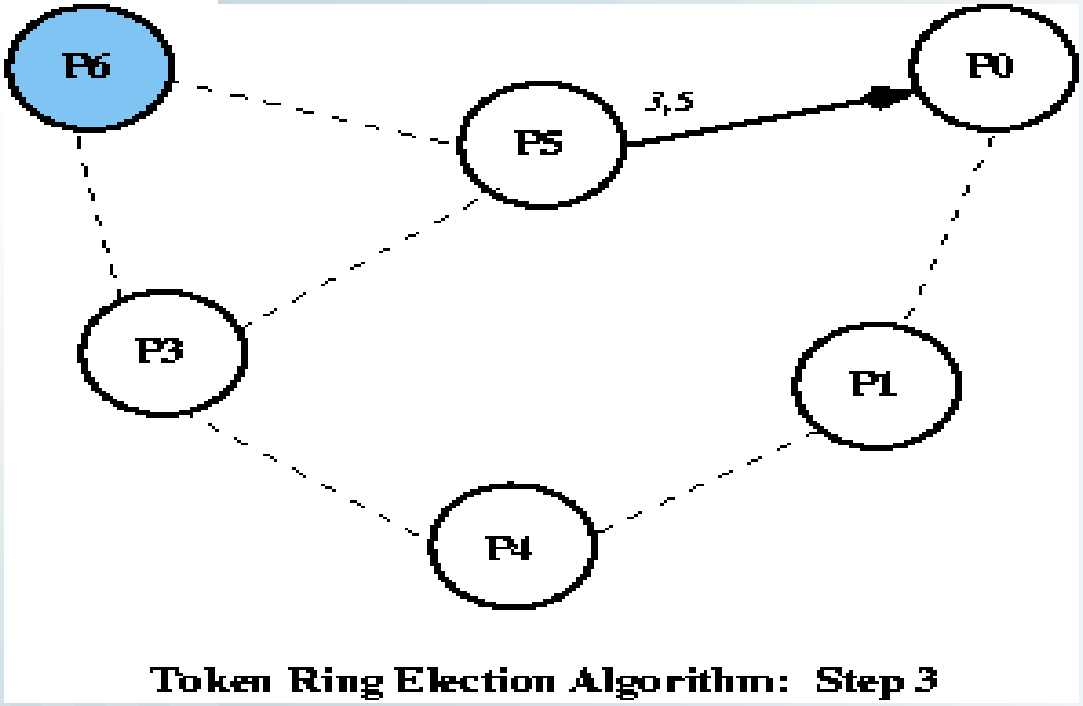


Token Ring Election Algorithm: Step 2

Process 3 notices that Process 6 does not respond

So it starts an election, sending a message containing its id to the next node in the ring.

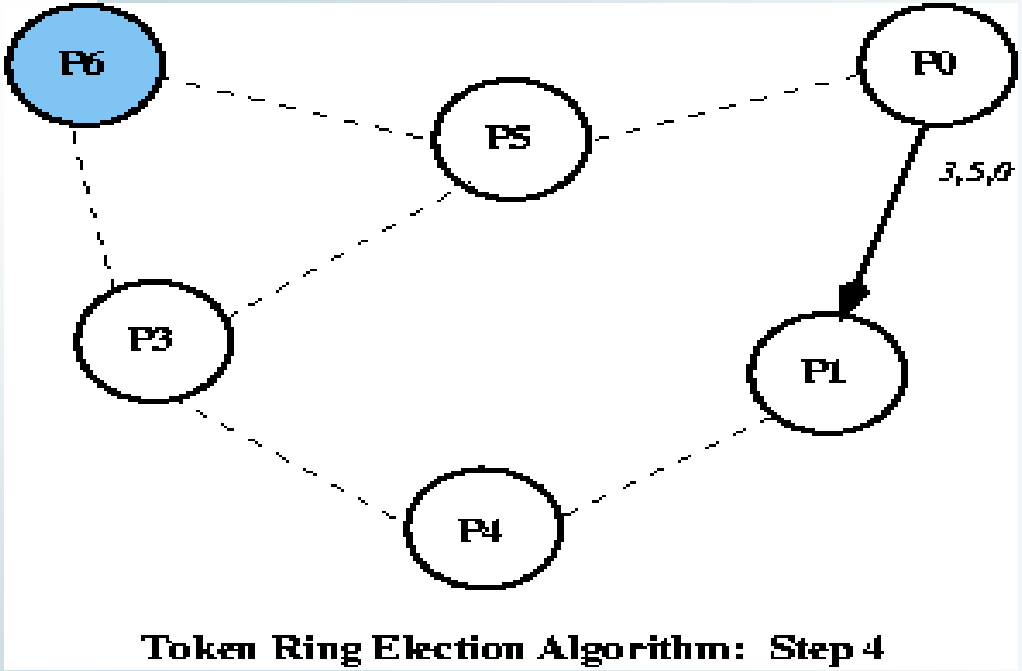
Ring Algorithm (continued)



Token Ring Election Algorithm: Step 3

Process 5 passes the message on, adding its own id to the message.

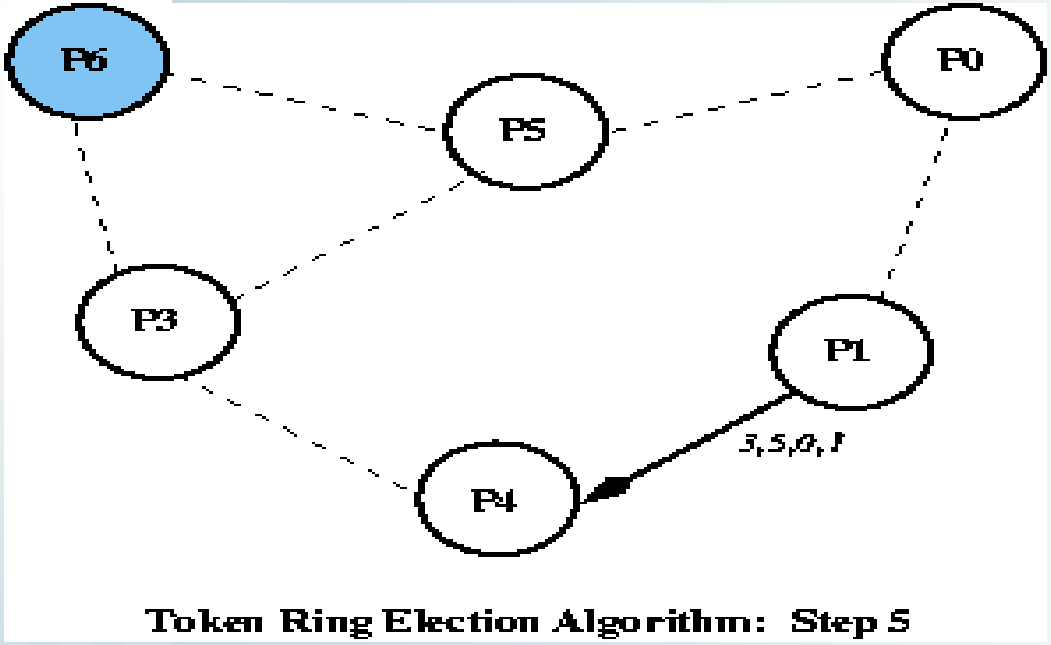
Ring Algorithm (continued)



Token Ring Election Algorithm: Step 4

Process 0 passes the message on,
adding its own id to the message.

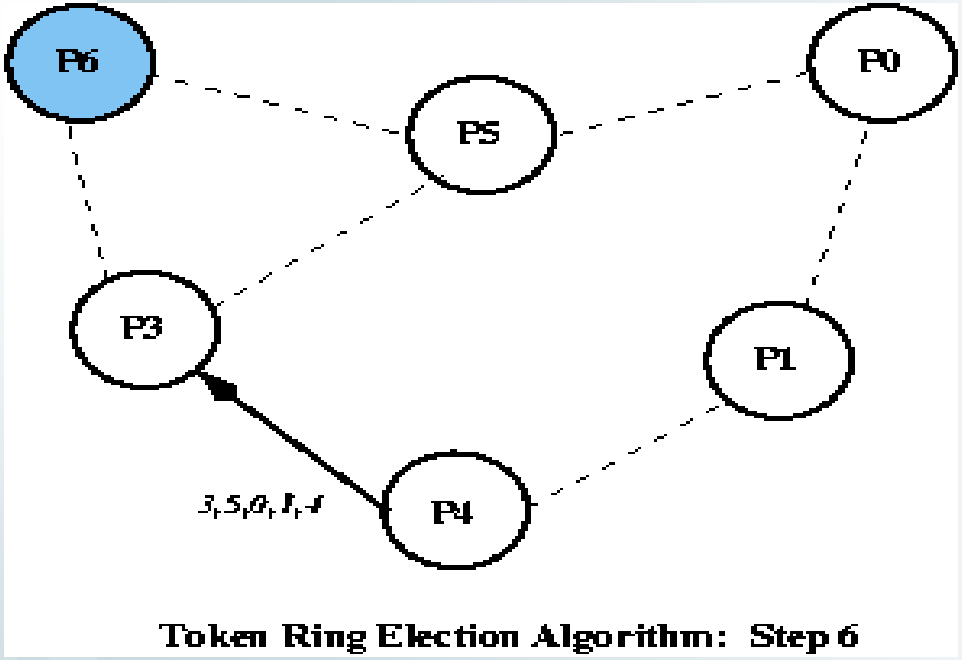
Ring Algorithm (continued)



Token Ring Election Algorithm: Step 5

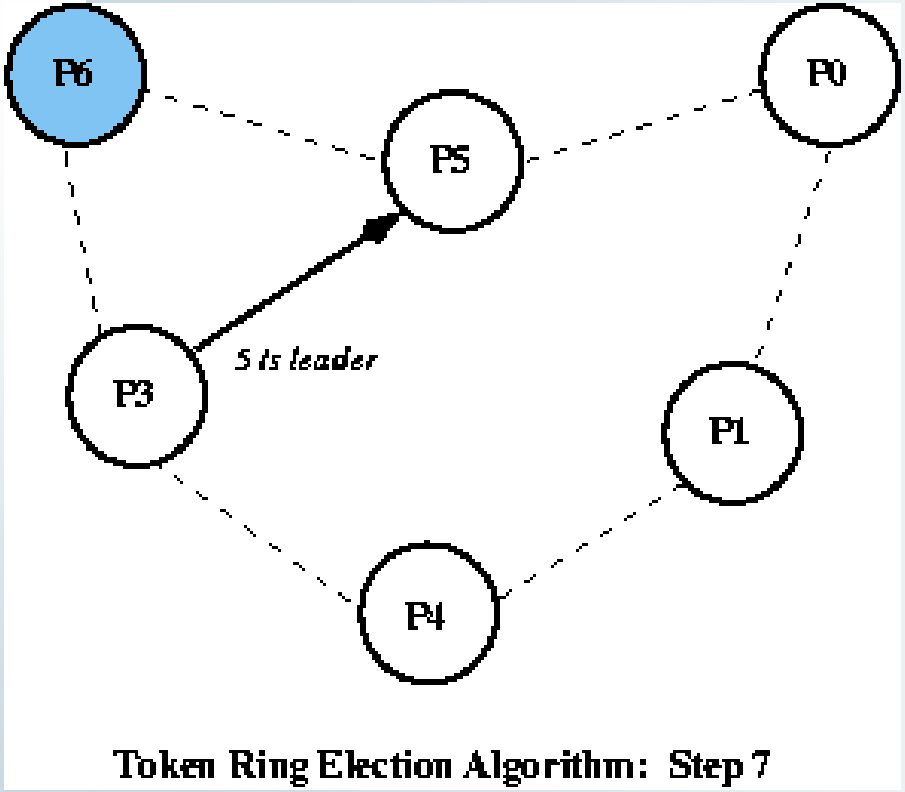
Process 1 passes the message on,
adding its own id to the message.

Ring Algorithm (continued)



Process 4 passes the message on, adding its own id to the message.

Ring Algorithm (continued)



Token Ring Election Algorithm: Step 7

When Process 3 receives the message back,

it knows the message has gone around the ring,

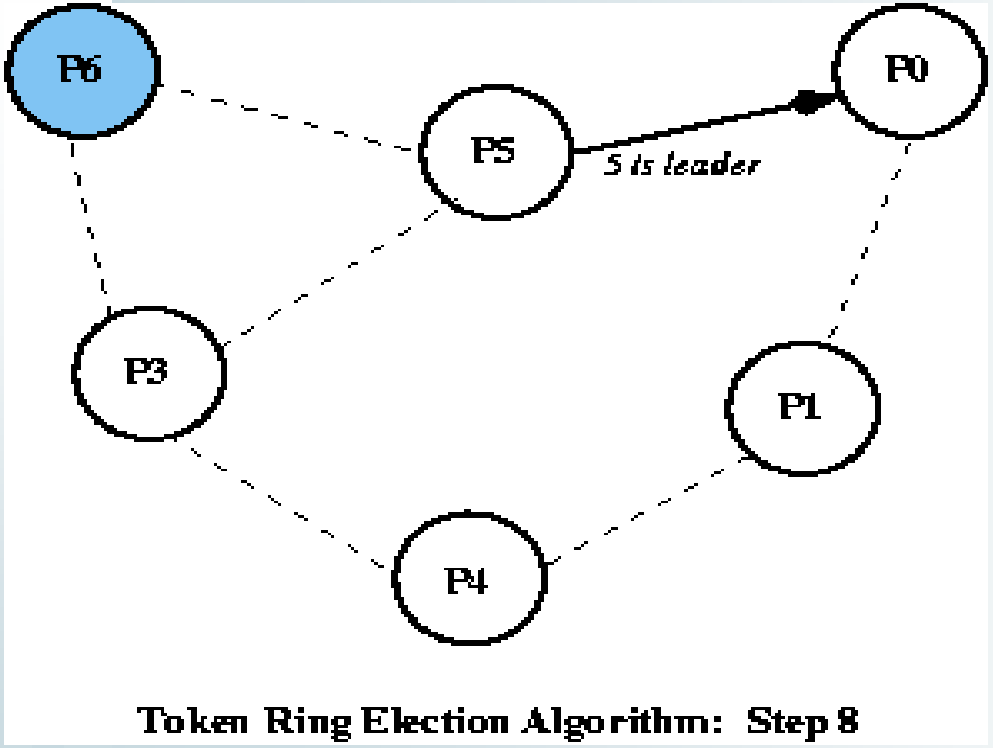
as its own id is in the list.

Picking the highest id in the list,

it starts the coordinator message

"5 is the leader" around the ring.

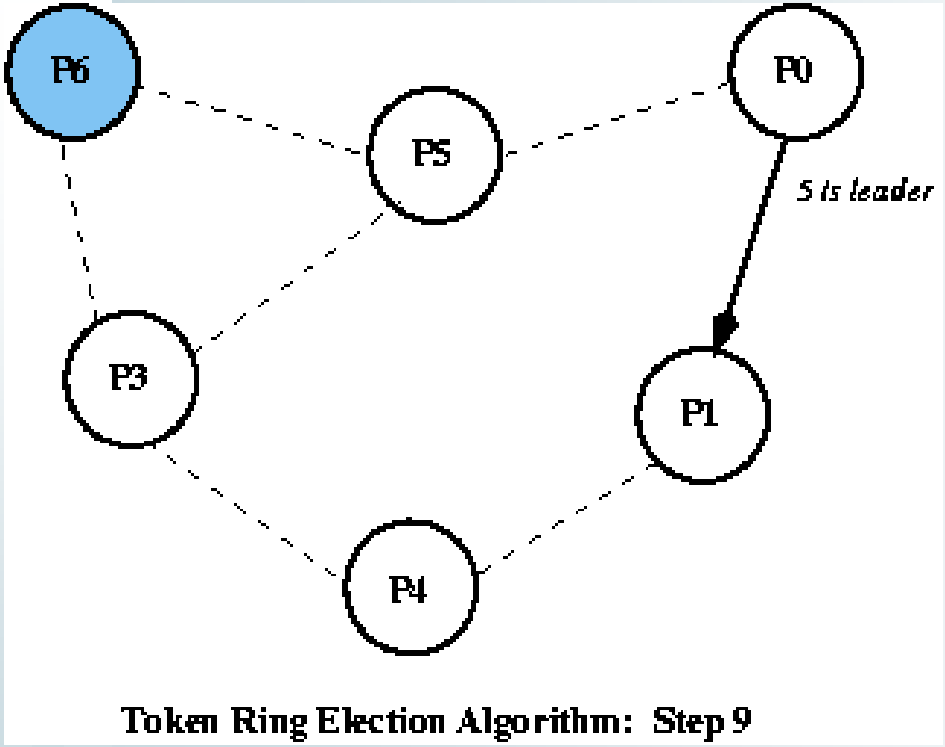
Ring Algorithm (continued)



Token Ring Election Algorithm: Step 8

Process 5 passes on the coordinator message.

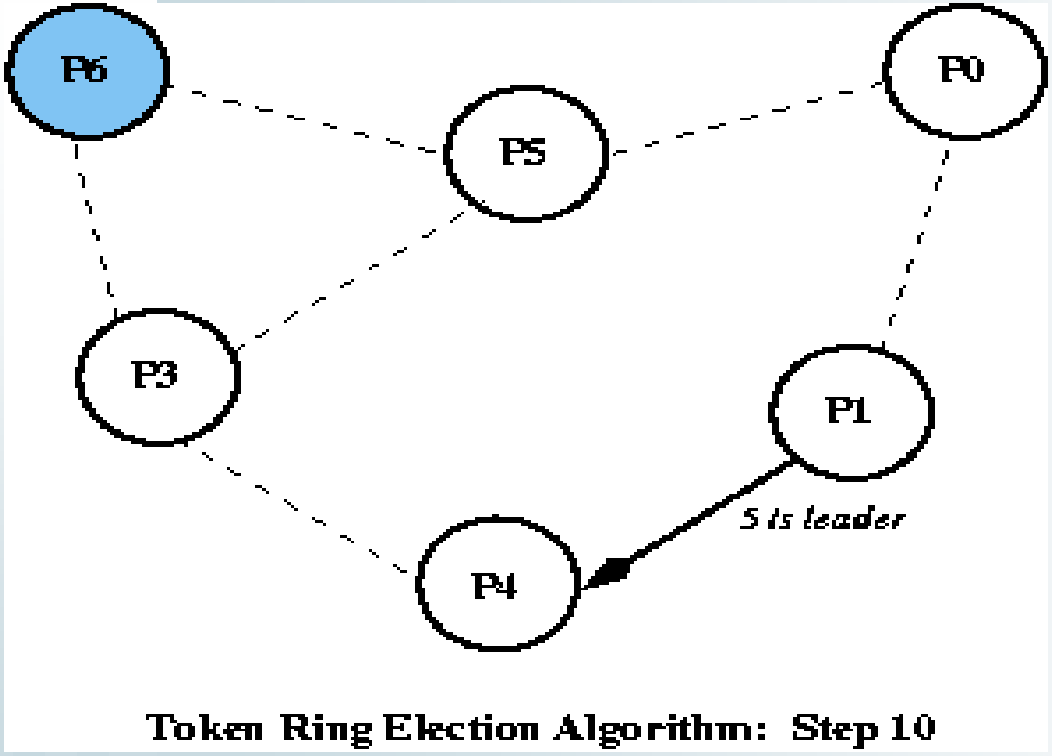
Ring Algorithm (continued)



Token Ring Election Algorithm: Step 9

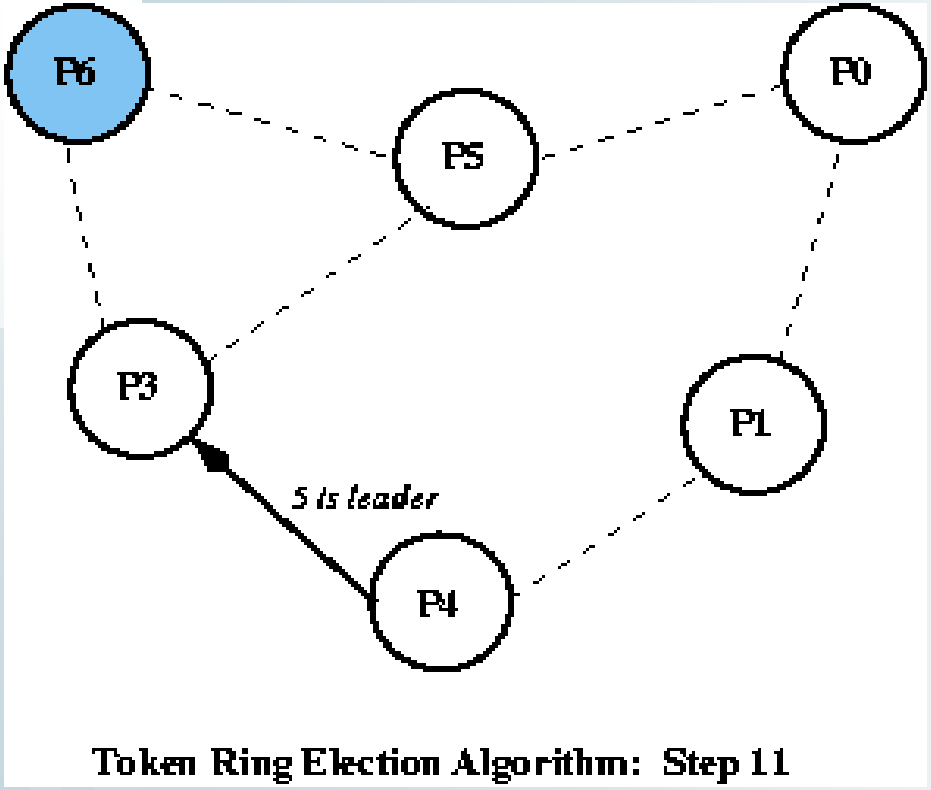
Process 0 passes on the coordinator message.

Ring Algorithm (continued)



Process 1 passes on the coordinator message.

Ring Algorithm (continued)



Token Ring Election Algorithm: Step 11

Process 4 passes on the coordinator message.

Process 3 receives the coordinator message, and stops it.

Ring Algorithm (concluded)

- Suppose *P* receives an election message with its own process number in body
  - Changes message to *coordinator* message, preserving body
  - All processes recognize *highest numbered process* as new coordinator
- If multiple messages circulate …
  - …they will all contain same list of processes (eventually)
- If process comes back on-line
  - Calls new election

Wireless Networks

- Different assumptions
  - Message passing is less reliable
  - Network topology constantly changing
- Expanding ring of broadcast
  - Election messages
  - Decision rules for when to yield
- Not very well developed.
  - Topic of current research

Very Large Scale Networks

- Sometimes more than one node should be selected
- Nodes organized as peers and *super-peers*
  - Elections held within each peer group
  - Super-peers coordinate among themselves