Distributed Algorithms (CAS 769)

Week 4: Leader Election

Dr. Borzoo Bonakdarpour

Department of Computing and Software
McMaster University

Acknowledgments

Most of the contents of these slides are obtained from the following:

- Distributed Algorithms: An Intuitive Approach - Wan Fokkink

Presentation outline

Leader Election

Election in Anonymous Networks

Leader Election

Goal

In an election algorithm, each computation should terminate in a configuration where one process is the leader.

The leader usually acts as the organizer of something or as an initiator of a centralized algorithm

Assumptions

- All processes have the same local algorithm.
- The algorithm is decentralized: The initiators can be any non-empty set of processes.
- Process id's are unique, and from a totally ordered set.

Election in Rings

We start with election algorithms in rings. In all these algorithms, the initiators determine among themselves which one has the highest ID.

Initially, the initiators are active and all nominators are passive.

Passive processes do not participate in the race to become a leader and simply pass the messages.

Chang-Roberts Algorithm

Consider a directed ring. The idea of the algorithm is that only the message with the highest id completes one round in the ring.

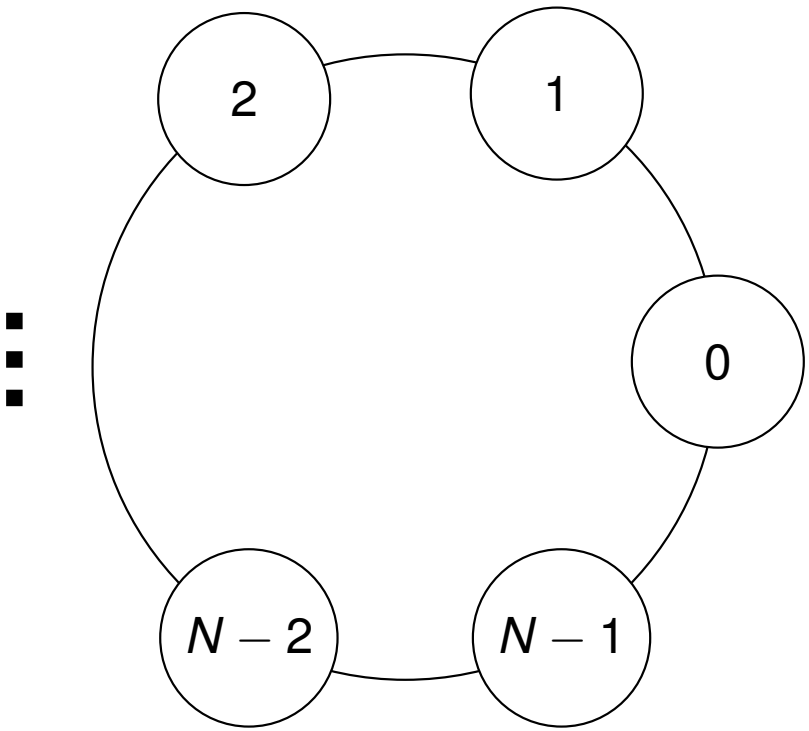
- Each initiator sends a message, tagged with its id.
- When p receives q with $q < p$, it purges the message.
- When p receives q with $q > p$, it becomes passive, and passes on the message.
- When p receives p, it becomes the leader.

Passive processes (including all non-initiators) pass on messages.

Worst-case message complexity: $O(N^2)$

Average-case message complexity: $O(N \log N)$

Chang-Roberts Algorithm - Example



- Clockwise:

$$\frac{N(N+1)}{2}$$

- Counter clockwise: $2N - 1$

Franklin's Algorithm

Franklin's algorithm aims at improving the message complexity of Chang-Roberts algorithm.

Each active process p repeatedly compares its own id with the id's of its nearest active neighbors on both sides.

If such a neighbor has a larger id, then *p* becomes passive.

This will avoid messages to travel unnecessarily.

Franklin's Algorithm

Consider an undirected ring.

- Initially, initiators are active, and non-initiators are passive. Each active process sends its id to its neighbors on either side.
- Let active process p receive q and r
  - if $\max\{q, r\} < p$, then p sends its id again
  - if $\max\{q, r\} > p$, then p becomes passive
  - if $\max\{q, r\} = p$, then p becomes the leader

Passive processes pass on incoming messages.

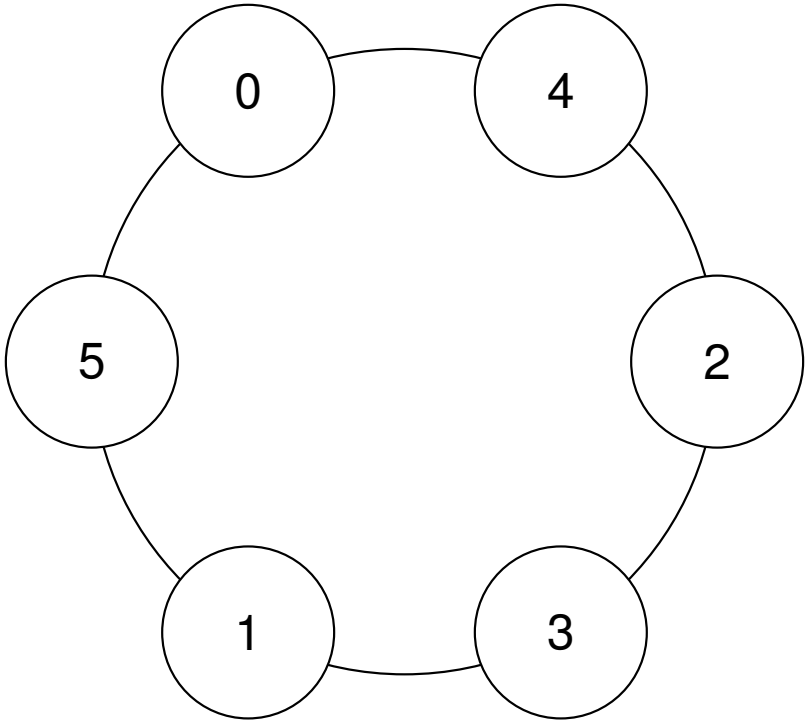
Franklin's Algorithm

Worst-case message complexity: $O(N \log N)$.

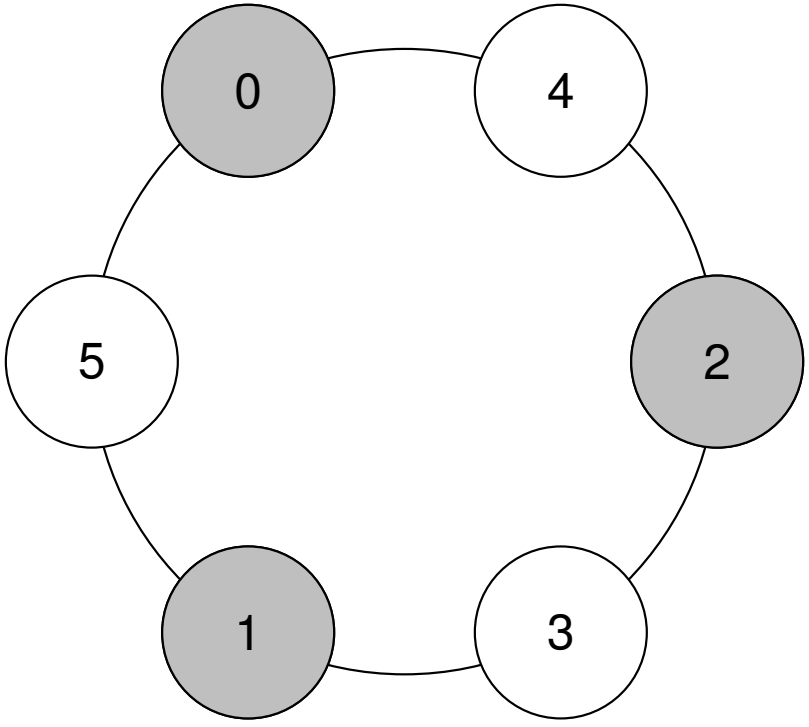
In each round, at least half of the active processes become passive. So there are at most $\log_2 N$ rounds.

Each round takes $2N$ messages.

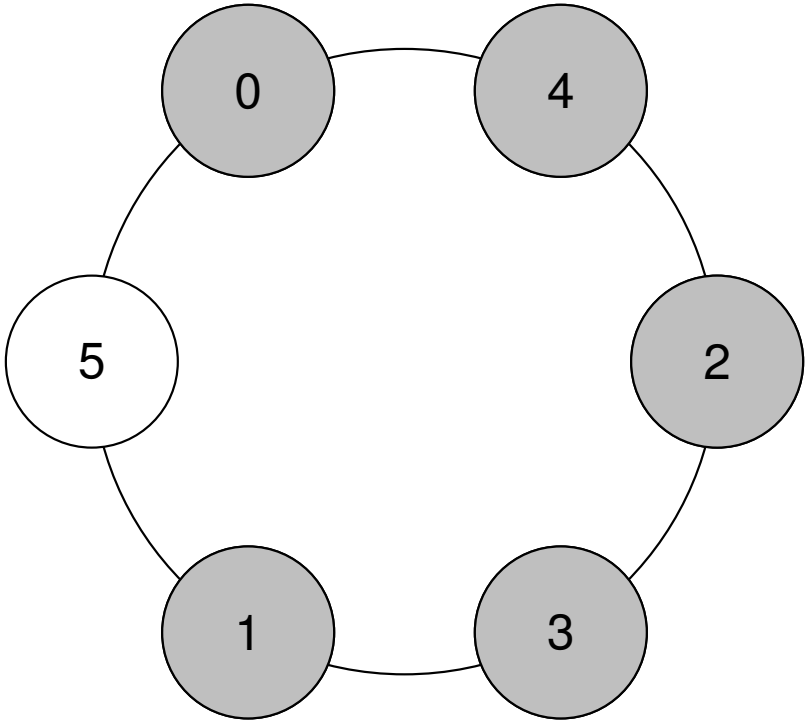
Franklin's Algorithm - Example



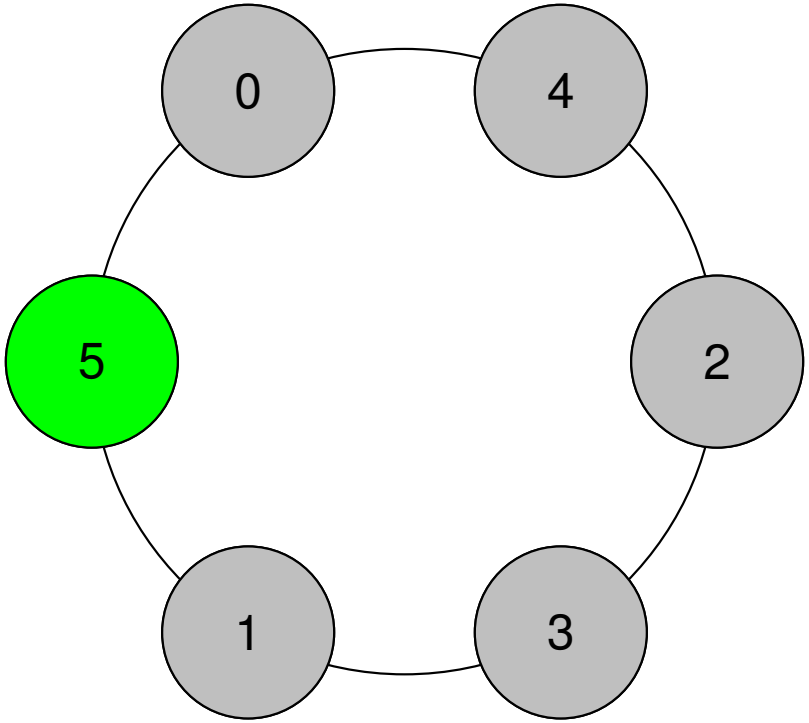
Franklin's Algorithm - Example



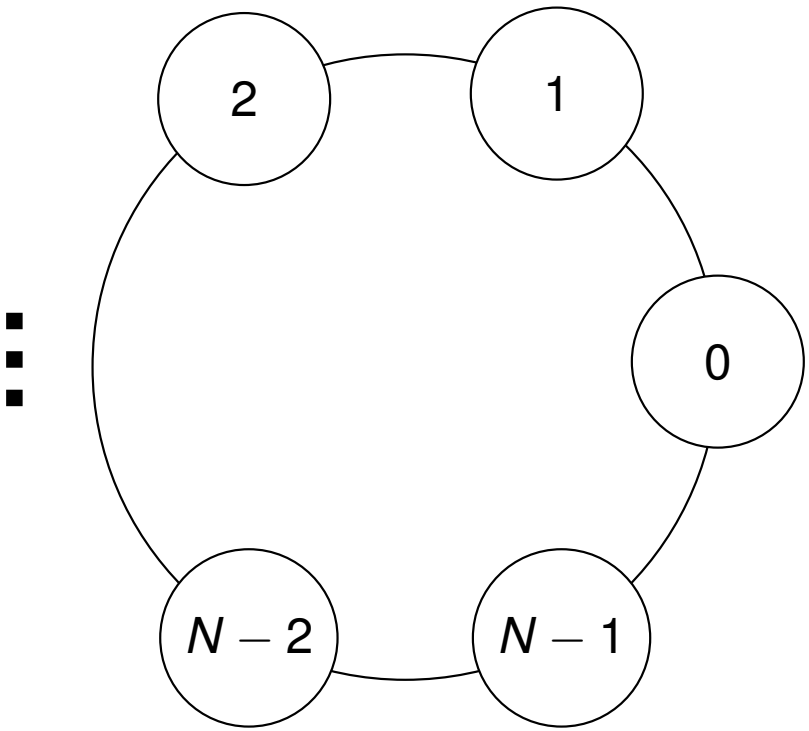
Franklin's Algorithm - Example



Franklin's Algorithm - Example



Franklin's Algorithm - Example



- after 1 round only node $N-1$ is active
- after 2 rounds node $N-1$ is the leader

Suppose this ring is directed with a clockwise orientation.

If a process would only compare its id with the one of its predecessor, then it would take N rounds to complete.

Dolev-Klawe-Rodeh Algorithm

Dolev-Klawe-Rodeh algorithm transposes the idea Franklin's algorithm to a directed ring.

Each active process p cannot easily compar its id with its nearest active neighbors q and r.

This is resolved by performing this comparison not at p, but at its next (in the direction of the ring) active neighbor r. I.e., the ids of p, q, and r are collected at r:

- If p is larger than q and r, then r remains active and progresses to the next round, where it assumes the id of r.
- If p is smaller than q or r, then r becomes passive.
- If p equals q and r, then r becomes the leader.

Dolev-Klawe-Rodeh's Algorithm - Details

Consider an directed ring.

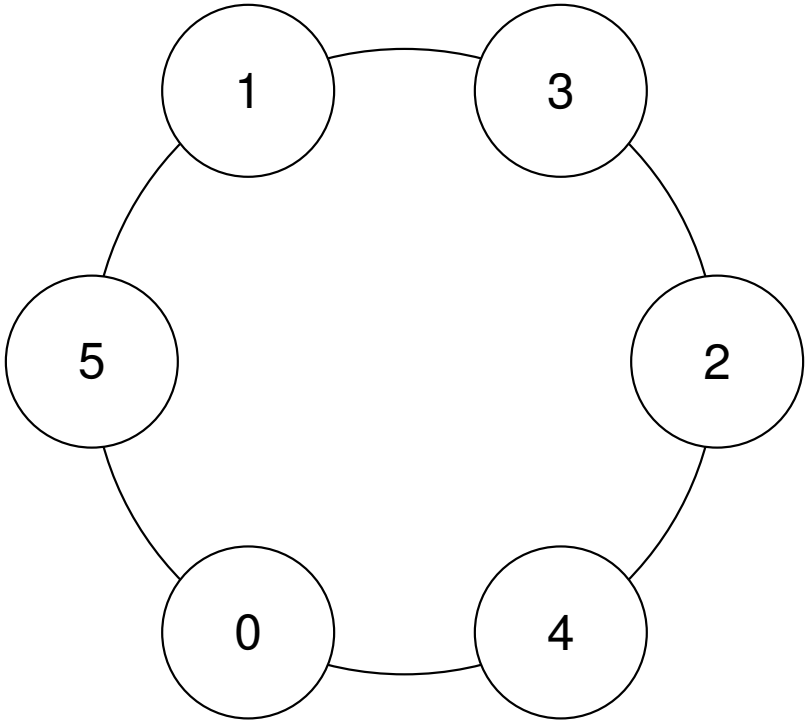
The comparison of id's of an active process p and its nearest active neighbors q and r is performed at r.

$$\longrightarrow s \longrightarrow q \longrightarrow p \longrightarrow r \longrightarrow t \longrightarrow$$

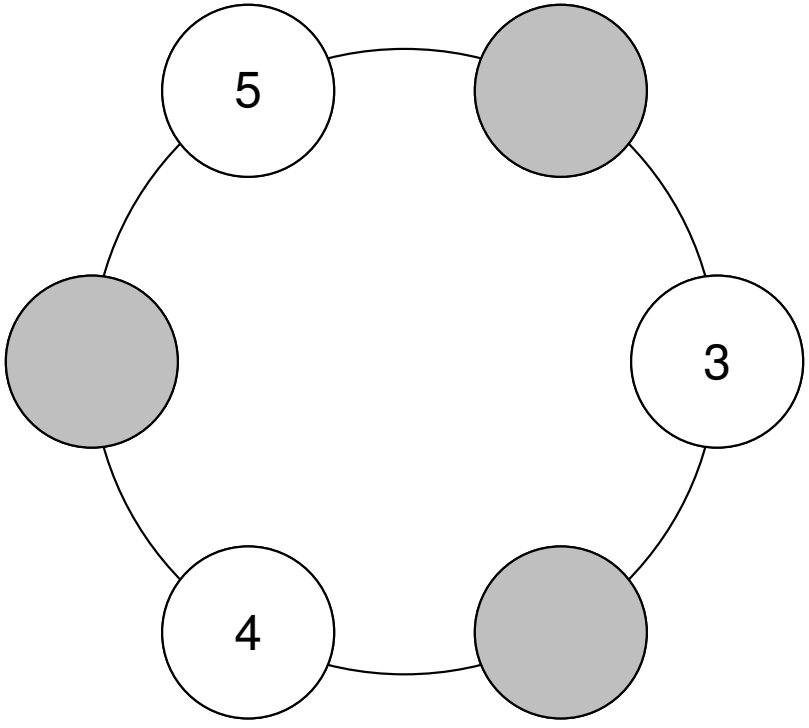
- If $\max\{q, r\} < p$, then r changes its id to p, and sends out p.
- If $\max\{q, r\} > p$, then r becomes passive.
- If $\max\{q, r\} = p$, then r announces this id to all processes. The process that originally had the id p becomes the leader.

Worst-case message complexity: $O(N \log N)$.

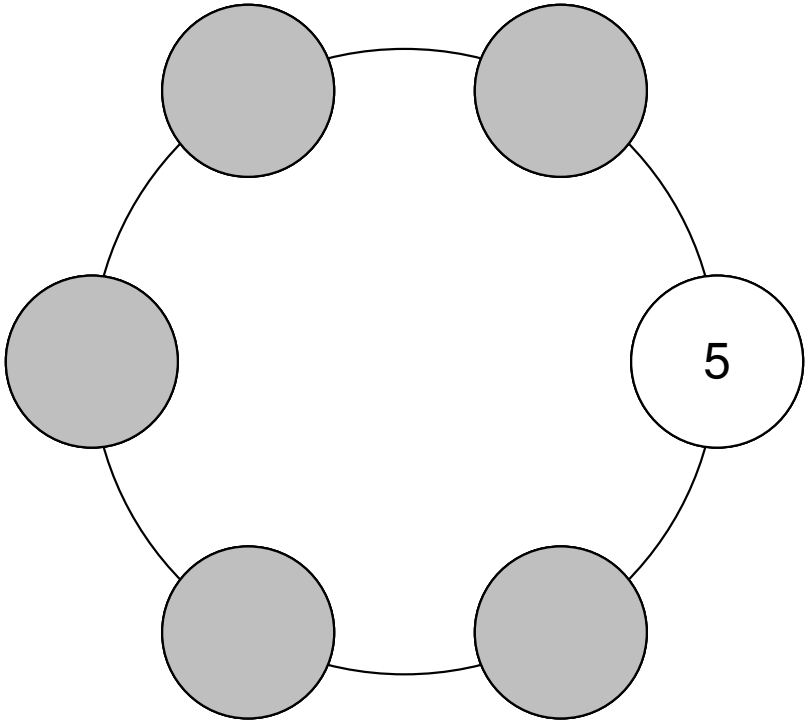
Dolev-Klawe-Rodeh Algorithm - Example



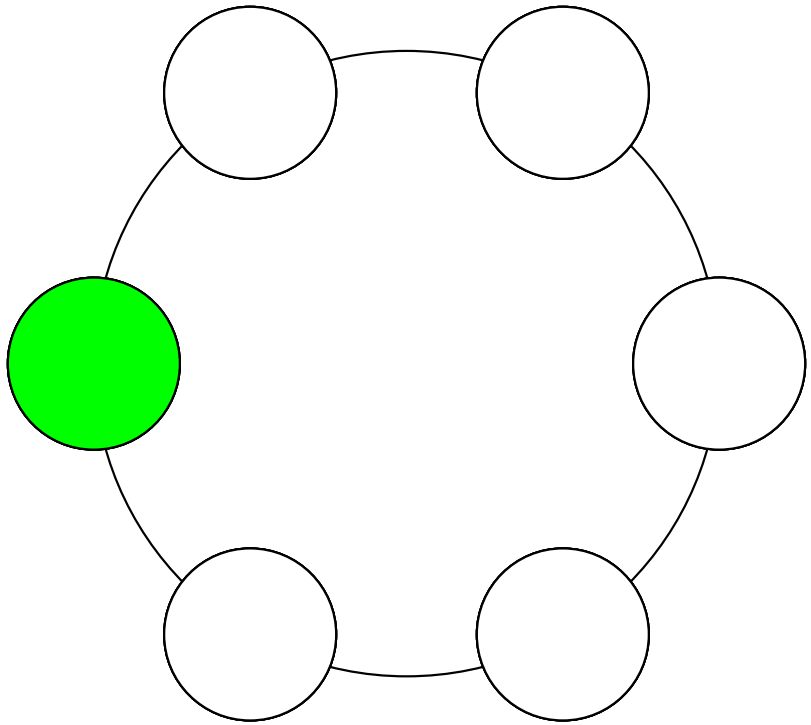
Dolev-Klawe-Rodeh Algorithm - Example



Dolev-Klawe-Rodeh Algorithm - Example



Dolev-Klawe-Rodeh Algorithm - Example



Leader Election in Trees

Given an undirected acyclic network.

The tree algorithm (from Week 2 graph algorithms) can be used as an election algorithm for undirected, acyclic networks

Let the two deciding processes determine which one of them becomes the leader (e.g. the one with the largest id).

The Idea

Each process *p* collects ids from its children, computes the maximum of these ids and its own, and passes on this maximum to its parent.

Later, *p* receives the overall maximum of the ids in the network from its parent, which *p* passes on to its children.

Leader Election in Trees - Wake-up phase

Start with a wake-up phase, driven by the initiators.

- Initially, initiators send a wake-up message to all neighbors.
- When a non-initiator receive a first wake-up message, it sends a wake-up message to all neighbors.
- A process wakes up when it has received wake-up messages from all neighbors.

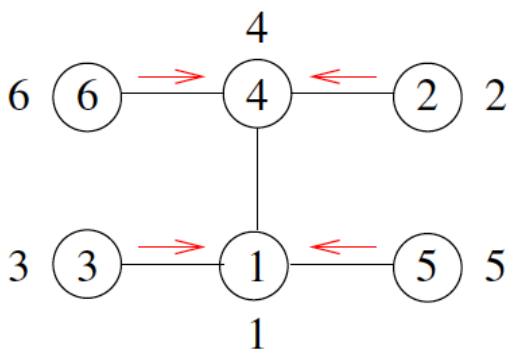
Leader Election in Trees - The Algorithm

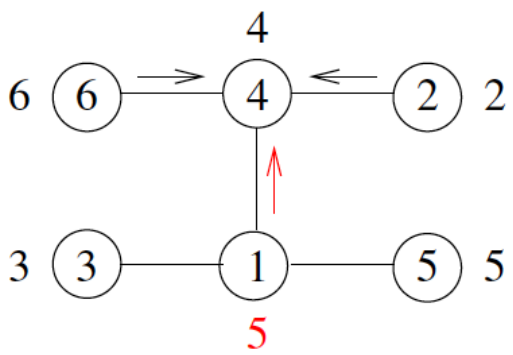
The local algorithm at an awake process p:

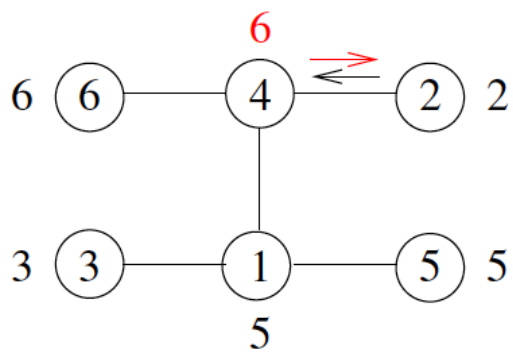
- p waits until it has received id's from all neighbors except one, which becomes its parent.
- p computes the largest id $\max_p$ among the received id's and its own id.
- p sends a parent request to its parent, tagged with $\max_p$.
- If p receives a parent request from its parent, tagged with q, it computes $\max'_p$, being the maximum of $\max_p$ and q.
- Next, p sends an information message to all neighbors except its parent, tagged with $\max'_p$.
- This information message is forwarded through the network.
- The process with id $\max'_p$ becomes the leader.

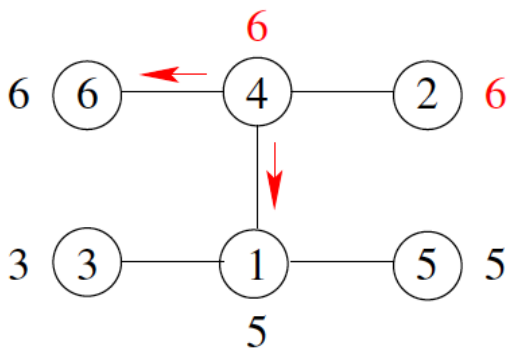
Message complexity: $2N - 2$ messages.

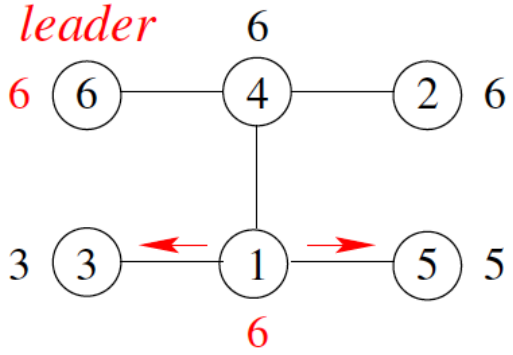
Leader Election in Trees - Example

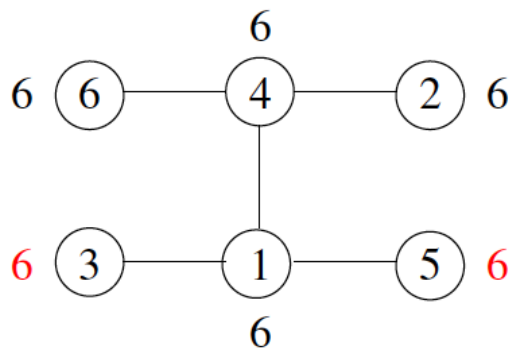












Echo Algorithm with Extinction

This is an election algorithm that works for any topology.

Idea

We let each process to start a run of the echo algorithm tagged with it id.

Only the wave started by the initiator with the largest id complete, after which the initiator become the leader.

Echo Algorithm with Extinction - Details

Election for undirected networks, based on the echo algorithm:

- Each initiator starts a wave, tagged with its id.
- At any time, each process takes part in at most one wave.
- Suppose a process p in wave q is hit by a wave r:
  - if $q < r$, then p changes to wave r (it abandons all earlier messages);
  - if $q > r$, then p continues with wave q (it purges the incoming message);
  - if $q = r$, then the incoming message is treated according to the echo algorithm of wave q.
- If wave p executes a decide event (at p), p becomes leader.

Non-initiators join the first wave that hits them.

Worst-case message complexity: $O(N.E)$

Presentation outline

Leader Election

Election in Anonymous Networks

Anonymous Networks

Processes may be anonymous for several reasons:

- Absence of unique hardware id's.
- Processes don't want to reveal their id (security protocols).
- Transmitting/storing id's is too expensive (IEEE 1394 bus).

Assumptions for election in anonymous networks:

- All processes have the same local algorithm.
- The initiators can be any non-empty set of processes.
- Processes (and channels) don't have a unique id.

If there is one leader, all processes can be given a unique id (using a traversal algorithm).

Election in Anonymous Networks

Impossibility Result

Theorem: There is no leader election algorithm for anonymous rings that always terminates.

A configuration is symmetric if all processes are in the same state and all channels carry the same messages.

Notice that, in a symmetric configuration there isn't one leader.
Proof: Consider an anonymous ring of size N. By contradiction, suppose there exists such an election algorithm.

- There is a symmetric initial configuration (all processes are in their initial configuration and all channels are empty).
- If γ_0 is symmetric and $\gamma_0 \rightarrow \gamma_1$, then there is an execution $\gamma_1 \rightarrow \gamma_2 \cdots \rightarrow \gamma_N$, where γ_N is symmetric.

Hence, the algorithm exhibits an infinite execution, where a symmetric configuration is reached infinitely often.

Election in Anonymous Networks

Fairness

An execution is strongly fair, if each event that is applicable in infinitely many configurations, occurs infinitely often in the execution.

An execution is weakly fair, if each event that is applicable continuously, eventually occurs in the execution.

Election algorithm for anonymous rings have fair infinite executions. This is because in transition $\gamma_0 \rightarrow \gamma_1$, it does not matter which event is used; we can always build the transition sequence to the symmetric configuration γ_N. This in turn means no event is ignored.

Hence, leader election in a fair anonymous ring is also impossible.

Election in Anonymous Networks

Question: What does the transition system of an election algorithm in an anonymous network look like?

It contains livelocks; i.e., cycles that prevent the algorithm to reach a terminating configuration.

Question: Is there any way to overcome this problem?

Probabilistic Algorithms

In a probabilistic algorithm, a process may flip a coin, and perform an event based on the outcome of this coin flip.

Probabilistic algorithms where all computations terminate in a correct configuration are not interesting: Letting the coin e.g. always flip heads yields a correct non-probabilistic algorithm.

Probabilistic Algorithms

Las Vegas Algorithms

A probabilistic algorithm is Las Vegas if:

- the probability that it terminates is greater than zero, and
- all terminal configurations are correct.

Question: When in a Las Vegas algorithm the probability of termination is one?

Monte Carlo Algorithms

An algorithm is Monte Carlo if:

- it always terminates, and
- the probability that a terminal configuration is correct is greater than zero.

Itai-Rodeh Election Algorithm

Assumptions

An anonymous, directed ring, where all processes know the ring size N.

Itai-Rodeh Election Algorithms is a Las Vegas algorithm that terminates with probability one. We adapt the Chang-Roberts algorithm: each initiator sends out an id, and the largest id is the only one making a round trip.

The Idea

Each initiator selects a random id from $\{1, \ldots, N\}$.
Challenge: Different processes may select the same id.

Solution: Each message is supplied with a hop count. A message arriving at its source has hop count N.

If several processes select the same largest id, they will start a new election round, with a higher number.

Itai-Rodeh Election Algorithm

Initially, initiators are active in round 0, and non-initiators are passive.

Let p be active. At the start of an election round n, p selects a random id_p, sends $(n, id_p, 1, false)$, and waits for a message (n', i, h, b). The 3rd value is the hop count. The 4th value signals if another process with the same id was encountered during the round trip.

- p gets (n', i, h, b) with $n' > n$, or $n' = n$ and $i > id_p$: it becomes passive and sends $(n', i, h + 1, b)$.
- p gets (n', i, h, b) with $n' < n$, or $n' = n$ and $i < id_p$: it purges the message.
- p gets (n, id_p, h, b) with $h < N$: it sends $(n, id_p; h + 1, true)$.
- p gets $(n, id_p, N, true)$: it proceeds to round $n + 1$.
- p gets $(n, id_p, N, false)$: it becomes the leader.

Passive processes pass on messages, increasing their hop count by one.

Itai-Rodeh Election Algorithm

Correctness: The Itai-Rodeh election algorithm is Las Vegas, where eventually one leader is elected, with probability 1.
Why?

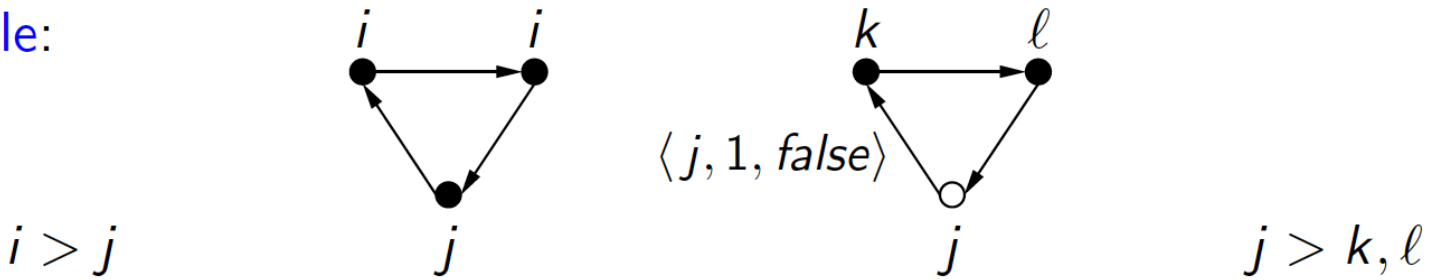
Average-case message complexity: $O(N \log N)$.

Lower bound: $\Omega(N \log N)$.

Itai-Rodeh Election Algorithm - Example

Without rounds, the algorithm would be flawed (for non-FIFO channels).

Example:



Election in Arbitrary Anonymous Networks

The echo algorithm with extinction, with random selection of id's, can be used for election in anonymous undirected networks in which all processes know the network size.

The Idea

- The algorithm works in rounds and round number is used to recognize messages from previous rounds.
- Each process randomly selects an id and runs the echo algorithm.
- When a process is hit by a wave with a higher round number than it current wave, or with the same round number but a higher id, the process becomes passive and moves to the other wave.

Election in Arbitrary Anonymous Networks

Algorithm Details

Initially, initiators are active in round 0, and non-initiators are passive.

Each active process selects a random $id \in \{0, \ldots, N\}$, and starts a wave, tagged with its id and round number 0.
Let process p in wave i of round n be hit by wave j of round n':

- if $n' > n$, or $n' = n$ and $j > i$, then p adopts wave j of round n', and treats the message according to the echo algorithm
- if $n' < n$, or $n' = n$ and $j < i$, then p purges the message
- if $n' = n$ and $j = i$, then p treats the message according to the echo algorithm

Election in Arbitrary Anonymous Networks

Processes need to know the size of the network to be able to determine at completion of their wave whether it has covered the entire network. This is because different wave with different ids can collide which cause pre-mature completion.

Thus, each message sent upwards in the constructed tree reports the size of its subtree.

All other messages report 0.

When a process decides, it computes the size of the constructed tree.

If the constructed tree covers the network, it becomes the leader.

Else, it selects a new id, and initiates a new wave, in the next round.

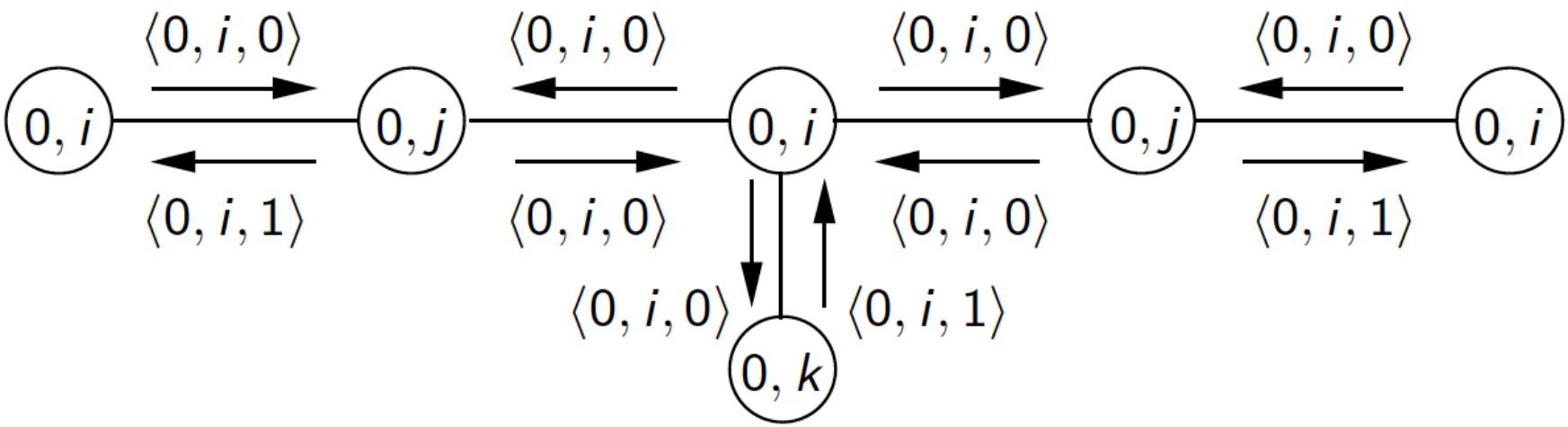
Election in Arbitrary Anonymous Networks

The echo algorithm with extinction, with random selection of id's is a Las Vegas algorithm that terminates with probability one.

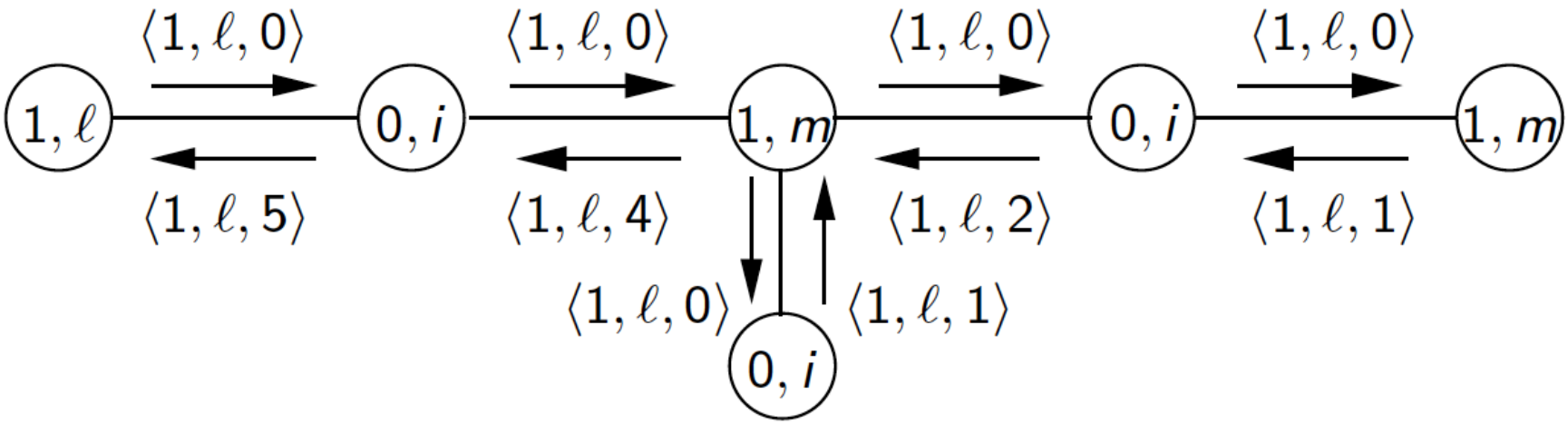
Election in Arbitrary Anonymous Networks - Example

Only waves that complete are shown.

$i > j > k > l > m$



Election in Arbitrary Anonymous Networks - Example



The process at the left computes size 6, and becomes the leader.

Computing the Size of an Anonymous Network

Impossibility Result

Theorem: There is no Las Vegas algorithm to compute the size of an anonymous ring.

This implies that there is no Las Vegas algorithm for election in an anonymous ring if processes don't know the ring size.

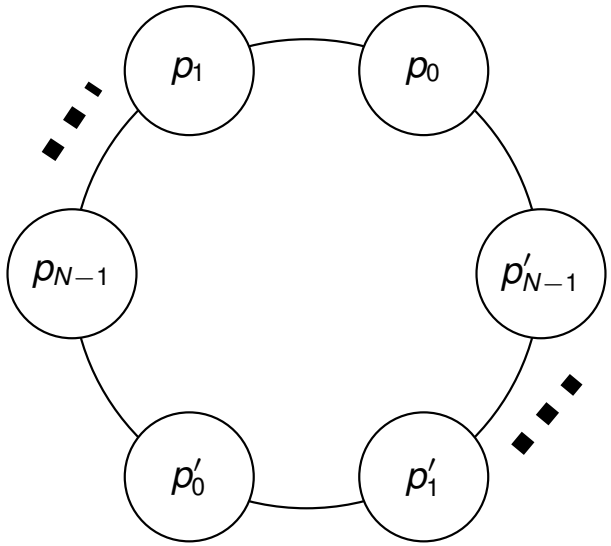
Because when a leader is known, the network size can be computed using a wave algorithm.

Computing the Size of an Anonymous Network

Proof: Consider an anonymous, directed ring $p_0, \ldots, p_{N-1}$.

Assume a probabilistic algorithm with a computation C that terminates with the correct outcome N.

Consider the ring $p_0, \ldots, p_{2N-1}$ build as follows where each p'_i is a copy of p_i:



Let each event at a process p_i in C be executed concurrently at p_i and p_{i+N}. This computation terminates with the incorrect outcome N.

Itai-Rodeh Ring Size Algorithm

Each process p maintains an estimate est_p of the ring size. Initially $est_p = 2$. (Always $est_p \leq N$.)

p initiates an estimate round (1) at the start of the algorithm, and (2) at each update of est_p.

Each round, p selects a random id_p in $\{1, \ldots, R\}$, sends $(est_p, id_p, 1)$, and waits for a message (est, id, h). (Always $h \leq est$.)

- If $est < est_p$, then p purges the message.
- Let $est > est_p$.
  - If $h < est$, then p sends $(est, id, h+1)$, and $est_p \leftarrow est$
  - If $h = est$, then $est_p \leftarrow est + 1$.
- Let $est = est_p$.
  - If $h < est$, then p sends $(est, id, h+1)$.
  - If $h = est$
    - If $id \neq id_p$, then $est_p \leftarrow est + 1$.
    - If $id = id_p$, then p purges the message (possibly its own message returned).

Itai-Rodeh Ring Size Algorithm

When the algorithm terminates, $est_p \leq N$ for all p. Also, esp_p converges to the same value for all p. Why?

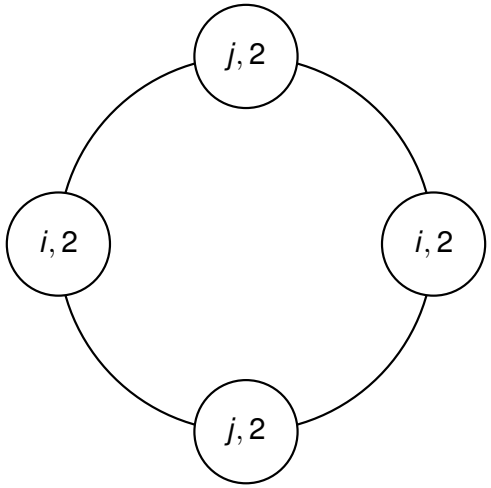
Question Under what condition the Itai-Rodeh algorithm terminates with an estimate smaller than N?

Itai-Rodeh Ring Size Algorithm

When the algorithm terminates, $est_p \leq N$ for all p. Also, esp_p converges to the same value for all p. Why?

Question Under what condition the Itai-Rodeh algorithm terminates with an estimate smaller than N?

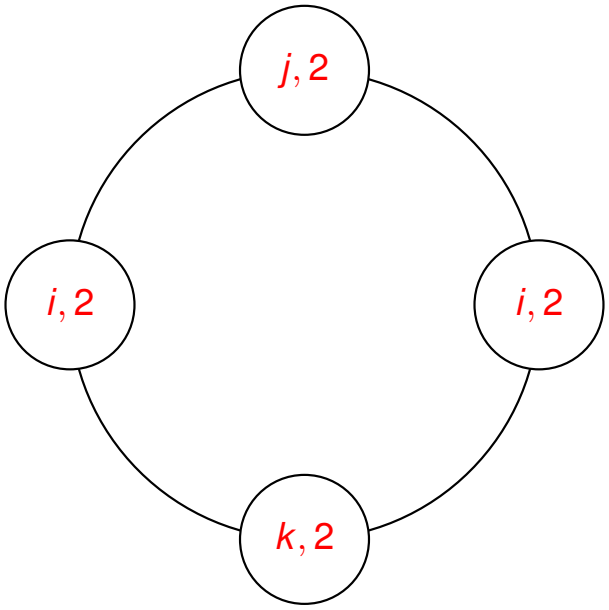
This happens when in a round with an estimate $est < N$, all processes at distance est from each other happen to select the same id.



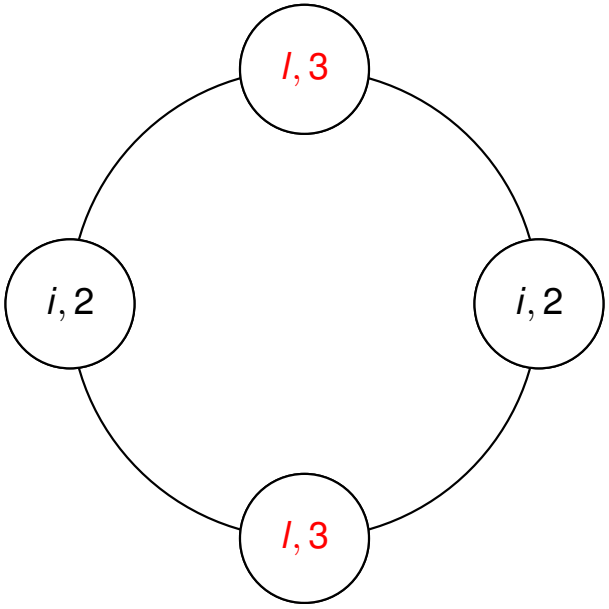
The algorithm will terminate with estimate of 2.

The probability that the algorithm terminates with an incorrect outcome tends to zero when R tends to infinity. (recall that $\{1, \ldots, R\}$ was the domain of ids)

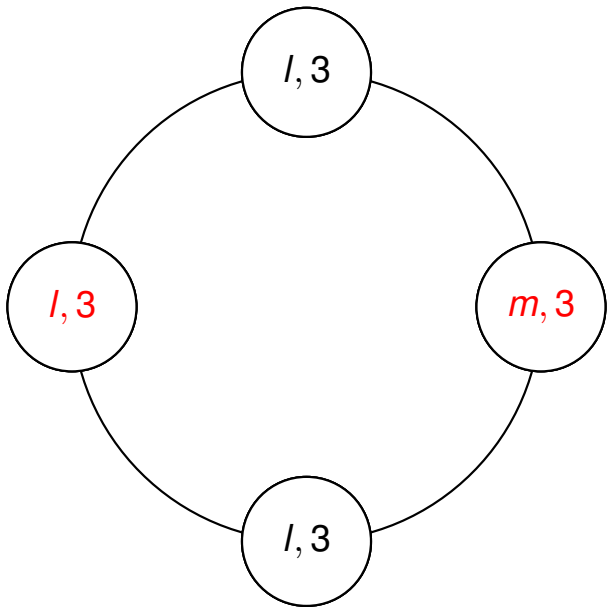
Itai-Rodeh Ring Size Algorithm - Example



Itai-Rodeh Ring Size Algorithm - Example



Itai-Rodeh Ring Size Algorithm - Example



Itai-Rodeh Ring Size Algorithm - Example