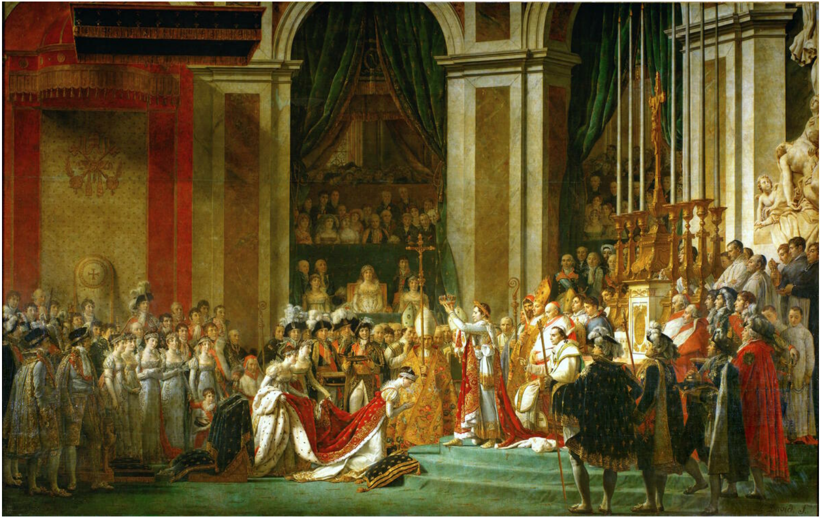
CPSC 465/565 Theory of Distributed Systems

James Aspnes

2023-09-06

Today's exciting topic

Leader election, mostly in rings



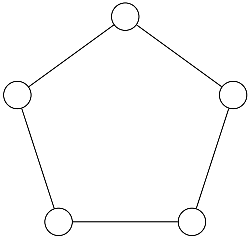
Motivation: where does initiator come form?

- Broadcast starts with an initiator.
- Where does the initiator come from?

Leader election: Protocol where one and only one process declares itself leader.

No requirement that losers learn who the leader is, but leader can always broadcast victory message.

Impossibility of leader election



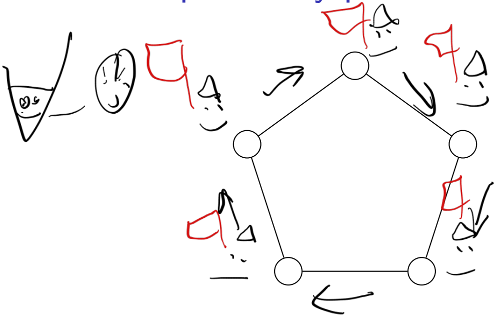
Leader election is *impossible* assuming

- Deterministic algorithm.
- Anonymous processes.
- Symmetric network.

(Angluin 1980)

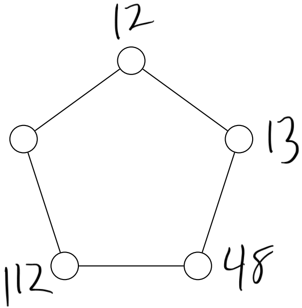
We'll use rings (cycles) as a test case since they are very symmetric.

Leader election impossibility proof



- Adversary chooses synchronous execution
- In initial configuration:
  - All processes have same state (anonymity)
  - $\Rightarrow$ All processes send same messages (determinism)
  - $\Rightarrow$ All processes receive same messages (symmetry)
  - $\Rightarrow$ All processes get same new state (determinism)
- By induction, maintain symmetry forever.
- $\Rightarrow$ If any process says it's leader, all processes do.

How to escape impossibility?



Need to break symmetry!

- Drop anonymity by giving processes ids.
- Or drop determinism by allowing randomness.

Most common approach is to use ids and elect max id.

Le Lann-Chang-Roberts (LCR)

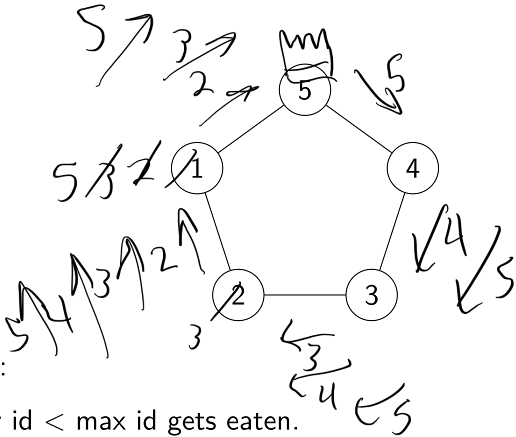
(Le Lann 1977, Chang-Roberts 1979)

```
initially do
    leader ← false
    maxId ← id_i
    send id_i to clockwise neighbor
upon receiving j from i − 1 do
    if j = id_i then
        leader ← true
    if j > maxId then
        maxId ← j
        send j to clockwise neighbor
```

Notes:

- We distinguish process position i from id_i.
- All arithmetic on positions is mod n.

Typical execution of LCR



Intuition:

- Any id $<$ max id gets eaten.
- Eventually max id goes all the way around.

Correctness of LCR: safety

Let ℓ be process with maximum id_ℓ.

Then either:

1. No $b_{j,j+1}$ contains id_ℓ, and
   1.1 For all messages m in transit, $m < \mathrm{id}_\ell$.
   1.2 For all i, $\mathrm{maxId}_i = \mathrm{id}_\ell$.
   1.3 $\mathrm{leader}_\ell = \textbf{true}$.
   1.4 For all $i \neq \ell$, $\mathrm{leader}_\ell = \textbf{false}$.
2. Exactly one $b_{j,j+1}$ contains id_ℓ, and
   2.1 For all messages m in transit, $m \leq \mathrm{id}_\ell$.
   2.2 For all $i \in [m, j]$, $\mathrm{maxId}_i = \mathrm{id}_\ell$.
   2.3 For all $i \in [j+1, m-1]$, $\mathrm{maxId}_i < \mathrm{id}_\ell$.
   2.4 For all i, $\mathrm{leader}_i = \textbf{false}$.

Essentially this just encodes intuition about reachable configurations.

Correctness of LCR: liveness

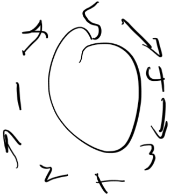
Let ℓ be process with maximum id_ℓ.

Then we can prove by induction on clockwise distance from ℓ:

1. Eventually, every process sends id_ℓ.
2. Eventually, every process receives id_ℓ.

This means that eventually ℓ receives id_ℓ and sets leader_ℓ to **true**.

Complexity of LCR



$\sum_{i=1}^{n} i = \Theta(n^2)$

Message complexity:

- $O(n^2)$ since each id is forwarded at most once per process.
- $\Omega(n^2)$ in synchronous execution if ids increase clockwise.
- $\Rightarrow \Theta(n^2)$ in worst case.

Time complexity:

- Exactly n from liveness induction.
- $(+n$ for optional victory broadcast.)

It's hard to see how to use less time, but maybe we can use fewer messages.

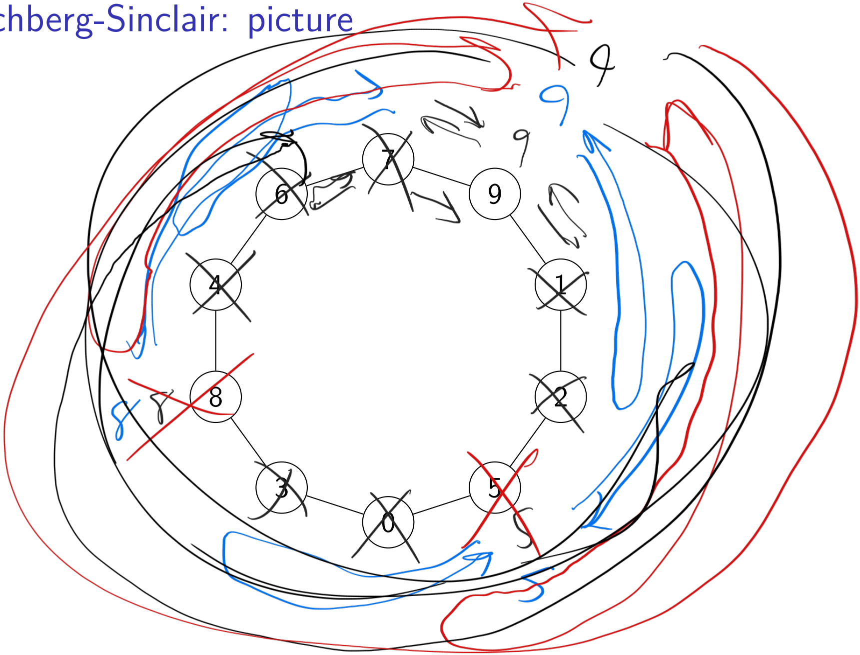
Hirschberg-Sinclair (1980)

Idea: Replace global probe to see if my id is max by local probes.

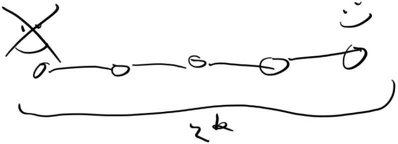
Probing scheme for one process:

1. Start as candidate leader.
2. In phase $k \in \{0, \ldots, \lceil \lg n \rceil\}$:
   - Send probe message 2^k hops in both directions.
   - Probe is eaten by nodes with higher id.
   - If probe not eaten, gets sent back.
3. If probe makes it all the way around, I win!

Hirschberg-Sinclair: picture



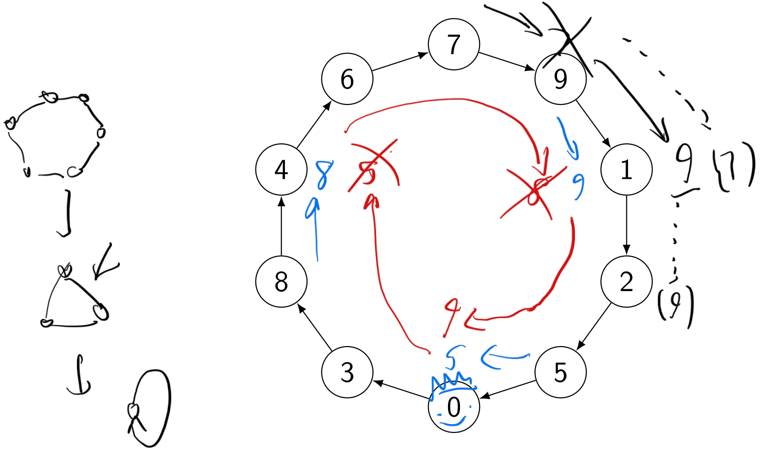
Hirschberg-Sinclair: complexity



- I finish phase k only if no node in range $[i - 2^k, i + 2^k]$ has larger id.
- $\Rightarrow$ if i, i' within 2^k, at most one finishes phase k.
- $\Rightarrow$ at most $n/(2^{k-1} + 1)$ nodes execute phase k.
- $\Rightarrow$ total messages in phase $k \leq \frac{n}{2^{k-1}+1} \cdot 2^k \cdot 4 < 8n$.
- $\Rightarrow$ total messages in all phases $= O(n \log n)$.

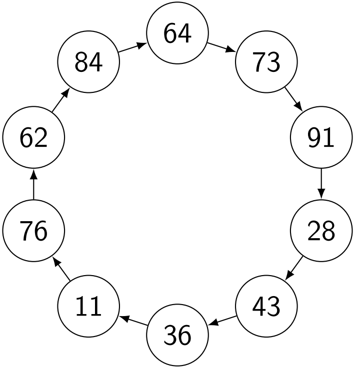
Gives $O(n \log n)$ messages and $O(n)$ time in two-way ring.

Peterson's algorithm for one-way ring



- Each candidate moves to next candidate position.
- Also sends value to next position after that.
- $\geq 1/2$ of candidates drop out in each phase.
- $\Rightarrow O(n \log n)$ messages.

Randomized LCR



1. Pick a random id for each node from range $\gg n^2$.
2. Run LCR.

The k-th largest id goes through $\leq n/k$ nodes on average.

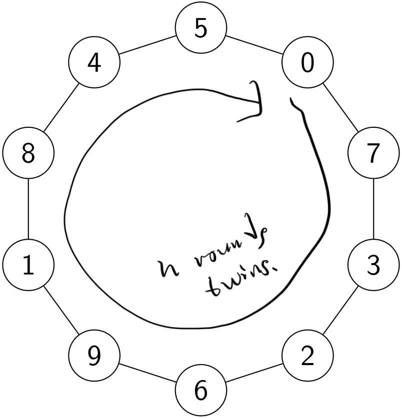
$\mathrm{E}\left[\text{total messages}\right] \leq \sum_{k=1}^{n} \frac{n}{k} = n \sum_{k=1}^{n} \frac{1}{k} = nH_n = \Theta(n \log n)$.

Small chance of failure if range too small; also requires knowing n.

Lower bound on messages?

Many $\Theta(n \log n)$-message algorithms. Maybe it's best possible?

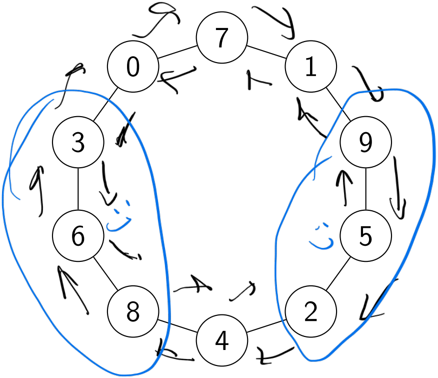
A perverse synchronous algorithm



- Run LCR where *minimum* id wins.
- Have process i wait until round $n \cdot \mathrm{id}_i$ to start.

Exactly n messages in every execution, but unbounded time.

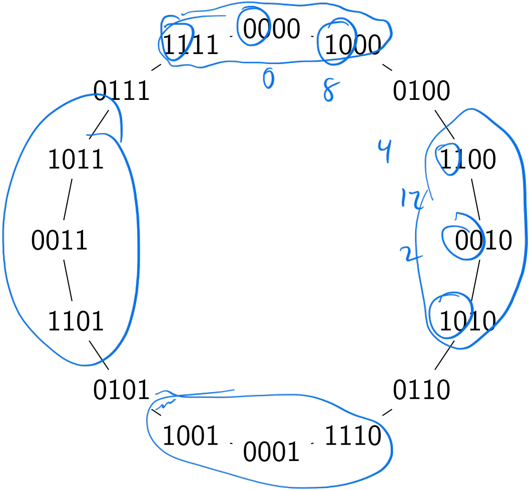
Frederickson-Lynch (1987)



Assumption: Synchronous **comparison-based** algorithm.

1. Comparison-based = *can't* evaluate ids but *can* test $\mathrm{id}_i < \mathrm{id}_j$.
2. **Effective round** = at least one message sent.
3. After k effective rounds, I learn ids within $\leq k$ of me.
4. If my k-neighborhood is ordered like your k-neighborhood, I send if you send!

Bit-reversal graph



Any node has $\Omega(n/k)$ order-equivalent k-neighborhoods, so:

1. $\Omega(n/k)$ messages sent in k-th effective round.
2. No unique leader until $k = \Omega(n)$.

Frederickson-Lynch continued

Total messages $= \sum_{k=1}^{\Omega(n)} \Omega(n/k) = \Omega(n \log n)$.

Can we drop comparison-based assumption?

- Alternative assumptions:
  1. Deterministic **time-bounded** algorithm.
  2. No knowledge of n (**uniform**).
  3. Unbounded ids.
- Allows **Ramsey theory** argument:
  1. Infinitely many id sequences in k-neighborhood.
  2. Finitely many possible bounded-time message patterns.
  3. $\Rightarrow$ Infinitely many id sequences give same pattern.

Repeat symmetry argument using message-pattern-equivalent id sequences instead of order-equivalent id sequences.

Burns (1980)

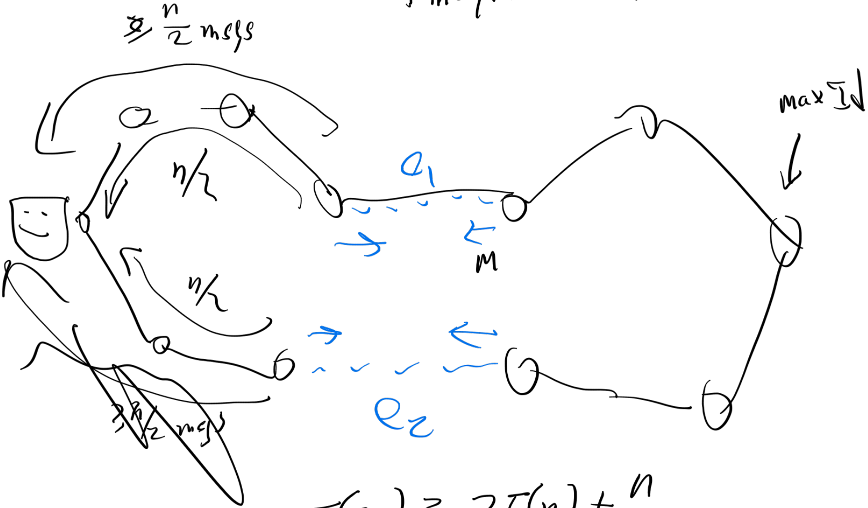
- $\Omega(n \log n)$ messages for asynchronous uniform algorithms.
- No time bound needed.

Burns: Proof outline

- Argue leader election $\equiv$ everybody learns max id (within $\pm\Theta(n)$ messages).
- Define **open execution** of size n as execution that delivers no messages across some edge e.
- Observe this is indistinguishable from execution on size-$2n$ ring with two missing edges e_1 and e_2.
- Each size-n execution uses $\geq T(n)$ messages (ind. hyp.).
- Combined execution uses $\geq 2T(n)$ messages without delivering across e_1 or e_2.
- Show delivering across one of e_1 or e_2 costs at least $n/2$ extra messages, while still being open since we didn't use one of the edges.
- This gives $T(2n) \geq 2T(n) + n/2 \Rightarrow T(n) = \Omega(n \log n)$.

Burns: Induction step

Imagine letting all msgs through



$$T(2n) \geq 2T(n) + \frac{n}{2}$$

open

Leader election in general graphs

- Simple LCR-style algorithm:
  - Everybody starts broadcast+convergecast with their id.
  - Only respond to convergecast if my id $<$ yours.
  - Only max id node finishes convergecast $\Rightarrow$ leader.
  - High message complexity!
- Afek-Gafni (1991):
  - Coalesce increasingly large neighborhoods.
  - Gets $O(n \log n)$ messages.
  - See notes for reference.