LCR Algorithm for Leader Election in Unidirectional Rings

- Value of $n$ unknown
- Leader performs output
- Each process has its unique id
LCR Algorithm
(by Le Lann, Chang and Roberts)

- Each process sends its id around the ring (relay id) (smallest id is winner)
- On receiving an id:
  - Compare with own id
  - If incoming id > own id, discard message
  - Else if incoming id < own id, pass id to next process
  - Else (incoming id = own id) declare self as leader
Leader needs to let others know when to stop.
Analysis

- **Termination**
  - When U(min) declares itself the leader (see below), it sends an announcement around ring; everyone knows

- **Correctness**
  - Lemma: After r rounds, r processes will know the id of U(min); Proof by induction
  - Lemma: After n rounds, one leader is elected

- **Complexity**
  - Time: 2n rounds
  - Message: O(n*n)
    - O(n) message per round (worst case)
    - n rounds
Better message complexity?  
(Hirschberg and Sinclair Algorithm)

- Bidirectional Rings; n can be unknown
- Several Phases
- At beginning of a phase, each active process initiates a token (has its id) and sends to both neighbors
- If both tokens come back, process proceeds to next phase
- Else a process becomes passive (cannot originate tokens cannot become a leader but forwards tokens)
At Phase $l$

- Phase $l$ consists of $2 \times 2^l$ phases
- Active process originates 2 tokens (sends in both directions)
  - token has id of originator, # of hops
- Token traverses $2^l$ hops.
  - If token is not discarded, it is returned to sender after it traverses $2^l$ hops.
- If both tokens return
  - Go to next phase
- Else become passive
Token (in forward phase) with id $u(i)$ reaches process with id $u(j)$

- If $u(i) = u(j)$, process $j$ (with id $u(j)$) is leader
- If $u(i) < u(j)$, process $j$ forwards token
  - (after decrementing hop count)
- Else discard token
- If hop count of token = 1, return token (in return phase)
Proof and Complexity

- **Proof: Exercise**
- **Message complexity**
  - Phase 0: Each process (originates one token)
    - Sends in both directions and may return in both directions (worst case)
    - 4 messages/process; 4n messages for phase 0
Phase $l$

- A process enters phase $l$ only after receiving both tokens sent in phase $l-1$ (which would have traversed $2^{(l-1)}$ hops in both directions)
- No other process among the $(2^{(l-1)}+1)$ consecutive processes can be active in phase $l$
- Max # of processes active in phase $l = n/(1+2^{(l-1)})$
- Each process in phase $l$ is charged $4*2^{l}$ messages
Message Complexity

- Total phase $l$ messages = 
  $4.2^l \cdot \left\lfloor \frac{n}{1 + 2^{(l-1)}} \right\rfloor \leq 8n$
- Total number of phases: $1 + \text{ceiling}(\log n)$
- Total number of messages: $O(n \log n)$
- Can we improve this further?
- Time complexity? $O(n)$
Non-comparison based algorithms

- Less than $O(n \log n)$ messages
- Time slice algorithm: ids are positive integers, $n$ is known
- Proceeds in phases, 1, 2, ...
- Each phase consists of $n$ rounds
- During beginning of phase $v$
  - process with id $v$ sends a token with $v$ around the ring if it has not received a token till that time
Time Slice Algorithm

- Process with Umin is elected leader; all others terminate after round Umin*n rounds
- Communication complexity:
  - n messages
- Time complexity
  - n.Umin rounds
- What if process with max id is to be elected?
  - 2n more messages/rounds
Variable Speeds Algorithm

- Unidirectional ring; n is unknown
- Process i with id Ui sends a token with Ui. This token travels one hop for each 2*Ui rounds.
- Consider process j
  - u = smallest id j has seen so far
  - j discards any token with an id larger than u
- Token returns to sender => it is the leader
- Process with smallest id is elected leader
Communication Complexity

- By the time Umin gets back, the second smallest id would have traversed at most $\frac{1}{2}$ the way around.
- $k^{th}$ smallest is would have traversed $\frac{1}{2}^{k-1}$ way around.
- Total number of messages = $n + \sum(n/2^{k-1}) = 2n$.
- Time complexity: $n.2^{Umin}$. 

S. Venkatesan   Department of Computer Science
Variable start times?

- Read from book.
Lower Bound for Comparison based Algorithms

- $\Omega(n \log n)$ messages even if communication is bidirectional and ring size $n$ is known
- All processes are identical except for ids

- Even if processes have unique ids, symmetry arises and certain communication is needed to break it.
- Comparison based algorithms: Send/receive messages that contain ids only.  
  - Make comparisons and decide
Main Ideas

- Show that there exists an assignment of ids in a ring that is “bad.”
- If an algorithm does not send “many” messages, then the id assignment fools the algorithm.
- In order to get out of the “bad” situation, algorithm is forced to send “many” messages.
Some Definitions

- **Order Equivalence:**
  - $U=(u_1, u_2, \ldots, u_k)$ and $V=(v_1, v_2, \ldots, v_k)$ are sequence of ids
  - $U$ is order equivalent to $V$ if for each $i,j$
    - $u_i \leq u_j \iff v_i \leq v_j$
  - Example: $(4,2,3,7,1)$ and $(8,6,7,9,4)$ are order equivalent
Definitions (Continued)

\[ \text{k-neighborhood of process } I \text{ (2k+1 processes)} \]

Process states \( s \) and \( t \) correspond wrt sequences \( U \) and \( V \) of IDs if

1. IDs of \( s \) are from \( U \)
2. IDs of \( t \) are from \( V \)
3. \( t = s \) except that each \( u_i \) in \( s \) is replaced by \( v_i \) in \( t \)
Lemma 3.5

A is comparison based algorithm in a ring R of n nodes. k is an int 0 ≤ k ≤ n/2. Let i and j be processes s.t. their k neighborhoods are Order equivalent.

After k rounds, i and j are in corresponding states

Proof by induction on k
Let $c, 0 \leq c \leq 1$ be a constant. $R$ is ring of size $n$.

$R$ is $c$-symmetric if for all $l \sqrt{n} \leq l \leq n$, for every segment $S$ of length $l$, there are at least $\frac{cn}{l}$ segments of $R$ that are order equivalent.

[Every bit reversal ring is $\frac{1}{2}$ symmetric]
Lemma 3.8

A is a comparison based algorithm in c-symmetric ring.

A elects a leader $k$ is an integer $\sqrt{n} \leq 2k+1$ and $\frac{cn}{2k+1} > 2$.

A has more than $k$ active rounds.

**Proof:** By contradiction. $i$ and $j$ are centers of 2 order equivalent $k$-neighborhoods.

$I$ and $J$ will be in “equivalent” states after $k$ rounds.
Theorem 3.9

Algorithm A needs \( \Omega(n \log n) \) messages

**Proof:** \( c \) is a constant between 0 and 1 and \( R \) is a \( c \)-symmetric ring of size \( n \)

\[
k = \frac{cn - 2}{4} \implies \sqrt{n} \leq 2k + 1 \quad \text{(if } n \text{ is large)},
\]

\[
\frac{cn}{2k + 1} \geq 2.
\]
By lemma 3.8, there are at least k+1 rounds

Consider round r s.t. $\sqrt{n}+1 \leq r \leq k+1$

Round r is active. Some process i is active
  – (i sends a message in round r)

S = (r-1) neighborhood of i.

There are $\frac{cn}{2r-1}$ order equivalent (to S) segments in R

Mid points of these are in corresponding states
  – The all send messages