T-79.4001 Seminar on Theoretical Computer Science
Spring 2007 – Distributed Computation

Advanced Election Techniques in Rings

Eero Häkkinen

2007-02-21

Rings

Properties:

- n entities: $x_0, x_1, \ldots, x_{n-1}$
- n links: $(x_i, x_{i+1}), (x_{n-1}, x_0)$
  $\Rightarrow$ Each entity has two neighbours (called *left* and *right*)
- Sparsest network topology after trees
- Complete structural symmetry

Restrictions for Elections in Rings

The standard set of restrictions (**IR**):

- Connectivity
- Total Reliability
- Initial Distinct Values
  - To break the complete symmetry
- Bidirectional Links

Possible additional and alternative restrictions:

- Unidirectional Links (instead of bidirectional)
  - Implies Oriented Ring
- Oriented Ring: $\mathit{right}(x_i) = x_{i+1}$, $\mathit{right}(x_n) = x_0$
- Message Ordering
- Known Ring Size

Description of Stages Protocol [1/3]

Protocol *Stages*:

- A *candidate* entity x sends election messages with $id(x)$ to the both directions.
- A *candidate* entity x receives two election messages with $id(y)$ and $id(z)$.
  - If $id(x) > \mathrm{Min}[id(y), id(z)]$, x becomes *defeated*.
  - If $id(x) < \mathrm{Min}[id(y), id(z)]$, x becomes a *candidate* entity for the next stage.
  - If $id(x) = id(y) = id(z)$, x becomes a *leader* and notifies all entities.
- A *defeated* entity forwards election messages.
- Non-initiator receiving an election message becomes
  - a *candidate* entity (*Stages*) or
  - a *defeated* entity (*Stages-Minit*)

  and acts accordingly.

Description of Stages Protocol [2/3]

Out of order messages are problematic because

- A *candidate* entity at stage i should receive exactly one election message from each port.
- A *candidate* entity at stage i cannot make a correct decision based on elections messages from lower stages $j < i$.
- A *defeated* entity at stage i should not forward messages from lower stages $j < i$ to avoid $O(n^2)$ message complexity.

Description of Stages Protocol [3/3]

Possible solutions to problem of out of order messages:

- Require *Message Ordering*.
- Send the current stage along the election messages and either
  - Enqueue locally until out of order messages arrive or
  - Keep track of out of order messages:

To keep track of out of order messages:

- A *candidate* entity x at the stage i receiving a message from the stage $j > i$ acts according to ids.
  - If x is defeated, it forwards the message.
  - If x survives, it does not have to wait for the next $j - i$ messages from the same port.
- An entity can drop messages below its stage.

Properties of Stages Protocol

Correctness:

- x_{min} is never defeated and defeats its neighbour candidates at each stage thus number of candidates decreases monotonically.

Messages:

- Bidirectional election message exchange between candidates thus $2n$ messages during each stage
- Only one from two consecutive candidates can survive to the next stage thus at most $\lceil \log n_0 \rceil + 1$ stages
- $\mathrm{M}[Stages] \leq 2n \log n + O(n)$
- $\mathrm{M}[Stages - Minit] \leq 2n \log k_* + O(n)$

Description of Stages with Feedback Protocol

Protocol *StagesFeedback*:

- A *candidate* entity x sends election messages with $id(x)$ and the current stage to both directions.
- If a *candidate* entity x receives two election messages with $id(y)$ and $id(z)$ from the same stage:
  - If $id(y) < \mathrm{Min}[id(x), id(z)]$, x sends a feedback to y.
  - If $id(z) < \mathrm{Min}[id(x), id(y)]$, x sends a feedback to z.
  - If $id(x) = id(y) = id(z)$, x becomes a *leader* and notifies all entities.

  If x sends a feedback, x becomes *defeated*.
- If a *candidate* entity x receives an election message from a higher stage, x becomes *defeated* and forwards the message.
- If a *candidate* entity x receives feedbacks from the both directions, x becomes a *candidate* entity for the next stage.

Properties of Stages with Feedback Protocol [1/2]

Correctness:

- x_{min} never sends feedbacks and always receives feedbacks from other entities thus number of candidates decreases monotonically.

Messages:

- $2n$ election messages during each stage
- Unidirectional feedback exchange between some candidates thus at most n feedbacks during each stage
- Only one from three consecutive candidates can survive to the next stage (a candidate cannot send feedbacks to the both of its neighbours) thus at most $\lceil \log_3 n_0 \rceil + 1$ stages
- $\mathrm{M}[\mathit{StagesFeedback}] \leq 1.893 n \log n + O(n)$
- $\mathrm{M}[\mathit{StagesFeedback} - \mathit{Minit}] \leq 1.893 n \log k_* + O(n)$

Properties of Stages with Feedback Protocol [2/2]

Bit complexity:

- $2n$ messages with $\log \mathbf{id}$ bits and at most n signals with $c = O(1)$ bits thus $n\,(c + 2\log \mathbf{id})$ bits during each stage
- $\mathrm{B}\,[\mathit{StagesFeedback}] \leq 1.262 n \log n \log \mathbf{id} + l.o.t.$ where $l.o.t.$ stands for "lower order terms"

Description of Alternating Steps Protocol

Protocol *Alternate*:

- Like *Stages* but instead of sending to and receiving from the both directions and making a decision
  1. Send to right.
  2. Receive from left.
  3. Make a decision.
  4. Swap directions.
  5. Repeat.

At each stage, all candidates should send to the same direction and receive from the other direction thus to avoid deadlocks:

- Require *Oriented Ring*.
- Implement a conflict resolution protocol.

Properties of Alternating Steps Protocol

Correctness:

- x_{min} is never defeated and defeats one of its neighbour candidates at each stage thus number of candidates decreases monotonically.

Messages:

- Unidirectional election message exchange between candidates thus n messages during each stage
- At stage i there are n_i candidates.
- $n_i \geq n_{i+1} + n_{i+2}$. Otherwise n_{i+2} candidates would not survived stage $i+1$. A reversed Fibonacci like series thus at most $1.44 \log n + O(1)$ stages.
- $\mathrm{M}\,[Alternate] \leq 1.44 n \log n + O(n)$

Unidirectional Stages

Protocol *UniStages*:

- Emulated *Stages*.
- Operates on envelope ids thus the leader will not be x_{min} but a candidate owning $id(x_{min})$ in the end.
- Each candidate entity sends to right and receives from the left twice at each stage.
- In *Stages*, any given candidate knows the previous, the given and the next candidate. The same is true for the next candidate in *UniStages*.

Messages:

- Similar to *Stages*.
- $\mathrm{M}[\mathit{UniStages}] \leq 2n \log n + O(n)$

Unidirectional Alternate

Protocol *UniAlternate*:

- Emulated *Alternate*.
- Operates on envelope ids thus the leader will not be x_{min} but a candidate owning $id(x_{min})$ in the end.
- In *Alternate*, any given candidate knows the previous, the given and the next candidate. The same is true for the next candidate in *UniAlternate*.

Messages:

- Similar to *Alternate*.
- $\mathrm{M}[\mathit{UniAlternate}] \leq 1.44n \log n + O(n)$

Unidirectional MinMax

Protocol *MinMax*:

- Like *UniAlternate* but prefer small ids at odd stages and large ids at even stages.

Messages:

- At stage i there are n_i candidates.
- $n_i \geq n_{i+1} + n_{i+2}$. Otherwise n_{i+2} candidates would not survived stage $i+1$. A reversed Fibonacci like series thus at most $1.44 \log n + O(1)$ stages.
- $\mathrm{M}[\mathit{MinMax}] \leq 1.44 n \log n + O(n)$

Unidirectional MinMax+ [1/2]

Protocol *MinMax+*:

- At even stage j
  - A message travels at most a predefined distance $dis(j)$.
  - If the message reaches the distance at a *defeated* entity z, z becomes a *candidate* entity at stage $j+i$ with value of the message.
  - If a *candidate* receives a message for the next step, it becomes *defeated* and forwards the message.
  - If a *candidate* becomes *defeated*, it remembers the stage and the value. If at the next stage, it receives a message with a smaller value, it becomes a *candidate* entity and starts the next stage with that value.
- At odd stage, if a *candidate* entity receives a message for the next step, it becomes *defeated* and forwards the message.

Unidirectional MinMax+ [2/2]

Messages:

- $\mathrm{M}[\mathit{MinMax}+] \leq 1.271 n \log n + O(n)$

Complexity of Bidirectional Protocols

	Worst case	Notes
Stages	$2n \log n + O(n)$	
StagesFeedback	$1.892n \log n + O(n)$	
Alternate	$1.44n \log n + O(n)$	*Oriented Ring*
BiMinMax	$1.44n \log n + O(n)$	
Lower bound	$0.5n \log n + O(n)$	ave., $n = 2^p$ known

Complexity of Unidirectional Protocols

	Worst case	Notes
UniStages	$2n \log n + O(n)$	
UniAlternate	$1.44n \log n + O(n)$	
MinMax	$1.44n \log n + O(n)$	
MinMax+	$1.271n \log n + O(n)$	
Lower bound	$0.69n \log n + O(n)$	
Lower bound	$0.25n \log n + O(n)$	ave., $n = 2^p$ known

Unidirectional rings are oriented and it seems that *Oriented Ring* is a better property than *Bidirectional Links*.