Localized failures: synchrony

Nicola Santaro:
Design and Analysis of Distributed Algorithms
Chapter 7.3

March 28th, 2007

Jani Lampinen
jalampin@cc.hut.fi

Single-Failure Disaster theorem

- States that EFT-Consensus (1, crash, n-1) is *unsolvable*.
  - I.e. fault tolerant consensus cannot be achieved even under the best of conditions.
- Additional Assumptions are needed
  - Synch = Unitary (Bounded) Delays + Synchronized Clocks
  - Failures can be detected simply by waiting enough time.

Today's topics

Synchrous Consensus

- With Crash failures in a complete graph.
- With Byzantine failures in a complete graph
  - Boolean case
  - General value case
- With Byzantine failures in an arbitrary graph

Syncronous Consensus with Crash Failures

Additional Assumptions

- Connectivity, Bidirectional links
- Synch
- The network is a compelete graph
- All entities start simultaneously
- The only type of failure is entity crash

Tell All(T)

- The basic form for crash failure algorithms in a complete graph.
- For a predeterminated time T send each time step before t before it a report to all nodes.
- If they don't respond by t+1 they are probably down.
- Used by TellAll-Crash(T)

Tell All – Crash (T)

```
Tell All - Crash
begin
  for t = 0, ..., f do // T == f
    compute rep(x, t)
    send rep(x, t)
  endfor
end


rep(x, t)
if(t == 0)
  return v(x)
else
  return AND(rep(x, t-1), rep(x1, t), .., rep(xn-1,t))
```

- If all entities start with initial value 1, they will decide 1.
- If an entity receives a 0 at time $t \leq f$ then all entities will receive a 0 at $t + 1$.
- If an entity receives a 0 during the execution, it will decide 0.

Tell All – Crash (T)

- Protocol TellAll-Crash solves EFT-Consesus(f, crash, n-1) in a fully synchronous complete network with simultaneous start for all $f \leq n - 1$.
- Bit complexity $\leq n(n-1)(f+1)$
- Time complexity $= f + 1$.

TellZero - Crash

```
TellZero-Crash
begin
  if(I_x) = 0 then
    send 0 to N(x);
  for(t = 1,...,f) do
     compute rep(x,t)
     if(rep(x,t) = 0 and rep(x, t-1) = 1) then
        send 0 to N(x);
  endfor
  O_x := rep(x, f+1)
end
```

- Only 0 gets propagated as a ”wake-up” message.
- Entities with initial state 0 are initially ”awake”.
- Bit complexity $\leq n(n-1)$

Syncronous Consensus with Byzantine Failures

Additional Assumptions (BA)

- Connectivity, Bidirectional links
- Synch
- Each entity has a unique id
- The network is a complete graph
- All enties start; simulteniously
- Each entity knows the ids of its neighbors

Boolean Consesus with Byzantine entities

- TellZero-Crash can be used as a starting point.
  - Additional assumptions.
  - Wake-up messages are now of the form: (0, id(s), t).
- Byzantine entities are malicious and lie..
  - Can claim to be someone else
    - Entities know their neighbours - no problem.
  - Can lie about the time
    - Just silly in a synchronous environment.
  - Can send false wake-up messages
    - Extra mechanism needed.

Dealing with false wake-ups

- If all nonfaulty entities accept the same information, then they will take the same decision.
- Wake-up message must be accepted only if
  - Originator is nonfaulty, or
  - Originator is faulty and all nonfaulty entities have received the message.
- RegisteredMail

RegisteredMail

- To send a registered wake-up (0, id(x), t), a nonfaulty entity x transmits a message (”init”, 0, id(x), t).
- If a y receives (”init”, 0, id(x), t) from x at time t+1, it transmits (”echo”, 0, id(x), t) to all entities.
- If y by the time t' ≥ t+2 receive ”echo”-message from at least f + 1 different entities, then y transmits it at time t' to all entities, if it already hasn't.

RegisteredMail

- If y by the time $t' \geq t+1$ has received (”echo”, 0, id(x), t) messages from at least n-f different entities, it accepts the wake-up message.

RegisteredMail

- Let $n > 3f$; then RegisteredMail satisfies:
  - If x is nonfaulty and sends the registered wake-up (0, id(x), t), then wake-up is accepted by all nonfaulty entities by $t + 2$.
  - If the wake-up (0, id(x), t) is accepted by any nonfaulty entity at time $t' > t$, it is accepted by all of them by $t'+1$.
  - If x is nonfaulty and does not send the registered wake-up (0, id(x), t), then it wont be accepted by nonfaulty entities.

TellZero-Byz

- Uses RegisteredMail.
- Implements a binary Byzantine agreement algorithm
- f+2 stages (0,...,f+1)
  - Stage i is composed of two step 2i and 2i+1.
- Solves EFT-Consensus (f, Byzantine, n-1) with Boolean initial values in a synchronous complete graph under BA (restrictions) for all $f \leq n/3 - 1$!
- Bit complexity $\leq (2f^2+4f+n+n^2-fn+n-f)(n-1)$
- Time complexity = 2(f+2)

TellZero-Byz

- At time 0, every nonfaulty entity x with initial state 0 starts RegisteredMail to send (0, id(x), x).
- At time 2i (the first step of stage i), $1 \leq i \leq f+1$, entity x starts RegisteredMail to send (0, id(x), 2i), iff if it has accepted wake-up messages from at least f+i-1 different entities and hasn't originated wake-up yet.
- At time 2(f+2) x decides on 0 iff it by that time has accepted wake-up, otherwise 1.

General Byzantine Agreement

- It is possible to transform any solution protocol from Boolean case to into one that work with arbitrary, a priori known, set of initial values.
- FromBoolean(BooleanProtocol) – algorithm
  - $\underline{v}$ is default value in IV.
  - ı,o are not equal and do not belong in IV.
  - In the protocol each entity x has four local variables x.a, x.b, x.c and x.d.

FromBoolean(BP)

- At time 0, each entity x sets $x.a := I_x$ and $x.b = x.c = x.d = \iota$, and sends (”first”, x.a) to all.
- At time 1, each entity x:
  - Sets x.b = v if it has received n-f or more copies of the same message (”first”, v); otherwise x.b = o.
  - Sends (”second”, x.b) to all.

FromBoolean(BP)

- At time 2, each entity x
  - Sets x.c to the value different from ı, that occurs most often among the ”second” messages, with arbitrary tie breaks. If all received ”second” messages contain ı, no change is made to x.c.
  - Sets x.d = 1 if it has received n-f or more copies of the same message. Otherwise it will set x.d = 0.
  - Starts execution of the BP using Boolean value x.d as its initial value.
- When execution of BP terminates each x:
  - Decides x.c if the Boolean decision is 1 and x.c is not o. Otherwise decides default $\underline{v}$.

FromBoolean

- Bit complexity **B**(FromBoolean(BP)) ≤ 2n(n-1) log v + **B**(BP)
  - v is the range of values and **B**(BP) complexity of the Boolean Protocol.
- Time complexity **T**(FromBoolean(BP)) = 2 + **T**(BP).

- Example for TellZero-Byz
  - $B = O(n^2 \log v + n^3 \log i)$, where i is range of ids
  - $T = 2f + 6$

Byzantine Agreement in Arbitrary Graphs

Additional Assumptions (GA)

- Connectivity, Bidirectional links
- Synch
- Each entity has a unique id
- All entities have complete knowledge of the topology of the graph and of the identities of the entities.
- All entities start simultaneously

Byzantine agreement in arbitrary graphs

- Because Crash failures are special case of Byzantine failures and with them around $f < c_{node}(G)/2$
  - $c_{node}(G)$ is the minimal number of nodes whose removal destroys the connectivity of G.
- On the other hand, the result $f \geq n/3$ makes EFT-Consensus(f, Byzantine, n-1) unsolvable.
  - And we really can't do better..
- $f \leq \text{Min}\{n/3, c_{node}(G)/2\} - 1$

Two-Parties ByzComm

- If G is 2f+1-node-connected then between any two pair of nodes x and y there are at least 2f+1 node-disjoint paths. (Chapt. 7.1)
- Each nonfaulty entities x and y select 2f +1 node-disjoint paths between them.
  - Complete knowledge of topology (Assumed)
  - More paths deliver the correct result than the wrong one.
  - Simulation of a direct link is possible.
  - New unit time: longest of the paths selected.

Two-Parties ByzComm

- Bit complexity = O(f n **B**(P) + fn² log n **T**(P))
- Time complexity ≤ diam(G)**T**(P)

Network G	Node Connectivity $c_{node}(G)$	Byzantine Entities f
Ring R	2	0
Torus Tr	4	1
Hypercube H	$\log n$	$\frac{1}{2} \log \frac{n}{2}$
CubeConnectedCycle CCC	3	1

FIGURE 7.12: Number f of Byzantine entities tolerated in common networks.

Summary

- Although fault resiliant algorithms are impossible to design in the common case, some solutions are possible if additional assumptions of the network can be made.
- These algorithms can be generalized to withstand even hostile entities in the network.