10-29-2019

After Synchronizer

Chapter 6: Consensus despite process failures.

No link failures. [All links are non-faulty]

Process failures

- Stopping (Fail-Stop)
  - ↓
  - Completely connected n-node graph
  - ~~Fail~~ faulty process can stop any time and will not restart
- Byzantine
  - ↓
  - faulty process can do anything.
  - next section

3 Conditions:

Agreement: No two processes decide a different values

Validity: If all process start with same initial value, $v \in V$, then v is the only possible decision value

Termination: All non-faulty processes eventually terminate.

Synchronous Networks only

f: max # of processes that can fail.

10-31-2019

Consensus Problem

Fail stop failures only, no link failures, Synchronous network, Completely Connected Comm network.

Each process i has $v \in V$ as its input value.

Maintain W_i : set of input values of all processes that i knows of.

f: max #of processes that can fail is known to all.

Code for process i

How many times?

- Broadcast W_i // initially W_i = its own input value
- Receive W sets for other processes
- merge received W sets into W_i

Run for f+1 ~~rounds~~ iterations / rounds.

if $|W_i| = 1$ agree on the single value in W_i

else agree on v_0, a default input value

③ Lemma 6.1 If no process fails in a round r, $(1 \le r \le f+1)$, then

$W_i^r = W_j^r$ $\forall$ i, j that are active after r rounds.

Proof: Straightforward

Lemma 6.2 Let $W_i(r) = W_j(r)$ for all i, j that are active after r rounds. Consider round r' : $r \le r' \le f+1$.

$W_i(r') = W_j(r')$ for all i, j that are active after r' rounds

Proof: Straightforward

Lemma 6.3: If processes i & j are active after $f+1$ rounds, then $W_i = W_j$ after $f+1$ rounds.

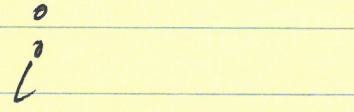
Termination: ✓

Agreement: ✓

Validity: ✓

④ Exponential Information Gathering Algorithms

or EIG Algorithms.

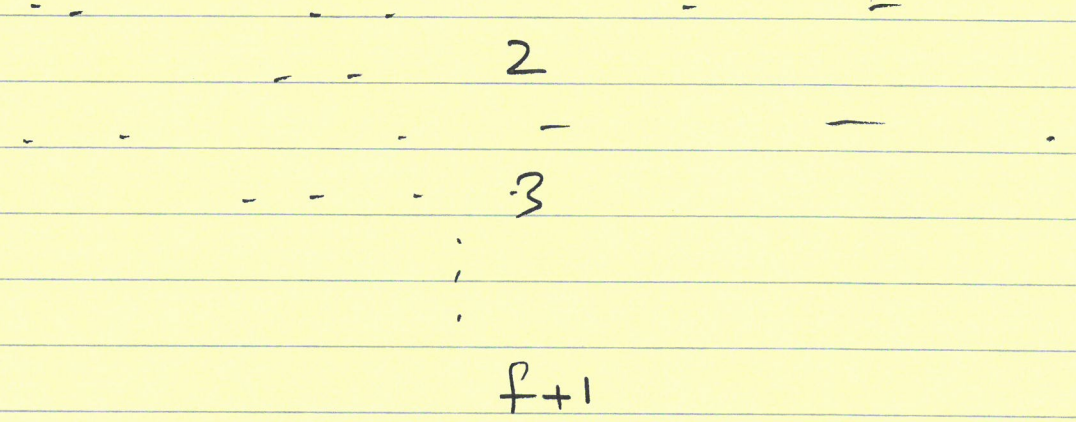


Proven i wants to know j's input value.

i hears (receives) input value of j

through all paths (without loops) of

Length 1



Formalize this via EIG tree

Main idea:

Consider a string (a sequence) of process id's. (without repetition)
(process ids are in the range 1..n)
Total # of processes = n.

String of length k

$i_1\ i_2 \ldots i_k$; no id appears more than once.

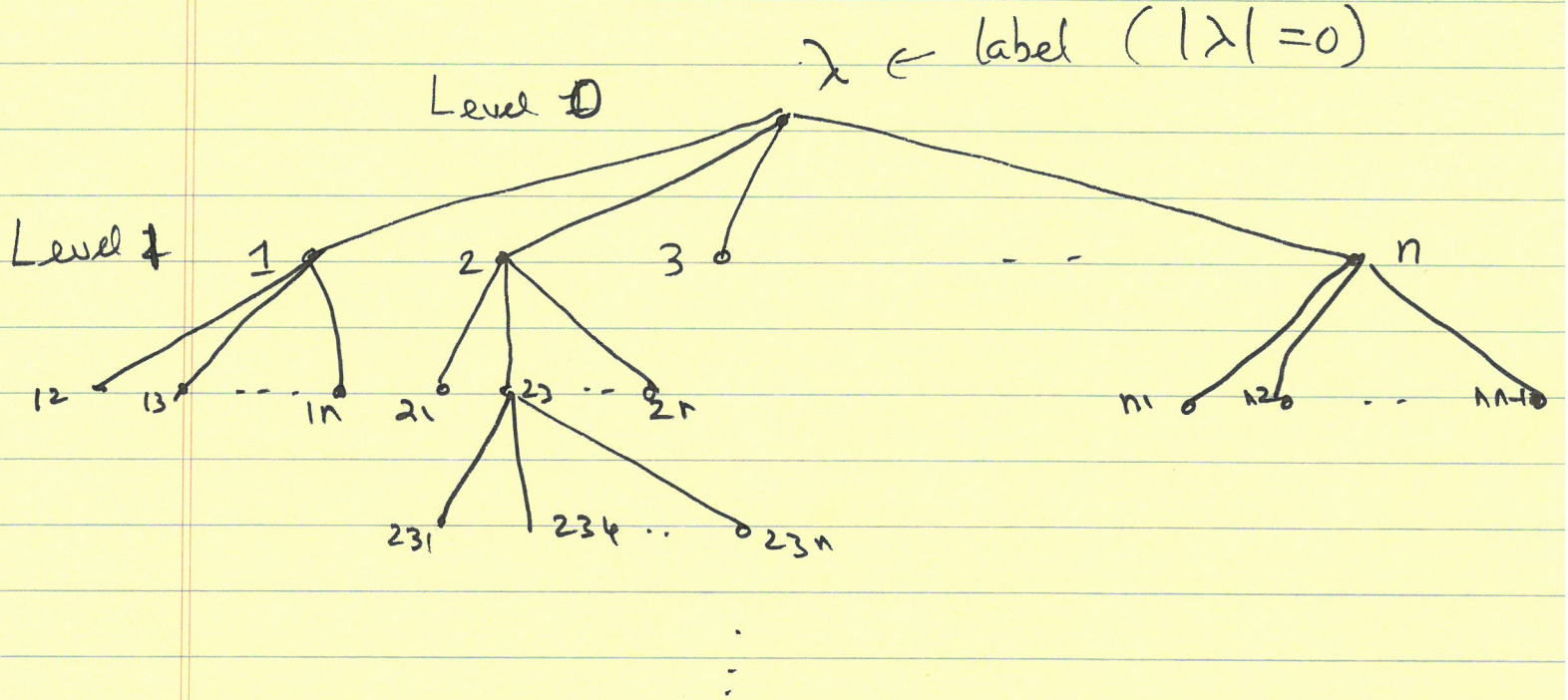
Consider a node in a tree with the label $i_1\ i_2 \ldots i_k$.
We also associate a value $val(i_1, i_2 \ldots i_k)$ with this node.

$Val(i_1\ i_2 \ldots i_k)$ = i_1's value, which was sent to i_2, which was forwarded (i_1's input value as sent to i_2), ~~[illegible]~~ to i_3, which was forwarded to i_4, ... which was forwarded to ~~[illegible]~~ i_k, which has been forwarded to me. is the value assigned to this node.

A process's input value reaches all processes via many paths (all paths of length 1, all paths of length 2, ...) without loops.

$T_{n,f}$ (Total # of process, max. # of faulty processes) is EIG tree

All processes maintain $T_{n,f}$. (Same tree)



EIG trees of all processes are same before the nodes are assigned val's.

Level $f+1$ nodes have labels of length $f+1$

EIG Stop algorithm (Stopping failures or Fail-Stop failure)

Round 1:

Process i assigns $val(\lambda) = i$'s input value

Send $(\lambda, val(\lambda))$ to all (including myself)

if (λ, v) is received by me (i) from process j, then set $val(j) = v$

else (no message is received from j) $val(j) = null$

Round k ($2 \le k \le f+1$):

Process i (me) broadcasts all pairs (x, v) where x is the label of level $k-1$ node in $T_{n,f}$ that does not contain i (my id), $v = val(x)$.

Then record received messages:

Case 1: If x_j is a level k node label of my tree $T_{n,f}$, x = string of process ids, j is a process index (id), and a message (x, v) reach me sent by process j. Then

$val(x_j) = v$;

else // no message of the form (x, v) // was sent by j to me

$val(x_j) =$ null;

After $f+1$ rounds:

if all the val's associated with all the nodes of my $T_{n,f}$ tree are same, decide on that value (ignore null values)

else decide on v_0, a default input value, all processes have agreed on a-priori