The Consensus Problem

Gregory Hill

May 1, 2018

1 Introduction

In a distributed system, we may require that all participating nodes reach an agreement (common outcome). The following document summarizes problems and solutions for synchronous systems - based partially on course notes from Distributed Systems (INFR11022) [3] at the University of Edinburgh.

The Byzantine General's (Coordinated Attack) Problem:

- Several generals independently plan attack.
- Success depends on attacking together.
- Coordination involves sending messages.

- Link Failure: messengers can get lost or captured.
- Process Failure: traitorous generals send incorrect message.

The Agreement Problem:

- Consider n processes in an arbitrary undirected graph - known globally.
- Input v from a set V - starting state.
- Processes make deterministic choices.
- At most f processes may fail.

Trivial to reach consensus in a fault free system:

- All-to-all broadcast then apply a common function to the result.

Correctness conditions:

- Agreement: No two processes decide on different values.
- Validity:
  1. If all start with 0, decision must be 0.
  2. If all start with 1 and all messages are delivered, decision must be 1.
- Termination: All processes eventually decide.

2 Link Failures

Deterministically coordinated attack with link failures [2, 1] - see the 'Two Generals Problem'.

2.1 Impossibility Result

Theorem 1. *Let G be a graph with two nodes connected by an edge. There is no algorithm that solves the coordinated attack problem on G.*

Proof. By contradiction.

- Suppose a solution exists.
- Suppose the algorithm takes r rounds to decide.
- Assume without loss of generality (WLOG) that both processes send messages at every round.

- Let e_0 be an execution where both processes start with 1 and all messages are delivered.
- In e_0 they will both (validity) decide (termination) 1 after r rounds.
- Let e_1 be the same as e_0 except the last message from 1 to 2 is lost.
- By termination and agreement, process 2 eventually decides 1.
- Let e_2 be the same as e_1 except the last message from 2 to 1 is lost.
- By termination and agreement, process 1 eventually decides 1.
- Let e_3 be an execution where all messages are lost after r rounds.
- In e_3 both eventually decide 1.
- Consider e_4 as e_3 but in which process 2 starts with 0.
- As the exchange is indistinguishable from e_3, process 1 decides 1 and so does process 2 by termination and agreement.
- Consider e_5 as e_3 but in which both processes start with 0.
- By termination and agreement, both processes must decide 1.
- This violates validity - both start with 0 and must decide 0.

□

This theorem describes a fundamental limitation. In order to solve the consensus problem, we need to strengthen the model or relax the requirements.

- Stronger Model:
  - Probabilistic assumptions about message loss.
  - Allow processes to use randomization.
- Weaker Requirements:
  - Allow some violation of agreement and/or validity.
  - Allow violation of termination.

3 Process Failures

There are two categories of fault:

- Stopping Failure: Processes may stop without warning - crash.
- Byzantine Failure: Arbitrary processor malfunction, possibly malicious.

Solutions will not necessarily work for both types of failure.

3.1 Stopping Agreement

Assume process may stop working at any point.

Correctness conditions:

- Agreement: No two processes decide on different values.
- Validity: If all processes start with the same v, then v is the only allowable decision.
- Termination: All non-faulty processes eventually decide.

3.1.1 Simple Algorithm

1. Process i maintains $W \subseteq V$ - initially contains i's initial value.
2. Repeatedly broadcast W and merge received sets to W.
3. After k rounds:
    - If $|W| = 1$, then decide on the unique value.
    - Else decide on default value $v_0 \in V$.

The number of rounds (k) depends on the number of failures (f) to be tolerated. In general, $k = f + 1$.

Complexity bounds:

- Time: $f + 1$ rounds.
- Communication: $\leq (f + 1)n^2$ messages - multiply by nb bits.
    - Improved: $\leq 2n^2$ messages - multiply by b bits.

Improved algorithm:

1. Each process broadcasts its own value in round 1.
2. May broadcast at one other round, just after it hears a different value.
3. Decide as before.

3.2 Byzantine Agreement

Faulty processes may exhibit "arbitrary behaviour".

Correctness conditions:

- Agreement: No two non-faulty processes decide on different values.
- Validity: If all non-faulty processes start with the same v, then v is the only allowable decision for non-faulty processes.
- Termination: All non-faulty processes eventually decide.

4 Exponential Information Gathering (EIG)

A strategy for consensus algorithms, which works for Byzantine Agreement as well as Stopping Agreement. Processes send and relay initial values for several rounds, recording the values they receive along various communication paths in a data structure called an EIG Tree - denoted $T_{n,f}$ for n processes and f failures ($f + 2$ levels). At the end, they use a commonly agreed-upon decision rule based on the values recorded in their trees.

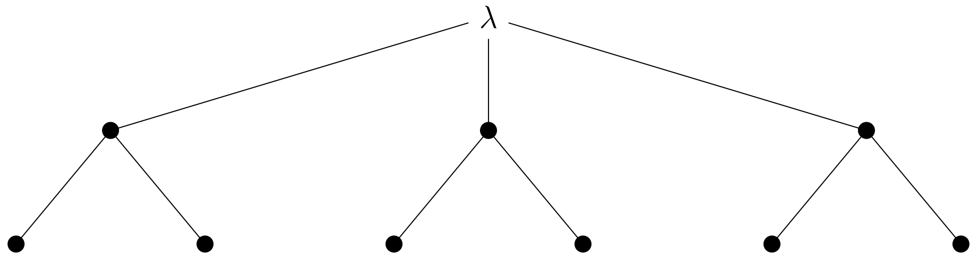

Figure 1: EIG Tree: $T_{3,1}$

4.1 EIG Stopping Agreement Algorithm

- Each process i starts with the same (empty) EIG tree structure $T_{n,f}$ - where λ is decorated with i's input value.
- Round ≥ 1:
  - Send all level $r - 1$ decorations for nodes to everyone - including yourself.
  - On receive, decorate tree by label.
  - If no message received, use $\perp$.
- Decision rule:
  - Let W = set of all values decorating the local EIG tree.
  - If $|W| = 1$ decide that value, else default v_0.

4.2 EIG Byzantine Agreement Algorithm

- Each process starts with the same (empty) EIG tree as before.
- Relay messages for $f + 1$ rounds.
- Decorate the tree with values from V, replacing and garbage messages with default value v_0.
- Decision rule:
  - Redecorate the tree, bottom-up.
  - Leaves remain intact.
  - Other nodes:
    - Strict majority of children in the tree.
    - Otherwise: v_0.
  - Return majority value at root.

Theorem 2. *BA is solvable in an n-node graph G, tolerating f faults, if and only if $n > 3f$ and $conn(G) > 2f$.*

Where $conn(G)$ is the minimum number of nodes whose removal results in either a disconnected or 1-node graph.

References

[1] P. S. Almeida, *Distributed consensus with link failures.* `http://gsd.di.uminho.pt/teaching/DC/2007/slides/dist-consensus-link-failures.pdf`, 2007.

[2] J. Gray, *The two generals problem.* `http://www.cs.rpi.edu/~pattes3/dsa_fall2016/TwoGenerals.pdf`, 2016.

[3] H. Sun, *Distributed consensus.* `https://www.inf.ed.ac.uk/teaching/courses/ds/slides1718/consensus.pdf`, 2018.