Distributed computing
A consensus perspective
A three-act play
A computer can do a lot
A computer can implement a state machine
State machine

• Executes *commands* and updates its state

• **Sequential**: one command at a time

• **Deterministic**: the new state solely depends on (a) the previous state and (b) the command
A computer can do a lot

But it can also do nothing

Or even anything
Immortality by replication
State machine replication

• N replicas of the same state machine are implemented by N distinct computers

• The illusion of one immortal state machine is achieved by consensus
Consensus

The computers *propose* each a value, and have to *decide* on one, among these values.
Consensus

**Agreement:** No two computers decide differently

**Validity:** Any value decided must have been proposed

**Termination:** Every correct computer eventually decides
Act 1

Consensus is universal

Immortal Turing machine
Consensus algorithm
Setting (model/assumptions)

• Each computer has a unique, public, identity

• Each computer executes the algorithm assigned to it (correct) or crashes

• Computers communicate by reliable channels

• Computers are synchronous
Setting
Setting

detect p1
Very simple consensus

• In round i:
  • Pi is **leader**
  • Pi **decides** and sends its decision to every pj, j > i
  • Pj, j >i, **adopts** pi’s proposal
Very simple consensus

propose(0) → decide(0) → propose(1) → decide(0) → propose(0) → decide(0)
Very simple consensus

P1
propose(0)
propose(1)
decide(0)
detect p1

P2
propose(0)
detect p1
decide(1)

P3
propose(0)
decide(1)

crash
Consensus

**Agreement:** No two computers decide differently

**Validity:** Any value decided must have been proposed

**Termination:** Every correct computer eventually decides
Simple consensus

- In round $i$:
  - Pi is *leader*
  - Pi sends its decision to every $p_j$, $j > i$
  - $p_j$, $j > i$, *adopts* pi’s proposal
  - $p_j$ *decides* if $i = n$
Simple consensus

P1 propose(0)

P2 propose(1)

P3 propose(0)

decide(0)
decide(0)
decide(0)
Simple consensus

P1
- propose(0)
- propose(1)

P2
- propose(1)
- detect p1

P3
- propose(0)

crash
decide(1)
decide(1)
Revising the setting

- Each computer has a unique, public, identity
- Each computer executes the algorithm assigned to it (correct) or crashes
- Computers communicate by reliable channels
- **What if computers are asynchronous?**
Asynchrony

P1

suspect p1

suspect p1

P2

P3

suspect p2

suspect p2
Asynchrony

P1
propose(0)

propose(1)
suspect p1

P2
decide(0)
decide(1)

P3
propose(0)
suspect p1
decide(1)
Consensus is impossible in an asynchronous system with at least one crash
Revising the setting

• Each computer has a unique, public, identity

• Each computer executes the algorithm assigned to it (correct) or crashes

• Computers communicate by reliable channels

• **What if computers are eventually synchronous?**
Eventual synchrony

P1
suspect p1

P2
suspect p1

P3
suspect p2
Assume 2 computers can crash

P1 propose(0)

P2 propose(1)

P3 propose(1)

decide(0)

decide(1)
Revisiting the setting

• Each computer has a unique, public, identity

• Each computer executes the algorithm assigned to it (correct) or crashes: at most a minority can crash

• Computers communicate by reliable channels

• Computers are eventually synchronous
Consensus algorithm

- Computer pi is **leader** in every round k such that $k \mod n = i$
- In round i, pi **tries to** decide and inform the others of the decision
Rotating leadership

P1's round | P2's round | P3's round | P1's round | etc
---|---|---|---|---
propose(0) | | | | 

P1

propose(1)

P2

propose(0)

P3
When does a leader decide?

• Pi decides if it is not suspected
  (those that suspect Pi inform Pi and move on)

• If Pi decides, Pi broadcasts the decision
3 steps for a decision at round i

• 1. Pi selects among a majority the latest adopted value (latest with respect to the round in which the value is adopted – see step 2)

• 2. Pi imposes that value to a majority: any process in that majority adopts that value – Pi fails if it is suspected

• 3. Pi decides and broadcasts the decision
Consensus algorithm

propose(0)
propose(1)
crash
nack
nack
Consensus algorithm

P1
- propose(0)
- propose(1)

P2
- propose(1)

P3
- propose(1)

p1’s round | p2’s round | p3’s round | p1’s round | etc

0

nack

1

0
Key idea

P1

round k

n/2

round k+1

Pn

= leader of that round
DLS-L-CT-LO

- Consensus can be solved with eventual synchrony and a majority of correct computers
How fast can consensus be?

- Best case complexity: $f+2$ rounds
How fast can consensus really be?

Ask system people

System people are like fish

System people know

“...the fish doesn’t think, because the fish knows....”

Iggy Pop
Act 2

Consensus is possible

Indulgent Turing machine
What is our business model?

Assumptions

Guarantees
The business model

• Revisiting the setting

• Revisiting the guarantees
Revisiting the setting
What if?

- Processes are weakly synchronous
- Failures are not independent
- Processes can be malicious
- Processes can be anonymous
- Communication is by shared memory
ABD

• Read-write shared memory is equivalent to reliable message passing

• Consensus is impossible in an asynchronous shared memory system
The *consensus number* of an object is the maximum number of processes among which the object implements *consensus*.
Revisiting the guarantees
What if?

• Probabilistic termination is enough
• Computers can decide on k values
K-set Consensus

**Agreement:** At most k different values are decided

**Validity:** Any value decided must have been proposed

**Termination:** Every correct computer eventually decides
Sperner’s Lemma: at least one triangle has three colors
Revisiting the guarantees
What if?

• Probabilistic termination is enough
• Computers can decide on k values
• Commands come in sets
“...we need to explore new techniques like transactional memory that will allow us to get the full benefit of all those transistors and map that into higher and higher performance.”

Bill Gates, Businessman
Act 3

Consensus is universal
Understanding consensus is a fantastic research agenda

There is more to distributed computing than consensus
Thank you for the attention