Optimal Synchronous Approximate Agreement with Asynchronous Fallback

Diana Ghinea[1] Chen-Da Liu-Zhang[2] Roger Wattenhofer[1]

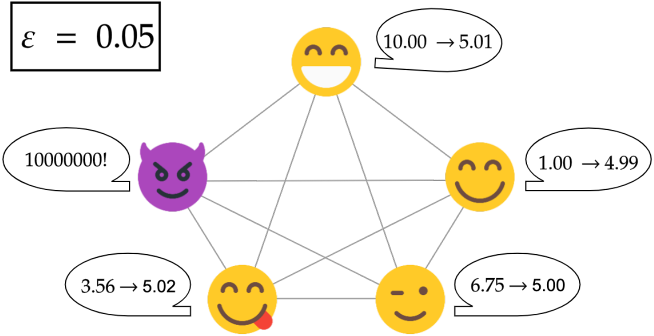
[1]ETH Zürich

[2]NTT Research

Approximate Agreement

Given $\varepsilon > 0$, the honest parties obtain outputs that:

- are within the range of their inputs (*validity*)
- are ε-close (ε-*agreement*)



Resilience Thresholds

- Synchronous model:
  (known message delay Δ, synchronized clocks)

- Asynchronous model:
  (delay unknown, clocks might not be synchronized)

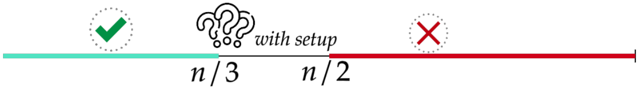
Resilience Thresholds

- Synchronous model:
  (known message delay Δ, synchronized clocks)

- Asynchronous model:
  (delay unknown, clocks might not be synchronized)

Resilience Thresholds

- Synchronous model:
  (known message delay Δ, synchronized clocks)



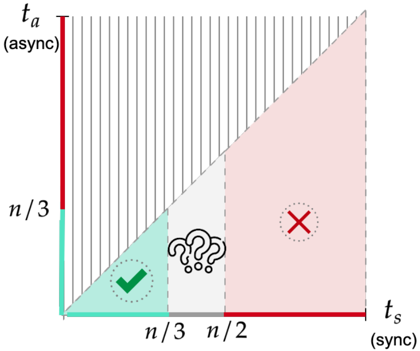
- Asynchronous model:
  (delay unknown, clocks might not be synchronized)

Main Question

The parties do not know the type of network they are in:

- synchronous $\implies$ $n/3 \leq t_s < n/2$ byzantine parties
- asynchronous $\implies$ $t_a < n/3$ byzantine parties

Can we achieve Approximate Agreement in this model?



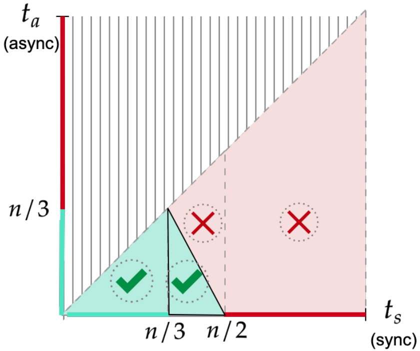
Main Question

The parties do not know the type of network they are in:

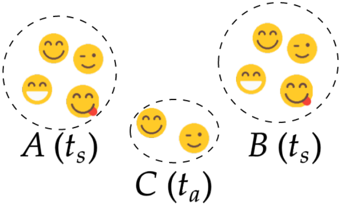
- synchronous $\implies n/3 \le t_s < n/2$ byzantine parties
- asynchronous $\implies t_a < n/3$ byzantine parties

Can we achieve Approximate Agreement in this model?

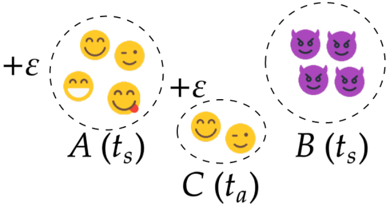
Yes, when $2t_s + t_a < n$!



What if $2t_s + t_a = n$?

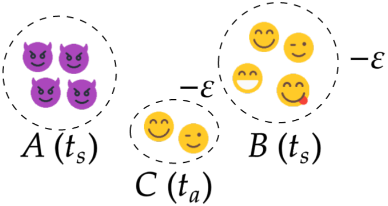


What if $2t_s + t_a = n$?



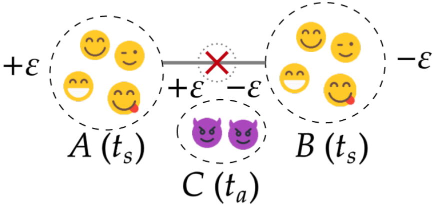
Scenario 1
(synchronous network)

What if $2t_s + t_a = n$?



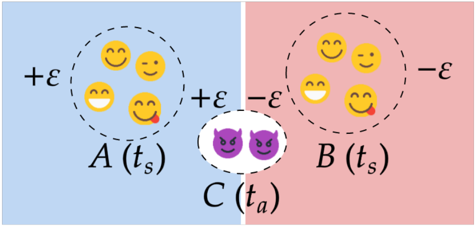
Scenario 2
(synchronous network)

What if $2t_s + t_a = n$?



Scenario 3
(asynchronous network)

What if $2t_s + t_a = n$?



Scenario 3
(asynchronous network)

Achieving Approximate Agreement

Multiple iterations.

In iteration i:

1. Distribute current value v_i to all the parties
2. When *enough* values v_i' are received, discard the outliers
3. Compute $v_{i+1} :=$ the average between the min and max values v_i' that were not discarded

How do we remove outliers?

- *Outlier* = value outside the range of honest values v_i
  $\implies$ sent by a corrupted party

How do we remove outliers?

- *Outlier* = value outside the range of honest values v_i
  $\implies$ sent by a corrupted party
- If the network is synchronous and $\mathbf{n} - \mathbf{t_s} + \mathbf{k}$ values are received, at most $\mathbf{k}$ of these are sent by corrupted parties.

1.2 4.5 4.76 4.85 7.30 90 100 10000000

How do we remove outliers?

- *Outlier* = value outside the range of honest values v_i
  $\implies$ sent by a corrupted party
- If the network is synchronous and $\mathbf{n} - \mathbf{t_s} + \mathbf{k}$ values are received, at most $\mathbf{k}$ of these are sent by corrupted parties.

1.2 4.5 4.76 4.85 7.30 90 100 10000000

$\implies$ we discard the lowest and the highest $\mathbf{k}$ values!

~~1.2~~ ~~4.5~~ 4.76 4.85 7.30 90 ~~100~~ ~~10000000~~

How do we remove outliers?

- *Outlier* = value outside the range of honest values v_i
  $\implies$ sent by a corrupted party
- If the network is synchronous and $\mathbf{n} - \mathbf{t_s} + \mathbf{k}$ values are received, at most $\mathbf{k}$ of these are sent by corrupted parties.

1.2 4.5 4.76 4.85 7.30 90 100 10000000

$\implies$ we discard the lowest and the highest $\mathbf{k}$ values!

~~1.2~~ ~~4.5~~ 4.76 4.85 7.30 90 ~~100~~ ~~10000000~~

- But what if the network is actually asynchronous and the missing values are honest but delayed?
  $\implies$ we discard the lowest and the highest $\max(\mathbf{k}, \mathbf{t_a})$ values!

Achieving Approximate Agreement

In iteration i:

1. Distribute current value v_i to all the parties
2. Out of the $\mathbf{n} - \mathbf{t_s} + \mathbf{k}$ values received:
   discard the lowest and the highest $\max(\mathbf{k}, \mathbf{t_a})$ values
3. Compute $v_{i+1} :=$ the average between the min and max
   values v_i' that were not discarded

Ensuring ε-Agreement is achieved

If the distributing step guarantees that every two parties receive $n - t_s$ common values:

😁 ~~1 2 3 4~~ [6 7] ~~8 9 10 11~~

😋 ~~1~~ [2 3 5 9 10] ~~11~~

Even after removing the outliers, there is some common range.

$\implies$ the range of honest values is halved in each iteration

Overlap All-to-All Broadcast

Each party P has an input v_P and outputs a set O_P upon termination.

Overlap All-to-All Broadcast

Each party P has an input v_P and outputs a set O_P upon termination.

If P_1 and P_2 are honest, then:

- $\left|O_{P_1} \cap O_{P_2}\right| \geq n - t_s$
- Synchronous network $\implies (v_{P_2}, P_2) \in O_{P_1}$

Overlap All-to-All Broadcast

Each party P has an input v_P and outputs a set O_P upon termination.

If P_1 and P_2 are honest, then:

- $|O_{P_1} \cap O_{P_2}| \geq n - t_s$
- Synchronous network $\implies$ $(v_{P_2}, P_2) \in O_{P_1}$
- $(v, P_3) \in O_{P_1}$ and $(v', P_3) \in O_{P_2}$

$$\implies v = v' \ (= v_{P_3} \text{ if } P_3 \text{ is honest})$$

Overlap All-to-All Broadcast

Each party P has an input v_P and outputs a set O_P upon termination.

If P_1 and P_2 are honest, then:

- $\left|O_{P_1} \cap O_{P_2}\right| \geq n - t_s$
- Synchronous network $\implies$ $(v_{P_2}, P_2) \in O_{P_1}$
- $(v, P_3) \in O_{P_1}$ and $(v', P_3) \in O_{P_2}$
  $$\implies v = v' \; (= v_{P_3} \text{ if } P_3 \text{ is honest})$$
- Synchronous network $\implies$ simultaneous termination

Final Protocol

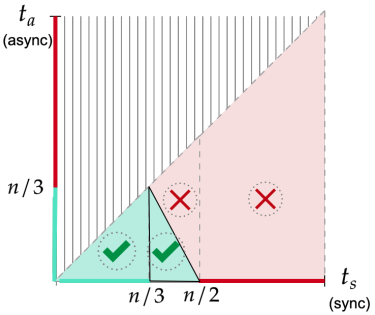
In iteration i:

1. Join Overlap All-to-All Broadcast with input v_i. Obtain O_P
2. Out of the $\mathbf{n} - \mathbf{t_s} + \mathbf{k}$ values in O_P:
   discard the lowest and the highest $\max(\mathbf{k}, \mathbf{t_a})$ values
3. Compute $v_{i+1} :=$ the average between the min and max
   values from O_P that were not discarded

Summary

The parties do not know if the network:

- is synchronous
  $\implies n/3 \leq t_s < n/2$ corruptions
- or asynchronous
  $\implies t_a < n/3$ corruptions



In this setting, Approximate Agreement is achievable iff $2t_s + t_a < n$.