Approximate Consensus

N processes, f crash faults (N > 2f)

Asynchronous systems

Properties:

Termination: eventually, each fault-free process has an output

Agreement:

Validity:

Approximate Consensus

N processes, f crash faults (N > 2f)

Asynchronous systems

Properties:

Termination: eventually, each fault-free process has an output

Agreement: each fault-free process has "roughly" the same output

difference <u>bounded</u> by a constant

Validity:

Approximate Consensus

N processes, f crash faults $(N > 2f)$

Asynchronous systems

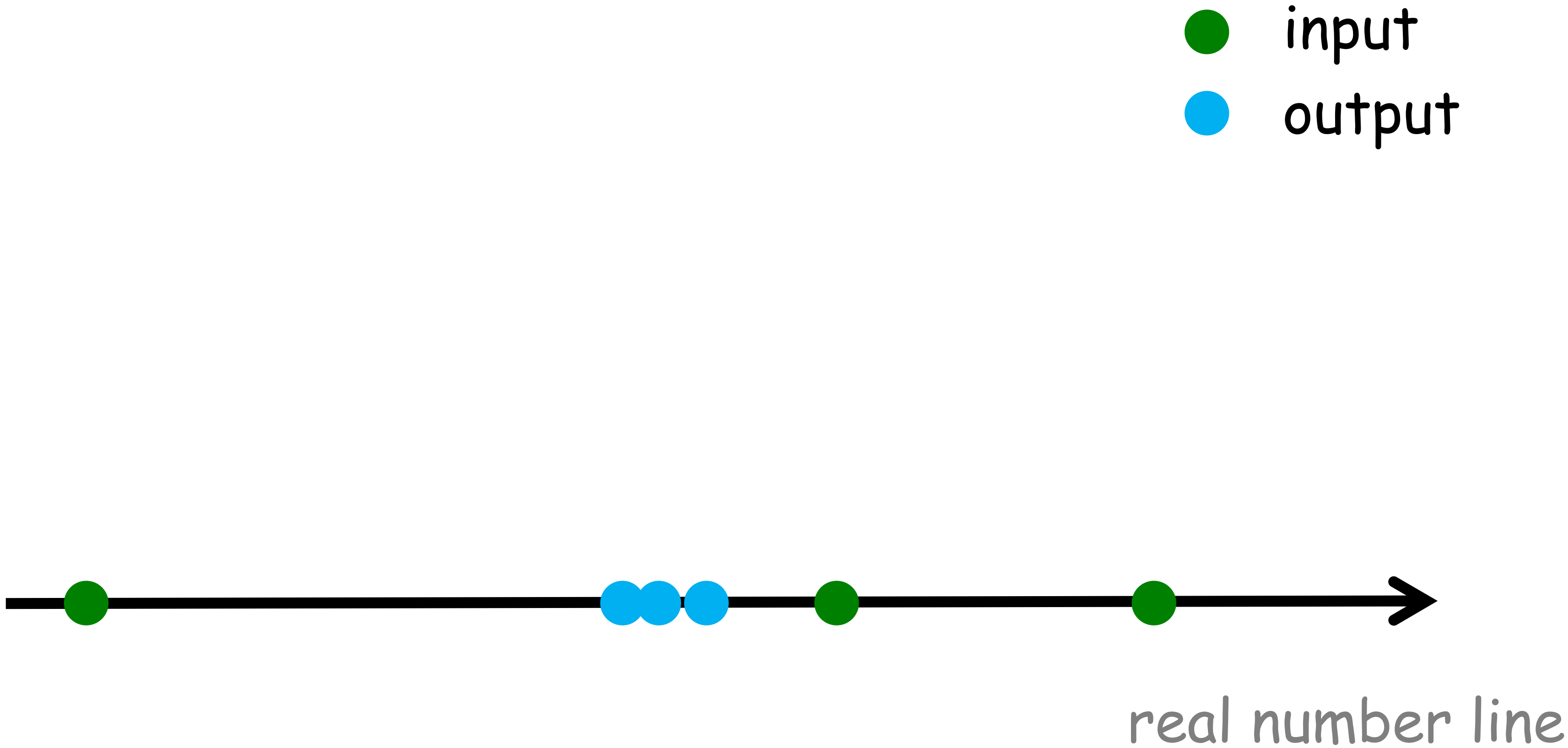
Properties:

Termination: eventually, each fault-free process has an output

Agreement: each fault-free process has "roughly" the same output

Validity: output inside convex hull

Approximate Consensus



Approximate Consensus Algorithm

Process i proceeds in asynchronous rounds

1. Initialization:
   $y_i := x_i$
   $r := 1$

2. Send message (y_i, r) to all the processes including self.

3. Wait until $(n - f)$ messages of the form $(*, r)$ are received (including message from self).

4. Update y_i = average of the $n - f$ values in the above $n - f$ messages. Note that the value is the first field in the tuple in each message.

5. $r := r + 1$

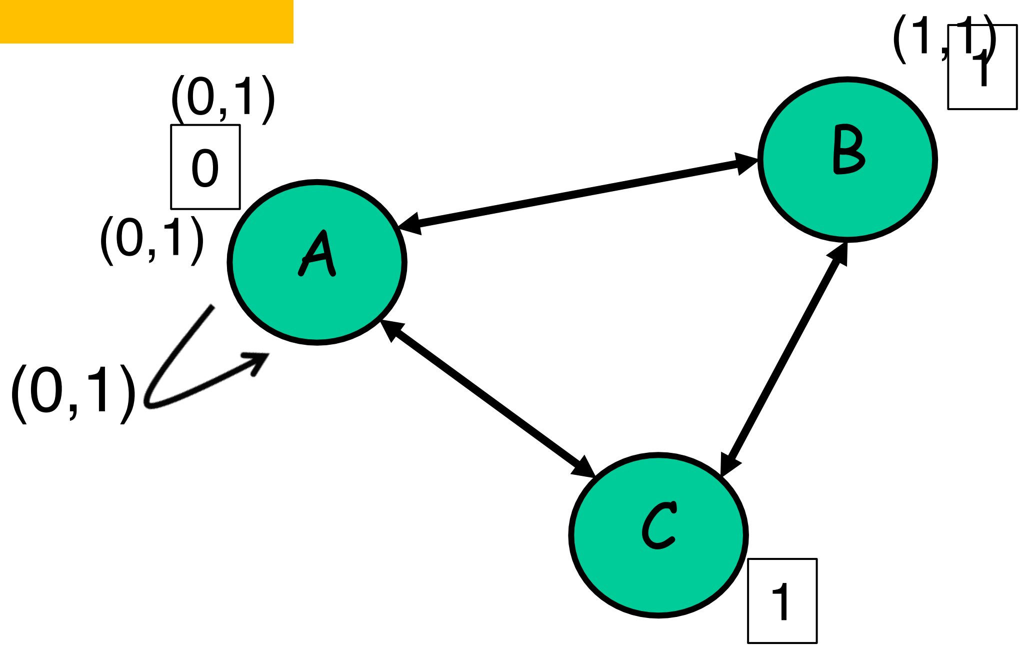
6. Go to step 2

Example Run of the Algorithm

- Round 1 (from perspective of A)

f = 1

A's new state
= (0+1)/2
= 0.5

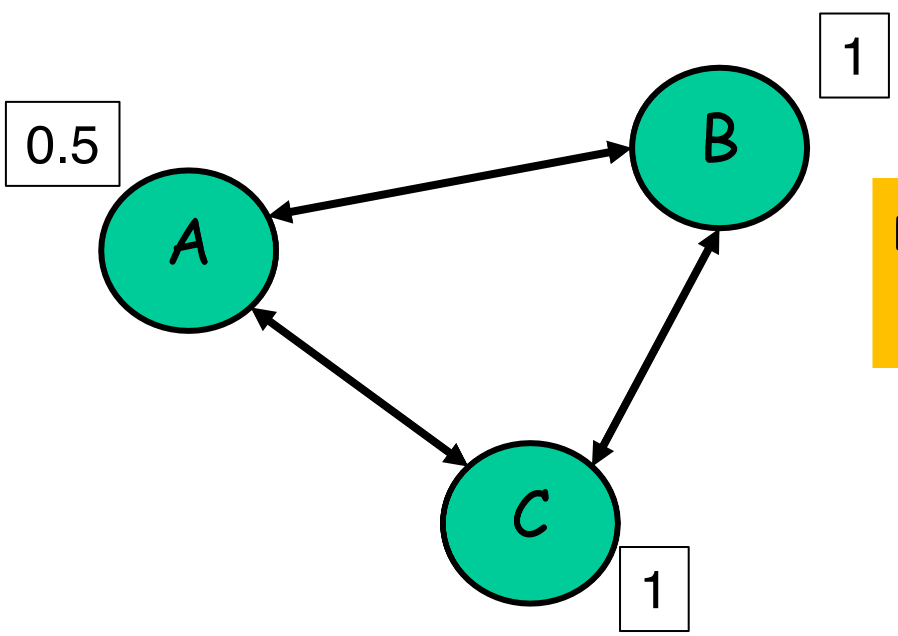


Example Run of the Algorithm

f = 1

- End of Round 1

  (suppose B and C did not wait for A's message)



range <u>shrinks</u>

[0,1] → [0.5, 1]

Correctness

- Termination is obvious
  - fixed number of asynchronous rounds
- Validity is also obvious
  - validity: output inside convex hull ➔ due to “average”

Agreement

- Tow processes i, j
  - Ri[t] = values received at i in iteration t
  - Rj[t] = values received at j in iteration t
  - yi[t] = state at i in the end of iteration t
  - yj[t] = state at i in the end of iteration t
- Key observation: Ri[t] ∩ Rj[t] is not empty

  N > 2f and |Ri[t]| = |Rj[t]| = N-f
- Exercise: show agreement

  |yi[t]-yj[t]| approaches 0 as t increases

Broadcast

Reach agreement on what the source S has said

Broadcast

Reach agreement on what the source S has said

Byzantine Broadcast

- Any process may be Byzantine faulty,
  …including the source S

- See relevant textbook section

Lower Bounds for Byzantine Broadcast in a Synchronous System

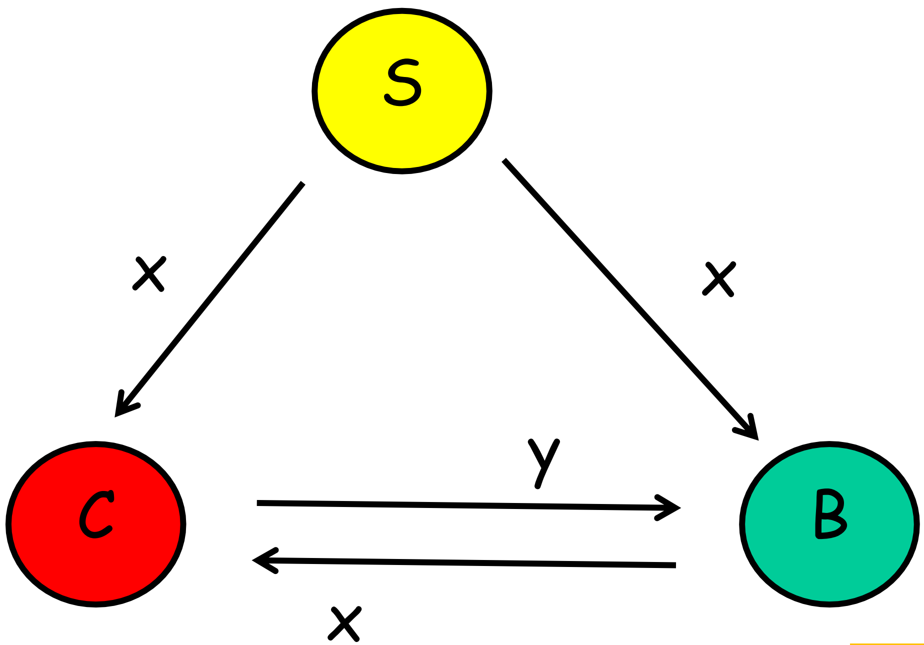
- Number of rounds must be at least $f+1$

- Number of processes must be more than $3f$

Number of Processes

N = 3
f = 1

Scenario 1: C is faulty

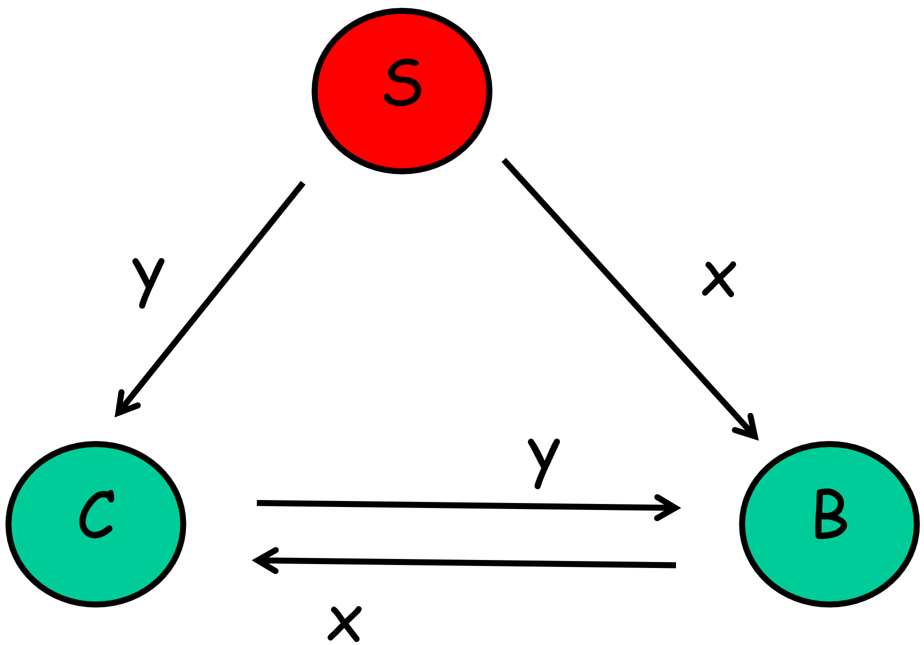


B should output x

Number of Processes

N = 3
f = 1

Scenario 2: S is faulty

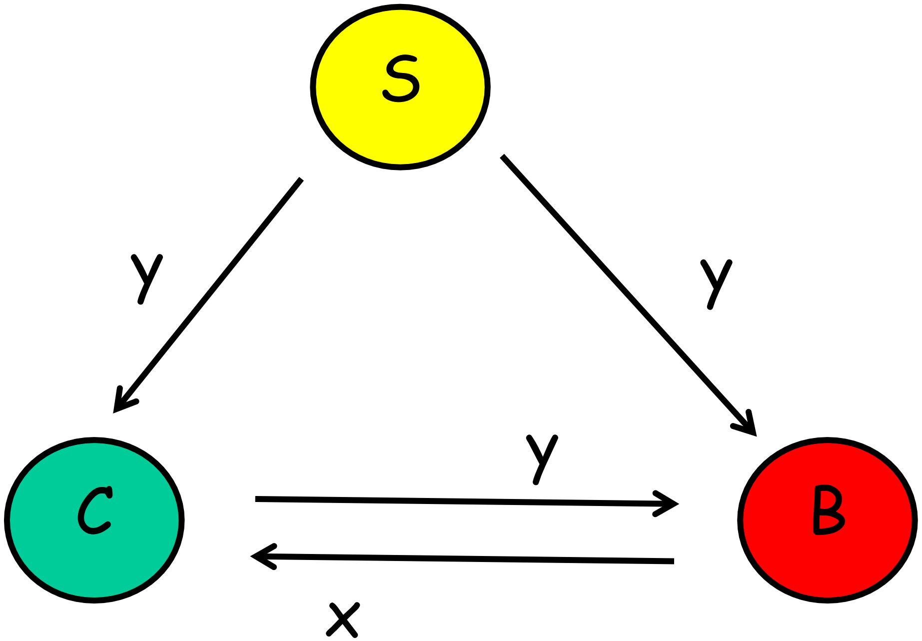


Indistinguishable from Scenario 1 for B
➔ B should output x, so as C

Number of Processes

N = 3
f = 1

Scenario 3: B is faulty



Indistinguishable from Scenario 2 for C
➔ C should output x

violating agreement