Consensus with Halting Failures in a Message-Passing Synchronous System (Cont'd)

Suppose we have a complete network. I.e. every process has a direct channel to every other process.

Recall that, during a round, every process can send a message to every other process. During the round in which a process fails, it may send out some, all or none of the messages it was supposed to send during that round. (In other words, a process can die "in the middle of" a round, sending messages to an arbitrary subset of the other processes.)

Reference: [AW98], Section 5.1; [Lyn96], Section 6.7; [DM90].

Recall from last day:

Definitions: Let α and β be two executions and let p be a processor.

$\alpha \stackrel{p}{\sim} \beta$ means executions α and β are indistinguishable to process p.

$\alpha \sim \beta$ means $\alpha \stackrel{p}{\sim} \beta$ for some process p that is nonfaulty in both α and β.

$\approx$ is the transitive closure of the $\sim$ relation.

Last day, we proved:

Lemma: For consensus algorithms, if $\alpha \approx \beta$, then α and β have same outputs.

Theorem: A consensus algorithm in a synchronous message-passing system that tolerates f halting failures requires at least $f+1$ rounds in some execution.

Proof: Assume there exists an algorithm that uses $\leq f$ rounds.

Let α_i be the failure-free execution, where $\mathrm{p}_1 \cdots \mathrm{p}_i$ have input 0 and $\mathrm{p}_{i+1} \cdots \mathrm{p}_n$ have input 1

α_0 $1\ 1\ 1\ 1 \cdots 1 \longrightarrow$ output 1

α_1 $0\ 1\ 1\ 1 \cdots 1$

α_2 $0\ 0\ 1\ 1 \cdots 1$

...

α_n $0\ 0\ 0\ 0 \cdots 0 \longrightarrow$ output 0

If we could show, for $\forall i$, $\alpha_i \approx \alpha_{i+1}$, then $\alpha_0 \approx \alpha_n$ and we would have a contradiction to the preceding lemma.

Now our goal is to prove $\alpha_i \approx \alpha_{i+1}$. Note that the only process who has different inputs in the two runs is process $i+1$, so, intuitively, we want to prevent that process from telling the others about what it knows. But if we just kill it off right at the beginning of the execution, other processes would notice its absence: the resulting run would look quite different from either α_i or α_{i+1}, in which process $i+1$ is nonfaulty. So, we need to produce a chain of executions that take us from α_i to the execution where process $i+1$ is dead, and a chain from there to α_{i+1}.

Consider the following executions:

μ_i with same input as α_i, and p_{i+1} fails immediately (and no other failures).

ν_i be same input as α_{i+1} ,and p_{i+1} fails immediately (and no other failures).

We want to show $\alpha_i \approx \mu_i \sim \nu_i \approx \alpha_{i+1}$

Lemma: Let ϕ be a run where at most one process halts in each round.

Let ψ be the execution that has same inputs as ϕ, and is the same up to and including round r, and has no failures after round r $(0 \le r \le f)$.

Then $\phi \approx \psi$.

Once we have proved this lemma, it follows immediately that
$\alpha_i \approx \mu_i$ (taking $r = 0$)
$\alpha_{i+1} \approx v_i$ (taking $r = 0$)
So all we have to do now is prove the lemma.

Proof of Lemma: by reverse induction on r.

Base case ($r = f$): ϕ, ψ are same up to (and including) round f, so:

$\phi = \psi$ (since there are only f rounds).

Induction step: Assume the claim is true for $r + 1$, prove it for r.

Let β be same as ϕ up to round $r + 1$, with no failure after. By I.H., $\phi \approx \beta$.

Case I: If no failure in ϕ in round $r + 1$, then $\beta = \psi$, so $\phi \approx \beta = \psi$.

Case II: One process p fails in round $r + 1$ of ϕ. (Refer to figures on next page).

When p fails in round $r + 1$, it sends messages to some subset of the processes during round $r + 1$ before dying. Let $j_1, j_2, \cdots, j_k$ be the processes that do *not* receive messages from p in round $r + 1$ of β.

For $0 \le i \le k$, let β_i be same as ϕ up to round r, then p fails in round $r + 1$, but in round $r + 1$, p sends out messages that p sends in ϕ plus messages to $j_1, j_2, \cdots, j_i$. And there are no failures after $r + 1$.

Our next goal: Show $\beta_i \approx \beta_{i+1}$.

If $r + 1 = f$, $\beta_i \sim \beta_{i+1}$ to any process except j_{i+1}, so $\beta_i \approx \beta_{i+1}$.

Now suppose $r + 1 < f$. Then, at the end of round $r + 1$, only process who can distinguish β_i from β_{i+1} is j_{i+1}.

So, intuitively, we should kill witness j_{i+1} in next round.

Let γ be the same as β_i, except process j_{i+1} is killed right at the start of round $r + 2$.

Let δ be the same as $\beta_i + 1$, except process j_{i+1} is killed right at the start of round $r + 2$.

$\beta_i \approx \gamma$, by I.H. (using β_i in the role of ψ and γ in the role of ϕ).

$\beta_{i+1} \approx \delta$, by I.H. (using β_{i+1} in the role of ψ and δ in the role of ϕ).

Furthermore, γ and δ are indistinguishable to any correct process. So, $\beta_i \approx \gamma \sim \delta \approx \beta_{i+1}$.

So far, we have $\phi \approx \beta = \beta_0 \sim \beta_1 \sim \beta_2 \sim \cdots \sim \beta_k$. So now we just have to show that $\beta_k \approx \psi$.

If $r + 1 = f$, then $\beta_k \sim \psi$ for every correct process.

Now suppose $r + 1 < f$. Recall that p dies in round $r + 1$ of β_k, but p sends a message to every process during round $r + 1$ before it dies. Let β' be the same as β_k except that p dies at the start of round $r + 2$ (before sending any messages in that round). Clearly $\beta' \sim \beta_k$ for all correct processes. By induction hypothesis, $\beta' \approx \psi$ (using β' in the role of ϕ). So we have $\beta_k \sim \beta' \approx \psi$.

QED

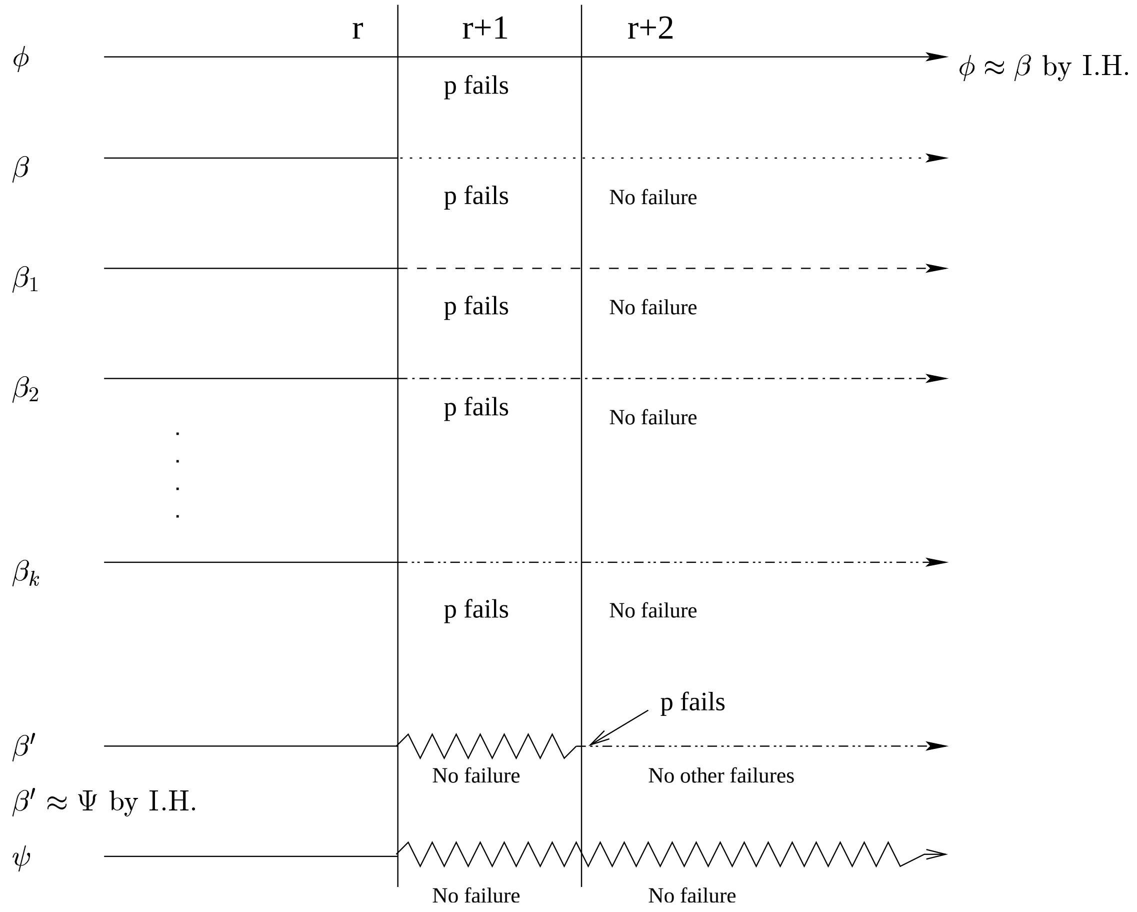

Figure 1: $\phi \approx \psi$

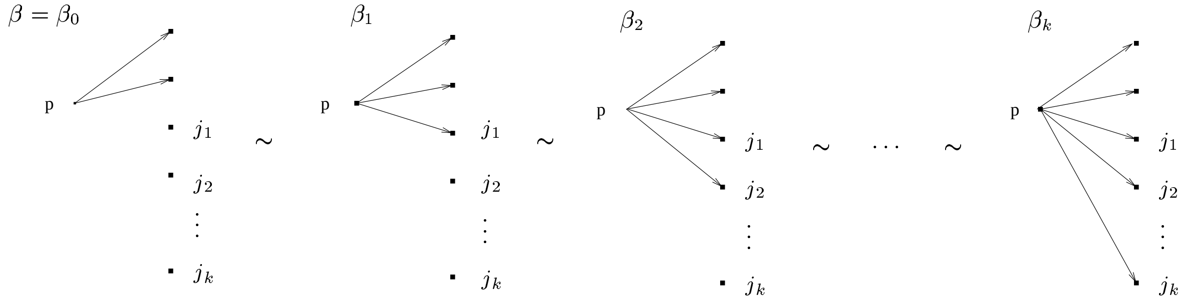

Figure 2: Picture of what p does in round $r+1$ of β_i

Summary: Consensus (Message Passing Systems)

Byzantine Consensus

- Impossible if $f \geq \frac{n}{3}$ or if $f \geq \frac{c}{2}$ (even for synchronous system).
- Possible in synchronous system if $f < \frac{n}{3}$ and $f < \frac{c}{2}$ (We saw an algorithm that uses $2f + 2$ rounds in complete graph (know $\geq f + 1$ are required)).

Consensus with halting failures

- For complete graph in synchronous systems, possible (we gave algorithm that uses $f + 1$ rounds). We just proved that any algorithm will require $\geq f + 1$ rounds.
- Whether this is solvable in synchronous system with arbitrary network graph is homework exercise 4.
- Impossible in asynchronous systems (hence in asynchronous system, Byzantine consensus is also impossible), even if $f = 1$. We'll prove this soon.