Synchronizer

Distributed Systems

Ali Kamandi, PH.D.

School of Engineering Science

College of Engineering

University of Tehran

kamandi@ut.ac.ir

2024

Synchronizers

Synchronizers **simulate** an execution of a failure-free synchronous system in a failure-free asynchronous system.

Synchronizers

Formally, a synchronizer sits between the underlying network and the processes and does one of two things:

- A **global synchronizer** guarantees that no process receives a message from round *r* until all processes have sent their messages for round *r*.
- A **local synchronizer** guarantees that no process receives a message from round *r* until all of that process's neighbors have sent their messages for round *r*.

The alpha synchronizer

The alpha synchronizer uses local information to construct a **local synchronizer**. In round *r*, the synchronizer at *p* sends *p*'s message (tagged with the round number) to each neighbor *p'* or noMsg(*r*) if it has no messages. When it collects a message or noMsg from each neighbor for round *r*, it delivers all the messages. It's easy to see that this satisfies the local synchronization specification.

پیچیدگی

- پیچیدگی زمانی تغییر نمی‌کند ولی پیام های اضافی ارسال می‌شود.
- چنانچه الگوریتم T دور تکرار داشته باشد، نیاز به $|E|.T$ پیام خواهد بود.

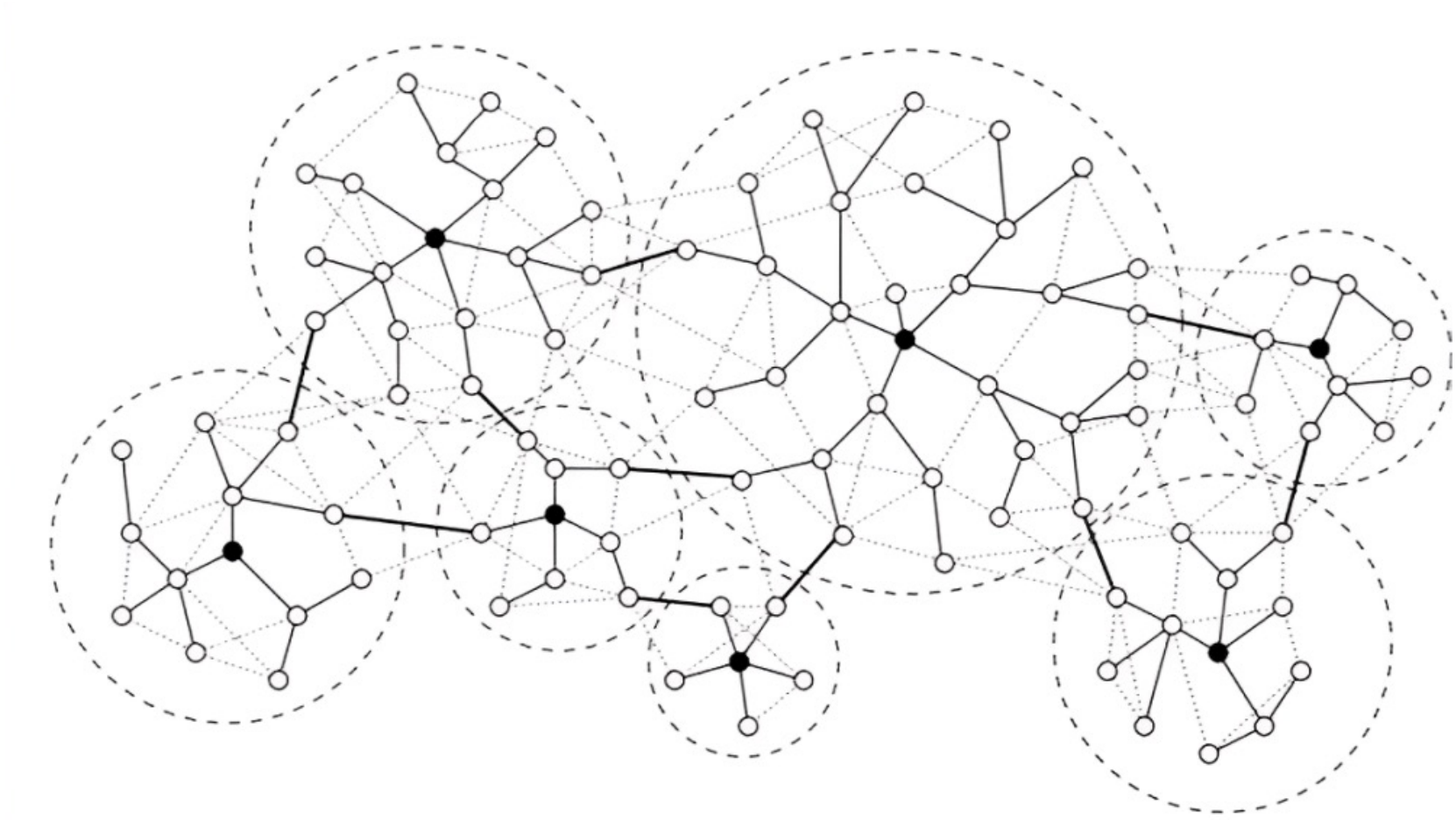
The beta synchronizer

The beta synchronizer centralizes detection of message delivery using a rooted directed spanning tree (previously constructed). When *p'* receives a round-*r* message from p, it responds with ack(*r*). When p collects an ack for all the messages it sent plus an OK from all of its children, it sends OK to its parent. When the root has all the ack and OK messages it is expecting, it broadcasts go. Receiving go makes *p* deliver the queued round-*r* messages.

پیچیدگی

- پیچیدگی پیامی از M به $2M + 2(n-1)$ افزایش می یابد.
- پیچیدگی زمانی متناسب با عمق درخت افزایش می یابد.

Visualization



The gamma synchronizer

The gamma synchronizer combines the alpha and beta synchronizers to try to get low blowups on both time complexity and message complexity. The essential idea is to cover the graph with a spanning forest and run beta within each tree and alpha between trees. Specifically:

- Every message in the underlying protocol gets acked (including messages that pass between trees).
- When a process has collected all of its outstanding round-r acks, it sends OK up its tree.
- When the root of a tree gets all acks and OK, it sends ready to the roots of all adjacent trees (and itself). Two trees are adjacent if any of their members are adjacent.
- When the root collects ready from itself and all adjacent roots, it broadcasts go through its own tree.

Reference

- **Aspnes, Chapter 7**