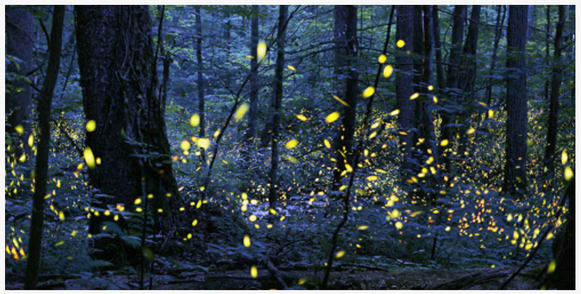
Asynchronous Concurrency in Anonymous Dynamic Networks

Peter Vargas, Computer Science
Mentor: Andréa Richa, President's Professor
School of Computing and Augmented Intelligence

Introduction & Motivations

Drawing inspiration from the remarkable collective behaviors observed in biological and social systems, our research endeavor revolves around the intricate interplay of weak computational entities operating within dynamic networks.

Specifically, our focus centers on distributed algorithms tailored for dynamic networks characterized by nodes devoid of unique identifiers (**anonymous**), equipped with **sub-logarithmic memory** (incapable of computing identifiers), and relying on **message-passing** between neighboring nodes as the means of communication.



A beautiful real-world example of synchronization within asynchronous networks can be illustrated through the mesmerizing dance of fireflies.

Problem Description

Environment: A time-varying graph that consists of nodes and edges. Nodes can be thought of as the entities that we want to synchronize. Edges are the connections between nodes, and it is the channel they use to communicate with each other.

Synchronizer: Node must have message from all neighbors to move forward. If an edge leaves the network, then clear all messages in the buffer associated to the edge leaving. If an edge joins, nodes send a message when activated to tell the new neighbor of their phase number, the higher of the two will be forced to wait until the lower catches up.

Objective: Show that with our synchronizer applied to the asynchronous environment, there is an equivalent synchronous execution (in terms of network topology and color of nodes).

Synchronous (top) and Asynchronous (bottom) Executions

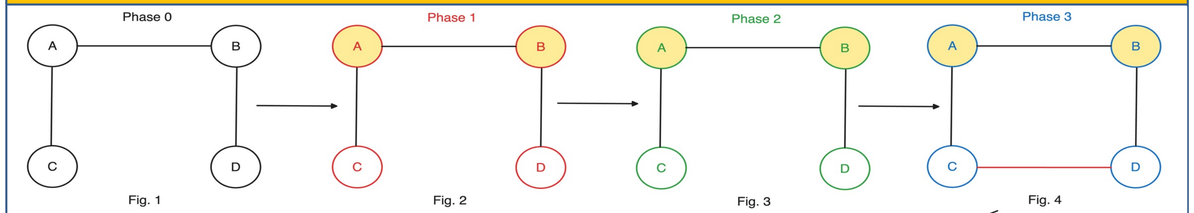


Fig. 1 Fig. 2 Fig. 3 Fig. 4 Fig. 5

Algorithm: If there are 2 or more connected edges to a node, it will determine its color to be yellow. If there are less than 2 edges to a node, it will determine its color to be white.

*** New incoming edges are shown as red.

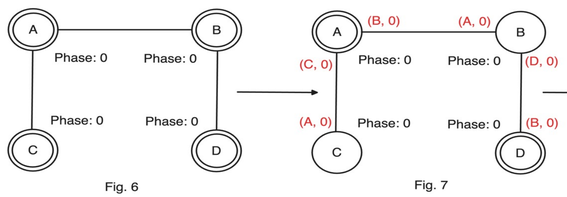


Fig. 6

Fig. 7

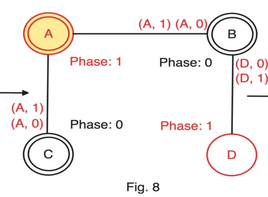


Fig. 8

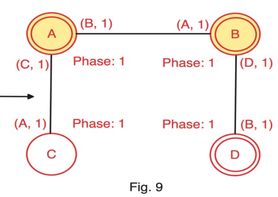


Fig. 9

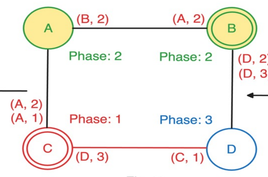


Fig. 11

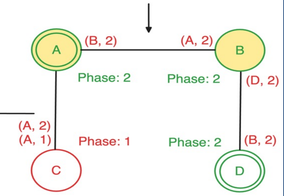


Fig. 10

*** The graph, from Fig. 8 to Fig. 9 can be read as follows: Nodes A, B, C are activated (denoted by double circles), and only nodes B and C have Phase 0 messages incoming from all edge buffers, thus nodes B and C will both consume the current messages in their edge buffers, increment their phase number, recalculate their color, then send a message along their synchronized neighborhood letting those neighbors know of their new current phase number.

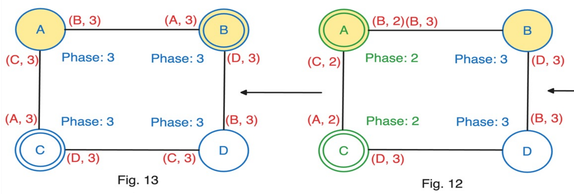


Fig. 13

Fig. 12

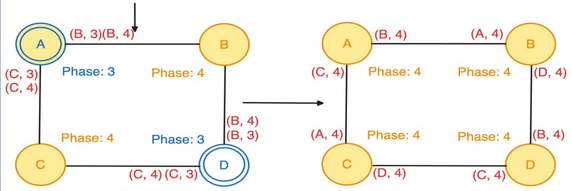


Fig. 14

Fig. 15

Results

The time varying graphs on the left assist our claim that with our synchronizer applied to an asynchronous environment there will be an equivalent execution in a synchronous environment.

Comparison of our synchronizer to other work

Synchronizer	Setting	Anonymous	Local/ Global	Assumptions
Alpha (α) [1,4]	Static	No	Local	Failure free, no node shared memory
Beta (β) [4]	Static	No	Global	Failure free, no node shared memory
Gamma (ϒ) [2,4]	Static	No	Semi	Failure free, no node shared memory
Sigma (σ) [3]	Static	No	Global	Complete network, unbounded memory, knows ID of all nodes
Zeta (ζ) [2]	Dynamic	No	Semi	Failure free, knows ID of nodes in its cluster
Ours	Dynamic	Yes	Local	Knows the status of all its edge ports

Ongoing Work & References

- We will continue to work on this to formally prove correctness and bound the overheads of our synchronizer, as well as expanding upon our simulator with an implementation of our synchronizer to see more results.
- Special thank you to Dr. Richa and PhD student Anya Chaturvedi for their endless help and support throughout this project.

References:
[1] Baruch Awerbuch, Boaz Patt-Shamir, David Peleg, and Michael Saks. 1992. Adapting to asynchronous dynamic networks (extended abstract). In Proceedings of the twenty-fourth annual ACM symposium on Theory of Computing (STOC '92). Association for Computing Machinery, New York, NY, USA, 557–570. https://doi.org/10.1145/129712.129767
[2] B. Awerbuch and M. Sipser, "Dynamic networks are as fast as static networks," [Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science, White Plains, NY, USA, 1988, pp. 206-219, doi: 10.1109/SFCS.1988.21938.
[3] L. Shabtay and A. Segall, "A synchronizer with low memory overhead," *14th International Conference on Distributed Computing Systems*, Poznan, Poland, 1994, pp. 250-257, doi: 10.1109/ICDCS.1994.302420.
[4] Baruch Awerbuch. 1985. Complexity of network synchronization. J. ACM 32, 4 (Oct. 1985), 804–823. https://doi.org/10.1145/4221.4227

FURI

Ira A. Fulton Schools of Engineering
Arizona State University