Computation of Lower Bounds and construction of optimal memory algorithm in Self-Stabilization.

Context

Self-stabilization is a paradigm suited to asynchronous distributed systems prone to transient failures. The occurrence of such a failure (e.g., memory corruption) may move the system to an arbitrary configuration, and an algorithm is self-stabilizing if it guarantees that whenever the system is in a configuration that is illegal with respect to some given boolean predicate Π, the system returns to a legal configuration in finite time (and remains in legal configuration as long as no other failures occur). Introduced in 1974 by Edsger Dijkstra, self-stabilization was promoted by Leslie Lamport (Turing Award winner in 2014) in the 1980s, with the latter considering his promotion of Dijkstra's ideas as 'one of my greatest contributions to computer science'."

We study self-stabilization in networks, the network is modeled as a graph G = (V, E), and we consider predicates defined on labeled graphs. For instance, in proper k-coloring, a configuration is legal if every node is labeled by a color {1, . . . , k} that is different from the colors of all its neighbors. Given a boolean predicate Π, a self-stabilizing algorithm for Π is a distributed algorithm enabling every node, given any input state, to construct a label such that the resulting labeled graph satisfies Π.

During the execution of a self-stabilizing algorithm, the nodes exchange information along the links of the network, and this information is stored locally at every node. Specifically, processes in a distributed system have two types of memory: the persistent memory, and the mutable memory. The persistent memory is used to store the identity of the process (e.g., its IP address), its port numbers, and the code of the algorithm executed on the process. Importantly, this section of the memory is not write enabled during the execution of the algorithm. As a consequence it is less likely to be corruptible, and most work in self-stabilization assumes that this part of the memory is not subject to failures. The mutable memory is used to store the variables used by the algorithm, and is subject to failures, that is, to the corruption of these variables. The space complexity of a self-stabilizing algorithm is the total size of all the variables used by the algorithm, including those used to encode the output label of the node. For instance, the space complexity of the algorithm for k-coloring is at least Ω(log k) bits per node, for encoding the colors in {1, . . . , k}.

Subject

The question addressed in this these is: under which circumstances is it possible to reach a space complexity as low as the size of the labels? And if not, what is the smallest space complexity that can be achieved? More specifically, we will focus on various problems such as:

- What is the minimum memory required to encode pointers to neighbors in order to encode spanning trees (BFS, DFS)?
- What are the minimum bounds for different problems such as tree construction or cluster construction?

We will approach these problems with different metrics such as the number of nodes, the degree of the graph, and the diameter of the graph.

References

- Joffroy Beauquier, Maria Gradinariu, and Colette Johnen. Memory space requirements for self- stabilizing leader election protocols. In 18th Annual ACM Symposium on Principles of Distributed Computing, PODC 1999, pages 199–207, 1999. doi:10.1145/301308.301358.
- Lélia Blin and Pierre Fraigniaud. Space-optimal time-efficient silent self-stabilizing constructions of constrained spanning trees. In 35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, pages 589–598, 2015. doi:10.1109/ICDCS.2015.66.
- Lélia Blin and Sébastien Tixeuil. Compact deterministic self-stabilizing leader election on a ring: the exponential advantage of being talkative. Distributed Computing, 31(2):139–166, 2018. doi:10. 1007/s00446- 017- 0294- 2.
- Lélia Blin and Sébastien Tixeuil. Compact self-stabilizing leader election for general networks. In LATIN 2018: Theoretical Informatics - 13th Latin American Symposium, pages 161–173, 2018. doi:10.1007/978-3-319-77404-6\ 13.
- Lélia Blin,Laurent Feuilloley,and Gabriel Le Bouder. Optimalspacelowerboundfordeterministic self-stabilizing leader election algorithms. In 25th International Conference on Prin- ciples of Distributed Systems, OPODIS 2021, volume 217 of LIPIcs, pages 24:1–24:12, 2021. doi:10.4230/LIPIcs.OPODIS.2021.24.
- Shlomi Dolev, Mohamed G. Gouda, and Marco Schneider. Memory requirements for silent stabi- lization. Acta Inf., 36(6):447–462, 1999. doi:10.1007/s002360050180.