Brief Announcement: Compact Self-stabilizing Leader Election in Arbitrary Graphs*

Lélia Blin[1] and Sébastien Tixeuil[2]

1 **Université d'Evry-Val-d'Essonne, CNRS, LIP6 UMR 7606, France**
`lelia.blin@lip6.fr`

2 **Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, France**
`sebastien.tixeuil@lip6.fr`

Abstract

We present the first self-stabilizing algorithm for leader election in arbitrary topologies whose space complexity is $O(\max\{\log\Delta, \log\log n\})$ bits per node, where n is the network size and Δ its degree. This complexity is sub-logarithmic in n when $\Delta = n^{o(1)}$.

1998 ACM Subject Classification B.8 Performance and Reliability, C.2.1 Distributed Networks, C.2.4 Distributed Systems

Keywords and phrases Leader Election, Self-stabilization, Memory Complexity, Arbitrary Graphs

Digital Object Identifier 10.4230/LIPIcs.DISC.2017.43

1 Context and Motivarion

This paper tackles the problem of designing memory efficient self-stabilizing algorithms for the leader election problem. *Self-stabilization* [5] is a general paradigm to provide recovery capabilities to networks. Intuitively, a protocol is self-stabilizing if it can recover from any transient failure, without external intervention. *Leader election* is one of the fundamental building blocks of distributed computing, as it enables a single node in the network to be distinguished, and thus to perform specific actions. Leader election is especially important in the context of self-stabilization as many protocols for various problems assume that a single leader exists in the network, even after faults occur. *Memory efficiency* relates to the amount of information to be sent to neighboring nodes for enabling stabilization. A small space-complexity induces a smaller amount of information transmission, which *(i)* reduces the overhead of self-stabilization when there are no faults, or after stabilization, and *ii)* facilitates mixing self-stabilization and replication [9].

A foundational result regarding space-complexity in the context of self-stabilizing silent algorithms is due to Dolev et al. [6], stating that in n-node networks, $\Omega(\log n)$ bits of memory per node are required for solving tasks such as leader election. So, only *talkative* algorithms may have $o(\log n)$-bit space-complexity for self-stabilizing leader election. So far, $o(\log n)$-bits solutions only exist for ring shaped networks, and the best protocol to date is due to Blin *et al.* [3], which present a deterministic solution for arbitrary shaped n-rings with $O(\log\log n)$ bits per node.

In general networks, self-stabilizing leader election is tightly connected to self-stabilizing tree-construction. On the one hand, the existence of a leader permits time- and memory-efficient self-stabilizing tree-construction [5]. On the other hand, growing and merging trees

* This work was partially supported by ANR ESTATE.

is the main technique for designing self-stabilizing leader election algorithms in networks, as the leader is often the root of an inward tree [5]. This high space-complexity is due to the implementation of two main techniques, used by all algorithms, and recalled below.

The first technique is the use of a *pointer-to-neighbor* variable, that is meant to designate unambiguously one particular neighbor of every node. For the purpose of tree-construction, pointer-to-neighbor variables are typically used to store the parent node in the constructed tree. Specifically, the parent of every node is designated unambiguously by its identifier, requiring $\Omega(\log n)$ bits for each pointer variable. In principle, it would be possible to reduce the memory to $O(\log \Delta)$ bits per pointer variable in networks with maximum degree Δ, by using node-coloring at distance 2 instead of identifiers to identify neighbors. However, this, in turn, would require the availability of a self-stabilizing distance-2 node-coloring algorithm that uses $o(\log n)$ bits per node. Unfortunately, self-stabilizing distance-2 coloring algorithms [10, 8, 8] use variables of $O(\log n)$ bits per node. To date, no self-stabilizing algorithm implements pointer-to-neighbor variables with space-complexity $o(\log n)$ bits in arbitrary networks.

The second technique for tree-construction or leader election is the use of a *distance* variable that is meant to store the distance of every node to the elected node in the network. Such distance variable is used in self-stabilizing spanning tree-construction for breaking cycles resulting from arbitrary initial state (see [5]). Clearly, storing distances in n-node networks may require $\Omega(\log n)$ bits per node. There are a few self-stabilizing tree-construction algorithms that are not using explicit distance variables (see, e.g., [11, 7, 4]), but their space-complexity is $O(\log n)$ bits per node. Using the general principle of distance variables with space-complexity below $\Theta(\log n)$ bits was attempted by Awerbuch et al. [1], and Blin et al. [2, 3]. These papers distribute pieces of information about the distances to the leader among the nodes according to different mechanisms, enabling to store $o(\log n)$ bits per node, however, these sophisticated mechanisms have only been demonstrated in rings. To date, no self-stabilizing algorithms implement distance variables with space-complexity $o(\log n)$ bits in arbitrary networks.

2 Compact Leader Election

In this “Brief Announcement”, we present a self-stabilizing leader election algorithm with space-complexity $O(\max\{\log \Delta, \log\log n\})$ bits in n-node networks with maximum degree Δ. This algorithm is the first self-stabilizing leader election algorithm for arbitrary networks with space-complexity $o(\log n)$ (whenever $\Delta = n^{o(1)}$). It is designed for the standard state model (a.k.a. shared memory model) for self-stabilizing algorithms in networks.

The design of our algorithm requires overcoming several bottlenecks, including the difficulties of manipulating pointer-to-neighbor and distance variables using $o(\log n)$ bits in arbitrary networks. Overcoming these bottlenecks was achieved thanks to the development of sub-routine algorithms, each deserving independent special interest described hereafter.

First, we generalize to arbitrary networks the results proposed [2, 3] for rings, and aiming at publishing the identifiers in a bit-wise manner. This generalization allows us to manipulate the identifiers with just $O(\log\log n)$ bits of memory per node.

Second, we propose the first *silent* self-stabilizing algorithm for distance-2 coloring that breaks the space-complexity of $\Omega(\log n)$ bits per node. More precisely this new algorithm achieves a space-complexity of $O(\max\{\log \Delta, \log\log n\})$ bits per node. As opposed to previous distance-2 coloring algorithms, we do not use identifiers for encoding pointer-to-neighbor variables, but we use a compact representation of the identifiers to break symmetries. This algorithm allows us to design a compact encoding of spanning trees.

Third, we design a new technique to detect the presence of cycles in the initial configuration resulting from a transient failure. This approach does not use distances, but it is based on the uniqueness of each identifier in the network. Notably, this technique can be implemented by a *silent* self-stabilizing algorithm, with space-complexity $O(\max\{\log\Delta, \log\log n\})$ bits per node.

Last but not least, we design a new technique to avoid the creation of cycles during the execution of the leader election algorithm. Again, this technique does not uses distances but maintains a spanning forest, which eventually reduces to a single spanning tree rooted at the leader at the completion of the leader election algorithm. Implementing this technique results in a self-stabilizing algorithm with space complexity $O(\max\{\log\Delta, \log\log n\})$ bits per node.

Due to space constraints, the details of our approach are presented in the companion technical report available as arXiv:1702.07605 (`https://arxiv.org/abs/1702.07605`).

References

1. B. Awerbuch and R. Ostrovsky. Memory-efficient and self-stabilizing network reset. In *PODC*, pages 254–263. ACM, 1994.
2. L. Blin and S. Tixeuil. Compact deterministic self-stabilizing leader election: The exponential advantage of being talkative. In *Proceedings of the 27th International Conference on Distributed Computing (DISC 2013)*, Lecture Notes in Computer Science (LNCS), pages 76–90. Springer Berlin / Heidelberg, 2013.
3. L. Blin and S. Tixeuil. Compact deterministic self-stabilizing leader election on a ring: The exponential advantage of being talkative. *Distributed Computing*, page to appear, 2017.
4. S. Delaët, B. Ducourthial, and S. Tixeuil. Self-stabilization with r-operators revisited. *Journal of Aerospace Computing, Information, and Communication (JACIC)*, 3(10):498–514, 2006. `doi:10.2514/1.19848`.
5. S. Dolev. *Self-stabilization*. MIT Press, March 2000.
6. S. Dolev, M. G. Gouda, and M. Schneider. Memory requirements for silent stabilization. *Acta Inf.*, 36(6):447–462, 1999.
7. B. Ducourthial and S. Tixeuil. Self-stabilization with path algebra. *Theoretical Computer Science (TCS)*, 293(1):219–236, February 2003. `doi:10.1016/S0304-3975(02)00238-4`.
8. M. Gradinariu and C. Johnen. Self-stabilizing neighborhood unique naming under unfair scheduler. In *Euro-Par 2001: Parallel Processing, 7th International Euro-Par Conference Manchester, UK August 28-31, 2001, Proceedings*, pages 458–465, 2001. `doi:10.1007/3-540-44681-8_67`.
9. T. Herman and S. V. Pemmaraju. Error-detecting codes and fault-containing self-stabilization. *Inf. Process. Lett.*, 73(1-2):41–46, 2000.
10. T. Herman and S. Tixeuil. A distributed tdma slot assignment algorithm for wireless sensor networks. In *Proceedings of the First Workshop on Algorithmic Aspects of Wireless Sensor Networks (AlgoSensors'2004)*, number 3121 in Lecture Notes in Computer Science, pages 45–58, Turku, Finland, July 2004. Springer-Verlag.
11. J.Beauquier, M. Gradinariu, C. Johnen, and J. O. Durand-Lose. Token-based self-stabilizing uniform algorithms. *J. Parallel Distrib. Comput.*, 62(5):899–921, 2002.