

Implementing the Silence to Reduce Energy Consumption in WSN

Karine Altisen

Wafa Badreddine

Stéphane Devismes

Scientific Context

Network technologies have shifted from a limited number of interconnected expensive computers with high performance processing units to networks composed of a huge number of low cost devices with limited processing capabilities. For example, Wireless Sensor Networks (WSNs) are made of a huge amount of tiny devices endowed with limited battery that are able to sense their environment and exchange data using unreliable wireless communication.

WSN's communication primary requirements are to increase reliability, to limit latency and to reduce sensors energy consumption. However, sensors are subject to crash failures because of their limited capacities such as their limited battery. Moreover, their communications use radio channels that are subject to intermittent message losses. In this context, fault-tolerance, i.e., the ability of a distributed algorithm to endure the faults by itself, is mandatory.

Self-stabilization [2] is a versatile lightweight technique to withstand transient faults in a distributed system. After transient faults hit and place the system into some arbitrary global state, a self-stabilizing algorithm resume correct behavior within finite time, and without external intervention.

We focus here on a particular class of self-stabilizing algorithms, called silent algorithms [3]. This class of algorithms is of prime interest, because self-stabilizing solutions for spanning structure [4] or leader election [1] are usually silent. By definition, once a silent algorithm has recovered a correct state, no more computation is needed. Now, this feature has been never exploited until now.

Implementation of self-stabilizing algorithms, in particular silent ones, requires the use of local heartbeat mechanisms: each process should regularly broadcast control packets to its neighbors, even after the system has stabilized. Indeed, heartbeat mechanisms are necessary to detect the occurrence of new transient faults. Now, the use of heartbeats drastically impact the energy consumption of the algorithm. We propose in this internship to exploit the inherent characteristics of silent algorithms to implement more sophisticated and less consuming heartbeat mechanisms. We believe that to enhance these mechanisms, the key idea is the fine-grained management of the timer used by the heartbeat.

Subject

To answer this question, we propose to implement and experiment different silent algorithms with several timer policies. The main difficulty will be to achieve a trade-off between the reactivity of the system to handle transient faults (mainly captured by the notion of stabilization time) and the overhead in messages exchanged after the stabilization. We may take inspiration from the trickle algorithm [5] which dynamically adapts timers to inconsistencies¹ in the network. We may also rely on data traffic to avoid unnecessary signaling.

¹the meaning of "inconsistency" relies on how a protocol uses Trickle

Precisely, the subject requires:

- A bibliographical study: self-stabilizing algorithms (especially silent ones) and trickle timer.
- Implementation: The implementation will be first carried out on a simulator (Omnet++) in a multihop Wireless Sensor Network (WSN).
- Comparison: Solutions will have to be compared according to the stabilization time, the volume of exchanged data during and after the stabilization phase, and the sleeping time of processes.

Required skills

The following abilities are welcomed: C/C++, networking, distributed algorithms.

Working context

The student will be integrated into the SDMA team of MIS Lab² with regular meetings with the Verimag³ team.

Location: Laboratoire MIS - 14 Quai de la Somme, 80080 Amiens

Contact

Wafa.badreddine@u-picardie.fr
stephane.devismes@u-picardie.fr

References

- [1] Karine Altisen, Alain Cournier, Stéphane Devismes, Anaïs Durand, and Franck Petit. Self-stabilizing leader election in polynomial steps. In SSS'2014, 16th International Symposium on Stabilization, Safety, and Security of Distributed Systems, pages 106–119, Paderborn, Germany, Sep 28 - Oct 1 2014. LNCS.
- [2] Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, 1974.
- [3] Shlomi Dolev, Mohamed G. Gouda, and Marco Schneider. Memory requirements for silent stabilization. *Acta Inf.*, 36(6):447–462, 1999.
- [4] Shing-Tsaan Huang and Nian-Shing Chen. A self-stabilizing algorithm for constructing breadth-first trees. *Inf. Process. Lett.*, 41(2):109–117, 1992.
- [5] Philip Levis, T Clausen, Jonathan Hui, Omprakash Gnawali, and J Ko. Rfc6206: The trickle algorithm. Internet Engineering Task Force (IETF) Request For Comments, <http://ietf.org/rfc/rfc6206.txt>, 2011.

²<https://www.mis.u-picardie.fr/equipe/systemes-distribues-mots-applications>

³<http://www-verimag.imag.fr/>