Brief Annoucement: Snap-Stabilization in Message-Passing Systems

Sylvie Delaët
LRI-CNRS UMR 8623
Université de Paris XI, France
delaet@lri.fr

Stéphane Devismes
LRI-CNRS UMR 8623
Université de Paris XI, France
devismes@lri.fr

Mikhail Nesterenko
Computer Science Dept.
Kent State University, USA
mikhail@cs.kent.edu

Sébastien Tixeuil
LIP6-CNRS UMR 7606
Université de Paris VI, France
sebastien.tixeuil@lip6.fr

ABSTRACT

In this announcement, we report recent results where we address the open problem of snap-stabilization in message-passing systems. (A full version of this paper is available as [3] at `http://arxiv.org/abs/0802.1123`.)

Categories and Subject Descriptors: C.2.4 [Distributed Systems]: Distributed applications

General Terms: Algorithms, Reliability

Keywords: Self-stabilization, Snap-Stabilization

1. INTRODUCTION

The concept of *snap-stabilization* [2] offers an attractive approach to transient fault tolerance. As soon as such fault ends a snap-stabilizing protocol *immediately* operates correctly. Of course, not all safety predicates can be guaranteed when the system is started from an arbitrary global state. Snap-stabilization's notion of safety is *user-centric*: when the user initiates a request, then the received response is correct. However, between the request and the response, the system can behave arbitrarily (except from giving an erroneous response to the user). However, all snap-stabilizing protocols presented thus far used a high-atomicity execution model: each process is able to read the states of its neighbors and update its own state in one atomic step. It was unclear if snap-stabilization is possible in more realistic finer atomicity execution models such as message-passing systems.

The contribution of this work is twofold. We first prove that for non-trivial problem specifications, there exists no snap-stabilizing solution in message-passing systems with unbounded yet finite capacity channels. This negative result stands even if the processes have unbounded memory. In contrast, there is a number of self-stabilizing message-passing protocols [4, 1]. We then show that snap-stabilization in the low level message passing model is possible if the channels have bounded capacity.

2. IMPOSSIBILITY RESULTS

We define *safety-distributed* problem specification. We then show that a safety-distributed specification cannot have a snap-stabilizing solution in message-passing systems even if each process is allowed to use an unbounded memory. Since most classical synchronization and resource allocation problems are safety-distributed, this result prohibits the existence of snap-stabilizing protocols in message-passing systems if no further assumption is made.

Intuitively, safety distributed specification has a safety property that depends on the behavior of more than one process. That is, certain process behaviors may satisfy safety if done sequentially, while violate it if done concurrently. For example, in the mutual exclusion problem, each process has to eventually enter the critical section yet not two processes can be in the critical section concurrently.

3. POSSIBILITY RESULTS

We show that snap-stabilization becomes feasible in message-passing systems if the channels are of bounded known message capacity. We present solutions to well-known distributed problems such as *propagation of information with feedback*, *identifier-discovery*, and *mutual exclusion*.

The protocols assume fully-connected networks and use finite local memory at each process. The channels are lossy, bounded and FIFO. The program execution is asynchronous. To ensure non-trivial liveness properties, we make the following fairness assumption: if a sender process s transmits infinitely many messages to a receiver process r then, r receives infinitely many of them. The message that is not lost is received in finite (but unbounded) time. If the channel is full when the message is transmitted, this message is lost. For simplicity, we consider single-message capacity channels. The extension to an arbitrary but known bounded message capacity channels is straightforward.

These positive results imply that the possibility of implementing snap-stabilizing protocols on real networks, and actually implementing them is a future challenge.

4. REFERENCES

[1] Y. Afek and G. M. Brown. Self-stabilization over unreliable communication media. *Distributed Computing*, 7(1):27–34, 1993.

[2] A. Bui, A. K. Datta, F. Petit, and V. Villain. Snap-stabilization and pif in tree networks. *Distributed Computing*, 20(1):3–19, 2007.

[3] S. Delaët, S. Devismes, M. Nesterenko, and S. Tixeuil. Snap-stabilization in message-passing systems. Research Report 6446, INRIA, February 2008.

[4] M. G. Gouda and N. J. Multari. Stabilizing communication protocols. *IEEE Trans. Computers*, 40(4):448–458, 1991.

PODC'08, August 18–21, 2008, Toronto, Ontario, Canada.
ACM 978-1-59593-989-0/08/08.