Non-Volatile Computability

Goal: Characterize the computing power of non-volatile memory models.

Tools: Logic, algorithmic reasoning, programming

Prerequisites: basic knowledge of distributed algorithms, basic concurrent programming skills, curiosity and persistence

Summary

Concurrent programs are often expected to provide safety and progress in asynchronous systems in the presence of *crash failures* [7]: a faulty process prematurely stops taking steps of its algorithm. Recently, a lot of attention has been paid recently to *crash-recovery* models [2] in which a process can resurrected after a crash. This was driven by the emerging *non-volatile memory* in which main conventional memory is equipped with the persistence feature. *Recoverable objects* designed for such models allow their operations to recover from crash failure [1, 5, 6].

The new model forces us to reconsider classical distributed computability results [3, 4, 7, 8], separating computable from not computable for a given model of computation. The goal of this project is to determine the computability bounds in the non-volatile context.

Contact

Prof. Petr Kuznetsov
`http://www.infres.enst.fr/~kuznetso/`
`petr.kuznetsov@telecom-paristech.fr`
INFRES, Télécom ParisTech
Office C213-2, 46 Rue Barrault

References

[1] H. Attiya, O. Ben-Baruch, and D. Hendler. Nesting-safe recoverable linearizability: Modular constructions for non-volatile memory. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018*, pages 7–16, 2018.

[2] H. Boehm and D. R. Chakrabarti. Persistence programming models for non-volatile memory. In *Proceedings of the 2016 ACM SIGPLAN International Symposium on Memory Management, Santa Barbara, CA, USA, June 14 - 14, 2016*, pages 55–67, 2016.

[3] T. D. Chandra, V. Hadzilacos, and S. Toueg. The weakest failure detector for solving consensus. *J. ACM*, 43(4):685–722, July 1996.

[4] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, Apr. 1985.

[5] W. Golab. Recoverable consensus in shared memory. *CoRR*, abs/1804.10597, 2018.

[6] W. M. Golab and D. Hendler. Recoverable mutual exclusion in sub-logarithmic time. In *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 211–220, 2017.

[7] M. Herlihy. Wait-free synchronization. *ACM Trans. Prog. Lang. Syst.*, 13(1):123–149, 1991.

[8] M. Herlihy and N. Shavit. The topological structure of asynchronous computability. *J. ACM*, 46(2):858–923, 1999.