Accepted Manuscript

On the complexity of basic abstractions to implement consensus

Claire Capdevielle, Colette Johnen, Alessia Milani

 PII:
 S0304-3975(18)30005-7

 DOI:
 https://doi.org/10.1016/j.tcs.2017.12.039

 Reference:
 TCS 11442

To appear in: Theoretical Computer Science

Received date:28 March 2017Revised date:15 November 2017Accepted date:27 December 2017



Please cite this article in press as: C. Capdevielle et al., On the complexity of basic abstractions to implement consensus, *Theoret. Comput. Sci.* (2018), https://doi.org/10.1016/j.tcs.2017.12.039

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

ACCEPTED MANUSCRIPT

On the Complexity of Basic Abstractions to Implement Consensus

Claire Capdevielle^a, Colette Johnen^{a,*}, Alessia Milani^a

^aUniv. Bordeaux, LaBRI, UMR 5800, F-33400 Talence, France

5 Abstract

1

2

3

4

⁶ Consensus is one of the central distributed abstractions. By enabling a collection of pro-⁷ cesses to agree on one of the values they propose, consensus can be used to implement any ⁸ generic replicated service in a consistent and fault-tolerant way. Therefore, complexity of ⁹ consensus implementations has become one of the most important topics in the theory of ¹⁰ distributed computing. Several concurrent objects have been proposed as building blocks ¹¹ to implement obstruction-free consensus or wait-free consensus in distributed systems ¹² augmented with failure detectors or strong synchronization primitives.

In this paper we study an important subset of these objects : adopt-commit [1], conflict-detector [2], value-splitter [3] and grafarius [4]. We show that while some of these objects (adopt-commits and conflict-detectors) ensure a superset of the properties ensured by the others (value-splitter and grafarius), their space and individual step complexity is the same if implemented anonymously (the algorithm does not use process IDs). On the other hand, adopt-commit and conflict-detector objects have a larger complexity if we consider non anonymous implementations.

¹³ Keywords: Distributed computing, shared memory, consensus, wait-freedom,

¹⁴ complexity, adopt-commit, conflict-detector, value-splitter, grafarius

15 **1. Introduction**

¹⁶ Consensus is one of the central abstractions in distributed computing since it can be
¹⁷ used to implement any generic replicated service in a consistent and fault-tolerant way.
¹⁸ In particular, consensus requires that a collection of processes agree on one of the values
¹⁹ they propose.

A fundamental result is that consensus cannot be solved deterministically in an asynchronous read-write shared memory system where a process is guaranteed to decide in a *wait-free* manner (in a finite number of its own steps) [6, 7]. The difficulty stems from handling contended executions. Due to the importance of consensus in dependable distribued computing a lot of work has been devoted to studying how to circumvent this impossibility and to compute the complexity of consensus implementations.

colette.johnen@labri.fr (Colette Johnen), alessia.milani@labri.fr (Alessia Milani)

^{\Leftrightarrow} The results of section 3 have been presented in [5], this short paper does not contain any proof. *Corresponding author

Email addresses: claire.capdevielle@labri.fr (Claire Capdevielle),

Download English Version:

https://daneshyari.com/en/article/6875603

Download Persian Version:

https://daneshyari.com/article/6875603

Daneshyari.com