Randomized Wait-Free Consensus Using an Atomicity Assumption

Ling Cheung*

Department of Computer Science, University of Nijmegen
P.O. Box 9010, 6500 GL Nijmegen, The Netherlands

Abstract. We present a randomized algorithm for asynchronous wait-free consensus using multi-writer multi-reader shared registers. This algorithm is based on earlier work by Chor, Israeli and Li (CIL) and is correct under the assumption that processes can perform a random choice and a write operation in one atomic step. The expected total work for our algorithm is shown to be $O(N \log(\log N))$, compared with $O(N^2)$ for the CIL algorithm, and $O(N \log N)$ for the best known weak adversary algorithm. We also model check instances of our algorithm using the probabilistic model checking tool PRISM.

Keywords: Asynchronous Consensus, Randomized Algorithms, Wait-Free Termination, Weak Adversary, Probabilistic Model Checking.

1 Introduction

Distributed consensus refers to a class of problems in which a set of parallel processes exchange messages in order to agree on a common preference. Initially, each process is given an input value from a fixed, finite domain and, at the end of the algorithm, each non-faulty process outputs a decision value. Correctness requirements are typically formulated as follows.

- **Validity**: the output of any non-faulty process must have been the input of some process.
- **Agreement**: all non-faulty processes decide on the same value.
- **Termination**: every non-faulty process decides after a finite number of steps.

As shown in [FLP85], there exists no deterministic algorithm that solves distributed consensus in a setting of asynchronous communication with undetected process failure. Nonetheless, many efficient solutions exist under stronger assumptions (e.g. partial synchrony [DLS88] and failure detection [ACT00]) or weaker correctness requirements (e.g. probabilistic termination [CIL87]).

Our algorithm falls into the category of randomized consensus algorithms, where processes may use coin tosses to determine their course of actions. In this

* Supported by DFG/NWO bilateral cooperation project Validation of Stochastic Systems (VOSS2).
setting, termination is weakened to a probabilistic statement: the set of all non-terminating executions has probability 0. We refer to [Asp03] for a comprehensive overview on randomized consensus.

The first randomized consensus algorithm was proposed by Chor, Israeli and Li [CIL87, CIL94]. It satisfies the following termination condition.

– Probabilistic wait-free termination: with probability 1, each non-faulty process decides after a finite number of steps.

We adopt the same requirement. In fact, the logical structure of our algorithm closely resemble that in [CIL94], while we borrow ideas from [Cha96] to reduce the amount of shared and local data. We shall refer to [CIL94] as the original CIL algorithm and our own as the modified CIL algorithm.

**Adversary Models and Work Bounds.** To prove probabilistic termination, we must reason about probability distributions on the set of executions. These distributions are induced by the so-called adversaries, which are functions from finite histories to available next steps.

The strength of an adversary varies according to the amount of information it can extract from a finite history. The strong adversaries have access to complete history of all processes and shared registers. Some weaker forms, such as write-oblivious and value-oblivious, delay the adversary’s knowledge of outcomes of internal coin tosses. Clearly, a stronger adversary model permits more possibilities and therefore renders consensus more difficult. Consensus against strong adversaries is shown to be $\Omega(N^2/\log^2 N)$ in expected total work, where $N$ is the number of processes participating in the algorithm [Asp98]. The best known algorithms achieve expected $O(N^2 \log N)$ total work [BR91] and $O(N \log^2 N)$ per process [AW96]. Against write-oblivious adversaries, one can achieve expected $O(\log N)$ per process work and $O(N \log N)$ total work [Aum97]. Against value-oblivious adversaries, the fastest algorithm is $O(N \log N e^{\sqrt{\log N}})$ in a single-writer single-reader (SWSR) setting [AKL99].

Our adversary model takes the form of an atomicity assumption: processes can perform a random choice and a write operation in one atomic step. In particular, the process increments its round number if and only if the coin lands heads; then immediately it writes 1 to the memory location $\text{mem}(r, v)$, where $r$ is the round number after the coin toss and $v$ is the current preference. This amounts to saying that the adversary cannot distinguish between the two locations $\text{mem}(r, v)$ and $\text{mem}(r + 1, v)$. The original CIL algorithm relies on a similar atomicity assumption [CIL94] and achieves expected $O(N^2)$ total work [CIL94]. In the present paper, we replace the single-writer multiple-reader (SWMR) registers of [CIL94] with multi-writer multi-reader (MWMR) registers, thereby reducing the expected total work to $O(N \log(\log N))$.

1 This is faster than other value-oblivious algorithms because SWSR is a weak primitive. More discussion can be found in Section 7.

2 The assumption in [CIL94] says that the adversary cannot distinguish between the values $r$ and $r + 1$ as they are written to the same memory location.