Brief Announcement: On the Uncontended Complexity of Anonymous Consensus*

Claire Capdevielle[1], Colette Johnen[1], Petr Kuznetsov[2], and Alessia Milani[1]

[1] Univ. Bordeaux, LaBRI, UMR 5800, F-33400 Talence, France
[2] Télécom ParisTech

Consensus is one of the central distributed abstractions. By enabling a collection of processes to agree on one of the values they propose, consensus can be used to implement any generic replicated service in a consistent and fault-tolerant way. Therefore, complexity of consensus implementations has become one of the most important topics in the theory of distributed computing.

It is known that consensus cannot be solved in an asynchronous read-write shared memory system in a deterministic and fault-tolerant way. One way to circumvent this impossibility is to only guarantee progress (using reads and writes) in executions meeting certain conditions, e.g., in the absence of *contention*. Alternatively, a process is guaranteed to decide in the *wait-free* manner, but stronger (and more expensive) synchronization primitives, such as *compare-and-swap*, can be applied in the presence of contention.

We are interested in consensus algorithms in which a *propose* operation is allowed to apply primitives other than reads and writes on the base objects only in the presence of *interval contention*, i.e., when another *propose* operation is concurrently active. Such algorithms are called *interval-solo-fast*.

Ideally, interval-solo-fast algorithms should have an optimized behavior in *uncontended* executions. It appears therefore natural to explore the uncontended complexity of consensus algorithms: how many memory operations (reads and writes) can be performed and how many distinct memory locations can be accessed in the absence of interval contention?

In general, interval-solo-fast consensus can be solved with only *constant* uncontended complexity. We therefore restrict our study to *anonymous* consensus algorithms, i.e., algorithms not using process identifiers and, thus, programming all processes identically. Besides intellectual curiosity, practical reasons to study anonymous algorithms in the shared memory model are discussed in [GR07].

Our results. On the lower-bound side, we show that any anonymous interval-solo-fast consensus algorithm exhibits non-trivial uncontended complexity that depends on n, the number of processes, and m, where m is the size of the set of input values that can be proposed. More precisely, we show that, in the worst case, a *propose* operation running *solo*, i.e., without any other process invoking *propose*, must write to $\Omega(\min(\sqrt{n}, \log m/\log\log m))$ distinct memory locations.

* Partially supported by the ANR project DISPLEXITY (ANR-11-BS02-014). This study has been carried out in the frame of the Investments for the future Programme IdEx Bordeaux- CPU (ANR-10-IDEX-03-02). The third author was supported by the Agence Nationale de la Recherche, under grant agreement N ANR-14-CE35-0010-01, project DISCMAT.

This metrics, called *solo-write complexity*, is upper-bounded by *step complexity* of the algorithm, i.e., the worst-case number of all primitives applied by an individual *propose* operation. In the special case of *input-oblivious* algorithms, where the sequence of memory locations written in a solo execution does not depend on the input value, we derive a stronger lower bound of $\Omega(\sqrt{n})$ on solo-write complexity. Our proof only requires the algorithm to ensure that operations terminate in solo executions, so the lower bounds also hold for *abortable* and *obstruction-free* consensus implementations.

On the positive side, we show that our lower bound is tight. Our matching consensus algorithm is based on our novel *value-splitter* abstraction, extending the classical *splitter* mechanism, interesting in its own right. Informally, a value-splitter exports a single operation *split* that takes a value in a value set V as a parameter and returns a boolean response so that (1) if a *split* operation completes before any other *split* operation starts, then it returns *true*, and (2) all *split* operations that return *true* were invoked with the same parameter value.

Our consensus algorithm adapts the classical splitter-based algorithm [LMS03] to the new value-splitter abstraction. We have implemented an anonymous and input-oblivious value-splitter that exhibits $O(\sqrt{n})$ space and solo-write complexity. Also we have slightly modified the *weak conflict detector* proposed in [AE14], to implement a regular (not input-oblivious) value-splitter exhibiting space and step complexity of $O(\log m / \log\log m)$. These two value-splitter read-write implementations combined with our consensus algorithm provide the matching upper bound.

Our results are summarized in Table 1 and detailed in [CJKM]. Overall,

Input-oblivious	Not input-oblivious	
$\Omega(\sqrt{n})$	$\Omega(\min(\sqrt{n}, \frac{\log m}{\log\log m}))$	
$O(\sqrt{n})$	if $\sqrt{n} \leq \frac{\log m}{\log\log m}$, $O(\sqrt{n})$	if $\sqrt{n} \geq \frac{\log m}{\log\log m}$, $O(\frac{\log m}{\log\log m})$ [LMS03,AE14]

Table 1. Space and solo-write complexity for anonymous interval-solo-fast consensus

they imply the first nontrivial *tight* lower bound on the space complexity for consensus known so far. They also show that there is an inherent gap between anonymous and non-anonymous consensus algorithms.

References

[AE14] J. Aspnes and F. Ellen. Tight bounds for adopt-commit objects. *Theory of Computing Systems*, 55(3):451–474, 2014.

[CJKM] C. Capdevielle, C. Johnen, P. Kuznetsov, and A. Milani. On the uncontended complexity of anonymous consensus. https://hal.archives-ouvertes.fr/hal-01180864/document.

[GR07] R. Guerraoui and E. Ruppert. Anonymous and fault-tolerant shared-memory computing. *Distributed Computing*, 20(3):165–177, 2007.

[LMS03] V. Luchangco, M. Moir, and N. Shavit. On the uncontended complexity of consensus. In *17th International Symposium on Distributed Computing*, (DISC'03), pages 45–59, 2003.