Brief Announcement: Easy Impossibility Proofs for k-Set Agreement in Message Passing Systems*

Martin Biely
Embedded Computing Systems Group (E182/2)
Technische Universität Wien, Treitlstrasse
1040 Vienna, Austria
martin.biely@epfl.ch

Peter Robinson
Nanyang Technological University
Division of Mathematical Sciences
Singapore 637371
peter.robinson@ntu.edu.sg

Ulrich Schmid
Embedded Computing Systems Group (E182/2)
Technische Universität Wien, Treitlstrasse
1040 Vienna, Austria
s@ecs.tuwien.ac.at

Categories and Subject Descriptors: C.4 [Computer Systems Organization]: Performance of Systems — *Fault Tolerance*

General Terms: Algorithms,Reliability,Theory.

1. INTRODUCTION

We study distributed algorithms that solve agreement problems, namely, k-set agreement. Their purpose is to compute and irrevocably set the output y_p of process p to some decision value, based on the proposal values $x_q \in V$, for $1 \leqslant q \leqslant n$ and $|V| \geqslant n$,[1] such that every correct process eventually decides on a value proposed by some process and the set of all decision values contains at most k values.

Despite of being quite similar agreement problems, consensus (= 1-set agreement) and general k-set agreement require surprisingly different techniques for proving the impossibility in asynchronous systems with crash failures: Rather than relatively simple bivalence arguments as in the impossibility proof for consensus in the presence of a single crash failure ($f = k = 1$), known proofs for the impossibility of k-set agreement in systems with $f \geqslant k > 1$ crash failures use algebraic topology or a variant of Sperner's Lemma.

In this paper, we present a generic theorem for proving the impossibility of k-set agreement in various message passing settings, which is based on a simple reduction to the consensus impossibility in a certain subsystem.

2. RESTRICTIONS OF ALGORITHMS AND INDISTINGUISHABILITY OF RUNS

We will occasionally use a subsystem $\mathcal{M}'$ that is a *restriction* of a model $\mathcal{M}$, in the sense that it consists of a subset of processes in Π, while using the same mode of computation (atomicity of computing steps, time-driven vs. message-driven, etc.) as $\mathcal{M}$. We make this explicit by using the notation $\mathcal{M} = \langle \Pi \rangle$ and $\mathcal{M}' = \langle D \rangle$, for some set of processes $D \subseteq \Pi$. Note that this definition does not imply anything about the synchrony assumptions which hold in $\mathcal{M}'$. All that is required is that $\mathcal{M}'$ is computationally compatible with $\mathcal{M}$: Any algorithm designed for $\mathcal{M}$ can also be run in $\mathcal{M}'$, albeit on a smaller set of processes.

Let A be an algorithm that works in system $\mathcal{M} = \langle \Pi \rangle$ and let $D \subseteq \Pi$ be a nonempty set of processes. Consider a restricted system $\mathcal{M}' = \langle D \rangle$. The *restricted algorithm* $A_{|D}$ for system $\mathcal{M}'$ is constructed by dropping all messages sent to processes outside D in the message sending function of A, obtaining the message sending function of $A_{|D}$.

Note that we do not change the actual code of algorithm A in any way. In particular, the restricted algorithm still uses the value of $|\Pi|$ for the size of the system, even though the real size of D might be much smaller.

We will use a concept of similarity/indistinguishability of runs that is slightly weaker than the usual notion [2, Page 21], as we require the same states only *until* a decision state is reached. This makes a difference for algorithms where p can help others in reaching their decision after p has decided, for example, by forwarding messages.

Two runs α *and* β *are indistinguishable (until decision) for a process* p, if p has the same sequence of states in α and β until p decides. By $\alpha \overset{D}{\sim} \beta$ we denote the fact that α and β are indistinguishable (until decision) for every $p \in D$.

Let $\mathcal{R}$ and $\mathcal{R}'$ be sets of runs. We say that *runs* $\mathcal{R}'$ *are compatible with runs* $\mathcal{R}$ *for processes in* D, denoted by $\mathcal{R}' \preccurlyeq_D \mathcal{R}$, if $\forall \alpha \in \mathcal{R}'\ \exists \beta \in \mathcal{R} \colon \alpha \overset{D}{\sim} \beta$.

3. THE IMPOSSIBILITY THEOREM

In this section, we will present our general k-set agreement impossibility theorem. Due to its very broad applicability, the theorem itself is stated in a highly generic and somewhat abstract way. It captures a reasonably simple idea, however, which boils down to extracting a consensus algorithm for a certain subsystem where consensus is unsolvable: Suppose that a given k-set agreement algorithm A for some system model $\mathcal{M}$ has runs, where processes start with distinct values and k partitions $D_1, \ldots, D_{k-1}$ and $\overline{D}$ can be formed: Processes in the $k-1$ partitions D_i decide on (at least) $k-1$ different values, and no process in partition $\overline{D}$ ever hears from any process in D_i before it decides. Note carefully that processes in $\overline{D}$ can communicate arbitrarily within $\overline{D}$. Then, the ability of A to solve k-set agreement would imply that the restricted algorithm $A_{|\overline{D}}$ can solve consensus in the

*This work has been supported by the Austrian Science Foundation (FWF) project P20529.
Peter Robinson has also been supported by Nanyang Technological University grant M58110000.

[1]The assumption $|V| \geqslant n$ allows runs where all processes start with different propose values.

PODC'11, June 6–8, 2011, San Jose, California, USA.
ACM 978-1-4503-0719-2/11/06.

restricted model $\mathcal{M}' = \langle \overline{D} \rangle$. However, if the synchrony and failure assumptions of $\mathcal{M}$ are such that consensus cannot be solved in $\mathcal{M}'$, this is a contradiction.

THEOREM 1 (k-SET AGREEMENT IMPOSSIBILITY). *Let $\mathcal{M} = \langle \Pi \rangle$ be a system model and consider the runs $\mathcal{M}_A$ that are generated by some fixed k-set agreement algorithm A in $\mathcal{M}$, where every process starts with a distinct input value. Fix some nonempty disjoint sets of processes $D_1, \ldots, D_{k-1}$, and a set of distinct decision values $\{v_1, \ldots, v_{k-1}\}$. Moreover, let $D = \bigcup_{1 \leqslant i < k} D_i$ and $\overline{D} = \Pi \setminus D$. Consider the following two properties:*

(dec-D) *For every set D_i, value v_i was proposed by some process in D, and there is some process in D_i that decides on v_i.*

(dec-$\overline{D}$) *If $p_j \in \overline{D}$ then p_j receives no messages from any process in D until after every process in $\overline{D}$ has decided.*

Let $\mathcal{R}_{(\overline{D})} \subseteq \mathcal{M}_A$ and $\mathcal{R}_{(D,\overline{D})} \subseteq \mathcal{M}_A$ be the sets of runs of A where (dec-$\overline{D}$) respectively both, (dec-D) and (dec-$\overline{D}$), hold.Suppose that the following conditions are satisfied:

(A) *$\mathcal{R}_{(\overline{D})}$ is nonempty.*

(B) *$\mathcal{R}_{(\overline{D})} \preccurlyeq_{\overline{D}} \mathcal{R}_{(D,\overline{D})}$.*

In addition, consider a restricted model $\mathcal{M}' = \langle \overline{D} \rangle$ such that the following hold:

(C) *There is no algorithm that solves consensus in $\mathcal{M}'$.*

(D) *$\mathcal{M}'_{A_{|\overline{D}}} \preccurlyeq_{\overline{D}} \mathcal{M}_A$.*

Then, A does not solve k-set agreement in $\mathcal{M}$.

PROOF. For the sake of a contradiction, assume that there is a k-set agreement algorithm A for model $\mathcal{M}$, sets of runs $\mathcal{R}_{(\overline{D})}$ and $\mathcal{R}_{(D,\overline{D})}$ and some sets of processes $D_1, \ldots, D_{k-1}$ such that conditions (A)–(D) hold. Due to (A) we have $\mathcal{R}_{(\overline{D})} \neq \emptyset$; then, (B) implies that $\mathcal{R}_{(D,\overline{D})}$ is nonempty too. Observe that (dec-D) ensures that there are $\geqslant k-1$ distinct decision values among the processes in D, in every run in $\mathcal{R}_{(D,\overline{D})}$. Since algorithm A satisfies k-agreement, the compatibility requirement (B) between runs $\mathcal{R}_{(\overline{D})}$ and $\mathcal{R}_{(D,\overline{D})}$ for processes in $\overline{D}$ implies the following constraint:

(Fact 1) *In each run in $\mathcal{R}_{(\overline{D})}$, all processes in $\overline{D}$ must decide on a common value.*

We will now show that this fact yields a contradiction. Starting from $\mathcal{M}'_{A_{|\overline{D}}}$, i.e., the set of runs of the restricted algorithm in model $\mathcal{M}'$, we know by (D) that for each $\rho' \in \mathcal{M}'_{A_{|\overline{D}}}$, there exists a run $\rho \in \mathcal{M}_A$ such that $\rho' \overset{\overline{D}}{\sim} \rho$. Obviously, no process $p \in \overline{D}$ receives messages from a process $q \in D$ in ρ' before p's decision, as such a process q does not exist in the restricted model $\mathcal{M}'$. Clearly, the same is true for the indistinguishable run ρ (even though such a process q does exist in model $\mathcal{M}$). Therefore, we have that, in fact, $\rho \in \mathcal{R}_{(\overline{D})}$, and due to (Fact 1), we know that in each run $\rho' \in \mathcal{M}'_{A_{|\overline{D}}}$ all processes decide on the same value. This, however, means that we could employ $A_{|\overline{D}}$ to solve consensus in $\mathcal{M}'$, which is a contradiction to (C). □

The proof of Theorem 1 neither restricts the types of failures that can occur in $\mathcal{M}$ nor the underlying synchrony assumptions of $\mathcal{M}$ in any way. Moreover, our impossibility argument uses a 2-partitioning argument but does *not* require the system to (temporarily or permanently) decompose into $k+1$ partitions. In particular, there is no further restriction on the communication among processes within D and within $\overline{D}$. Despite its main purpose of showing impossibilities, our theorem is also useful when developing new algorithms for achieving k-set agreement. For example, suppose that we are given some unproven but seemingly promising new algorithm A for a model close to asynchrony. Then, checking whether the runs of A are such that the conditions of Theorem 1 are satisfied will allow us to determine already at an early stage (i.e., before developing a detailed correctness analysis) whether it is worthwhile to explore A further. In particular, if (dec-D) can be satisfied in some runs, i.e., (A) holds, the algorithms is very likely flawed, as the remaining conditions are typically easy to construct in sufficiently asynchronous systems.

4. DERIVED RESULTS

In the full paper [1], we have used our result to derive impossibility results both for partially synchronous systems and for asynchronous systems augmented with failure detectors:

THEOREM 2. *There is no algorithm that solves k-set agreement in a system $\mathcal{M}$ of n processes where*

- *processes are synchronous,*
- *communication is asynchronous,*
- *a process can broadcast a message in an atomic step, and*
- *receiving and sending are part of the same atomic step,*

for any

$$k \leqslant \frac{n-1}{n-f}, \tag{1}$$

even if, of the f possibly faulty processes, $f-1$ can fail by crashing initially and only one process can crash during the execution.

THEOREM 3. *There is no $(n-1)$-resilient algorithm that solves k-set agreement in an asynchronous system with failure detector (Σ_k, Ω_k), for all $2 \leqslant k \leqslant n-2$.*

5. CONCLUSION

The main advantage of our approach is that we are independent of a specific system model, since Theorem 1 neither makes assumptions on the available amount of synchrony, nor on the power of computing steps and communication primitives available to the processes. This genericity allows to apply our theorem in very different contexts.

6. REFERENCES

[1] Martin Biely, Peter Robinson, and Ulrich Schmid, *Easy impossibility proofs for k-set agreement in message passing systems*, Research Report 2/2011, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2011, available at ArXiv `http://arxiv.org/abs/1103.3671`.

[2] Nancy Lynch, *Distributed algorithms*, Morgan Kaufman Publishers, Inc., San Francisco, USA, 1996.