Brief Announcement: Towards the Minimal Synchrony for Byzantine Consensus

Achour Mostefaoui, and Gilles Trédan
IRISA/IFSIC, University of Rennes
35042 Rennes, France
achour@irisa.fr, gtredan@irisa.fr

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of Systems—*Fault tolerance*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed applications*; D.4.1 [**Operating Systems**]: Process Management—*concurrency, multiprocessing, synchronization*; D.4.6 [**Operating Systems**]: Security and Protection—*Unauthorized access, Authentication*

General Terms

Algorithms,Reliability,Theory

Keywords

Authentication, Asynchronous Distributed System, Byzantine Process, Consensus, Distributed Algorithm, Eventually Timely Link, Fault-Tolerance, Resilience

In the Consensus problem, each process proposes a value, and the non-faulty processes have to eventually decide (termination property) on the same output value (agreement property) that should be a proposed value (validity property). This problem, whose statement is particularly simple, is fundamental in fault-tolerant asynchronous distributed computing as it abstracts several basic agreement problems. Unfortunately, the Consensus problem has no deterministic solution in asynchronous distributed systems where even a single process can crash. A process crashes if it simply stops its execution (fail-stop process). Otherwise a faulty process can exhibit an arbitrary behavior. Such a process is called *Byzantine*. This bad behavior can be intentional (malicious behavior, e.g., due to intrusion) or simply the result of a transient fault that altered the local state of the process, thereby modifying its behavior in an unpredictable way. We are interested here in solving agreement problems (more precisely, the *Consensus* problem) in asynchronous distributed systems prone to byzantine process failures whatever their origin. For byzantine failures, byzantine processes may propose unexpected values, the consensus validity property has to be changed into: if all correct processes propose the same value v then only v can be decided.

To allow deterministic solutions to the consensus problem [4], asynchronous systems need to be enriched with additional synchrony assumptions. In the context of crash failures, this approach has been abstracted in the notion of unreliable failure detectors [3]. A failure detector can be seen as a distributed oracle that gives (possibly incorrect) hints about which processes have crashed so far. Nearly all implementations of failure detectors consider that, eventually, the underlying system behaves in a synchronous way. More precisely, they consider the *partially synchronous system* model that assumes there are bounds on process speeds and message transfer delays, but these bounds are not known and hold only after some finite but unknown time (called *Global Stabilization Time*).

In the crash failure model, a weakest failure detector has been proposed for the consensus problem, however this weakest condition has not been translated to a lower bound on the communication sub-system. For a system of n partially synchronous processes where at most t may crash, researchers defined many models [1, 6, 5] that relax more and more link synchrony assumptions. In this setting, a link between two processes is said to be timely at time τ if a message sent at time τ is received not later than $\tau + \delta$. The bound δ in not known and holds only after some finite but unknown time τ_{GST} (called *Global Stabilization Time*). A link is called eventually timely if it is timely at all times $\tau \geq \tau_{GST}$. The considered system models assume at least one correct process with t outgoing eventually timely links. Such a process is called an $\diamond t$-source. This is conjectured to be a lower bound.

In the context of byzantine processes, [2] proposes a system model that allows to cope with at most t byzantine failures when solving the consensus problem. Namely, it assumes at least one correct process with all its outgoing and incoming links eventually timely and all the links are reliable. Such a process is called an $\diamond$ bisource. This means that the number of eventually timely links could be as low as $2(n-1)$. Their protocol does not need authentication but uses a very costly broadcast procedure similar to the consistent broadcast and the authenticated broadcast procedures. Each round is made up of 10 communication steps and needs $O(n^3)$ messages.

Although an $\diamond$ bisource appears to be weak, an interesting open problem is to prove that this is a lower bound on the number of timely links or to propose a weaker system that allows to solve the byzantine consensus. Hence the following question:

> *"In a partially synchronous distributed systems with fair lossy links links and prone to process failures (at most t processes may experience a byzantine behavior), can the byzantine consensus problem be deterministically solved if only few*

PODC'07, August 12–15, 2007, Portland, Oregon, USA.
ACM 978-1-59593-616-5/07/0008.

links are eventually timely. More precisely, there exist an $\diamond x$ bisource where $x(< n-1)$?"

The paper [7] shows that the answer to this question is "*yes*". It first presents a parametrized system model whose parameter x represents the scope of the bisource. Indeed, the bisource assumed by [2] has a maximal scope ($x = n-1$). Informally, an x-bisource is a correct process where the numbers of privileged neighbors is x instead of $(n-1)$. The bisource is eventual in the property holds only after some finite but unknown time. In this system model, a byzantine consensus protocol is proposed. It uses authentication and assumes an $\diamond 2t$-bisource that we expect to be a tight bound. We assume $t < n/3$ meeting the resiliency lower bound byzantine consensus. The proposed protocol enjoys the nice property of being very simple and elegant in its design. Moreover, in good settings, the decision is reached within 5 communication steps whatever is the behavior of byzantine processes. Good settings occur either when all correct processes propose the same value whatever is the behavior of the byzantine processes, or when the first coordinator is a $2t$-bisource (and behaves correctly).

1. PROPOSED COMPUTATION MODEL

The system model is patterned after the partially synchronous system described in [4]. The system is made up of a finite set Π of n $(n > 1)$ fully-connected processes, namely, $\Pi = \{p_1, \ldots, p_n\}$. Moreover, up to t processes can exhibit a *Byzantine* behavior. We assume that processes are partially synchronous, in the sense that every non-crashed process takes at least one step every θ steps of the fastest process (θ is unknown). Instead of real-time clocks, time is measured in multiples of the steps of the fastest process like in [4]. In particular, the (unknown) delay bound δ is such that any process can take at most δ steps while a timely message is in transit. Hence, we can use simple step-counting for timing out messages. The communication network is unreliable: messages can be lost, reordered, or duplicated. However, if a message is sent infinitely many times then it arrives at its destination infinitely many times. Moreover, messages are altered and the receiver of a message knows who the sender is. In other words, we are using authenticated asynchronous fair-lossy links.

DEFINITION 1. *A link from process p to process q is eventually at time τ if no message sent by p at time τ is received at q after time $(\tau + \delta)$.*

Note that the definition above does not require the receiver to be correct; in such a case, the link is considered timely.

DEFINITION 2. *A process p is an x-bisource at time τ if:*
- (1) There exist a set X of x processes, such for any process q in X both links from p to q and from q to p are timely at time τ. The processes of X as said to be privileged neighbors of p
- (2) Process p takes at least one step during the θ_X next steps of the fastest process in X.

DEFINITION 3. *A process p is an $\diamond x$-bisource if it is correct and there is a time τ such that, for all $\tau' \geq \tau$, p is an x-bisource.*

The paper considers a partially synchronous system where the only assumed synchrony properties are those needed by the $\diamond x$-bisource. This means that all the links that do not participate in the $\diamond x$-bisource could be asynchronous fair lossy all the time and all the other processes can have finite but unbounded relative speeds. This assumption is extremely weak when compared to existing assumptions.

2. AUTHENTICATION

To achieve such a low number of eventually timely links (an $\diamond 2t$-bisource), the proposed protocol uses authentication. A process may be byzantine and disseminate a wrong value (different from the value it would have obtained if it behaved correctly). To prevent such a dissemination, the protocol uses certificates. We assume that message carry certificates we call it a certified message (although this is not mentioned in the protocol) and that the byzantine processes are not able to subvert the cryptographic primitives. This implies the use of application level signatures (public key cryptography such as RSA signatures). A straightforward implementation of certificates would consist of including a set of signed messages received in a previous message exchange.

It proceeds in consecutive asynchronous rounds each coordinated by a deterministically fixed coordinator to provide processes with the same value. Each round is composed of four communication phases. During the first phase processes communicate with the coordinator of the round and use a time-out for preventing from deadly blocking waiting a silent leader. If the coordinator is a $(2t)$-bisource, then at least $(2t + 1)$ processes will receive the value of the coordinator. among those $(2t + 1)$ processes, at least $(t + 1)$ are correct processes. Thus a simple exchange allows each process to get the value of the coordinator. However, as the $(2t)$-bisource is only eventual and as the id of the bisource is not a priori known, it is necessary to have filter/lock mechanism as used by rotating coordinator failure detector-based protocols.

3. REFERENCES

[1] Aguilera M.K., Delporte-Gallet C., Fauconnier H. and Toueg S, Communication-efficient leader election and consensus with limited link synchrony. *Proc. 23nd ACM PODC*, 2004.

[2] Aguilera M.K., Delporte-Gallet C., Fauconnier H. and Toueg S, Consensus with byzantine failures and little system synchrony. *Proc. DSN*, 2006.

[3] Chandra T.D. and Toueg S., Unreliable Failure Detectors for Reliable Distributed Systems. *JACM*, 43(2):225-267, 1996.

[4] Dwork C., Lynch N.A. and Stockmeyer L., Consensus in the presence of partial synchrony. *JACM*, 35(2):288-323, 1988.

[5] Fernandez A., and Raynal M., From an intermittent rotating star to a leader. *Tech Report*, IRISA, Université de Rennes 1, (France), 2007. http://www.irisa.fr/doccenter/publis/PI#2007

[6] Hutle M., Malkhi D., Schmid U., and Zhou L., Chasing the Weakest System Model for Implementing. *Research Report 74/2005, Technische Universität Wien, Institut für Technische Informatik*, July, 2006.

[7] Hamouma M., Mostefaoui A. and Tredan G., Byzantine Consensus with Very Few Synchrony. *Tech Report*, IRISA, Université de Rennes 1, (France), 2007. http://www.irisa.fr/doccenter/publis/PI#2007