Non-blocking Asynchronous Byzantine Quorum Systems

Rida A. Bazzi

Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287-5406 U.S.A.
bazzi@asu.edu

Abstract. Quorum systems have been used to implement many coordination problems in distributed systems. In this paper, we propose a reformulation of the definition of Byzantine quorum systems. Our reformulation captures the requirement for non-blocking access to quorums in asynchronous systems. We formally define the asynchronous access cost of quorum systems and we show that the asynchronous access cost and not the size of a quorum is the right measure of message complexity of protocols using quorums in asynchronous systems. We also show that previous quorum systems proposed in the literature have a very high asynchronous access cost. We present new quorum systems with low asynchronous access cost and whose other performance parameters match those of the best Byzantine quorum systems proposed in the literature. In particular, we present a construction for the disjoint failure pattern that outperforms previously proposed systems for that pattern.

Keywords: quorum, tolerance, Byzantine, failures, distributed, asynchronous, access cost.

1 Introduction

A quorum system is a collection of sets (quorums) that mutually intersect. Quorum systems have been used to implement mutual exclusion [1,8], replicated data systems [7], commit protocols [15], and distributed consensus [11]. For example, in a typical implementation of mutual exclusion using a quorum system, processors request access to the critical section from all members of a quorum. A processor can enter its critical section only if it receives permission from all processors in a quorum.\(^1\) Work on quorum systems traditionally considered crash failures [1,2,4,5,6,7,8,14,13]. Malkhi and Reiter [9] proposed the interesting notion of Byzantine quorums - quorum systems that can tolerate Byzantine failures. They showed that the traditional definition of quorums is not adequate to handle Byzantine failures: in the presence of Byzantine failures, the intersection of two quorums should contain enough correct processors so that correct failures are not possible.

\(^1\) Additional measures are needed to insure that the implementation is fair and deadlock free.
processors can unambiguously access the quorums. They presented protocols to implement a distributed shared register variable using Byzantine quorums. Their implementation requires a client accessing a quorum to wait for responses from every processor in a quorum set, but they did not study the problem of finding a quorum set whose elements are available - an available quorum.

In this paper, we study the cost of finding an available quorum in the presence of Byzantine failures. We introduce non-blocking Byzantine quorum systems and show that they can be achieved at a low cost and we present non-blocking Byzantine quorum constructions for two failure models. The constructions we present are the first that do not require blocking and that have a low cost. Also, the construction we present for the Disjoint failure model yields a Byzantine quorum system that has better performance parameters than previously proposed systems. Our construction rely on a new access model we call partial access. With partial access, a processor need not wait for a reply from each process in a quorum set. The quorum system should be designed to ensure that any two partial accesses have a large enough intersection to ensure consistency. It turns out that the set of partial accesses of a non-blocking Byzantine quorum system is a Byzantine quorum system as defined in [9].

It should be emphasized that partial access is relevant for general quorum access in asynchronous systems and is not only relevant to asynchronous Byzantine quorum systems. In this paper, we only consider partial access for the case of Byzantine failures. The same methods we propose are applicable to cases in which the failures are not Byzantine, but that are restricted to a predefined failure pattern. A probabilistic model of failure does not have a predefined failure pattern and therefore our results are not directly applicable to it.

The rest of the paper is organized as follows. Section 2 discusses related work and Section 3 summarizes our contributions. Section 4 presents basic definitions and introduces the notion of asynchronous access cost. Section 5 gives examples of the asynchronous access cost of two Byzantine quorum systems. Section 6 reformulates the definition of Byzantine quorums to capture the asynchronous access cost as a design objective. Section 7 presents non-blocking quorum systems with low asynchronous access cost and whose other performance parameters match those of the best Byzantine quorum systems proposed in the literature. Section 8 concludes the paper.

2 Related Work

The problem of finding an available quorum has been addressed by researchers for the case of detectable crash failures [2,13]. In [13], the probe complexity of a quorum system is defined. The probe complexity is the minimum number of processors that need to be contacted to establish the existence or non-existence of an available quorum. In the definition of probe complexity, processors can be probed incrementally and the identity of the processor to be probed next can depend on the responses received from previous probes. In [2], the author formally defined the concept of cost of failures, which can be thought of as