EFFICIENT DISTRIBUTED ALGORITHMS BY USING
THE ARCHIMEDEAN TIME ASSUMPTION.*

by
Paul Spirakis(1),(2) and Basil Tampakas(1)
(1) Computer Tech. Institute, Greece, and
(2) Courant Inst. Math. Sciences, USA

Abstract
This work examines the effect of limited asynchrony on three fundamental problems of distributed computation: The problem of symmetry breaking in a logical ring, that of mutual exclusion and the problem of readers and writers. We assume our distributed system to be Archimedean in the sense that processors know upper and lower bounds on the message delays and processor speeds. We use the knowledge of those bounds to get algorithms for the above mentioned problems which well improve the efficiency of algorithms presented by previous research. For the symmetry breaking problem we get a protocol which admits arbitrary initiation, and uses only linear number of message bits and linear time on the average. For the mutual exclusion problem we break the lower bound on the number of messages which holds in case of unrestricted asynchrony. We also find an important difference between Archimedean and Synchronous networks. Our algorithms are practical in the sense that any existing distributed system up to now follows the Archimedean time assumption.

1. Introduction
This work examines the effect of limited asynchrony on three fundamental problems of distributed computation: The problem of mutual exclusion, that of readers and writers and the problem of symmetry breaking in a logical ring. The amount of asynchrony among local clocks of the various sites is limited as follows: We assume the distributed system to be Archimedean (see [Vitányi, 84]). That is, the duration of a step of any process in any site is bounded above by rmax and below by rmin units of (absolute) time and the (absolute) time it takes for any message to be sent through a direct communication link is bounded above by dmax and below by dmin. Although processes are assumed to know the

* This research was funded in part by the NSF contract DCR 8503497 and by the Ministry of Industry, Energy and Technology of Greece.
bounds $r_{\min}$, $r_{\max}$, $d_{\min}$ and $d_{\max}$, they do not have access to any global clock showing the absolute time.

Any practical distributed system up to now follows the Archimedean time assumption. However, there is very little previous research investigating the gains in efficiency that distributed algorithms may have by exploiting such assumptions. The work of [Vitányi, 84] presented an algorithm to elect a leader in an Archimedean ring of $N$ processors with distinct names, by using only $O(N)$ messages, while it had been previously shown that $O(N\log N)$ messages are needed by any distributed algorithm in order to elect a leader in or ring, if unlimited asynchrony is assumed. [Reif, Spirakis, 84,85] used Archimedean asynchronism in their distributed algorithms for interprocess communication and resource allocation.

On the other hand, there is a considerable amount of research on distributed algorithms that work in a synchronous network. Most of those techniques depend crucially on the exact timing one can do in the absence of any asynchronism. It is therefore not at all clear whether such techniques can be easily modified so they can apply to an environment of limited asynchrony. (See, for example, [Frederickson, Lynch, 84], [Gafni, 85], [Itai, Rodeh, 81], [J.van Leeuven et al, 87], [Santoro, Rotem, 85], [Frederickson, Santoro, 86] as a representative sample of work in synchronous distributed computation).

We consider here three major problems of distributed computation: The first is the achievement of mutual exclusion in a complete network of sites. We show how the notion of Archimedean time can be exploited to get a message complexity which is below the lower bound for the same problem when unlimited asynchrony is assumed. We also provide an analysis of the time delay needed to achieve mutual exclusion. Our second problem is that of synchronizing readers and writers, both by preventive (i.e. mutual exclusion based) techniques and by optimistic techniques (each process is allowed to execute and, if the result is unsatisfactory, has to start again). We show that the optimistic techniques are much improved (e.g. starvation is avoided) by a simple use of the knowledge of limited asynchrony. Finally, we consider the problem of electing a leader in an anonymous ring of processors (symmetry breaking). We show how to adopt the synchronous algorithm of [Frederickson, Santoro, 1986] in order to get a solution for an Archimedean ring, which has the same message complexity and much better time complexity that the algorithm of [Vitányi, 84]. It seems that modifying a synchronous distributed algorithm to work in an Archimedean environment is a general technique which optimizes both message and time complexity simultaneously.