Approximation of $\delta$-Timeliness*

Carole Delporte-Gallet$^1$, Stéphane Devismes$^2$, and Hugues Fauconnier$^1$

$^1$ Université Paris Diderot, LIAFA
$^2$ Université Joseph Fourier, Grenoble I, VERIMAG UMR 5104
Stephane.Devismes@imag.fr

Abstract. In asynchronous message-passing distributed systems prone to process crashes, a communication link is said $\delta$-timely if the communication delay on this link is bounded by some constant $\delta$. We study here in which way processes may approximate and find structural properties based on $\delta$-timeliness (e.g., find $\delta$-timely paths between processes or build a ring between correct processes using only $\delta$-timely links).

To that end, we define a notion of approximation of predicates. Then, with help of such approximations, we give a general algorithm that enables to choose and eventually agree on one of these predicates. Finally, applying this approach to $\delta$-timeliness, we give conditions and algorithms to approximate $\delta$-timeliness and dynamically find structural properties using $\delta$-timeliness.

1 Introduction

Assume an asynchronous message-passing system prone to process crash failures and consider the following problem: we know that after some unknown time there is at least one path from process $p$ to process $q$ such that every message sent along this path arrives to process $q$ by $\delta$ time units (such a path is said $\delta$-timely). Now, how can we determine one of these paths or at least one path that is $\Delta$-timely for a $\Delta$ close to $\delta$? By “determine” we mean that eventually all processes agree on the chosen path.

To that end, the processes must be at least able to test the $\delta$-timeliness of paths and one of the contribution of this paper is to give some necessary and sufficient conditions to do this. In particular, we prove that without synchronized clocks, the system has to ensure strong synchrony properties in the sense that there must not only exist $\delta$-timely paths from $p$ to $q$ but also from $q$ to $p$.

The $\delta$-timeliness of a path is a property that is only eventually ensured. Moreover, $\delta$-timeliness of a link can only be approximated by processes. We would like that an approximation algorithm outputs boolean values in such a way that if the path is truly $\delta$-timely, then eventually only true is output, and if the path is not $\delta$-timely, then false is output infinitely often. However as we will see, such approximations of $\delta$-timeliness are generally too strong because there is some incertitude on the time it takes to go from $p$ to $q$ on the path. Therefore the approximation algorithm only ensures that if the path is $\delta$-timely, then true is eventually output forever and if the path is not $\Delta$-timely (with

* This work has been supported in part by the ANR project SHAMAN and the INRIA project GANG.
\( \Delta > \delta \), then false is output infinitely often. Hence, between \( \delta \) and \( \Delta \) the approximation algorithm may output true or false just as well.

The existence of such approximations enables to answer to our initial problem (but with the little restriction on \( \delta \) and \( \Delta \)): if at least one \( \delta \)-timely path exists from \( p \) to \( q \), it is then possible to ensure that all processes eventually agree on the same \( \Delta \)-timely path from \( p \) to \( q \). An algorithm which ensures that all correct processes eventually agree on the same structure verifying some properties is called an extraction algorithm [7]. Many other problems can be solved in the same way: for example extraction of a tree containing all correct processes whose all paths from the root are \( \Delta \)-timely, or a ring containing all correct processes and whose links are \( \Delta \)-timely, etc.

Actually, all these aforementioned algorithms use methods very similar to the ones used, for example, in the eventual leader election problem in [1–4, 9, 11]. In fact, we prove here that this approach can be expressed in a very general framework. Approximation of \( \delta \)-timeliness and extraction of some structures based on the \( \delta \)-timeliness relation are special cases of the more general problem of approximating properties on runs and extracting structures based on these properties. Instead of \( \delta \)-timeliness, one can consider more general properties on runs and consider when they are eventually true forever. Then, assuming approximation algorithms for these predicates, it is possible to extract (i.e., choose and agree) one of these predicates that is true in the run. More precisely, as for \( \delta \)-timeliness, the approximation is defined by pairs of predicates \( (P, Q) \), where \( P \) specifies when the approximation has to eventually output true forever and \( \neg Q \) when the approximation has not to eventually output true forever (in other words has to output infinitely often false). Then, given a set of pairs of predicates \( (P_i, Q_i) \) indexed by some set \( I \), the extraction algorithm, assuming that at least one \( P_i \) is true, will converge to some \( i_0 \) for all correct processes such that \( Q_{i_0} \) is true. This generalization enables to have the same algorithms for many different problems.

In this way, the extraction of structures based on the \( \delta \)-timeliness relation is only a special case of this general case. For example, assuming that processes have a trusting mechanism (like a failure detector) giving to each processes lists of process supposed to be alive, and consider the predicate “\( p \) is eventually in all lists”, then the extraction algorithm gives one of such processes. Assuming that these lists are given by failure detector \( \diamond S \) ([5]), then the extraction algorithm gives an implementation of failure detector \( \Omega \) and the algorithm is rather close to algorithm of [6].

Contributions. In this paper, we first define a general framework in which processes may approximate predicates on runs and give a generic extraction algorithm that enables processes to converge on one of the predicates satisfied in the run. Then, we apply these concepts by proposing algorithms and impossibility results about the approximation of \( \delta \)-timeliness. In particular we give sufficient conditions to approximate \( \delta \)-timeliness on links. More precisely, we prove that we need either to have perfectly synchronized clocks or to assume very strong timeliness requirements. Finally, we give examples of general extractions based on approximation of link \( \delta \)-timeliness. These illustrations emphasizes the two mains points of our contribution. Firstly, this general approach allows to drastically simplify the design of algorithm. Indeed, from a simple algorithm that approximates a local predicate, like “a link is \( \delta \)-timely”, we can easily derive an algorithm to extract a more complex structure such as a \( \delta \)-timely path or a