Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Journal of Computer and System Sciences 68 (2004) 123–133

JOURNAL OF COMPUTER AND SYSTEM SCIENCES

http://www.elsevier.com/locate/jcss

A necessary and sufficient condition for transforming limited accuracy failure detectors

E. Anceaume,[a] A. Fernández,[b] A. Mostefaoui,[a] G. Neiger,[c] and M. Raynal[a,*]

[a] IRISA, Campus de Beaulieu, Université de Rennes 1, 35042 Rennes Cedex, France
[b] Universidad Rey Juan Carlos, 28933 Móstoles, Madrid, Spain
[c] Intel Corporation, JF3-332, 2111 NE 25th Avenue, Hillsboro, OR 97124, USA

Received 3 January 2000; revised 19 August 2003

Abstract

Unreliable failure detectors are oracles that give information about process failures. Chandra and Toueg were first to study such failure detectors for distributed systems, and they identified a number that enabled the solution of the *Consensus* problem in asynchronous distributed systems. This paper focuses on two of these, denoted $\mathcal{S}$ (strong) and $\Diamond\mathcal{S}$ (eventually strong). The characteristics of a given unreliable failure detector are usually described by its completeness and accuracy properties. Completeness is a requirement on the actual detection of failures, while accuracy limits the mistakes a failure detector can make. Let the *scope* of the accuracy property of an unreliable failure detector be the minimum number (k) of processes that may not erroneously suspect a correct process to have crashed. Usual failure detectors implicitly consider a scope equal to n (the total number of processes). Accuracy properties with limited scope give rise to the classes of failure detectors that we call $\mathcal{S}_k$ and $\Diamond\mathcal{S}_k$. This paper investigates the following question: "Given $\mathcal{S}_k$ and $\Diamond\mathcal{S}_k$, under which condition is it possible to transform their failure detectors into their counterparts with unlimited accuracy, i.e., $\mathcal{S}$ and $\Diamond\mathcal{S}$?". The paper answers this question in the following way. It first presents a particularly simple protocol that realizes such a transformation when $f < k$ (where f is the maximum number of processes that may crash). Then, it shows that there is no reduction protocol when $f \geqslant k$.

Keywords: Asynchronous distributed system; Crash failure; Failure detector accuracy; Reduction protocol; Unreliable failure detector

[*] Corresponding author.

E-mail addresses: anceaume@irisa.fr (E. Anceaume), afernandez@acm.org (A. Fernández), achour@irisa.fr (A. Mostefaoui), gil@acm.org (G. Neiger), raynal@irisa.fr (M. Raynal).

0022-0000/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcss.2003.08.003

1. Introduction

Background: It has been proven that several practical agreement problems such as *atomic broadcast*, *leader election*, *group membership* and *atomic commit*, encountered in the design of reliable applications built on top of unreliable asynchronous distributed systems, cannot be solved in a deterministic way [5,9]. Many agreement problems can be characterized by a single problem, namely the *Consensus* problem. This means that any solution to *Consensus* can be used as a building block on top of which solutions to particular agreement problems can be designed. From a theoretical point of view, this means that an agreement problem cannot be solved in systems where *Consensus* cannot be solved. This is due to the uncertainty created by asynchrony and failures: their combination can actually prevent processes from making consistent decisions.[1] From an operational point of view, this is due to the impossibility of a process distinguishing a failed process from a slow process or from a process with which communications are very slow. Solving such distributed computing problems requires augmenting the asynchronous distributed system with some synchrony assumptions. Minimal Synchronism [3] and Partial Synchrony [4] have been advances in this direction. The additional assumptions required to solve such problems in the presence of process crashes have been abstracted in a modular way by Chandra and Toueg in their definition of the *Unreliable Failure Detector* concept [1]. This paper considers asynchronous distributed systems equipped with such unreliable failure detectors.

A failure detector can informally be seen as a set of *oracles*, one per process. The failure-detector module (oracle) associated with a process provides with a list of processes it guesses have crashed. A failure detector can make mistakes by not suspecting a crashed process, or by erroneously suspecting a correct process. In their seminal paper, Chandra and Toueg defined two types of properties that characterize classes of failure detectors. A class is defined by a *Completeness* property and an *Accuracy* property. A completeness property concerns the actual detection of crashes. The completeness property in which we are interested basically states that "every crashed process is eventually suspected by every correct process". An accuracy property limits the erroneous suspicions a failure detector can make. In this paper, we are mainly interested in *Weak Accuracy*. This property basically states that "there is a correct process that is not suspected". Weak accuracy is *perpetual* if it must be satisfied at all times. It is *eventual* if it may be satisfied only after some (unknown but finite) time. The class of failure detectors satisfying completeness and perpetual (respectively, eventual) weak accuracy is denoted $\mathcal{S}$ (respectively, $\Diamond\mathcal{S}$). $\mathcal{S}$-based *Consensus* protocols were developed by Chandra and Toueg [1] and by Mostefaoui and Raynal [10]. They assume $f<n$, where n is the number of processes in the system and f is the maximum number that may crash. $\Diamond\mathcal{S}$-based *Consensus* protocols were developed by Chandra and Toueg, by Hurfin, Mostefaoui and Raynal [7,8,10], and by Schiper [14]. The $\Diamond\mathcal{S}$-based protocols all require $f<n/2$; Chandra and Toueg proved that this requirement is necessary. Thus, these protocols are all optimal with respect to the maximum number of crashes that can be tolerated.

Problem and results: The weak accuracy properties implicitly have a *scope* spanning the whole system: there is a correct process that is not suspected *by any other process* (either always or after

[1] This means that in some circumstances, some processes cannot progress (and a liveness property may be violated) or their decisions are inconsistent (and a safety property may be violated).

some time). Here, the important issue is that the ‘‘non-suspicion of a correct process’’ concerns all processes. In this paper, we investigate failure-detector classes whose accuracy property has a limited scope: the number of processes that are required not to suspect a correct process is limited to k ($k \leqslant n$, where n is the total number of processes).

The parameter k defines the *scope of the weak accuracy* property, thereby giving rise to the classes $\mathcal{S}_k$ and $\diamond\mathcal{S}_k$ of unreliable failure detectors ($\mathcal{S}_n$ and $\diamond\mathcal{S}_n$ are identical to $\mathcal{S}$ and $\diamond\mathcal{S}$, respectively). Practically, this means that there is a (not statically predefined) set of k processes, containing a correct process, whose failure-detector modules do not suspect that correct process. An important question then is the following:

> ‘‘Is there a necessary and sufficient condition that allows transformation of any failure detector in $\mathcal{S}_k$ (respectively, $\diamond\mathcal{S}_k$) into a failure detector in $\mathcal{S}$ (respectively, $\diamond\mathcal{S}$)?’’

This paper answers this question in the following way (recall that f is the maximum number of processes that may crash):

- First, a protocol is presented that transforms any failure detector FD_{in} into a failure detector FD_{out} such that (1) if $FD_{\text{in}} \in \mathcal{S}_k$, then $FD_{\text{out}} \in \mathcal{S}$ and (2) if $FD_{\text{in}} \in \diamond\mathcal{S}_k$, then $FD_{\text{out}} \in \diamond\mathcal{S}$. This protocol requires $f < k$.
- Then, it is shown that there is no protocol that transforms any failure detector in $\mathcal{S}_k$ (respectively, $\diamond\mathcal{S}_k$) into a failure detector in $\mathcal{S}$ (respectively, $\diamond\mathcal{S}$) when $f \geqslant k$.

Notice that the correctness condition of the transformation protocol (namely, $f < k$) does not require the assumption that a majority of processes always remain correct. This has an interesting practical consequence: when f is small (i.e., $\ll n$), k (the accuracy scope) too can be small ($\ll n$). This is particularly attractive when facing fault-tolerance or scaling problems. In an extreme example, this means that the *Consensus* problem can be solved in presence of one process failure (i.e., $f = 1$) in an asynchronous distributed system equipped with a failure detector in $\mathcal{S}_2$, since the transformation protocol yields a failure detector in $\mathcal{S}$ which can be used to solve *Consensus* [1]. ($\mathcal{S}_2$ contains failure detectors for which there is a correct process p_i that is never suspected by some other process p_j.)

The paper is made up of six sections. Section 2 introduces the computation model and the failure detectors of Chandra and Toueg. Section 3 defines the classes of limited accuracy failure detectors. Section 4.1 then presents and proves correct, for $f < k$, a protocol transforming $\mathcal{S}_k$ (respectively, $\diamond\mathcal{S}_k$) into $\mathcal{S}$ (respectively, $\diamond\mathcal{S}$). Section 5 shows there is no such protocol when $f \geqslant k$. Finally, Section 6 concludes the paper.

2. Asynchronous distributed systems and unreliable failure detectors

2.1. Asynchronous systems with process crashes and fair lossy links

We consider a system to consist of a finite set Π of $n > 1$ processes, namely, $\Pi = \{p_1, p_2, \dots, p_n\}$. Each process has a unique identity. A process can fail by *crashing*, i.e., by prematurely halting. It behaves correctly (i.e., according to its specification) until it (possibly) crashes. By definition, a *correct* process is a process that does not crash. A *faulty* process is a process that is not correct. As previously indicated, $f < n$ denotes the maximum number of processes that can crash.

Processes communicate and synchronize by sending and receiving messages through *channels*. Every pair of processes is connected by a channel. Channels do not create or alter messages. They are not required to deliver messages in order. Moreover, they may lose messages provided that, if a process p_i sends an infinite number of messages to a process p_j and if p_j executes receive actions infinitely often, p_j receives an infinite number of messages from p_i (this is a *fair lossy channel* assumption). Finally, there is no assumption about the relative speed of processes or message transfer delays.

The protocol proposed in Section 4 is correct under the assumptions detailed above; it tolerates fair lossy channels that may deliver messages out of order. The impossibility result of Section 5 applies even to systems with channels that never lose message and always deliver them in order.

2.2. Chandra–Toueg's unreliable failure detectors

Informally, a failure detector consists of a set of modules, each one attached to a process: the module attached to p_i maintains a set (named $suspected_i$) of processes it currently suspects to have crashed. Any failure-detector module is inherently unreliable: it can make mistakes by not suspecting a crashed process or by erroneously suspecting a correct one. Moreover, suspicions are not necessarily stable: a process p_j can be added to or removed from a set $suspected_i$ according to whether p_i's failure-detector module currently suspects p_j or not. As in other papers devoted to failure detectors, we say "process p_i suspects process p_j" at some time t, if we have $p_j \in suspected_i$ at that time.

As indicated in Section 1, a failure-detector class is formally defined by two abstract properties, namely a *Completeness* property and an *Accuracy* property. In this paper, we consider the following completeness property defined by Chandra and Toueg [1]:

- *Strong Completeness*: Eventually, every process that crashes is permanently suspected by every correct process.

Among the accuracy properties defined by Chandra and Toueg we consider here the following two:

- *Perpetual Weak Accuracy*: Some correct process is never suspected by any process.
- *Eventual Weak Accuracy*: There is a time after which some correct process is never suspected by any process.

Combined with Strong Completeness, these accuracy properties define the following two classes of failure detectors [1]:

- $\mathcal{S}$: The class of *Strong* failure detectors. This class contains all the failure detectors that satisfy the strong completeness property and the perpetual weak accuracy property.
- $\diamond\mathcal{S}$: The class of *Eventually Strong* failure detectors. This class contains all the failure detectors that satisfy the strong completeness property and the eventual weak accuracy property.

Clearly, $\mathcal{S} \subseteq \diamond\mathcal{S}$.

3. Failure detectors with limited weak accuracy

3.1. Limited weak accuracy

As noted in Section 2.2, the weak accuracy properties involve all the correct processes, and consequently spans the whole system. This "whole-system spanning" suggests that weak accuracy might be easier to satisfy if it involved only a subset of processes (one of which is correct and never suspected by the others). This observation motivates the following definition, where the parameter k defines the scope of the accuracy property. For set of processes Q and process $p \in Q$, k-accuracy is $\langle Q, p \rangle$—satisfied if the following hold:

1. Scope: $|Q| = k$.
2. No suspicion: p is correct and is not suspected by any process $q \in Q$.

A failure detector satisfies perpetual (respectively, eventual) weak k-accuracy is satisfied if, for each execution, there is a set of processes Q and process $p \in Q$ such that k-accuracy is $\langle Q, p \rangle$—satisfied at all times (respectively, after some point in time) in that execution. (Different executions may use different sets Q and processes p.) Given a scope parameter k, we get the two following classes of failure detectors in the style of Chandra and Toueg:

- $\mathcal{S}_k$: This class contains all the failure detectors that satisfy strong completeness and perpetual weak k-accuracy.
- $\diamond\mathcal{S}_k$: This class contains all the failure detectors that satisfy strong completeness and eventual weak k-accuracy.

Clearly, $\mathcal{S}_k \subseteq \diamond\mathcal{S}_k$ and, for each $k < n$, $\mathcal{S}_{k+1} \subseteq \mathcal{S}_k$ and $\diamond\mathcal{S}_{k+1} \subseteq \diamond\mathcal{S}_k$. It is easy to see that "traditional" (unlimited) weak accuracy is obtained with $k = n$. More precisely, $\mathcal{S}_n = \mathcal{S}$ and $\diamond\mathcal{S}_n = \diamond\mathcal{S}$. Moreover, $\mathcal{S}_1$ contains failure detectors whose accuracy need have no scope (i.e., such failure detector need not provide any information). For this reason, the rest of this paper considers only failure detectors with k-accuracy where $k > 1$.

3.2. Related work

Guerraoui and Schiper [6] used failure detectors with a kind of limited-scope accuracy to address problems raised by distributed system partitioning. They assumed that a minority of processes crash and that the scope of the failure detector comprises a majority of processes, i.e., $f < n/2 < k$.

Failure detectors with limited-scope accuracy (or *limited-scope failure detectors*) were previously investigated by Yang et al. [15], who defined the failure-detector classes $\mathcal{S}_k$ and $\diamond\mathcal{S}_k$ using a slightly different notation (they used S_k for the class denoted here $\mathcal{S}_{n+1-k}$). A part of their work focused on the K-set agreement problem.[2] They proved that K-set agreement can be solved (1) with $\mathcal{S}_{n+1-K}$ for any number of failures and (2) with $\diamond\mathcal{S}_{n+1-K}$ if $n > 2f$.

[2] The *K-set agreement* problem [2] allows study of the relation between the number of choices allowed for decided values and the maximum number of crashes. It is a generalization of the *Consensus* problem: distinct processes can decide on different values, but the number of decided values must be at most K. There are trivial protocols solving the K-set agreement problem if $f < K$. Without a failure detector, there is no solution when $f \geqslant K$. (Note that the 1-set agreement problem is the *Consensus* problem.)

A $\diamond\mathcal{S}_k$-based *Consensus* protocol was presented by Mostefaoui and Raynal [11]. It requires $f < \min(k, n/2)$. The same authors also investigated limited-scope failure detectors to solve the K-set agreement problem [12]. They proposed two protocols. The first considers an underlying failure detector of the class $\mathcal{S}_k$ and requires $f < K + k - 1$. The second considers a failure detector of the class $\diamond\mathcal{S}_k$ and requires $f < \max(K, \max_{1 \leqslant \alpha \leqslant K}(\min(n - \alpha\lfloor n/(\alpha + 1)\rfloor, \alpha + k - 1)))$.

Failure detectors where both the completeness property and the accuracy property hold on subsets of correct processes have been investigated by Raynal and Tronel [13]. Their completeness and accuracy properties are *restricted* in the following sense: they are not required to involve all the correct processes but only k_1 and k_2 of them (k_1 are involved in the completeness property, and k_2 in the accuracy property). It is shown that transforming such restricted failure detectors into their non-restricted counterparts requires $k_1 + k_2 > n$ (for the transformation to be safe) and $\max(k_1, k_2) \leqslant n - f$ (for the transformation to be live).

4. From k-accuracy to full accuracy

This section describes a protocol for process p_i (given in Fig. 1) that transforms any failure detector of either class $\mathcal{S}_k$ or $\diamond\mathcal{S}_k$ into a failure detector the corresponding class with unlimited accuracy. That is, if the input failure detector is in $\mathcal{S}_k$ (respectively, $\diamond\mathcal{S}_k$), then the resulting failure detector is in $\mathcal{S}$ (respectively, $\diamond\mathcal{S}$). Although the constant k does not appear explicitly in the protocol text, the protocol's correctness condition requires $f < k$ (see Section 4.2). Surprisingly, the proposed transformation protocol is quite simple.

4.1. A protocol

In Fig. 1, process p_i consults its underlying failure detector ($\mathcal{S}_k$ or $\diamond\mathcal{S}_k$) whenever it refers to the set pseudovariable *lim_suspect*$_i$. If $p_\ell \in$ *lim_suspect*$_i$, we say "p_i lim-suspects p_ℓ". The protocol provides the upper layer with a set denoted *suspected*$_i$. When $\ell \in$ *suspected*$_i$ we say

```
init: $suspected_i \leftarrow \emptyset$

task $T1$:
    repeat forever
        foreach $j$ do send SUSPICION($lim\_suspect_i$) to $p_j$ enddo
    end repeat

task $T2$:
    repeat forever
        wait until ( SUSPICION($lim\_suspect$) received from $(n - f)$ distinct processes );
        let $X$= the set of the $lim\_suspect$ sets received at the previous line;
        $suspected_i \leftarrow \bigcap_{\ell s \in X} \ell s$
    end repeat
```

Fig. 1. Protocol to convert from $\mathcal{S}_k$ (or $\diamond\mathcal{S}_k$) to $\mathcal{S}$ (or $\diamond\mathcal{S}$) for process p_i (requires $f < k$).

"p_i suspects p_ℓ". Section 4.2 will prove that the sets *$suspect_i$* have the properties required of the target failure detector ($\mathcal{S}$ or $\Diamond\mathcal{S}$).

To ensure that the *$suspected_i$* sets satisfy the properties defined by $\mathcal{S}$ (respectively, $\Diamond\mathcal{S}$), processes continually exchange their *$lim\_suspect_i$* sets. More precisely, p_i's behavior is made up of two tasks. The task $T1$ periodically sends *$lim\_suspect_i$* to all processes (including itself). The task $T2$ is an infinite loop in which p_i first receives *lim_suspect* sets from $(n-f)$ distinct processes and then defines the current value of *$suspected_i$* as their intersection.

4.2. Proof of correctness

Theorem 1. *Let $f<k$. The protocol described in Fig.* 1 *transforms any failure detector in $\mathcal{S}_k$ (respectively, in $\Diamond\mathcal{S}_k$) into a failure detector in $\mathcal{S}$ (respectively, in $\Diamond\mathcal{S}$).*

Proof. The proof comprises two parts, addressing the strong completeness property and the perpetual/eventual weak accuracy property, respectively.

1. Strong completeness: Note first that (by assumption) the underlying failure detector (whether it belongs to $\mathcal{S}_k$ or to $\Diamond\mathcal{S}_k$) satisfies strong completeness. We show that the protocol preserves this property, i.e., if a process p_ℓ crashes and if p_i is correct, then eventually p_ℓ remains permanently in *$suspected_i$*.

First observe that there is a time t after which all faulty processes have crashed. Moreover, there is a time $t' \geqslant t$ after which all SUSPICION messages that are received (1) have been sent by correct processes and (2) carry the identity of p_ℓ (as the underlying failure detector satisfies strong completeness). Furthermore, as at most f processes crash and the channels are fair lossy, each correct process p_i infinitely often receives SUSPICION messages from $(n-f)$ different processes, and after t' all those messages carry the identity of p_ℓ. It follows that, after t', line 5 is executed infinitely often and, each time it is executed after t', p_ℓ belongs to X. Thus, there is a time after which every crashed process is suspected by every correct process.

2. Perpetual/eventual weak accuracy: Consider first the case in which the underlying failure detector belongs to $\Diamond\mathcal{S}_k$. Thus, the *$lim\_suspect_i$* sets satisfy eventual weak k-accuracy. We show that the *$suspected_i$* sets satisfy eventual weak accuracy.

Consider a specific execution of the protocol. By definition, there is a set of a processes Q and a process $p \in Q$ such that k-accuracy is $\langle Q,p \rangle$—satisfied after some point in time in that execution. More specifically, there is a time t, a set Q, and a correct process $p \in Q$ such that $|Q| = k$ and, after t, p is never lim-suspected by any of the processes in Q. Let $t' \geqslant t$ be a time after which no SUSPICION messages are received that were sent by the processes of Q before t.

After t', p can appear in the SUSPICION messages of at most $(n-k)$ processes (namely the processes in $\Pi - Q$). As $k>f$, we have $n-f>n-k$, from which we conclude that, at line 4, there is at least one SUSPICION message that does not carry the identity of p. It follows that after t', p can no longer appear in a set *$suspected_i$* for any process p_i. Hence, the eventual weak accuracy property holds.

If the underlying *$lim\_suspect_i$* sets satisfy perpetual weak k-accuracy (i.e., the underlying failure detector belongs to $\mathcal{S}_k$), the proof that the *$suspected_i$* sets satisfy perpetual weak accuracy property is the same as previously, considering $t = t' = 0$. □

5. An impossibility result

The previous transformation relies on the condition $f < k$. This section shows that this condition is actually necessary: no such transformation is possible if $f \geqslant k > 0$. The proof of the impossibility result relies on the following lemma.

Lemma 1. *There is no protocol that implements a failure detector of class $\Diamond\mathcal{S}$ in an asynchronous distributed system where $f > 0$.*

Proof. The proof is by contradiction and is as follows. We first assume that there is a protocol P implementing a failure detector FD of the class $\Diamond\mathcal{S}$ in an asynchronous system Σ with $n > f > 0$. We then use the strong completeness property of FD to construct a fault-free execution of the protocol P in which FD does not satisfy the eventual weak accuracy property. Since FD was assumed to be in $\Diamond\mathcal{S}$ and thus satisfy eventual weak accuracy in all executions, we will have a contradiction, implying that protocol P cannot exist.

For the sake of contradiction, assume there is a protocol P implementing FD, a failure detector in the class $\Diamond\mathcal{S}$ in system Σ. P thus provides each process p_i, $1 \leqslant i \leqslant n$, with a set *suspected*$_i$ (holding the processes that p_i currently suspects). We use *suspected*$_i(t)$ to denote the value of such a set at time t. We will construct an execution E of P in Σ such that the following both hold:

1. E is fault-free (no process fails).
2. There is an infinite sequence of times $t_1 < t_2 < t_3 < \cdots$ such that, for all $i > 0$, process $p_{((i-1) \bmod n)+1}$ is suspected by all other processes at time t_i (i.e., $(\forall j \neq ((i-1) \bmod n) + 1)$ $(p_{((i-1) \bmod n)+1} \in \mathit{suspected}_j(t_i)))$.

Clearly, in the execution E of P, each process is infinitely often suspected by the other processes and, consequently, the eventual weak accuracy is not satisfied.

For simplicity, we define $t_0 = 0$. We will construct execution E inductively. For $i \geqslant 0$, assume that E is already constructed up to time t_{i-1} (t_0 in the base case); we show how to define t_i and construct the interval $(t_{i-1}, t_i]$ of the execution E such that item (2) above is satisfied for the value i.

Consider another execution E_i of P in Σ that is identical to E up to time t_{i-1} and in which $p_{((i-1) \bmod n)+1}$ crashes after t_{i-1} and all other processes are correct. By strong completeness, there is some time t after which P has all correct processes (i.e., all but $p_{((i-1) \bmod n)+1}$) permanently suspect $p_{((i-1) \bmod n)+1}$. Let $t_i = \max(t, t_{i-1} + 1)$, and make E behave in the interval $(t_{i-1}, t_i]$ as follows:

- Process $p_{((i-1) \bmod n)+1}$ does not crash, but all the messages it sends during this interval remain undelivered at the end of the interval (namely, at t_i). Messages sent by $p_{((i-1) \bmod n)+1}$ in $[0, t_{i-1}]$ but not delivered in that interval are delivered (or not) in $(t_{i-1}, t_i]$ of E exactly as they are (or not) in that interval of E_i.
- The other processes behave as they do in the interval $(t_{i-1} + 1, t_i]$ of E_i. Messages sent by these processes in $[0, t_i]$ of E but not delivered in $[0, t_{i-1}]$ are delivered (or not) in $(t_{i-1}, t_i]$ of E exactly as they are (or not) in that interval of E_i.

The above imply that each process p_j, $j \neq ((i-1) \bmod n) + 1$ cannot distinguish executions E_i and E through time t_i. Since all processes but $p_{((i-1) \bmod n)+1}$ suspect $p_{((i-1) \bmod n)+1}$ at time t_i of E_i, they also suspect $p_{((i-1) \bmod n)+1}$ at time t_i of E.

Repeating this process for every $i > 0$, we construct an infinite sequence of intervals that constitutes the execution E. Hence we have the execution E that has properties 1 and 2 mentioned above, which completes the proof. □

Lemma 2. *Let $f \geqslant k > 0$. There is no protocol transforming any failure detector in $\mathcal{S}_k$ into a failure detector in $\Diamond\mathcal{S}$.*

Proof. Consider a system Σ with n processes of which up to $f \geqslant k$ may fail. Assume for a contradiction that, in Σ, any failure detector in $\mathcal{S}_k$ can be converted into a failure detector in $\Diamond\mathcal{S}$. Partition $\Pi = \{p_1, p_2, \ldots, p_n\}$ into sets A and B as follows: $A = \{p_i | 1 \leqslant i \leqslant n - (k-1)\}$ and $B = \{p_j | n - (k-1) < j \leqslant n\}$. Note that $|B| = k - 1 < f$. Consider a failure detector FD with the following properties:

- In executions in which all processes in B crash (such crashes can occur as $f \geqslant k > |B|$), the process suspicions are as follows:
  - Each process in A always suspects all processes except itself.
  - Each process in B never suspects any process.
- In executions in which some process in B is correct, FD satisfies strong completeness and perpetual weak accuracy.

Let us first show that FD is in the class $\mathcal{S}_k$. This is done through the following cases:

- Consider any execution in which all processes in B crash. This implies that each correct process always suspects all processes except itself; thus, each correct process eventually permanently suspects all faulty processes, satisfying strong completeness. In addition, let $p \in A$ be a correct process (such a process must exist as $n > f$) and consider the set $Q = \{p\} \cup B$. Clearly, $|Q| = k$ and Q contains a correct process, p. Moreover, no process in Q ever suspects p: p never suspects itself, and the other processes in Q never suspect any process. Thus, k-accuracy is always $\langle Q, p \rangle$—satisfied in this execution. Thus, FD satisfies perpetual weak k-accuracy.
- Consider any execution in which some process in B is correct. By definition, the failure detector satisfies strong completeness and perpetual weak accuracy. The latter implies that it always $\langle Q, p \rangle$—satisfies k-accuracy for any set Q such that $p \in Q$ and $|Q| = k$. Thus, FD satisfies perpetual weak k-accuracy.

Assume for a contradiction that C is a protocol that transforms FD into a failure detector in $\Diamond\mathcal{S}$. Let Σ' be a system made up of a set Π' of $n' = n - (k-1)$ processes $p_1, \ldots, p_{n-(k-1)}$ of which $f' = f - (k-1)$ can fail (notice $f \geqslant k$ implies $f' > 0$). We will use C to construct a protocol C' that will implement a failure detector of class $\Diamond\mathcal{S}$ in Σ' with no underlying failure detector.

C' has each process $p_i \in \Pi'$ emulate protocol C, running it with the following exceptions:

- p_i never sends a message to any p_j, $n - (k-1) < j \leqslant n$ (there is no such $p_j \in \Pi'$).

- Whenever C would have p_i consult FD, C' has p_i behave as if FD has returned the set $X = \{p_\ell | 1 \leqslant \ell \leqslant n \wedge \ell \neq i\}$. (Notice that this is consistent with the definition of FD when all the processes in B have crashed.)
- Whenever C would report a set X for $\diamond\mathcal{S}$ in Σ, C' will report $X \backslash B$ in Σ'.

We now show that the protocol C' implements a failure detector of the class $\diamond\mathcal{S}$ in Σ' without the help of an underlying failure detector. Let us consider any execution E' of C' in Σ'. The processes' emulation of C corresponds exactly to their behavior in an execution E of C in Σ with FD in which each process in B crashes before sending any message (thus the following refers only to p_i such that $1 \leqslant i \leqslant n - (k-1)$). Because C implements a failure detector of class $\diamond\mathcal{S}$ in Σ augmented with FD (from $\mathcal{S}_k$), we conclude that C' implements a failure detector of class $\diamond\mathcal{S}$ in Σ' with no underlying failure detector. More precisely, we have the following:

- *Strong Completeness*: If a process p_i crashes in E' during the emulation C', the same process has crashed in E and, by the strong completeness of C in Σ with FD, p_i will be permanently suspected by all correct processes in the emulation C'.
- *Eventual Weak Accuracy*: If p_i is correct in E', it is also correct in E. Thus, each process correct in E will eventually never suspect p_i in E. Since each process correct in E is also correct in E' and any process suspected in E' is also suspected in E, each process correct in E' will eventually never suspect p_i in E'.

  If a process p_i is not suspected by some correct process p_j (note that $1 \leqslant i,\ j \leqslant n - (k-1)$) in E, the same holds for the emulation C', and consequently the failure detector obtained with C' satisfies eventual weak accuracy.

Since $f' > 0$, C' implements a failure detector of class $\diamond\mathcal{S}$ in a system with at least one failure and without the help of an underlying failure detector. This contradicts Lemma 1. Thus, the assumption stating that $\mathcal{S}_k$ can be converted to $\diamond\mathcal{S}$ in a system with $f \geqslant k$ is erroneous. □

Theorem 2. *Let $f \geqslant k$. There is no protocol transforming any failure detector in $\mathcal{S}_k$ (respectively, $\diamond\mathcal{S}_k$) into a failure detector in $\mathcal{S}$ (respectively, $\diamond\mathcal{S}$).*

Proof. This theorem follows directly from the definitions of $\mathcal{S}_k$ and $\diamond\mathcal{S}_k$ (which states $\mathcal{S}_k \subseteq \diamond\mathcal{S}_k$), the definitions of $\mathcal{S}$ and $\diamond\mathcal{S}$ (which states $\mathcal{S} \subseteq \diamond\mathcal{S}$), and Lemma 2. □

6. Conclusion

The *scope* of the accuracy property of an unreliable failure detector is the minimum number k of processes that must not suspect a correct process. Such a scope definition has allowed us to define the classes of failure detectors $\mathcal{S}_k$ and $\diamond\mathcal{S}_k$. The well-known failure detectors $\mathcal{S}$ and $\diamond\mathcal{S}$ implicitly consider a scope equal to n (the total number of processes).

This paper has addressed the following question: "Given $\mathcal{S}_k$ and $\diamond\mathcal{S}_k$, under which condition is it possible to transform their failure detectors into failure detectors of $\mathcal{S}$ and $\diamond\mathcal{S}$, respectively?". The paper has answered this question in the following way. Let $\mathcal{T}(n,f,k)$ be the predicate “there is a transformation from $\mathscr{S}_k$ (respectively, $\Diamond\mathscr{S}_k$) to $\mathscr{S}$ (respectively, $\Diamond\mathscr{S}$) in a system with n processes of which f may fail”. Sections 4.1 and 5 have shown the following:

$$(\forall n, f, k)(f < k \Leftrightarrow \mathscr{T}(n, f, k)).$$

References

[1] T. Chandra, S. Toueg, Unreliable failure detectors for reliable distributed systems, J. ACM 43 (2) (1996) 225–267.

[2] S. Chaudhuri, More choices allow more faults: set consensus problems in totally asynchronous systems, Inform. Comput. 105 (1) (1993) 132–158.

[3] D. Dolev, C. Dwork, L. Stockmeyer, On the minimal synchronism needed for distributed consensus, J. ACM 34 (1) (1987) 77–97.

[4] C. Dwork, N. Lynch, L. Stockmeyer, Consensus in the presence of partial synchrony, J. ACM 35 (2) (1988) 288–323.

[5] M.J. Fischer, N. Lynch, M.S. Paterson, Impossibility of distributed consensus with one faulty process, J. ACM 32 (2) (1985) 374–382.

[6] R. Guerraoui, A. Schiper, Γ-accurate failure detectors. in: Ö. Babaoğlu, K. Marzullo (Eds.), Proceedings of 10th Workshop on Distributed Algorithms (WDAG’96), Bologna, Italy, Lecture Notes in Computer Science, Vol. 1151, Springer, Berlin, 1996, pp. 269–285.

[7] M. Hurfin, A. Mostefaoui, M. Raynal, A versatile family of consensus protocols based on Chandra–Toueg’s unreliable failure detectors, IEEE Trans. Comput. 51 (4) (2002) 395–408.

[8] M. Hurfin, M. Raynal, A simple and fast asynchronous consensus protocol based on a weak failure detector, Distributed Comput. 12 (4) (1999) 209–223.

[9] S. Moran, Y. Wolfstahl, Extended impossibility results for asynchronous complete networks, Inform. Process. Lett. 26 (1987) 145–151.

[10] A. Mostefaoui, M. Raynal, Solving consensus using Chandra–Toueg’s unreliable failure detectors: a general quorum-based approach. in: P. Jayanti (Ed.), Proceedings of 13th Symposium on Distributed Computing (DISC’99), (Formerly WDAG), Bratislava, Slovakia, Lecture Notes in Computer Science, Vol. 1693, Springer, Berlin, September 1999, pp. 49–63.

[11] A. Mostefaoui, M. Raynal, Unreliable failure detectors with limited scope accuracy and an application to consensus, in: V. Raman, R. Ramanujan (Eds.), Proceedings of 19th International Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS’99), Lecture Notes in Computer Science, Vol. 1738, Springer, Chennai, India, December 1999, pp. 329–340.

[12] A. Mostefaoui, M. Raynal, k-set agreement with limited accuracy failure detectors, Proceedings of 19th ACM Symposium on Principles of Distributed Computing (PODC’00), Portland, OR, 2000, pp. 143–152.

[13] M. Raynal, F. Tronel, Restricted failure detectors: definition and reduction protocols, Inform. Process. Lett. 72 (1999) 91–97.

[14] A. Schiper, Early consensus in an asynchronous system with a weak failure detector, Distrib. Comput. 10 (1997) 149–157.

[15] J. Yang, G. Neiger, E. Gafni, Structured derivations of consensus algorithms for failure detectors, Proceedings of 17th ACM Symposium on Principles of Distributed Computing, Puerto Vallarta, Mexico, 1998, pp. 297–308.