Accrual Failure Detectors*

Naohiro Hayashibara

School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
E-mail: nao-haya@jaist.ac.jp

1 Introduction

Failure detection is a fundamental building block for ensuring fault tolerance in distributed systems. For this reason, many people have been advocating that failure detection should be provided as some form of service [1, 2, 3, 4, 5], similar to IP address lookup (DNS) or time synchronization (e.g., NTP). Unfortunately, in spite of important technical breakthroughs, this view has met little success so far. We believe that one of the main reasons is the conventional binary interaction (i.e., trust vs. suspect) that makes it difficult to meet the requirements of several distributed applications running simultaneously. For this reason, we advocate a different abstraction that helps decoupling application requirements from issues related to the underlying system.

It is well-known that there exists an inherent tradeoff between (1) *conservative* failure detection (i.e., reducing the risk of wrongly suspecting a running process), and (2) *aggressive* failure detection (i.e., quickly detecting the occurrence of a real crash). Thre exists a continuum of valid choices between these two extremes, and what defines an appropriate choice is strongly related to application requirements.

A major obstacle to building a failure detection service is that simultaneously running distributed applications with different quality-of-service (QoS) requirements must be able to tune the service to meet their own needs without interfering with each other. Furthermore, some classes of distributed applications require the use of different qualities of service of failure detection to trigger different reactions (e.g., [6, 7, 8]). For instance, an application can take precautionary measures when the confidence in a suspicion reaches a given level, and then take more drastic actions one the confidence raises above a second (much higher) level.

2 Solution highlight

Failure detectors are traditionally based on a binary interaction model wherein processes can only eigher trust or suspect the processes that they are monitoring [9]. In contrast, we propose a novel abstraction, called accrual failure detector, whereby a failure monitor service outputs a value on a *continuous scale* rather than information of a binary nature. Roughly speaking, this value captures the degree of confidence with which a given process is believed to have crashed. If the process actually crashes, the value is guaranteed to *accrue* over time and tend toward infinity, hence the name. It is then left to ap-

*This is a collaborative work with Xavier Défago, Takuya Katayama, Péter Urbán and Rami Yared

plication processes to set an appropriate suspicion threshold according to their own QoS requirements. A low threshold is prone to generate many wrong suspicions but ensures a quick detection in the event of a real crash. Conversely, a high threshold generates fewer mistakes but needs more time to detect actual crashes.

2.1 Concept of accrual failure detectors

The principal of failure detectors is simple. Instead of outputting information of a binary nature, accrual failure detectors output suspicion information on a continuous scale. It means that the higher the value, the higher the chance that the monitored process has crashed.

Also we revisited the interaction model between processes that require to monitor some others, and failure detectors. In conventional failure detectors decide suspicion on monitored processes and give application processes its decision as suspicion information (see Fig.1). In contrast, accrual failure detectors leave the decision on suspicion up to application processes (see Fig.2). They just provide the degree on suspicion and each application decides that a certain process should be suspected or not. Therefore, accrual failure detectors can treat different (sometime may be diverse) requirements simultaneously and provide information on suspicion to many processes in a scalable manner.

2.2 Implementation

Accrual failure detectors can be implemented in many different ways. Now we present a practical implementation of accrual failure detectors, called *the φ failure detector*. It outputs the value φ_p that means the degree of suspicion that the corresponding process p has been crashed. It is expressed on a scale that is dynamically adjusted to reflect current network conditions. Given some threshold Φ_p, and assuming that we decide to suspect p when $\varphi_p \geq \Phi_p = 1$, then the likeliness that we will make a mistake (i.e., the decision will be contradicted in the future by the reception of a late heartbeat) is about 10%. The likeliness is about 1% with $\Phi = 2$, 0.1% with $\Phi = 3$, and so on.

2.3 Mechanism

The method used for estimating φ_p is in fact rather simple. This is done in three phases. First, heartbeats arrive and their arrival times are stored in a sampling window. Second, these past samples are used to determine the distribution of inter-arrival times. Third, the distribution is in turn used to compute the current value of φ_p.

2.4 Experimental result

We have compared the results obtained using the φ failure detector with those of two adaptable failure detectors proposed by Chen and Bertier [10, 11] when used over a wide-area network (between Switzerland and Japan) and a local area network (LAN in JAIST). These adaptive failure detectors have similar mechanism for estimating the next heartbeat arrival time. However they still use timeout to decide suspicion. As we said, they are implemented based on conventional failure detectors and it means that it is difficult for them to deal with various QoS requirements simultaneously.

Now we just observe the adaptability for network condition on the φ failure detector. The purpose of the experiment is to confirm that the φ failure detector does not incur any significant performance cost.

When compared with Chen's failure detector, both failure detectors follow the same general tendency (see Fig. 3). In our experiment, the φ failure detector behaves a little better in the aggressive range of failure detection, whereas Chen's failure detector behaves a little better in the conservative range. Bertier's failure detector did not perform very well

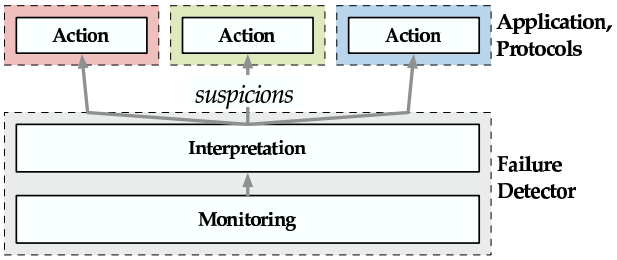

Figure 1: Interaction of conventional failure detector and application processes

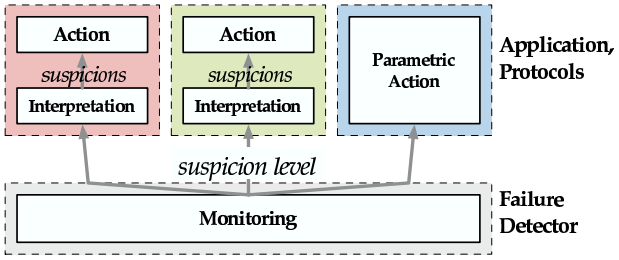

Figure 2: Interaction of accrual failure detector and application processes

in this experiment because it was primarily designed to be used over LANs.

In a LAN, the φ failure detector generated nearly one order of magnitude less mistakes than Chen's failure detector. In this experiment, Bertier's failure detector performed as a good aggressive failure detector, which seems consistent with its design goals. Although Bertier's failure detector performed better than Chen's one, the former has disadvantage that it cannot be tuned as significantly as the latter.

3 Progress and future direction

We have implemented the φ failure detector as an instance of accrual failure detectors and analyzed its experimental result [12]. In the result, we confirmed that the φ failure detector does not have any performance hit compared with other adaptable failure detectors. Moreover, it can provide information on suspicion to many applications in a scalable manner.

We are now implementing an another variation of accrual failure detectors, called *the κ failure detector* [13]. It can handle message losses. We also try to improve the precision of accuracy for failure detection. While, we are describing the definition and the specification of accrual failure detectors precisely as a theoretical framework [14]. These issues are used as a part of a generic failure detection service.

References

[1] P. Felber, X. Défago, R. Guerraoui, and P. Oser, "Failure detectors as first class objects," in *Proc. 1st IEEE Intl. Symp. on Distributed Objects and Applications (DOA'99)*, Edinburgh, Scotland, Sept. 1999, pp. 132–141.

[2] N. Hayashibara, A. Cherif, and T. Katayama, "Failure detectors for large-scale distributed systems," in *Proc. 21st IEEE Symp. on Reliable Distributed Systems (SRDS-21), Intl. Workshop on Self-Repairing and Self-Configurable Distributed Systems (RCDS'2002)*, Osaka, Japan, Oct. 2002, pp. 404–409.

[3] I. Sotoma and E. R. M. Madeira, "Adaptation - algorithms to adaptive fault monitoring and their implementation on CORBA," in *Proc. 3rd Intl. Symp. on Distributed-Objects and Applications (DOA'01)*, Rome, Italy, Sept. 2001, pp. 219–228.

[4] P. Stelling, I. Foster, C. Kesselman, C. Lee, and G. von Laszewski, "A fault detection service for wide area distributed computations," in *Proc. 7th IEEE Symp. on High Performance Distributed Computing*, July 1998, pp. 268–278.

[5] R. van Renesse, Y. Minsky, and M. Hayden, "A gossip-style failure detection service," in *Middleware'98*, N. Davies, K. Raymond, and J. Seitz, Eds., The Lake District, UK, Sept. 1998, pp. 55–70.

[6] B. Charron-Bost, X. Défago, and A. Schiper, "Broadcasting messages in fault-tolerant distributed

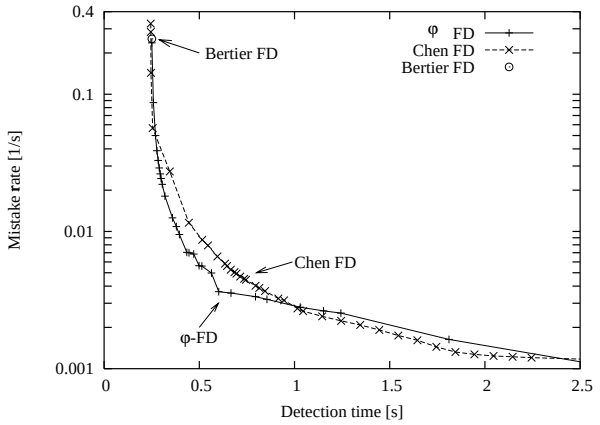

Figure 3: Comparison of failure detectors in WAN. Mistake rate and detection time obtained with different values of the respective parameters. Heartbeat sending interval is 100 ms. in this experiment

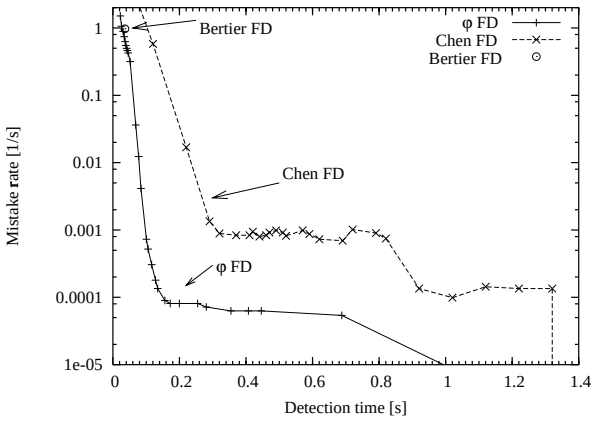

Figure 4: Comparison of failure detectors in a LAN. Heartbeat sending interval is 20 ms. in this experiment

systems: the benefit of handling input-triggered and output-triggered suspicions differently," in *Proc. 21st IEEE Intl. Symp. on Reliable Distributed Systems (SRDS-21)*, Osaka, Japan, Oct. 2002, pp. 244–249.

[7] X. Défago, A. Schiper, and N. Sergent, "Semi-passive replication," in *Proc. 17th IEEE Intl. Symp. on Reliable Distributed Systems (SRDS-17)*, West Lafayette, IN, USA, Oct. 1998, pp. 43–50.

[8] P. Urbán, I. Schnayderman, and A. Schiper, "Comparison of failure detectors and group membership: Performa nce study of two atomic broadcast algorithms," June 2003, pp. 645–654.

[9] T. D. Chandra and S. Toueg, "Unreliable failure detectors for reliable distributed systems," *J. ACM*, vol. 43, no. 2, pp. 225–267, 1996.

[10] W. Chen, S. Toueg, and M. K. Aguilera, "On the quality of service of failure detectors," *IEEE Trans. Comput.*, vol. 51, no. 5, pp. 561–580, May 2002.

[11] M. Bertier, O. Marin, and P. Sens, "Implementation and performance evaluation of an adaptable failure detector," in *Proc. 15th Intl. Conf. on Dependable Systems and Networks (DSN'02)*, Washington, D.C., USA, June 2002, pp. 354–363.

[12] N. Hayashibara, X. Défago, R. Yared, and T. Katayama, "The φ accrual failure detector," in *Proc. 23nd IEEE Int'l Symp. on Reliable Distributed Systems (SRDS'04)*, Florianópolis, Brazil, October 2004, pp. 66–78. [Online]. Available: http://ddsg.jaist.ac.jp/en/pub/HDY+04.html

[13] N. Hayashibara, X. Défago, and T. Katayama, "Flexible failure detection with $\kappa-$fd," Japan Advanced Institute of Science and Technology, Ishikawa, Japan, RR IS-RR-2004-006, February 2004. [Online]. Available: http://ddsg.jaist.ac.jp/en/pub/HDK04.html

[14] X. Défago, P. Urbán, N. Hayashibara, and T. Katayama, "Definition and specification of accrual failure detectors," in *Proc. Int'l Conf. on Dependable Systems and Networks (DSN'05) (to appear)*, Yokohama, Japan, June 2005.