Modeling of Failure Detector Based on Message Delay Prediction Mechanism

Bin Liu
Shengda Economics Trade & Management College, Zhengzhou University, Zhengzhou, China 451191
Email: liu_bin1221@yahoo.com.cn

Shifei Yang, Lei Shi
School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China
Email: sfy2008@yahoo.com.cn, shilei@zzu.edu.cn

Xiaoguang Ding
School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
Email: xgding@163.com

Qian Zhang[1, 2]
[1]School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China
[2]Henan Provincial Key Lab on Information Network, Zhengzhou 450052, China
Email: zhangqian@yahoo.com.cn

Abstract—**Failure detection is a key technology in tolerant system. Failure detectors without adaptive mechanism cannot meet the requirements of QOS (quality of service) of applications because of the variations of the network in actual distributed system. Adaptive failure detectors should dynamically adjust the detecting quality according to the variations of the real-time state of the network. Assuming that the delay and loss of the messages is a random probability, a failure detection model based on the predicted message delay is proposed in this paper. A P_{AC}-AFD adaptive failure detection algorithm is realized based on the above model which is on the basis of the prediction from historical message delay and contains checking idea. Experimental results show that the algorithm can relieve the effect of the delay and loss of the message on the failure detection while ensuring the accuracy and completeness of detection.**

Index Terms—**failure detection, QOS, distributed system, adaptive, checking.**

I. INTRODUCTION

With the development of networks, distributed systems have taken an important position in our society, and they are playing an essential role in many activities. Fault-tolerant distributed systems are designed to offer reliable and continuous service despite the failure of some its components. Normally, users of these systems expect them to remain available when there are some failures, even if some components have crashed. Because of long running time, the crashing of the hosts is inevitable, regardless of the physical reliability of each host. Thus, a system which can tolerate a reasonable number of host failures must be designed. Failure detection is a key technology to realize high reliability in distributed system which is widely applied in communication protocol, web server, and cluster management.

As an important building block for fault-tolerant systems, failure detector plays a central role in such dependable systems. Therefore, ensuring QOS of failure detector is very important for ensuring fault tolerance of distributed systems. Chandra and Toueg [1] firstly proposed metrics of unreliable failure detectors which can resolve some radical problems of unreliable system. Failure detector is defined an eventual behavior with the subsequent research of it. The research of failure detection is mainly about failure detection model, failure detection level, failure detection algorithm at present [2].

II. RELATED WORK

The state of network is multivariant in actual distributed system. At the same time, there are many kinds of applications in distributed system and different one of them has different requirements of failure detection [3]. And then, Adaptive failure detectors are presented to meet different requirements of QOS [4]. They fit the constantly variation state of network via adjusting the period of sending messages (Δti) and the value of overtime (Δt) automatically.

Fetzer [5] firstly proposed a simple adaptive failure detection mechanism which gained a maximal delay time to be the upper limit of overtime by collecting the reached heartbeat message delay. It enjoys the nice property of relying as much as possible on application messages to perform the monitoring. It uses control messages only if no application message is sent by monitoring process to observed process. The

Corresponding author, Prof. Lei Shi, Zhengzhou University measurements show that the number of wrong suspicions can be reduced through requiring each process to keep track of the maximal round trip delay time between executions. After proposing the measurement system of QOS, Chen [6] presented some adaptive algorithms based on probability network model to realize quantitative control of adjusting the parameters of failure detectors by OOS. The model uses arrival times sampled in the recent past to compute a prediction of the arrival time of next heartbeat. The overtime value is set according to this prediction and a constant safety margin which is recomputed for each interval. This technique presents a good prediction for the next arrival time. Bertier [7] and Hayashibara [8] improved Chen's adaptive failure detection algorithm realizing less detection time.

A failure detection service must propose a good QOS for users. However, so far as I know, many algorithms cannot resolve it perfectly. Each failure detector mentioned above has its shortcoming. They cannot meet the requirement of QOS perfectly.

In this paper, an adaptive failure detection method which is based on the prediction from historical record of messages delay time and contains checking idea is used after studying and analyzing the existing adaptive failure detection algorithm. It realizes an adaptive failure detector and relieves the effect of message delay and message loss on the failure detection to some extent ensuring the accuracy and completeness of detection.

In this paper, we first talk about motivation and related work about failure detection. And then the main contributions of this thesis are presented.

III. BASAL THEORY OF FAILURE DETECTOR

Failure detection is the process that the detector detects the detected process to estimate whether it is still alive or not at sometime through making use of definite detection model and algorithm. Different failure detector has different detection capability.

A. Failure Detector Model

The model of failure detector can be defined as follows [1]: assume a system with N processes: $\Pi = \{P_1, P_2, \ldots P_n\}$. There is an independent global clock in the system and a time set T obtained from clock time signal which is natural number. Suppose $p \in \prod$, $t \in T$, then failure detector can be defined as: $FD_p(t)$: $\Pi \times T \rightarrow 2^{\Pi}$. For $q \in \Pi$, if $q \in FD_p(t)$, the failure detector of p deems that q is failed at t. The output of FD is a set of failed objects: $Failed = \cup_{t \in T} FD_p(t)$. Suppose that failure detector will not fail itself and will fail only if the detected nodes fail. Failure model of nodes fits Fail-stop [9] model.

For a detected process q, there are two basal states: up and down. The state up means the detected process q runs normally, the state down means the detected process q is failed. In a general way, failed detector is based on the eventually assumption, it means that a suspected process will eventually become failed. But the above method does not consider the actual time delay in distributed system. Therefore, the state suspect should be used. The state show that the detected process q which is not judged failed or not definitely is suspected failed. It means that the detected process q is possibly judged failed falsely. Suspect is the set of processes which are suspected failed in running of system, therefore, $failed \subseteq suspect$. Obviously, there are four state transition of detected process: US transition, UD transition, SU transition and SD transition. Suspect is the mid state, which is transparent to failure detector. US transition is carried out or not in failure detection system depends on the method of it. The figure of state transition for failure detector is like fig1.

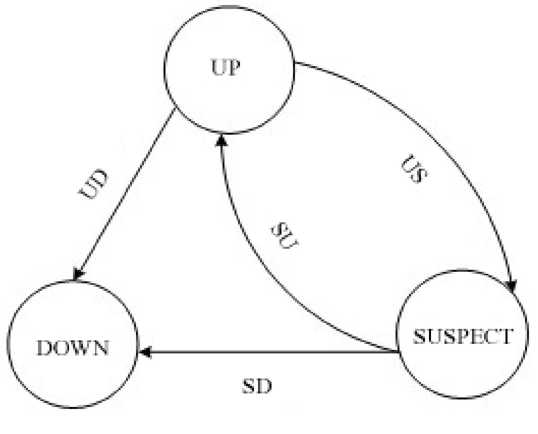

Figure 1. Transition of failure detection state

We give two classifications of failure detectors according to different behavior of detected processes: Unreliable failure detector and Byzantine failure detector. Byzantine failure detector is based on the state machine mechanism and the code mechanism. Unreliable failure detector is on the basis of overtime mechanism.

B. Failure Detector Level

Failure detector has two basic metrics: completeness and accuracy. Completeness stands for the capability that all the correct applications judges eventually every failed process perpetually failed and accuracy stands for the probability that all correct processes will not be considered failed. Chandra and Toueg [1] firstly give the formal definition of failure detector and classified it according to accuracy and completeness.

Completeness can be classified two types:

Strong Completeness: Every failed process will eventually be judged failed perpetually by all the correct applications in any running. It can be formally described: $\exists t0$: $\forall t \geq t0$, $\forall p \in correct(t)$, $\forall q \in failed$, $q \in suspectp(t)$.

Weak Completeness: Every failed process will eventually be judged failed perpetually by partial correct applications in any running. It can be formally described: $\exists t0$: $\forall t \geq t0$, $\forall p \in correct(t)$, $\exists q \in failed$, $q \in suspectp(t)$.

Accuracy contains four types:

Strong Accuracy: Every correct process will not be judged failed perpetually in any running. It can be formally described: $\forall t$, $\forall p,q \in correct(t)$, $q \notin suspectp(t)$.

Weak Accuracy: Partial correct process will not be judged failed perpetually in any running. It can be formally described: ∀t, ∃q∈correct(t), ∀p∈correct(t), q∉suspectp(t).

Eventual Strong Accuracy: Every correct process will not be judged failed after some moment t in any running. It can be formally described: ∃t0: ∀t≥t0, ∀p,q ∈ correct(t), q∉suspectp(t).

Eventual Weak Accuracy: Partial correct process will not be judged failed after some moment t in any running. It can be formally described: ∃t0, ∀t≥t0, ∃q∈correct(t), ∀p∈correct(t), q∉suspectp(t).

Strong accuracy and weak accuracy are perpetual accuracy, thus eventually strong accuracy and eventually weak accuracy are eventually accuracy. Strong accuracy is the highest level accuracy. It means that a failure detection system which can fit the strong accuracy can meets every other accuracy level. Eventually weak accuracy is the lowest level accuracy. But, we can not compare eventually strong accuracy and weak accuracy in theory.

Failure detector can be classified eight classifications [10] according to completeness and accuracy, as Table1.

TABLE I
Eight Classifications of Failure Detector

	Strong completeness	Weak completeness
Strong accuracy	P	Q
Weak accuracy	S	W
Eventual strong accuracy	◇P	◇Q
Eventual weak accuracy	◇S	◇W

Completeness contains two types:

Chandra and Toueg give eight classifications of failure detectors according to accuracy and completeness: P, S, W, Q, ◇P, ◇S, ◇W and ◇Q.

Generally speaking, a good failure detector should meet the definition of ◇P (a set of eventually perfect failure detector) which should fit strong completeness and eventual strong accuracy.

Failure detectors which meet strong completeness have four classifications:

P (perfect failure detection): Failure detection of this level meets strong accuracy and strong completeness.

S (strong failure detection): Failure detection of this level meets weak accuracy and strong completeness.

◇P (eventual perfect failure detection): Failure detection of this level meets eventual strong accuracy and strong completeness.

◇S (eventual strong failure detection): Failure detection of this level meets eventual weak accuracy and strong completeness.

Failure detectors which meet weak completeness have four classifications:

Q: Failure detection of this level meets strong accuracy and weak completeness.

W (weak failure detection): Failure detection of this level meets weak accuracy and weak completeness.

◇Q: Failure detection of this level meets eventual strong accuracy and weak completeness.

◇W (eventual weak failure detection): Failure detection of this level meets eventual weak accuracy and weak completeness.

P failure detector meets strong accuracy and strong completeness. Therefore, it is the perfect failure detector. As Q≌P, W≌S, ◇Q≌◇P [11], therefore, generally, we only study four classifications of failure detectors: P failure detector, ◇P failure detector, S failure detector, ◇S failure detector. The relationship of their failure detection level is shown as figure2. We cannot compare the failure detection level of S failure detector with the failure detection level of ◇P failure detector.

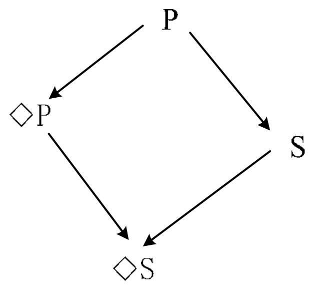
$$\begin{tikzcd} & P \arrow[dl] \arrow[dr] & \\ \Diamond P \arrow[dr] & & S \arrow[dl] \\ & \Diamond S & \end{tikzcd}$$

Figure 2. Order of the detection level of four failure detectors

C. Failure Detector Technical

Failure detection is a key technical implementing high availability. The research of failure detection is mainly about failure detection model, failure detection level, failure detection algorithm at present. Heartbeat is general technical implementing failure detection. According to different implementing way, it can be classified two types which are push and pull.

Push mode failure detection: In push mode failure detection, the detected process sends the message "I am alive" to its failure detector periodically to tell them its normal state. If failure detector does not receive the message from detected process in a certain time interval, it will suspect detected process failed. There are two important parameters in this mode: heartbeat service interval and overtime interval. In addition, detected process is active in this mode. It sends the message to its failure detector continuously. Its way of message exchange is shown as the following figure3.

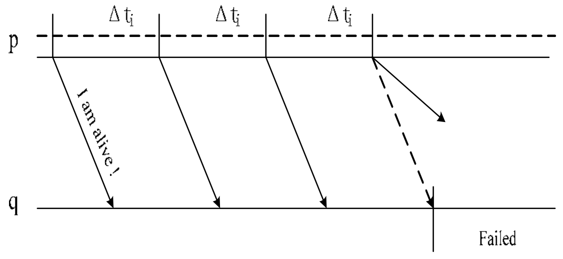

Figure 3. Process of push mode failure detection

Pull mode failure detection: In pull mode failure detection, failure detector is active. It sends inquiry message “are you alive” to detected process periodically and detected process gives a response message “I am alive” to tell it its normal state. If failure detector does not receive the response message from detected process in a certain interval, it will suspect detected process failed. Be similar to push mode failure detection, there are two important parameters in pull mode failure detection which are inquiry interval of failure detector and overtime interval of response message. In addition, failure detector is active in this mode. Its way of message exchanges is shown as the following figure4.

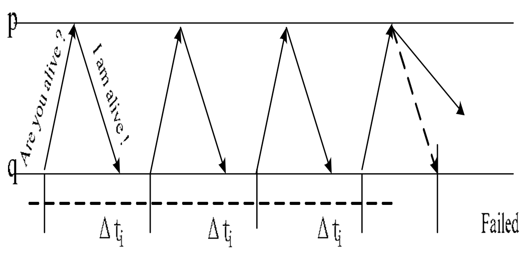

Figure 4. Process of pull mode failure detection

Traditional failure detector which is usually set an overtime value Δt is on the basis of overtime mechanism. Detected process will be judged failed if failure detector does not receive the predicted message in Δt. Obviously, when we use pull mode to failure detection, failure detector need to send inquiry message and receive response message from detected process. It means that the time of failure detection will become bigger. In addition, there will be twice messages increasing network load because that failure detector need to send inquiry message and detected process need to send response message. But pull mode is an active detection method which apply to the feature that changing detection state usually in distributed system so that failure detector can control detection time point and change the period of sending heartbeat according to different requirement. Thus, in push mode failure detection, detection period will not be changed easily after it is set. The most different feature between pull mode and push mode is that they have different overtime basis of judging failed or not. Pull mode is used to absolute delay of heartbeat message and push mode used to relative delay of heartbeat message. Of course, the variety of relative delay is leaded by actual absolute delay.

IV. Failure Detection Model Based Message Delay Prediction

Adaptive failure detection algorithm based on the predicted message delay is proposed in this paper. After analyzing and computing the record of historical message delay, the algorithm proposes an overtime value which has strong adaptability, changing along with the variation of message delay. In addition, it can meet different requirements of QOS through adjusting parameters, realizing a universal adaptive failure detector to some extent.

The frequency that a correct process is misjudged failed by other correct processes in a running of system is called error rate, which is a staple norm to measure the accuracy of the failure detection system. In distributed systems, the error rate is caused by the reason that heartbeat messages loss and heartbeat messages delay exceed the default overtime value. These two events are independent. It means: when a message is loss, it cannot be overtime and a message will eventually be received if it is only late because of overtime. The current network state will influence the heartbeat message loss and heartbeat message delay which is also impacted by the QOS demand of failure detection. In this paper, P_A called query accuracy is used to represent the QOS demand of application, here, $P_A \in (0, 1)$. It can be set flexibly according to different applications to meet different requirements of QOS. When detection accuracy is more important, P_A is set with a large value. When detection speed is more important, P_A is set with a small value.

A. Basic Failure Detector Algorithm

Definition 1: Let $\sup\{x:p(t\le x)=1\}$ and $\mathrm{low}\{x:p(t\ge x)=1\}$ which are called sup and low for short respectively be the upper and lower bounds for the random variable t. Obviously, if t has bound, then sup and low are bounded and exclusive.

Definition 2: If the random variable t has its bound, then we call $A(t)=\frac{\sup(t)+\mathrm{low}(t)}{2}$ the arithmetic mean of t. And if low≥ 0, we call $G(t)=\sqrt{\sup(t)\times \mathrm{low}(t)}$ the geometric mean of t.

Theorem 1: Let t be a random and bounded variable, the mathematical expectation and variance of t are E(t) and V(t), and low>0, then

$$V(t)\le 2E(t)*[A(t)-G(t)] \quad (1)$$

Proof: Because $P\{(t-m)(M-t)\ge 0\}=1$, then $E[(t-m)(M-t)]\ge 0$. Here, m= low (t), M= sup (t). It can be obtained above that:

$$E(t^2)\le (M+m)E(t)-Mm=2A(t)E(t)-G^2(t)$$

So

$$\begin{aligned} V(t) &= E(t^2)-E^2(t) \\ &\le 2A(t)E(t)-G^2(t)-E^2(t) \\ &\le 2A(t)E(t)-2G(t)E(t) \\ &= 2E(t)*[A(t)-G(t)] \end{aligned}$$

Theorem 2: Assume that E(D) is the mathematical expectation of D which is the interval between two heartbeat messages delay. Then for arbitrary t>0,

$$P(D>t)\le \frac{2E(D)*[A(D)-G(D)]}{[t-E(D)]^2} \quad (t>E(D)) \quad (2)$$

Proof: Because the interval between two heartbeat messages is a bounded random variable, so it can be obtained from theorem 1 that:

$$V(D) \le 2E(D) * [A(D) - G(D)]$$

Here we assume that V(D) is the variance of D which is the interval between two heartbeat messages delay. The interval between two heartbeat messages delay is a random probability event, so it can be obtained from Chebyshev inequality that:

$$P(D > t) \le \frac{V(D)}{[t - E(D)]^2} \quad (t>E(D)) \qquad (3)$$

So, it can be obtained from (1) and (3) that:

$$P(D > t) \le \frac{2E(D) * [A(D) - G(D)]}{[t - E(D)]^2} \quad (t>E(D))$$

The algorithm's realization can be divided into two parts: sending and receiving messages and predicting message delay. It is not necessary to build a permanent connection between the servers and clients of failure detection services. UDP protocol which does not require connectivity and control has a high efficiency being- used to realize the communication process.

Each node sends heartbeat messages to other failure detectors using UDP protocol packets periodically. Heartbeat message contains the sequence number of the message which is increasing with the number of it. The heartbeat messages sent out by every failure detector are uniform and the sequence number of the messages is consistent. The sequence numbers sent to the new joined detected nodes after running the failure detector do not start from zero, but consist with the sequence numbers sent to other detected nodes. The interval of sending heartbeat message of each node is consistent.

Statistic the intervals of recent N heartbeat messages from detected nodes calculating E(D)、A(D) and G(D), to propose the possible delay time of the next heartbeat message. On this basis, historical predicted interval is calculated with weighted value. Assume that the predicted time of this time is T_1 with a weighted value P_1, the previous predicted time is T_2 whit a weighted value P_2……the predicted time of the previous ith time is T_{i+1} with a weighted value P_{i+1}……the predicted time of the previous w-1th time is T_w with a weighted value P_w. Here w is the window size of historical record. In addition， $\sum_{i=1}^{w} P_i = 1$ and $P_i = \frac{K}{i}$ can be obtained from the first Zip law. K is a parameter which can be known from normalization calculation. $\sum_{i=1}^{w} P_i = \sum_{i=1}^{w} \frac{K}{i} = K\sum_{i=1}^{W} \frac{1}{i} \approx K*(\ln w + r) = 1$, r is Euler constant. $K \approx \frac{1}{\ln w + r}$.

The overtime value of timer for the next heartbeat message reaching can be set according to the reaching interval of historical heartbeat messages and the predicted possible interval for the next heartbeat message delay. If the detector does not receive the heartbeat messages from the detected node in predicted delay interval, it will be checked. The checking process is:

Detection process p sends inquiring messages to the detected process q. The format of inquiring message is ask(q, count), here q represents the process inquired and count represents the sequence number of heartbeat messages sent to inquired processes. If process p receives the response message ack(q, count, yes) from process q in the predicted time, it will be considered normal. If process p does not receive the response message in the predicted time, process q will be considered failed. This can improve the accurate of the failure detector.

```
The $P_{AC}$-AFD algorithm is described as follows:
Input: heartbeat messages;
Output: status of the process being detected;
1.  for process p and q, initialize UDP socket;
2.  initialize others correlative arguments;
Process q:
3.  if current time is i*□t
4.  send heartbeat message to process p; /* □ t is the
    period of sending heartbeat message */
Process p:
5.  initialize that process q is live;
6.  loop
    {
7.  timer ($t_n$) start; /* $t_n$ is the predicted interval of
    heartbeat delay */
8.  wait for receiving message from q;
9.  if $t_c \le t_n + t_p$ and $k < s_{min}$ /* received overtime message
    $t_c$ is the reaching time currently of the heartbeat
    message， $t_p$ is the reaching time of the previous
    heartbeat message, $s_{min}$ is the smallest sequence
    number of the heartbeat message which does not
    receive its response message up to now */
     {
10. judge that process q is live;
11. timer($t_{n+1}$) restart;
     }
12. else if tc≤tn+tp and k≥smin /* received message
    which is being waiting */
     {
13. j←k%w; /* k is the sequence number of the
    response message of q */
14. $t_d \leftarrow t_c - t_p$; /* $t_d$ is the detection time of heartbeat
    message */
15. add($t_d$) to sw[j];/* Save the detection time into the
    sliding window*/
16. $s_{min} \leftarrow k+1$; $t_p \leftarrow t_p + $□t;
17. calculate out E(D)、A(D)、G(D);
18. $T_d = \sqrt{\frac{2E(D)*[A(D)-G(D)]}{1-P_A}} + E(D)$;
    /* the possible arrived delay time of the next
    message*/
19. $t_n = T_d P_1 + \sum_{i=2}^{w} T_i P_i$;
    (4)
    /* the predicted arrival time of the next message*/
     }
20. else
     {
```

```
21. send inquiring message ask(q, count);
22. if receive ack(q, count, yes) in $t_r$ /* $t_r$ is the
    predicted response time of inquired processes*/
    {
23. judge that process q is live;
24. timer(tn+1) restart;
    }
25. Else
    {
26. judge that process q is dead;
27. tp←tp+□t;
28. add tn to sw[smin%w];
29. smin←smin+1;
    }
    }
    }
```

B. *Failure Detection Level of the Algorithm*

The algorithm proposed in this paper meets strong completeness and eventual strong accuracy. In order to facilitate the proof, we assume that there is an upper limit of message delay which is unknown in each running.

Property 1(Strong Completeness): Algorithm meets strong completeness. Based on the assuming that there is an upper limit of heartbeat message delay, every correct process will receive the last heartbeat message from process q at sometime T_i ($i = 1, 2, \ldots, n-1$). For correct process P_1, assume that it receives the last heartbeat message from process q at time T_1. We can know from the algorithm that process P_1 will calculate an interval t_n of message delay for the next heartbeat message sent by process q after receiving a heartbeat message from it. If process P_1 does not receive the heartbeat message from process q in that interval, then the checking will be carried out. During checking, if P_1 cannot receive the message ack(q,count,yes) in predicted interval, then P_1 will consider q failed. We know that all the data needed to calculate the next heartbeat message delay in this algorithm, including E(D)、A(D) and G(D), has a determinate value. So, the interval of message delay is bounded.

We know from the algorithm that the total detection time is bounded. We use T_{fi} to represent the value of this upper limit and each process has that moment. Let T_{fm}= max $\{T_{fi}\}$ as its maximum value, so all processes consider that process q has failed after T_{fm}. It means: $\exists T_{fm}$: $\forall t \geq T_{fm}$, $\forall p \in$ correct(t), $\forall q \in$ failed, $q \in suspect_p(t)$ so, algorithm meets strong completeness.

Property 2(Eventual Strong Accuracy): Algorithm meets eventual strong accuracy. If the message delay time calculated by the algorithm is smaller than the actual message delay time, heartbeat message will arrive after the delay time predicted by the algorithm, then the process will be checked to judge whether it is failed or not. During checking, if detecting process does not receive the response message from detected process in the predicted time, the detected process will be considered failed. In this algorithm, assume that the heartbeat message is overtime, process p will dynamically increase detection time to adapt the changes of networks of the process q.

The detection time will increase with time increasing. Until after a certain time t_0, process p will receive heartbeat message sent by process q in detection time. It means: $\exists t_0$: $\forall t \geq t_0$, $\forall p$, q $\in$correct (t), $q \notin$ suspect p (t). Therefore, algorithm meets eventual strong accuracy.

V. Experiment and Analysis

In order to prove the effectiveness of the algorithm P_{AC}-AFD, we do the experiment in the actual environment. The configuration of the environment is as follows: Two computers with the same configuration are connected via the internet. One is detector and the other is a detected machine. The detected machine sends heartbeat messages to the detector periodically, and the detector will return a response message to the detected machine after receiving the heartbeat message from the detected machine. The interval between two adjacent heartbeat messages is one second. The detection metrics are error rate and average detection time in different window size and different value of P_A.

A. *Impact of P_A Values on the Detection*

Firstly, we set the window size of historical a fixed value 1000, and P_A is set a value 0.6, 0.7, 0.8 and 0.9 respectively. Then we compare the obtained detection time and the actual delay time of the message. Fig5 shows a group of heartbeat message sequence selected randomly in 24 hours which has a consecutive sequence number. We can see from the chart: The larger the P_A is, the longer the detection time will be.

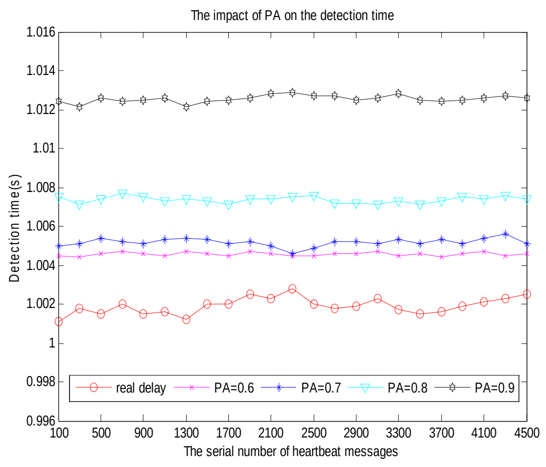

Figure 5. Impact of P_A on the detection time

It can be seen from the table 2: P_A has larger influence on error rate, that is, the larger the P_A is, the longer the average detection time is and the smaller the error rate is. The result meets predicted effect.

TABLE II
Impact of P_A on the average detection time and error rate

P_A	0.6	0.7	0.8	0.9
Average detection time(s)	1.004 257	1.005 719	1.007 347	1.012 531
Error rate	0.011 697	0.005 238	0.001 072	0.000 142

B. *Impact of Historical Window size on the Detection*

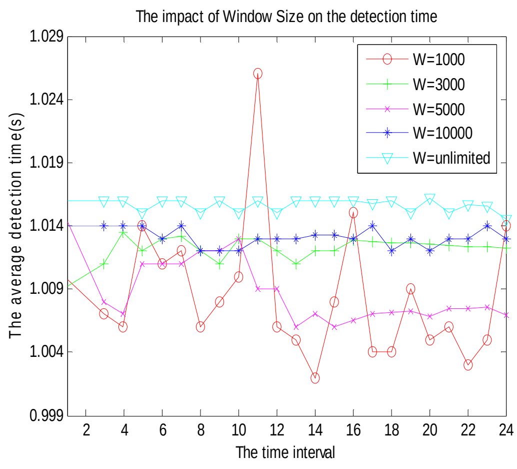

Figure 6. Impact of window size on the detection time

From Fig6 we can see that the average detection time becomes larger with the size of window becoming larger and the average detection time becomes smaller with the size of window becoming smaller. Here, the average detection time is the longest when the size of window is unlimited.

TABLE III
Impact of window size on the average detection time and error rate

Window size	1000	3000	5000	10000	unlimited
Average detection time(s)	1.012 532	1.012 792	1.013 082	1.013 962	1.015 672
Error rage	0.000 143	0.000 117	0.000 085	0.000 051	0.000 026

Fig 6 and Table 3 show that: The size of window has a large impact on the fluctuation of average detection time. The smaller the size of window is, the larger the fluctuation of average detection time is and the higher the error rate is. The larger the size of window is, the smaller the fluctuation of average detection time is and the lower the error rate will be.

C. *Analysis of Performance*

We compare our algorithm the with Chen's failure detector in the same network environment. In our algorithm, the size of window is set a value 1000 and the period of sending heartbeat message is one second. In Chen's algorithm, safe margin α is set a value 0.005. We set P_A a value 0.85. Table 4 shows: When the size of window is 1000, α is 0.005 and P_A is 0.85, P_{AC}-AFD algorithm has smaller error rate than Chen's algorithm ensuring almost same average detection time. The performance of failure detector is improved.

TABLE IV
Comparison with Chen's algorithm

	P_{AC}-AFD	Chen
Parameter	P_A =0.85	α=0.005
Window size	1000	1000
Average detection time(s)	1.011792	1.011835
Error rate	0.001023	0.002442

D. *Conclusions*

A predicted method on the basis of the prediction from historical message delay is studied in this paper. The checking idea is used in this method. The algorithm predicts the next message delay time according to the historical record of message delay based on the premise that message transmission and message loss in failure detection are random. The analysis of experiment and performance proved the failure detection level of the new algorithm. Experimental results show that the algorithm has strong adaptability and it can relieve the effect of message delay and message loss on the failure detection while the detection accuracy and detection completeness is satisfied.

REFERENCES

[1] T. D Chandra and S. Toueg, "Unreliable Failure Detectors for Reliable Distributed Systems," Journal of the ACM. vol. 43(2), pp. 225-267, 1996.
[2] N. Xiong, Design and Analysis of Quality of Service on Distributed Fault-tolerant Communication Networks [Ph.D. Thesis], Japan Advanced Institute of Science and Technology. 2008.
[3] J. Dong, Research on key techniques of failure detection in distributed systems [Ph.D. Thesis]. Harbin institute of Technology. 2007.
[4] N. Xiong, A.V Vasilakos and L Yang, "Comparative analysis of quality of service and memory usage for adaptive failure detectors in healthcare systems," IEEE Journal on Selected Areas in Communications. vol. 27(4), pp. 495-509, 2009.
[5] C. Fetzer, M Raynal and F Tronel, "An adaptive failure detection protocol," IEEE the 8th Pacific Rim International Symposium on Dependable Computer. pp. 146-153, 2001.
[6] W. Chen, S. Toueg and M.K Aguilera, "On the quality of service of failure detectors," IEEE Trans. on Computers. vol. 51(5), pp. 561-580, 2002.

[7] M. Bertier, O. Marin and P Sens, "Implementation and performance evaluation of an adaptable failure detector," Martin DC, ed. Proc. of the 15th Int'l Conf. on Dependable Systems and Networks. Bethesda, IEEE CS Press. pp. 354-363, 2002.

[8] N. Hayashibara, X. Defago and T Katayama, "Implementation and performance analysis of the φ -failure detector," JAIST Research Report. IS-RR-2003-013, 2003.

[9] Yair, Amir, Danny, Dolev, Shlomo, Kramer, Dalia and Malki, "A communication sub-system for high availability," in the proceedings of the 22nd Annual International Symposium on Fault-Tolerant Computing. Boston, pp. 76−84, 2002.

[10] I. Gupta, T. Chandra and G Goldszmidt, "On scalable and efficient distributed failure detectors," in the proceedings of 20th Annual ACM Symposium on Principles of Distributed Computing. pp. 170-179, 2001.

[11] Y. Horita, K. Taura and T Chikayama, "A scalable and efficient self-organizing failure detector for grid applications," in Katz DS,ed. proc. of the 6th IEEE/ACM Int'l Workshop on Grid Computing. Washington. pp. 202-210, 2005.

Bin Liu received the M. Sc. degree (2003) in Multi-media and virtual reality from LEUVEN GROUPT College. Now he is a lecturer in Shengda Economics Trade & Management College, Zhengzhou University. His research interests are in the areas of distributed and Internet systems, 2D-3D image convert and transmit, web mining and high graphic performance computing.

Shifei Yang received his B.Sc. degree from Zhongyuan University of Technology, China, in 2007. He is now a M.Sc. candidate of Zhengzhou University. His research interests include distributed Internet systems, failure detection, web mining.

Lei Shi received the M.Sc. degree (1992) in computer science from Nanjing University and Ph.D. degree (2006) in computer science from Beijing Institute of Technology. He has published over 70 papers in journals and international conferences and 5 books in his research field. More than 20 papers are indexed by SCI, EI and ISTP. Currently he is a full professor in the School of Information Engineering, Zhengzhou University. He served as workshop chair, program committee member of many international conferences. His research interests are in the areas of distributed and Internet systems, web prefetching, web mining and high performance computing.

Xiaoguang Ding received the M.Sc. degree in computer science from Shenyang Institute of Technology, China, in 2000. Now he is a PhD candidate in computer science in Beijing Institute of Technology. His research interests include distributed Internet systems, failure detection, and high availability.

Qian Zhang received her B.Sc. degree from PLA Information Engineering University, China, in 2008. She is now a M.Sc. candidate of Zhengzhou University. Her research interests include peer-to-peer overlay networks and media streaming.