Cooperative Failure Detection in Overlay Multicast *

Mengkun Yang and Zongming Fei

Department of Computer Science, University of Kentucky,
301 Rose Street, 2nd floor, Lexington, KY 40506, USA
myang0@cs.uky.edu, fei@cs.uky.edu

Abstract. Node failures and ungraceful departures are important issues to be dealt with in overlay multicast. Fast detection is key to minimizing the disruption of service to the affected nodes participating in the multicast session. In this paper, we propose a cooperative failure detection mechanism that can greatly reduce the failure detection time. A significant contribution of the paper is that we quantify three important measures, i.e., the expected detection time, the probability of false failure detection, and the overhead. This allows us to study the fundamental tradeoff among them in the failure detection mechanisms. The analysis and simulations show that the proposed cooperative failure detection mechanism can significantly reduce the failure detection time while maintaining the probability of false positive at the same level, at the cost of slightly increased overhead.

1 Introduction

Overlay multicast (also known as application-layer multicast) [1,2] has been widely investigated as an alternative to IP multicast to implement group communications for its easy deployment. It builds an overlay topology (usually a tree) among end hosts participating in the multicast session, by using the unicast service provided by the substrate network. The duplication functions are implemented at end hosts rather than routers.

Management of the overlay multicast tree faces a key problem. The non-leaf nodes in the tree are end hosts, which are more likely to fail than routers and may leave the multicast tree voluntarily without informing other nodes. In these cases, all of its downstream nodes are partitioned from the multicast tree and cannot get the multicast data any more. It is important to recover from partitioning quickly so that the disruption of service to those downstream nodes is minimized. The time to resume the data flow to those affected nodes is an important measure of the *responsiveness* of failure recovery mechanisms.

The recovery process consists of two steps, *failure detection* and *tree reconstruction*. Failure detection means that when a node in overlay multicast fails or leaves the multicast session, other nodes can detect the event. A departing node may leave the multicast tree gracefully, by sending a message to relevant nodes. The detection is not a problem in this case. However, it is not uncommon in overlay multicast that a node leaves the tree accidentally, such as in case of failures, or leaves voluntarily without informing other nodes about its status. We need a detection mechanism for affected nodes

* This work was supported in part by the National Science Foundation under Grants CCR-0204304 and EIA-0101242.

to reach the conclusion that the node is gone as soon as possible. In the rest of the paper, we will use “failure” or “a node fails” to include the case in which a node leaves the multicast tree without informing others.

The tree reconstruction is the process that those orphaned nodes or subtrees find new parents to reconnect to the multicast tree. Several recent researches have addressed the problem. A centralized approach depends on a coordinator to find the locations in the tree for those affected nodes [3]. SpreadIt adopts a distributed approach in which disconnected nodes try to get help from their grandparents or the root of the tree [4]. A proactive approach pre-computes rescue plans before failures happen to reduce the recovery time [5]. They improve the performance of the tree reconstruction process after failures have been detected.

In contrast, failure detection itself has not been widely studied in the context of overlay multicast. They are usually described as a part of the tree construction procedure [1,6], rather than as a focal topic being investigated as it deserves. Lack of quantitative analysis of different detection mechanisms hinders a full exploration of different design options.

In this paper, we perform an analysis of a basic heartbeat mechanism and study the fundamental tradeoff among the failure detection time, the probability of false positive, and the overhead. Based on that, we propose a cooperative approach to node failure detection in overlay multicast to speed up the detection process. The basic idea is that all neighbors interested in the well-being of a node cooperate with each other. To detect the failure of a non-leaf node in overlay multicast, all its children and its parent can form a logical group to help each other to make a decision. When any of them loses a heartbeat, it tells others in the group about this event. If a node cannot receive the heartbeats from the target node, and also receive information from cooperating nodes that heartbeats are missing, it can quickly reach the conclusion with higher confidence that the target node has failed. We perform a formal analysis of the cooperative scheme and design a probabilistic notification technique to deal with big group sizes. Through analyses and simulations, we find that the cooperative scheme can achieve shorter failure detection time and a smaller probability of false positive at the same time, with a small increase of overhead, compared with the basic heartbeat mechanism.

The rest of the paper is structured as follows. Section 2 proposes the cooperative failure detection mechanism. Section 3 gives an analytical study on the gains and costs of different detection schemes. Performance evaluations are presented in Section 4 and related work is discussed in Section 5. We conclude the paper in Section 6.

2 A Cooperative Failure Detection Mechanism

The node failure and its detection can be illustrated by an example in Fig. 1. When non-leaf node 5 fails, all the links (shown as dotted lines) between 5 and other nodes will be affected. All the downstream nodes, 8, 9, 10, and 15-22, are affected and experience data disruptions. Failure detection is the process that neighboring nodes (e.g., nodes 2, 8, 9, and 10) come to a conclusion that the node being monitored (node 5) has failed.

In the following discussion, the node being monitored will be called the *target node*. The nodes participating in the failure detection of the target node are usually its *neigh-*

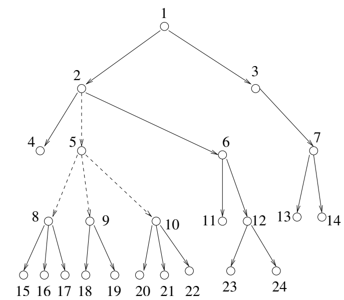


Fig. 1. An example of node failure

bors, such as nodes $2, 8, 9$, and 10 for target node 5. In the discussion, we also use terms *monitor nodes* and *detecting nodes* interchangeably with term *neighbors*.

2.1 Basic Heartbeat Scheme

The basic heartbeat scheme works as follows. Every node sends a heartbeat message to each of its monitor nodes every $\mathcal{T}$ seconds. If a monitor node does not receive k heartbeat messages in a row, it will derive that the target node has failed. k and $\mathcal{T}$ are design parameters that can be adjusted. They determine the performance of the detection mechanism.

Since each node independently makes its own decision about the failure of other nodes based on its observation on the heartbeat losses, we call this basic heartbeat scheme *non-cooperative* failure detection in our later discussion, in contrast to the cooperative scheme discussed next.

2.2 Cooperative Scheme

The idea of the cooperative scheme is to let monitor nodes of a target node cooperate with each other as a group so that each node can reach the conclusion faster. The goal is that in most cases, a monitor node can detect the failure of a target node within one heartbeat interval, with the help of other nodes in the same group.

Similar to the non-cooperative case, every target node sends a heartbeat message to each of its monitor nodes every $\mathcal{T}$ seconds. When a monitor node does not receive the heartbeat message at the expected time, it will send a notification message to each node in the cooperating group with regard to the same target node. When a monitor node receives the heartbeat message, no notification will be sent. So no extra traffic is generated when the target node works normally.

For each target node, each monitor node maintains two counters in the cooperative scheme. One is the number of heartbeats lost, denoted as N_h, and the other is the number of notifications received, denoted as N_n. They are both initialized to 0.

– When a monitor node receives a heartbeat message from the target node, it resets both counters, N_h and N_n, to 0.

– When it does not receive an expected heartbeat from the target node, it increases the lost heartbeat counter N_h by 1, sends a notification to each node in the group, and performs the following check. If $N_h + N_n \geq k$, it concludes that the target node has failed.
– When it receives a notification from other nodes in the cooperating group with regard to the target node, it increases the notification counter N_n by 1 and performs the following check. If $(N_h + N_n \geq k) \wedge (N_h > 0)$, it concludes that the target node has failed.

Note k is a threshold parameter that can be adjusted. It can be different from the k in the basic heartbeat scheme. A monitor node reaches the conclusion that the target has failed only after it has lost at least one heartbeat from the target node.

3 Analytical Study

To better understand the gains and cost of the failure detection approaches, we perform a formal analysis in this section. Specifically, we want to quantify the following performance measures.

1) *Failure Detection Time.* It is the interval between the time a node fails and the time a monitor node reaches the conclusion that it has failed.

2) *Probability of False Positive.* It is the probability that while a target node works fine, a monitor node reaches the conclusion that it has failed. This may happen when the heartbeat messages are lost.

3) *Overhead.* It is the traffic generated for the failure detection purpose. Specifically, we calculate the average number of messages generated per unit time for *one* monitor node to be able to detect the failure of a target node.

We assume the probability of message loss between any pair of nodes in the overlay multicast tree is p $(0 \leq p \leq 1)$ and losses are independent. For clarity of analysis, we assume the end-to-end path latency between two nodes is negligible and a monitor node knows a heartbeat is lost at the exact same time when the heartbeat is supposed to be sent out. These paths include those not in the overlay tree and yet used in the cooperative failure detection. Every node in the overlay multicast tree fails with probability q $(0 \leq q \leq 1)$. In the cooperative failure detection, we assume the number of nodes in a cooperating group is n. When $n = 1$, there is no other node in the group. The other two important parameters are threshold value k, which determines when a monitor node concludes that a target node has failed, and the heartbeat interval $\mathcal{T}$, which specifies how frequently the heartbeat messages are sent out.

3.1 Failure Detection Time

Non-cooperative case In the non-cooperative failure detection, a node can detect the failure of a target node if the number of heartbeats it has missed in a row exceeds the threshold k. Therefore, the failure detection time T_{nc} is between $(k-1)\mathcal{T}$ and $k\mathcal{T}$, as shown in Figure 2. Specifically, $T_{nc} \approx (k-1)\mathcal{T}$ if the target node fails immediately before the expected time at which the next heartbeat should be sent out (point B); and

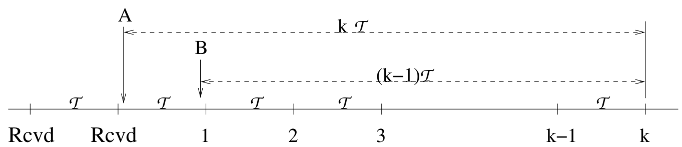


Fig. 2. Failure detection time for the non-cooperative approach

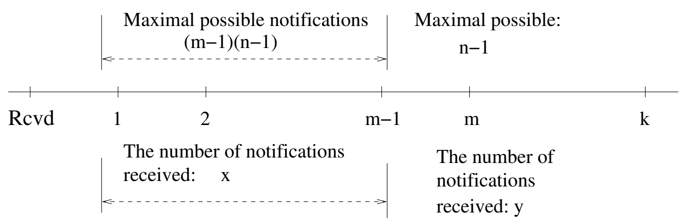


Fig. 3. Failure detection time for the cooperative approach

$T_{nc} \approx k\mathcal{T}$ if the target node fails immediately after a heartbeat is sent out (point A). Assuming that the failure event is uniformly distributed in the interval (between point A and point B), we can get the expected failure detection time for the non-cooperative approach as $E(T_{nc}) = \left(k - \frac{1}{2}\right)\mathcal{T}$.

Cooperative case In the cooperative failure detection, the longest failure detection time is $T_{co} = k\mathcal{T}$, when the target node fails immediately after a heartbeat is sent out and the detecting node gets no notifications from other nodes in the group. The shortest failure detection time is achieved when the target node fails immediately before a heartbeat is supposed to be sent out, and the detecting node gets all notifications from all other nodes in the cooperating group. For every scheduled heartbeat, the detecting node gets $n-1$ notifications and one missing heartbeat. It takes $\lceil \frac{k}{n} \rceil$ missing heartbeats to get the total bigger than or equal to the threshold k. So the minimal detection time is $T_{co} = \left(\lceil \frac{k}{n} \rceil - 1\right) \times \mathcal{T}$.

We are interested in the expected detection time. If the probability the detecting node can reach the conclusion after m ($\lceil \frac{k}{n} \rceil \le m \le k$) missing heartbeats is $g(m)$, the expected detection time will be

$$E(T_{co}) = \mathcal{T} \sum_{m=\lceil k/n \rceil}^{k} m g(m) - \frac{\mathcal{T}}{2} \tag{1}$$

To get the probability $g(m)$, we explore all possible cases that a detecting node cannot make the conclusion in $m-1$ intervals, but can in m intervals. Assume the number of notifications received in the first $m-1$ intervals is x, and the number of notifications received in the m-th interval is y, as shown in Fig. 3. Because the maximum number of notifications that can be received in one interval is $n-1$, we have $0 \le x \le (n-1)(m-1)$ and $0 \le y \le n-1$. The fact that the detecting node cannot make the conclusion in the first $m-1$ intervals implies $x + m - 1 \le k - 1$, i.e.,

$x \leq k - m$. In order for the node to make the conclusion in the m-th interval, we have $x + y + m \geq k$.

From $y \leq n - 1$ and $x + y + m \geq k$, we get $x \geq k - m - (n - 1)$. Therefore, x can take values from $max(0, k - m - (n - 1))$ to $min(k - m, (n - 1)(m - 1))$, and y can take values from $k - m - x$ to $n - 1$.

Because the target node has failed, all other nodes in a cooperating group will send a notification to the detecting node. Each notification will reach the detecting node with probability $1 - p$. Under the independent loss assumption, the number of notifications reaching the detecting node in the first $m - 1$ intervals follows the binomial distribution with parameters $((n - 1)(m - 1), 1 - p)$. The number of notifications reaching the detecting node in the m-th interval follows the binomial distribution with parameters $(n - 1, 1 - p)$. We use notation $P_{n,p}(i)$ to represent the probability of i successes in the binomial distribution with parameters (n, p). We know $P_{n,p}(i) = \begin{pmatrix} n \\ i \end{pmatrix} p^i(1 - p)^{n-i}$, where $\begin{pmatrix} n \\ i \end{pmatrix} = \frac{n!}{i!(n-i)!}$.

The probability of getting x notifications in the first $m - 1$ intervals is $P_{(n-1)(m-1),1-p}(x)$. The probability of getting y notifications in the m-th interval is $P_{n-1,1-p}(y)$. Therefore, if we let $\alpha = (n - 1)(m - 1)$, and $\beta = k - m - (n - 1)$, the probability of reaching the conclusion in the m-th interval is

$$g(m) = \sum_{x=max(0,\beta)}^{min(k-m,\alpha)} \left(P_{\alpha,1-p}(x) \sum_{y=k-m-x}^{n-1} P_{n-1,1-p}(y) \right)$$

Combining this with formula (1), we get the expected detection time of the cooperative approach.

3.2 Probability of False Positive

Non-cooperative case In the non-cooperative failure detection, a false failure detection at node x is caused by the loss of k consecutive heartbeats that the target node sends to x. Therefore, the probability of false positive is $F_{nc} = p^k$.

Cooperative case In the cooperative detection, both the consecutive loss of heartbeats sent to a monitor node v itself and the notifications from cooperating nodes contribute to the false failure detection. The probability that a cooperating node will send a notification to v while the target node is still sending heartbeats is $\theta = p(1 - p)$, which is the probability (p) that the heartbeat to that node is lost times the probability $(1 - p)$ that the notification successfully reaches v.

The probability that v concludes the target node has failed after the target node sends out m heartbeat messages is the probability that v misses all m heartbeat messages times the probability that an appropriate number of notifications reach v. Note the number of notifications reaching v in the first $m - 1$ intervals follows the binomial distribution with parameters (α, θ), and the number of notifications reaching v in the m-th interval follows the binomial distribution with parameters $(n-1, \theta)$. Following the similar reasoning as in deriving the expected time, the probability of false positive when node v reaches a conclusion that the target node has failed after m ($\lceil k/n \rceil \leq m \leq k$) heartbeat messages is

$$f_{co}(m) = p^m \sum_{x=\max(0,\beta)}^{\min(k-m,\alpha)} \left(P_{\alpha,\theta}(x) \sum_{y=k-m-x}^{n-1} P_{n-1,\theta}(y) \right) \tag{2}$$

where $\alpha = (n-1)(m-1)$, and $\beta = k - m - (n-1)$. Therefore, the total probability of false positive in the cooperative case is $F_{co} = \sum_{m=\lceil k/n \rceil}^{k} f_{co}(m)$.

3.3 Overhead

Non-cooperative case The overhead for one monitor node is one heartbeat message per $\mathcal{T}$ seconds, if the target node does not fail. Therefore, the overhead is $H_{nc} = \frac{1-q}{\mathcal{T}}$.

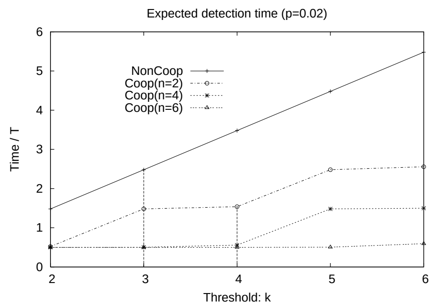
Cooperative case For the cooperative failure detection, in addition to the heartbeat message, we have notification messages transmitted between cooperating nodes. When the target node fails (with probability q), each monitor node will lead to $n-1$ notifications being transmitted. When the target node sends heartbeats normally (with probability $1-q$), each of $n-1$ cooperating nodes may miss the heartbeat message with probability p and send a notification message to the monitor node. Therefore, the overhead for the cooperative case is

$$H_{co} = \frac{q(n-1) + (1-q)\left(1 + p(n-1)\right)}{\mathcal{T}} \tag{3}$$

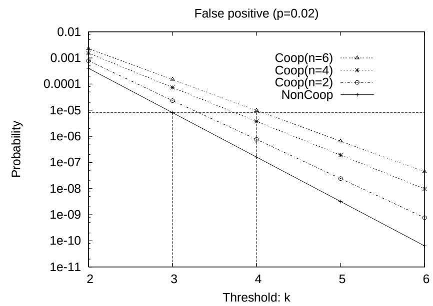
3.4 Numerical Results and Analysis

Non-cooperative vs. Cooperative To compare different approaches, we show the numerical results in Fig. 4(a)-4(c). For the cooperative detection, the number of nodes in a cooperative group (n) is either 2, 4 or 6. As the threshold k increases, the detection time increases in all schemes (Fig. 4(a)), but the non-cooperative approach increases much faster than the cooperative approach. For a given threshold k, the cooperative approach can detect the failure in significantly shorter time. For example, when $k = 4$, the cooperative approach with $n = 6$ only takes $0.5\mathcal{T}$, which is 1/7 of the time ($3.5\mathcal{T}$) by the non-cooperative approach. Fig. 4(b) and Fig. 4(c) show that the non-cooperative approach has a lower probability of false positive and lower overhead than the cooperative approach, when they use the same threshold value (k).

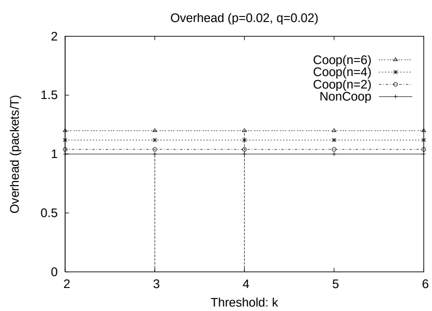
An interesting observation is that the cooperative scheme with $n = 4$ nodes in a group and the threshold $k = 4$ achieves *both* a lower probability of false positive and the shorter detection time than the non-cooperative approach with $k = 3$. A more general observation from the plot is that given a non-cooperative scheme with some $k > 1$, we can always find a cooperative scheme with appropriate k and n such that it has shorter detection time and a lower probability of false positive at the same time. In all cases, the extra overhead is always no bigger than 20%.



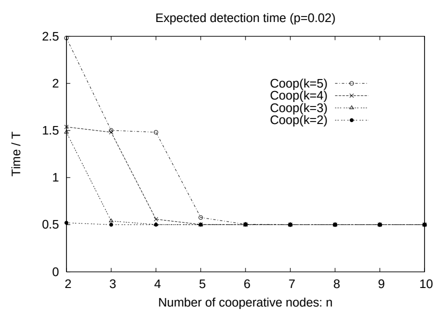
(a) The expected detection time



(b) The probability of false positive



(c) The detection overhead



(d) The effect of group size on detection time

Fig. 4. Quantitative comparison of failure detection schemes

The Effect of the Number of Monitor Nodes in a Cooperative Group In Fig 4(d), as the number of monitor nodes in a group increases from 2 to 10, the detection time decreases. Due to lack of space, we do not show the figures for the probability of false positive and the overhead. Both metrics increases as the size of the cooperative group increases. The key observation is that when the group size increases beyond threshold k, the detection time reaches the minimum value $\mathcal{T}/2$ and cannot be further decreased, while the probability of false positive and the overhead continue increasing. On one hand, this tells us that the cooperative scheme with $n \geq k$ achieves the design goal of having monitor nodes detect the failure of a target node within one heartbeat interval in most cases. On the other hand, this gives us a hint that the number of nodes in a group should not be too large. A value close to or a little bit bigger than k is enough.

An Implementation Implication When implementing the cooperative scheme, we let the parent and the children of a target node form a cooperative group, which can be much larger than k. The implication of the above observation is that the large number of members in a group does not help much to reduce the detection time further, but increases the traffic generated when a heartbeat is lost. One approach to solving the problem is to limit the number of cooperating nodes in a group to k. In this paper, we adopt a different approach, i.e., probabilistic notification. We can let a node send notifications with probability p_c ($0 \leq p_c \leq 1$) if the number of members in a group is too large. Assume the number of nodes in the group is n $(n > k)$. We can set p_c to about $\frac{k-1}{n-1}$, so that each monitor node can roughly receive $k-1$ notifications within one interval and detect the failure if there is one. Also the overall traffic is reduced. We call this method the *probabilistic notification* scheme in the performance evaluation. In contrast, the original method will be called *non-probabilistic notification* scheme.

4 Performance Evaluations

1) Simulation Setup.

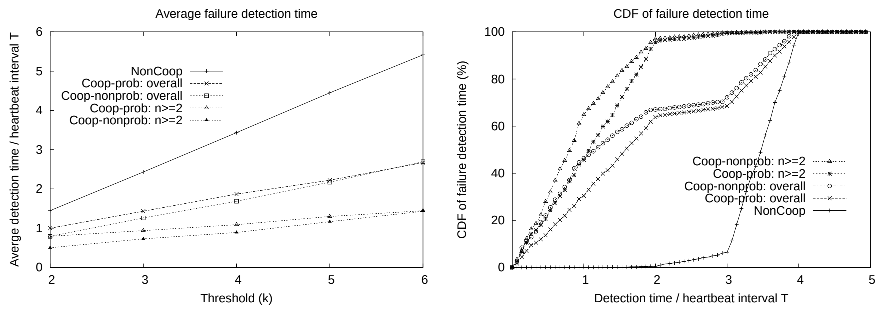
In the simulation, we use GT-ITM [7] to generate 1600 node transit-stub topologies as the underlying network. The source and the multicast members are randomly distributed in stub domains. The network-layer link latencies, represented by the edge weights in the graph, range from 1 ms to 81 ms. The application-level distance (path latency) between two end-hosts is the sum of link latencies on the shortest path between them. It ranges from 1 ms to 220 ms with the average equal to 96 ms in the generated topology. The maximum number of neighbors that a node can have in the overlay multicast tree, called node degree, is uniformly distributed in range $[2, 2 \times d - 2]$, where d is the average node degree. In the simulations, $d = 5$ if not stated otherwise.

All experiments begin with a multicast tree with 160 end-hosts. Then nodes join and leave the tree dynamically, following the Poisson process with leaving and join rate $\lambda = 0.2/second$. Each experiment lasts for two hours. We use the Gilbert model [8] to simulate the packet loss on the network-layer links. The average link loss rate is set as $p_{link} = 1\%$ if not mentioned otherwise. The end-to-end path loss probability is $p = 1 - (1 - p_{link})^l$ if the path consists of l links. The average number of links between neighbors in the generated multicast tree is about 6. In the experiments, the heartbeat interval is $\mathcal{T} = 15$ seconds.

In the following figures, labels "Coop-nonprob" and "Coop-prob" denote the cooperative failure detection using the non-probabilistic and the probabilistic notification scheme respectively. "NonCoop" represents the non-cooperative approach.

2) Failure Detection Time.

Fig. 5(a) compares the average failure detection time, measured in the number of heartbeat intervals. Fig. 5(b) depicts the cumulative distribution of the failure detection time when the threshold is $k = 4$. Curves "overall" plot the average detection time of all nodes in the multicast tree, while curves "$n \geq 2$" only consider those nodes that have $n - 1$ nodes to cooperate with in monitoring target nodes. We observe that the cooperative detection achieves much smaller average detection time, compared with the non-cooperative approach. For example, for the nodes in the cooperating groups with sizes $n \geq 2$, when $k = 4$, the average detection time is only 30% of the non-cooperative approach. Moreover, more than 55% of them detect the target node failures within just one interval and more than 95% of them detect the failures within two intervals. In contrast, using the non-cooperative approach, more than 95% of the failures are detected after three intervals. The detection time of the "overall" curves are not as



(a) Average failure detection time (b) CDF of the failure detection time

Fig. 5. Failure detection time

good as "$n \geq 2$" because the nodes that have nobody to cooperate with need longer time to detect the failures. We vary the average node degree d between 2.5 and 5 and find that about $70\% \sim 85\%$ of the nodes in the multicast tree belong to the monitoring groups with sizes no less than 2. This means that most of the tree nodes cooperate with somebody else and thus can benefit from the cooperative approach to detect the target failures faster. Also we observe that in the cooperative detection, using the probabilistic notification can detect failures almost as fast as using the non-probabilistic scheme.

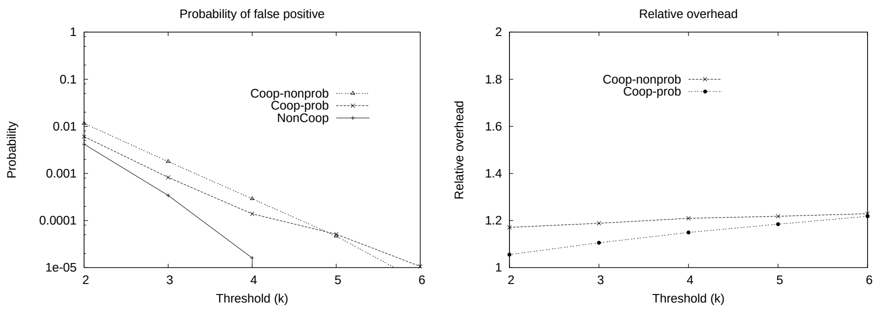
3) Probability of False Positive.

Fig. 6(a) plots the probability of false positive. Given a threshold k, the cooperative approach has more false positives than the non-cooperative approach. However, we also note that using a larger threshold k in the cooperative approach (e.g., $k = 4$) can result in the smaller probability of false positive and the smaller detection time This revalidates the observation in the formal analysis. Moreover, when the threshold k is less than the sizes of most cooperating groups, the probabilistic notification scheme can get smaller probability of false positive for the cooperative approach, compared with the non-probabilistic scheme. This is because using the probabilistic notification, when the cooperating group size n is greater than the threshold k, only about $k - 1$ instead of $n - 1$ cooperating nodes of a monitor node can send false notifications to it. A related problem is how to choose k. We may have a target probability of false positive we want to achieve. Then we can determine the value for k that satisfies the goal.

4) Overhead.

The overhead of detection approaches is measured by the number of control packets sent per second for the detection purpose. In the non-cooperative approach, the heartbeat packets are counted. For the cooperative detection, both the heartbeats and the notification packets are included in the calculation. We define the *relative overhead* of the cooperative approach as the ratio of its overhead to that of the non-cooperative approach, under the same experiment configuration.

Fig. 6(b) depicts the relative overhead of the cooperative failure detection using two different notification schemes. We observe that using the non-probabilistic notification, the cooperative approach introduces about 20% more traffic than the non-cooperative approach. This allows us to reduce the average detection time for all tree nodes by 50%



(a) Probability of false positive (b) Relative overhead

Fig. 6. Probability of false positive and overhead

and for nodes having somebody to cooperate with by roughly $70 \sim 75\%$, compared with the non-cooperative approach. Moreover, the overhead can be further reduced by using the probabilistic notification scheme. As demonstrated in the figure, its overhead is always less than or equal to that of the non-probabilistic scheme. For example, with $k = 2$, the probabilistic notification only gives rise to 5% more overhead than the non-cooperative approach, in contrast to 17% in the non-probabilistic scheme.

5 Related Work

Failure detection has been studied in distributed systems [9]. The goal is to design a scalable detection mechanism that all nodes in the distributed system can reach a consensus that a failure has occurred. In the context of overlay multicast, failure detection is usually described as a part of the tree construction procedure. In the End System Multicast [1], each node sends *refresh* (heartbeat) messages to its neighbors. If a node does not receive refreshments from a neighbor for a long period of time, a probe is sent to this neighbor. It is considered to be dead if no response is returned. For those nodes it does not receive refreshments for some time, but not long enough, it probabilistically probes them. In the Intradomain Overlays [10], each overlay node periodically sends *keepalive* (heartbeat) messages to its tree parent. Failure in receiving such a message makes the parent to probe the child. No response for the probe leads to declaration of child failure.

Both of them use the heartbeat mechanism to detect the failure. They also use probe to make sure that the target has failed, in order to reduce the probability of false positive. Notice that the final probe itself may also have a certain probability of false positive, depending on which probing method is used. While the probing can also be used in our systems, the cooperative scheme we proposed can reduce the time the monitor node detects the problem at the target node and increase the confidence of the conclusion. By choosing an appropriate threshold and forming a cooperating group, we can decrease the probability of false positive to a very low level and make the probing unnecessary.

The Resilient Overlay Network [11] also addressed the failure detection problem. It aims at detecting path failures instead of node failures in overlay networks, by using an active probing mechanism. If no response is returned after probing a node, a higher probing frequency replaces the normal probing frequency. If no response is received for a certain number of consecutive probings, the probed path is determined to be dead. The traffic generated by the probing process doubles that of heartbeat mechanism, and the chance of message loss is also doubled. Therefore, the probability of false positive is higher than one-way heartbeat based detection mechanisms. Based on the paper, the detection time of the probing mechanism is about $2\mathcal{T}$ if the probing interval is $\mathcal{T}$. In contrast, our cooperative scheme can detect the failure within one heartbeat interval if we have enough cooperative nodes.

6 Concluding Remarks

Failure detection and recovery is a very important problem in overlay multicast. The effectiveness of the detection mechanism has direct impacts on the service quality to the receivers in the session. In this paper, we proposed a cooperative approach to speed up the failure detection process. We gave a quantitative study of three important performance measures of detection mechanisms, and analyzed the fundamental tradeoff among them. We also discussed some implementation issues and evaluated the performance of the proposed scheme. While the focus of this paper is on failure detection in overlay multicast, the cooperative detection mechanism is also applicable to other overlay networks as well.

References

1. Chu, Y.H., Rao, S.G., Seshan, S., Zhang, H.: A case for end system multicast. In: Proceedings of ACM SIGMETRICS'00. (2000) Santa Clara, CA.
2. Chawathe, Y., McCanne, S., Brewer, E.A.: An architecture for Internet content distribution as an infrastructure service (2000) Available at http://www.cs.berkeley.edu/ yatin/papers/scattercast.ps.
3. Padmannabhan, V., Wang, H., Chou, P.: Resilient peer-to-peer streaming. In: Proceedings of the 11th IEEE ICNP'03. (2003)
4. Deshpande, H., Bawa, M., Garcia-Molina, H.: Streaming live media over a peer-to-peer network (2001) Technical Report CS-2001-31, CS Dept. Stanford University.
5. Yang, M., Fei, Z.: A proactive approach to reconstructing overlay multicast trees. In: Proceedings of the IEEE INFOCOM'04. (2004)
6. Jannotti, J., Gifford, D.K., Johnson, K.L., Kaashoek, M.F., James W. O'Toole, J.: Overcast: Reliable multicasting with an overlay network. In: Proceedings of the 4th OSDI'00. (2000)
7. Zegura, E.W., Calvert, K., Bhattacharjee, S.: How to model an internetwork. In: Proceedings of INFOCOM'96. (1996)
8. Yajnik, M., Moon, S., Kurose, J., Towsley, D.: Measurement and modelling of the temporal dependence in packet loss. In: Proceedings of INFOCOM'99. (1999) New York.
9. van Renesse, R., Minsky, Y., Hayden, M.: A gossip-style failure detection service. In: Proceedings of Middleware'98. (1998) 55–70 The Lake District, England.
10. Kommareddy, C., Guven, T., Bhattacharjee, B., La, R., Shayman, M.: Intradomain overlays: Architecture and applications. Technical Report UMIACS-TR 2003-70, University of Maryland, College Park (2003)
11. Anderson, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient overlay netorks. In: Proceedings of ACM SOSP'01. (2001)