Efficient Location-aware Replication Scheme for Reliable Group Communication Applications

Yuehua Wang[1,2,3], Zhong Zhou[1,2], Ling Liu[3], Wei Wu[1,2]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, China
[2]School of Computer Science and Engineering, Beihang University, China
[3]College of Computing,Georgia Institute of Technology, USA
yuehua.wang1981@gmail.com,{zz,wuwei}@vrlab.buaa.edu.cn
lingliu@cc.gatech.edu

***Abstract*—In this paper, we present an efficient replication scheme designed to provide scalable and reliable group communication services for end system nodes that are widely dispersed in the network. This scheme is highlighted with three features. First, it employs both remote and neighbor replicas and intelligently avoids the services from interruptions while keeping replication cost low. Second, by using a tunable parameter, this scheme enables the nodes to adaptively make the most promising replica placement decision according to their specific network statuses. Third, a novel recovery technique is presented to minimize service recovery delay. The experimental results show that the proposed scheme is highly efficient in improving service reliability compared with existing neighbor-based replication scheme, random replication scheme and simple alternative schemes. The service recovery delay is greatly reduced by combining with the proposed recovery scheme.**

Keywords-location; replication; reliable; flexible; group communication;

I. INTRODUCTION

Overlay network has emerged as a promising paradigm to support group communication applications such as Video Conference, IPTV [1], Distributed Interactive Simulation, Local-based Notification [2] and Really Simple Syndication, which are characterized by exchanging contents among multiple end system nodes. It provides high scalability and availability for applications by harnessing loosely coupled and inherently unreliable end system nodes at the edge of the Internet.

However, one of the main challenges in overlay network currently facing is how to build reliable group communication services based on that. Due to the unreliable nature of nodes, it is necessary to find an effective way to deal with unpredictable node departures and nondeterministic network partitions so that it can provide stable and reliable support for the applications of group communication. One approach to address this problem is to create and maintain a set of node copies to mask failures from applications in the presence of node failures and network partitions. While data replication has been well-studied in the context of both unstructured and structured networks, they are often costly or perform inefficiently in dealing with the failures when facing a high churn rate.

In this paper, we develop an efficient replication scheme that is dedicatedly designed to provide scalable and reliable group communication applications such as Multi-party Online Game, News Feed Propagation, Instant Messaging, Popular Internet Radio Service, and so on.

This scheme has three features. First, it uses both remote and neighbor replicas and intelligently avoids the services from interruptions while keeping replication cost low. Second, this scheme provides the nodes with a large degree of flexibility in deciding replica placement. By using a tunable parameter, each node can adaptively make its replica placement decision according to its specific network statuses. Third, it provides a novel technique to minimize service recovery delay. This technique is derived from independence of service recovery and connectivity recovery.

To investigate the performance of the proposed schemes, they are applied to GeoCast [3], a geographically distributed overlay system. The experimental results show that compared to existing replication schemes [4–7] our scheme is highly efficient in improving the service reliability and the service interruption time is greatly reduced by combining with the proposed recovery scheme.

The rest of this paper is organized as follows. In Section 2, existing studies related to our work are discussed. Section 3 gives an overview of GeoCast. The design of our replica placement algorithm is presented and the setting of replication parameter is discussed in Section 4. In Section 5, extensive simulations and performance evaluations are presented. Finally, Section 6 concludes the paper and discusses future research directions.

II. RELATED WORK

A large number of research proposals [8–12] have been well designed to address the reliable content delivery problem by using overlay. In particular, a proactive fault-tolerant multicast routing protocol [11] is proposed to use backup paths to deal with link or path failures. Fei *et al.* [12] devised a dual-tree scheme to improve the resilience of multicast

 ISBN:978-1-61208-113-7

dissemination by constructing a secondary tree in addition to the primary tree normally used. Once failure occurs, it activates the path in the secondary tree.

PRM [8] is a multicast data recovery scheme proposed to achieve high delivery ratio in the presence of node/link failures. In PRM, each node forwards data to both its child nodes and a constant number of random nodes during content dissemination. Once failure occurs, those data duplications can be used to feed the orphans isolated by network partitions. However, the volume of extra data duplications and traffic might be significant for the applications like on-demand news propagation and multi-party online game.

Overcast [10] is motivated to provide service for bandwidth-intensive content dissemination. In overcast, nodes perform content caching and failure recovery operations to alleviate the influence of node or link failures. Starting with the root, some number of nodes is configured linearly, that is, each has only one child. The drawback of this technique is the increased latency of content distribution as the data has to traverse all those extra nodes before reaching the rest of overcast nodes.

Scattercast [9] proposed a infrastructure-supported architecture for scalable and reliable multicast support. In Scattercast, the multicast problems are partitioned into a set of smaller and simpler sub-problems that can be easily addressed with local knowledge of nodes in the same region. However, such divide-and-conquer approach might be expensive, especially in a network with high link and node failure rate. A large number of additional messages may be generated for the service recovery and new region formation when failures happen.

Our work differs from all other schemes by providing a proactive component used to augment the performance of application protocols. Instead of changing the infrastructures of the current applications as well systems, it is employed to provide reliable services to them while in a scalable fashion. In this paper, this is achieved by utilizing both remote replica nodes and neighbor replica nodes. We will present the scheme in details in the following.

III. Background

In this section, we describe architecture of GeoCast and introduce multicast mechanism on which the algorithm of this paper is based.

A. Architecture

GeoCast is a structured geographical proximity-aware overlay network, consisting of nodes equipped with GeoCast middle-ware. In GeoCast, nodes are equivalent in functionalities, each of which can perform: message publishing, message subscribing, message routing or all of them. Each node keeps a set of information about other nodes in the network in its $peernodelist$. Such list contains two kinds of nodes: immediate neighbor node and shortcut node. Similar to CAN, two nodes are considered to be immediate neighbors when their intersection is a line segment. The term *shortcut node* refers to the node which is *old neighbor node* for a given node. As nodes arrive or depart, neighbor nodes may become shortcut nodes when they are not adjacent to the given node. Instead of removing old neighbors from the lists, GeoCast keeps those nodes in $peernodelist$s and uses them to speed up the procedure of message delivery.

B. Multicast Service

In GeoCast, multicast service is introduced as one component for supporting group communication between nodes. It has two benefits. First, it releases high link stress caused by message transitions among nodes in group communication session and consequently improves network resource utilization. Second, it reduces the delay of the message delivery from the publishers to the subscribers. Instead of transversing the data from the publishers, the subscribers often can get it from their parent nodes within a short delay. For each session, there exists a spanning tree that is an acyclic overlay connecting all the participants of the session. It is used by the publisher node for content dissemination. Detailed algorithms and examples of the multicast service establishment and maintenance process with node arrivals and departures are presented in [3].

IV. Location-aware Replication Scheme

To provide reliable group communication services, a location-aware replication scheme is designed. It takes advantages of both the neighbor node and the remote node to improve the reliability of the services offered by the unreliable nodes while keeping the overhead low. The idea is derived from that the random replica nodes can always provide high reliable services for the nodes in the presence of failures mentioned in [6, 7]. However, it comes at high cost of replica detection and maintenance. To avoid that, in our scheme, only the nodes in the $peernodelist$s are considered in replica placement. The benefits are two-fold. First, it reduces the replication cost by employing the nodes in vicinity. Second, it alleviates the influence of network partitions on the service quality by deploying the data copies on shortcut nodes.

In this section, we first introduce patterns of failures and then present details of our replication scheme.

A. Failure Pattern

We consider a network consisting of a number of unreliable nodes. At any point in time, nodes can become unavailable for various reasons such as node departure, computer crash, improper program termination and traffic congestion. Note that similar operations will be performed to deal with node departure and failure in GeoCast, we use the term failure to mean either departure or improper failure.

 ISBN:978-1-61208-113-7

Based on the distribution of node failures, two failure patterns are identified: distributed failure and centralized failure (i.e., network partition). In the pattern of distributed failure, node failures are scattered over network. There always exist available end nodes around single failed node that can detect and repair its failure. In the centralized failure pattern, some nodes of the network appear to be unreachable from certain nodes but not others. Once it happens, the entire network might be partitioned into multiple, isolated overlay network parts. In this case, if multicast services continue to operate on this disconnected network, it might be interrupted and have nothing to do but wait for self-recovering from network partition. To provide the best quality of service, it is necessary for replication schemes of applications to deal with the failures of different patterns while keeping cost low. Inspired by this, our scheme is proposed and we will present the details of our scheme in the next section.

B. Replication Scheme

This section focuses on the two main components of the replication scheme design. First, the parameter used to leverage the benefits of neighbor nodes and shortcut nodes is defined. Second, we specify how the replication scheme is performed in a network of nodes with heterogeneous capacities.

1) Parameter Definition: It is desirable for nodes to maintain both the neighbor replica node and the shortcut replica node to improve replication efficiency in terms of both reliability and scalability. In our design, a tunable parameter α is introduced to adjust the importance of those nodes. The parameter value ranges from 0 to 1. The smaller it is, the less important the shortcut nodes are, the more likely it leads to a higher replication cost than that of scheme employing nodes residing in adjacent and vice-versa.

Interestingly, looking for the best values for α is essentially finding the best tradeoff between reliability and scalability. Consider a node p who has a replica placement task with replication degree r. When increasing α, node p reduces the number of neighbor replica nodes which causes the probability of service remaining available in the presence of network partitions to increase. However, larger α also leads to more shortcut replica nodes which in turn increases replica creation and maintenance cost. Based on those facts, we find the α value can not be too high or too small for the purpose of minimizing the replication cost and optimizing the reliability of service. Given an average of $O(2d)$ immediate neighbors is kept on each node in GeoCast [3], we have: for any node p,

$$\alpha = \begin{cases} 0.5, & r \leq 2 \\ \frac{1}{r}, & 2 < r \leq \tau \\ \frac{r-\tau}{r}, & r > \tau \end{cases} \quad (1)$$

where τ is the number of neighbor nodes in the $peernodelist$ of node p. The motive behind that is to relatively reduce the importance of shortcut replica so as to minimize the link stress imposed by placing and maintaining shortcut replicas. Typically, for a given τ, as increasing r, the number of neighbor replica nodes also increases. After r reaches $\tau + 1$, neighbor replica number reaches its upper bound τ, that is, no neighbor node is left in the $peernodelist$ that can be selected as the replica node. In this case, more shortcut nodes tend to be involved in the replica placement.

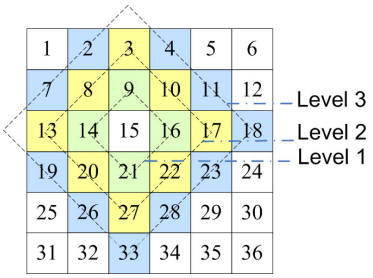


Figure 1. Relationship among nodes

2) Replica Placement: Each node in the multicast sessions deploys r nodes with available capacity in the network as replica nodes by performing the following operations.

Neighbor Selection The objective of this operation is to create neighbor replica nodes for nodes. Since the nodes' $peernodelist$ may or may not have "enough" available neighbor nodes, we introduce a new notation of *k-hop neighbor* B^k and define it as:

$$B^k_{E_i} = \begin{cases} \{immediate\ neighbors\ of\ E_i\} & k = 1 \\ \cup_{j=1}^{m_1} B_{B^1_{E_i}(j)} - B^1_{E_i} & k = 2 \\ \cdots & \cdots \\ \cup_{j=1}^{m_{l-1}} B_{B^{l-1}_{E_i}(j)} - B^{l-1}_{E_i} & k = l \end{cases} \quad (2)$$

where E_i denotes a node in the network and m_{l-1} is the number of nodes in $B^{l-1}_{E_i}$. When k=1, all immediate neighbors of node E_i are 1-hop neighbors of E_i. As mentioned earlier, two nodes are considered as immediate neighbors when the intersection of their regions is a line segment. If k=2, $B^2_{E_i}$ consists of the nodes that are the immediate neighbors of $B^1_{E_i}$'s entry nodes. For consistency, it requires the nodes in $B^1_{E_i}$ are not in $B^2_{E_i}$.

Fig. 1 gives a simple example of network to illustrate the relationship between nodes. In Fig. 1, node E_9, E_{16}, E_{21} and E_{14} are the immediate neighbors of node E_{15}. Nodes E_2, E_{10}, E_{17}, E_{22}, E_{27}, E_{20}, E_{13} and E_8 are immediate neighbors of $S^1_{E_{15}}$ that are included in $B^2_{E_{15}}$.

Neighbor selection starts with $B^1_{E_i}$. Node E_i first looks through its B^1 to see if there are $(1-\alpha)r$ nodes with available capacities. If so, node E_i adds them into its *replicalist* and replicats its multicast information on them, where *replicalist* is a list created for reordering the information of nodes which are selected as the replica nodes. Then this selection procedure terminates. Otherwise, E_i first adds all capable nodes at the first level into the *replicalist* and increases k by 1. Then the neighbor selection is performed at the second level. This procedure is executed repeatedly until either $(1-\alpha)r$ neighbor replica nodes are fetched or

 ISBN:978-1-61208-113-7

k reaches its maximum K. The parameter K is a system constant which is configured by default. The purpose of introducing K is to limit the cost of replica selection within a certain level. As suggested by [13], we set it to 2 in order to limit the replication searching cost within $O(4nd^2-2nd)$, where n is the network size.

Shortcut Selection The shortcut selection is performed in a similar manner as the first phrase. It starts with the set $S^1_{E_i} = peernodelist_{E_i} - B^1$, where $peernodelist_{E_i}$ is the*peernodelist* of node E_i. Then this procedure is continuously executed at the next level as there is no enough capable shortcut nodes. Similar to the definition $B^k_{E_i}$, $S^k_{E_i}$ is defined as $S^k_{E_i} = \cup_{j=1}^{m_{k-1}} B_{S^{k-1}_{E_i}(j)} - S^{k-1}_{E_i}$. This selection procedure is terminated when either αr shortcut replica nodes are fetched or k reaches its maximum K. Fig. 2 gives an example to illustrate the shortcut selection with setting of r=8 and α= 0.25. Node H first looks through $S^1_H = \{E_1, E_2, E_4, E_{14}, E_{18}\}$ and checks if there exists 2 shortcut nodes with available capacities. If so, no selection is necessary as enough replica nodes are detected. However, due to the heavy loads of nodes E_1, E_2, E_{14}, E_{18}, only E_4 is selected at the first level. To meet the requirement of $\alpha r = 2$, node H extends its search region and E_3 resided at the second level is then included in the *replicalist*. Similarly, capable nodes E_6, E_7, E_9, E_{11}, E_{12}, E_{16} that are marked with green circle are selected and then added into the *replicalist* after neighbor selection.

C. Replica Management and Failure Recovery

To avoid introducing additional overheads, the maintenance information of replicas are appended to the existing heartbeat message in our scheme. Every T seconds, heartbeat mechanism is performed, as well as replica maintenance. The parameter T is a constant that refers to the parameterized heartbeat period. We do not eagerly notify replicas to update theirs backup information if the relationship of multicast tree has been changed. It would be not done until heartbeat message between them is issued. In our scheme, multicast messages also serve as implicit heartbeat messages avoiding the need for explicit heartbeat messages. Long-time absence of heartbeat indicates that the node is gone and its corresponding region becomes an *island*, and thus boots the recovery process mentioned in [14].

Once a replica failure is detected, the update of $replicalist$ is triggered and one new replica node is selected by using the replica placement algorithm. In the procedure of replicalist updating, only the failed node is removed and replaced with a node in the set $B^k \cup S^k$. The new replica may or may not be a neighbor node of the host node, which relates to the character of the failed replica node and status of the nodes in the set $B^k \cup S^k$.

A node's failure triggers the recovery process. In terms of nodes functionalities in the network, there are two distinct recovery actions:

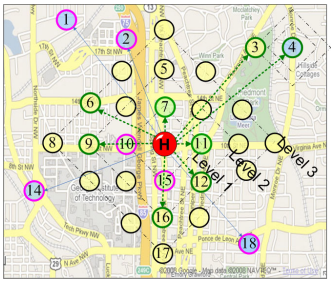


Figure 2. Shortcut Selection

- **Path Recovery** The goal of this action is to restore the services from interruptions when the failures of nodes in the multicast session happen. Rather than rebuilding the routing paths from the leaf nodes to the root nodes, the multicast information stored on the replica nodes are used for service recovery. It is done by replacing the failed node with one replica node and reconnecting with the parent and child nodes of the failed node. In our scheme, the node with high available capacity owns higher preference to be selected as the alternative node.
- **Region Recovery** This action is to search an alive node to take over the *island*. Rather than searching for leaf node of the *island*, our scheme tries to merge it locally with the help of failed node's neighbors. This process is stated by broadcasting merge request to all failed node's neighbors. Those nodes then check their regions to see if two of them can be merged. If more than one pair of siblings is around *island*, the pair with highest capacity is elected to handle such failure.

D. Replica selection policy

It is worth noting that there is wide variability in the available capacity of end nodes in the overlay network. Each node in the network has a fixed capacity. For different applications, it may refer to different factors such as processing power for responds, available storage space for data sharing services, or the available uploading network bandwidth for multimedia streaming applications. In our work, we use it to refer to the storage capacity of the nodes, denoted by c. It ranges from 0 to 1. c_i represents the maximum amount of storage that node E_i is willing to devote for serving other nodes. If $c_{E_i} = 1$, node E_i is overloaded; otherwise the node is under loaded. We use light and heavy to informally refer to nodes with low and high *utilization*, respectively. Periodically, each node calculates its load c and piggybacks it in heartbeat messages to update its status.

E. Analysis

In this work, one multicast service will be interrupted on a node only when all its replica holders fail in a short time interval. If it happens, a large number of re-subscription requests will be issued and routed among the nodes. To avoid this, it is a necessity for the replication schemes to employ enough replica nodes to deal with the failures of different

 ISBN:978-1-61208-113-7

pattern. Therefore, we next study the service reliability of a node archived by replication under an example setting to provide some insights into the performance of our scheme.

Consider a service s offered by node p. Let $R = \{e_1, e_2, ..., e_r\}$ be the $replicalist$ of p. Let P_i be the reliability of node e_i at time t_c. Then service reliability R_s can be calculated as:

$$R_s = 1 - \prod_{r}^{i=1}(1 - P_i) \quad (3)$$

$$P_i = P_i\{t > t_c | t > t_{s_i}\} = \frac{P_i\{t > t_c\}}{P_i\{t > t_{s_i}\}} = \frac{1 - P_i\{t \leq c\}}{1 - P_i\{t \leq s_i\}} \quad (4)$$

where R_s is the probability of at least one node in set R remaining in the network in next time slot. t_c and t_{s_i} refer to current time and arrival time when node e_i joins in the network. We use a Pareto distribution with shape parameter of 1.1 mentioned in [15] to estimate the distribution of node life time, represent by $F(x) = 1 - (1 + x/0.05)^{-1.1}$. As reported in [15], the average lifetime of peers remaining in the network is 0.5 hours. We have $R_s = 0.97$ with setting of r=4 and $Min\{P_i\} = 0.5$. It is clearly seen that the service reliability is greatly improved by the use of small number of replica nodes. This will also be confirmed by our experimental results in Section 5.

Our algorithm makes two effects to ensure scalability of the applications. One is to reduce the replica creation cost by intelligently using the information of the other nodes stored on the nodes. Rather than broadcasting its query request to number of nodes in the network, each node in the proposed scheme first inquires the nodes in its *peernodelist* for replica placement, which can improve the resource utilization. The other is to avoid high cost caused by remote-replica maintenance in message delay. In the procedure of replica placement, the nodes with available capacities residing in vicinity have high priority to be selected as the replica nodes. It enables either data updating or replicas' communication to be completed within a short delay.

V. Performance Evaluation

In this section, we conduct simulation experiments to evaluate the performance of the proposed replication scheme (called NR) and compare it against three schemes, the neighbor-based replication scheme (Neighbor) [4, 5], the random replication scheme (Random) [6, 7], and the hybrid replication scheme (Neighbor&Random) that is simply an extension of Neighbor and Random, which has the advantage of replication cost and reliability. The major difference in those schemes is in the replica creation.

A. Experimental Environment

We use Transit-Stub graph model from the GT-ITM topology generator to generate network topologies for our simulation. All Experiments in this paper run on 10 topologies with 8080 routers. Each topology consists of 8080 nodes with heterogeneous capacities. Similar to [16], the nodes in the network are assigned with various resource capacities: 5% nodes have 1000 units of capacity, 15% nodes possess 100 units of capacity, 30% nodes have 10 units of capacity, and the rest of nodes has 1 unit of capacity. The processing capacity of node is proportional to its resource capability. The more resource it has, the more powerful it is. Each unit of resource capacity allows nodes to maintain 10 files in theirs local memory.

Given that there is no linear relationship between the nodes' location and their associated link latency for delivering the message in IP network, we assign link latencies by following a uniform distribution on different ranges according to their types: [50ms, 80ms] for intra-transit domain links, [10ms, 20ms] for transit-stub links, and [1ms, 5ms] for intra-stub links. It allows messages transited between two nodes in the same domain have low network transmission delay. Nodes are randomly attached to the stub domain routers and organized into the GeoCast overlay network.

In each simulation, the multicast groups are built first. Both publisher nodes and subscriber nodes sequentially participate in the groups, following independent and identical uniform distribution. To simplify, we do not consider the case that the subscribers in the same group have different interests in publisher's content since it can be handled in a similar manner.

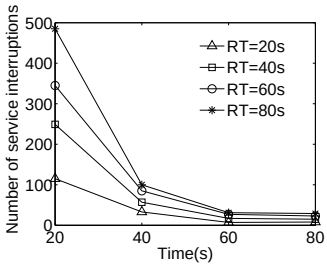
We run each trial of simulation for 60T simulated seconds, where T with setting of 120 is the parameterized heartbeat period. In our figures, each data point represents the average of measurements over 50 trials (5 trials in each topology with same setting in such a way the inaccuracy incurred by stochastic selection is minimized).

B. Results

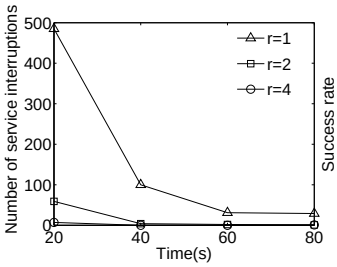
In this section, we investigate three main subjects using sets of experiments. We first evaluate how the efficiency of replication scheme in the presence of node failures. Then we study the effect of replication schemes on the system performance in terms of replication overhead. Third, the performance of the proposed recovery scheme is investigated.

1) Reliability: Similar to [17], we generate a sequence of node failures to study the effect of replication scheme on the service quality in terms of reliability. By following independent and identical exponentially distribution, the failure time of sequence nodes that are randomly chosen is assigned. As mentioned earlier, both the nodes and their replica holders may fail in a short time interval. If it happens, the multicast services of the nodes are interrupted and a large amount of re-subscription requests may be issued and transited on network for service recovery. To avoid this, high failure resiliency is desired in the current applications. As a matter of fact, it mainly depends on two factors, turnover

 ISBN:978-1-61208-113-7

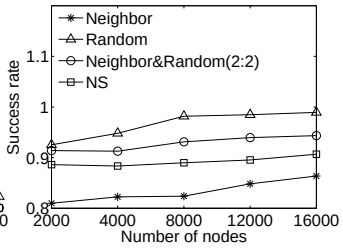


(a) r=1(FP=0.6)

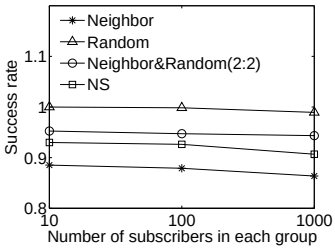


(b) r(FP=0.6 RT=80s)

Figure 3. Service interruption during runtime

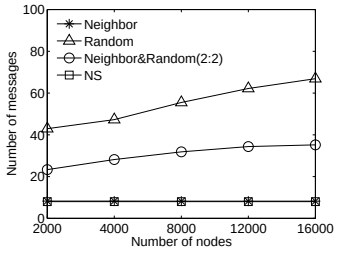


(c) different system size

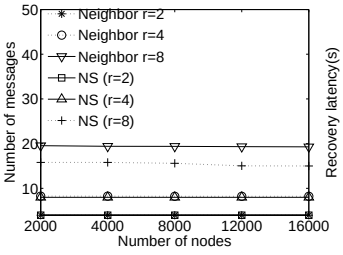


(d) different group size

Figure 4. Success rate(FP=0.6 RT=80s)



(a) different schemes



(b) different r

Figure 5. Overhead of replica creation

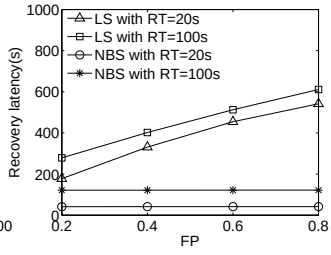


Figure 6. Effect of FP on region recovery latency

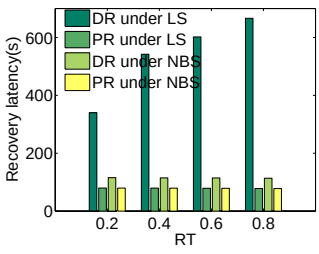


Figure 7. Effect of RT on recovery latency(FP=0.6)

time RT and replication degree r. RT refers to the time required for node initialization right after it is selected as replacement node to take over the *island* previously owned by the failed node. r refers to the number of replica nodes that are required to be placed by each node in the network.

To study the effect of those factors on the service interruptions, the following experiments are simulated in a set of 16000-node systems with FP=0.6. The value 16000 is the maximum number that we are able to simulate under constrains of the computer resources. Failure probability FP represents the faction of nodes failed at the run time. For instance, if FP is set to 60%, 9600 nodes will leave from the system at runtime. From results in Fig. 5, we observe that as time goes by, the number of service interruptions significantly drops. This is due to the fact that majority of service interruptions happen at the beginning stage of simulation. Meanwhile, it is important to notice that the metric become smaller when r is larger. When it increases to 4, the number of service interruptions is steadily below 10. Thus, we set r to 4 in the following discussion.

Fig. 4 shows the success rate as a function of system size. Success rate represents the ratio of the number of recoverable failures to the number of failures happened at runtime. We use the term *recoverable failure* to mean the service of a failed node that can be restored by one of its replica node after the failure happens.

For this and following plots wherein we vary the number of nodes N in the system, we use $N \in \{2000, 4000, 8000, 12000, 16000\}$. In Fig. 4, it is clearly shown that Random achieves a better performance than the other schemes. This is essentially because in Random replication, the replica nodes are likely to have higher probability to remain alive even facing high churn rates, which further confirms the conclusion mentioned in Section 4. Similar to Random, both Neighbor&Random and NS yield higher success rate than Neighbor. This can be explained by the fact that the nodes in Neighbor scheme are likely to replicate the information on their neighbor nodes. If the centralized failures happen, both the nodes and their replicas tend to fail simultaneously and thus it is hard to find one living replica node to restore the interrupted services.

2) Replica Creation Overhead: Fig. 5(a) shows the replication overhead as a function of system size. It is observed that both Neighbor and NS replication can successfully replicate the information on replicas with lower overheads than the others. This is due to the fact that the majority of replicas are discovered within 2 levels. Fig. 5(b) shows how the number of replicas affects the overheads of the replication schemes. As r increases, the overhead also increases. This is because more messages are issued from the nodes for replica selection when r is larger, which may lead to a poor application performance in scalability. To avoid that, NS enables the nodes to adaptively tune their parameters and deploy more shortcut replica nodes. The benefits are two folds. First, it reduces the replica searching cost. Second, it enhances the reliability of services in some sense. It indicts the efficiency of NS replication in overhead.

3) Recovery Latency: Recovery latency is the time interval between the time when a node detects a failure event and the time when new responsible node offers its services as the owner of *island*. To evaluate the performance of different recovery schemes in a dynamic environment, we vary failure probability from 20% to 80 % in 16,000-node network.

Fig. 6 shows the recovery latency as a function of fail-

 ISBN:978-1-61208-113-7

TABLE I
SUCCESS RATIO COMPARISON

	Hop	FP			
		0.2	0.4	0.6	0.8
NBS	1	86.70%	85.80%	85.70%	85.57%
	2	100%	100%	100%	99.99%
	3	100%	100%	100%	100%
	4	100%	100%	100%	100%
	Type	FP			
		0.2	0.4	0.6	0.8
LS	RF	60.75%	59.90%	59.60%	58.92%
	URF	39.25%	40.10%	40.40%	41.08%

ure probability FP. We observe that the proposed scheme (called NBS) steadily achieves better performance than leaf node scheme (LS) [14]. As FP increases, the performance improvement of our techniques is even more pronounced. It is obvious that LS scheme takes long time to recover the interrupted service than our scheme in terms of hop-count. This is due to the fact that some recovery are delayed by searching the alternative node [14], which is confirmed by the results in Table I. FR refers to the recoverable failure and UFR refers to the unrecoverable failure. In Table I, we can see that NR performs better than its competitor. All node failures can be recovered within 3 hops in all cases. For LS, when we set FP=0.8, 41.08% of the failures is unrecoverable at runtime. This is because that the failures can not be recovered until the alternative nodes are fetched in LS. In this case, the recovery latency increases exponentially, as shown in Fig. 6. The maximum latency of LS is around 60 times as many as that of NBS when FP = 0.8.

As a matter of fact, for both approaches, the recovery latency increases as increasing RT as shown in Fig.7. We observe that the scheme NBS beats its competitor in all cases and it is much more steady than LF in service recovery. Since both path recovery and region recovery conduct simultaneously, the recovery latency of NBS can be further minimized and is equal to $Max\{L_{RR}, L_{PR}\}$, where L_{RR} and L_{PR} refer to the latency of the region recovery and path recovery, respectively.

VI. CONCLUSION AND FUTURE WORK

We have presented a dynamic passive replication scheme. It is an extension of our previous work [3, 18]. In this paper, the proposed scheme is used to provide reliable group communication services for nodes who are unreliable. Through the simulations, the effectiveness of our scheme in different scenarios is demonstrated. The experimental results show that in the presence of node failures, the proposed scheme can efficiently recover services from interruptions within short recovery delay, which results in a better performance, compared to existing replication schemes for large scale group communication applications. For the future work, we would like to explore it in working systems and develop enhancement to make the proposed schemes more efficient in group communication applications.

VII. ACKNOWLEDGMENT

The first author conducted this work as a visiting PhD student at Georgia Institute of Technology. This work is partially supported by NSF NetSE, NSF CyberTrust, IBM SUR, Intel Research Council, 2008 China Next Generation Internet Application Demonstration sub-Project (No.CNGI2008-123) and Fundamental Research Funds for the Central Universities of China. The first author also thanks China Scholarship Council for supporting the visiting.

REFERENCES

[1] K. Kerpez, D. Waring, G. Lapiotis, J. Lyles, and R. Vaidyanathan, "IPTV Service Assurance," *IEEE communications magazine*, vol. 44, no. 9, p. 166, 2006.
[2] P. Persson, F. Espinoza, P. Fagerberg, A. Sandin, and R. Cöster, "GeoNotes: A Location-based Information System for Public Spaces," *Designing Information Spaces: The Social Navigation Approach*, pp. 151–173, 2002.
[3] Y. Wang, L. liu, C. Pu and G. Zhang, "GeoCast: An Efficient Overlay System for Multicast Application," *Tech. Rep., GIT-CERCS-09-14, Georgia Tech*, 2009.
[4] J. Zhang, L. Liu, C. Pu, and M. Ammar, "Reliable peer-to-peer end system multicasting through replication," in *P2P'04*.
[5] T. Chang and M. Ahamad, "Improving service performance through object replication in middleware: a peer-to-peer approach," in *P2P'05*, pp. 245–252.
[6] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the 16th international conference on Supercomputing*, 2002, pp. 84–95.
[7] Y. Liu, X. Liu, L. Xiao, L. Ni, and X. Zhang, "Location-aware topology matching in P2P systems," *INFOCOM 2004*, pp. 2220–2230.
[8] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient multicast using overlays," *ACM SIGMETRICS Performance Evaluation Review*, vol. 31, no. 1, pp. 102–113, 2003.
[9] Y. Chawathe, "Scattercast: an adaptable broadcast distribution framework," *Multimedia Systems*, vol. 9, no. 1, pp. 104–118, 2003.
[10] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, *et al.*, "Overcast: Reliable multicasting with an overlay network," in *Proc. Usenix Fourth Symp. Operating System Design and Implementation*, 2000.
[11] W. Jia, W. Zhao, D. Xuan, and G. Xu, "An efficient fault-tolerant multicast routing protocol withcore-based tree techniques," *IEEE Transactions on Parallel and Distributed Systems*, vol. 10, no. 10, pp. 984–1000, 1999.
[12] A. Fei, J. Cui, M. Gerla, and D. Cavendish, "A dual-tree scheme for fault-tolerant multicast," in *Proceedings of IEEE ICC*, 2001.
[13] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti, "A local search mechanism for peer-to-peer networks," in *Proceedings of the eleventh international conference on Information and knowledge management*, 2002, pp. 307–314.
[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *SIGCOMM'01*, vol. 31, no. 4, pp. 161–172.
[15] X. Wang, Z. Yao, and D. Loguinov, "Residual-Based Estimation of Peer and Link Lifetimes in P2P Networks," vol. 17, no. 3, 2009, pp. 726–739.
[16] J. Zhang, L. Liu, L. Ramaswamy, and C. Pu, "PeerCast: Churn-resilient end system multicast on heterogeneous overlay networks," *Journal of Network and Computer Applications*, vol. 31, no. 4, pp. 821–850, 2008.
[17] S. Saroiu, P. Gummadi, S. Gribble, *et al.*, "A measurement study of peer-to-peer file sharing systems," in *Proceedings of Multimedia Computing and Networking*, vol. 2002, p. 152.
[18] G. Zhang, L. Liu, S. Seshadri, B. Bamba, and Y. Wang, "Scalable and Reliable Location Services Through Decentralized Replication," in *International Conference on Web Services 2009*, pp. 632–638.

 ISBN:978-1-61208-113-7