Topological Similarity-based Scheme for Large-scale Group Communication Services

Yuehua Wang[1,2], Zhong Zhou[1,2], Ling Liu[3], Liang Cheng[4], Wei Wu[1,2]
[1]State Key Lab 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
[4]Department of Computer Science and Engineering, Lehigh University, USA
Email:yuehua.research@gmail.com, zz@vrlab.buaa.edu.cn, lingliu@cc.gatech.edu,
cheng@cse.lehigh.edu, wuwei@vrlab.buaa.edu.cn

***Abstract*—Group communication is essential for multi-user applications. However, due to unpredictable node departures and non-deterministic network partitions, providing reliable and scalable group communication services is challenging when the applications are utilized by the users with heterogeneous capacities on a large scale. To address this challenge, we propose a novel replication scheme to achieve high reliability and low-cost scalability in group communication with following three features. First, it introduces a new concept of replication based on topological similarity, which empowers each node with an ability of measuring similarity between the nodes in topology. By eliminating the topological similarity between the replicas, it intelligently mitigates service interruptions caused by node failures and network partitions. Second, instead of specifying the number of replicas, it provides a technique for nodes to dynamically adapt the replication placement schemes by exploiting functionality importance of the nodes in the group-communication session. It eliminates the bottleneck problem and improves the network resource utilization. Third, the scheme is self-converging and it can stabilize within a few adaptations even facing a high churn rate. Extensive simulations show that it yields significant improvements in reduction of replication overhead and service interruption when comparing to existing approaches.**

***Keywords*- Topological similarity, Importance, Reliability, Scalability, Replication, Group communication.**

I. INTRODUCTION

The explosive growth of network applications and the increasing popularity of handhold devices in recent years have made reliability and scalability important and challenging to achieve large-scale communication. The applications include online game, real-time conference, large-scale distributed interactive simulation, instant messaging and RSS services, which are characterized by exchanging information contents among multiple unreliable participants with heterogeneous capacities. In those applications, a large number of participants who are often geographically dispersed access the services and require reliable support by applications even in a network of unpredictable node departures and non-deterministic network partitions.

However, it is difficult to design a new network infrastructure or add a new service to the network layer to satisfy the requirements of the participants given the existing infrastructures of systems deployed by the applications. One intuitive approach to address such issue is to do replication, where data/file copies are created and placed on other participants in the network for high reliability and scalability. This approach provides a proactive component that can be used to augment the performance of the applications.

In the past decades, a fair amount of research [1–8] with replication scheme has been developed. Generally, there are three categories: ID-based replication [1, 2, 6], neighbor locality based replication [7, 8] and path-based replication [3–5]. These approaches either have mainly focused on achieving query efficiency without considering network resource utilization, or ignore the influence of network partitions on the system performance. In fact, the occurrence of network partitioning could make the problem of replication even more severe and consequently deteriorate the system performance because of massive volumes of messages that are generated for replica maintenance and service recovery.

Fig. 1 depicts three different scenarios that are group communication sessions obtained in the presence of failures in different patterns. In each scenario, there are 11 nodes that are divided into two classes: content publisher nodes (i.e., darkened nodes), being responsible for collecting and disseminating data to content subscriber nodes (i.e., shaded nodes) that are located at the edge of the network, and content subscriber nodes, listening to their parent nodes and receiving data from them as the data arrives.

To minimize the replication cost and improve reliability of group communication services provided by nodes, one approach can choose close nodes and replica nodes' content on them. In Fig. 1(a), nodes 7 and 9 are two replicas deployed by node 8 with this approach [3–5, 7, 8]. This approach brings two unique benefits. First, it enables the node in the network with the ability of reacting quickly to the changes of network. Second, it reduces the creation and maintenance cost of replication. However, such an approach does not work well when a network partition occurs as shown in Fig. 1(b). In Fig. 1(b), a circle region of three nodes 7, 8 and 9 may appear to be unreachable from the other nodes. As a result, nodes 2, 5, and 11 are isolated from the session and have no data received from the content publisher node. To address

978-1-4673-1544-9/12/$31.00 ©2012 IEEE

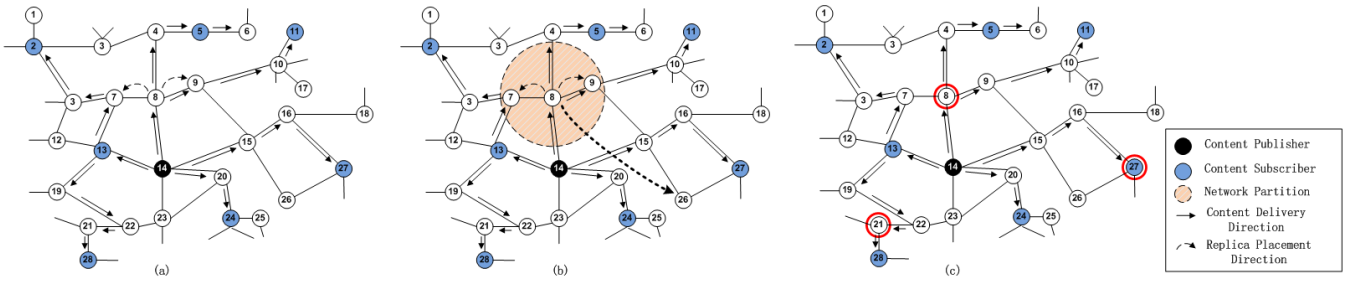


Fig. 1. A motivating example

that, one may suggest placing the replicas in a hybrid manner such that node 2 could also employ a remote node 26 as a replica node to improve the efficiency of replication scheme. It, however, subjects to a difficulty of determining the locations and the number of remote nodes.

We also argue that the data/file replication often leads to inefficient network resource utilization due to a large volume of data/file copy propagation. Given accessing to the most popular ones is frequently skewed, the replication approaches, such as [3], [4] and [5], may exhaust the capacity of those nodes and decrease the network resource utilization and service quality. To avoid that, one intuitive approach is to place more replicas for important nodes (e.g., node 8) while the nodes with less importance have only a few. However, this approach ignores the heterogeneity of nodes' importance in the different sessions. In fact, it is a normal case that some nodes may be very important in one session but not in others and some nodes may participate in multiple sessions while playing less important roles.

In this paper, we focus on characterizing the replication problem and devise a novel replication scheme to provide scalable and reliable group communication services for the participants at the edge of network. The scheme is unique in three aspects. First, a new concept of replication based on topological similarity is exploited. It empowers each participant with the ability of measuring similarity between the nodes in topology. Instead of making replication among the physically close nodes, the nodes with low topological similarity are employed as replicas in our replication scheme in such a way it increases the immunity of replication placement strategies to the network partitions. Second, we outline node importance in groups in replication placement. With node importance consideration, the nodes adaptively determine the number of replicas by themselves which eliminates unnecessary replicas, resulting in a reduction of replica creation and maintenance overhead. In addition, it avoids exacerbating the bottleneck problem. Third, in contrast to the existing replication schemes [3, 4, 9–11], our replication scheme is self-converging and it can stabilize within a few adaptations even facing high churn rate.

The rest of this paper is organized as follows: we first give a formal definition of replication problem. In Section 3, the design of the replication algorithm is presented. We provide extensive simulations in Section 4 and review some related work in Section 5. Finally, Section 6 concludes the paper and discusses the research directions.

II. Problem Definition

We study replica placement problem in a general distributed network $G = (V, E, W)$, where V is the set of nodes, $E = V \times V$ is the set of edges and W is the set of weights of the edges. Nodes represent users who participate in the overlay network G, edges represent links between the nodes. Each node v_i in V has a limited capacity C_i. C_i refers to the maximum of $c_i^{avail}(t)$ over all t, where $c_i^{avail}(t)$ denotes the available storage capacity of the node i at time t given the storage capacity is one main factor in the Internet applications like content searching and media streaming dissemination. Weights represent communication cost of the links. Concretely, for each weight ω_k in W, it refers to the propagation delay $l_{i,j}$ required for a packet to transmit from node v_i to node v_j through the link $e_k = (v_i, v_j)$,$\forall\ v_i, v_j \in V, e_k \in E$.

Given the dynamic nature of the network, we consider G as a network where the delays of the links and the available capacities of the nodes may change as the nodes continuously join or leave the network. In such a dynamic network, the problem of replication can be modeled as a multi-objective optimization problem, which can be represented as: for any node $v_i \in V$,$t \in Q^+$

$$\min_{v_j \in R} (\sum_j l_{i,j},\ \sum_j c_j^{avail}(t),\ 1 - R_R(t+\tau|t))\ s.t. \qquad (1)$$

$$\begin{array}{lll} 1) & R = \{v_j\} & (\forall\ j \in [1,N], i \neq j) \\ 2) & c_k^{avail}(t) \leq C_i & (\forall\ k \in [1,N]) \\ 3) & R_R(t+\tau|t)) > P_i(t) & (\forall\ \tau > 0, R \neq \emptyset) \end{array}$$

where the symbols Q^+ and N denote the set of positive rational numbers and the number of nodes in network G respectively. $R_R(t+\tau|t)$ refers to the service reliability offered by node v_i and its replica nodes v_j. It is the probability of a set of nodes $\{v_j\}$ remaining in the network in next time slot τ, which is defined by $R_R(t+\tau|t) = 1 - P_i(x < t+\tau | x > t) \prod_j P_j(x < t+\tau | x > t) = 1 - \frac{P_i(x<t+\tau)}{1-P_i(x<t)} \prod_j \frac{P_j(x<t+\tau)}{1-P_j(x<t)}$. $P_i(t)$ is the probability of node v_i with a lifetime of t seconds.

Our goal is to find a replica placement strategy that minimizes the replica maintenance cost and replica resources usage while maximizes the service reliability offered by nodes $\{v_j\}$. However, it is important to note that this problem does not have an exact solution since practical systems often contain a large number of nodes and the storage capacity of the nodes may range from a few units to a few thousands of units. Potentially,

it is impossible to enumerate all of the solutions. Therefore, our replication algorithm is developed and we will describe it in details in the following section.

III. Topological similarity-based replication scheme

In this section, we start off by introducing topological similarity and describing replica placement consideration. Then, the design of the scheme is described.

A. Topological similarity

Consider the example showed in Fig. 1(b), where the nodes in the region marked with the dash line depart from the multicast session because of the network partition. To avoid that, one remote node is preferred to be placed by node 8 as mentioned before. But the challenge is how to find this type of nodes (e.g., node 26) while keeping cost low. Based on the analysis of the relationship between nodes, we find that two nodes in vicinity have a high similarity in topology. The more shared nodes they have, the closer and more similar they are. Given that, we define a quality index named *topological similarity* as follows:

Definition 1: *Given a node v_i, the topological similarity of v_i to node v_j, denoted by $Sa(i,j)$, is used to describe the similarity between v_i and v_j in topology, which is given by:*

$$Sa(i,j) = \frac{|S_i^{neigh} \cap S_j^{neigh}|}{|S_i^{neigh}| + |S_j^{neigh}|} \tag{2}$$

where S_i^{neigh} and S_j^{neigh} refer to the neighbor set of node v_i and v_j that contain the two-hop neighbors of node v_i and v_j respectively.

Intuitively, the value of $Sa(i,j)$ reflects how similar the nodes are in topology. Given a node v_i, a large value of topological similarity means the node v_j is located close and there is high overlap of the sets of the two nodes, while a small value of that means that the node v_j is located far away from v_i and there is a large space between those two nodes. For simplicity, we use Sa to represent $Sa(i,j)$ if there is no confusion.

Consider the simple network given in Fig. 1. After receiving update messages from nodes 4, 7, 9 and 14, node 8 can have a neighbor set S_8^{neigh} consisting of nodes 3, 4, 5, 7, 9, 10, 12, 13, and 15. S_8^{neigh} shares 3 and 4 nodes with that of node 7 and node 9, respectively. Then, we can get $Sa(7,8) = 0.19$ and $Sa(9,8) = 0.21$. For node 26, it has nodes 14, 15, 16 and 27 in S_{26}^{neigh} which contains 1 node of S_8^{neigh} and $Sa(26,8) = 0.07$. Compared to nodes 7 and 9, node 26 shares fewer nodes and has lower similarity with node 8 in topology. Given the probability of one network partition involving both nodes 8 and 26 is lower than that of network partition marked with dash line, exploring node 26 as one replica node can be a better choice for node 8 to improve its service reliability. Therefore, in this paper, the technique that is related to topological similarity is developed.

B. Algorithm Description

To begin with, we first introduce some important notions.

- The *one-hop neighbor* of node v_i refers to the immediate neighbor node of node v_i [12].
- The *k-hop neighbor* of node v_i refers to the immediate neighbor of node v_i's k-1 hop neighbor, $\forall k > 1$.
- The *incidence vector* of *neighbor set* S_i^{neigh}, denoted by U_i, is a vector whose entries are labeled with the members of S_i^{neigh}. It subjects to: $(U_i)_k = 1 \Leftrightarrow v_k \in S_i^{neigh}$, otherwise $(U_i)_k = 0$.
- The *multicast session set* of the network, denoted by S^M, contains the multicast sessions that are announced by nodes to transmit information to multiple demanding nodes. $S^M(i)$ and $S^M(j)$ are considered to be different if and only if they are published by two different nodes.
- The *multicast session set* of node v_i, denoted by S_i^M, contains the multicast sessions that v_i involves in. It satisfies: for $0 \le i \le N$, $S_i^M \subset S_G^M$, where $S_G^M = \cup_{i=0}^{N} S_i^M$.
- The *importance* of node v_i in $S^M(j)$, denoted by Imp_i^j, is defined as $Imp_i^j = \frac{Number\ of\ descendants\ of\ v_i}{Number\ of\ nodes\ in\ S^M(j)}$, where the descendants of v_i are the nodes who are listening data message from v_i through the message paths.

With the above notions, each node along the multicast tree creates its replica nodes by performing following procedure, which consists of five steps: *candidatelist* construction, reliability determination, topological similarity minimization, cost reduction and replica placing.

Step 1: *Candidatelist* Construction It is accomplished by two operations: *filtration* and *reliability calculation*.

Filtration A tree nodes v_i first builds a new list $list_{v_i}^{ca}$ (called *candidatelist*) and then adds the nodes in S_i^{neigh} whose capacity satisfies $c^{avail}(t) > 0$ to the list. Neither failed nodes nor heavy loaded nodes are considered as the candidate nodes.

Reliability Calculation This operation is triggered as the filtration is finished. Node v_i begins to calculate conditional reliability of nodes in $list_{v_i}^{ca}$ using the formula mentioned in section II. Given $R_R(t+\tau|t)$ measures the probability of the list nodes remaining in the network in next time slot τ, it is utilized to help node v_i choose one appropriate placement strategy, which will be further described in the following steps.

Step 2: Replica Degree Determination The goal of this step is to determine an appropriate value for $|R|$ by carefully combining the reliability theory and importance-aware replication strategy. The procedure of replica degree determination is as follows.

- Node v_i first lists all the nodes in the *candidatelist* with their corresponding reliability in the descending order and computes the minimum number of replica nodes r_1 that satisfies $|\varphi(r_1+1) - \varphi(r_1)| < \delta$, where service reliability $\varphi(x) = 1 - \sum_{i=1}^{x} R_R(t|\tau)$ and δ is a system parameter that is configured by default. Based on both reliability theory and experimental results, we find that there is no apparent improvement in service reliability after δ reaches a certain value. In GeoCast, it is set to 0.01 (as suggested by results in Fig. 5).

- The importance of node v_i Im is then measured by using the information collected from its children nodes, which is defined as $Im = \frac{|S^M(j)|}{\sum |S^M(j)|} \sum Imp_i^j$. Given the importance of node v_i Im, the number of replica nodes r_2 desired by node v_i can be calculated as $r_2 = rounded(1 + Im * W)$, where W represents the local knowledge of node v_i about the network, defined as $W = \frac{log_{2d}G.size - log_{2d}R.size}{|immediate\ neighbors|}$. The method of r_2's calculation ensures that the more important v_i is, the larger r_2 is.
- At last, the replication degree is fixed by choosing the bigger one from r_1 and r_2, which satisfies $|R| = max\{r_1, r_2\}$. To minimize the topological similarity between replica candidate nodes, the first $|R|$ nodes in sorted *candidatelist* are selected and added to a new set (called *prereplicaset*).

Step 3: Topological Similarity Minimization It is achieved by iteratively choosing the candidate nodes with low topological similarity but not including in *prereplicaset* to replace the *prereplicaset* nodes. At each time, node v_i checks if the set *prereplicaset* satisfies: $\varphi_{R'}(|R|) \leq \varphi_R(|R|)$, where $R^{'}$ is a temporary set such that $R^{'} \subset candidatelist$. If so, the node with highest topological similarity is removed and the candidate node is added into *prereplicaset*. This procedure is executed repeatedly until $Sa_{R'} \geq Sa_R$, where $Sa_R = \overline{Sa(i,j)} = \frac{1}{|R|} \sum Sa(i,j),\ \forall v_i, v_j \in R$.

In this procedure, nodes in *candidatelist* that have lower topological similarity to the nodes in *prereplicaset* are only considered. If there is a node u_j such that $Sa_{\{prereplicaset - \{u_i\}\} \cup \{u_j\}} < Sa_R$, u_i that is the node with highest topological similarity is replaced with u_j in *prereplicaset*. Another operation could also be triggered in the similar way. It executes repeatedly until there is no replica placement having the lower topological similarity than Sa_F, where R is the improvement replica placement strategy after the m_{th} operation. In such a way, Sa_R is convergent.

Step 4: Cost Reduction Similar to Step 3, this step is done by replacing the *prereplicaset* nodes with the candidate nodes with less replication cost which satisfying the conditions: $\varphi_{R'}(|R|) \leq \varphi_R(|R|)$ and $Sa_{R'} \leq Sa_R$.

Step 5: Replica Placing Nodes remaining in *prereplicaset* are selected as replicas and node v_i places its files/information on them.

To avoid introducing extra overhead, we attach the maintenance information of replicas to the existing heartbeat messages. Every T seconds, heartbeat mechanism is performed where T is a constant that refers to the parameterized heartbeat period. Long time absence of heartbeat or its responds indicates that the node has left, an then initials the recovery process mentioned in [13].

Once a replica departure is detected by node v_i, the update of replica list is triggered by performing the following steps.

- Node v_i first checks if the condition of $|R| \geq max\{r_1, r_2\}$ is still satisfied. If so, end this update. Otherwise, go to next step.

TABLE I
SIMULATION SETUP

Parameter	Value
Number of Routers	8080 (80 transit routers and 8000 stub routers)
Number of Nodes	[1000,2000,4000,6000,8000]
Link Latency	intra-transit link in [50ms,80ms] transit-stub link in [10ms, 20ms] intra-stub link in [1ms, 5ms]
Capacity Distribution	5% nodes with 1000 units 15% nodes with 100 units 30% nodes with 10 units 5% nodes with 1 unit
Group Number	$\lfloor (log_{1.25}\frac{N}{11.5})^{-1} \rfloor$ -1
Group Size	$[\lfloor N * r^{-1.25} + 0.5 \rfloor, 11]$
Lifetime Distribution	Pareto ($\alpha = 1.1$,$\beta = 0.05$)
Simulation Time	7200
r	[0,8]
RT	[30,120]s

- The *candidatelist* and *prereplicaset* are created and initialized by adding the qualified nodes that are in the replica set. Then, go to next step.
- With such setting of *candidatelist* and *prereplicaset*, the proposed replication algorithm is performed.

The computation complexity of the topological similarity-based replication algorithm is $O(n(logk + k))$, where n is the number of nodes contained in S^{neigh} and k is a variable ranged from 1 to n. Since a network may consist of a large number of nodes, it is necessary for the nodes to reduce the computation complexity. The most effective way is to reduce the number of nodes in the neighbor sets and the number of replica nodes. But from experiments we find that n and k are normally within [10, 20] and [1,6] respectively that enable the complexity of the proposed scheme is low even in a large network. In addition, it is important to point out that not all nodes in S^{neigh} are valid for replica selection since some nodes may be overloaded at that point in time.

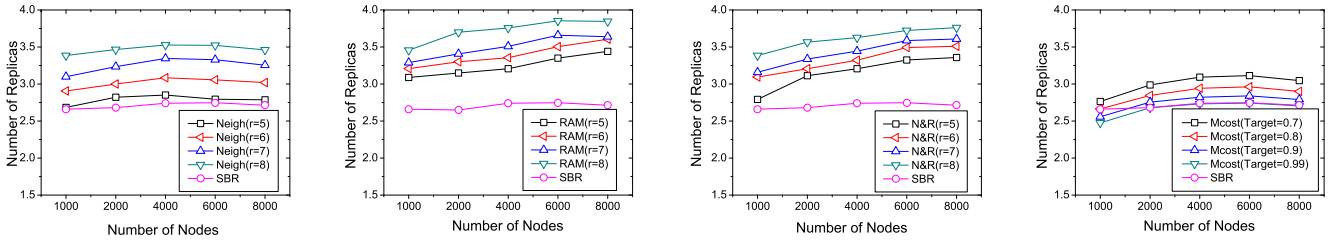
IV. PERFORMANCE EVALUATION

In this section, we use extensive simulations to evaluate the proposed algorithm.

A. Experimental environment

We use Transit-Stub graph model of the GT-ITM topology generator to generate network topologies for our simulation. All experiments in this paper are run on 10 topologies with 8080 routers. Table I lists the detailed parameter setting used in our simulations.

With the lack of real-world trace data to drive these experiments, we use the method mentioned in [14] to simulate distribution of group size and group number. A modified range $[\lfloor N * ra^{-1.25} + 0.5 \rfloor, 11]$ is set on the group size, where N is the number of nodes in the system and ra is a system configured variable, ranging from 2 to $\lfloor (log_{1.25}\frac{N}{11.5})^{-1} \rfloor$. We simulate the behaviors of joining nodes in the network using Pareto lifetime function [15] with $\alpha = 1.1$ and $\beta = 0.05$, where the average lifetime of joining nodes is around 0.5 hour.



(a) Neigh (b) RAM (c) N&R (d) Mcost

Fig. 2. Comparison of replica number in terms of node number

To study the effectiveness of TSRS, we compare the scheme TSRS to RAM, Neigh, N&R and Mcost in various scenarios. RAM [10, 16] in our simulations refers to the random replication scheme where the locations of data copies are selected by following an uniform distribution. Neigh refers to the neighbor-based replication approaches [8, 17]. N&R is a simple extension scheme of Neigh and RAM designed to reduce construction cost caused by RAM and improve reliability of Neigh. Mcost refers to the replication schemes [9, 18] with the aim of minimizing both replica construction cost and its maintenance cost while satisfying system reliability requirement.

B. Results

We first make a comparison between the replication schemes in networks with different size, as shown in Table II. In Table II, n_1 and n_2 denote total number of interrupted services with and without replication scheme respectively. The results lead to two observations. First, all of three replication schemes: Neigh, RAM and N&R, can improve their performances by increasing the replica number r. However, after r reaches 5, increasing the number of replica nodes does not achieve dramatic improvement in the metric. This is, essentially, because of the redundancy among the replica nodes. Second, the scheme TSRS yields better performance in all cases than Mcost and other counterparts with smaller r (i.e., $r < 5$). Similar to Neigh and N&R with $r = 5$, TSRS achieves a reduction from 99.35% to 99.61% in service interruption. This benefits from the flexibility and reliability of the TSRS replica nodes, which enables them to have high probability to be alive and consequently be able to detect and react to the changes of the network quickly. The results also show that Mcost achieves a poor performance. This is due to the influence of the failures of nodes with high reliability. In Mcost, the nodes with higher reliability may have fewer replicas, but heavier backup workload.

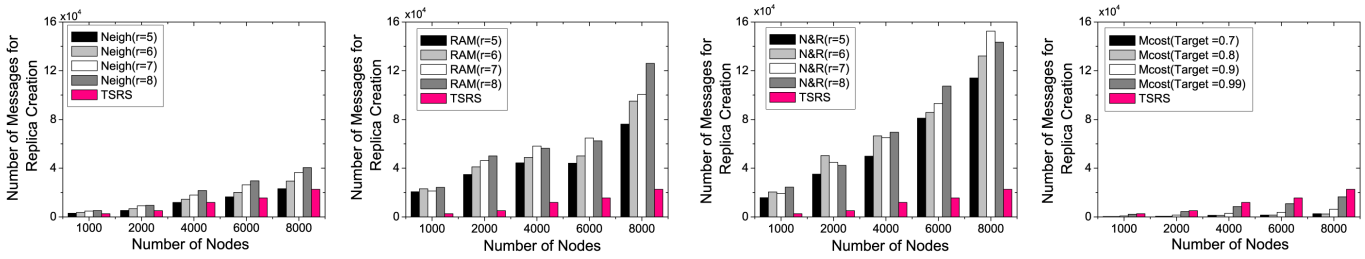
Fig. 2 measures the average number of the valid replica nodes per node as a function of network size for the schemes. As the results in Table II suggest, we vary r from 5 to 8 to obtain the optimal performance of the schemes so that we can study the substantial benefits of TSRS. We observe that both TSRS and Mcost yield better performance than the other schemes in terms of replica node number. But unlike TSRS, Mcost fails to achieve high efficiency in terms of the number of service interruptions, as shown in Table II. Potentially, this indicts that the scheme of TSRS provide a better way for data

TABLE II
COMPARISON BETWEEN REPLICATION SCHEMES IN NETWORKS WITH DIFFERENT SIZE

		n_1				
		N=1000	N=2000	N=4000	N=6000	N=8000
Neigh	r=1	82	220	397	480	690
	r=2	36	131	197	353	495
	r=3	35	92	182	284	234
	r=4	23	72	105	161	168
	r=5	14	49	54	126	145
	r=6	11	40	36	76	111
	r=7	9	16	33	65	91
	r=8	5	11	21	44	82
RAM	r=1	84	211	380	585	649
	r=2	45	150	248	402	437
	r=3	36	80	180	177	277
	r=4	13	76	96	153	136
	r=5	8	23	53	97	110
	r=6	6	25	48	9p	75
	r=7	5	23	41	64	70
	r=8	2	20	33	50	65
N&R	r=1	80	203	385	496	687
	r=2	47	91	333	296	403
	r=3	46	74	124	210	301
	r=4	17	65	142	197	217
	r=5	13	45	90	105	154
	r=6	11	38	84	99	107
	r=7	8	17	50	95	103
	r=8	2	16	49	58	72
Mcost	Target = 0.7	122	246	562	700	898
	Target = 0.8	124	247	558	702	897
	Target = 0.9	120	217	465	588	760
	Target = 0.99	102	224	372	563	690
TSRS		13	46	77	93	135
n_2		1988	7232	12409	22302	34419
Reduction Rate(%)= $\frac{n_2-n_1}{n_2}$		99.35%	99.36%	99.38%	99.58%	99.61%

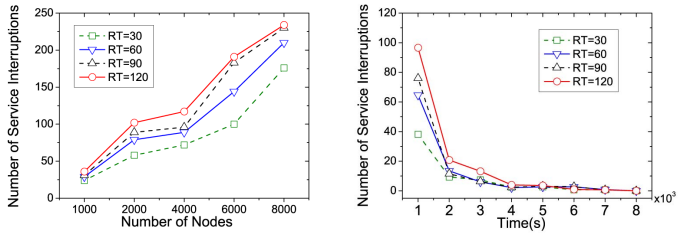
replication with consideration of the reliability of nodes. Given Mcost is not practical due to the poor performance of Most in dealing with node failures and the network partitions, we ignore the analysis of such method in the rest of paper for reliability purposes.

Fig. 3 shows that the performance of RAM and N&R heavily depends on the setting of parameter r. The larger r is, the more messages are generated. By comparing RAM with N&R, we see that RAM generates less messages for replica creation than N&R, especially when the network size is larger. This is because of the extra overhead generated for neighborhood inquisition. In N&R, the inquire messages are initiated and sent to both remote nodes and neighbor nodes. Since it is hard to determine the range of the inquisition, a large number of messages might be issued during this procedure and consequently degrades the performance of applications.



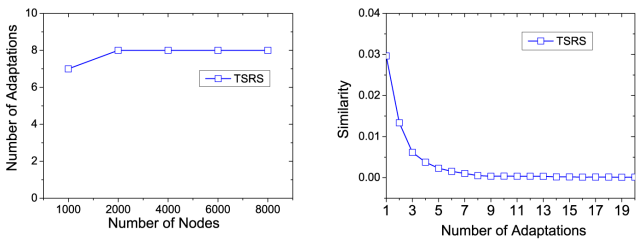
(a) Neigh (b) RAM (c) N&R (d) Mcost

Fig. 3. Comparison of Creation Cost in Terms of Node Number



(a) $N \in [1000, 8000]$ (b) N=4000

Fig. 4. Effect of RT



(a) $\Delta Sa \leq 10^{-4}$ (b) N=4000

Fig. 5. Convergence Speed

Different with them, TSRS yields better performance. The replica number in TSRS are delicately determined by nodes based on their statuses and the network condition.

Fig. 4(a) illustrates the effectiveness of TSRS under different RT. RT refers to the time required for node initialization right after it is selected as a replacement node to take over the *island* previously owned by failed node. We vary RT from 30s to 120s. It is observed that more services tend to be interrupted when RT is larger, as well as when the network size is larger. This can be explained by the fact that as growing RT the nodes in the network are more likely to fail in a long time interval and it consequently leads to a poor performance in terms of service interruption. However, given the results showed in Table II, we find that the scheme of TSRS with RT= 60s can still have a high reduction rate. For instance, in the network of 8000 nodes, it achieves 99.3% saving. The results in Fig. 4(b) show the distribution of service interruptions over time in the network of 4000 nodes. We note that the metric gradually drops with the run time. This is because that majority of node failures happen at the beginning stage of simulation, which confirms the nature of nodes in the underlay network.

Fig. 5(a) illustrates the convergence of the algorithm TSRS when the network size changes. The upper bound of parameter ΔSa, defined as $\Delta Sa = Sa_{s_{i+1}} - Sa_{s_i}$ is set to 10^{-4}. This value is chosen to make sure that there is no big difference between the topological similarity of the replica placement strategies when the algorithm is stable. The results show that in all cases the time required for convergence to stable is short(less than 10). Fig. 5(b) provides a further study of the performance of the algorithm in the network of 4000 nodes. We see that after a few adaptations, TSRS reduces the topological similarity of placement strategies to a small value, which implies the fast convergence of the algorithm.

V. Related Work

Many research studies have also been conducted to cope with massive node failure and keep high data availability by using the technique of file replication. In those studies, location of the replica nodes are determined based on node ID [1, 2, 6], query path [3–5], or neighbor locality [7, 8].

The ID-based replication determines replica nodes based on the relationship between the node ID and the data/file's ID, where the replica nodes are the nodes whose IDs match most closely to the data/files' IDs. In PAST [1], each file is replicated on a set of nodes whose node ID are numerically closest to the files' ID. A new replica is created immediately following a failure. CFS [2] divides files into blocks and spreads them evenly over the available servers to prevent large files from causing unbalanced use of storage. In recent work, Hirokazu et al. [6] propose a distributed interval tree replication scheme for adaptively setting the replicated objects considering the scale of networks. The replicas are assigned to the nodes of the interval tree created by the node with file.

In the path-based replication, the data/file copies are placed on the nodes along the file routing path from the requester node to the provider node. LAR [3] creates replicas on the source of the query and adds routing hints to the replica on nodes along the message query path. In Freenet [4], each node along the path from the source node to the query node stores a copy of the file. OceanStore [5] places the replicates data on or near the client machines where the data is accessed. Those replicas function as local access points for the data. However, due to time-varying file popularity and node interest variation, most of the replicas cannot be fully utilized.

The neighbor locality based replication makes file replication among the nodes located in the neighborhood of the data/file host node. Plover [7] makes file replication among physically close nodes based on node available capacities.

It enables the file replication and consistency maintenance to be conducted among physically close nodes. Peercast [8] developed a neighbor-based replication scheme, where data copies are put on nodes who are locating in the neighborhood. However, this type of schemes has two drawbacks. First, it ignores the heterogeneity of nodes' importance in the services. Second, it subjects to a difficulty in determining replication degree.

There are other studies [9–11, 19] for file replication based on the file popularity or request rate. Most of them focus on the relationship among the number of replicas, file search time, and load balance, but do not investigate the impact of replica location on file query efficiency.

There are two important features distinguish our approach from the existing approaches. First, our approach leverages node heterogeneity in lifetime, capacity and importance to improve the replication flexibility and reduce the replication cost caused by replica creation and maintenance. Second, our approach makes good use of the information encoded in the link structure of the nodes and reduces the impact of different network partition on the service quality.

VI. Conclusion and future work

This paper proposes a cost-effective replication scheme to provide scalable and reliable group communication services. It is novel as it exploits nodes' topological similarity and importance heterogeneity in a distributed fashion to improve the service reliability while keeping the replication overhead low. Through extensive experiments, we demonstrate that TSRS increases the reduction rate up to 99.63% and reduces the number of data copies up to 60% when comparing to the existing replication degree based approaches, the cost-aware replication schemes, and their simple alternative schemes. The results also show that TSRS is practical as it can deal with failures in different patterns without scarifying the service quality in terms of service interruption and replication overhead. Our future work will focus on developing techniques to increase the utilization of the replica nodes and improve the efficiency of our proposed scheme.

VII. Acknowledgment

This paper is supported by the National Basic Research 973 Program of China under Grant No. 2009CB320805, the National Natural Science Foundation of China under Grant No. 61170188, the National High Technology Research and Development 863 Program of China under Grant No. 2012AA011803, and Fundamental Research Funds for the Central Universities of China. The third author is partially supported by grants from NSF CISE NetSE program, CyberTrust program, an IBM faculty award and an Intel ISTC on Cloud Computing.

References

[1] P. Druschel and A. Rowstron, "Past: A large-scale, persistent peer-to-peer storage utility," in *HotOS'01*, 2001, p. 0075.

[2] F. Dabek, M. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with cfs," *ACM SIGOPS Operating Systems Review*, vol. 35, no. 5, pp. 202–215, 2001.

[3] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, and P. Keleher, "Adaptive replication in peer-to-peer systems," in *ICDCS'04*, 2004, pp. 360–369.

[4] I. Clarke, O. Sandberg, B. Wiley, and T. Hong, "Freenet: A distributed anonymous information storage and retrieval system," pp. 46–66, 2001.

[5] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, C. Wells *et al.*, "Oceanstore: An architecture for global-scale persistent storage," *ACM SIGARCH Computer Architecture News*, vol. 28, no. 5, pp. 190–201, 2000.

[6] H. Yoshinaga, T. Tsuchiya, H. Sawano, and K. Koyanagi, "A study on scalable object replication method for the distributed cooperative storage system," in *ICDT'09*, 2009, pp. 96–101.

[7] H. Shen and Y. Zhu, "Plover: a proactive low-overhead file replication scheme for structured p2p systems," in *ICC'08*, 2008, pp. 5619–5623.

[8] 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(JNCA)*, vol. 31, no. 4, pp. 821–850, 2008.

[9] H. Yu and A. Vahdat, "Minimal replication cost for availability," in *PODC'02*, 2002, pp. 98–107.

[10] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *ICS'02*, 2002, pp. 84–95.

[11] S. Tewari and L. Kleinrock, "Proportional replication in peer-to-peer networks," in *Infocom'06*, 2006.

[12] 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.

[13] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *SIGCOMM'01*, vol. 31, no. 4, 2001, pp. 161–172.

[14] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, "SCRIBE: A large-scale and decentralized application-level multicast infrastructure," *IEEE Journal on Selected Areas in communications*, vol. 20, no. 8, pp. 1489–1499, 2002.

[15] X. Wang, Z. Yao, and D. Loguinov, "Residual-based estimation of peer and link lifetimes in P2P networks," *IEEE/ACM Transactions on Networking(TON)*, vol. 17, no. 3, pp. 726–739, 2009.

[16] Y. Liu, X. Liu, L. Xiao, L. Ni, and X. Zhang, "Location-aware topology matching in P2P systems," in *Infocom'04*, 2004, pp. 2220–2230.

[17] T. Chang and M. Ahamad, "Improving service performance through object replication in middleware: a peer-to-peer approach," in *P2P'05*, 2005, pp. 245–252.

[18] Y. Saito and M. Shapiro, "Optimistic replication," *ACM Computing Surveys (CSUR)*, vol. 37, no. 1, p. 81, 2005.

[19] S. Tewari and L. Kleinrock, "Analysis of search and replication in unstructured peer-to-peer networks," vol. 33, no. 1, pp. 404–405, 2005.