A Decentralized Scheme for Network-Aware Reliable Overlay Construction

Shinichi Ikeda, Tatsuhiro Tsuchiya, and Tohru Kikuno

Graduate School of Information Science and Technology, Osaka University,
Osaka 565-0851, Japan
{t-tutiya, kikuno}@ist.osaka-u.ac.jp

Abstract. Peer-to-peer (P2P) systems are often constructed in overlay networks at the application layer without taking the physical network topologies into consideration. The mismatch between physical topologies and logical overlays can cause a large volume of redundant traffic and considerable performance degradation of the P2P systems. In order to alleviate this mismatching problem, we propose an algorithm that iteratively reshapes the topology of an overlay. The algorithm is fully decentralized and only relies on local information available at each node. Also, the algorithm preserves the total number of links during the process of iterative modifications, thus maintaining the resiliency to failures.

1 Introduction

In the past few years, we have seen the rapid growth of peer-to-peer (P2P) applications and the emergence of overlay infrastructures for the Internet. A challenging research problem in this area is to develop overlay architectures that can support these P2P applications without overloading network resources.

We discuss the issue of topology mismatching in this paper. Overlay networks are typically constructed at the application layer without taking the physical network topologies into consideration. The mismatch between physical topologies and logical overlays can cause a large volume of redundant traffic and considerable performance degradation of the P2P systems [1].

Recent approaches to building efficient overlay routing networks employ the abstraction of a distributed hash table (DHT) [2,3,4]. These DHT schemes use a global naming scheme based on hashing to assign keys to data items and organize the nodes into a graph that maps each key to a responsible node. The graph is hierarchically structured to enable efficient routing with the DHT. These protocols assume obedience to the protocols and ignore participants' incentives. In reality, however, nodes may behave selfishly, seeking to maximize their own benefit.

In this paper, we consider unstructured overlay networks consisting of nodes that behave selfishly; we propose a solution to the topology mismatching problem for such networks. Unstructured overlays are widely used in Internet-wide deployed systems. These overlays do not rely on global naming or any additional hierarchical structure. Instead, they use flooding, epidemic protocols, or random walks to disseminate messages or to query content stored by overlay nodes.

Schemes for building unstructured overlays are typically oblivious to the underlying network topology. Hence, communication between nodes on the overlay tends to impose a high load on the network and to result in unnecessary performance degradation.

The proposed algorithm optimizes unstructured overlays according to a proximity criterion in order to reduce network load. The algorithm iteratively reshapes the topology of an overlay in a way that reflects geographic locality so as to reduce network load. The algorithm is fully decentralized and only relies on local information available at each node. Also, the algorithm evenly balances the node degree while preserving the total number of links during the process of iterative modifications. Because of this property, the overlay can maintain and even improve the resiliency to failures.

Our work is the most closely related to the work by Massoulié et al [5]. In [5], they proposed an algorithm called Localiser, which shares many characteristics with our algorithm. Apart from small design details, the most significant difference between Localiser and our algorithm is that in ours each modification to the topology is performed only when the nodes involved in this action will benefit from this modification, which means that our algorithm allows each node to behave selfishly. In contrast, in the Localiser algorithm, the node that initiates a modification action must sacrifice its benefits by disconnecting a link to a neighbor node. Hence their algorithm can work well only when the overlay nodes are obedience to this algorithm and act in a self-sacrificing manner.

Other related work differs from ours, for example, in that dedicated servers are required [6], or in that resiliency is not taken into consideration [7,8].

The remaining part of this paper is organized as follows. In Section 2 we outline our design requirements for overlay networks. In Section 3 we present a basic idea behind the proposed algorithm and give the description of our algorithm. Results of an experiment are presented in Section 4. Section 5 concludes the paper.

2 Background

2.1 Unstructured Peer-to-Peer Overlay Networks

We can view an unstructured P2P overlay network as an undirected graph, where the vertices correspond to nodes in the network, and the edges correspond to open connections maintained between the nodes. A node i is said to be a *neighbor* of another node j iff they maintain a connection between themselves. The *node degree* of a node i is the number of i's neighbors. Messages may be transferred in either direction along the edges. For a message to travel from node i to node j, it must travel along a path between these node in the graph.

There are many schemes for building unstructured-overlay networks. In the most traditional scheme, a node that wishes to join the overlay network connects to highly publicized nodes or nodes known by a bootstrapping node. This preferential attachment feature and the incremental growth nature of P2P systems account for scale-free network-type topologies [1,9]. In these networks the

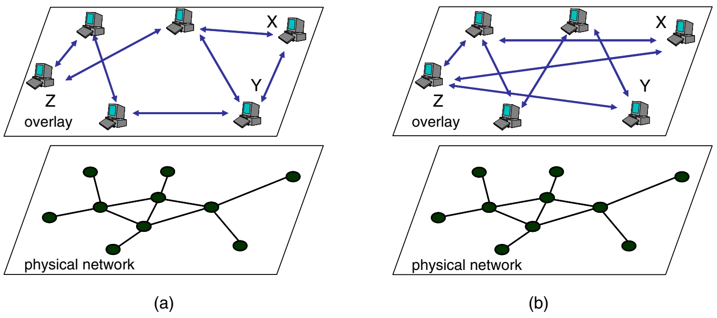

Fig. 1. Two different mappings of the overlay to the underlying network

node degree distribution follows a power-law; most nodes have few links and a tiny number of hubs have a large number of links. Due to this property, these networks are highly robust when facing random node failures, but vulnerable to well-planned attacks.

More sophisticated protocols can be used to obtain more "balanced" structures of overlay networks. For example, Scamp [10], a decentralized protocol for building unstructured overlay networks, yields an overlay network where the degree distribution is a normal distribution-like curve. Scamp can do this without maintaining any global information about the overlay.

These protocols are typically oblivious to the underlying network topology; all connections are treated as if they are equivalent, and so the resulting overlay does not reflect the underlying network topology.

2.2 The Topology Mismatching Problem

As stated before, overlay networks are often constructed without taking the physical network topologies into consideration. The mismatch between physical topologies and logical overlays is a major cause of performance degradation. Given the considerable traffic volume generated by P2P applications, it is crucial also from the perspective of their impact on the network infrastructure that they efficiently utilize available networking resources. The larger the mismatch between the underlying physical network topology and the P2P application's overlay topology, the bigger the stress on the underlying network.

This problem of topology mismatching is illustrated in Fig. 1. Fig. 1(a) and Fig. 1(b) depict overlay networks where six nodes are participating. As shown in these figures, these two overlays are built on the same physical network, which is schematically illustrated as the undirected graph where a vertex represents a router or an end host. In Fig. 1(a), the overlay closely matches the underlying network topology. The transmission of a message generated by node X to Y involves only two physical links.

In the right picture, on the other hand, the overlay topology does not match the infrastructure that well. In this case, the shortest path from X to Y in the overlay is X-Z-Y which is two-hop long. If the message from X to Y is routed along this path, at least eight physical links are involved in this communication.

3 The Proposed Algorithm

To alleviate the topology mismatching problem, we propose an algorithm that reshapes the overlay topology by making each node modify the connections between their neighbors. Each node following this algorithm relies only on local information in determining how to modify its connections autonomously. The self-organizing behavior of an overlay network emerges from a collection of such autonomously behaving nodes.

To evaluate the global cost of an overlay topology G, we introduce the following cost function, as in [5]:

$$C(G) \equiv \sum_{(i,j)\in E} c(i,j) + w \sum_{i\in V} d_i^2 \tag{1}$$

where

- E is the set of edges.
- $c(i,j)$ is the cost of communication between nodes i and j. $c(i,j)$ is a proximity metric in the underlying physical network and could be the round trip time measured with ping or a more complex measure incorporating bandwidth availability on the path between i and j.
- w is a non-negative real constant.
- V is the set of nodes.
- d_i is the degree of node i.

This cost function is used in our algorithm to determine whether each possible topology transformation is effective or not; that is, if $\Delta C \equiv C(G') - C(G)$ is less than zero, then the transformation from G to G' is effective in the sense that the cost function value will be decreased.

As will be stated later, our algorithm maintains the total number of links. Hence the parameter w specifies the emphasis placed on degree balancing relative to locality. If $w > 0$, then the right term in (1) is minimized when d_i is the same for all i. On the other hand, if $w = 0$, the graph is optimized only to take into account network proximity.

3.1 Algorithm Steps

The proposed algorithm is fully decentralized. Each node i executes the following steps locally and periodically.

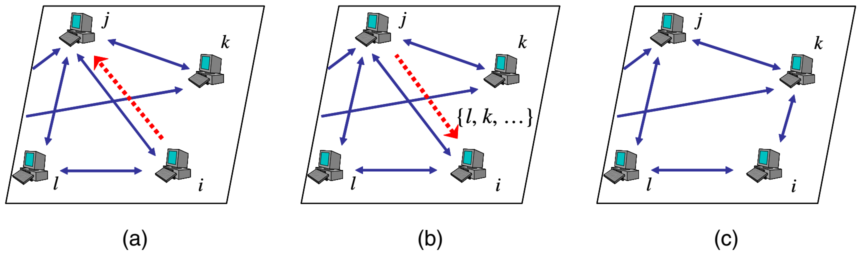

Fig. 2. Transformation steps

1. Choose one of its neighbors, node j, uniformly randomly. Send a message to node j (Fig.2(a)).
2. Node j sends back to i the set L of the addresses of j's neighbors (Fig.2(b)).
3. Choose $\min\{f, |L|\}$ nodes $\{k_1, k_2, \ldots, k_{\min\{f,|L|\}}\}$ in L that are not neighbors of i, uniformly randomly. If no such node exists in L, then stop. Otherwise, measure the cost from i to each chosen node $k' \in \{k_1, k_2, \ldots, k_{\min\{f,|L|\}}\}$ and evaluate the cost resulted from replacing link (i, j) with link (i, k'). The cost is the difference of global costs between these topologies, which is

$$\Delta C \equiv c(i, k') - c(i, j) + 2w(d_{k'} - d_j + 1).$$

4. Choose the node, say node k, that leads to the minimum ΔC. If the minimum $\Delta C \geq 0$, then reject this transformation; otherwise, perform it as follows: Send a message to k asking it to establish a connection with i. If k accepts this request, then establish the link between i and k, and remove the link between i and j (Fig.2(c)).

Note that the node i, which is the initiator of this transformation, does not lessen its benefits in this transformation, since it simply replaces a high-cost link with a new link which is lower cost. Also, node k can reject to establish the link between i and k if it expects that the new connection would incur additional cost. Thus our algorithm does not impose on the nodes the burden of behaving in a self-sacrificing fashion. As stated in the first section, this is in contrast to LOCALISER, in which the node that initiates a modification action must bear cost of disconnecting a link to a neighbor node [5].

It should also be noted that the algorithm makes no changes that would increase the global cost. This is subtle but important difference from LOCALISER. LOCALISER is a Metropolis algorithm, which permits the system to transit a worse state with a certain probability to avoid to get stuck at local minima of the cost function.

Since our algorithm does not allow such transitions, one might think that our algorithm would cause the system to converge to undesirable local optima. From our preliminary experiments, however, we found that this is not the case.

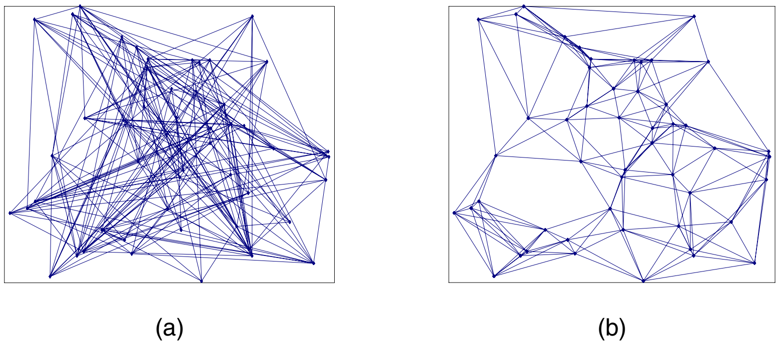

Fig. 3. Topologies of overlay networks: (a) the initial network; (b) the network yielded by the proposed algorithm

In the preliminary experiments, we repeated many runs of the algorithm while varying the initial networks and the probability of accepting a bad transition; but we found no clear case where accepting bad transitions caused a better topology or faster convergence. Our conjectured reason for this finding is the distributed nature of the algorithm; in the algorithm every node can have a chance to initiate a topology change, which means that there are always many candidate transformations that are checked in Step 4, so are there acceptable transformations.

3.2 An Illustrative Example

Here we describe an illustrative example of 50 nodes placed uniformly at random on a two-dimensional space. In this example, the communication cost between two nodes is defined as the Euclidean distance between them on the two-dimensional space. Fig. 3(a) depicts an example of an unstructured overlay which is not optimized. The proposed algorithm refines this initial overlay by make each node modify its connections so as to favor near nodes as its neighbors. Fig. 3(b) shows the topology that was obtained by repeating such a local modification sufficiently many times. Note that the number of edges is the same in Figs. 3(a) and 3(b). Hence the apparent sparsity of the latter network is due to the edges being much shorter on average.

4 Experimental Results

In this section, we show the results of our experiments. We first constructed the initial networks of 100 nodes and 200 nodes. We adopted the Barabasi-Albert scale-free network model [9]; we started with a small number m_0 nodes, at every

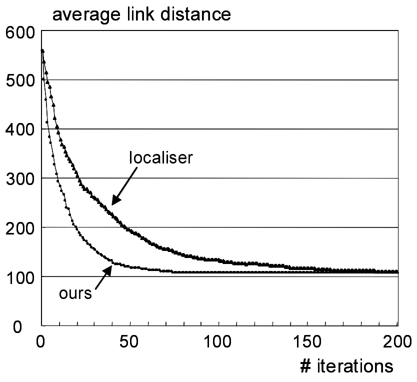

Fig. 4. Average distance of the links in the overlay as a function of number of iterations

time step we added a new node with $m(\leq m_0)$ links that connect the new node to m different nodes already present in the network. The probability Π that a new node will be connected to node i depends on the connectivity d_i of that node, so that $\Pi(d_i) = d_i / \sum_i d_i$. We set $m_0 = 3$ and $m = 3$.

The communication cost $c(i, j)$ between two neighboring nodes i, j was set to the Euclidean distance between them on a 1000×1000 square where all nodes are uniformly randomly placed.

We simulated the behaviors of our proposed algorithm ($f = 5$) and the Localiser algorithm [1] when they were applied to these two initial topologies. We set w in the cost function $C(G)$ to 20. In doing so, we assumed that all nodes are loosely synchronized and execute each iteration of these algorithms in rounds. We also assumed that a node does not reject the request for establishing a link in Step 4 in our algorithm.

4.1 Link Distance

To evaluate the performance of our algorithm in reflecting the underlying physical network topology, we measured the average communication cost between two end nodes of a link in the overlay.

Fig. 4 presents how the average communication cost varied as the time elapsed in the network of size 100. One can clearly see that the average cost gradually decreases as the number of iterations executed increases and that our algorithm is faster to converge than Localiser. Almost the same curves were obtained for the network of size 200.

Table 1 summarizes the average link distance when the algorithms were iterated 50, 100, and 200 times, together with the initial values. These results demonstrate that our algorithm is more effective than the Localiser algorithm in capturing the underlying network topology.

[1] The design parameter T was set to 1.

Table 1. Average distance of the links in the overlay

network	100 nodes				200 nodes			
iteration	0	50	100	200	0	50	100	200
localiser	558	193	132	110	541	187	110	78
ours	558	118	108	107	541	97	76	73

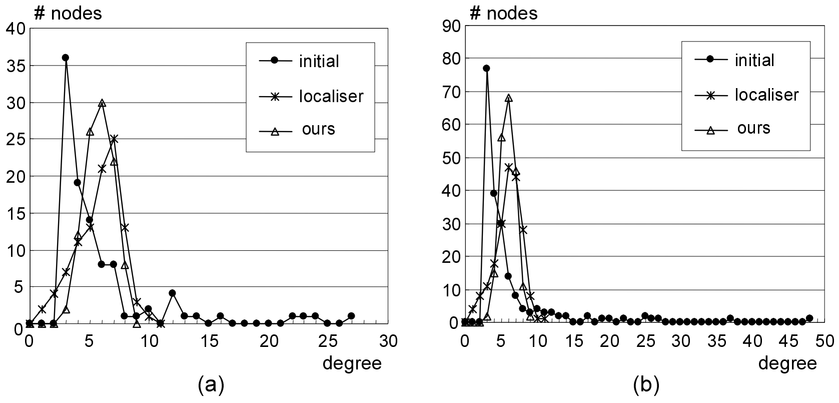

Fig. 5. Distribution of node degrees

4.2 Node Degree

Here we show how the proposed algorithm affects the node degree. Figs. 5(a) and 5(b) depict the distributions of the degree for the networks obtained just after 50 iterations. The results shown in Figs. 5(a) and 5(b) are those obtained for the networks of size 100 and 200, respectively. The horizontal axes represent the node degree, while the vertical axes denote how many nodes have that degree. From these graphs, one can see that the distribution becomes sharply concentrated around the average value when our algorithm or Localiser is used. In the initial network, a few nodes have very high connectivity, and the other most nodes have only few links. Intuitively speaking, this helps decrease the sensitivity to failures by balancing the reliance evenly on all the nodes. The resiliency to failures is more directly evaluated in the following subsection.

4.3 Resilience

To address the resilience of the networks, we study the probability that the network is fragmented into isolated subnetworks when some fraction of the nodes have failed. Here we consider the six topologies: the two initial topologies of sizes 100 and 200; the two topologies yielded by our algorithm; the two topologies yielded by Localiser. We performed 50 iterations for each algorithm.

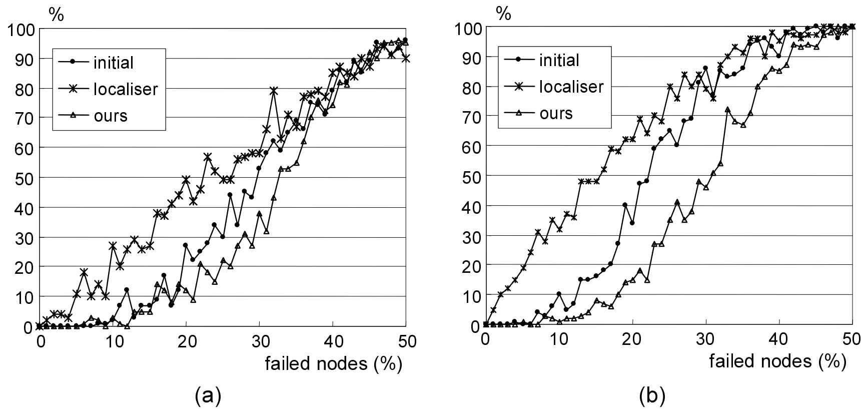

Fig. 6. Network fragmentation probability in the face of node failures

We selected 100 random patterns of failed nodes for each different percentage of failed nodes, and measured how many times that network fragmentation occurred for each of these topologies.

The results of these experiments are depicted in Fig. 6. Figs. 6(a) and 6(b) display the results for the networks of size 100 and 200, respectively. In these graphs, the horizontal axes represent the percentage of failed nodes, while the vertical axes denote the mean percentage that correct nodes are isolated into two or more partitioned network fragments.

As clearly seen in these graphs, in a large range of percentage of failed nodes, the robustness to node failures was significantly improved, by applying our algorithm. For example, if failed nodes were less than 10%, almost no fragmentation occurred in the topologies optimized by our algorithm. On the other hand, network fragmentation was observed for the initial networks and those yielded by Localiser with a greater likelihood.

5 Conclusions

In this paper, we have proposed an algorithm which optimizes unstructured large-scale overlay networks by reshaping the overlay via iterative refinements. The proposed algorithm helps the overlay reflect the underlying network topology while maintaining the resiliency to failures. The algorithm is fully decentralized and only relies on local information available at each node. We present the results of simulation which demonstrate the effectiveness of the algorithm. Future work includes the study of the algorithm in more practical settings. Specifically, we consider using the GT transit-stub model [11], as a model of underlying networks.

Acknowledgments

This research was supported in part by "Priority Assistance for the Formation of Worldwide Renowned Centers of Research — The 21st Century Center of Excellence Program" of the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

1. Ripeanu, M., Foster, I., Iamnitchi, A.: Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. IEEE Internet Computing Journal **6** (2002)
2. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A scalable content-addressable network. In: SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, New York, NY, USA (2001) 161–172
3. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A scalable Peer-To-Peer lookup service for internet applications. In: Proceedings of the 2001 ACM SIGCOMM Conference. (2001) 149–160
4. Rowstron, A., Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany (2001)
5. Massoulié, L., Kermarrec, A.M., Ganesh, A.J.: Network awareness and failure resilience in self-organising overlay networks. In: Proceedings of the 22nd IEEE Symposium on Reliable Distributed Systems (SRDS '03). (2003) 47–55
6. Xu, Z., Tang, C., Zhang, Z.: Building topology-aware overlays using global soft-state. In: ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems, Washington, DC, USA (2003) 500–508
7. Liu, Y., Zhuang, Z., Xiao, L., Ni, L.M.: A distributed approach to solving overlay mismatching problem. In: 24th International Conference on Distributed Computing Systems (ICDCS 2004), Tokyo, Japan, IEEE (2004) 132–139
8. Ren, S., Guo, L., Jiang, S., Zhang, X.: Sat-match: A self-adaptive topology matching method to achieve low lookup latency in structured p2p overlay networks. In: 18th International Parallel and Distributed Processing Symposium (IPDPS 2004). (2004)
9. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286** (1999) 509–512
10. Ganesh, A.J., Kermarrec, A.M., Massoulié, L.: Peer-to-peer membership management for gossip-based protocols. IEEE Transactions on Computers **52** (2003) 139–149
11. Zegura, E.W., Calvert, K.L., Bhattacharjee, S.: How to model an internetwork. In: IEEE Infocom. Volume 2., San Francisco, CA, IEEE (1996) 594–602