P2P Overlay Networks of Constant Degree

Guihai Chen[1,2], Chengzhong Xu[2], Haiying Shen[2], and Daoxu Chen[1]

[1] State Key Lab of Novel Software Technology, Nanjing University, China
[2] Department of Electrical and Computer Engineering, Wayne State University, USA
{gchen, czxu, shy}@ece.eng.wayne.edu

Abstract. This paper proposes an abstract and generic topological model that captures the essence of P2P architecture. Such model should in return facilitate the exploitation of the new design space for more novel P2P systems. The paper also proposes a variant of the cube-connected cycles as a P2P overlay network that achieves O(log N) path length with O(1) neighbors. The simulation result has shown that our system is no worse than other systems of the same complexity such as Viceroy and Koorde.

1 Introduction

Several research groups have independently proposed a second generation of P2P systems that support scalable routing and location schemes. Examples include Chord[10], CAN[6], Pastry[8], Tapestry[12], Koorde[2] and Viceroy[4]. We find that all such new generation of P2P systems share a common feature that peers are connected at application-level as a regular network such as a ring, a mesh, or a hypercube that are often called overlay network. As the P2P architecture is becoming popular and more systems are introduced, we think an abstract and generic topological model that captures the essence of P2P architecture is needed. Such model should in turn facilitate the exploitation of the new design space for more novel P2P systems.

Based on the above investigation, we propose a generic topological model in Section 2 to abstract the new generation of P2P systems. We also address the fundamental metrics to evaluate the performance of P2P systems and the parameters that affect the metrics. While the typical P2P overlay networks have O(log N) path length and O(log N) routing table space, we propose yet another new topology for P2P overlay network in Section 3 that achieves O(log N) path length with O(1) neighbors. Section 4 concludes the paper by compareing our system with other systems[2,4] that have the same complexity and pointing out the possible direction for future research. We expect more novel P2P overlay networks to be found in the near future. We also expect the well-established technologies for interconnection networks could be borrowed for the study of P2P overlay networks.

2 Topological Model

In developing our abstract model, we made two observations. First, in any P2P system, we can see that there are multiple nodes spreading across the entire Internet and contributing their local resource (*e.g,.* storage and data) to the global P2P system. The Internet can be modeled as a general interconnection network that connects these nodes. Second, we note that the main functions of peer node in a P2P system are message routing and data locating besides data storing and caching. The routing function routes a given request/reply message from the source to the destination node. The data locating function attempts to find out the location or *home node* of a data item, with the help of routing table. All peer nodes are connected as an overlay network in somewhat regular way so that both the routing table size and the routing path are dramatically reduced. In such systems, entries of routing table (*e.g,.* IP address of neighbors) serves as logical links to other neighboring nodes and thus are regarded as edges of the overlay networks.

The complexity of an overlay network often determines the size of a P2P system that can be constructed. Likewise, the attainable performance of a P2P system is ultimately limited by the characteristics of the overlay network. Apparently, the selection of the overlay network is the first and also the most crucial step for the design of the new generation of P2P systems. Fig 1 displays our simple model. Each component will be clear after we define and descript the following terms

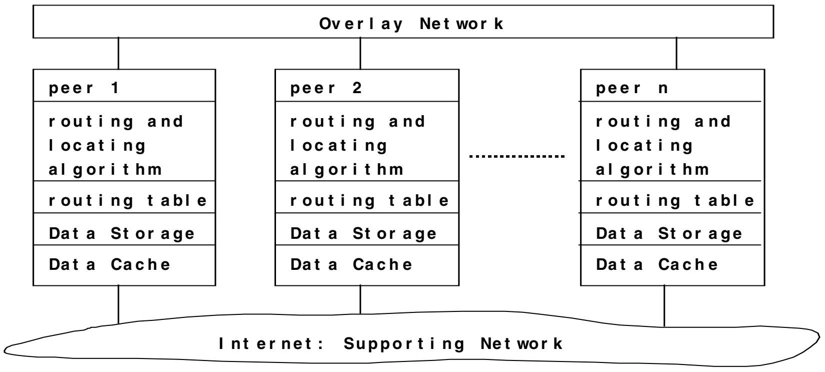

Fig. 1. A Generic Topological Model of P2P Systems

Data ID and Node ID: Each data (*e.g,.* a file) is assigned a globally unique identifier or a key that corresponds to the cryptographic hash of the file's textual name, the owner's public key, and/or a random salt. All keys are ordered linearly or circularly in 1-dimensional or d-dimensional key space. *Note that the ordered key space is the first necessary requirement for an overlay network.* Each node(*i.e.,* a peer) is also mapped into the same key space as data. It can be computed as a cryptographic hash of the node's IP address or public key.

Data Placement: Data ID and Node ID belong to the same space, but the number of nodes is often much less than the number of Data. Each node is responsible for storing a certain range of keys in the key space. Data is stored on the node that has the same or the closest ID as data ID, as implemented in Pastry and Viceroy. Different implementation might have a slightly changed policy. In Chord, data is stored in its successor node; this simplicity can bring about the convenience in self-organization. In CAN, all data of one zone is stored on the representative node of that zone.

Hash function: All data or nodes are mapped by hashing into a point in the same key space. The functionality of Hash should guarantee a uniform and random distribution of data or nodes within the key space. Consistent hashing[3] was proposed to guarantee the uniformity no matter how the input keys are intentionally chosen. SHA-1 function is used in practice as a good substitute for consistent hash function since producing a set of keys that collide under SHA-1 can be seen as inverting or decrypting the SHA-1 function that is believed to be hard to do[10].

Overlay Network: All live nodes are connected logically as an overlay network at application level with the help of routing table. Each entry of the routing table serves as a logical edge, usually containing the IP address of one neighbor. Recent P2P systems have adopted structured topologies such as ring, mesh and hypercube and showed advantages in scalability and lookup efficiency. Interestingly and importantly, none of them is a simple application of a traditional interconnection network. Each one embodies an insightful anatomy of an existing interconnection network and provides a resilient connection pattern for fault tolerance and self-organization. Seeking for better topologies for overlay network is becoming a new search topic. Recent work focus on the constant-degree overlay networks[1, 2, 4].

Routing Table: The size (or the number of entries) of the routing table is determined by the outgoing degree of the overlay network. It is the space complexity of an overlay network, *i.e.,* the memory overhead of each node. Moreover, it is not just a measure of the state required to do routing but also a measure of how much state needs to be adjusted when nodes join and leave. For this reason, interconnection networks with small degree (especially with constant degree) should be restudied as possible candidates for P2P overlay network.

Routing and Locating Algorithm: When a *lookup(key)* is issued or received, the *lookup* is routed through the overlay network to the home node responsible for that *key*. Each hop makes the most progress towards resolving the lookup. The notation of progress differs from algorithm to algorithm, but any routing algorithm should be convergent in the sense that each hop makes the distance between the current node ID and the home node ID becomes smaller and smaller. *Note that the convergent routing algorithm is the second necessary requirement for an overlay network.* The real distance performance is determined not only by the path length in key space but also by the communication delay of each hop in the Internet. Proximity in key space does not mean proximity in geological space**.** If the supporting network is represented as a graph in some regular way, the proximity problem can be studied in the form of embedding the overlay network into the supporting network with the smallest possible dilation and congestion. An adjustable Hash function might also be helpful if such a function could map nearby IP addresses into close values in key space. Other possbile solutions include the neighborhood set[8] and the landmark vector[11].

Self-organizing: To make P2P systems completely distributed, the routing table should be automatically reconfigured when a node joins or departs from the system. A node might leave abruptly due to crash or gracefully by informing its neighbors and submitting its keys. Systems are supposed to be small enough at beginning so that nodes could just exchange information directly to build the initial routing table. After that, nodes should have an automatic way to detect the node join or node leave to reconfigure their routing table state in such cases. Since not all nodes are alive in the key space and they often join and depart dynamically, the resilient topological connection pattern must ensure the routing table has full or enough number of neighbors. When one node fails, another node can substitute. *Note that self-organizing is the third necessary requirement for overlay network.*

Now we give the following thumb rule to judge if a network can be manipulated as an overlay network for P2P systems. This rule is empirical since we cannot give a mathematical proof.

A network can be manipulated as an overlay network if and only if the network meets the above-mentioned 3 requirements, *i.e.*, an ordered key space which is used to measure the distance between any two nodes, a convergent routing algorithm that sends a message closer to the destination at each step, and self-organizing ability that reconfigure the network topology when nodes join or depart.

3 Cubical Cycles: A Novel Overlay Network

We were first motivated by the open question in [7], *i.e.,* Can one achieve O(logN) path length or better with O(1) neighbors? If it were possible, how are some aspects of routing affected? Our recent investigation into CCC (cube-connected cycles) has shown that a modified CCC works well as an overlay network with O(1) neighbors and O(log N) path length. In addition to Koorde[2] and Viceroy[4], this is yet another new overlap network that has ever been found to meet the super-scalable requirement.

3.1 Resilient Connection Pattern

In a one-dimensional circular key space with $N = n \times 2^n$ nodes, each node is represented as $(k, a_n a_{n-1} \cdots\cdots a_1)$, where $1 \le k \le n$ and $a_i = 0$ or 1. $a_n a_{n-1} \cdots\cdots a_1$ is the cubical index and k is the cyclic index. Nodes are ordered according to the value of $a_n a_{n-1} \cdots\cdots a_1$. Nodes with the same $a_n a_{n-1} \cdots\cdots a_1$ are ordered again as a circular cycle but they are of equal valence to the outside. So, data with key $(k, a_n a_{n-1} \cdots\cdots a_1)$ will be put on the node that is first numerically closest to $a_n a_{n-1} \cdots\cdots a_1$ and then numerically closest to k. In the routing table, for example in Fig. 2, each node $(k, a_n \cdots a_k a_{k-1} \cdots a_1)$ has a cubical neighbor $(k-1, a_n \cdots a_k x \cdots x)$, where x denotes an arbitrary bit value, and two cyclic neighbors, $(k-1, a_n \cdots a_k b_{k-1} \cdots b_1)$ and $(k-1, a_n \cdots a_k c_{k-1} \cdots c_1)$, where

$$(k-1, a_n \cdots a_k b_{k-1} \cdots b_1) = Min\{\forall (k-1, a_n \cdots a_k y_{k-1} \cdots y_1) \mid y_{k-1} \cdots y_1 \ge a_{k-1} \cdots a_1\}$$

$$(k-1, a_n \cdots a_k c_{k-1} \cdots c_1) = Max\{\forall (k-1, a_n \cdots a_k y_{k-1} \cdots y_1) \mid y_{k-1} \cdots y_1 \le a_{k-1} \cdots a_1\}.$$

The cubical and cyclic neighbors allow us to change cubical index from left to right. It is easy to see that the network will be the traditional cube-connected cycles if all nodes are alive. We refer authors to [5] for topological properties of CCC. However, our connection pattern is resilient in the sense that even if many nodes are absent, the remaining nodes are still capable of being connected.

Node ID (5,101-1-1010)	
Routing table	
cubical neighbor: (4,101-0-xxxx)	
cyclic neighbor: (4,101-1-1100)	
cyclic neighbor: (4,101-1-0011)	
2 Leaf Sets(half smaller, half larger)	
Inside Leaf Set	
(3,101-1-1010)	(6,101-1-1010)
Outside Leaf Set	
(7,101-1-1001)	(6,101-1-1011)

Fig. 2. Routing table state of a hypothetical CCC node (5,101-1-1010). x indicates an arbitrary value, 0 or 1. Inside leaf set maintains the closest neighbors in the same cycle. Outside leaf set maintains the links to the previous and the next cycles.

We borrow the idea of the leaf set from Pastry[8]. But differently, each node $(k, a_n a_{n-1} \cdots\cdots a_1)$ maintains two leaf sets in addition to the routing table. The routing algorithm is greatly assisted by the leaf sets for the purpose of effectiveness and fault tolerance. The inside leaf set points to the predecessor and the successor in the same cycle. That is, nodes with the same cubical index are connected as a doubly linked cycle. The outside leaf set point to nodes with the highest cyclic index in the preceding cycle and the succeeding cycle. The outside leaf sets is important for self-organization such as node join, node departure and stabilization, as discussed in Section 3.3.

3.2 Routing Algorithm

The routing algorithm is used not only for looking up a data with data ID=$(l, b_n \cdots\cdots b_1)$, but also for joining a new node with node ID=$(l, b_n \cdots\cdots b_1)$. We suppose MSB is the most significant bit where $a_n \cdots\cdots a_1$ and $b_n \cdots\cdots b_1$ differs. The following routing algorithm of Cubic Cycles emulates the routing algorithm of CCC[5] from current node A=$(k, a_n \cdots\cdots a_1)$ to destination node B=$(l, b_n \cdots\cdots b_1)$, incorporating the resilient connection pattern of Cubic Cycles.

Routing Algorithm

1. (Termination Condition) If D is within the leaf sets, inside leaf set or outside leaf set, forward to the nearest node in the leaf sets. In particular, if D is within or closer to the local cycle than to the 2 neighboring cycles, forward to the predecessor or the successor in the same cycle, otherwise forward to one of the 2 neighboring cycle which is numerically closer to D.
2. (Ascending) If $k <$ MSB, change k until $k \geq$ MSB by using inside leaf and outside leaf set. In particular, the local cycle is first searched in index-increasing order for the first node with cyclic index $\geq$ MSB. If such a node is not found locally, the neighboring cycle is searched continuously until a node is found to have cyclic index $\geq$ MSB. Among 2 neighboring cycles, preceding cycle and succeeding cycle, the one that is numerically closer to D is always chosen.

3. (Descending) If $k \geq$ MSB, change $(a_n \cdots\cdots a_1)$ so that it shares a longer common prefix as $b_n \cdots\cdots b_1$ by using cubical links and cyclic links. In particular, if $a_k \neq b_k$, the cubical link is followed. If $a_k = b_k$, the cyclical link is followed. Between the 2 cyclical links, the one that is numerically closer to D is chosen. In the cases when the appropriate entry in the routing table is empty or the associated node is not reachable, the neighboring cycle closer to D is chosen.

The algorithm has the following features:

1. It is convergent and thus correct, because each step sends the message to a node that either shares a longer prefix with the destination than the current node, or shares as long a prefix with, but is numerically closer to the destination than the current node.
2. It is deadlock free. This feature derives directly from the convergence of the algorithm.
3. It can be easily augmented to increase the fault tolerance. When the cubical link or the cyclic link is empty or faulty, the message can be forward to neighbors in the leaf sets. Our simulation experiment has shown that 4-element leaf sets can greatly reduce the number of lookup failures. Discussion in this paper is based on the 2-element leaf sets.
4. It is executed in O(log N) hops since each of 3 steps is bounded by O(log N) hops. The detailed analysis of probability is omitted due to the space limitation. Interested authors may write to get the full version of this paper.

3.3 Self-Organization

P2P systems are notoriously dynamic in the sense that nodes are frequently joining in and departing from the network. Accordingly the overlay network should be capable of self-organization without using a centralized database like Napster or a flooding-based multicast like Gnutella. When a node joins and departs, only local or a small part of nodes are disrupted.

Node join: When a new node joins, it needs to initialize its routing state tables and then inform other related nodes of its presence. Like other systems, we assume the new node knows initially about a live node A. Let us assume the first contact node is A=$(k, a_n \cdots\cdots a_1)$ and the new node is X=$(l, b_n \cdots\cdots b_1)$. By the routing algorithm discussed in section 3.2, the contact node A routes the join message to the existing node Z whose ID is numerically closest to the ID of X. Moreover, Z's leaf sets are the basis for X's leaf sets. In particular, the following two cases are considered:

1. If X and Z are in the same cycle, Z's outside leaf set directly becomes X's outside leaf set. Z's inside leaf set needs a slight modification to become X's inside leaf set, considering that Z should be among X's inside leaf set. Note that the inside leaf set points to one immediate predecessor and one immediate successor in the same the cycle. For fault tolerance, multiple predecessors and multiple successors might be pointed by the inside leaf set.
2. If X and Z are not in the same cycle, this is because that X is the first node in its cycle. Z's outside leaf set need a slight modification to become X's outside leaf

set, considering that Z should be among X's outside leaf set. Note that the outside leaf set points to one immediate preceding cycle and one immediate succeeding cycle. For fault tolerance, multiple preceding cycles and multiple succeeding cycles might be pointed by the outside leaf set. Z's inside leaf set is of no use for X's inside leaf set and instead all entries of X's inside leaf set refers to X itself since X is the only member in the cycle.

Now we consider the initialization of 3 neighbors in routing table. If we can find a node with ID= $(l, b_n \cdots b_l x \cdots x)$ where x represents any bit value, 0 or 1, we can borrow the cubical neighbor from this node to X. We start from one member of X's outside leaf set since X's outside leaf set members share the longest prefix with X and thus are close to such a required node $(l, b_n \cdots b_l x \cdots x)$. The possibility of such an existing required node depends on the value of l or the length of string $x \cdots x$. If there is no such node at all, it is usually because l is very small. In this case we just leave the cubical neighbor entry blank, as we have seen in the routing algorithm that it doesn't affect the routing too much. A more sophisticated scheme is to find such a required node during the routing path when inserting node X, and a close investigation shows that such a node exists with a high probability although such a node can not always be find in any case. Similarly for the 2 cyclic neighbors, we first search for $(l-1, b_n \cdots\cdots b_1)$ in the same cycle. If it doesn't exist, we continue to search in the next neighboring cycle that is reachable from the outside leaf set until we find such required cyclic neighbor on either side of node X.

Node departure: Nodes might leave the network abruptly or break down suddenly. Routing tables should be dynamically updated to reflect the newest network situation. Both lazy algorithm and greedy algorithm can be used. The lazy algorithm updates the routing table only when the entry is found to be obsolete. The greedy algorithm sends the hello message to all neighbors periodically and thus incurs some unnecessary messages. The initializing algorithm discussed above for node join can also be used for node departure. In particular, if a neighbor is found to be obsolete, the initializing algorithm is started to find a new neighbor to replace the obsolete one.

4 Conclusion

Our system bears the most resemblance with Viceroy[4]. Table 1 lists the difference between two systems. Our recent simulation[9] has shown the obvious performance advantage of cubic cycles over Viceroy.

Table 1. Comparison of Cubical Cycles and Viceroy

	Viceroy	Cubic Cycles
Topology	Butterfly	CCC
Node number	2^n	$n \times 2^n$
Fault tolerance	Difficult to be augmented	Easy by using larger leaf set
Climbing of levels	To the top level	Not to the top level

Hundreds of networks have been proposed in the past decade or so. Only few of them are proposed and modified as the overlay networks for P2P systems. We believe that more interconnection networks will be studied as promising candidates for P2P systems. Meanwhile, the new generation of topology-aware P2P systems will ignite a new chance for the study of Interconnection network.

As should be clear by our discussion, this paper is not about a finished works. We hope that presenting such a discussion will promote synergy between research groups in this area and help clarify some of the underlying issues. We expect in the near future to answer such a question: which kinds of networks topologies, after being modified, can be used for the overlay network and which cannot.

Acknowledgement. The work is partly supported by China NSF grant (No. 60073029), China 973 project (No. 2002CB312002), TRAPOYT award of China Ministry of Education, and US NSF grant (No. ACI-0203592).

References

1. G. Chen and C.Z. Xu, Topological Analysis of P2P Systems, Proc. of 1st Int. Workshop on Grid and Cooperative Computing, Hainan, China, December 2002.
2. M.F. Kaashoek and D.R. Karger, Koorde: A simple degree-optimal distributed hash table, Proc. of 2nd Int. Workshop on Peer-to-Peer Systems, March 2003.
3. D. Karger, *et al,*. Consistent hashing and random trees: distributed cashing protocols for relieving hot spots on the World Wide. In Proc. ACM STOC'97, El Paso, TX, May 1997.
4. D. Malkhi, M. Naor and D. Ratajczak, Viceroy: A scalable and dynamic emulation of the butterfly, PODC2002.
5. F.P. Preparata and J. Vuillemin, The cube-connected cycles: A versatile network for parallel computation. CACM, 24(5):300--309, 1981.
6. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, A scalable content-addressable network, Proc. ACM SIGCOMM'01, San Diego, CA, Aug. 2001.
7. S. Ratnasamy, S. Shenker, and I. Stoica, Routing algorithms for DHTs: Some open questions, Proc. of 1st Int. Workshop on Peer-to-Peer Systems, March 2002.
8. Rowstron and P. Druschel, Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems, Proc. of 18th IFIP/ACM CDSP'01, Nov. 2001.
9. H.Y. Shen, C.Z. Xu and G. Chen, Cycloid: A consistent-degree and Location efficient P2P overlay network, technical report ECE-03-08, Wayne State University, 2003.
10. Stoica, *et al*, Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proc. ACM SIGCOMM'0*1, San Diego, CA, Aug. 2001.
11. Zhichen Xu, Chunqiang Tang and Zheng Zhang, Building Topology-Aware Overlays Using Global Soft-State, ICDCS03.
12. B.Y. Zhao, *et al,* Tapestry: An infrastructure for fault-resilient wide-area location and routing. Technical Report UCB//CSD-01-1141, U. C. Berkeley, April 2001.