ISSN:0975-9646

Route Redirection in Structured Network Using Chord

A.Joyce, E.Nagarajan
Department of Computer Science and Engineering,
Sathyabama University,
Chennai,Tamilnadu,Iindia

Abstract

In a structured peer-to-peer (p2p) network, Distributed Hash Tables (DHT) are maintained by nodes to store information and to route the packets. But sometimes a node can be attacked and so it may deny the DHT key access or misroute the packets. Replica Placement is a key mechanism to overcome these problems by placing replica for nodes in p2p network. MAXDISJOINT replica placement strategy creates disjoint routes that redirects the routes from source to destination nodes. Another replica placement strategy is Spaced replica placement where replica identifiers are separated by uniform spacing. Our proposed work is to make efficient placement of replicas by combining MaxDisjoint and Spaced replica placement and thereby reducing its implementation cost.

Keywords – distributed hash tables, peer-to-peer networks, replica placement

1. Introduction

1.1 Peer-To-Peer Networks

A p2p network is a type of decentralized and distributed network architecture. Nodes in the network are called peers which act as both client and server. The p2p networks can be classified as

- Unstructured
- Structured

An unstructured p2p network does not have a particular structure on the overlay network as shown in Fig.1. It is formed by nodes which randomly form connections to each other. The main drawback lies in the structure of the network. When a peer wants to find a packet in a network, it has to flood the query through the network which causes high traffic and more CPU power.

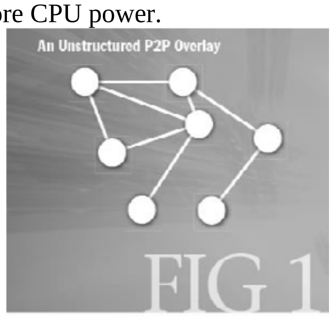

Fig.1 Unstructured p2p network

The Structured p2p network follows a specific topology as shown in Fig.2 and any node can efficiently search the network for a file. Generally it can be implemented using DHT where a hashing is used to assign ownership of each file to a particular peer.

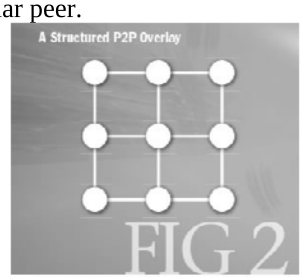

Fig.2 Structured p2p network

This enables peers to search for resources on network using a hash table; that is *(key, value)* pairs are stored in DHT and any participating node can efficiently retrieve the value associated with a given key.

1.2 Distributed Hash Table

A DHT is a class of decentralized distributed system as shown in Fig.3 that provides a lookup service similar to a hash table; *(key, value)* pairs are entered in a DHT and any node in the network can retrieve value for a given key.

Responsibility for maintaining the mapping from keys to values is distributed among the nodes in such a way that a change in the set of participants causes minimal amount of disruption. This allows a DHT to scale to extremely large number of nodes and to handle continual node arrivals, departures, and failures.

DHT's emphasizes the following properties,

1) autonomy and decentralization
   - The nodes collectively form the system without any central coordination.
2) Fault Tolerance
   - The system should be reliable even with nodes continuously joining, leaving, and failing.
3) Scalability
   - The system should function efficiently even with thousands of nodes.

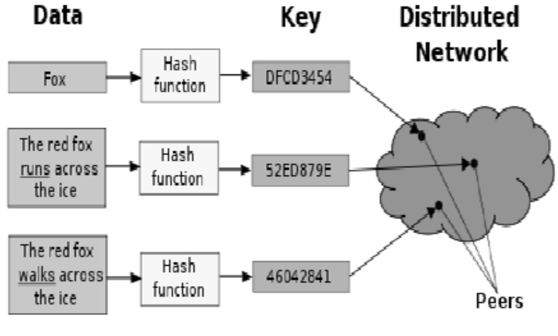

Fig.3 Distributed Hash Tables

2. Related Works

[4]discusses the initial replica placement strategies say neighborhood based and random replica placement which does not create efficient replicas. It proposes an equally spaced replica main placement scheme. The main limitation in this paper is it does not reach the expected level of routing robustness since no routing schemes had been incorporated.

[3]shows the number and placement of replicas necessary to produce disjoint routes using MaxDisjoint replica placement, a new replica placement strategy along with neighbor set routing. But the complexity in neighbor set routing is the inability to create maximum disjoint routes.

[1]explains the number and placement of replicas necessary to produce disjoint routes using the MaxDisjoint replica placement and tree-based routing. The practical limitation of using MaxDisjoint algorithm is the high cost of implementing replicas in a large p2p networks.

3. Existing Methodology

3.1 MaxDisjoint Algorithm

The algorithm assigns each replica an identifier, which is used to determine its placement. The algorithm is as follows,

Input:

- N, the identifier space size of the DHT;
- B, the branching factor;
- d, the desired number of disjoint routes;

Algorithm:

1. To create *d* disjoint routes, replicas are placed in $m+1$ rounds,
   where $m = \left\lfloor \frac{d-1}{B-1} \right\rfloor$
2. Each round consists of $B-1$ steps except for the final round, which consists of n steps, where $n = (d-1) \bmod (B-1)$.
3. In the i^{th} round, B^{i-1} replicas are placed at equally spaced locations over the entire identifier space at each step.
4. In step j of round i, the replica locations are given by:

$$R_{i,j} = \{k_{i,j}, k_{i,j} + s_i, k_{i,j} + 2s_i, \ldots\ldots k_{i,j} + (B^{i-1} - 1)s_i\} \pmod N$$

where $k_{i,j} = k + j\frac{N}{B^i}$

3.2 MaxDisjoint Replica Placement

To replicate an object with id 101, node 121 routes to this object through the routing table entry marked *10x*. Fig.4 shows the MaxDisjoint replica placement for object 101 using chord ring. Suppose we replicate the object with the id 111 to target the routing table entry *11x*. This approach creates an additional disjoint route for any lookups for object 101 originating at node 121. One route is forwarded through the entry *10x* and the other is forwarded through the entry *11x*.

However, consider another source node 221. This node routes to the object 101 and 111 through the same entry marked *1xx* and therefore does not gain an additional disjoint route. To move toward a more effective approach, consider all the replicas of object 101 that would create an additional disjoint route for node 121. These are 001, 111, 120, 122, 123, 131, 201, and 301. There are total of nine possible disjoint routes which is the number of routing table entries for node 121. Of these replicas, there are only three that can create an additional disjoint route for every possible source node: 001, 201, and 301. These replicas create disjoint routes by targeting entries in the first row of the routing table.

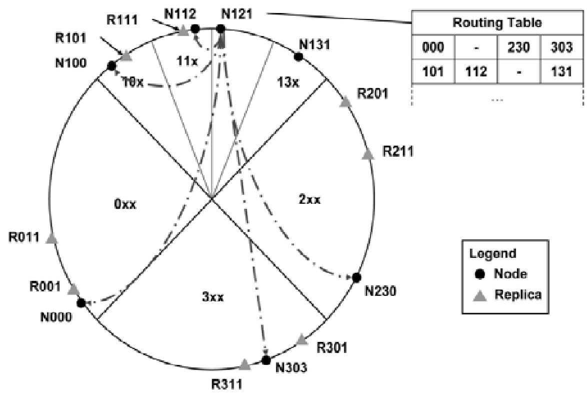

Fig.4 MaxDisjoint replica placement for object 101 with N=64, B=4

In general, replicas are simply placed such that replicas exist in different parts of each node's partitioning of the id space. The main drawback here is the cost of implementing replicas according to the created number of disjoint routes.

4. Proposed Methodology

Our paper suggests that, instead of placing all the replicas in the created disjoint routes, we can further reduce the total number of replicas by using the spaced replica placement strategy. It provides a spacing degree according to which replicas which are farther from the destination node can be taken off.

And also instead of using SHA-1 algorithm for key generation which is used in existing methodology, MD5 is used since it generates the key value for the node identifiers very faster than SHA-1.

4.1 Chord Protocol

The DHT table for nodes in p2p network is implemented using chord protocol. It is a protocol and algorithm for a p2p DHT. Chord specifies how keys are assigned to nodes, and how a node can discover the value for a given key by first locating the node responsible for that key.

IDs and keys are assigned an *m*-bit identifier using consistent hashing. The MD5 algorithm is the base for consistent hashing. Using the chord lookup protocol, node keys are arranged in a circle that has at most 2^m nodes as shown in Fig.5. The circle can have IDs/Keys ranging from $0 - 2^{m-1}$.

Each node has a successor and a predecessor. The successor to node (or key) is the next node in the identifier circle in a clockwise direction. The predecessor is counter-clockwise. Since the successor or predecessor node may disappear from network because of failure or departure, each node records a whole segment of the circle adjacent to it. A logical ring with positions numbered $0 - 2^{m-1}$ is formed among nodes. Key K is assigned to node successor (k), which is the node whose identifier is equal to or follows the identifier of k. If there are N nodes and K keys, then each node is responsible for roughly K/N keys.

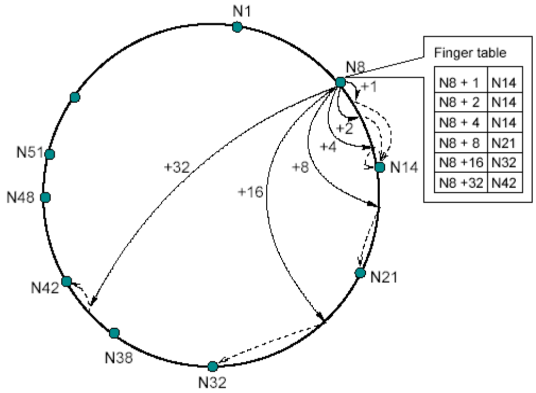

Fig.5 Node localization in p2p using chord ring along with finger table for node 8.

For nodes, the identifier is the hash of node's IP address. For Keys, it is the hash of a keyword, such as a filename. When a new node joins or leaves the network, responsibility for *O(K/N)* keys changes hands. Chord requires each node to keep a finger table as shown in Fig.5 containing up to m entries. The i^{th} entry of node n will contain the address of the successor $((n+2^{i-1}) \bmod 2^m)$.

4.2 SPACED REPLICA PLACEMENT

With MaxDisjoint, the additional replicas are placed at the locations determined by performing another step in the placement algorithm leaving the existing replicas in their current locations.

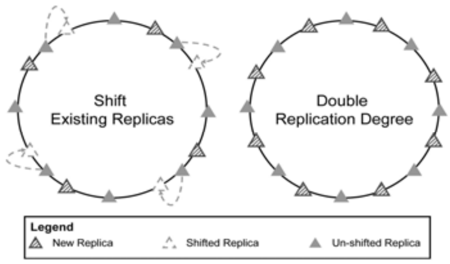

Fig.6 Spaced replica placement

Spaced replica placement can be done in following two ways,

→ Shift existing replicas
→ Double replication degree

As shown in Fig.6, with shift existing replicas, the replicas can shift to new location and sometimes the unwanted replicas can be taken off from the ring. The second option of doubling the replication increases the cost of shifting the existing replicas which may be a burden.

5. CONCLUSION

In this paper, the replica placement is done with MaxDisjoint along with another mechanism for managing the replica placement called spaced replica placement. MaxDisjoint has a stronger impact on routing robustness than other replica placement strategies.

However, when the mechanisms are combined, substantial benefit is gained especially with the implementation cost of replicas. Therefore, using two or more route diversity mechanisms, like MaxDisjoint replica placement and spaced replica placement, can have a positive impact on routing robustness.

This replica placement is a simple and special case of MaxDisjoint and is an obvious choice for both unstructured and structured p2p networks.

REFERENCES

[1] Cyrus Harvesf and Douglas M.Blough, "Replica Placement for Route Diversity in Tree-based Routing Distributed Hash Tables," Proc. IEEE Int'l Conf. Dependable and Secure Computing, vol.8, no.3, May/June 2011
[2] M.Castro, P.Druschel, A.Ganesh, A. Rowstron, and D.Wallach, "Secure routing for structured peer-to-peer overlay networks," Proc. Symp. Operating Systems Design and Implementation (OSDI'02), pp.299-314, 2002.
[3] C. Harvesf and D.M. Blough, "The Design and Evaluation Techniques for Route Diversity in Distributed Hash Tables," Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P '07), pp.237-238, 2007.
[4] C. Harvesf and D.M. Blough, "The Effect of Replica Placement on Routing Robustness in Distributed Hash Tables," Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P '06), pp.57-66, 2006