International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 12 (2015) pp. 30329-30335

http://www.ripublication.com

Replica Consistency Maintenance In Unstructured Peer To Peer Network Using Push and Pull Approach

A. Sai Pavanee Geervani
Dept. of CSE, Sree Vidyanikethan Engineering College
NH 205, Sree Sainath Nagar,
A. Rangampet, Tirupati, Andhra Pradesh 517102
N Papanna
Dept. of CSE, Sree Vidyanikethan Engineering College
NH 205, Sree Sainath Nagar,
A. Rangampet, Tirupati, Andhra Pradesh 517102

Abstract

Un-Structured peer to peer network is an highly dynamic network where there is no restriction on network topology or resource placement. The connections between nodes in the network are formed by nodes randomly joining and leaving the network .In order to make the file more available in an environment where the nodes can join or leave at any point replication strategies are employed. Maintaining consistency of replicas by keeping them up to date to all the changes in such a dynamic network is profoundly challenging. Distribution of replicas plays an important role in the process of replicating the file. The current methodologies for maintaining replicas do not consider the significance of distribution of replicas in maintaining consistency. In the proposed approach the distribution of replicas is controlled by modeling the nodes from a time and space perspective. The consistency of replicas is maintained using a push and pull approach. The main focus is to keep up consistency using less number of update messages and increase the update success rate. This approach is simulated using ns2 and results show that this approach uses less number of update messages.

Keywords: unstructured peer to peer network , replica consistency, time and space perspective, replica distribution

Introduction

Peer-to-peer (**P2P**) is a type of network formed by two or more functionally equivalent nodes. This kind of network is mainly used for resource sharing. An node can join or leave the network at any time. Distinct to the client/server architecture it

does not require any centralized node that direct's resources to serve other nodes. A node in a P2P network plays dual role of both a client that consumes resources and a server that provides resources. This dissimilarity helps to maintain the network stable by removing the single point of failure problem existed in client/server systems. The profoundly distributed and self-governing nature of P2P makes it more suitable for more number of applications.

P2P network can be broadly classified as structured P2P and unstructured P2P. A structured P2P is one in which the nodes are rigidly connected and follow a particular structure while forming the network. A structured P2P network, represented by DHT network, strictly defines the mapping relationship between a peer and its managed file indices, and thus enabling a high retrieval success rate. The structured nature of network makes it less resistant towards churn problem.

An unstructured P2P network is loosely connected and does not follow any particular structure to build the network. This unstructured nature makes it resistant to high churn rates. The phenomenon where large number of nodes joining or leaving the network at the same time is referred as churn problem.

As P2P networks are widely used for resource sharing in order to increase the resource availability file replication techniques are employed. In a highly dynamic network like unstructured P2P network with readable and writable files maintaining the consistency of replica's is not an easy task as there are no specific nodes to maintain file indices.

Existing approaches for maintaining consistency have the shortcomings of requiring more number of update messages and low update success rate. And also does not consider the distribution of replicas to maintain consistency.

Push and Pull approach is used to solve the consistency maintenance problem. Using the push approach the update message is propagated to the target nodes i.e nodes consisting the replica. Pull approach is used to obtain the update content from the source node that initiated the update message.

Related Work:

ACO Based Approach

In this approach [1] the consistency maintenance problem is tackled based on Ant Colony Optimization algorithm. Ant algorithm is mainly used to find optimal paths in an network based on an ants behavior while searching for food. After an ant finds a food resource while returning to its nest it leaves a trail called PHEROMONE trace which can be used by other ants to find the food resource successfully and easily.

Though the actions and amount of work done by an single ant is simple. The amount of work done by multiple ants by cooperating whith each other is huge and can solve complicated tasks. In such environment loss of an single ant does not effect the whole task. These features make ACO more suitable for maintaining replicas in unstructured P2P systems with high churn rates.

In the same manner whenever a node initiates an update message the node creates an ant i.e an update message. This ant is forwarded to the target node i.e Food

Resource consisting the replica . In the return process while the ant is returning to the initial node i.e Nest after updating the replica the Pheromone trace is updated. The churn problem is also tackled in the return process.

The entire process can be consolidated into the following steps:

Creation of ant: When an update occurs on a file at a node the node create a ant containing the content of the update message which has to be forwarded in the network to update the replicas. The ant also consist of additional information like source node id, target node id , time stamp, TTL(time to live), Tabulist the path in which the ant is forwarded.

Forwarding of ant : After an ant is created the source node searches its neighbor node for pheromone trace of the file on which the update occurred. If pheromone trace is available it forwards the ant in that direction if that node is not present in the tabulist. Else it will randomly select an node from its neighbor.

Processing ant: When ant arrives at a node that node processes the ant i.e using the update content of the ant the file is updated. after updation it forwards the ant .If the time to live value becomes zero after updating at that node the ant is returned to the source node.

Return of ant: When an ant reaches the last node to be updated or when the TTL value becomes zero the return process is activated. In the return process the pheromone trace is updated along the path present in the tabulist. If the ant reaches the source node the process ends.

This approach results in less number of update messages and resistant towards high churn rates. But there is no guarantee that the ant reaches the exact required node accurately.

Update propagation through replica chain in decentralized and unstructured P2P systems

This is an replica-chain based replica consistency maintenance approach[2], where each node forwards an update message to its k neighbors along the replica chain, and thereby reducing the number of redundant update messages.

For each file a logical replica chain is created which contains all replica peers (RPs) . RPs are peers that have replicas of the file. Each RP obtains partial knowledge of the bi-directional chain by maintaining a list of information about k nearest RPs, called probe peers, in each direction. When an RP initiates an update, it pushes the update to all possible online (active) RPs through the replica chain. A reconnected RP pulls an online RP to synchronize the replica status and the information of the probe peers.

However, since each node must maintain a replica-chain for each shared file, results in too much maintenance cost.

An optimized strategy for update path selection in unstructured P2P systems

This proposes a concept of the spanning graph [5], by which a node could share the up-date path information as far as possible by using clone, variation and crossover operations for the update paths so as to reduce the redundant propagations of the update messages. A repeated update strategy is presented to cope with the churn

problem so as to maintain replica consistency as far as possible. However, this strategy needs a delayed period to conduct a crossover operation, and what is more it is difficult to predict the length of a delayed period in a specific P2P network.

Proposed Approach

In the proposed approach replica's consistency in unstructured P2P network is maintained using an push and pull based approach. In this approach first the nodes are modelled based on time and space perspective order to control the distribution of the replicas making it more viable for maintaining the replicas in such highly dynamic environments. The consistency is maintained using an push and pull approach.

To define the time and space perspective of a node we need to consider service time and service scope. Service time is for how long a node is online to serve the resources. Service scope refers to hoe far a node can propagate its services in its neighborhood . In time and space perspective of a node we consider the interaction between two nodes. One is the node that serves the resource and the other node is one which consumes the resources.

Time distance is difference of the available time of node x and available time of node y. If the difference is more then the probability that both nodes are available at the same time is low and might no be able to effectively share resources. If the difference is less then it is more likely that they are available at the same time and can effectively share resources.

Space distance is the number of hops taken for a request message to propagate between two nodes.

Every node occupies certain region of time and space. The intersection of the occupied time and occupied space of an node creates an effective region in which the node can serve effectively is called the time and space perspective of a node.

Push and Pull approach: The node on which the update is initiated is called the source node. The nodes that carry the replicas are called the destination nodes on which the update should take place.

For every file f we maintain L_r (local reference) and F_r (foreign reference) . The local reference contains all the nodes that are in the same space and time region as the source node. The foreign reference contains all the nodes that are in different space and time region .

The file also maintains a counter called the U_{cr} (update counter) which is incremented every time after the update has taken place. For all the foreign nodes TTR (time to refresh) value is set in order to pull the update content from source node from time to time. When an update has occurred at a node then the push (f_{update}) process is trigerred.

Push Algorithm

1. push (f_{update})
2. Begin
3. check the L_r
4. Obtain peer id's of the nodes in the $L_{r.}$
5. push the update message to the obtained peer's.

6. U_{cr} ++ at the updated nodes.
7. check the F_r
8. Obtain peer id's of the nodes in the F_r.
9. push the update intimation to the obtained peer ids
10. if (U_{cr}(current node) = U_{cr}(source node)
11. set ttr to low
12. else
13.set ttr to high
14.End

At the foreign nodes after receiving the update intimation pull(f_{update}) process is triggered.

Pull Algorithm

1. Pull(f_{update})
2. Begin
3. Obtain source peer id.
4. pull the update message from source peer.
5. U_{cr} ++
6.End.

Example

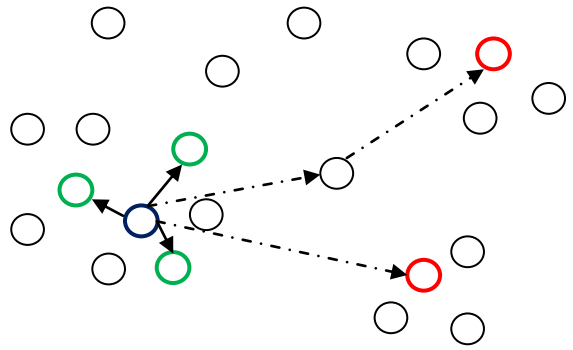


Figure: Pushing An Update Message.

The blue node represents the source node on which the update has occurred. The green nodes represent the local nodes present in the L_r that contain replicas of file updated on the source node. The red nodes represent the foreign nodes present in F_r that contain replicas of file updated on the source node.

The local nodes are in same space and time region as the source node. The foreign nodes are in different time and space zone compared to that of the source node.

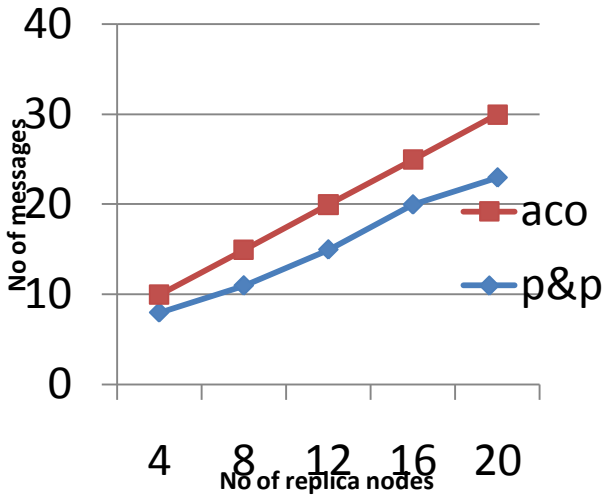
After the update has occurred at the source node the L_r is checked for local nodes and the update message is forwarded to their peer ids. After updating the replicas the U_{cr} is incremented at the respective nodes. After the local nodes are updated F_r is checked and the foreign nodes peer ids are obtained. Update intimation message is sent to the respective peer ids. The U_{cr} value at the present node is compared with the U_{cr} value of the source node if the difference is high ttr value is set high if it equal the ttr value is set low.

At the Foreign nodes after receiving the update intimation the source node id is obtained. Based on the ttr value the update message is pulled and the replica is updated. After updating replica the U_{cr} is incremented.

Simulation Model

The scenario of maintaining consistency is simulated using ns2 . The efficiency this approach is calculated by considering the number of replica nodes and number of update messages required to update them.

Total of 40 nodes are considered among which the total number of replica nodes are twenty.



As the number of nodes to be updated increase the number of update messages required to update them also increase but when compared to the existing ant based approach the proposed push and pull approach requires less number of update messages.

Conclusion

In the proposed approach replicas are kept up to date by using an push and pull approach. By using the L_r and F_r to identify the destination nodes carrying the replicas the accuracy of successfully updating the replicas is high compared to the existing approach.

By modeling the nodes in time and space perspective we can facilitate replica consistency even easily.

References

[1] Xianfu Meng , Changyuan Zhang" An ant colony model based replica consistency maintenance in unstructured peer to peer network" in Elsevier Computer Networks 62 (2014) 1–11 .

[2] K. Xie, D.F. Zhang, G.G. Xie, J.G. Wen, A trace label based consistency maintenance algorithm in unstructured P2P systems, J. Software 18 (1) (2007) 105–116.

[3] X.F. Meng, Y.L. Wang, Y.L. Ding, "An optimized strategy for update path selection in unstructured P2P networks," Comput. Networks 56 (2012) 3744–3755.