The Method of Maintaining Data Consistency in Allocating Resources for the P2P Network Model

Ha Huy Cuong Nguyen[1(✉)], Hong Minh Nguyen[2], and Trung Son Doan[3]

[1] Department of Information Technology, Quangnam University, Tam Ky, Vietnam
nguyenhahuycuong@gmail.com
[2] People's Security University, Ho Chi Minh City, Vietnam
hongminhnguyen1982@gmail.com
[3] Faculty of Information Technology and Security, People' Security Academy, Hanoi, Vietnam

Abstract. Maintaining consistency of mutable data replication in a Peer-to-Peer (P2P) system is a fundamental challenge. That is the frequency problem of structural change in P2P system i.e. how often does the actual structure of the tree change. There are several models which exhibit better characteristics for updating structure. However, it is worth separating the changes of structure from effect of data. An novel approach for balanced consistency maintenance in structured tree in P2P systems supporting heterogeneous distributed platforms is presented in this article. Nodes of each object are organized into a tree structure for disseminating update processes. An analytical model to optimize is showed theoretically. Several simulations for experiment results will be carried out in the dynamic network conditions.

Keywords: P2P · Distributed system · Data consistency · Multi-copy

1 Introduction

P2P networking systems use an application level organization of the overlay network for flexibly sharing resources (e.g., files and multimedia documents) which are stored across a wide range of computer networks. Structured P2P systems have been effectively designed for many data applications [1]. Each computer of P2P system has both roles client and server at the same time. The advantages of P2P approach enable to improve data availability, fault tolerance, and scalability for dynamic content sharing. Because dynamic characteristics of network combined with diverse application require heterogeneously distributed resource hardware, it emanates challenges for P2P consistency management [2,3]. The communication and exchange of information is performed directly between the participating peers and the relationships between the nodes which are heterogeneous and distributed, in the network are not equal. Some messages will be lost

P. Cong Vinh et al. (Eds.): ICCASA 2017/ICTCC 2017, LNICST 217, pp. 155–165, 2018.
https://doi.org/10.1007/978-3-319-77818-1_15

or attenuated when transmitted over the network. P2P networking systems differ from other Internet applications, hereby some user tends to share data from a large number of the more central virtual machines and Web Servers. Several well-known P2P networking systems P2P that are able to share files such as CAN, Chord, Gnutella, Free-net, Napster, Pastry, and Tapestry [3,13,14]. P2P networking systems, more importantly, allow the location of arbitrary data objects. Applying sequential consistency maintenance leads to prohibitively prolonging synchronization delays due to a large number –even unreliable– of peer. The "deadlock" may occur when a crashed heterogeneous nodes causes other nodes to wait forever. At the other extreme, eventual consistency are allowed to concurrently update their local copy. And it only requires that all heterogeneous nodes become identical after a long enough failure-free and update-free interval [2,4,5].

An important characteristic of P2P networking systems is their ability to provide a large storage, CPU power coupled with other resources while imposing a low cost for scalability, and for entry into and exit from the heterogeneously distributed platform [6]. The participation and exit of various nodes, as well as insertion and deletion of objects, are dynamic. The P2P networking systems exhibit a high level of self-organization and are able to operate efficiently despite the lack of any heterogeneous infrastructure [6]. This model requires that if a heterogeneous node wants to enjoy the services which is provided by other nodes, that node should provide services to other heterogeneous nodes.

In the past, both distributed computing and parallel computing have been commonly used for large-scale computation. Internet applications for global sharing like Napster, Gnutella and FreeNet have recently gained popularity based on P2P architecture. Several research projects aim at constructing other types of P2P applications. P2P systems can be characterized as distributed systems in which all nodes have identical capabilities and responsibilities and all communication is symmetric [2–6].

Several key challenges in P2P networking systems include: object storage mechanisms, object look-up in an efficient manner, retrieval in a scalable manner, dynamic reconfiguration as well as objects joining and leaving the network randomly, replication strategies to expedite object search, tradeoffs between object size latency and table sizes, anonymity, privacy, and security.

This article presents model structure based on tree broadcast, a generic decentralized P2P content location, and routing system for very large, self-configuring overlay networks of nodes which are connected via the Internet. The problems of routing and locating content in large scale P2P networking systems are considered in looking for efficient algorithms. Particularly, an algorithm Ary-Construction, constructing tree update to maintain consistency in structured tree of P2P networking systems, is proposed in this work.

The work is organized as follows: Sect. 2 summarizes briefly related works, Sect. 3 describes existing models, Sect. 4 proposes analytical model for resource allocation in heterogeneous distributed platforms, Sect. 5 describes algorithm in detail, Sect. 6 shows experiments and results and finally Sect. 7 presents conclusions and suggestions for future work.

2 Related Work

In [7], the problem of partitioning web-link graph for web ranking in P2P is formulated as a minimal cut-set with density balanced partitioning. The problem is proved to be an NP-Hard, by reducing to the minimum bi section problem [4,7]. In P2P based PageRank [5,7], each computational peer contains a local web-link graph and its PageRank in computed locally. P2P is a viable choice to address such limitation. To be able to compute the global ranking, a special node, so called word-node, is constructed to store the linkpage information of the other peers. In [8], toward authentication between familiar peer in P2P networking systems. A secure environment can only be achieved, when peers are sure, that they are communicating with partner desired.

There are two classes of P2P overlay networks: Structured and Unstructured. The technical meaning of structured P2P, strong consistency is provided by organizing heterogeneous nodes to an auxiliary structure on top of the overlay for update propagation [6]. Examples include the tree structure in SCOPE [9], the two-tiered structure in OceanStore, and a hybrid of tree and two-tiered structure in build a tree by recursively partitioning the identifier space and selecting a representative node as a tree node for each partition [4]. Only leaf nodes store object copies, all the intermediate nodes only store information of the tree structure in their sub-space. Nodes who may not be interested in the object are in the object's update dissemination tree, which adds unnecessary overhead of maintaining the tree from node failures. To the contrary, SCOPE, OceanStore constructs the tree structure dissemination by only involving heterogeneous nodes who are interested in the object which greatly reduce the overhead of maintenance and update propagation. The problem also efficiently builds the tree structure dissemination and binary tree decomposition to make it balanced and robust under the node churn.

In unstructured P2P systems, mainly two types of bounded consistency are provided rumor spreading and replica chain are used to ensure a certain probability of an update being received. The probability is tuned by adjusting the redundancy degree in propagating an update to balance the communication overhead with the consistency strictness. The message broadcast is an unstructured byte array that is delivered to all members of the group through the send method probabilistic bounded consistency. In previous work [10], propose method to be the best algorithm for constructing unstructured P2P graphs suitable for heterogeneous random selection. Here is a brief overview of other unstructured approaches. In extends Gnutella [14] by making both graph construction and query-resolution sensitive to node capacities. High-capacity nodes here have higher degrees, and are more likely to be traversed by random walks. While Swaplinks shares these two features with GIA, Swaplinks exhibits more accurate control over degree and probability of selection. Other examples of unstructured graph construction schemes include Araneola, an approach by Law and Siu, and Jianming Fu et al. None of these take node heterogeneity into account. In [9] mechanism can be used as a random node selection primitive, but as was the case with the previously mentioned schemes, does not take into account node

heterogeneity. In systems P2P a node sequentially contacts the cached ancestors to avoid conflict relocation decisions while in a node uses multiple side links in parallel to retrieve the lost packets. Previous works have proposed continuous models for consistency maintenance, which have been extended by a maintain consistency model in [1] for P2P applications. In that reference, the usage of core technique for maintaining consistency is a hybrid of push and pull methods, which are also used to provide application tailored cache consistency in [9]. Although each node can specify its consistency requirement, the model in [14] makes each node perform the strongest consistency maintenance from all its descendant nodes in the overlay replica hierarchy. Thus, the overhead of maintaining consistency at a node is not reduced even it only requires a weak consistency as long as one of its descendant nodes requires a strong consistency.

3 The Model Structure Tree Broadcasts Resource Allocation in Heterogeneous Distributed Platforms

The model structure tree broadcasts operation stores objects at a user defined number of diverse nodes. An operation reliably retrieves a copy of the requested object if one exists. The objects are retrieved from the live node, there is an issue how does look-up among the nodes storing the object. The mode P2P is measured here in terms of a scalar, application defined metric, such as the numbers of network hops or network delay. The model is completely decentralized, is self-configuring, it automatically adapts to the arrival, departure, and failure of nodes and it is scalable. The number of nodes traversed, as well as the number of messages exchanged while routing a client request is at most algorithms in the total number of nodes in the system.

P2P computing is carried out over application layer network where in all interactions among the processors are at a "peer" level, without any hierarchy among the processors. Thus, all processors are equal and play a symmetric role in the computation. P2P computing arose as a paradigm shift from client-server computing where the roles among the processors are essentially asymmetrical. P2P networks are typically self-organizing and may or may not have a regular structure to the network. No central directories (such as those used in domain name servers) for name resolution and object look-up are allowed.

An analytical model for adaptive update window protocol is presented in [12], where the window specifies the number of uncommitted updates in each replica node's buffer. The information of each node's update rate and propagation latency are required to optimize the window size in [12]. Such optimization is unrealistic for P2P systems due to their required global information. To the contrary, every BCoM node has a fair amount of consistency maintenance overhead because of the uniform buffer size and node degree in dDT. Moreover, BCoM provides incentives for nodes to contribute more bandwidth to update dissemination, as in proposed approach, it allows that faster nodes are closer to the root and they will receive updates sooner. The window size optimization

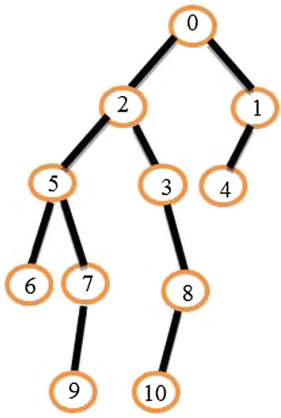

Fig. 1. A simple structure tree decomposition

model in BCoM only requires limited information that can be obtained in a fully distributed way (Fig. 1).

$$\sum_{i}^{N} E_{ij} = \sum_{i}^{N}(A_{ij} + \sum_{i=1}^{V} C_{\mathrm{ij}}) \tag{1}$$

E_{ijt} obeys the rules as follows:

$$\begin{aligned} &E \geq \sum_{i=1}^{N}\sum_{j=1}^{V_N} E_{\mathrm{ijt}} \\ &E_{\mathrm{ijt}} \geq C_{\mathrm{ij}} \geq 0\,(i = 1, \ldots, N; j = 1, \ldots, V_N) \end{aligned} \tag{2}$$

The resource allocation problem is how to control the resource allocation to VMs with the goal of minimizing the function F_t, giving the limited resources. The following formulation is obtained:

$$\begin{aligned} &F_t = min \sum_{i=1}^{N}\sum_{i=1}^{V_i} \frac{f_{\mathrm{ij}}(EN_{\mathrm{ijt}}, \sum_{x=1}^{V} EO_{\mathrm{ijt}}^{x}, D_{ijt})}{\Phi_{ij}} \times SP_{ij} \\ &\begin{cases} \sum_{i=1}^{N}\sum_{i=1}^{V_i} E_{ijt} \leq E \\ E_{\mathrm{ijt}} \geq C_{\mathrm{ij}} \;\; (i = 1, 2, \ldots, V; j = 1, 2, \ldots, N) \\ \sum_{i=1}^{V_i} EN_{\mathrm{ijt}} + \sum_{j=1}^{V_i} EO_{\mathrm{ijt}}^{i} \leq E_i \\ E_{it} \geq C_{ij} \;\; (i = 1, 2, \ldots, V; j = 1, 2, \ldots, N). \end{cases} \end{aligned} \tag{3}$$

This method can be used to avoid deadlock to solve optimal resource model provides $\boldsymbol{V}$ VM-out-of-$\boldsymbol{N}$ PM. The proposed algorithm is based on wait-for graphs (WFG) algorithm and presented in Sect. 5. The probability of obtaining a particular random graph $G = (V, L)$, where $|V| = n$ and $|L| = l$. With Eq. (4) as below:

$$P(G) = \rho^{l}(1-\rho)^{n(n-1)/2-l} \tag{4}$$

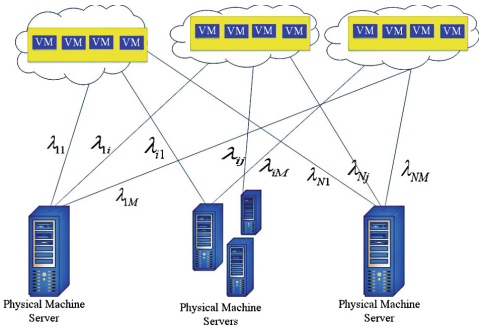

Fig. 2. A simple model $\boldsymbol{V}$ VM-out-of-$\boldsymbol{N}$ PM

4 Analytical Model for Resource Allocation in Heterogeneous Distributed Platforms

The frequent node churn in P2P systems forbids us to use any complicated optimization techniques that require several hours of computation (Fig. 2).

4.1 Queueing Model

Assuming the total number of heterogeneous nodes is N, the node degree is d, and there are $L(L = O(\log_d N))$ layers of internal nodes with an update buffer of size k_i. The leaf nodes are in layer L and do not have any buffer. The update arrivals are modeled by a Poisson process with an average arrival rate λ_i, since each update is issued by a heterogeneous node independently and identically at random. The latency of receiving an update from the parent and an acknowledgment from the child is denoted as the service time for an update propagation. The service time for one layer to its adjacent layer below is the longest parent-child service tim these two layers. μ_l denotes the service time for update propagation from layer l to layer $(l+1)$.

This queuing model explains that given a k_i, the effective buffer size $l * k_i$ is determined by l, which is the layer of the bottleneck node. The larger the effective buffer size, the lower the discard probability. When the bottleneck node is a leaf node $(l = L)$, buffer resources of dDT_i are fully used with an effective buffer size $L * k_i$. The discard probability of an update is computed based on the queuing model of dDT_i after being optimized by tree node migrations as shown. The queue becomes an $M/M/1$ queue with a buffer size $L * k_i$, an arrival rate λ and a service time μ_{L-1}.

4.2 Availability and Latency Computation

Define the update request intensity as ρ.

$$\rho = \frac{\lambda}{\mu_{L-1}} \tag{5}$$

Define the cost spread intensity as $cost_i$

$$\cos t_i = \log \frac{N}{d^0} + \log \frac{N}{d^1} + \ldots + \log \frac{N}{d^{i-1}} \tag{6}$$

$$\cos t_i = \log N + (\log N - b) + \ldots + (\log N - b * (i-1)) \tag{7}$$

$$\cos t_i = i * \log N - \frac{b * \rho * (i-1)}{2} \tag{8}$$

$$\cos t_i = \rho_1 * \cos t_1 + \rho_2 * \cos t + \ldots \rho_i * \cos t_L = \sum\nolimits_{i=1}^{i=L} \rho_i * \cos t_i \tag{9}$$

Define the probability of n updates in the queue as Π_n. Based on the queueing theory for $M/M/1$ finite queue, Π_n is represented as Eq. 9.

$$\Pi_n = \rho^n * \pi_0 \tag{10}$$

The discard probability is π_{L*k_i}, which indicates the buffer overflow. From $\sum_{n=0}^{L*k_i} \pi_n = 1$, we get $\pi_0 = \frac{1-\rho}{1-\rho^{L*k_i}}$. And the discard probability is computed in Eq. 10.

$$\pi_{L*k_i} = \frac{1-\rho}{1-\rho^{L*k_i}} \rho^{L*k_i} \tag{11}$$

The expected number of packets in the queue $E[N_{L*k_i}]$ is calculated in Eq. 11.

$$E[N_{L*k_i}] = \sum_{0 \le n \le L*k_i} n * \pi_n \tag{12}$$

Plug in the Eq. 12 for π_n, the final form of $E[N_{L*k_i}]$ is given in Eq. 13.

$$E[N_{L*k_i}] = \frac{(L*k_i+1) * \rho^{L*k_i+1}}{(\rho^{L*k_i+1}-1)} + \frac{\rho}{1-p} \tag{13}$$

5 Algorithm

In this article, an algorithm dDT construction is proposed for deadlock avoidance maintains property of n-vertex directed graph when the new is added in the graph using two-way search. The dDT construction algorithm uses the number of sub-tree nodes as the metric for insertions, instead of the tree depth used in traditional balanced tree algorithms. This is because a rejoining node with a sub-tree may increase the tree depth by more than one, which is beyond the one by one tree height increase handle by traditional balanced tree algorithms.

The method update dissemination in P2P systems has four requirements: (1) awareness of heterogeneous peer capacities, (2) scalability with a large number of peers, (3) a bounded delay for update delivery, and (4) robustness to the frequent node churn and different workload petterns. The constructs an update dissemination tree by considering each user's capacity and latency requirements to address (1) and (3), both of which are also handled by the tree node migration [1]. The major difference is that Pastry improves the performance to meet the

individual replica node's requirement, while the tree node migration in Pastry improves the overall system performance. Moreover, Pastry requires information on each user's latency requirement and capacity, which are infeasible to be implemented in P2P systems. To the contrary, Pastry node migration only involves local information, which is also performed on demand to support (2) without asking a replica node to specify requirements in advance.

The method ancestor cache has extra benefits by avoiding deadlock communication overhead to maintain end nodes on other sub trees in this article.

Algorithm 1. Algorithm dDT construction (p, q)

Input: node p receives node q's join request;
Output: parent of node q in dDT;

BEGIN
$i.locate.successor(key)$, where key $\neq$ i:
if key $\in$ (i, $successor$) then
return($successor$)
else if p does not have d children **then**;
$Sub_{no.}(p) += Sub_{no.}(q)$ **return** p **else**
find a child f of p s.t. f has the smallest $Sub_{no.}$
$Sub_{no.}(f) += Sub_{no.}(q)$
return dDT Construction(f,q)

END.

In many systems, a heterogeneous environment is preferable to one that is homogeneous. However, it provides better performance particular systems and workload. Even if the workload itself is more suitable to a heterogeneous distributed platforms, the systems rescheduling algorithm should exploit heterogeneity well to benefit from it. Time-bound consistency: TTL guided push and/or pull methods are used to indicate a valid period for a replica copy. When the period expires, the replica node checks the validity of the replica copy with the source to serve the following read requests.

6 Experiments and Results

Nakashima [4] propose a construction of an updated tree statically only from replica nodes. It connects to the tree by computation of the balance of the number of subtree nodes. When new updates are received, the root propagates to all nodes below, then other new updates arrive in the root that will be discarded (Fig. 3).

Yi [6] propose BCoM for construction of an updated tree statically from replica nodes which are ordered by their arrival times and balance of the number

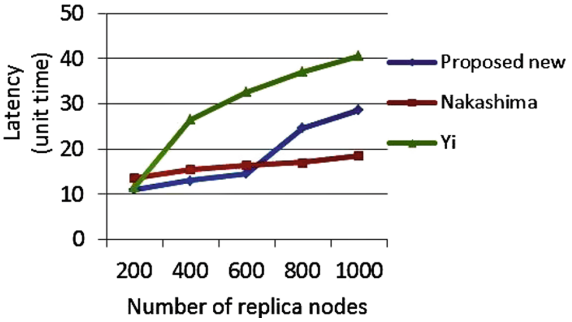

Fig. 3. Comparison the optimal time of proposed algorithms with Nakashima and Yi algorithms

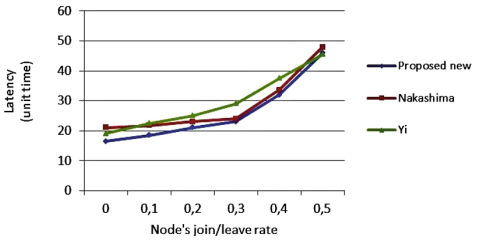

Fig. 4. Comparison the optimal time of proposed algorithms with Nakashima and Yi algorithms

of subtree nodes. Each replica node (except leaf nodes) is given a buffer of size k for buffering updates.

The root receives an update, then sequentially sends it to all its children. Such operation is repeated until the leaf nodes received.

The sliding window size k is critical for balancing the consistency strictness, object availability, and latency. So BCoM proposed a class that each node requires a maximum and sequences of generated updates, balanced among the parameter above.

The comparative analysis of experimental result can be seen in many times, apter task execution, although there were individual time improved proposed algorithm response time was not significantly less than an optimal time algorithm. In most cases, improved algorithm is better than the optimal time algorithm, thus validated the correctness and effectiveness. The process of rescheduling parallel tasks determines the order of task execution and the processor to which each task is assigned (Fig. 4).

7 Conclusion and Future Works

The article provides a solution based on algorithm for allocating effectively resources. The method of optimizing the resource usage is exhibited in the for-

mulae 2. The algorithm proposes a novel way to build and maintain the update tree and then perform the optimal replication based on the update rates. Moreover, it is attributed a the effective method to update propagation. The resultant experiments show that node's join/leave rate, the update rate, and the number of replica nodes increase. Therefore, the proposal is more stable and effective than the work of Nakashima and Yi about the latency and the ratio of the successful update.

Additionally, maintaining the total number of nodes in each sub-tree is simpler and more efficient in term of time than maintaining the depth of each sub-tree. Internal nodes need to wait until an insertion completion. In that case, the updated tree depth can be collected layer by layer from leaf nodes back to the root. This makes the real-time maintenance of the tree depth difficult and unnecessary when tree nodes are frequently joining and leaving. However, internal nodes can immediately upadate the total number of sub-tree nodes after forwarding a new node to a child.

References

1. Rowstron, A., Druschel, P.: Pastry: scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Guerraoui, R. (ed.) Middleware 2001. LNCS, vol. 2218, pp. 329–350. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45518-3_18
2. Bermbach, D., Kuhlenkamp, J.: Consistency in distributed storage systems. In: Gramoli, V., Guerraoui, R. (eds.) NETYS 2013. LNCS, vol. 7853, pp. 175–189. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40148-0_13
3. Kshemkalyani, A.D., Singhal, M.: Distributed Computing Principles, Algorithms, and Systems. Cambridge University Press, Cambridge (2008)
4. Nakashima, T., Fujita, S.: Tree-based consistency maintenance scheme for peer-to-peer file sharing systems. In: 2013 First International Symposium on IEEE Computing and Networking (CANDAR), pp. 187–193 (2013)
5. Shen, H., Liu, G., Chandler, H.: Swarm intelligence based file replication and consistency maintenance in structured P2P file sharing systems. IEEE Trans. Comput. **64**, 2953–2967 (2015)
6. Li, Z., Xie, G., Li, Z.: Efficient and scalable consistency maintenance for heterogeneous peer-to-peer systems. IEEE Trans. Parallel Distrib. Syst. **19**, 1695–1708 (2008)
7. Nassermostofi, F.: Toward authentication between familiar Peers in P2P networking systems. In: Proceeding of the 9th GI Conference Autonomous Systems, pp. 88–103 (2016)
8. Pang, X., Wang, C., Zhang, Y.: A new P2P identity authentication method based on zero-knowledge under hybrid P2P network. TELKOMNIKA Indones. J. Electr. Eng. **11**(10), 6187–6192 (2013)
9. Chen, X., Ren, S., Wang, H.: SCOPE: scalable consistency maintenance in structured P2P systems. In: 24th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM, pp. 1502–1513 (2005)
10. Vishnumurthy, V., Francis, P.: On heterogeneous overlay construction and random node selection in unstructured P2P networks. In: Proceedings of the IEEE INFOCOM (2006)

11. Adami, D., Gabbrielli, A., Giordano, S., Pagano, M., Portaluri, G.: A fuzzy logic approach for resources allocation in cloud data center. In: Proceedings 2015 IEEE Globecom Workshops (GC Wkshps), pp. 1–6 (2015)
12. Zhao, B.Y., et al.: Tapestry: a resilient global-scale overlay for service deployment. IEEE J. Sel. Areas Commun. **22**, 41–53 (2004)
13. www.napster.com/
14. www.gnutella.com/