CC-Buddy: An Adaptive Framework for Maintaining Cache Coherency Using Peers

Song Gao
Dept. of Computer Science
National University of Singapore
gaosong@comp.nus.edu.sg

Wee Siong Ng
Singapore-MIT Alliance
National University of Singapore
smangws@nus.edu.sg

Weining Qian
Dept. of Computer Sci. & Eng.
Fudan University, China
wnqian@fudan.edu.cn

Aoying Zhou
Dept. of Computer Sci. & Eng.
Fudan University, China
ayzhou@fudan.edu.cn

ABSTRACT

In this paper, we propose a framework called CC-Buddy, for maintaining dynamic data coherency in peer-to-peer environment. Working on the basis of peer heterogeneity in data coherency requirement, peers in CC-Buddy cooperate with each other to disseminate the updates by pushing. Simulation results show that our solution not only improves the fidelity in data, but also reduces the workload of servers, therefore achieves high-scalability.

Categories and Subject Descriptors

C.2.4 [**Computer Networks**]: Distributed Systems

General Terms

Management

Keywords

Peer-to-peer, Cache Coherency, Dynamic Data

1. INTRODUCTION

The applications of online dynamic data processing have exponentially grown in recent years. Dynamic data are the data which vary rapidly and unpredictably, such as stock price or network monitoring measurements. Most of these applications are built over centralized systems due to easy-management and implementation. However, they suffer from scale and single-point failure problems. P2P technology has emerged as a popular way in terms of processing and providing data, because of the many benefits they offer: self-organization, load-balancing, fault-tolerance, availability through massive replication, etc. Unfortunately, existing P2P systems provide static data objects management predominantly, such as music and video files. Maintenance of data coherency is an key issue in that peer users always cache dynamic data items to increase the efficiency of query processing. Current dynamic data coherency techniques in P2P environment are inefficient. In this paper, we focus on the problem of deploying P2P technology to online dynamic data processing and management in an overlay network of cooperative peers. In summary, we have made the following contributions: 1. We have implemented the CC-Buddy framework layered on our P2P infrastructure, BESTPEER [3]. 2. We provide heuristical approaches to optimize peer dynamic attributes and network resources usage. 3. We evaluate the performance of CC-Buddy by real trace, and the results show that our framework can not only achieve more fidelity of data than centralized approaches, but also reduce workload of servers so as to scale to large population.

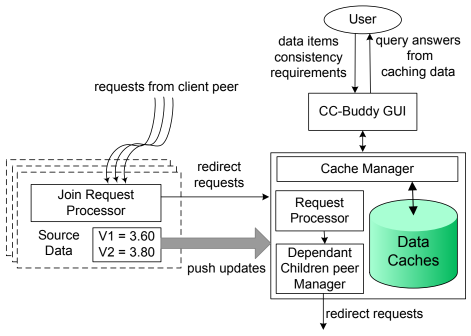

Figure 1: CC-Buddy System Architecture

2. CC-BUDDY ARCHITECTURE

As shown in Figure 1, in CC-Buddy, each peer manages a pool of cached data whose coherency are maintained by cooperation among peers. The proposed framework consists of a collaboration protocol and a set of topology maintenance mechanisms.

2.1 Collaboration Model

We assume that there are a number of server peers taking charge of dynamic data updating and dissemination, and a large number of client peers where dynamic data items are being monitored and processed. Peer users specify coherency requirement, cr, for each cached data item of interest. The value of cr denotes the maximum permissible deviation of the cached value from the value at the server peer and thus constitutes the user-specified tolerance [1]. Each peer maintains his local cached data coherency by participating CC-Buddy overlay. For a particular dynamic data item, peers with high stringent coherency requirement can feed the peers with lower coherency requirement by pushing updates. Server peers are regarded as the highest stringent peers for the data they disseminates. The previous are called *parent peers*, the latter are called *children peers*. Server peers only push data updates to their children peers. Parent peer pushes updates to its children peers in turn. Since each peer may cache numerous data items with variety of coherency requirements, the entire overlay consists of quite a few of logical connected tree topologies rooted by the server peers.

2.2 Overlay Topology Maintenance

We investigate the overlay construction algorithms, recovery mechanisms of peer departure and self-adaptive procedures to show how the overlay works by cooperating with numerous peers. Client peer submits request to specific server peer in order to join a particular data item dissemination tree. If server peer has available capacity, it establishes a logical connection with the client peer as its child peer; if server peer has no enough capacity, it redirects the request to its immediate child peers. The procedure recurs until potential parent peer is found. Client peer register the data item of interest and the associated coherency requirement with its parent peer.

CC-Buddy provides two tree construction policies. One is randomized for choosing redirect targets. Parent peer redirects the target peers at random. Randomized construction requires minimum state storage and computation cost at parent peer. However, the constructed overlay in its way is no optimal. The other is locality-biased, parent peer redirects the request to children that has least access latency with the client peers. The construction procedure takes network locality into consideration. It provides CC-Buddy near-optimal data updates delivery delay so that enhance the fidelity of cached data. The enhancement of performance is achieved at the cost of distance estimation [2]. Collecting the potential parent peers from the overlay, client peer chooses the optimal candidate as the parent peer to join the tree measured in workload, proximity and data availability of each candidate. Furthermore, client peer backups the rest of the peers for future adjustment.

Peers join and leave unexpectedly. In order to adapt to P2P scenario, CC-Buddy provides robust recovery mechanisms to peer failure or departure problem. In the case of a peer's graceful leaving, the peer finds target peers to hand over his children peers. There are two methods: top-down and bottom-up. As a trivial solution, top-down method lets all the peer's children peers rejoin the overlay by re-submitting request to server peers. In a more elegant approach, bottom-up, the peer requests to his parent peer to hand over all the disconnected child peers, the procedure recurs until all the children peers are reconnected. Note that these two approaches remain all the setting of the peer's grandchildren without adjustments. In the case of a peer's ungraceful leaving or failure, the departing peer is unable to notify its children. CC-Buddy provides soft maintenance messages for children peers to detect the failure periodically. Disconnected peers recover by connecting to backup parent peers, or just rejoin the overlay by requesting server peers. Transient peers, such as mobile computing devices or PCs connected in with low-bandwidth network links, are unstable. In the case of transient peers with stringent coherency requirements, the whole overlay will suffer from their churning. An efficient method to reduce the impact of transient peers is to push the transient peers to the bottom of the topology, and let them act as the leaf nodes.

2.3 Heuristic Adaption Policies

CC-Buddy framework is integrated with heuristic algorithms to improve the usability and self-adapt to dynamic attributes. The initial establishment of overlay can not be considered with the dynamic network attributes. Self-adaptive procedures are provided to improve the overlay efficiency, reduce the workload of intermediate peer's heavy burden and adapt to dynamic capacity of peers. When a new coming peer joins the overlay, he can share the workload of its ancestor peers. Furthermore, each child peer periodically compute the measurements to characterize workload, network proximity and bandwidth capacity among his parent peer and each backup parent peer, meanwhile, adjust the overlay by moving to the most beneficial position. Network resource usage has been identified as factors influencing usability in large-scaled networks. Children peers can only be connected with logical connections due to the limitation of available network resources. However, data delivery by logical connections needs to initialize physical connections and release them after finishing disseminating. We provide network reconfiguration to adapt the network dynamics. The goal of our optimization is to assign a set of neighbors to each peer, so that there is a high probability for each client peer to obtain or deliver the updates in short latency. Since the number of allowed network connections is expected to be small, each connection is assigned a benefit value dynamically to reconfigure the topology. As expected, after periodically collecting the statistics, we can figure out the most beneficial connections, establish the physical connections to reduce the dissemination latency.

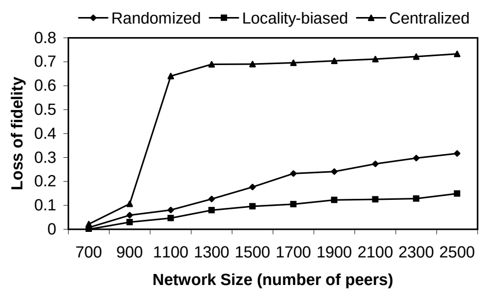

Figure 2: Performance Evaluation

3. PERFORMANCE EVALUATION

We evaluate our framework using real trace. The goal is to minimize loss in fidelity of data. We collected 50 fluctuating stock price traces from the *http://finance.yahoo.com*. We setup 50 server peers, and varied the size of client peers from 700 nodes to 2500 nodes. The server peers disseminate updates periodically. Meanwhile, we set 60% of the cached data items have stringent coherency requirements at each peer, 40% have less stringent coherency requirements. We compare the performance of CC-Buddy taking centralized approach as baseline. As can be seen in Figure 2, CC-Buddy can achieve more efficiency.

4. CONCLUSIONS

Based on application-level multicast techniques, we presented an adaptive framework for maintaining dynamic data using peer-to-peer technology. Experimental results verified that our framework achieves better performance than centralized approach and indicated the scalability of our approach.

5. REFERENCES

[1] P. Deolasee, A. Katkar, A. Panchbudhe, K. Ramamritham, and P. J. Shenoy. Adaptive push-pull: disseminating dynamic web data. In *Proc. of WWW10*, pages 265–274, 2001.

[2] J. Liu, X. Zhang, B. Li, Q. Zhang, and W. Zhu. Distributed distance measurement for large-scale networks. *Int'l Journal of Computer and Telecommunications Networking*, 41, 2003.

[3] W. S. Ng, B. C. Ooi, and K. L. Tan. Bestpeer: A Self-Configurable Peer-to-Peer System. In *Proc. of ICDE2002 (poster)*.