Computer Networks 56 (2012) 1731–1744

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

P2P consistency support for large-scale interactive applications

Yi Hu *, Laxmi N. Bhuyan, Min Feng

Department of Computer Science and Engineering, University of California at Riverside, 900 University Ave., Riverside, CA 92521, USA

ARTICLE INFO

Article history:
Received 20 May 2011
Received in revised form 7 January 2012
Accepted 16 January 2012
Available online 31 January 2012

Keywords:
P2P networks
Consistency maintenance
Computer games

ABSTRACT

Peer-to-Peer (P2P) systems have been widely used by networked interactive applications to relieve the drawback and reduce the reliance on well-provisioned servers. A core challenge is to provide consistency maintenance for a massive number of users in a P2P manner. This requires propagating updates on time by only using the uplink bandwidth from individual users instead of relying on dedicated servers. In this paper, we present a P2P system called PPAct to provide consistency maintenance for large-scale fast-interactive applications. We use massive multi-player online games as example applications to illustrate PPAct. The design can be directly applied to other interactive applications. We adopt the Area-of-Interest (AOI) filtering method, which is proposed in prior works [1,2], to reduce bandwidth consumption of update delivery. We solve the AOI's critical problem of bandwidth shortage in hot regions by dynamically balancing the workload of each region in a distributed way. We separate the roles of view discovery from consistency maintenance by assigning players as "region hosts" and "object holders." A region host is responsible for tracking objects and players within a region, and an object holder is responsible for sending updates about an object to interested players. Lookup queries for view discovery are processed by region hosts, while consistency maintenance of objects is taken by object holders. Separating the roles not only alleviates the workload overflow in hot regions, but also speeds up view discovery and update delivery. Another key idea is that peers contribute spare bandwidth in a fully distributed way to forwarding updates about objects of interest. Thus popular, high-demand objects will have more peers forward updates. We also present how to select capable and reliable players for region hosts and object holders.

A P2P network simulator is developed to evaluate PPAct on two major types of online games: role-playing games (RPGs) and first-person shooter (FPS) games. The results demonstrate that PPAct successfully supports 10,000 players in RPGs and 1500 players in FPS games. PPAct outperforms SimMud [2] in RPGs and Donnybrook [3] in FPS games by 40% and 30% higher successful update rates respectively.

1. Introduction

Modern networked interactive applications, such as online games [3], performance monitoring [4], and collaborative editing [5,6], have moved from dedicated server designs to P2P paradigms for better scalability and robustness. A core challenge for P2P systems is to provide consistency maintenance for all users by only using resources from these users.

P2P managed massive multi-player online games (MMOGs) are representative applications that have been extensively studied [1–3]. Some online game platforms, e.g., Microsoft XBox Live [7], have deployed their P2P services for commercial use. P2P systems for managing MMOGs have the following goals: (1) scalability to serve a large number of players; (2) consistency support for high-frequency interactions; (3) compatibility to different types of games; (4) easy configurability by heterogeneous players. Unfortunately, there is no existing P2P system that achieves all these goals. In this paper, we design a P2P

* Corresponding author.
E-mail address: yihu@cs.ucr.edu (Y. Hu).

1389-1286/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2012.01.012

system, PPAct, that accomplishes all these goals supporting large-scale fast-interactive applications. MMOG is used as an example to illustrate how these goals are satisfied.

Insufficient uplink bandwidth. Online game applications require that each player should have a correct view of game status. The views of all players should be consistent. Thus, a player needs to get timely updates. In the worst case, a player needs every update from all other players to maintain consistency. The total number of updates sent by players is quadratic to the total number of players. However, the total supply of uplink bandwidth used for update delivery increases only linearly with the number of players. The asymmetric bandwidth allocation is dominated by downlink and thus uplink bandwidth is quite limited for normal players. The mismatch between the supply and demand of uplink bandwidth is a major obstacle to good scalability.

We adopt Area-of-Interest (AOI) filtering [1,2] to reduce the number of updates sent to each player. The game map is partitioned into regions, and each player receives updates about surrounding regions. These regions are called *view regions*. Each player subscribes to appropriate view regions in real-time. However, the AOI filtering only alleviates bandwidth shortage but does not solve it fundamentally.

We make additional bandwidth savings by reducing the overhead of view region discovery. We organize all regions into a two-dimensional DHT (2-D DHT) based on their horizontal and vertical positions. As game mobility studies [8] suggest, player movements are mostly continuous. A 2-D DHT provides low-cost region discovery for continuous movements. In addition, the frequency of receiving updates from subscribed regions is tuned temporarily to offset the gap between the supply and demand of bandwidth due to update bursts. Since new updates are sent out periodically, players may ignore obsolete updates in favor of newer ones.

Workload imbalance. The interests of players are usually clustered to a few regions according to the power law distribution of player population density [9]. The players who are responsible for sending updates about the hot regions may not have enough bandwidth to deliver them on time, which is called the AOI "hot spot" problem. A straightforward solution is to adjust region size related to popularity as in the previous works [10,1]. A popular region is divided into several small regions while an unpopular region is kept large. However, varying region size complicates region partitioning in P2P systems. Players change interests as the game evolves, consequently, locations and sizes of hot areas keep changing. It is hard to correctly partition regions because all players should be synchronized and provided with such frequently changing information. Another solution is migrating peers from unpopular regions to popular regions to share the workload as in the previous work [11]. However, it is difficult to correctly perform migration in dynamic P2P systems. In PPAct, we use a uniform region size so that region partitioning information is easily maintained by every player. In this paper, we use peers and players interchangeably, both mean the users of a P2P system.

We solve the AOI "hot spot" problem by breaking down the workload of each region. We separate the object discovery from the object consistency maintenance by assigning each region to a *region-host* and each object to an *object-holder*. A *region-host* is a peer who tracks objects inside a region and players who want to receive updates about this region. An *object-holder* is a peer who holds the primary copy of an object and sends out updates about the object. All objects inside a region are registered to the region-host by their object-holders. Thus, players only query the region-host to subscribe to all objects within that region.

We further break down the workload of update delivery by having each player contribute spare bandwidth to subscribed objects. This avoids overloading object holders of popular objects. Since demand for receiving updates of an object is determined by the number of object subscribers, our scheme ensures that the objects with higher demands have more players to contribute. Moreover, bandwidth is contributed in a fully distributed manner as players piggyback bandwidth information onto their region discovery queries without adding extra overhead.

From our observation, the workload of a region host is more stable than that of an object holder. The number of regions assigned to a region host varies by the region host's capacity. When assigning multiple regions to a peer, we avoid assigning neighboring regions to the same peer so the regions are less likely to have player crowds simultaneously. A region host should also be reliable since it is the contact point for players and object holders. We develop an analytical model for region host selection with both capacity and reliability threshold requirements.

Incompatibility to various games. There are two types of popular online games: action games and role-playing games (RPGs). Each type has its distinctive features and thus different application requirements. An action game requires players to use quick reflexes, accuracy, and timing to overcome obstacles. A popular sub-genre is called first person shooter (FPS). Thus, an action game involves frequent updates and imposes stringent latency bounds for update delivery. Most RPGs cast the player in the role of one or more "adventurers" who specialize in specific skill sets and progress through a predetermined storyline. Compared to FPS games, RPGs have lower update rates and tolerate to longer update latency, but support a larger number of players. RPGs have updates from a great number of Non-Player Characters (NPC) and player avatars, unlike FPS games, which only have updates from player avatars [12]. Existing techniques may address either game type but overlook adaptability. For instance, Donnybrook [3] works well for FPS games, but only scales up to 900 players and does not handle NPCs for RPGs. Another work, SimMud [2], manages NPCs for RPGs, but can rarely meet the latency bound of FPS games.

PPAct is the first P2P system designed to be compatible for different types of games. The separation of region-hosts from object holders not only speeds up update delivery but also provides flexible consistency maintenance for both avatars and NPCs. With all these fully distributed techniques, PPAct satisfies the stringent update latency bound of FPS games and provides good scalability for RPGs. Our extensive simulations show that PPAct successfully supports 10,000 players in RPGs with successful update rate