Supporting MMOG Using P2P Social Network

Author
Author

***Abstract*—This is abstract**

I. Introduction

A peer in the game needs to send game states to other peers whose AOI contains it so that each peer can use game states from others to render the avatars in the virtual world. The problem is that a peer in the hot zone needs to send the game states to many other players, which usually exceeds the peer's upload capacity; on the other hand, a peer in the cold zone has its upload resource idled and wasted since it only needs to send to few neighbors.

In a game based on social network, players have friends in the social network, who may not be playing the game concurrently, but can be used as relays to distribute the game states. We give an incentive mechanism that peers who are playing in the cold zones or not playing at all will help friends in the hot zones, in return, they can get help when they move into the hot zones. For some players without enough friend helpers, strangers (multiple hop friends) are encouraged to assist them in distributing the game states.

II. Design

In [3], M. Varvello *et al*. divide the virtual world of the game into different zones, and have the peer information stored via DHT by their locations in the game zones. Zone division is a simple way to maintain the peers' information in the design, however, it loses the simplicity and brings in great overhead for maintaining the protocols.

In our design, peers are not organized into different zones, instead, each peer directly communicates with the neighbors nearby. When a peer is in a region of the virtual world, it needs to receive the game state messages from the neighbors in its AOI.

The peer also exchange neighbor information between themselves, so that a peer can get new neighbors when it moves in the game world. Each peer in the game holds some neighbors nearby,

Incentives: the contribution of a peer can be: (1) the assistance that peers help friends to relay their game state messages; (2) the share of neighbor information, i.e., a peer can ask a friend or a stranger for the neighbor information. In our design, we give different incentive mechanisms to that between friends and among strangers, and also a combination strategy for the two different mechanisms.

A. Game State Messages

Game states can be considered as a stream with different bit rate that a peer needs to send to all the players, who are close to the peer. The bit rate of the game state messages depends on: (1) the game controls a peer is performing, e.g. a peer running in the virtual world might have a higher bit rate than one walks slowly in the world; (2) when a peer is in different regions, it might need to contact with the environment either, with different contacting frequency. In one word, we give variable bit rates to peers.

III. Incentives

A. Contributions in The Design

Both relaying game state messages or sending neighboring information to other peers are the contributions a peer can make to the system. A peer in some hot regions might need other peers to help it distribute its game state messages to all the neighbors, on the other hand, it has much more chances to send other peers the necessary neighbor information for them to find neighbors. So a peer who always stays in hot regions or cold regions are both able to contribute to the system, by relaying and sending neighboring information, respectively.

In our design, we try to find the best quantization scheme for the contribution, so that peers in the system can get best viewing quality without impairing the fairness among peers.

- Relay
- Neighbor information

B. Incentives

- For friends: "drugs" (contracts or something)
- For strangers: "tools" (game points or something)

C. Quantize the contribution

A peer can contribute to other peers by either relaying game state messages for them or giving neighbor information to them. So total contribution is the combination of the two contributions.

- Relay: what impact the contribution of relay? (1) the bit rate the game state messages r_g; (2) n_r, the number of receivers that peer i is to relay for peer j; (3) the promised time to distribute the messages t_r. The contribution by relay CR that peer i has done for peer j then can be calculated:

$$CR_{ij} = a_1 \cdot r_g \cdot t_r \cdot n_r$$

  a_1 will be the adjust parameter to combine the relay contribution and the neighbor information contribution.
- Neighboring information: the following metrics impact the neighbor information contribution that peer i has done

for peer j, (1) the size of the returned neighbor set N_r; (2) weights of these neighbors w_n;

$$CN_{ij} = a_2 \cdot \sum_{n \in N_r} w_n$$

w_n can be evaluated according to the importance of the requesting peers, e.g. the distance of the two peers.

a_1 and a_2 are the parameters to adjust the weights for the two contributions. We have the total contribution that a peer i has done for peer j :

$$C_{ij} = CR_{ij} + CN_{ij}$$

D. Participants in the Game

- Friend-Friend: when a peer wants some help (relaying or neighbor information) from other peers, it is easier to get the help from a friend because they have close bounder in the social network, so that a friend can offer the help without having being helped before.
- Stranger-Stranger: for strangers, there might be two approaches for one to help the other for the first time: (1) find a friend chain which connects the two strangers by media intermediate friends, which are the "warrantors" to make sure that the helper is able to get the contribution back some time; (2) use global currency to pay for the help, so that the helper can later get help by the credits. In our design, for strangers, we use the later approach for incentives among strangers, and the credits are the game points which are commonly used in the online games.

We give a bound B_{ij} to two peers in the design, so that the contribution between the two peers can not exceed this bound $|C_{ij}| = |C_{ji}| \leq Bij$, which can be decided by the closeness of the two peers in the social network, e.g., friends in the design have much larger bound than totally strangers.

As mentioned above, game points are used for a peer to "buy" some contribution from strangers. The number of game points that is charged also depends on the contribution C_{ij}. Here we use a simple transfer that the number of game points to be charged by peer i from peer j is:

$$GP = \alpha \cdot \frac{\beta}{B_{ij}} \cdot C_{ij}$$

α is a exchange parameter to convert the contribution value to the game points. $\frac{\beta}{B_{ij}}$ is used to for friends to make the transaction, e.g., when a peer j has required so much help from its friend peer i that the contribution balance exceeds the bound, peer j can't get more contribution from i without returning any contribution back. At this moment, it can still get help from i by using game points. The rationale of $\frac{\beta}{B_{ij}}$ lies that friends charge less game points than that among strangers.

How do exchange the game points gained from strangers and the contribution gained from friends. Because a peer sometimes might need help from strangers when its friends are not able to provide the help, although it has contributed to the friends.

TABLE I
TRANSACTIONS

	Contracts	Game Points
Friends	Update C_{ij}, C_{ji} if $\|C_{ij}\| < B_{ij}$	Transfer GP if $\|C_{ij}\| \geq B_{ij}$
Strangers		Transfer GP
C/S (backup)		Transfer GP

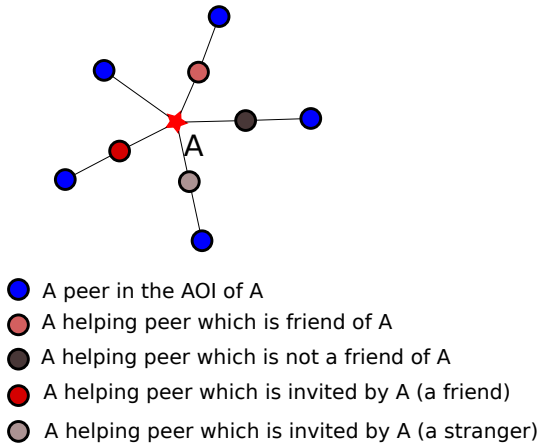

Fig. 1. Pull framework

IV. RELAY DESIGN

A peer moves in the virtual world, and decides which neighbors' game states should be received. Only the interested neighbor's game states should be received, i.e., the peer is staring at at aiming at, while other neighbors' avatars, whom the peer is not interested in, could be guided by AI algorithms. The peer sends a control message to the neighbors that it is interested in, who then will add this peer to a receiver list. Peers need to distribute its game state messages to these neighbors in the receiver list.

More importantly, a peer needs to receive messages from the neighbors, in some modern online MMOGs, the peer is forced to go back several frames ago if some of the neighbors' game state messages are found not received. Thus, receiving all the messages from the neighbors is a big incentive for peers, since this can improve the fluency of the game controls.

In our design, peers actively pull the game state messages from the neighbors in its AOI, which are maintained by the neighbor discovery mechanism we will discuss in Sec.**??**. For the neighbors whose game state messages should be received, (1) the peer asks the neighbor to send the game state messages directly, and it will get the stream directly from the neighbor who has enough upload capacity to send one more stream; (2) or the peer will some relay peers for the neighbor if its upload capacity is exceeded, then the peer tries these relay peers to get the stream from the neighbor; (3) when the relay peers are still unavailable, the peer will invites a peer to relay for the neighbor by inserting it into an existing distribution path of the neighbor.

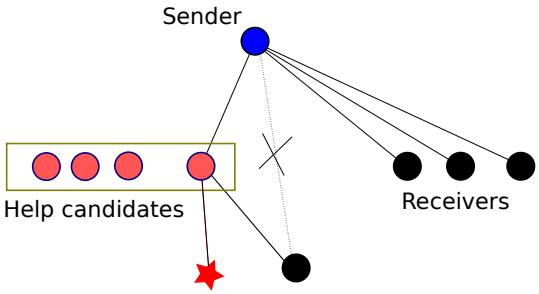

Fig. 2. When the peer A finds the neighbor (and the relay peers for it) are not able to send the game state messages, it invites a relay peer from its relay candidate pool. An existing link has to be broken to insert the new relay

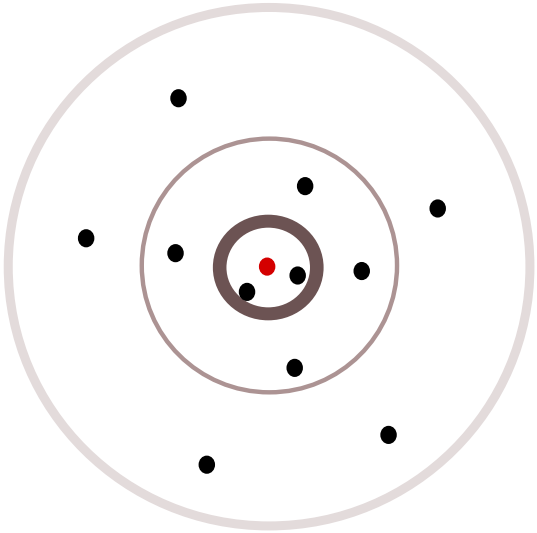

Fig. 3. Neighbors in the first circle are the closest neighbors, who may request the peer's game state messages. Neighbors in the second circle are the full maintain neighbors, they don't exchange game states, but the neighbor list is complete. The third circle is the far neighbors, which are randomly maintained for neighbor discovery, the larger the distance is, the fewer neighbors are stored.

A. The strategy of selecting a relay helper

V. Peer Discovery

What neighboring information a peer stores:

A. Backup Schemes for The Missing Neighbors

VI. Related Works

A. Zone-division and DHT-based approaches

For neighbor discovery, M. Varvello *et al.* [3] use Kad for peer discovery (todo: cite more DHT based designs). S.Y. Hu *et al.* [1] use an unstructured mechanism for peer management. J. Kim *et al.* [2] use a hybrid where DHT is used for discovery of super peers, who then manage the sub-overlays.

B. Our Distributed Design

As mentioned in Sec. II, a peer find new neighbors from the current neighbors when it moves into different regions in the virtual world.

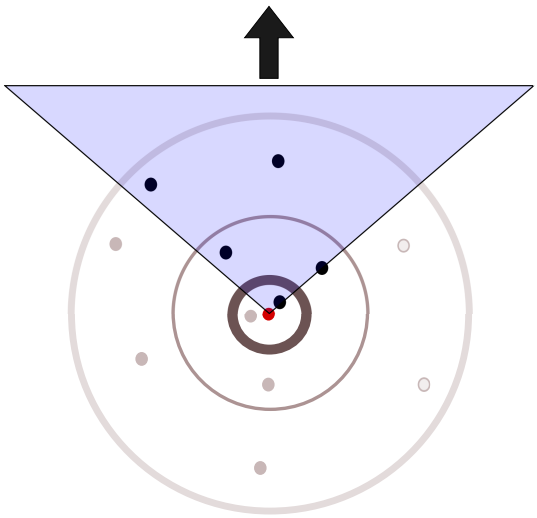

Fig. 4. Neighbors in the circles are stored with directions. When a peer moves into a direction, it will ask the neighbors in that direction for the neighbor information, in return, the neighbors also calculate the best fit peers to return to the requesting peer. The nodes in black will be requested for the neighbors held in their neighbor information.

VII. To be removed (backup)

Contract: when a peer requires a friend to assist in the dissemition of the game state messages, a contract is made, so that the relay peer will help for a time span T_h.

$C_{i,j}$: the number of receivers that peer i is to send to as relay peer for peer j.

$B_{i,j}$: the bound on node i and node j, and value of $B_{i,j}$ is decided by the closeness of two friends i and j.

$$C_{i,j} \leq B_{i,j} \tag{1}$$

C_i: the total number of receivers peer i has sent to for all its neighbors:

$$C_i = \sum_k C_{i,k}$$

U_p: peer p's upload capacity:

$$C_p \cdot r + |receiver(p)| \cdot r \leq U_p \tag{2}$$

Here $receiver(\cdot)$ is the direct receivers of a peer. r is the average bitrate of the game state streaming.

D_p: peer p's download capacity:

$$\sum_k C_{k,p} \cdot r + |receiver^{-1}(p)| \cdot r \leq D_p \tag{3}$$

$receiver^{-1}(\cdot)$ is the set of peers that have sent game state messages to peer p.

R_p: the total number of receivers a peer p needs to send to when it is in some zone:

$$R_p = \sum_k C_{k,p} + |receiver(p)|$$

R_i is the sum of peers who are to receive its game states, including the direct receivers and the peers who receive from the relay peers.

$$R_p \geq z(p) \tag{4}$$

$z(p)$ is the number of peers which peer p needs to send to when it is in some zone in the game. $z(p)$ is also an indicator of the popularity of the zone.

I: the imbalance indicator which indicates the imbalance situation in the system:

$$I = [\frac{1}{N}\sum_{p}(C_p - \mu)^2]^{1/2} \tag{5}$$

μ is the mean value of C_p for all peers and N is the total number of peers.

When a peer has promised to assist another peer for T_h seconds, even when it enters the hot zones. We believe this mechanism can reduce the impact of peer fluctuation.

Schedule problem: The schedule is how peers select the relay peers when it needs other neighbors' help, and here we want to minimize the imbalance value I, so that peers are easy to get help in future for the overall system to enhance the performance.

We want to find a feasible contracts assign, *i.e.*, peers are able to distributed their game states via relay peers, with

$$\min(I) \quad s.t. 1, 2, 3, 4, 5$$

TODO: balance is a problem? More importantly, we need to find a feasible soultion, i.e., the $receiver(p)$ and $C_{i,j}$ should be given proper values so that 1, 2, 3, 4, 5.

REFERENCES

[1] S. Hu, S. Chang, and J. Jiang. Voronoi state management for peer-to-peer massively multiplayer online games. In *Proc. of CCNC*, pages 1134–1138. Citeseer, 2008.

[2] J. Kim, E. Hong, and Y. Choi. Measurement and analysis of a massively multiplayer online role playing game traffic. In *Proceedings of Advanced Network Conference (Network Research Workshop)*.

[3] M. Varvello, C. Diot, and E. Biersack. P2P Second Life: experimental validation using Kad. *Proceedings of INFOCOM09*, 2009.