The Optimization Problem

May 5, 2010

Abstract

THE ABSTRACT

1 Design

1.1 The flow

1.1.1 Basic idea

We aims to use P2P technology to distribute the information in the MMOGs, where each peer sends to and receives from other peers whose avatars are nearby in the virtual game world. Due to the peers' local interests, their upload capacities can be used to alleviate the game servers. However, the popularity of the regions in the game world is highly skewed [1], so that some peers in the "hot" regions need to disseminate much more messages to neighbors than others in the system. To make best use to peers' resources, we have peers in relatively "cold" regions help those in hot regions, in return, they will also be able to receive the help from others when they are in the hot regions, due to the avatar movement in the game. In our design, peers also consider the social relationship in the system, i.e., friends who have stronger relation may be more likely to help each other.

1.1.2 Incentives

We observe that the incentive mechanism here is quite different from that in Bittorrent systems, e.g., the trading between two peers is not happening at the same time (i.e., a peer might need to help another peer even if it can't get the help back immediately). In our design, receipts and game points are traded among friends and strangers respectively, to achieve such indirect reciprocity in the system.

We amount of receipts or game points are evaluated by the contribution a peer has done to others: (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.

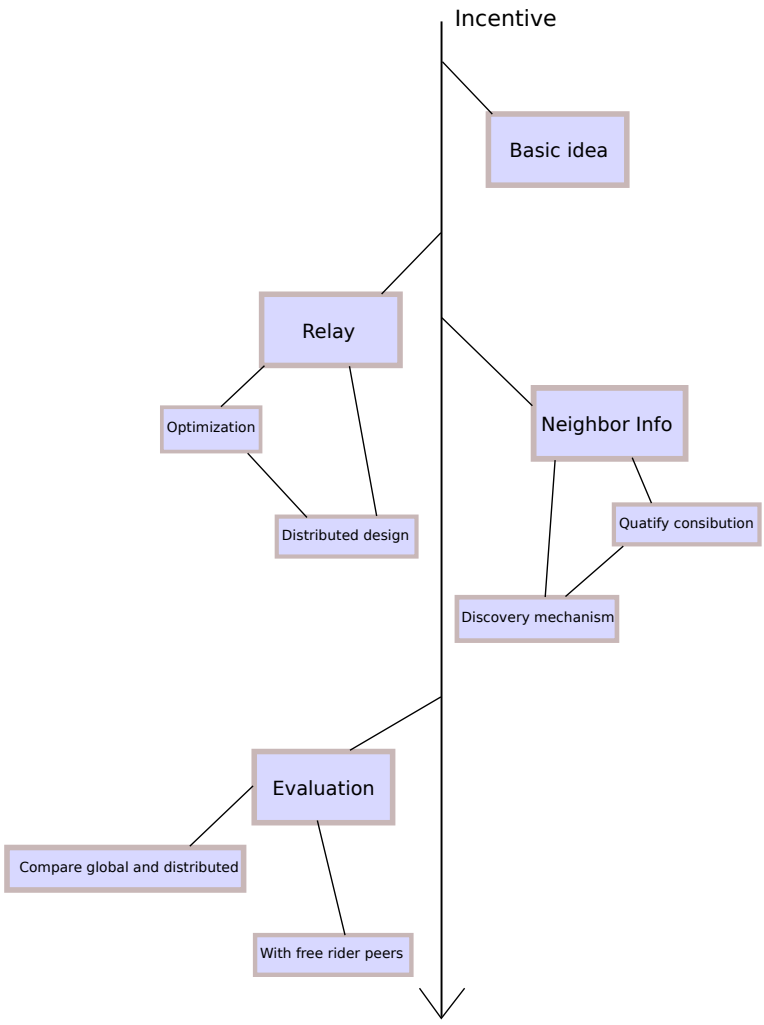

Figure 1: The flow

2 Incentives

2.1 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.

2.2 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: (1) upload resource; (2) latency

- Neighbor information

2.3 Quantify the contribution

2.3.1 Contribution types

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) 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:

$$CR_{ij} = a_1 \cdot \log(r_g) \cdot T_r$$

T_r is the time each relay takes, i.e., i promises to help for T_r seconds.

- 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;

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

The importance indicator w_{nj} of a neighbor to peer j is evaluated by the distance $dis(j, n)$ between j and n in the virtual game world:

$$w_{nj} = \frac{1}{\log(dis(j, n))}$$

2.3.2 Cost of contribution between friends

We use the matrix $\{f_{ij}\}_{N \times N}$ to denote the social relationship between peers in the system. The larger f_{ij} is, the closer the two peers are to each other. For strangers, the f_{ij} value is much larger than that of friends.

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 :

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

2.3.3 Game points among strangers

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:

$$GP_{ij} = \frac{\alpha}{f_{ij}} \cdot C_{ij}$$

α is a exchange parameter to convert the contribution value to the game points. How to 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 1: Transactions

	Contracts	Game Points
Friends	Update CC_{ij} if balance holds	Transfer GP_{ij} ($B_i = B_i + GP_{ij}$,$B_j = B_j - GP_{ij}$) if $\lvert CC_{ij} - CC_{ji}\rvert > \beta \cdot f_{ij}$
Strangers		Transfer GP_{ij}
C/S (backup mechanism)		

2.4 Balance in the system

We use the CC_{ij} to denote the relay and neighboring information contribution that peer i provides to peer j, where i and j are friends, and B_i to denote the balance of peer i's game points.

Balance between friends: To keep the balance, the contribution peer i do for j should be similar to that j has done for i, and the difference should not exceed some value, which is related with the closeness of the two peers:

$$|CC_{ij} - CC_{ji}| < \beta \cdot f_{ij}$$

Here *beta* is a parameter to transfer the entry in the closeness matrix $\{f_{ij}\}_{N\times N}$ to the balance value.

No-bankruptcy balance of game points: For game points, we simply require that peers' balance of game points B_i greater than 0. This rationale ties that in most MMOGs, players are demanded to buy game points to play the games, and they always need to charge the account when the game points are used up.

$$B_i \geq 0, \quad i = 1, 2, \ldots, N$$

3 Relay Design

Why we need relay design?

(A) Upload resource: Since the popularity of the game regions is highly skewed, a few regions have much more peers in them than those in other regions. On the other hand, a peer always needs to receive all game state messages from all the peers nearby in its AOI. When in the hot regions, the peer may not be able to get the game state messages from some interested peers, due to the lack of upload capacities among these peers. At this moment, the peer can ask its friends or other helper peers to join the stream overlays to help relay the game state messages, so that the peer can receive indirectly from the relay helper peers.

(B) Latency concern: Latency is one of the most important factors that impact the user experience in MMOGs (todo:ref). Thanks to the prevalence of triangle inequality of the end-to-end latencies, it is possible for the peers to

make use of their friends' relaying to reduce the latencies of receiving the wanted game state messages.

Incentives

The incentive for relay peers to help lies: (1) a peer is likely to help its friends in the system, since it trusts the friends, and sometime its friends will help it back; (2) credits could be paid to get the help, and the money can be then used to "buy" help from stranger relay peers.

Credits in the game: in our design, we use game points for incentive, as many online games require players to buy the game points to join the game. A part of the money earned by selling game points is spent on the operation of the game, so if a peer can help more other peers, it is reasonable to give it some "bonus" game points, since it help to alleviate the server cost by contributing resource to the system.

Structure of this section

In this section, we first formulate the problem to a resource allocation schedule with social network relationship of peers, and then we give an optimization framework to solve the problem, including both global optimization and distributed algorithm.

3.1 Relay Problem

In this subsection, we describe the relay problem, including the system's view and an individual peer's view. **(1) For the whole system**, we need to schedule the peers to receive and relay the different streams (for different peers, the bitrates are different) strategically to improve the stream quality for all the peers; **(2) For an individual peer**, it needs to maximize the number of streams received from interested peers and the minimize the latencies, and keep debt low with the friends.

3.2 Problem Formulation

We still use a directed graph $G_n = (R_n, N, E)$ to represent a stream overlay, where n is the source peer, and R_n is the set of peers who are to receive the stream. N is the set of all peers in the system, which can be used as relays, and E is the application-layer connections between peers.

For each stream in the system whose source is peer n, peers in R_n will try to receive the stream, whose bit rate is r_n. When a peer enters the overlay, it first tries to receive from the existing peers if they have enough upload capacities to send the stream; or it will brings in a new relay peer to receive the stream.

How a peer schedules: when a peer moves into a new region, it will find the neighbors close to it by the peer discovery mechanism (we will discuss in Sec. **??**, and try to receive messages from the ones in its AOI.

S_n is the set of peers whose streams should be received by peer n, and $S1_n$ the set of peers whose streams are not address by peer n, i.e., not received due to the lack of upload resource or large latency. F_n is the set of friends and other peers that peer n can make use of, to get the un-addressed streams in $S1_n$.

In our design, the peer will try to schedule the relay peers in F_n to minimize the number of unaddressed streams.

Let's first list the constraints: a_{ij} is the upload allocation of peer i for stream j.

$$\sum_j a_{ij} \leq u_i$$

The price of a friend's contribution

Here we first denote the c_{ij}

3.3 Optimization Framework

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.

3.4 Global Optimal Problem

a_{ij} is the upload bandwidth that node i contribute to the stream j. Here is the constraints in the problem:

- Upload capacity constraint:

$$\sum_{j=1}^{N} a_{nj} < u_n \quad n = 1, \ldots, N$$

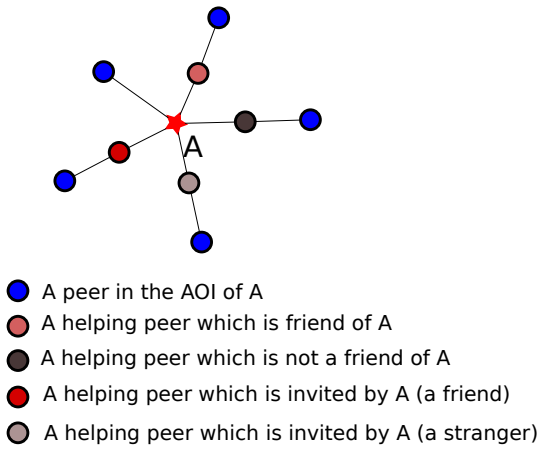

Figure 2: Pull framework

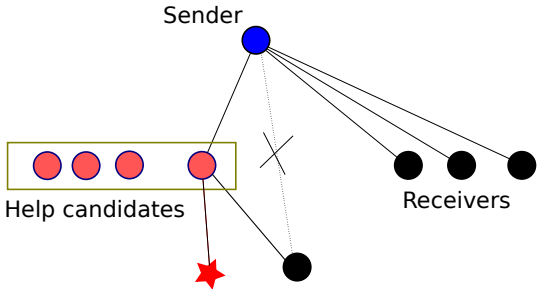

Figure 3: When the peer A finds the neighbor (and the relay peers for it) are not able to send the game state messages, it invites a relay peer from its relay candidate pool. An existing link has to be broken to insert the new relay

- Download capacity constraint:

$$\sum_{j \in R_n} r_j + \sum_{j \notin R_n} r_j \cdot f_{nj} < d_n, \quad n = 1, \ldots, N$$

Here, $f_{ij} = \begin{cases} 1 & a_{ij} > 0 \\ 0 & a_{ij} = 0 \end{cases}$

The Objective Function: The objective function is used to evaluate the streaming quality for all streams in the system. For stream n:

$$e(n) = \frac{\sum_i a_{in}}{r_n \cdot (|R_n| + \sum_{i \notin R_n} f(i))}$$

Table 2: Notations

Symbol	Definition
a_{ij}	The upload capacity that peer i contribute to stream j
u_i	The upload capacity of node i
d_i	The download capacity of node i
r_i	The bitrate of the stream of node i
S_i	The set of peers who needs to receive the stream of node i (AOI)
R_i	The set of peers whose streams node i needs to receive (AOI)

And we should try to maximize the objective function:

$$\max \min_n e(n)$$

3.5 The Distributed Optimization

Table 3: Notations (distributed algorithm)

Symbol	Definition
H_n	The set of peers that can be used as relays
R_n	The set of unaddressed source peers
u_i^c	Available upload capacity of peer i in H_n
d_i^c	Available download capacity of peer i in H_n
r_j	The stream bitrate of source peer j in R_n
f_{ij}	The indicator function whether peer i is a relay peer of stream j

H is the set of friends and some stranger neighbors of the peer n. R_n is the source peers whose streams are not received due the lack of upload capacities. The problem then is how to schedule peers in H_n to help relay the streams in R_n.

- For peer $i \in H_n$, the aggregate upload bandwidth used is: $u_i = \sum_j 2 \cdot f_{ij} \cdot r_j$ and
$$u_i \leq u_i^c$$
- For peer $i \in H_n$, the aggregate download bandwidth used is: $d_i = \sum_j f_{ij} \cdot r_j$ and
$$d_i \leq d_i^c$$

To maximize the number of addressed streams, our objective function then is:

$$\max \sum_j g_j, \quad \text{where } g_j = \begin{cases} 0 & f_{ij} = 0, \forall i \\ 1 & \text{otherwise} \end{cases}$$

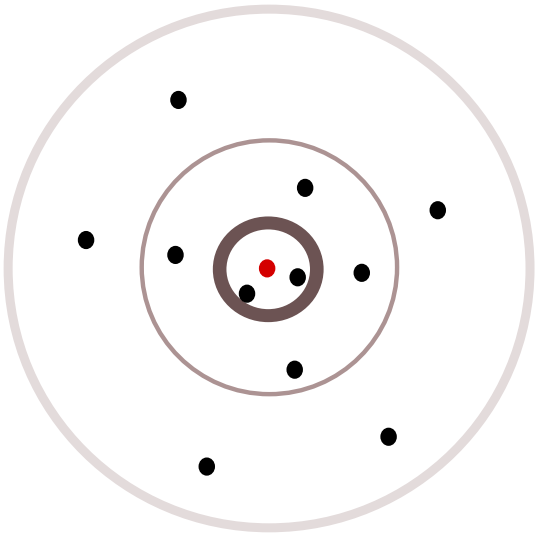

Figure 4: Neighbors in the first circle are the closest neighbors, who may request the peer's game state messages. Neighbors in the second circle are the full maintain neighbors, they don't exchange game states, but the neighbor list is complete. The third circle is the far neighbors, which are randomly maintained for neighbor discovery, the larger the distance is, the fewer neighbors are stored.

4 Peer Discovery

A peer needs to receive game state messages from different neighbors when it is in different regions, since the set of peers in its AOI keeps varying due to the movements of the peer and other neighbors nearby. It is important for players in the MMOGs to discover new neighbors when it joins the game or moves to a new region in the game world. (todo: cite, to show that the peer discovery can impact the consistency and latency)

4.1 Basic idea

When a peer moves to a new region:
When a peer first joins the game:

4.2 The peers to return to the requesting peer

What neighboring information a peer stores:

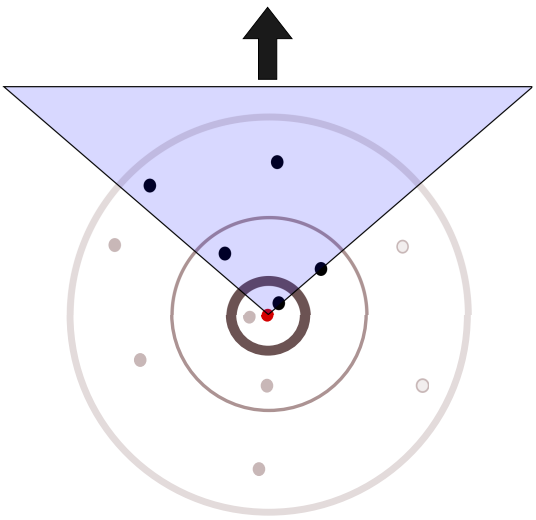

Figure 5: Neighbors in the circles are stored with directions. When a peer moves into a direction, it will ask the neighbors in that direction for the neighbor information, in return, the neighbors also calculate the best fit peers to return to the requesting peer. The nodes in black will be requested for the neighbors held in their neighbor information.

4.3 Backup Schemes for The Missing Neighbors

5 Evaluation

Consider free ride?

6 Conclusion

References

[1] D. Pittman and C. GauthierDickey. A measurement study of virtual populations in massively multiplayer online games. In *Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games*, pages 25–30. ACM, 2007.