Draft

Monday 17th May, 2010

1 Design

1.1 The flow

1.1.1 Basic idea

We aim to use P2P technology to distribute the information (i.e. game state messages and neighbor information) in the MMOGs, where each peer sends to and receives from other peers whose avatars are nearby in the virtual game world. Thanks to the locality of peers' interests, they only need to know what happens nearby. Peers can receive the game state messages from these peers directly can play their avatars, to alleviate the server. Nevertheless, the popularity of the regions in the game world is highly skewed [1], and some peers in the "hot" regions (i.e. where many avatars are there) need to disseminate much more messages to neighbors than others in the system. To make best use of peers' resources, we have peers with extra upload capacities help those in hot regions, in return, they will 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, the social friendship is also considered 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.

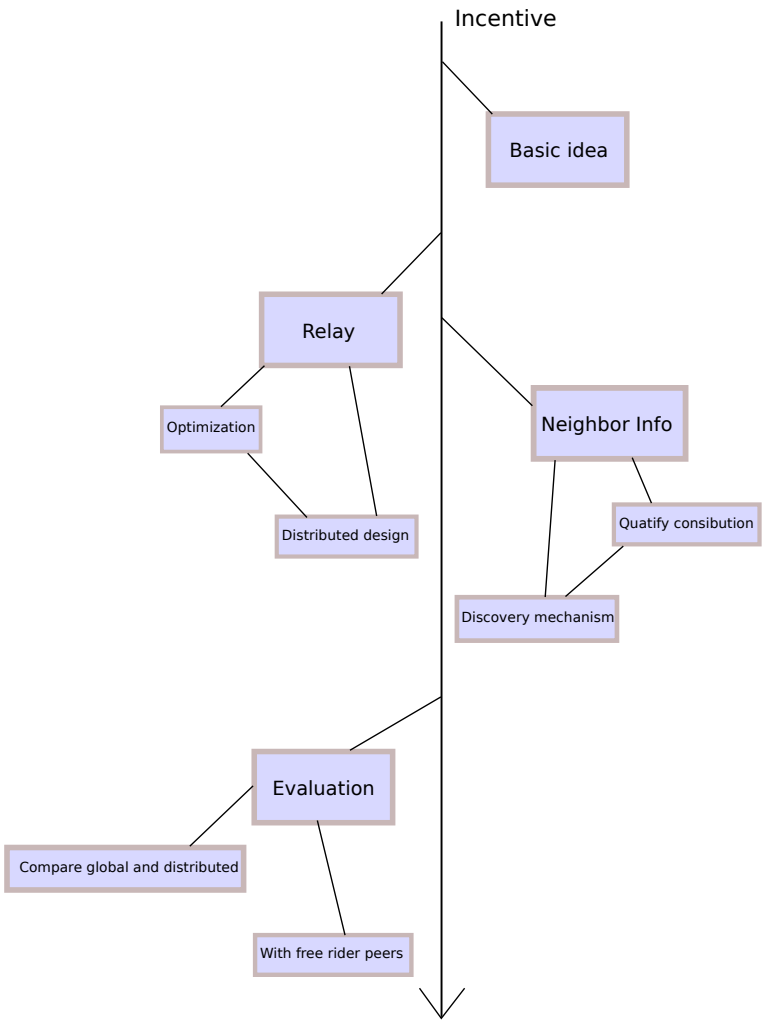

Figure 1: The flow

2 Terminology

Stream:

Game state message:

AOI:

Relay:

3 Notation table

Table 1: Important notations

Symbol	Definition
x_{ij}^s	The net upload resource peer i contributes to help peer j to receive stream s
r_s	The bit rate of stream s
f_{ij}	The closeness of two peers i and j in $[0, 1]$, the larger the closer.
c_{ij}	The connection of two peers i and j, $\{0, 1\}$
$Gain_{ij}^s(x_{ij}^s)$	The gain for peer i to contribute x_{ij}^s to help peer j to receive stream s
$GP_{ij}(x_{ij}^s)$	the game points of the upload resource contribution x_{ij}^s
$E_{ij}^s(x_{ij}^s)$	The streaming quality gain from the contribution x_{ij}^s
$S(j)$	The set of unaddressed streams that peer j needs to receive
$R(s)$	The set of peers who has to receive the stream s
u_i	The upload capacity of peer i
u_i'	The estimated upload capacity of peer i
t_{ij}^s	The time length that peer i helps peer j

4 Incentives

Why discuss incentives here? Peers are selfish and not willing to contribute their resource to others, unless they can gain some thing or be helped later by other peers. However, when considering social network, friends are much more likely to help each other than strangers.

4.1 Participants in the Game

- Friends: friends usually afford to help others without requiring any pay back immediately. In our design, we use a "balance" mechanism among friends assistance. Each peer records the contribution it has done for all its friends and the contribution its friends has done for it, and keeps the difference below some value, which is a proportion of the closeness of the two peers.

- Strangers: for strangers, they usually need some "strong" incentive to help others. In our design, a peer uses the global game points to "buy" assistance from a stranger peer.

4.2 Relaying game state messages is the contribution in the design

Relaying game state messages is the contribution of a peer in the system, i.e., a peer contribute its download/upload capacity to help others receive some streams.

A peer in some hot regions might need other peers to help so as to receive all game state messages from the neighbors in its AOI.

Why are some peers need others to help them to receive the messages: (1) the upstream peer (the source of the stream) has limited upload capacity to send out its game state messages to all the receivers (especially in hot regions); (2) although the upload capacity is sufficient, the latency is too large. It is possible to reduce the latency by having a relay peer to forward the message due to the common existence of triangle inequality of end-to-end latencies on the internet.

4.3 Gain of the contribution

x_{ij}^s is the upload capacity that peer i contributes to help peer j in receiving the stream s, whose bit rate is r_s.

What impacts the quantification of the contribution of relaying: (1) the upload capacity that a peer i contributes to help peer j; (2) the closeness of the two peers.

$$Gain_{ij}^s(x_{ij}^s) = GP_{ij}(x_{ij}^s) + FG_{ij}(x_{ij}^s)$$

$GP_{ij}(x_{ij}^s)$ is the function to evaluate the game points of the upload resource contribution x_{ij}^s. It is defined as follows.

$$GP_{ij}(x_{ij}^s) = \begin{cases} \alpha \cdot x_{ij}^s & i \text{ and } j \text{ are strangers} \\ \alpha \cdot x_{ij}^s & i \text{ and } j \text{ are friends, and } \alpha \cdot CC_{ji} - CC_{ij} < -\beta f_{ij} \\ 0 & i \text{ and } j \text{ are friends, and } \alpha \cdot CC_{ji} - CC_{ij} \geq -\beta f_{ij} \end{cases}$$

$FG_{ij}(x_{ij}^s)$ is the gain from helping friends. This is the incentive for friends to help each without receiving as many game points as that by helping strangers.

$$FG_{ij}(x_{ij}^s) = \phi \cdot \log(fun(CC_{ij} - CC_{ji} - \beta f_{ij}) - x_{ij})$$

$fun(\cdot)$ is a increasing function.

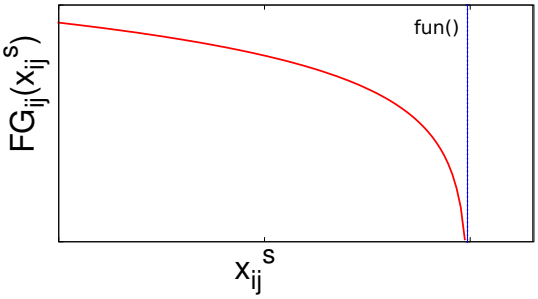

Figure 2: $FG_{ij}(x_{ij}^s)$

Table 2: Transactions

	Contracts	Game Points
Friends	Update CC_{ij} if balance holds	Transfer GP_{ij} if $\|CC_{ij} - CC_{ji}\| > \beta \cdot f_{ij}$
Strangers		Transfer game points

4.4 Balance in the system

We use the CC_{ij} to denote the total contribution that peer i provides peer j, where i and j are friends.

Change CC_{ij}: each time peer i has helped peer j, CC_{ij} will be increased by some value according to the contribution, which is decided by both the upload capacity and the time.

$$CC_{ij}(t) = CC_{ij}(t-1) + \sum_{s \in S(j)} a_{ij}^s \cdot t_{ij}^s$$

Balance between friends: To keep the balance, the contribution peer i does 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 β is a parameter to transfer the entry in the closeness matrix $\{f_{ij}\}_{N \times N}$ to the balance value. A peer needs to use game points to buy assistance from its friends when the balance is not kept.

5 Relay Design

A peer in the virtual world has to receive the game state messages of the neighbors in its AOI. Receiving these messages correctly and timely is important, since in some online MMOGs, a peer is forced to go back if some of the neighbors' game state messages are not received consistently. In our design receiving

all the messages from the neighbors is one of the most important objective for peers, since it improves 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.**??**. The peer receives the messages in two manners: (1) the peer asks the neighbor to send the game state messages directly when the neighbor has enough upload capacity to send one more stream; (2) or the peer will ask a helper peer to relay the stream when the source peer's upload capacity is exceeded or the latency is too large.

Why we need relay design?

(A) Upload resource: Since the popularity of the game regions is highly skewed (todo:ref), a few regions have much more peers in them than others. A peer always has to receive all game state messages from all the peers 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 capacity 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. The reason helper peers can increase the capacity is that they only consumes $1x$ but brings in $2x$ or even more upload capacity.

(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) game points could be paid to get the help, and the money can be then used to "buy" help from stranger relay peers.

Game points 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 by the game operator, so if a peer can help more other peers, it is reasonable to make it save more game points, since it help to alleviate the server cost by contributing resource to the system.

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.

5.1 How a requesting peer schedules (downstream peer j)

5.1.1 Optimization problem

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_{in}^j is the net upload allocation of peer i for peer n for relaying stream j, i.e., the upload resource the peer i contributes minus the download resource it consumes, since it first needs to receive and then relay it.

For peer j:

$$\max \sum_{i:c_{ij}=1} \sum_{s \in S(j)} E_{ij}^s(x_{ij}^s) - GP_{ij}(x_{ij}^s)$$

subject to

$$\sum_{i:c_{ij}=1} x_{ij}^s \geq r_s, \quad s \in S(j)$$

$$x_{ij}^s \geq 0$$

$$\sum_{s \in S(j)} x_{ij}^s \leq u_i', \quad \forall i : c_{ij} = 1$$

$E_{ij}^s(x_{ij}^s)$ is an evaluation function on the streaming quality.

$$E_{ij}^s(x_{ij}^s) = \log(1 + \epsilon \cdot \frac{x_{ij}^s}{r_s})$$

5.1.2 Schedule algorithm

For the downstream peer, it will try to receive all the streams in $S(j)$ by the assistance of friends and other peers with the least game points consumption.

1. Initiate the upload capacities of the helpers;
2. Solve the local optimization problem, and asks the helper peers for the allocation;
3. Update the estimated upload capacities of the helper peers: (todo: we use AIMD?)
   - For $a_{ij}^s \geq x_{ij}^s$, increase u_i' by a constant value;

- For $a_{ij}^s < x_{ij}^s$, decrease u_i' by a constant value;

4. Until all the streams are addressed;

5.2 How the service peer responds (upstream peer i)

5.2.1 Optimization problem

For a peer, deciding which requests to serve when the upload capacity is not sufficient to supply all of them, is also an optimization problem.

For peer i:

$$\max \sum_{j:c_{ij}=1} \sum_{s \in S(j)} Gain_{ij}^s(a_{ij}^s)$$

subject to

$$\sum_{j:c_{ij}=1} \sum_{s \in S(j)} a_{ij}^s < u_i$$

$$a_{ij}^s < x_{ij}^s$$

5.2.2 Schedule algorithm

For the upstream peer i, it uses the following strategy to responds all the requests:

1. Calculate the "unit price" of upload capacity from all the peers ($\frac{Gain_{ij}^s(x_{ij}^s)}{x_{ij}^s}$)
2. Serve the one with the highest unit price;
3. If there is still upload capacity left, continue the process;

5.3 Global optimization

The distributed algorithm actually solves a global optimization problem:

$$\max \sum_{i=1}^{N} \sum_{j:c_{ij}=1} \sum_{s \in S(j)} E_{ij}^s(y_{ij}^s) - GP_{ij}(y_{ij}^s)$$

subject to

$$\sum_{i:c_{ij}=1} y_{ij}^s \geq r_s, \quad j = 1, 2, \ldots, N, \forall s \in S(j)$$

$$y_{ij}^s \geq 0$$

$$\sum_{j:c_{ij}=1} \sum_{s \in S(j)} y_{ij}^s \leq u_i, \quad i = 1, 2, \ldots, N$$

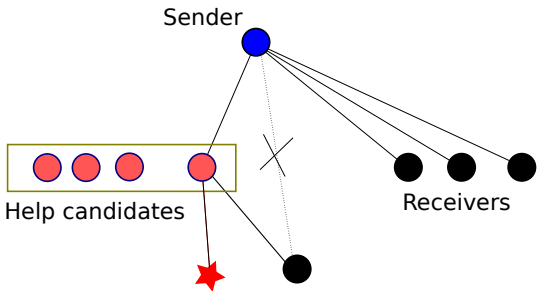

Figure 3: When the peer A finds the neighbor (and the relay peers for it) are not able to send the game state messages, it invites a relay peer from its relay candidate pool. An existing link has to be broken to insert the new relay

6 Evaluation

Consider free rider?

7 Conclusion

8 Appendix: problems

todo: probelms:

1. no global optimization?
2. quantify friendship
3. the problem of service peer

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.