SatTorrent – Satellite-Aided P2P Content Distribution for Virtual Environments

[Data Distribution for Massive Multiplayer Online Games]

Bernd Klasen*
SES ASTRA TechCom
Chateau de Betzdorf
L-6815 Betzdorf, Grand Duchy of Luxembourg
research@berndklasen.de

ABSTRACT

Distributed virtual environments and their most prominent representative *massive multiplayer online games* enjoy a high and still growing popularity. The distribution of client updates is a challenging task already and will become much worse when users are allowed to apply extensive changes respectively additions to the virtual world by means of user generated content. This paper presents a satellite based peer-to-peer approach that is able to reduce the download duration, server load, overall traffic and thus the costs involved with the content distribution. In order to achieve this, SatTorrent – a modification of the BitTorrent protocol – has been developed which is presented and evaluated in this paper.

Categories and Subject Descriptors

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

General Terms

Performance

Keywords

content distribution, peer-to-peer protocols, sattorrent

1. INTRODUCTION

The Internet is connecting devices and people from all around the world and enables them to communicate instantly and in various ways. One that offers a high degree of immersion are *distributed virtual environments* (DVE). Among them, massively multiplayer online games (MMOG) are probably the most widespread representatives. A well known example here is *World of Warcraft* (WoW). Even though it reached the respectable age of 7 years it still has more than 10 million subscribers. There is a multitude of other games like *Lord of the Rings Online*, *Star Trek Online*, *Star Wars: The old Republic* just to name a few, which are also very popular respectively gaining a lot of attention and are expected to reach similar scales. The majority of the most prevalent MMOGs relies on pre–installed clients that contain all relevant world data such as textures, maps et cetera. While these software packages are updated once in a while to add new content, new features or for bugfixing reasons, the world does not change within the periods between two updates. This makes the provision of such games much easier, but besides other limitations that adhere to this circumstance, it prevents players from creating and adding user generated content (UGC).

From the paradigm change in the Internet (*web 2.0*) we learned how much creative power users are willing to invest. A prominent example are online video platforms where users can upload their self made videos and share it to the rest of the world. The most famous among those is *YouTube*. According to the YouTube statistics of May 2011 there are 48 hours of new video uploaded every minute and more than three billion views per day[1]. Similar enthusiasm can be observed at single player games that permit changes to the game data. We can find manifold user generated game modifications – commonly called MODs – that vary from small changes like more detailed eye textures up to complete packages that can be considered as entirely new games. There are also cases where the community of players took a leading role in patch development that made the game playable in the first place[2].

MMOGs are expected to benefit a lot from UGC, but there are several challenges that prohibited the adoption of this feature so far. One of them is consistency and another one the distribution of the corresponding data – which can easily become an enormous amount – to all users in general. While the consistency problem might also be addressed with

*Funded by FNR Luxembourg.

Simutools 2013, March 05-07, Cannes, France

DOI 10.4108/icst.simutools.2013.252047

[1]Source:`http://youtube-global.blogspot.com/2011/05/thanks-youtube-community-for-two-big.html`
[2]This was the case with Gothic 3, a single player fantasy role playing game released in 2006. Only the community patches removed the most serious bugs and made the game run more or less stable.

Originally from Simutools 2012
DISIO 2012, March 19-23, Desenzano del Garda, Italy

announce		the URL of the corresponding tracker
info		
	piece length	the number of bytes per piece
	pieces	concatenation of the hash values of all pieces
	name	recommended name
	length	the length of the complete file (in bytes)

Table 1: Torrent file structure.

the presented model, it is not subject of this paper. Here we concentrate on the aspect of content distribution.

Recent research considers Peer-to-peer (P2P) techniques in order to improve scalability of highly populated virtual environments, for example in [3] which aims at the distribution of user avatar data (position, movement). Further, P2P approaches are already applied for the distribution of updates and patches and greatly reduce the traffic and server load for the content providers. However, they still leave the Internet service providers with the problem of cost intensive traffic to external provider networks [8] since they do not reduce the corresponding web traffic and even increase the ISP outbound traffic. Therefore this paper presents SatTorrent, a satellite supported peer–to–peer protocol which is a modification of the well known BitTorrent protocol. It is supposed to be able to reduce the number of messages and/or the time needed to transfer the files to a certain number of users as well as the individual download time per user. Whether all or just some of these targets can be reached depends on the parameters used for SatTorrent as well as the number of participants providing satellite reception capabilities. However, by the end of 2010 about 80 million households were able to directly receive satellite broadcasts[3] only in Europe. The following section provides a short introduction to the fundamentals of the BitTorrent protocol. This is followed by a presentation of the general content distribution approach applied in this paper in section 3. Then the SatTorrent protocol is described in detail in section 4. This is followed by the evaluation of the protocol (section 5). It is worth noting that satellite broadcasts suffer from information loss due to the large distance the information has to travel. However this is compensated by deducting a relatively large fraction of the maximal bandwidth for error correction. This measure to ensure data integrity is always accounted for in all considerations within this research.

2. UNDERSTANDING BITTORRENT

As it is common for P2P protocols, files distributed via BitTorrent are partitioned into smaller pieces, These are exchanged between the participating clients, the peers, within the overlay network. In order to distribute a file via bittorrent, a *torrent file* must be created for it. Further every file exchanged with BitTorrent needs a *tracker*. This is a server whose role it is to keep track of all peers downloading a certain file. A tracker can be – and usually is – responsible for several files. The entries provided of a torrent file are shown in table 1.

[3]Source: `http://www.ses.com/4676073/highlights`

info_hash	20 byte hash value of the info part in the torrent file
peer_id	a string providing an ID for the peer
ip	the peer's IP address
port	the port number the peer is listening on
uploaded	bytes uploaded by the peer
downloaded	bytes downloaded by the peer
left	bytes still missing
event	peer status (STARTED, COMPLETED, STOPPED, EMPTY)

Table 2: GetRequest parameters

The torrent file is then given to the potential downloaders. This can happen in any possible way but the common practice is to offer it for download on an Internet server. Users who get the file send a GetRequest to the Tracker provided by its announce entry. The parameters of a GetRequest are shown in table 2.

The tracker responds with a list providing *peer ID*, *IP address* and *port* of a specified number of randomly selected peers from the set of all peers downloading the same file. This response further includes the tracker specific announce interval, indicating the frequency in which update messages should be sent to the tracker. In case the tracker does not receive messages from a peer for more than this interval, the peer is recognized as being retired.

When the requesting peer receives this answer it starts connecting to the peers provided by the tracker's response. Peer connections start with a handshake introducing both peers to each other. In case of the BitTorrent protocol version 1.0, a handshake message's size is 68 byte. All further inter-peer communication uses the *peer wire protocol*. The official protocol has 10 different types of peer wire messages which are shown in table 3. A `bitfield` message is exchanged only immediately after the handshake message. Further updates of pieces available at connected peers are made by means of `have` messages, which a peer sends out to all connected peers when it finished the download of a piece. Based on this information peers send `interested` or `not interested` messages. Further they send `choke` and `unchoke` messages according to their download respectively upload capabilities and the choking algorithm. The latter varies between different client implementations but its aim

bitfield	complete list of available and unavailable pieces
cancel	self-explanatory
choke	prohibit download
have	inform that a certain piece has been completed
interested	announce demand for a piece
keep-alive	self-explanatory
not interested	no demand for the available pieces
piece	subset of a piece as payload
request	request for a specific piece
unchoke	permit download

Table 3: Peer wire protocol message types

is always to avoid free riders – peers who do not upload any data – and to reach a balanced piece replication. Only a peer that has been unchoked by a potential exchange partner will be permitted to download pieces. Interested peers send a `request` message as soon as they get unchoked. The correspondig peer will then answer with a `piece` message which contains a block of data of the corresponding file. Usually, there are always several concurrent requests sent to a peer in order to achieve a good data flow. When a peer has downloaded all pieces, it becomes a *seeder*, which means it will not download anymore but just upload pieces. Peers who are still uploading and downloading are referred to as *leechers*.

3. CONTENT DISTRIBUTION MODEL

The key aspect of this content distribution model is the shift of redundantly transmitted information from unicast messages to a broadcast. Therefore a concurrent access to a broadcast network (satellite) and a unicast network (Internet) is postulated for at least a part of the involved users. A detailed description of this *hybrid network* approach has been given in preceding work [6]. For this paper, it is sufficient to assume that both networks coexist and the transmission can be routed via either of them for each submitted packet. The recipients just receive the messages and are not aware of the delivery channel. Therefore they need a device refered to as the Home Gateway – which is described in [6] and [7] – that accepts packets from both networks and routes them to any device attached to the home network.

In the context of a BitTorrent like P2P exchange, there exist two strategies to reach this target. One is to deliver the actual payload via broadcast, the other is to broadcast the metadata involved with P2P approaches. While we would intuitively expect the most traffic savings from the first mentioned, it also occupies a bigger amount of satellite bandwidth. The latter – broadcasting metadata – is much more difficult to assess. The messages carrying metadata are of small size but exist in high quantities. Thus they should consume only little satellite bandwidth, but it is not obvious how frequently they must be sent. Also no reliable estimation on the potential traffic reductions can be made without exhaustive analyses. Further these messages are not sent in their original form but contain accumulated data of a multitude of single messages. For example instead of sending each peer an individual tracker response containing a list with a limited number of potential candidates for a file exchange, one complete list of all peers downloading the specific file is broadcasted to all participants. This significantly changes the situation, since now the clients can select the subset of peers they want to connect to. By adding additional information to the BTRMs, the knowledge which is the basis for the peer selection algorithm can be considerably increased. Just to give some examples, this additional information might be available bandwidth, the bitfield vectors or location information of peers. The exploitation of the latter might not necessarily reduce the absolute number of messages in the unicast network, but is assumed to reduce the physical distance between exchanging peers. Thus it can reduce the costly inter-ISP traffic that is one of the ISPs major points of critique concerning the P2P technology. While location awareness in P2P protocols has also been studied for application in unicast networks [2], any supplemental information added to messages causes serious problems on that infrastructure. An extra of 10 KB in a unicast response sums up to a tremendous traffic explosion. Let us consider a World of Warcraft update that has a package size of 100 MB, an average download speed for the peers of 512 KB/sec and an announce interval of 300 seconds. Further we assume that only half of the 11 Millions registered WoW players download the update within 7 days after the release and that they stay online after their own download for at least 15 minutes being a seeder. For general P2P file exchanges it would be venturesome to make such an assumption, but here we have a situation where the reason for the download is the desire to play afterwards and then this is rather a conservative playing time estimation. In that scenario, the extra 10 KB would result in approximately 190 GB of additional traffic. In case of broadcasts, even if the responses are sent more frequently, lets say every 15 seconds, the extra traffic is less than 400 MB.

Both strategies, the payload broadcast and the metadata broadcast, are appealing subjects to research and deserve a discretely investigation. This paper focuses on the second strategy, the broadcast of metadata while the payload approach is subject to future work. The following section provides a specification of the SatTorrent protocol and thus will give a better understanding on how it can lead to improved efficiency and traffic reduction.

4. SATTORRENT PROTOCOL

The general procedure of a SatTorrent download is very similar to what we know from the BitTorrent protocol. In fact, when SatTorrent is used on a device which is only connected to a unicast network – denoted as not sat-enabled or with no sat-capability – its functionality is identical to BitTorrent. Therefore this section focuses on the specific aspects of SatTorrent.
The procedure of a peer starting a download is as follows: The joining peer first downloads the torrent file containing, beside other information, the address of the corresponding tracker. The peer then sends a GetRequest to this tracker in order to get a list of potential peers for file exchange. Depending on the satellite capability of the requesting peer, the tracker will respond either via unicast or broadcast. In case of a unicast, the tracker immediately sends a response via Internet which is equivalent to a BitTorrent tracker response. In case the peer is able to receive satellite broadcasts, the tracker will add this peer to the broadcast subscription list for this specific file and schedule a broadcast, if not already scheduled. At this point we differentiate between normal *tracker response messages* (TRM) that are unicasted and the *broadcast tracker response messages* (BTRM). The latter may include additional information (specified in section 4.1.4). In case that a tracker should receive an incorrect request, the corresponding error message is always sent via unicast, since this is individual information and neither of interest nor of use for the other peers.

The further procedure for unicast messages is well known from the BitTorrent protocol, thus only the broadcast case is explained here. When a peer receives a BTRM, it extracts all included information and adds it to its local database. In a next step, the peer starts connecting to other peers. Since it already received a BTRM, usually the bitfields of some

potential exchange partners are available and thus the connection attempts are limited to those peers that are known for being able to provide missing pieces. Just in case that the number of such guaranteed helpful peers is too small, connecting to the remainder is permitted. Like in BitTorrent, all connections start with both peers choked and uninterested and the exchange of a connection handshake. However, this is not necessarily followed by abitfield message, as it is explained in section 4.1.1.

A peer can start downloading from another peer when it has been unchoked. Periodically each peer executes its unchoking algorithm, which follows the same basic rules as the official BitTorrent protocol, but is enhanced by means of a limitation to the best exchange partners that can be determined on the basis of the information received by the BTRM. Criteria for this selection can be, for example, the availability of missing pieces, the distance to the remote peer or it free upload capability. For future SatTorrent protocol versions it is intended to include even more information in the broadcasts that will allow an even better selection of exchange partners. All further features of SatTorrent are described separately in the following subsection.

4.1 Protocol Features

This section describes the features of the SatTorrent protocol. Depending on the given parameters, they may deliver different results with regard to message complexity and distribution time. Therefore, they can be explicitly disabled in order to deliver the best results.

4.1.1 BitField Suppression

This feature allows to avoid the exchange of a BitField between two peers which usually happens immediately after a connection has been established. On the one hand this reduces the number of BitField messages that are exchanged, on the other it speeds up the start of data exchange.
If this feature is active, a handshake message is followed by a peer wire message (PWM) of type *Interested* – instead of a BitField PWM as it is the case for BitTorrent – for those peers that are known to possess missing pieces. The remote peer that receives an Interested PWM does not yet possess a BitField of the sender, it concludes that this peer must have just joined the network and thus has an empty BitField. Only if – in case of an shortage of information about the available peers – a connection has been established with a peer for which the BitField is not known, a PWM of type BitField might be sent that will trigger the remote peer to respond with his own BitField. In that case, an interest message is sent afterwards if appropriate.

4.1.2 Have Message Suppression

This is an existing extension to the BitTorrent protocol which is also implemented in SatTorrent. It avoids sending of have messages to peers that are known to possess the corresponding piece already and thus can not make any use of this information. However, in case they are able to receive broadcast messages, those peers will still receive the updated information with the next BTRM.

4.1.3 Strict Have Message Suppression

Having this feature enabled will restrict the sending of have messages to the tracker that is in charge of that specific torrent and to peers that are not sat-enabled. The only exception from this rule are sat-enabled peers that are currently downloading from the concerned peer. This exception is meant to ensure that they are able to immediately schedule queued piece requests in order to enable continuous downloads. Obviously, in this mode the frequency of broadcast is crucial to the performance of SatTorrent since it determines the time other peers have to wait for updated information. The evaluation in section 5 will show this behavior.

4.1.4 Tracker Response Mode (Simple/Extended)

This feature only affects the broadcast tracker response messages, which differ from the usual BitTorrent responses in a way that they contain more information. A BTRM always contains the ID of the tracker that sent the message. In the standard BitTorrent protocol this is not necessary since the requesting peers knows the trackers identity by the TCP session the response was received. This is not the case when we receive a satellite broadcasted message. Further we differentiate between two types of broadcast tracker responses. For both modes, there is no limit regarding the number of peers, as it was the case for unicast responses. The tracker adds all known peers to the response message.

In the **simple mode**, for each peer there is also its location information, the bytes uploaded by that peer so far for this torrent, the peers satellite capability and the corresponding BitField. The scope is limited to one torrent, thus a broadcast must be scheduled for each file that is downloaded. In this mode, peers can completely ignore BTRMs that are not containing information on files they are currently downloading

In the **extended mode**, all available information for all files that the tracker is responsible for is included in one single broadcast. This includes information on the peers uploaded bytes, downloaded bytes, missing bytes and the BitField per file, the total amount of bytes uploaded, the satellite capability, the AS number, upload and download bandwidth, further whether the peer is seeding files. Beyond improving the selection of exchange partners for the current download(s), this automatically provides peers with information they might need in the future.

4.1.5 Extended Get-Request

Additional to the information included in the standard Get–Request, this message contains a complete description of the peer which includes uploaded bytes, downloaded bytes, missing bytes and the bitfield per file, the satellite capability, the AS number, upload and download bandwidth, and what files the peer is seeding (if any). Since they are relatively large in size, extended Get-Requests should not be sent in the general case but only once in a while when it makes sense to quickly provide the tracker with all this information, e.g. when a peer connects to the tracker for the first time.

4.1.6 Location Awareness

This feature affects the behavior of the SatTorrent client in multiple ways. The location information of the other peers is considered in the unchoking, optimistic unchoking, peer selection for connecting, peer selection for piece requests. If

not enough peers are available within the same AS, a connection might be allowed to distant peers with a certain probability. The latter depends on the amount of connections that can be established without making an exception. The smaller this number, the higher the probability that the exception is made. This ensures that a peer can connect to a sufficient number of exchange partners even if it is alone in its autonomous system. In case we allow multiple trackers for a torrent, the location information might also be used to select the closest tracker.

The information about the location of a network node is included in the BTRMs. For the time being, we utilize the autonomous system number (ASN) to specify the location of a peer. If two peers are within the same AS, they are assumed to be within the same provider network. While this does not reveal too many details about the exact position of a peer – and thus ensures a adequate level of privacy – it allows to keep more traffic within one Internet service provider's (ISP) network which reduces the corresponding cost for the ISP. This may appear of little interest for the protocol, but it is expected to be crucial for its success because it is supposed to avoid blocking or throttling of bandwidth available for P2P exchange. This is a behavior that could be often observed for several ISPs in the past. While there are countermeasures from the P2P community that aim on masking the P2P traffic, keeping both involved parties satisfied is assumed to be the better solution.

At the time being, only two possible 'distances' are used: Within the same autonomous system or not (distance is 1 or 0). This might be changed to a more fine grained distinction of the number of network hops that a message needs to reach the destination.

5. EVALUATION

In order to evaluate the SatTorrent Protocol and to compare its performance to BitTorrent, both protocols have been implemented in a simulation framework. There are some specialties concerning the P2P distribution of user generated data in MMOGs which are shortly described in the next paragraphs before the evaluation results are discussed.

Churn.

The simulation is designed to be able to distinguish between the leaving-probability (churn) of a leecher and of a seeder. This is very important for common peer to peer file exchanges, since usually the reason for users to join the network is their desire to download one or several files. This leads to an increased probability for peers to leave after they finished their downloads, in other words shortly after they became seeders. However, in a MMOG context the user behavior differs from what is described in other P2P studies [1]. In contrast the player's intention is playing the game and they are not even aware of the progress of world data updates respectively their ongoing downloads. As a consequence their decision to leave the game is not depending on their download progress of a certain data set. Consequently both probabilities are set to the same value. It is worth noting that since we assume that players stay online at least for 15 minutes before they leave – which is longer that the time needed to distribute the relatively small files to all players – churn does not emerge in the current simulations.

Data persistency.

It is postulated that a peer after having once downloaded a piece of a file will not loose it again. This also applies for peers that quit and reconnect again later (which does not happen under the simulation settings that have been used). This is also a specialty of the specific use case: A player will never explicitly cancel a download. Either his client software will decide that other game data is needed more urgently or he quits the game. In both cases he will potentially need this data later, when he starts the game again or returns to the corresponding region in the virtual world which makes makes keeping also partially downloaded data a reasonable decision.

During first simulation runs it could be recognized that SatTorrent with all features disabled resulted in a much higher number of overall messages than BitTorrent. The reason was that due to the higher number of available peers, the maximum number of allowed connections was reached much earlier in SatTorrent. In consequence, there were more peers to send have messages and who did send interested/not-interested PWMs accordingly. After some experiments with varying upper limits for open connections it turned out that for SatTorrent – even without its additional features – this limit can be reduced from 80 to 20 open connections per download without negatively affecting the performance. The reason is the improved peer selection. While BitTorrent opens also several connections to "useless" peers, this is not the case for SatTorrent.

Further there was a considerabe delay before any downloads did start when all peers were sat-enabled. The reason was quickly diagnosed: The unicast tracker response messages are instantly sent and the peers can immediately start downloading while BTRMs are depending on the setting of the *TrackerBroadcastInterval* parameter and thus arrive delayed after a get request – at least unless the replies are not constantly broadcasted. This delay hurts the overall performance most during the early phase when only a few peers are connected and just wait instead of contacting the seeder.

This led to a modification of the SatTorrent protocol, introducing an additional parameter that defines a threshold for the requests that a tracker must have received by satellite enabled peers. As long as the number of satellite enabled peers known by a tracker is below that threshold, broadcast messages are still scheduled as before but also an immediate unicast response is sent. This feature is called *Broadcast Bootstrap Accelerator* (BBA) and the corresponding threshold the *BBA threshold* (BBAT). With this modification the aforementioned delay gets eliminated.

After this little protocol modification SatTorrent was examined during extensive simulation runs. All confirmed the expected reduction of total messages with SatTorrent compared to BitTorrent. Figure 1 shows the progression of the total number of messages that are sent from a sender to a receiver in a different autonomous system. These messages should be avoided as far as possible, which is the purpose of the location awareness feature. As the plot shows, this is the case. In the respective simulation only two distinct autonomous systems were available. The higher their quantity, the more peers are needed to give SatTorrent with location

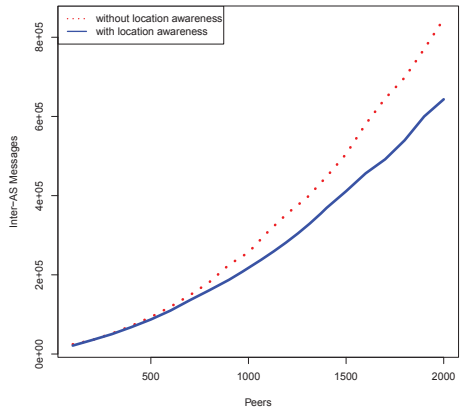

Figure 1: Inter-AS Messages with and without location awareness.

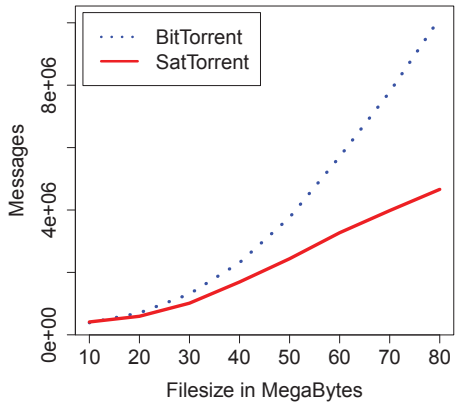

Figure 2: Message counts for varying file sizes.

awareness a significant advantage. The file size used for these simulations was 10 MB. It is worth noting that the advantage of SatTorrent significantly increases as the size of files grows. This is illustrated in figure 2 which shows a comparison of the overall messages that are needed to distribute files of different sizes to 1000 users. In the simulations used for the results of figure 2 for BitTorrent HMS was active while for SatTorrent all its protocol features were enabled.

Figure 3 shows how the quantities of *overall*, *have* and *bitfield* messages change when we enable different features of the SatTorrent protocol and gives a comparison to BitTorrent. The size of the distributed file was again 10MB and the protocol settings were are follows:

(a) BitTorrent without extensions

(b) BitTorrent with Have-Message Suppression (HMS)

(c) SatTorrent with HMS

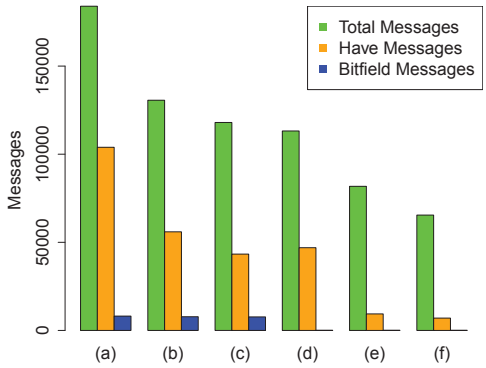

Figure 3: Message counts for varying features.

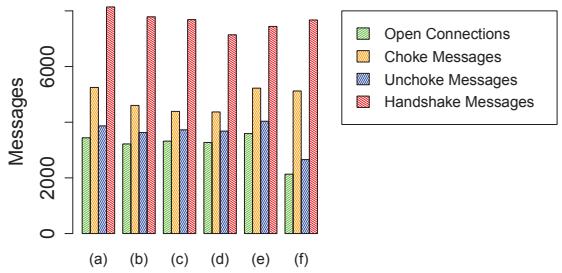

Figure 4: Message counts for varying features 2

(d) SatTorrent with HMS, BitField Suppression (BS), Extended GetRequest (EGR) and Extended Tracker Response Mode (ETRM)

(e) SatTorrent with HMS, BS, EGR, ETRM and Strict Have-Message Suppression (SHMS)

(f) SatTorrent with HMS, BS, EGR, ETRM, SHMS and location awareness

This shows that the more SatTorrent features are enabled, the more the number of messages can be reduced. It might be surprising at a first glance that the location awareness also reduces the overall message count. This is due to the restrictions that come along with this feature. When connecting, unchoking, stating interest, requesting a piece, there is always priority given to local peers and distant peers are restrained. Thus the maximum number of allowed peers is not reached in many cases, which reduces especially the handshake and unchoke messages. It is worth noting that this in some cases slightly increased the time needed to distribute the file to all peers. This retardation is inversely proportional to the number of peers and thus not a problem for highly crowded MMOGs. What can also be observed here that almost no BitField messages are needed when ETRM is enabled. Only those peers who are concerned by the BBA exchanged BitFields at all, the remainder of clients received them by broadcasts. As an important aspect of this observation, this does not notably affect the download duration.

Neither does the SHMS, whose efficiency gets evident also in figure 3. Figure 4 shows the comparison of the same features with regard to different message types. As we can see, not all message types experience a reduction even though the overall message count is decreased. This is an effect that future protocol versions will try to improve on. A parameter of general importance is the broadcast interval. The higher its value is chosen, the higher is the delay of information arrival at the peers. This can seriously degrade the download duration. However, the higher the update frequency the more satellite bandwidth is needed. The simulation results revealed that a good trade off is to set this parameter to a value between 10 and 15 seconds when SHMS is active. Otherwise, values up to 50 seconds do not hurt the performance very much. With one exception, which is during the bootstrap phase. Thus the BBAT must be adjusted accordingly then. Admittedly switching off the SHMS is not a good idea due to its good performance. Further it could be observed that the number of unicast messages can be reduced by more than 80% when connections are consequently restricted to peers that are known for being able to provide pieces. This comes at a certain cost since it increases the distribution time by approximately 30%. However, in cases where time is not a crucial factor, this might be a very valuable option.

6. CONCLUSIONS AND FUTURE WORK

This article presented SatTorrent, a modification of the successful BitTorrent protocol. SatTorrent is supposed to reduce web traffic utilizing a unicast and a broadcast network collaboratively (hybrid network) in order to shift redundant traffic into the broadcast network. The simulation results show that this target can be achieved with SatTorrent by only broadcasting metadata. Thus further improvements can be expected in the future when also payload is transfered via broadcast network. It is expected that this especially will improve the situation during the peak times. According to typical daily activity patterns of users in online games, peaks can be observed especially in the evenings between 8 and 11 pm during the week and starting earlier at the weekends, as has been shown by [5]. Studies in [4] show that the same time–activity correlation applies for online UGC in social networks. Reducing the peak loads is the key to reduce the need for infrastructural upgrades.

Another finding of the simulations was the magnitude of the PWMs of Type *Interested* and *NotInterested*, which make up a significant part of the total messages. Thus it will be analyzed in the future how the protocol behaves when these messages are not send from one peer to another directly if the remote peer is satellite enabled. Instead, the latter extracts the need of other peers and thus their interest status by comparing this information to their own bitfield. Then they include those peers in the choking algorithm as interested peers.

Further future work will take into consideration information we can retrieve from social structures within MMOGs, which is assumed to have a significant influence on the peer behavior. For example for general P2P simulations it is a common practice to use an independent churn probability for each peer. However, if P2P is used to distribute game data to online players the situation is different. Players are commonly organized in communities which are named as guild, kinship e.g. With other words, they are part of a social gaming network. This comes with an increased probability that users within the same community will meet in the virtual world at the same time due to appointments they made. Further the probability is increased that they leave at the same time after the quest is finished. Another implication of the presence of such groups is the fact that those players doing a quest together will need the same world data at approximately the same time. Thus such groups are a potential basis for establishing P2P overlay networks. These social aspects will be considered in future work and are assumed to allow further efficiency improvements.

7. ACKNOWLEDGMENTS

This research is supported by the *National Research Fund* (FNR) of Luxembourg.

8. REFERENCES

[1] V. Aggarwal, O. Akonjang, A. Feldmann, R. Tashev, and S. Mohrs. Reflecting P2P User Behaviour Models in a Simulation Environment. *16th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2008)*, pages 516–523, Feb. 2008.

[2] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPS and P2P users cooperate for improved performance? *ACM SIGCOMM Computer Communication Review*, 37(3):29, July 2007.

[3] M. Esch, J. Botev, H. Schloss, and I. Scholtes. P2P-Based Avatar Interaction in Massive Multiuser Virtual Environments. In *Proceedings of the 1st International Workshop on Virtual Environments and NetworkOriented Applications VENOA 2009*, pages 977–982. Ieee, 2009.

[4] L. Guo, E. Tan, S. Chen, X. Zhang, and Y. E. Zhao. Analyzing patterns of user content generation in online social networks. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*, page 369, 2009.

[5] J. Kim, J. Choi, D. Chang, T. Kwon, Y. Choi, and E. Yuk. Traffic characteristics of a massively multi-player online role playing game. *Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games - NetGames '05*, page 1, 2005.

[6] B. Klasen. Efficient Content Distribution in Social-Aware Hybrid Networks. *Journal of Computational Science*, 2011.

[7] B. Klasen, A. Vinzl, and T. Martinez. Shopping Assistant. In Petros Daras and Oscar Mayora Ibarra, editors, *User Centric Media First International Conference, UCMedia 2009, Venice, Italy, December 9-11, 2009, Revised Selected Papers*, pages 1–4. Springer Berlin Heidelberg, 2010.

[8] B. Sigcomm, D. Choffnes, and F. Bustamante. Taming the Torrent : A Practical Approach to Reducing Cross-ISP Traffic in P2P Systems Intro. *ACM SIGCOMM Computer Communication Review*, 38(4):363–374, 2008.