Using Epidemic Hoarding to Minimize Load Delays in P2P Distributed Virtual Environments

Ingo Scholtes[1], Jean Botev[1], Markus Esch[2], Hermann Schloss[1], and Peter Sturm[1]

[1] Systemsoftware and Distributed Systems, University of Trier, D-54286 Trier, Germany,
{scholtes,botev,schloss,sturm}@syssoft.uni-trier.de,
WWW home page: http://syssoft.uni-trier.de
[2] Faculté des Sciences, de la Technologie et de la Communication, University of Luxemburg, L-1359 Luxembourg, Luxembourg,
markus.esch@uni.lu

Abstract. Distributed Virtual Environments (DVEs) have grown popular in various fields of application. Apart from providing great collaborational opportunities in an immersive setting, large-scale DVEs pose severe scalability challenges. Although P2P approaches have proven to be effective for tackling many of these issues, still load delay problems remain in regions with high object or avatar density. In this article we present and evaluate a hoarding approach that is suitable to minimize such delays in P2P-based DVEs with a real-time distribution of dynamic data. The prediction of what data shall be hoarded is based on an epidemic aggregation algorithm working solely with local knowledge. Evaluation results that have been obtained using a DVE simulation environment will be presented.

Key words: DVE, P2P, Hoarding, Gossiping, Epidemic Aggregation

1 Introduction

The potential of Distributed Virtual Environments (DVEs) for facilitating immersive and intuitive user collaboration is well-recognized. Until now being predominant in the form of Massively Multiplayer Online Games (MMOGs), they are increasingly used for learning or telepresence purposes. While today MMOGs with several thousand concurrent players constitute the largest DVEs, it is justifiable to think about future environments with millions of participants. An interesting example for such a global-scale future DVE is a 3D representation of the real world in which all information that can currently be found in the WWW is embedded. In analogy to Neal Stephenson's novel "Snow Crash", this vision of a collaborative 3D environment as an extrapolation of today's World Wide Web and systems like "Google Earth"[1] is often called "MetaVerse".

[1] http://earth.google.com

The realization of such a MetaVerse scenario poses severe technical challenges. Current MMOGs usually rely on several gigabytes of predistributed data. This approach is not practical when considering future environments with huge amounts of dynamic objects. While the real-time distribution of these data is alluring, it poses severe scalability issues to any centralized infrastructure. Even with today's - comparably moderate - user numbers and predistributed data, server-based DVEs are suffering from scalability problems. In order to overcome these issues, various P2P-based approaches have been proposed over the last few years. While these have effectively addressed server-side scalability issues, in crowded virtual regions there remain problems that result from inevitable limitations of the clients' bandwidth. Having motivated the resulting load delay issues in more detail in the following section 2, we describe a scalable probabilistic solution to these issues in section 3. It has been implemented within the HyperVerse project[2] [3], which investigates a P2P-based infrastructure for global-scale virtual environments that provide a maximum user experience in face of unlimited user numbers, object and avatar densities. The proposed algorithm has been implemented and evaluated in a simulation environment for DVEs. Results will be presented within this article.

2 Motivation

In order to achieve scalability in face of an unlimited amount of objects and participants, it is crucial for P2P-based DVEs to somehow minimize the state required in each component. For this, interest management schemes have been developed. Due to the implicit locality of interest, usually space-based approaches are used. In these, a client's knowledge about objects and other users is limited to a certain surrounding area. In the following we intend to motivate load delay problems that arise from the usage of a dynamic and adaptable space-based interest management scheme. Similar ones are widely deployed in DVEs and suffer from the same problems. The interest management scheme that has been implemented for the HyperVerse infrastructure is based on Euclidean distance and defines two spheres around a user's position p in the virtual world. A sphere with radius d defines the user's Field of View (FoV), i.e. the range within objects and other avatars shall be visible. A second sphere with radius $d + \Delta$ is called Area of Interest (AoI) and represents the range within a client is aware of objects and other users. The motivation for using an AoI is to make clients aware of nearby objects and avatars before they enter the FoV so that there remains enough time to retrieve data necessary for rendering.

We assume that the AoI remains static unless the user moves more than a certain distance Λ away from its center. In this case a new AoI centered around the user's current position will be computed and objects therein will be retrieved. Depending on the client's capabilities, motion speed and object density, the sizes of the FoV and the AoI as well as the threshold Λ need to be adapted. In general,

[2] http://hyperverse.syssoft.uni-trier.de

the AoI size should be as small as possible since it minimizes a client's knowledge. It must however be large enough to provide sufficient time for data retrieval and to keep the refresh frequency at a manageable level.

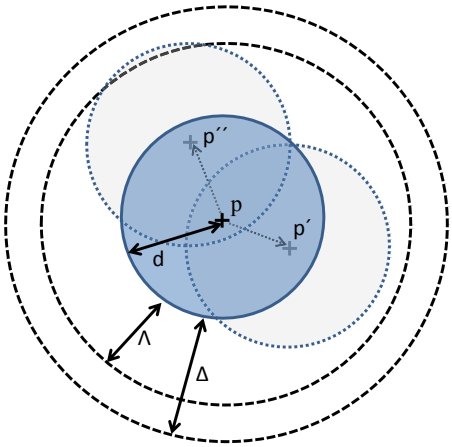


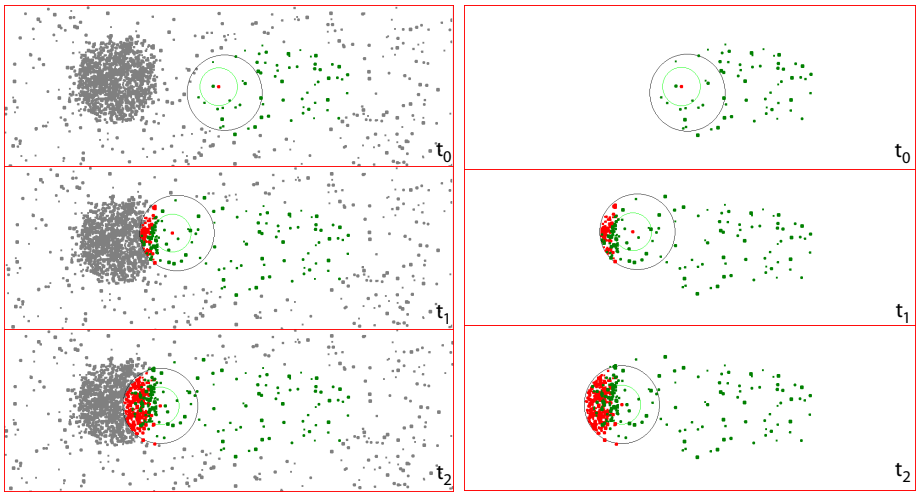
Fig. 1. Area of Interest (AoI) and Field of View (FoV) of a client

Using the interest management scheme described above, load delays may occur when approaching areas with high object or avatar density, so-called hot spots. This situation is illustrated by the following example. Let's assume a client becomes aware of a hot spot within its AoI at time t_1. Let's further assume at t_2 the client has moved to a point where these objects are within its FoV. This situation can be observed in the simulation snapshots in figure 2. The outer circle is the client's AoI, the inner circle represents its FoV.[3]

If - based on the client's downlink bandwidth - the time taken to retrieve data for objects in the client's FoV at time t_2 exceeds $t_2 - t_1$, there will necessarily occur a load delay, even assuming unlimited resources at the data provider. Consequently, the question arises how such situations can be avoided. For this, clients must be made aware of hot spots at time $t_0 < t_1$, so that $t_2 - t_0$ is sufficient to retrieve all relevant data. As can be seen in figure 2(b), at time t_0 the client has however no knowledge about the nearby hot spot.

One possible solution would be to increase the client's AoI, so that the hot spot can be identified at time t_0. Since it conflicts with the minimization of a client's knowledge horizon, this is however not a scalable solution. This becomes especially clear when looking at the dynamics of user movements. A hot spot resulting from a user crowd can exist at the same position for a long time regardless of the dynamics of its constituting users. Tracking a huge number of users

[3] As can be seen in Figure 2(b), the client is aware of some objects outside it's direct awareness radius: These are cached objects that have been discovered on its way towards the current position.



(a) Global View (b) Client's Local View

Fig. 2. Problem Motivation

in a wide range in order to identify such hot spots clearly is not scalable. Since in P2P DVEs there is no central instance with global knowledge the question arises how the "crowd wisdom" of peers can be utilized to solve these issues. In the following section we provide a scalable solution to this question.

3 Epidemic Hoarding

The basic idea that can be used to prevent load delays when entering hot spot regions is to get rid of the inherently bursty traffic pattern that results from the interest management scheme described in section 2: Traffic bursts occur whenever the AoI changes. In the following we will answer the question whether it is possible to distribute traffic across time better and thus equalize these traffic bursts. We use a hoarding mechanism in which a certain fraction of a client's bandwidth is constantly dedicated to speculatively prefetch data regardless of a client's AoI. If this bandwidth is dedicated to selectively prefetch data only from within hot spot areas, load delays can be minimized or totally anticipated. For this, hot spots need to be identified based on a peer's local knowledge. In the following we will show how this information can be efficiently retrieved using a modification of the epidemic algorithm described in [12]. Since for this the P2P overlay topology of the HyperVerse infrastructure is used, we will describe it briefly in the following section.

3.1 P2P Overlay Topology and Data Distribution

For the sake of scalability in the face of a potentially unlimited number of users, within the HyperVerse infrastructure data are exchanged directly between peers whenever possible. For any communication between clients, a P2P overlay topology is used. An interesting property of DVEs is that communication most likely occurs between peers that are close to each other in respect of the virtual geography. Reasons for this include direct avatar to avatar interaction or mutual visibility. Consequently, by maintaining direct overlay connections between those peers, complex routing mechanisms can be avoided. Besides supporting scalability, a positive effect of this is that no deterministic network structure needs to be maintained. This allows for high peer dynamics and avoids network maintenance overhead.

The P2P topology of the HyperVerse infrastructure is built in a way that guarantees direct connections between all peers with intersecting AoIs. Such peers will most likely have a common interest in data, can potentially collaborate via an object present in both of their AoI and might thus be required to communicate. In order to make real-time distribution of data scalable, clients in the HyperVerse infrastructure will first ask peer neighbors whenever data from within their AoI needs to be retrieved. Only if this fails, data distribution will fall back to a distributed backbone service. For more detailed information on this data distribution scheme and how neighbor discovery works, we refer to [3].

3.2 Epidemic Hot Spot Aggregation

In this section we will present an algorithm that utilizes the P2P overlay described above in order to identify hot spots within a certain range based on a client's local knowledge. The algorithm is based on the concept of epidemic aggregation that has been described in [11]. The main advantage of this scheme is that it does not require any network structure, which makes it suitable for networks with highly dynamic constituents. In order to describe the algorithm in more detail, we rely on the following definitions. They are based on the number of bytes that need to be transmitted via the network in order to retrieve model and texture data of a given object or a user's avatar.

Definition 1 *Let p_i be a client with n renderable objects and users within its AoI. With $s_1, \ldots, s_n$ denoting their transmission sizes in bytes, we define the mass M_i of the client's AoI as $M_i = \sum_{j=1}^{n} s_j$. With $r_1, \ldots, r_n$ being the objects' (or users') positions in virtual space, a peer p_i's center of mass C_i is defined as $C_i = \frac{\sum_{j=1}^{n} r_j \cdot s_j}{M_i}$.*

The basic idea of our epidemic hot spot aggregation approach is to let peers exchange information on their local center of mass. For this, we assume that each peer p_i keeps a local fixed-size vector S_i of "most crowded spots". Each

entry (C_k, M_k) in S_i consists of a peer p_k's AoI mass M_k and its center of mass C_k. Furthermore, each peer p_i defines a maximum lookahead distance L_i which determines the range in which it desires to "search" for the biggest hot spots.

The algorithm requires each client to be aware of a fixed number of neighbors (according to the overlay topology described above) along with their AoI and their lookahead radii. Each peer p_i applies the basic epidemic scheme as described in [12], i.e. in random intervals information contained in S_i will be exchanged with a random neighbor p_j in the overlay. The following algorithm describes the aggregation scheme when information is exchanged between two peers p_i and p_j:

1. p_i selects the entry (C_k, M_k) from its local vector S_i with $dist(C_k, p_j) < L_j$ and maximum mass M_k. If there is no such entry or the mass M_i of p_i's current AoI is bigger than M_k, (C_i, M_i) will be sent to p_j, otherwise (C_k, M_k) will be sent.
2. On reception of an entry (C_l, M_l), p_j will add this entry to the size-constraint vector S_j, possibly replacing an existing entry with smaller mass. It will then select the entry (C_k, M_k) from its vector S_j with maximum mass M_k and with $dist(C_k, p_i) < L_i$. If such an entry does not exist or if the mass M_j of p_j's current AoI is bigger than M_k, the local information (C_j, M_j) will be sent back to p_i. Otherwise (C_k, M_k) will be sent.

The scheme resembles the decentralized gossip-based maximum aggregation as proposed in [12] but it has been extended by some additional rules. First of all information that is aggregated is dynamic, since a client's AoI as well as the objects and avatars within (and thus the total mass as well as the center of mass) can change at any time. In order to account for this dynamics, on each gossip iteration clients check whether the mass of their current AoI is bigger than their currently known maximum. Furthermore, range constraints have been introduced: By having a client check whether the maximum center of mass falls within the random neighbor's lookahead range, only information from within the client's lookahead will be aggregated.

The main advantage of using epidemic aggregation is that is has proven to produce fast-converging results in highly dynamic networks [12] with constant communication cost. In our case, each client will retrieve a fixed-size set of most-crowded places that are within its lookahead radius and which are known to some peer in its connected component. For this, only a small number of information exchanges is required. Since only aggregated fixed-size information on peer and objects density and "masses" are exchanged in periodic intervals, the bandwidth consumption of the algorithm is constant. The main contribution is that a client can efficiently retrieve aggregate information without being burdened with detailed dynamic information on objects or users within the aggregated area.

3.3 Hoarding of Hot Spot Data

The information that has been retrieved by the algorithm described in the previous section can be utilized by clients in order to prevent load delays. The aggregate mass and the center of mass can be consulted by clients in order to estimate

the required bandwidth and loading time and thus mitigate delays. Client-side information like movement trajectories, speed and favorite venues can be used in order to predict which of the hot spot areas will be most probably accessed in future.

All this information can be used to speculatively prefetch relevant data from within nearby crowded areas outside a client's direct AoI. For this purpose, clients can dedicate a certain fraction of bandwidth. The maximum fraction of a client's bandwidth that shall be used for hoarding and the client's lookahead radius need to be adapted to each other: A larger lookahead radius provides for smaller hoarding bandwidth, since it will give more time for prefetching data. It is important to note that there is no additional communication cost associated with an increased lookahead radius. Arguments about which lookahead sizes are suitable to be used in practice are currently under investigation.

4 Evaluation Results

In order to implement and evaluate the proposed hoarding scheme, comprehensive simulation support is required. For this, TopGen [15] - a topology generator created within our working group - has been extended by DVE simulation facilities. For realistically simulating large-scale distributed virtual environments it provides a deterministic, event-based simulation of user movement based on different mobility models. Besides the random way point model, a DVE-specific model has been created which will be described in more detail in the following section. Apart from users with simulated movement, objects with different transmission sizes can be placed in the virtual world. By this means one can realistically simulate the resulting object retrieval traffic for different mobility patterns, interest management and prefetching schemes.

An important feature of TopGen is the possibility of extending it by so-called experimental modules. By hooking into certain simulation events (join and exit of clients, avatars becoming mutually visible, refreshing of the AoI, etc.), simulations can easily be extended with own code. For the evaluations in this section we have created an experimental module which implements interest management, object retrieval and epidemic hoarding as described in sections 2 and 3. In each simulation step, every peer was allowed to transfer only a limited amount of data in order to simulate client-side bandwidth limitations.

4.1 Preferential Way Point Mobility Model

An important aspect when performing DVE simulations is the choice of a realistic model for user mobility. The most simple mobility model known e.g. from the field of ad hoc networks is random way point [14]. In this model each simulated agent selects a random point with uniform probability. The agent then starts moving towards this point and selects a new target once having reached it. This model is not realistic because it has no notion of Points of Interest (PoIs). Since

it seems intuitive that users move to PoIs with higher probability, we propose a variation of the random way point model that respects objects, users and their densities. For this we assume that DVE users are collaborative by nature, i.e., that they are primarily interested in interacting with objects and other users.

Whenever a simulated user in our "preferential way point" mobility model selects a new movement target, it will select one among all possible object and user positions with a probability proportional to the object and user density in a surrounding region. Therefore users will preferentially move to hot spot regions. The name has been chosen in resemblance of the "preferential attachment" generation model for power law graphs [1]. In this model newly added nodes create links to a existing nodes with probability proportional to the target's link number. While here every new edge will further increase the "attractiveness" of a node, a similar effect occurs in our mobility model: Each user that moves towards a certain PoI will increase the probability that other users will go there as well and thus further its attractiveness.

4.2 Simulation Results

In this section we provide simulation results of the hoarding scheme that have been retrieved using the TopGen simulation environment. A background set of 200 objects has been randomly distributed in a virtual region of 1000 x 350 pixels in size. Additionally two PoIs consisting of 400 objects each have been added at random positions. Synthetic data transmission sizes between 1 and 5 units have been randomly assigned to objects and user avatars. 200 randomly distributed users moving according to the preferential way point model have been simulated. In each simulation step, each of the simulated clients could retrieve a maximum of 15 of the synthetic data units from a fictional data provider. Furthermore each client was using a limited cache that could take at most 3000 data units. As described in section 3.1, clients with intersecting AoIs were interconnected via an edge in the overlay topology. The initial distribution of objects and peers in the simulated setting is shown in Figure 3.

A simulation consisting of 1250 iterations has been performed. For all simulations, the interest management scheme described in section 2 has been used with a FoV radius of $d = 50$ pixels, an AoI radius of $d + \Delta = 75$ and $\Lambda = 12$ pixels. The motion speed of peers has been set to a maximum of 1 pixel per simulation step, i.e. the AoI of a client was refreshed at most every 12 simulation steps. The simulated traffic arising from object retrievals has been recorded for each peer. In order to capture and evaluate situations with visible load delays, the amount of object data from within the FoV which could not be loaded in time has been recorded for each peer in every simulation step. This value can be used to identify situations as motivated in section 2, i.e. where the client's bandwidth was not sufficient to retrieve data in time. The following paragraphs present results that have been obtained in a single random client. The initial position of this client can be seen in Figure 3.

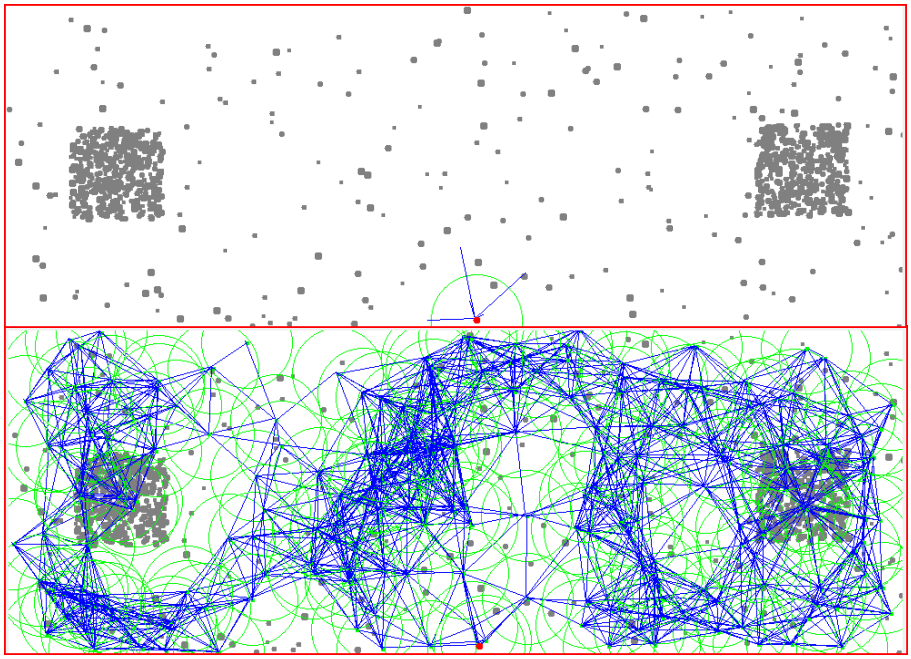


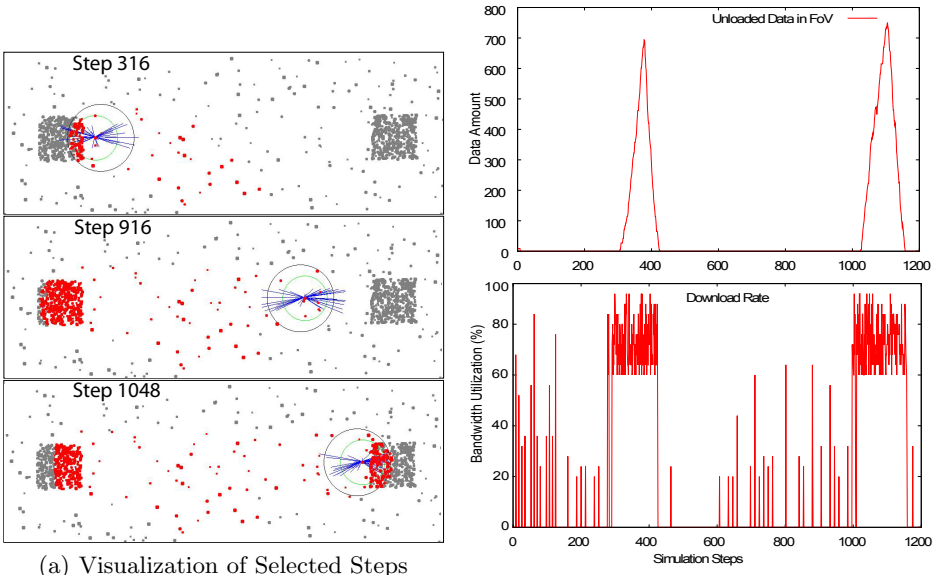
Fig. 3. Initial Situation with all Clients and P2P Overlay (Bottom) and Selected Peer only (Top)

Simulation without Epidemic Hoarding : Figure 4 shows simulation results of a random client with epidemic hoarding disabled. For the sake of clarity only the selected client is shown in Figure 4(a) at three selected simulation steps. Links to peers in the overlay topology (i.e. nearby clients with intersecting AoIs) are indicated by edges. Based on the preferential way point mobility model, the selected client first moved towards the hot spot which can be seen in the left part of the simulated area in Figure 4(a). Having reached this in step 316 of 1250, it proceeded to the hot spot that can be seen in the right part of the simulated area. The client reached this hot spot in step 1048. A video of the simulation can be found at the website of one of the authors[4].

Looking at the random client's bandwidth that is utilized for retrieving data within its AoI, in the bottom part of Figure 4(b) one recognizes numerous peaks which occurs whenever the AoI changes. Usually - because the AoI is bigger than the FoV - there remains enough time to load objects coming into the FoV shortly. Accordingly there are no objects with unloaded data in the client's FoV. This is the case during the first 315 simulation steps and can be seen in Figure 4(b). In step 316, the client's FoV enters the crowded area. Due to the density of objects, data could not be retrieved in time. The amount of pending data is shown in the upper part of Figure 4(b). One clearly recognizes two

[4] http://syssoft.uni-trier.de/~scholtes/VideoA.avi

peaks when the client entered the two hot spot areas in simulation steps 316 and 1048. The maximum value of pending data was 750 units. Based on the maximum download bandwidth of 15 units per simulation step, in this situation at least 50 additional time steps would have been required to load all objects within the FoV in time. In a real setting this would have resulted in visible load delays. Another effect than can be seen in Figure 4(b) is that, beginning from simulation step 280 until it has left the hot spot area in simulation step 410, the client's available download bandwidth is saturated. In reality this might affect other communication protocols (chat, avatar interaction, etc.) and thus degrade responsiveness.



(a) Visualization of Selected Steps

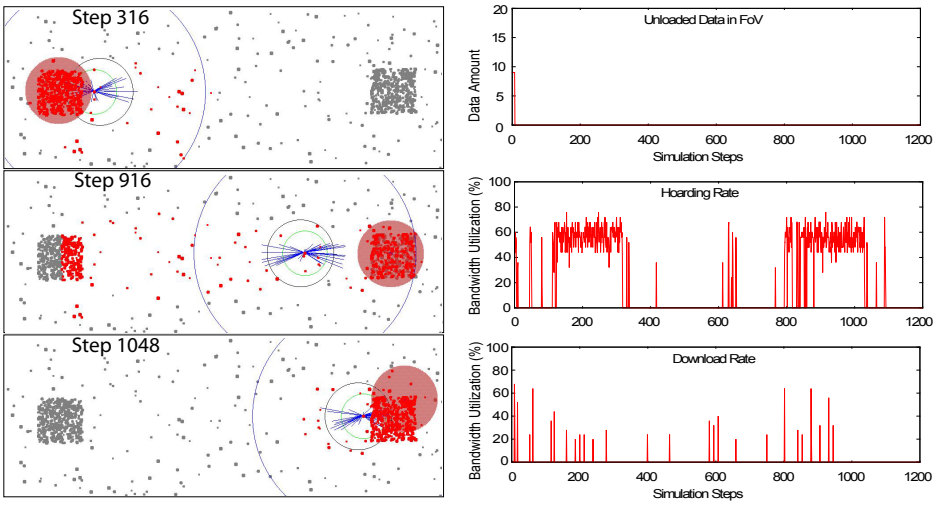
(b) Object Retrieval Traffic and Pending Data Units

Fig. 4. Simulation Results without Epidemic Hoarding (Lines are drawn to guide the eye)

Simulation with Epidemic Hoarding : Figure 5 shows the results of another simulation run for the same client using the same random seed. Here the epidemic hoarding algorithm described in section 3 has been enabled. No client-side information (like e.g. movement trajectory) has been used to improve the prediction which data shall be preloaded, i.e. all data from within predicted hot spot have been prefetched. Pending data transfers resulting from interest management have been prioritized, i.e. only otherwise unused bandwidth was used for hoarding. Clients were using a lookahead range of 5 times their AoI size. The maximum bandwidth utilization for hoarding has been set to 70 %. In case of a full client

cache, an additional rule was used which disables the hoarding of objects that are farther away than all objects in the cache. This prohibits thrashing situations in which hoarding displaces entries in the cache that are immediately needed for rendering. Again, a video of the simulation can be retrieved from the website of one of the authors[5].

Figure 5 shows results obtained in this simulation. In figure 5(a), the same selected simulation steps as in 4(a) are visualized. The lookahead radius is indicated by the outermost circle around the selected client. The client's local hot spot prediction resulting from epidemic aggregation is visualized by the shaded circle. Due to hoarding, all data have already been retrieved when the client enters the left hot spot in step 316. As soon as the right hot spot is in the client's lookahead range in step 916, the hot spot is correctly identified by the epidemic aggregation algorithm and hoarding of data begins. Finally, when the client's FoV enters the hot spot region in step 1048, again all data have already been retrieved and no load delays occur.



(a) Visualization of Selected Steps

(b) Object Retrieval Traffic

Fig. 5. Simulation Results with Epidemic Hoarding (Lines are drawn to guide the eye)

These claims are substantiated by the results shown in figure 5(b). The middle diagram shows the rate at which data have been speculatively prefetched. Comparing both diagrams to Figure 4(b), one recognizes that the amount of time at which the client's bandwidth is saturated could be significantly minimized. A positive side-effect of this is that it leaves more resources for communication unrelated to object retrieval. The most interesting result, the amount of unloaded data in the client's FoV, is shown in the topmost diagram. It shows that due

[5] http://syssoft.uni-trier.de/~scholtes/VideoB.avi

to epidemic hoarding, load delays could be avoided since there are at no time objects with unloaded data in the client's FoV[6].

Although these results look promising, an important question is the overall additional cost resulting from data hoarding. Consequently, the integral amount of data retrieved in both simulation has been recorded for the client considered above. The result is shown in Figure 6. One recognizes a roughly 11% increase in the total amount of data that have been transferred. An important question is how much of this difference results from situations in which a client not using epidemic hoarding leaves a hot spot before all objects have been loaded. This results in any queued downloads being canceled, thus underestimating the amount of total data that needed to be transferred. A further evaluation has shown that these situations account for a difference of 94 data units between both simulated scenarios. After accounting for these canceled transfers, the overall amount of transferred data is roughly 9% higher when using epidemic hoarding.

	Data Transferred	Unloaded Data in FoV	Canceled Transfer Data
without hoarding	5794	85109	94
with hoarding	6462	0	0

Fig. 6. Integral Transfer Statistics

5 Related Work

Several P2P-based approaches to improve scalability of large-scale distributed environments have been proposed during the last couple of years. Some of the existing approaches include a specific handling of non-uniform object and avatar distributions. The P2P framework ATLAS [13] introduces a user-specific object popularity that is based on repeated accesses and which is used to improve prefetching and caching. There are however no means to mitigate load delays for hot spots that have not yet been accessed or that form dynamically.

Most solutions to problems related to hot spots - or flash crowds - have been developed with a focus on the data provider side. Dynamic partitioning schemes like the one presented in [5] can e.g. be used to equally distribute objects or avatars from densely populated regions among a set of peers or servers. In the context of P2P-based DVEs, the approach presented in [17] combines octrees with the Chord [16] protocol to achieve the same task. In order to maintain a minimum quality of service for at least a limited number of users in server-based MMOGs, [7] propose an "early warning system". It can be used to detect hot spots by monitoring performance degradations. In case a hot spot is identified, no more users are admitted to the identified crowded regions.

[6] The reason for the small peak in the first simulation step is, that in the initial situation the client was set to a position where some objects were in its FoV already.

None of the aforementioned approaches addresses the problem of the limited bandwidth of clients that will most likely lead to load delays regardless of the data provider's bandwidth. Although some work has been done in this area as well it mostly focuses on traffic resulting from mutual avatar visibility. [2] deals with avatar interactions in crowded DVEs and introduces a scheme that aggregates individual avatars to crowds in order to maintain scalability. Another active field of research are scalable interest management schemes for large-scale DVEs. A survey of different schemes for MMOGs has been performed in [4]. [8] describes a group-based filtering mechanism that minimizes movement updates of nearby avatars in crowded areas of a DVE. It does however not consider delays resulting from real-time data loading. Based on interest management, a common practice is to reduce the AoI size when object or avatar density reaches a certain threshold. This technique is used for example in the Voronoi-based clustering that can be found in VON [10]. The usage of adjustable AoI shapes is another direction of research, which is e.g. being investigated for the P2P DVE infrastructure FLoD [9].

Another way of minimizing load delays is to predict a user's movement. This information can then be used to prefetch data from those regions that will be accessed with highest probability. A prediction that is based on the user's mouse movements has e.g. been investigated in [6].

6 Conclusion and Future Work

In this article we have presented an efficient probabilistic and localized approach to identify hot spot regions in DVEs that rely on real-time data distribution. It can be used to enrich interest management with hot spot predictions, so data can be speculatively hoarded before clients are too close for data still being loaded in time. The only knowledge required for this prediction is the local list of objects and avatars in the client's AoI. Another advantage is the fact, that neither user mobility nor peer dynamics constitutes a problem, since epidemic aggregation makes no assumptions about the structure of the overlay topology: Aggregated information on hot spots will remain stable even though the constituents of this hot spot can be highly dynamic. Distant clients are not burdened with this dynamics.

A disadvantage of existing prediction-based approaches is the fact that they do not consider hot spots specifically. Rather than requiring an exact prediction of user mobility, our approach efficiently provides those regions that will potentially incur load delays. Prioritizing hot spots retrieved by this means is considerably easier than obtaining an exact prediction of the user's movement.

From our simulations we draw the conclusion that the proposed scheme can efficiently be used to avoid load delays in DVEs with real-time data transfer and thus improve user experience. In the simulated setting the hoarding scheme resulted in a 9 % increase of a client's total data traffic while totally eliminating load delays. It remains to investigate how local knowledge like movement

trajectories, motion speed or favorite venues can be used to further improve predictions and thus minimize this overhead. Since motion traces of avatars from existing DVEs are slowly becoming available, we plan to investigate the usage of such heuristics as well as the viability of our mobility model.

Another future direction is the optimization of what regions are treated as hot spots by clients. For this, we plan to apply a second epidemic aggregation algorithm which efficiently determines the average data density within the virtual world. If this is known to all clients, only those hot spots that significantly exceed average density can be advertised by gossiping and thus improve prediction quality. While the scheme currently only works for locality resulting from a continuous avatar movement, we currently investigate extensions that also consider avatar relocation through teleportation. Finally, another direction for improvements is the consideration of additional information while aggregating information: When moving in a certain direction, e.g. gossip messages received from peers in this direction can be prioritized.

Ultimately, it has to be investigated whether interest management as a whole can be replaced in favor of a self-organizing and probabilistic prediction similar to the hot spot identification presented in this article. For this a fixed fraction of bandwidth could be dedicated to hoarding while a self-organizing process ensures that all required data are available in time.

References

1. A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
2. S. Benford, C. Greenhalgh, and D. Lloyd. Crowded collaborative virtual environments. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 59–66, New York, NY, USA, 1997. ACM.
3. J. Botev, M. Esch, A. Hoehfeld, H. Schloss, and I. Scholtes. The hyperverse - concepts for a federated and torrent based "3d web". In *Proceedings of the First International Workshop on Massively Multiuser Virtual Environments at IEEE Virtual Reality 2008*, March 2008.
4. J.-S. Boulanger, J. Kienzle, and C. Verbrugge. Comparing interest management algorithms for massively multiplayer games. In *NetGames '06: Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games*, page 6, New York, NY, USA, 2006. ACM.
5. C. Bouras, E. Giannaka, and T. Tsiatsos. Partitioning of distributed virtual environments based on objects' attributes. In *DS-RT '07: Proceedings of the 11th IEEE International Symposium on Distributed Simulation and Real-Time Applications*, pages 72–75, Washington, DC, USA, 2007. IEEE Computer Society.
6. A. Chan, R. W. H. Lau, and B. Ng. Motion prediction for caching and prefetching in mouse-driven dve navigation. *ACM Trans. Interet Technol.*, 5(1):70–91, 2005.
7. X. Chen and J. Heidemann. Flash crowd mitigation via adaptive admission control based on application-level observations. *ACM Trans. Interet Technol.*, 5(3):532–569, 2005.
8. S. Han, M. Lim, and D. Lee. Scalable interest management using interest group based filtering for large networked virtual environments. In *VRST '00: Proceedings*

of the ACM symposium on Virtual reality software and technology, pages 103–108, New York, NY, USA, 2000. ACM.

9. S.-Y. Hu. A case for 3d streaming on peer-to-peer networks. In *Web3D '06: Proceedings of the eleventh international conference on 3D web technology*, pages 57–63, New York, NY, USA, 2006. ACM.
10. S.-Y. Hu, J.-F. Chen, and T.-H. Chen. Von: a scalable peer-to-peer network for virtual environments. *IEEE Network*, 20:22–31, 2004.
11. M. Jelasity and A. Montresor. Epidemic-style proactive aggregationin large overlay networks. In *Proceedings of The 24th International Conference on Distributed ComputingSystems (ICDCS 2004)*, pages 102–109, Tokyo, Japan, 2004. IEEE Computer Society.
12. M. Jelasity, A. Montresor, and O. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(3):219–252, 2005.
13. D. Lee, M. Lim, and S. Han. Atlas: a scalable network framework for distributed virtual environments. In *CVE '02: Proceedings of the 4th international conference on Collaborative virtual environments*, pages 47–54, New York, NY, USA, 2002. ACM.
14. G. Lin, N. G, and R. Rajaraman. Mobility models for ad hoc network simulation. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 1, 2004.
15. I. Scholtes, J. Botev, M. Esch, A. Hoehfeld, H. Schloss, and B. Zech. Topgen - internet router-level topology generation based on technology constraints. In *Proceedings of the First International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools)*, February 2008.
16. I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160, New York, NY, USA, 2001. ACM Press.
17. E. Tanin, A. Harwood, and H. Samet. Using a distributed quadtree index in peer-to-peer networks. *VLDB*, 16(2):165–178, 2007.