Community Discovery for Interest Management in DVEs: A Case Study

Emanuele Carlini[1(✉)], Patrizio Dazzi[1], Matteo Mordacchini[2], Alessandro Lulli[3], and Laura Ricci[3]

[1] ISTI, National Research Council (CNR), Pisa, Italy
{emanuele.Carlini,patrizio.dazzi}@isti.cnr.it
[2] IIT, National Research Council (CNR), Pisa, Italy
matteo.mordacchini@iit.cnr.it
[3] University of Pisa, Pisa, Italy
ricci@di.unipi.it

Abstract. An efficient interest management is a fundamental requirement to support Distributed Virtual Environments (DVEs). As avatars move across the virtual environment, they often forming *communities* by gathering around hotspots. Distributed community discovery is a research area that gained momentum in the last years. In this paper we propose a case study evaluation on the impact of communities and community discovery on a distributed gossip-based interest management architecture for DVEs. Our experimental evaluation shows that communities have a positive impacts on interest management, at the expense of a small computational and communication overhead.

1 Introduction

Distributed Virtual Environments (DVEs) are geographically distributed applications in which multiple users interact simultaneously in a shared virtual environment. A common trait of DVEs architectures is the distribution of the state of the virtual environment to the client machines. Many distribution approaches have been proposed over the years [20], ranging from unstructured solutions [12,21] to structured ones [3,15]. The fundamental requirement arising from the distribution of the state is making sure that each client machine has its view of the virtual environment up-to-date. This requirement is commonly called *Interest Management* (IM), as the interest of each client is represented by only a specific portion of the whole virtual environment. A large numbers of approaches have been proposed to efficiently and effectively support IM (see [23] for a comprehensive list) in a distributed fashion. One of the most commonly used strategies to support IM is to keep up-to-date only the portion of the virtual environment in the visual/interaction Area of Interest (AoI) of the participants. AoIs can overlap one each other, especially in presence of hotspot, i.e. popular areas of the virtual environment. In a sense, when many participants gather together in the proximity of an hotspot, they form a *virtual community* whose interest is related to the events happening around the hotspot. In such scenarios, it is interesting

S. Hunold et al. (Eds.): Euro-Par 2015 Workshops, LNCS 9523, pp. 273–285, 2015.
DOI: 10.1007/978-3-319-27308-2_23

to understand if the identification of such communities can be of any support to the management of the IM. In our context, the entities participating to the communities are the virtual agent of the users of the virtual environment (the so called *Avatars*). Due to the distributed nature of the DVEs, we are interested in approaches that perform community discovery in a fully distributed fashion, i.e. by only exploiting the information local to each entity and without the help of any central authority.

The aim of this paper is to evaluate the impact on DVEs of communities and the process of communities discovery. In a previous work [5], we provided preliminary results of an initial attempt to introduce communities in a distributed IM approach. Here, we provide a more organic and structured approach, which aims to provide an answer to the following research questions: (i) can communities identify in a timely manner the hotspots in a virtual environment? (ii) will the inclusion of community leaders in the process of information diffusion benefit to the effectiveness of the IM? (iii) what is the impact in terms of computational and communication overhead of the process of community discovery?

To answer these questions, we considered a solution for the management of IM that employs a set of gossip protocols for the dissemination of the updates in the virtual environment [7]. We integrated this solution with two algorithms for distributed community discovery: (i) GROUP, a gossip-based community discovery approach, and (ii) a gossip-based version of the AffinityPropagation algorithm. We conducted an experimental analysis by means of simulations, which considered two different mobility models. Results showed that the inclusion of communities improves the performance of IM when there is a tendency of avatars to spread in the virtual environment (i.e. to gather less frequently around hotspots), at the expense of an acceptable computational and communication overhead.

2 Related Work

Many DVE architectures and protocols have been proposed over the years [23]. A number of these approaches focused on solving distributed interest management by exploiting structured and unstructured Peer-to-Peer (P2P) solutions.

Approaches based on structured P2P (often based on Distributed Hash Tables) offer a stable and reliable network for the distributed dissemination of the state of the avatars [3,13,15]. However, these solution are often based on a distributed index, which needs to be maintained causing latencies and disruption in the interest management. Compared to structured approaches, unstructured P2P solutions focus on building the overlay according to the spatial proximity of the peers in the virtual environment [12,21]. In order to perform IM, each peer connects with a subset of its neighbours so that peers may warn each other both about their movements and about new peers entering their AoI. These approaches naturally adapt to the rapid evolving scenarios of DVEs and allow for large scalability, but introduce overhead in the complexity of the approach and make strong assumptions on the capability of peers. To combine the advantages of both structured and unstructured P2P solutions, we proposed an hybrid

solution based on a combination of a centralized server and a best-effort mechanism providing support for distributed interest management [5]. In this paper we refer to this architecture as the *reference architecture* and it is presented in Sect. 3.

Community discovery is a well studied research problem that has applications to many research fields [17]. In general, a community is defined as a set of entities that share similar interests, which are encoded into a profile, one for each entity. Often associated to the concept of community is the concept of *exemplar* or leader of a community, which is the entity that best represents a community. Some popular solution for distributed community discovery, like CDC from Ramaswamy et al. [19], have the issue that their division of the nodes in clusters is highly dependent on the choice of the starting nodes. Other solutions, like USP2P from Datta et al. [8], assume that interests are uniformly distributed among the peers. However this assumption not necessarily holds in our scenario. For these reasons, for our evaluation we selected two solutions that do not require starting points and perform no strong assumptions, namely AffinityPropagation [10], described in Sect. 4.2, and GROUP [2,9], described in Sect. 4.1.

To the best of our knowledge, there are few gossip-based solutions for hotspot detection in DVE. [22] proposed an approach based on attraction of avatar according to a *mass* criterion. Similarly, [4] developed a gossip-based protocols that exploits a flock model to identifies cluster of avatars and the relative super peers. Both these solutions share many similarities with our approaches, and provide an experimental validation for gossip-based protocols in DVEs. However, we differentiate as we strongly focus on the concept of community, and their ability to approximate hotspots. In addition, the concept of exemplar of a community is slightly different from the concept of super peers. In particular, the latter directly participates to the topology of a network, rather the former represents a community and not necessarily is part of a defined topology.

3 Reference Architecture

The reference architecture considers an IM model where the information is delivered by the combination of an unstructured P2P network and a static server. A detailed presentation of this architecture can be found in [6,7]. Figure 1 presents an overview of the reference architecture. The server works as a regular VE server, i.e. receives information from the clients and in turn periodically informs the client with fresh information about the state of the virtual world. The P2P network is based on multiple layers of gossip protocols. The gossip networks are built and maintained in a purely distributed fashion, without any intervention of the server. The purpose of the P2P network is to give the possibility to increase the period of the server communications with client. In other words, the idea is to reduce the load on the server, and put more load on the set of peers. With a reduction of communication interval, the server can manage more clients in the same region and/or manage a larger region. For the sake of simplicity we consider a single server serving a close region of a generic DVE. Note that our

approach applies also to the multi server case, provided that each client knows from which server to communicate with. Clients communicate with the server and among each other to receive update of the content of their AoI.

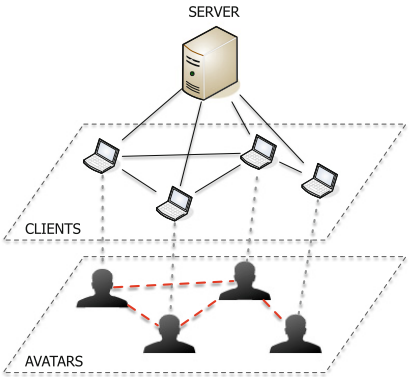


Fig. 1. The general architecture

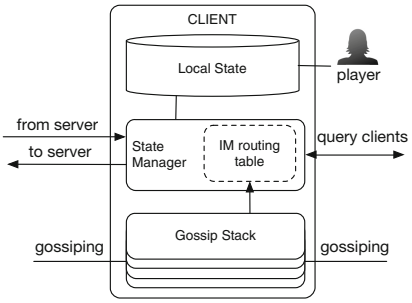


Fig. 2. Components of a client

The component architecture of a client is presented in Fig. 2. The local state contains all the entities that are relevant for the avatar. Players can interact with them by modifying their and other entities' state. The state manager cares to forward these update to the server. Beside that, the other task of the State Manager is to keep the local state up-to-date. To perform this operation it exploits two mechanisms: the periodic feeding from the server and by querying other clients. These two mechanism affect each other, as the more efficient the querying of client is, the lower can be the frequency of the server feeding. However, since the state manager cannot query all the clients, the choice of which client to query must be limited in number and effective.

To provide an effective set of clients to query is indeed the task of the gossip stack component. As its name suggests, this component is composed by a stack of gossiping protocols, each one with a specific purpose. Each protocol feeds with information the protocols above and receive information from the protocols below. In a generic case this information can be whatever; since our ultimate goal is to find an effective set of clients, the information passing through the layers are client descriptors. Beside the connectivity information (such as the IP address), a client descriptor contains the information about the avatar of a particular client, namely: (i) the avatar id, (ii) the avatar's position, and (iii) a time stamp of creation.

By passing peer descriptors across layers is possible to refine the selection of peers. Also, a layered protocol model simplifies the composition of each layer. In our case the upper layers runs the *coverage peer sampling* protocol, which has been initially presented in [6]. We provide more details about the coverage peer sampling in Sect. 3.1. Running CPS alone would result the system to fall

into a local minimum solution, and eventually peers may be disconnected from the network. Instead, by exploiting the layered architecture, the bottom layers can provide fresh information that can be used to introduce an amount of wise randomization in the selection of the peer. It is clear that the choice of protocol, especially if community oriented, to execute below CPS have an impact on the capacity of the system to perform interest management. This impact is evaluated experimentally in Sect. 5.

3.1 Coverage Peer Sampling

The goal of the *Coverage Peer Sampling* (CPS) is to provide the best selection of peers that maximise the retrieval of entities in the virtual environment. To clarify its purpose, let us explain the CPS from the point of view of a generic avatar A.

At an arbitrary point in time, A has in its local representation of the virtual environment the replicas of the entities that belong to its AoI. When A travels across the virtual environment, to maintain the content of A's AoI up-to-date, the peer (P) represented by A must discover the new entities belonging to the new AoI. Being in a fully distributed context, P searches for the peers that can efficiently provide such information. The criterion driving the CPS is the area coverage, which is defined as the following. Given a set S of peers and a pivot peer P, the area coverage can be defined as the intersection between the union of the AoIs of the peers in S and the AoI of P. Intuitively, a peer would maintain a view that maximises the coverage of its own AoI, so to have higher chances to obtain relevant information about entities in its proximity.

Therefore, this layer gossips with other clients to obtain a view that maximizes the area coverage defined above. This task is carried out by means of two functions: with *whom* and *what* to gossip. Let us consider a generic peer P. If P has some peers in the AoI, it selects the one that maximises the euclidean distance with his position. Conversely, the algorithm selects the peer that minimises the euclidean distance with the current position of P. The rationale behind this behaviour is that by choosing a peer at the borders of its AoI increases the knowledge of a region that could be explored in the close future. Once T has been selected, P evaluates the peer in its AoI with respect to the position of T. The evaluation is an heuristics that partitions the AoI's areas into a set of discrete tiles. and assigns a score to each tile equal to the reciprocal of the number of intersected AoIs. The heuristics then computes the score of a peer as the sum of the scores of each intersected tiles. We covered in full details the effectiveness and the cost of the heuristics in [7].

4 Distributed Community Discovery Protocols

The stack of gossip protocols that feed the CPS have a direct impact on the quality of the interest management. In particular, in terms of the reference architecture, providing proper information to the CPS can improve the area coverage and in turn, reduce the frequency of the communication with the server.

In general, the best information for the CPS are those avatars whose AoIs maximize the area coverage of the client's avatar. In this context, as the avatars gather around in the proximity of the hotspots, they can be seen as communities of avatar. Identifying these communities in real time would improve the results of the CPS, in particular by considering the *exemplar* of the communities. In order to be fully integrated in the reference architecture, the community discovery algorithm must be fully distributed and capable of be implemented as a gossip protocol. To this end, we created an enhanced architecture that improve the reference architecture with a gossip layers of community discovery, which run AffinityPropagation or GROUP.

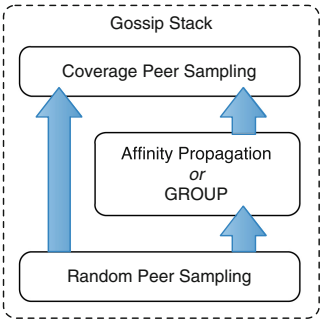


Fig. 3. The gossip protocols stack in the enhanced architecture

Figure 3 shows the stack of the gossip protocol in the enhanced architecture where a new layer for community discovery has been added to the stack. Note that, despite the addition of the community discovery layer, the CPS is still fed by the random peer sampling, for reasons of connectivity and for avoiding to rely only on communities in situations when peers are not gathered into hotspots.

4.1 GROUP

GROUP is a protocol that exploits a gossip-based collaborative process to cooperatively build communities in a P2P network. In GROUP, peers try to identify the nodes that are best suited to represent a community using a distributed voting scheme. Each node expresses a set of votes (i.e. endorsements to potential community representatives) to the most similar peers, among the ones it knows. The number of votes received by each peer represents the consensus achieved by the peer as a community representative. Eventually, the peers that has received the largest amount of votes are elected as representatives and, together with the peers that contributed to their election, form a community. The profile of a representative is used as the community exemplar, and in our case the profile correspond to the position of the avatar in the virtual environment.

The representatives election mechanism is divided in three steps:

1. *Similar Peer Detection*: In this phase each peer selects and votes its k most similar other peers taken from the peer's neighbourhood. This preliminary voting phase will be used in the next step to determine the most central peers in the similarity-based overlay.
2. *Potential Candidates Selection*: After the completion of the previous phase, each peer enters a second phase, in which it exploits the information coming from the previous step. At this point, each peer has to select a potential community representative from its neighbourhood. A peer chooses as a potential representative the neighbour that has received a number of votes higher than a representative threshold θ_r. Then, it "endorses" (i.e., it sends a *potential candidate* vote) the chosen neighbour as a potential representative.

3. *Representative Election Phase*: Finally, in the last phase, a peer p chooses as its representative the peer that has gained the highest number of *"potential candidate"* votes. In case two or more neighbours has received the same number of votes, p chooses the one with the most similar profile to its own. It then uses the profile of the chosen representative as its community identifier.

One remarkable feature of the GROUP protocol is that peers in a community are not required to keep track of their community structure. Peers generally do not have any explicit knowledge about the composition of their own community. The flow of votes during the election mechanism is the only interaction and information required by GROUP.

4.2 Affinity Propagation

AffinityPropagation [10] is an algorithm for clustering data that derives from the belief propagation [14] model. The aim of AffinityPropagation is to discover the best exemplars from a set of data points, namely the points that are more suited for representing a whole set of points. The algorithm is based on a message passing approach, which makes it suitable to be applied in a distributed context.

Starting from its original formulation, we implemented a gossip-based version of the algorithms that has been integrated into our reference architecture. The information needed to discover avatar communities in are organized in three matrix-like data structures: similarity, availability and responsibility. The *similarity* matrix contains the information about the suitability of an avatar j to serve as the exemplar for another avatar i. The similarity between avatars is computed as the negative Euclidean distance between the location of the avatars in the virtual environment. The *availability* matrix counts the messages sent from candidate exemplars to potential members of the community, indicating how appropriate that candidate would be as an exemplar. Finally, the *responsibility* matrix counts the messages sent from community members to candidate exemplars, indicating how well-suited the avatar would be as a member of the candidate exemplar's community.

AffinityPropagation differs from other clustering algorithms such as k-means or k-medoids: these algorithms begins with an initial set of randomly selected exemplars, and iteratively refine this set. aiming at reducing the sum of squared errors between the exemplars and the points they represent. Conversely, Affinity Propagation does not require any a-priori knowledge about the number of community to be built and the quality of the result it provides does not depend by the initial selection of potential exemplars.

5 Experimental Evaluation

We conducted our experimental evaluation by using PeerSim [18], considering a virtual environment composed by a squared region with a side of 1200 points. The AoIs of the avatars have a radius of 50 points. The region has 10 circular

hotspots, whose radius is 100 points. Each simulation is divided into 1000 steps, and we set to perform a periodic feeding from the server every 10 steps. 1000 avatars move on the map according to the two following mobility models.

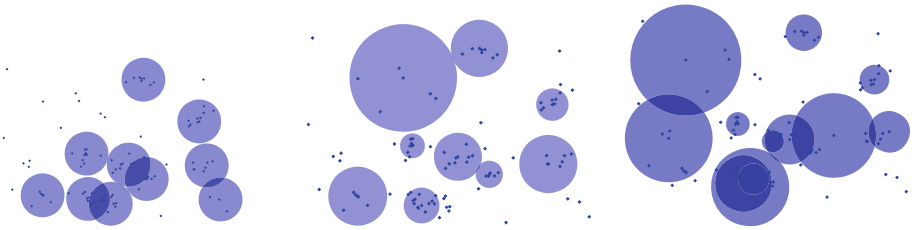
- BLUEBANANA, presented by Legtchenko et al. [16], which simulates avatars movement in a commercial MMOG, Second Life [1]. When an avatar reaches a hotspot, it explores the hotspot for some time, and eventually it moves to another hotspot. This mobility model exposes a fair balance between the time spent by avatars in hotspots and outland, and there is no predefined path connects two hotspots;
- Random Way Point (RWP) presented by Hong et al. [11], and initially thought to evaluate the impact of mobility in ad-hoc wireless network. In RWP each avatar moves independently toward a random chosen way point (the hotspots in our case). As soon as an avatar reaches an hotspot it stops there for a random time interval, and afterwards, it chooses another random hotspot.

5.1 Hotspots Approximation

An interesting analysis is to understand how GROUP and AffinityPropagation identify the clusters of players as they gather around the hotspots in the virtual environment. In a sense, an hotspot can be seen like a community of avatars with an interest in a specific area of the DVE. In our setup, community discovery mechanisms can only provide an approximation of the hotspots, mainly for two reasons: (i) in a DVE, not all avatars belong to an hotspot at any given time, while communities usually contain all entities, and (ii) we defined hotposts as circle having fixed radius, while communities can vary in size. To conduct this analysis, we exploited the BB mobility model as in this model the avatars tends to cluster more than with RWP.

The three images in Fig. 4 represent a snapshot of the system, i.e. the position of the avatars at a given point in time. Figure 4a shows the hotspots as circles, while the points represents the avatar. In Fig. 4b and c communities are represented as circles, having the centre with the coordinates of the exemplar avatar, and the radius is equal to the average distance of the avatars participating to the community from the exemplar. From the figures it can be seen that the communities algorithms behave differently, and it can be noted a tendency of GROUP in building larger communities with respect to AP.

In order to study the relationship between communities and hotspot over time, we defined *hotspot approximation* as the percentage of avatars whose closest hotspot is the same than the hotspot associated to the exemplar of the avatar's community. Figure 5 shows the hotspot approximation for GROUP and AffinityPropagation. From the figure, it is evident that AffinityPropagation achieves a better approximation than GROUP. In particular, GROUP has a very stable value of the approximation, indicating that is not reactive enough to sustain the movement of the avatars. By comparison, the communities of AffinityPropagation seem more dynamics, allowing for a better approximation of the moving avatars.



(a) HotSpot (b) Affinity (c) Group

Fig. 4. A visual representation of hotspots approximation during a simulation cycle

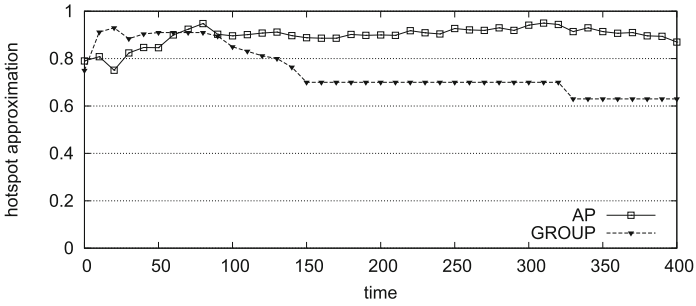


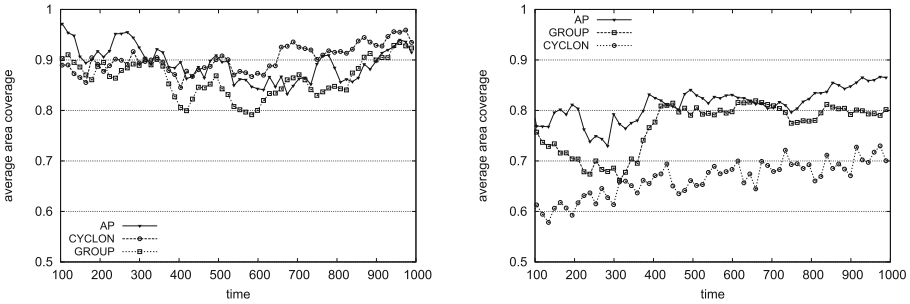
Fig. 5. Approximating hotspots: AffinityPropagation vs GROUP

5.2 Area Coverage

This section compares the impact of GROUP and AffinityPropagation on the interest management. To measure this impact we used the *average area coverage* (AAC). As a baseline, we consider a basic version that just employs a random peer sampling. We computed the AAC by averaging the area coverage of all avatars, computed as defined in Sect. 3.1 and considering the peers in each peer's view.

Figure 6a and b show the AAC with respectively the BLUEBANANA and RWP mobility models. The results of the two models are different, and highlight interesting insight on the impact of communities on the interest management. When considering BLUEBANANA, all the versions behave similarly. The version equipped with just the RPS obtains the best performances, even if the difference is very marginal with the other versions. When considering the RWP instead, the RPS version obtains by far the worst performance, and the AffinityPropagation achieve the best results.

These results suggest a sort of counter-intuitive behaviour when adding community discovery to the reference architecture. When avatars are naturally more clustered (as it happens in BLUEBANANA) communities have a little impacts, rather when avatars are more distributed in the virtual environment (as in RWP) the communities improve the quality of the interest management. In other words, when avatars stay closer to each other, the random peer sampling is enough to



(a) Average Area Coverage with BLUEBANANA

(b) Average Area Coverage with RWP

Fig. 6. Comparison of average area coverage

discover potential peers that contains useful information for interest management. When avatars are more scattered, a random strategy is not enough and the identification of the communities identifies such peers that contain useful information.

5.3 Message Number and Computational Overhead

In this section we compared how GROUP and AP would impact on the network and the CPU of peers. Specifically the following metrics were used for the analysis: (i) *message number*: the number of message sent by each peer in a step of the simulation; (ii) *computational overhead*: the relative time spent by the coverage peer sampling protocol to merge its view and the views provided by the additional layers and to calculate the AoI coverage.

Table 1 provides an overview of the results, considering BLUEBANANA and RWP. Each value represents the average taken out of 1000 steps, starting from step 100. For the computational overhead is also shown the standard deviation.

In terms of computational overhead we observed that GROUP spent more time in computing the coverage peer sampling. We believe this is a direct consequence of the fact that in GROUP the communities tend to be larger than in AffinityPropagation, as can be observed in the analysis of Fig. 4c. In addition, for the BLUEBANANA mobility model the computational overhead for GROUP is doubled with respect to RWP due to the fact that in BLUEBANANA peers tend to create higher density near hotspots. The analysis of the number of messages shows two interesting insights. First, GROUP requires lesser messages than AP to organize the peer in communities. Second, for AP the number of messages depends neither on the particular mobility models or the specific cycle of the simulation. Conversely, the number of messages of GROUP depends on the mobility models, having less messages when the avatar are more spread thorough the virtual world.

The presented results suggest that there exists a trade off between message number and computational overhead between AP and GROUP. On one side,

Table 1. Comparison of Message number and Computational overhead

		Comp. overhead	Message number
BLUEBANANA	AffinityPropagation	34.56 ± 8.60	32
	GROUP	87.79 ± 23.88	20
RWP	AffinityPropagation	30.09 ± 6.52	32
	GROUP	48.93 ± 10.87	18

GROUP has an higher computational overhead (and an higher standard deviation), however, it uses around 40 % messages less than AP. If analysed together with the results on area coverage discussed in Sect. 5.2 we can conclude that AP would be the protocol of choice, unless the number of messages is a critical parameter of the infrastructure managing the DVE.

6 Conclusion

Whether community discovery can improve the quality of interest management in a DVE is an interesting research question. In this paper we provided an empirical analysis in the effort of answering this question. Even if our experiments are limited to a particular reference architecture, we have found that the introduction of community discovery mechanisms actually can improve interest management especially when there is the tendency of avatars to spread around the virtual environment. Also, this comes with an acceptable computational and communication overhead. To understand if community discovery would have an impact on different scenarios, in the future we plan to investigate further distributed community discovery protocols (e.g. [4,22]). In addition, we plan to evaluate these protocols in combination with other popular solutions that perform IM in DVEs.

References

1. Second life website. http://secondlife.com/, Last Accessed on 6 May 2015
2. Baraglia, R., Dazzi, P., Mordacchini, M., Ricci, L., Alessi, L.: GROUP: a gossip based building community protocol. In: Balandin, S., Koucheryavy, Y., Hu, H. (eds.) NEW2AN 2011 and ruSMART 2011. LNCS, vol. 6869, pp. 496–507. Springer, Heidelberg (2011)
3. Bharambe, A., Douceur, J., Lorch, J., Moscibroda, T., Pang, J., Seshan, S., Zhuang, X.: Donnybrook: enabling large-scale, high-speed, peer-to-peer games. ACM SIGCOMM Comput. Commun. Rev. **38**(4), 389–400 (2008)
4. Botev, J., Rothkugel, S.: Flora: flock-based resource allocation for decentralized distributed virtual environments. In: Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques, pp. 257–264. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) (2011)

5. Carlini, E., Dazzi, P., Mordacchini, M., Ricci, L.: Toward community-driven interest management for distributed virtual environment. In: an Mey, D., Alexander, M., Bientinesi, P., Cannataro, M., Clauss, C., Costan, A., Kecskemeti, G., Morin, C., Ricci, L., Sahuquillo, J., Schulz, M., Scarano, V., Scott, S.L., Weidendorfer, J. (eds.) Euro-Par 2013. LNCS, vol. 8374, pp. 363–373. Springer, Heidelberg (2014)
6. Carlini, E., Ricci, L., Coppola, M.: Reducing server load in MMOG via P2P gossip. In: Proceedings of the 11th Annual Workshop on Network and Systems Support for Games, p. 11. IEEE Press (2012)
7. Carlini, E., Ricci, L., Coppola, M.: Integrating centralized and peer-to-peer architectures to support interest management in massively multiplayer on-line games. Practice and Experience, Concurrency and Computation (2014)
8. Datta, S., Giannella, C., Kargupta, H.: Approximate distributed k-means clustering over a peer-to-peer network. IEEE Trans. Knowl. Data Eng. **21**(10), 1372–1388 (2009)
9. Dazzi, P., Felber, P., Leonini, L., Mordacchini, M., Perego, R., Rajman, M., Rivière, É.: Peer-to-peer clustering of web-browsing users. In: Procedings LSDS-IR, pp. 71–78 (2009)
10. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315**(5814), 972–976 (2007)
11. Hong, X., Gerla, M., Pei, G., Chiang, C.C.: A group mobility model for Ad hoc wireless networks. In: Proceedings of the 2nd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp. 53–60. ACM (1999)
12. Hu, S.Y., Chang, S.C., Jiang, J.R.: Voronoi state management for peer-to-peer massively multiplayer online games. In: 5th IEEE Consumer Communications and Networking Conference, 2008, CCNC 2008, pp. 1134–1138. IEEE (2008)
13. Kavalionak, H., Carlini, E., Ricci, L., Montresor, A., Coppola, M.: Integrating peer-to-peer and cloud computing for massively multiuser online games. Peer-to-Peer Networking and Applications, vol. 8(2), pp. 1–19. Springer, New York (2013)
14. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Trans. Inf. Theor. **47**(2), 498–519 (2006)
15. Kulkarni, S., Douglas, S., Churchill, D.: Badumna: a decentralised network engine for virtual environments. Comput. Netw. **54**(12), 1953–1967 (2010)
16. Legtchenko, S., Monnet, S., Thomas, G.: Blue banana: resilience to avatar mobility in distributed MMOGS. In: 2010 IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 171–180. IEEE (2010)
17. Malliaros, F.D., Vazirgiannis, M.: Clustering and community detection in directed networks: a survey. Phys. Rep. **533**(4), 95–142 (2013)
18. Montresor, A., Jelasity, M.: PeerSim: a scalable P2P simulator. In: Proceedings of the 9th International Conference on Peer-to-Peer (P2P 2009), pp. 99–100. Seattle, WA, September 2009
19. Ramaswamy, L., Gedik, B., Liu, L.: A distributed approach to node clustering in decentralized peer-to-peer networks. IEEE Trans. Parallel Distrib. Syst. **16**, 814–829 (2005)
20. Ricci, L., Carlini, E.: Distributed virtual environments: from client server to cloud and P2P architectures. In: 2012 International Conference on High Performance Computing and Simulation (HPCS), pp. 8–17. IEEE (2012)
21. Ricci, L., Genovali, L., Carlini, E., Coppola, M.: Aoi-cast in distributed virtual environments: an approach based on delay tolerant reverse compass routing. Practice and Experience, Concurrency and Computation (2012)

22. Scholtes, I., Botev, J., Esch, M., Schloss, H., Sturm, P.: Using epidemic hoarding to minimize load delays in P2P distributed virtual environments. In: Bertino, E., Joshi, J.B.D. (eds.) CollaborateCom 2008. LNICST, vol. 10, pp. 578–593. Springer, Heidelberg (2009)
23. Yahyavi, A., Kemme, B.: Peer-to-peer architectures for massively multiplayer online games: a survey. ACM Comput. Surv. (CSUR) **46**(1), 9 (2013)