Scalable Data Management Using User-Based Caching and Prefetching in Distributed Virtual Environments

Sungju Park, Dongman Lee, Mingyu Lim, and Chansu Yu
Information & Communications University
58-4 Hwaam-dong, Yusung-ku
Taejon 305-732, Korea
Tel. +82-42-866-6163, Fax. +82-42-866-6222
{lazz, dlee, cats, cyu}@icu.ac.kr

ABSTRACT

For supporting real-time interaction in a distributed virtual environments (DVEs), it is common to replicate virtual world data from the server at the client. For efficient replication, two schemes are used together in general – prioritized transfer of objects and a caching and prefetching technique. Existing caching and prefetching approaches for DVEs exploit the spatial relationship based on the distance between a user and objects. However, spatial relationship fails to determine which types of objects are more important to an individual user, not reflecting the user's interests. We propose a scalable data management scheme using user-based caching and prefetching exploiting the object's access priority generated from spatial distance and individual user's interest in objects in DVEs. We also further improve the cache hit rate by incorporating user's navigation behavior into the spatial relationship between a user and the objects in the cache. By combining the interest score and popularity score of an object with the spatial relationship, we improve the performance of caching and prefetching since the locality between the user and objects are reflected in addition to spatial locality. The simulation results show that the proposed scheme outperforms the hit ratios of existing caching and prefetching by 20% on average when the cache size is 2% of virtual world database.

Keywords

Distributed virtual environments, Scalable Data Management, User-Based Caching and Prefetching, Distributed Virtual Environments

Scalable Data Management Using User-Based Caching and Prefetching in Distributed Virtual Environments

Sungju Park, Dongman Lee, Mingyu Lim, and Chansu Yu
Information & Communications University
58-4 Hwaam-dong, Yusung-ku
Taejon 305-348, Korea
+82-42-866-6163
{lazz, dlee, cats, cyu}@icu.ac.kr

ABSTRACT

For supporting real-time interaction in a distributed virtual environments (DVEs), it is common to replicate virtual world data from the server at the client. For efficient replication, two schemes are used together in general – prioritized transfer of objects and a caching and prefetching technique. Existing caching and prefetching approaches for DVEs exploit the spatial relationship based on the distance between a user and objects. However, spatial relationship fails to determine which types of objects are more important to an individual user, not reflecting the user's interests. We propose a scalable data management scheme using user-based caching and prefetching exploiting the object's access priority generated from spatial distance and individual user's interest in objects in DVEs. We also further improve the cache hit rate by incorporating user's navigation behavior into the spatial relationship between a user and the objects in the cache. By combining the interest score and popularity score of an object with the spatial relationship, we improve the performance of caching and prefetching since the locality between the user and objects are reflected in addition to spatial locality. The simulation results show that the proposed scheme outperforms the hit rate of existing caching and prefetching by 20% on average when the cache size is 2% of virtual world database.

Keywords

Distributed virtual environments, Scalable Data Management, User-Based Caching and Prefetching, Distributed Virtual Environments

1. Introduction

In a distributed virtual environment (DVE), a user explores a virtual world by browsing and interacting with objects in the world. For supporting real-time interaction, it is common to replicate virtual world data from the server at the client. As the size of the virtual world becomes large, it becomes significant transmission overhead, especially when to replicate the whole virtual world to the client. To reduce the overhead, on-demand transmission techniques [1, 3, 8, 14] are proposed. Instead of downloading the whole virtual world objects into the clients' machine, copied are only objects that the user needs - partial replication of a virtual world rather than full replication [11]. Therefore, a key aspect is how to efficiently replicate the required data lest that user's immersion in virtual world is disturbed for the loss of data. For efficient replication, two schemes are used together in general – prioritized transfer of objects and a caching and prefetching technique. Prioritized transfer of objects filters objects within a user's viewing range and transmits only the objects which provide high fidelity to the user using LODs (Level of Details), or multi-resolution techniques [1, 3, 8, 14]. It maximizes the graphical fidelity of world objects as well as interactive performance by mediating the graphical detail and the transmission overhead of the objects. Caching and prefetching make the demanded data immediately available in a timely manner by exploiting the locality of data even when the client's access pattern changes over time. Since the performance of caching and prefetching depends on the priority of cached data, accurate prediction of a user's behavior for the prioritization of world objects is an important factor to scalable data management in DVEs.

Existing caching and prefetching [3, 8, 14] approaches for DVEs exploit the spatial relationship based on the distance between a user and objects. The spatial relationship is formed using the observation that the nearer the object lies to the user, the more probability it has to be accessed again [3]. However, the existing approaches have some drawbacks not efficiently coping with a scalability problem. The spatial relationship used in the existing approaches just guesses the user's behavior from the proximity between user and objects, so it fails to determine which types of objects are more important to an individual user, not reflecting the user's interests – a significant factor affecting the user's behavior. Since the diverse types of objects become existent as the number of objects in a virtual world increases, it is more difficult to correctly predict the user's behavioral pattern according to the types of objects.

In this paper, we propose a scalable data management scheme using user-based caching and prefetching exploiting the object's access priority generated from spatial distance and individual user's interest in objects in DVEs. The proposed scheme leverages the locality resulted from the interactions between a user and objects during user's navigation in a virtual world, so called "user-based". We assume that the behavioral pattern of a user is explained using the user's interest in objects in the world. To incorporate the user's interest into access priority of objects, we leverage the fact that a user tends to repeatedly visit objects interesting to it or highly popular objects likely attracting it. To enumerate the level of interest and popularity of an object, we introduce two values, the interest score and popularity score of an object, respectively. The interest score of an object is set per user and represents how much the user expresses its interest to the object. The popularity score of an object is set per world and represents how many people in the world express their interest to the object. By combining these two values with the spatial relationship, we improve the performance of caching and prefetching since the interaction locality between the user and objects are reflected in addition to spatial locality. Interest and popularity scores are determined by the number of access times for a given object. For further improvement of cache hit rate, we incorporate user's navigation behavior into the spatial relationship between a user and the objects in the cache. We observe that a user usually alternates a navigation mode between wandering and moving. An object residing at the opposite side of the user's moving direction should have the same priority as an object residing at the user's moving direction in case of a wandering mode since it implies the possibility of rapid rotation. Apparently the former object should have higher priority than the latter object in case of a moving mode. This provides more accurate information for caching and prefetching. The simulation results show that the proposed scheme outperforms hit ratios of existing caching and prefetching by 20% on average when the cache size is 2% of virtual world database.

The rest of the paper is organized as follows. Section 2 examines the previous works related to the data management, especially caching and prefetching in distributed virtual environment. Section 3 describes the detailed description of our scalable data management scheme. Section 4 presents the experimental result. Conclusion follows in Section 5.

2. Related Works

In this section, we identify and discuss the data management schemes to improve the interactive performance by reducing the data transmission overhead in DVEs.

Several efforts have been proposed for data distribution management in DVEs by reducing the amounts of world data distributed to the client sides. While most existing systems, such as DIVE [2] and SIMNET [12], use full replication of the whole world data to the clients, recent works try to replicate the partial data of whole database to the clients as they need, i.e., on-demand transmission [1, 3, 8, 14]. Though on-demand transmission of world data merits in that it doesn't need the complete copy of the world data at the client, repetitive data requests and transmissions between the server and the client become a major burden to networks. It thus results in degradation of interactive performance at the client. To solve the problem, two schemes are proposed: prioritized transfer of objects, and cache replacement and prefetching.

2.1 Prioritized Transfer of Objects

Prioritized transfer of objects improves the graphical fidelity and interactive performance by allowing objects in a virtual world to have multiple representations using LODs or multi-resolution technique [9] and an application to choose the best one that mostly contributes the fidelity of user experience [1, 3, 8, 14]. While the detailed representation of an object gives high graphical fidelity to users, it results in longer data transmission time so that users might demand it as low graphical fidelity for quick response time. So the mediation between two is required based on the priority of objects according to the user's fidelity experience. In [3, 8, 14], motivated from awareness management [6, 13], they use 'aura' as a basis for prioritizing the objects to determine whether objects should be rendered in detail. Based on the distance between the user and objects, the nearer objects have more detailed graphical representation, resulting more network transmission. In QUICK [1], task-specific asset prioritization is proposed. It assumes that the fidelity of a user might be different per his or her task. A task-dependent annotated importance value is added to the priority of an object for maximizing the individual user's task dependent fidelity..

2.2 Cache Replacements and Object Prefetching Technique

While the efforts for prioritized transfer of objects is recently active, there exist a few research dedicated to caching and prefetching reflecting the characteristics of DVEs. Since cache replacement and object prefetching eliminate the possibility of unnecessary network transmission of duplicate data, it also significantly affects the interactive performance of DVEs.

Similar to prioritization in transferring objects, spatial distance based on the distance between the user and objects is included for cache replacement algorithm in MRM (Most Required Movement) [3]. Compared to a traditional LRU replacement scheme, MRM shows the better performance in the situation where the strong spatial coherence is existent [4] in applications like a virtual walkthrough . Since this method ignores individual user's interest in objects, it does not support efficient cache replacement and prefetching adapting various behavioral pattern of users, as the number of users increases.

QUICK proposes a task dependent importance value as a means to incorporating a user's behavioral pattern into cache replacement and prefetching [1]. Since each object has a task dependent importance value, the application can make a "user-based" decision when the object is replaced from or prefetched to cache. However,

3. Data Management Using User-Based Caching and Prefetching

3.1 Design Principles

To provide scalable data distribution management, we leverage real world heuristics to our scheme. First, each user has its own interest in objects residing at a virtual world and access these objects with the same interest repetitively. Second, the objects that many users tend to frequently access has a high chance of being accessed again. These two findings are based on the locality generated by the interactions between the user and objects in DVEs.

For example, in a virtual shopping mall, every user has its own interest. Some users want to buy clothes, or some users does to look around a book store. Although these interests are diverse per each user and vary in time, they are eventually converged to some set of interests when it comes to individual user at a given period. So the prioritization of objects having high interest values with each user is achievable per user basis. It is used as a basis for caching and prefetching. While it is possible to classify an object into a highly or low interested object according to the user's interest, it is also possible to classify an object into a highly or low popular object according to its access popularity. High popularity means that the object has been accessed by lots of users, i.e., most users get interest with this object. The popularity value of each object can be used as one of criteria for caching and prefetching instead of the user's interest value in the case that the interest value is not applicable. Typical situations that object's popularity has precedence over user's interest are two cases – one is that a user is a newcomer to a virtual world, and another is that a user navigates the newly updated portion of world data. Both interest and popularity values attached to each user and an object represent the strong indication of a user's access pattern independent of the spatial distance between the user and the object. Thus, combined with spatial relationship, it would produce an enhanced caching and prefetching algorithm.

Another observation of a user's movement pattern is that a user alternates two modes, a moving mode and a wandering mode, of an action at some point. When we apply the spatial relationship to a replacement algorithm, we should decide whether angular distance is included in it or not. Chim et al. [3] assume that angular distance is an important factor affecting a cache replacement algorithm. But contradictory observations [1, 14] argue that considering the possibility of rapid rotation of user movement, ignoring angular distance is more reasonable for determining the priority of objects. To apply these two conflicting observations into the proposed scheme, we consider a wandering mode as well as a moving mode. During a wandering mode, the user wanders about some place - moves any direction possible, so rapid rotation is also assumed resulting in ignoring angular distance for calculating a cache replacement algorithm. During a moving mode, the user moves to some destination, so integrating angular distance into caching decision affects the performance of a cache replacement algorithm. These two modes are exactly the same as a user behavior in a virtual world. For example, in a virtual shopping mall, a user would wander about a specific location to buy something, or simply to look around or to chat with another user. Then, the user would move to another place and wanders repeatedly. Actually, it is possible to assume that user alternates these two modes in all the virtual environment applications .

With these findings based on the user's movement observation, we designate the user-based cache replacement and object prefetching algorithms to reflect the user's interest. To convert the real user's interest for objects into the values of interest and popularity score of our scheme, we maintain the world database per world for popularity values and the user database per user for interest values. Whenever a user navigates a virtual world, we identify the objects that the user observes at that time. They are reflected to the database by updating their access count. In such a manner, interest and popularity values in the database become the reflection of user's interests of user and object's popularity.

In the following sub-sections, we describe the factors that affect the priority of objects and present the proposed scheme – caching and prefetching mechanism.

3.2. Access Priority of Objects

A cache replacement algorithm selects objects that will be least likely accessed in near future while a prefetching mechanism does objects that will be most likely accessed in near future. So if we can prioritize every object accurately according to some criteria, caching and prefetching mechanisms become more efficient. The formula below is used in the proposed scheme for calculating access priority of a given user and object. Access priority between user A and object O, *AP(A,O),* is defined as

$$AP(A,O) = \alpha DS(A,O) + (1-\alpha)[\beta IS(A,O) + (1-\beta)PS(O,W)]$$

where *DS(A,O)*, *IS(A,O)* and *PS(O,W)* represent distance score, interest score and popularity score, respectively. We introduce two modulation parameters, α and β, where α controls the degree of spatial relationship in the calculation of access priority of objects, and β determines which one contributes more to access priority between interest and popularity . Then, *how to determine the parameter β to gain acceptable value of the latter part of formula* is closely related to the correlation between the interest value and popularity value of a given object. As we mentioned in previous sub-section, the popularity value is regarded as a candidate for determining the access priority of a given object only when the interests of a user are not sufficiently reflected by the object in user database. It is because guessing user's interest from object's popularity is indirect incorporation of the user's access pattern to objects. Thus, we want to set the value of β to large only when the user's database is accessed at sufficiently frequent time, in which enables more accurate reflection of user's interests to objects. Accordingly, we set β as follows, following the principle of statistical analysis – the more information we have, the more accurately we predict.

$$\beta = \frac{CN - Time_{A,W}}{CN - Time_{\max,W}}$$

where $CN\text{-}Time_{A,W}$ is the measured cumulative navigation time of a user *A* in World *w*. This formula indicates that as a user navigates long, we can use a more accurate interest value. By assigning $CN\text{-}Time_{\max,w}$ to a sufficiently large value enough such that all the objects are navigated several times by a user, we can obtain appropriate modulation parameter β.

Now, let's look at each function consisting of the formula. First, distance score, *DS(A,O)*, is defined as

Walking mode (When |a| = 0 and |v| > 0 during some period)

$$DS(A,O) = u(1 - \frac{D_{A,O}}{D_{\max}}) + (1-u)(1 - \frac{\theta_{A,O}}{\pi})$$

Wandering mode (When |a| != 0 and |v| = 0 during some period)

$$DS(A,O) = 1 - \frac{D_{A,O}}{D_{\max}}$$

where $D_{A,O}$ is the distance and $\theta_{A,O}$ is the angular distance between user A and object O. These two formulae are similar to those in [5, 8, 10] except that we distinct the mode of a user action. A walking mode is identified as a situation in which a user navigates the world at consistent speed (possibly not zero) during some period. It is based on the fact that existing DVE applications provide a few modes of speed, for example, walking, running, and crawling. Otherwise, the user mode is identified as a wandering mode. User action modes would vary from application to application. If an application provides an interface for speed change, then we easily identify the required action modes when speed change occurs. Or even if it is not practical or impossible for an application to detect the speed changes , it is not a difficult problem since we can easily calculate the speed from position update messages.

The next part of the formula is to get the interest and popularity value from the databases. Interest value, *IS(A,O)*, between user A and object O, and popularity value, *PS(O, W)*, of an object O in the world W are defined as

$$IS(A,O) = \frac{AC_O \text{ in user A's database}}{AC_{\max} \text{ in user A's database}} \qquad PS(O,W) = \frac{AC_O \text{ in world W's database}}{AC_{\max} \text{ in world W's database}}$$

where AC_O means the access count of an object O. Access count is updated whenever user sees the object. User database is maintained at the client side, while world database is maintained at the server side. When a user logs in, the user database is initiated. It is updated during navigation, and after a user logs off, the updated count of object are transmitted to the server for updating world database.

3.3. Caching and Prefetching Mechanism

In the last sub-section, we define the formula calculating the access priority of objects incorporating the concept of user's interest and object's popularity in addition to the spatial distance between the user and a given object. We utilize the formula when cache replacement is needed. During a user navigates within a virtual world, the user identifies the cacheable objects inside a user's viewing region, and checks these objects stored in cache. If they are not in the client's cache, they are requested to the server. And if the objects are not in the cache which is full, object replacement should be performed according to the priority of objects. We prioritize the objects described in 3.2, and the object with lowest priority is removed one after the other.

Prefetching process is similar to the cache replacement. We use a larger aura for prefetching than one for caching, and access priority of objects is again used for the selection of objects to prefetch.

4. Experimental Results

A set of parameters used for simulation is presented at Table I. We aim through the simulation to measure the performance of the proposed data management. For comparison with the existing method, we measure cache hit ratio between the proposed scheme and MRM [3] with the same set of parameters. Since the proposed scheme might show different results according to the amount of information that database has about the interest and popularity values of objects, we repeat the simulation across a number of times using the same database with cumulative information. We set the navigation times fixed, so the calculation of β is approximated as following.

$$\beta = \frac{\text{current cumulative count of user login to the world}}{\text{total count of user login to the world}}$$

The size of virtual world	2,500 unit x 2,500 unit
The number of objects in virtual world	25,000 unit
Viewing angle of the user	120°
The radius of aura representing user's view	100 unit
The speed of user	10 unit / sec
Alternation interval	0 ~ 100 sec
Cache size	2% of world database
Navigation time	20,000 sec

Table I. The list of parameters used for simulation

The simulation is performed as follows. Before a user enters a virtual world, we assume that the world database is filled with the popularity values of the world objects accessed by other users enough to reflect the popularity scores of objects accurately. For obtaining the world database containing accurate popularity values, we do some preprocessing before the simulation. Step 1 is required for world generation and preprocessing. Step 2 simulates the user behavior in a virtual world.

Initially, world objects are created with their own popularity value representing actual popularity in a virtual world. In step 1, world database for object popularity is created and updated. In step 2, user starts the simulation a number of times with user database initially empty and world database updated in step 1. Each time user's navigation times are expired in step 2, i.e., user log offs his session, cache hit ratio is measured and recorded. We measure the cache hit ratios as a user performs logins and logoffs 200 times.

Step 1: This is the preprocessing step before the simulation. We generate a set of world objects according to the parameters specified in Table I. Let 1,000 users navigate the virtual world sequentially. As a result of step 1, the generated world database has the access count of every objects representing object's popularity value.

Step 2: One user having its own interest type navigates the virtual world 200 times. During navigation, the objects, which come within the range of user's viewing region, are cached, and a replacement process is performed if needed. Each time user's navigation is done, cache hit ratio is recorded.

For the simulation, user behavior is assumed as follows. The user has its own interest type. We classify the virtual objects into 10 types when they are created. During the navigation, the user is apt to move to the objects that is same with his interest type or has high popularity value in the world. Also, the user alternates between a moving mode and a wandering mode at the random alternation interval.

Figure 1 shows the comparison of hit ratios between MRM replacement algorithm and the proposed scheme. We simulate user's navigation 200 times, and average the hit ratios per 10 times. The figure shows that proposed data management scheme improve the cache hit ratios by 20% on average when the size of cache is 2% of the virtual world database.

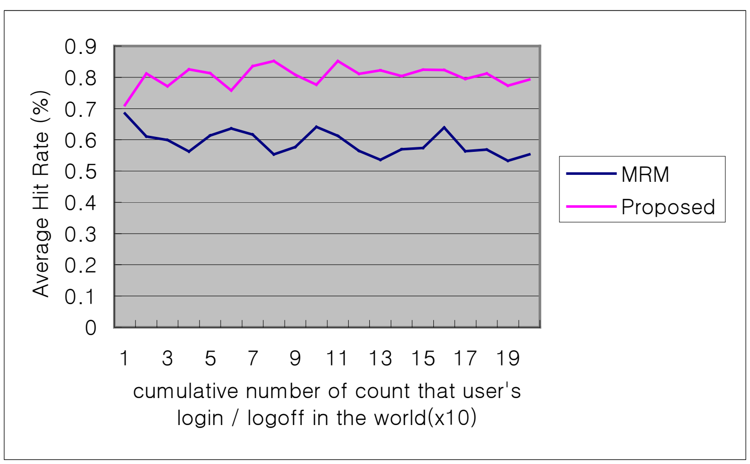

Fig. 1 Comparison of hit rate between MRM replacement algorithm and proposed algorithm

The proposed scheme depends on the cumulative history of object access counts in the world database and user history database. In step 1 of the simulation, we deliberately make the world database populated with user's access counts, which implies the exact estimation of object popularity values is enabled. In a real world situation, it is reasonable to assume that the world database is always fulfilled with object access count by other user because the number of users participating in distributed virtual environments scale in times. For this reason, the proposed scheme shows fairly good performance in figure 1 even when user's navigation time is not sufficient to reflect the user's interests.

5. Conclusion

In this paper, we have proposed a scalable data management scheme using user-based caching and prefetching in distributed virtual environments. To efficiently eliminate the unnecessary network transmission of world data, proposed scheme reflects the user's interests and object's popularity into the replacement algorithms of caching and prefetching added to spatial distance between the users and objects. By maintaining the database of access count for the world objects, we approximate the user's preference and object's popularity. Also, the proposed data management scheme is further enhanced by taking a different action depending on the mode of user action, a wandering mode and a walking mode. The proposed scheme improves the scalability of distributed virtual environments in that it reduces the communication overheads arisen from the on-demand data distribution. Simulation results revealed that it outperformed existing cache replacement algorithm by 20% on average when the cache size is 2% of the virtual world database.

While a cache replacement algorithm determines a coarse-grained degree of communication overhead, the prioritized transfer of objects affects a fine-grained degree of communication overhead. It is another issue - how they affect the overall performance of networked virtual environment to what degree. We are currently investigating how these correlations are exploited to maximize the performance of distributed virtual environments by implementing the proposed approach into our DVE network framework, ATLAS [3].

6. References

[1] M. Capps, "The QUICK Framework for Task-Specific Asset Prioritization in Distributed Virtual Environments," in Proceedings of IEEE Virtual Reality 2000, pages 143-150, March. 2000.

[2] C. Carlsson and O. Hagsand. "DIVE – a Multi-User Virtual Reality System," in Proceedings of IEEE Virtual Reality Annual International Symposium, pages 394-400, 1993.

[3] J. Chim, R. Lau, A. Si, H. Leong, D. To, M. Green, and M, Lam, "Multi-Resolution Model Transmission in Distributed Virtual Environment," in Proceedings of ACM Virtual Reality Software Technology (VRST'98), pages 25-34, Taipei, Taiwan, Nov. 1998

[4] M. Franklin, M. Carey, and M. Livny, "Global Memory Management in Client-Server DBMS Architectures," in Proceedings of the International Conference on Very Large Database, pages 596-609, 1992.

[5] T. Funkhouse, C. H. Sequin, and S.J.. Teller. "Management of large amounts of data in interactive building walkthroughs," in D. Zeltzer, editor, Computer Graphics (1992 Symposium on Interactive 3D Graphics), volume 25, pages 11-20, Mar. 1992.

[6] C. Greenhalgh, S. Benford, "MASSIVE: A Collaborative Virtual Environment for Teleconferencing," ACM Transactions on Computer-Human Interaction, Vol 2, No. 3, 239-261, September, 1995

[7] S. Han, M. Lim, and D. Lee, "Scalable Interest Management Using Interest Group based Filtering for Large networked Virtual Environments," in Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST 2000), pages 103-108, Oct. 2000.

[8] G.. Hesina and D. Schmalstieg, “A Network Architecture for Remote Rendering,” in Proceedings of Distributed Interactive Simulation and Real-Time Applications, pages 88-91, 1998.

[9] H. Hoppe, “Progressive meshes, ” in Proceedings of SIGGRAPH ’96, pages 99-108, 1996.

[10] D. Lee et al, “ATLAS: Scalable network framework for large distributed virtual environments,” MIC Project Report, August, 1999.

[11] M. Macedonia, and M. Zyda, “Taxonomy for Networked Virtual Environments,” IEEE multimedia, vol. 4 no.1, pages 48-56, January-March, 1997

[12] D. Miller, and J. Thorpe, “SIMNET: the advent of simulator networking”, in Proceedings of IEEE, vol.83, no.8, pages 1114-1123, August 1995

[13] J. Purbrick, and C. Greenhalgh, “Extending Locales: Awareness Management in MASSIVE-3”, in Proceedings of Virtual Reality, Feb.2000

[14] D. Schmalstieg and M. Gervautz, “Demand-Driven Geometry Transmission for Distributed Virtual Environments,” in Proceedings of EUROGRAPHICS ’96, pages 421-433, Poitier, France, August 1996.