Delayed Consistency Model for Distributed Interactive Systems with Real-Time Continuous Media*

QIN Xiao

(*Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE* 68588-0115, *USA*)

E-mail: xqin@cse.unl.edu

http://www.cse.unl.edu/~xqin

Received Marcy 31, 2001; accepted August 16, 2001

Abstract: The advanced multimedia and high-speed networks make distributed interactive systems more promising and practical. These systems are distributed systems, which allow many clients located in different locations to concurrently explore and interact with each other. The systems can be built either in the local area network (LAN), or the wide area network (WAN), such as the Internet. Operations issued at one site are immediately executed at the local sites for a good response time, and are propagated to other sites. One of the challenging issues raised in the systems is consistency maintenance. Such issue in the discrete interactive media has been studied in many literatures. However, the consistency maintenance scheme for discrete interactive media is not suitable for continuous media domain. This paper illustrates a consistency problem in continuous interactive media by a simple example. The absolute consistency model, a strong requirement, is suitable for LAN and results in a bad responsiveness in WAN. To make the model more practical for WAN, a new consistency model, named delayed consistency model (DCM), is proposed. In this model, if an operation on an object *x* is issued at site *i*, every site is required to execute the operation at a specified time. The essential idea behind the proposed model is that other sites are enforced to update the state at a certain amount of time later than site *i* does. Thus, other sites will finally view the same state of *x* as that of site *i*. The DCM model is flexible, since it is unnecessary for all sites to have the identical delayed time. In case that the system is based on a real-time network, another advantage of the model is providing the real-time network scheduling with important timing parameters.

Key words: consistency maintenance; interactive object; continuous media; distributed systems

1 Introduction

With the increasing development of distributed systems, multimedia and high speed networks[1], building distributed interactive systems becomes promising and practical. Using such systems, groups of geographically distributed users and services can share information that is created and updated dynamically. Examples of distributed interactive applications include distributed games, distributed simulation[5~7], distributed interactive

---

* Supported in part by an ARC (Australia Research Council) Large Grant No.A00000711 (澳大利亚研究委员会大型项目部分资助)

QIN Xiao was born in 1974. He received the B.S. and M.S. degree both in Computer Science from Huazhong University of Science and Technology (HUST), China in 1992 and 1999, respectively. From July 1999 to June 2000, he worked as a research assistant at HUST. Since August, 2001, he has been a Ph.D. candidate with the Department of Computer Science and Engineering at University of Nebraska- Lincoln (UNL), USA. Prior to joining UNL, he was a Ph.D. student with School of Computing and Information Technology at Griffith University, Australia. He had served as an editor for IEEE Distributed System Online for one year since October 2000. He is a student member of IEEE. His research interests include parallel processing, real-time computing, collaborative computing and distributed systems. The work reported in this paper had been done by the author when he was a Ph.D. student at Griffith University.

learning[3] and distributed virtual environment[2,4], as well as CSCW applications such as collaborative editing systems[8~11]. In the distributed interactive systems, clients can extract the relevant objects by updating their states, or communicate real-time with other clients who also work in the same system.

Due to the fast development of the Internet computing, there are increasing number of distributed interactive systems developed based on the Internet. Distributed interactive system provides an infrastructure to build a large scaled environment for interactive activities by interconnecting many clients, in which people may cooperate with each other. This system has brought a new set of open issues and challenging problems to solve.

A well-designed distributed interactive system has four basic features. (1) The proper interpretation of the time, (2) Consideration of the transmission delays of the operations, (3) Execute the interactive operations in a correct causal order, (4) Real-time response, users make their judgment and reaction based on the situation presented to them by human-computer interfaces. These features can be guaranteed by consistency maintenance scheme, which is one of the most important issues in distributed systems.

In an interactive system with continuous media, the objects are moving around. Even without operations issue on them, the objects is able to move by themselves according to the rules, which have been specified at design stage. It is a basic requirement that all sites should have the same view upon the object. Hence, when one site issues an operation, the operation together with its time-stamp is propagated to other sites so that the state of the object can be update by the remote sites in accordance with the received operation.

In this paper, we focus on consistency issue for interactive continuous objects. The distributed interact system studied in this paper has no centralized management, and each object in the system has the replicated copies at all sites. The network delay is unpredictable bound, therefore, this model can be applied for the Internet based interactive systems.

The rest of this paper is organized as follows. In Section 2, an overview of related work is presented. Section 3 gives a simple example to illustrate an inconsistency problem, which is named as delayed inconsistency. To keep the system delayed consistent, section 4 presents a new consistency model, named delayed consistency model (DCM). Section 5 described the consistency maintenance for distributed interactive objects. Finally, we summarize the contributions of this paper and suggest the future research work.

2 Related Work

Consistency maintenance is one of the most significant challenges in designing and implementing distributed interactive systems. Such issue raised in discrete interactive media has been explored and discussed in many literatures. Real-time collaborative editing system is a good example of interactive system with discrete media[8,10~12,14].

In Ref.[8], a consistency model, with properties of convergence, causality preservation, and intention preservation, is proposed as a framework for consistency maintenance in real-time collaborative editing systems. The integrated set of schemes and algorithms, which support the proposed model, are presented in Ref.[8]. To study the consistency maintenance in real-time collaborative editing system, Li et al. studied how the user intentions might be impacted when the finite duration of drawing operations is considered[10]. The problem of maintaining consistency in the fact of high-latency communication network is also studied in Ref.[10]. Sun and Chen proposed a novel distributed multi-version approach to conflict resolution for real-time collaborative graphics editor[14,15].

Storm *et al*. proposed a model, in which several objects may be atomically updated, and these objects automatically maintain consistency with their replicas using an optimistic algorithm[16]. The algorithms are based on optimistic guess propagation principles, adapted for fast commit by using primary copy replication approach.

Unfortunately, the consistency maintenance schemes devised in these papers only concern about the discrete interactive systems. Unlike the object with discrete media, a continuous media object not only changes its state according to the operations issued on it, but also continuously updates the state due to the passing of time as well. Thus, the object's state not only depends on what operation issued on it, but also depends on the time at which this operation is executed.

Recently, continuous media has been employed in various kinds of distributed interactive systems, such as multi-user computer games[22], distributed virtual reality[17,18,20,24] and simulations[7,19]. Distributed virtual environment is a distributed system, which allows many clients who are located in different locations to concurrently explore and interact with each other in a high resolution, 3-dimensional, graphical virtual environment[24]. Greenhalgh et al. studied a Qos architecture for collaborative virtual environments[18], and their work focus on the management of streamed video within shared virtual worlds. Choukair and Retailleau explored a model of the Distributed Virtual Environment Collaboration Model (DVECOM), which aims to provide an end-user QoS support for distributed virtual reality applications[20]. Bieber and Siron proposed a security architecture that aims to protect the intellectual property of firms participating to a distributed interactive simulation[7]. As the number of simulated entities grows, the number of messages that need to be sent per unit of time can grow to unmanageable numbers. To reducing the number of messages, Messina et. al presented a approach to keep track of what entities need to know about which other entities and only send information to the entities that need to know[19]. However, these works did not study the consistent control in the systems.

Consistency maintenance in continuous media remains many open issues. Zhou *et al*. introduced a concept of time-space inconsistency for continuous distributed interactive applications[13], and three kinds of inconsistency problems were addressed in Ref.[13]. This paper, however, addresses a problem that is not covered in Ref.[13]. Diot and Gautier described the design and implementation of a distributed multiplayer game on the Internet, in which the buckets synchronization mechanism is devised to guarantee the consistency of the game[22]. Mauve studied the important tradeoff relationship between the responsiveness of the medium and the appearance of short-term inconsistencies[21]. Reference [21] shows that the fidelity of the application can be significantly raised by voluntarily decreasing the responsiveness of the medium. This approach is named local lag, which means instead of immediately executing an operation issued by a local user, the operation is delayed for a certain amount of time before it is executed. This scheme is very similar as the buckets synchronization addressed by Diot and Gautier[22].

Unlike the model proposed in Refs.[21,22], DCM shows that it is unnecessary for all the sites to have the same state at the same time. Different sites can share the same view of the objects at different time, which is specified at the designed stage and predictable for user.

3 An Example of Inconsistency in Continuous Interactive Objects

Since continuous interactive objects may move by themselves even without any operation issued on them, the states may continuously update not only according to the operations issued on it, but also because of the passage of time as well. In DCM model, it is assumed that the local clocks of all the sites are synchronized periodically. The clock synchronization can be guaranteed by Network time Protocol[23] or GPS clocks, and the clock deviation is less than 50ms. The work reported in Ref.[21] has a similar assumption.

As shown in Fig.1, we illustrate an example in a 2-dimension environment. It is straightforward to extend this example into 3-D environment. Suppose the initial position of object x at site 1 and site 2 is the same, and set to be (0,0), and the initial velocity of x is (0,0). Site 1 issue three operations: operation 1 makes x move, operation 2 changes the direction and accelerates x, and operation 3 stops x. The final positions of x at two sites are determined as follow.

At time $t_1(o_1)$=0, the operation o_1 from site 1 makes x move constantly at the velocity v=(10,0). At time

$t_1(o_2)$=0.5, operation o_2 changes the direction of the object x, and also increases speed to v'=(0,20). Therefore, object x changes the direct at the coordinate (5,0). At time $t_1(o_3)$=1.5, x is stopped by o_3 at the position of (5,20).

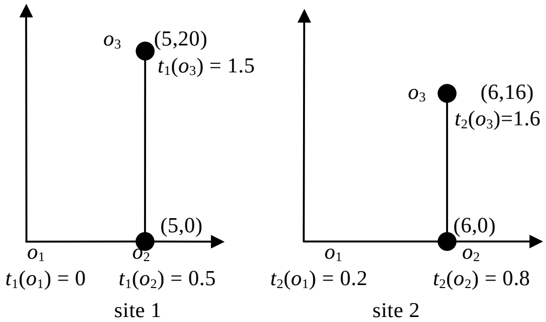

Fig.1 Example of an inconsistency problem in object with continuous media. Site 1 issues three operations at time 0, 0.5 and 1.5, respectively. The communication delay for operations are 0.2, 0.3 and 0.1. The final position at site 1 is (5, 20) and the position after executing o_3 at site 2 is (6,16).

Each operation is immediately transmitted to site 2 in parallel with its execution at site 1. It is assumed that the communication delays for three operations are d_1=0.2, d_2=0.3 and d_3=0.1, respectively. The time when three operations are executed at site 2 is expressed as $t_2(o_i)=t_1(o_i)+d_i$, where i=1,2 and 3, supposing that site 2 executes operations without any artificial delay at site 2. Similarly, x starts moving at time $t_2(o_1)$=0.2, and x changes its direction at position (6,0) at time $t_2(o_2)$. Object x stops in response to o_3 at time $t_2(o_3)$=1.6, approaching to the final position (6,16). Obviously, final positions of x at two sites become different, which is an inconsistency case that can be avoided in DCM model. The reason for the inconsistency problem shown in the above example is that the communication delays associated with the operations are various because of the unbounded network delay.

4 Consistency Model

The object of this work is to devise an approach to attack the inconsistency problem illustrated in the previous section. The underlying network setting is modeled as a graph $G=(V,E)$, where $V=\{v_1,v_2,...,v_n\}$ represents a set of sites connected by the network, and elements in set E denote communication delay between two sites. Let d_i^j be the upper bound communication delay for a message transmitted via this channel. In the following definitions, t is the wall-clock time.

4.1 Absolute consistency

Definition 1. Given an object x, the state of x is defined by its current coordinate and its current property, such as the directional velocity, size, colour, etc. At the wall-clock time t, let $s_x^i(t)$ denotes the state of x at site I at time t, where $1\le i\le n$.

Definition 2. Given an operation o, $v(o)$ denotes the site at which o is issued at time $g_i(o)$, and $e(o)$ denotes the object on which o is operated. Let $a_i(o)$ and $c_i(o)$ be the arrival time and the execution time of o at v_i. It is obvious that $g_i(o)\rightarrow v(o)=i$, and $a_i(o)\rightarrow v(o)\neq i$.

Definition 3. Given two operations o_i and o_j, o_i is *causal order preceding* o_j, i.e. $o_i \rightarrow o_j$, iff: (1) $v(o_i) = v(o_j) = k$, and $g_k(o_i)< g_k(o_j)$; (2) $v(o_i) \neq v(o_j)$, $a_k(o_i) < g_k(o_j)$, where $k = v(o_i)$; (3) There exists an operation o_k, such that $o_i\rightarrow o_k$, $o_k\rightarrow o_j$.

Definition 4. Given an object x, the object is *absolute consistent* at time any t, iff $\forall 1\le i, j\le n, i\neq j$: $s_x^i(t)=s_x^j(t)$.

The system is absolute consistent, iff $\forall x$, t>0: *x is absolute consistent*.

In most cases, especially for the system that is built on the Internet, it is impossible to achieve absolute consistency, due to the unbounded transmission delay. This fact is described and proved in theorem 1 and 2, which assume that the system clocks are synchronized.

Theorem 1. Assume that system is initialized to be absolute consistent. The system is absolute consistent, if and only if any operation at all sites execute at the same time.

Proof. (1) Firstly, it is straightforward to prove that any operation at all sites must execute at the same time $\rightarrow$ the system is absolute consistent. Since the initial state of the system is consistent and each operation is executed at all the sites at the same time, the state at each sites update the state at the same time. Therefore, system is absolute consistent is proved.

(2) Secondly, we prove the system is absolute consistent $\rightarrow$ any operation at all sites must execute at the same time. This can be proved by contradiction. Given an operation o issued on the object x, where $v(o)=i$. Let t be $g_i(o)$. We suppose $\forall t'<t$:($\forall 1\le j\le n$, $i\ne j$: $s_x^i(t')=s_x^j(t')$), thus, the system is absolute consistent at any time before t. If site i executes o at t_1 whereas site j execute o at $t_2 > t_1$, then it is clear that, $s_x^i(t_1) \ne s_x^j(t_2)$, thus, the system is not absolute consistent at time t_1. Therefore, the operations must be executed at all sites at the same time. □

Theorem 2. Given an operation o issued on the object x, where $v(o)= i$, if o executes at site i immediately after it is issued, then system is not absolute consistent at time $g_i(o)$.

Proof. Let t be $g_i(o)$. We suppose $\exists\varepsilon>0$ ($\forall 1\le j\le n$, $i\ne j$: $s_x^i(t-\varepsilon)=s_x^j(t-\varepsilon)$), thus, the system is absolute consistent at time $t-\varepsilon$. (1) If at time t, one site j ($j\ne i$) updates the state of x with operation $o'\ne o$, then $\exists 1\le j\le n$, $i\ne j$: $s_x^i(t)\ne s_x^j(t)$. Theorem holds for case (1). (2) We prove the case that at time t, every site j ($j\ne i$) does not issue operation on x. Since o updates the state of x at time t, we have $s_x^i(t)\ne s_x^i(t-\varepsilon)$. Because it takes time $\delta>0$ to propagate operation from i to other site, the state of x at site j ($j\ne i$) remains unchanged at time t, thus, $\forall 1\le j\le n$, $i\ne j$: $s_x^j(t) = s_x^j(t-\varepsilon)$. We get $\forall 1\le j\le n$, $i\ne j$: $s_x^i(t)=s_x^j(t)=s_x^i(t-\varepsilon)$. Therefore, theorem also holds for case (2). Based on (1) and (2), the proof is completed. □

Theorem 2 indicates that, to keep the system absolute consistent, the operation generated at the local site could not execute immediately after it is issued. On the contrary, the operation must delay for a certain of time before it is executed. To solve the absolute consistency problem, we have the following Theorem 3.

Theorem 3. Given an operation o issued on the object x, to keep the absolute consistent, the execution time of o at site i is equal or greater than $g_i(o)+\mathrm{MAX}_{j=1,j\ne i}^{n}(d_i^j)$, where $i=v(o)$, and $d_i^j(o)$ is the transmission delay for o to be transferred from site i to j.

Proof. We prove this theorem by contradiction. Let $d_i^k=\mathrm{MAX}_{j=1,j\ne i}^{n}(d_i^j)$ and suppose site i does not delay for a certain of time d_i^k, thus, site i executes operation at time $g_i(o)+d$, where $d<d_i^k$. Since site k receives and executes the operation at $g_i(o)+d_i^k$, site i and site k execute the o at two different time. According to Theorem 1, the system is not absolute consistent. This completes the proof. □

4.2 Delayed consistency of objects

An observation from Theorem 3 is that if the delay time for local operations is long enough, the number of absolute inconsistencies can be sharply decreased. Similar conclusion can also be found in the work of Mauave[21] and Cristian[25]. Unfortunately, employing a long delay for local operations leads to a long response time, which may not be tolerated by most users. Thus, the problem of maintaining consistency is posed as a tradeoff between high consistency and a good response time.

It is extremely difficult to meet the requirement of absolute consistency on the Internet, since the Internet gives rise to the unpredictable bounds and large variations on message transmission delay, and the delay is normally significant noticeable. In the Internet environment, we either have a bad responsiveness with few inconsistencies or have a great number of inconsistencies with a short response time. To solve this problem, we propose the concept of the late consistency. In DCM model, we assume that only one site owns a given object at a given period of time, and a site issues operations on the objects it owns. For example, in a distributed car racing game, player A steers his car and controlling other players' car is prohibited. Let player A's car be an object, the owner of this object is player A.

Definition 5. Given an object x, the ownership of x at time t is denoted as $w_x(t)$. For simplicity, it is to be assumed that the ownerships are not transferable, therefore, we have $\forall t \geq 0$: $w_x(t)=w_x$.

In accordance with the definition of w_x, we obtain $v(o) = i \rightarrow e(o) = x \wedge w_x = i$. (1)

Definition 6. Given an object x, the object is *delayed consistent* at time $t \geq 0$, iff $\forall 1 \leq i, j \leq n, i \neq j$: $w_x = i \rightarrow s_x^i(t) = s_x^j(t+\delta_i^j)$. δ_i^j is the phase difference between two sites.

The above definition implies that if an object is owned by site i with the state s at time t, the object at other sites will have the same states. It is noted that s appears at the remote sites at a certain amount of time later than t. For example, from time step 1 to 20, the state of the object x at site 1 is $\{s_x^1(1), s_x^1(2), \ldots, s_x^1(20),\}$, and x has the same sequence of the state at site 2 starting from time 5 to 24. In other words, we have $\{s_x^2(5) = s_x^1(1), s_x^2(6) = s_x^1(2), \ldots, s_x^2(24) = s_x^1(20)\}$. The phase difference between site 1 and 2 is 4, thus, $\delta_1^2 = 4$.

To maintain a delayed consistency for an object, the local operation o on x is allowed to be executed immediately after it is issued by site $v(o)$. When o is propagated to other sites, o is executed at a specific time by delaying it for a certain amount of time. Hence, at site $j \neq v(o) = i$, o is executed at time $g_i(o)+\delta_i^j$. That is

$$c_j(o) = g_i(o)+ \delta_i^j. \quad (2)$$

This approach ensures that the x satisfies the delayed consistency. This is described in the property as below.

Property 1. Give an object x, $\forall o$, $1 \leq i, j \leq n, i \neq j$: $e(o) = x \wedge v(o) = i \wedge c_j(o) = g_i(o)+ \delta_i^j \wedge (\forall t < g_i(o)$: *x is delay consistent*$) \rightarrow x$ is delayed consistent**.**

Suppose the objects in site i are delayed consistent, and two operations o_1 and o_2 are issued at site i, we have $c_j(o_2) - c_j(o_1) = g_i(o_2) - g_i(o_1)$, where $j \neq i$. This feature is formally described in Theorem 4.

Theorem 4. Given two operations issued on site i, we have $v(o_1) = v(o_2) = i \rightarrow \forall 1 \leq j \leq n, j \neq i$: $c_j(o_2) - c_j(o_1) = g_i(o_2) - g_i(o_1)$.

Proof. This theorem can be proved straightforward from the approach describe above. According to expression (2), we obtain (a)$\forall 1\leq j\leq n, j \neq i$:$c_j(o_1)= g_i(o_1)+\delta_i^j$ and (b)$\forall 1\leq j\leq n, j \neq i$:$c_j(o_2) = g_i(o_2)+\delta_i^j$. Subtracting expression (b) by (a), we have$\forall 1\leq j\leq n, j \neq i$: $c_j(o_2)-c_j(o_1)=g_i(o_2)-g_i(o_1)$. □

Unlike the model proposed by Mauave[21] and Cristian[25], this DCM model does not force every site to execute o at the same time. δ_i^j is an input parameter specified by users, this makes the system design more flexible. As shown in Fig.2, delay time from site 2 to site 1 is 9, whereas delay time from site 2 to site 3 is 5.

Let d_i^j be the upper bound on the communication delay from site i to site j, it ensures that all operations arrive before d_i^j. Hence, the specified delay time δ_i^j must be equal or greater than d_i^j as formally described below.

Theorem 5. Give an object x, and $i=v(o)$, if the object is *delayed consistent*, then $\forall 1\leq j\leq n, i \neq j$: $\delta_i^j \geq d_i^j$.

Proof. Assume at $t=g_i(o)$, site i update the state of x by executing o. We suppose the transmission delay equals to the upper bound d_i^j, and site j changes the state in response to o at time $t + d_i^j$. Thus, $s_x^i(t) = s_x^j(t+\delta_i^j)$. In this case, we have $\delta_i^j = d_i^j$. If site j delays for a certain amount of time τ after receiving o at time $t + d_i^j$, then we get $s_x^i(t) = s_x^j(t + \delta_i^j + \tau)$. Therefore, $\delta_i^j = d_i^j + \tau \geq d_i^j$. □

4.3 Delayed consistency of sites

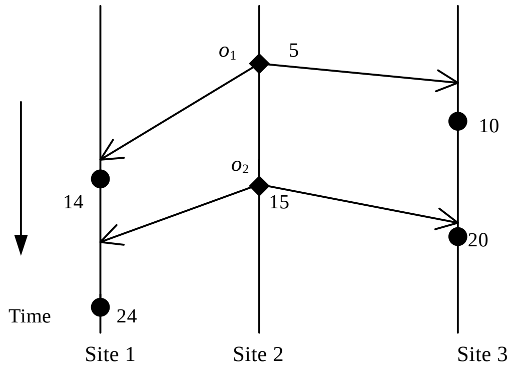

Fig.2 Delay time from site 2 to site 1 is 9, whereas the delay time from site 2 to site 3 is only 5

Given two site i and j ($i \neq j$), DCM model makes site i view the same states of the objects that belongs to site j at a certain amount of time later. Given another site k ($k \neq i$ and $k \neq j$), site i also views the same states of the objects that belong to site k at a later time. We are interested in the views of objects belonging to sites j and k from site i. Even though all the objects belong to sites j and k are delayed consistent, some interesting problems may occur. We will demonstrate one problem by a simple scenario that is depicted in Fig.3. For simplicity, we only show the example in one dimension and it is straightforward to extend the example to 2-D and 3-D environment.

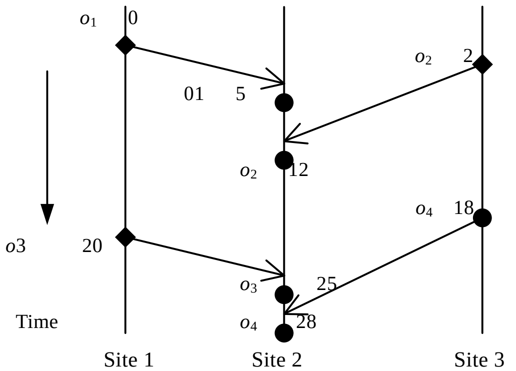

Fig.3 $\delta_1^2 = 5$, $\delta_3^2 = 10$, operation o_1 and o_3 are issued on object x at site 1, whereas o_2 and o_4 are issued on y at site 2

In an Internet based racing game, player A and C at site 1 and 3 drives car A and C, which are denoted as objects x and y, respectively. Player B at site 2 is an observer who is interested in this racing game. We assume that $\delta_1^2 = 5$, $\delta_3^2 = 10$. Operation o_1 and o_3 are issued on x at site 1, whereas o_2 and o_4 are issued on y at site 2. It is observed from Fig.3 that both x and y are delayed consistency.

Let the initial positions of x and y be 0. At time step 1, o_2 moves x move at a constant speed of 10, and at time step 2, y starts moving with a constant speed of 10 by o_2. Therefore, the distance between x and y is 20, and x is 20 ahead of y when y begins moving. However, from the observer's point of view, the distance between x and y is 70 instead of 20. This is because at site 2, x starts moving at 5, whereas y begins moving at 12.

At time step 18, operation o_4 stops y, thereby making the final position of y be 160. However, object x is stopped by o_3 at time 20 with the final position 200. Player B at site 2 observes that x stops 3 time steps earlier than y, thus, y continues moving when x stops. The explanation of this scenario is that the specified delays δ_1^2 and δ_3^2 are not identical. The states of the objects in site 1 and site 3 have been delayed for different amount of time before they are viewed at site 2. One efficient solution for the inconsistent views is assigning a same value to δ_1^2 and δ_3^2.

To study this inconsistent problem in a general way, we introduce the definition of the delay consistency for a site. Let X be the set of object, it has n subsets, denotes as $X=X_1\cup X_2\cup\ldots\cup X_n$, where $X_i=\{x\in X|w_x=i\}$. It is clear that, $\forall 1\le i, j\le n, i\ne j$: $X_1\cap X_2=\varnothing$.

Definition 7. The state of the site *i at time t* is represented as, $S^i(t)=S^i_I(t)\cup S^i_E(t)$, where $S^i_I(t)$ is composed by the states of the objects owned by site i, whereas $S^i_E(t)$, contains the states of the objects that belong to other site. Thus, $S^i_I(t)=\{S^i_x(t)$, where $x\in X_i\}$ and $S^i_E(t)=\{S^i_x(t)$, where $x\notin X_i\}$.

Definition 8. Given a site i, it is *delayed consistent*, iff $S^i_E(t)=\bigcup_{j=1,j\ne i}^{n} S_I^j(t-\delta^i)$.

The system is consistent, iff $\forall 1\le i\le n$: *all site i are delayed consistent*.

As defined in Define 8, if the system is consistent, it must have the following property.

Property 2. For any site i in the system, it satisfies: $\forall 1\le j\le n, j\ne i$: $\delta_j^i=\delta^i$.

Parameter δ^i is derived *from* d_j^i $(1\le j\le n, j\ne i)$. δ^i is given and proven in the Theorem 6 as below.

Theorem 6. If the system is consistent, then, $\forall 1\le i\le n$: $\delta^i\ge MAX_{1\le j\le n,j\ne i}d_j^i$.

Proof. Given a site i $(1\le i\le n)$, we assume $d_k^i=\mathrm{MAX}_{1\le j\le n,j\ne i}d_j^i$. According to the Theorem 5, we have $\delta_k^i\ge d_k^i$. Since the system is consistent and based on the Property 2, we get $\delta_k^i=\delta^i$. Therefore, we obtain $\delta^i\ge\mathrm{MAX}_{1\le j\le n,j\ne i}d_j^i$. Thus, we get $\forall 1\le i\le n$: $\delta^i\ge\mathrm{MAX}_{1\le j\le n,j\ne i}d_j^i$. □

To keep the delayed consistency for a given site i, our approach is very simple: The operation o is delayed before it is executed by site j. This approach ensures that the execution times of o at site i is δ^i later than that at site j. Thus,

$$c_i(o)=g_j(o)+\delta^i. \quad (3)$$

Assume a site i is delayed consistent, and given two operations o_1 and o_2 issued at site j and k, respectively. We have $c_i(o_2)-c_i(o_1)=g_k(o_2)-g_j(o_1)$, where $j\ne i$ and $k\ne i$. Theorem 7 explains this feature formally.

Theorem 7. If a site i is delayed consistent, we have $\forall 1\le j, k\le n, j\ne i, k\ne i$: $v(o_1)=j\wedge v(o_2)=k\rightarrow c_i(o_2)-c_i(o_1)=g_k(o_2)-g_j(o_1)$.

Proof. From Eq.(3), we have (a)$\forall 1\le j\le n, j\ne i$: $v(o_1)=j\rightarrow c_i(o_1)=g_j(o_1)+\delta^i$ and (b)$\forall 1\le k\le n, k\ne i$: $v(o_2)=k\rightarrow c_i(o_2)=g_k(o_2)+\delta^i$. Subtracting expression (b) by (a), we have $\forall 1\le j, k\le n, j\ne i, k\ne i$: $v(o_1)=j\wedge v(o_2)=k\rightarrow c_i(o_2)-c_i(o_1)=g_k(o_2)-g_j(o_1)$. □

5 Conclusion

We address the consistency issue in distributed interactive systems with continuous media. A consistency problem is illustrated by a simple example. This type of consistency problem is due to the fact that objects can move and update automatically even without manipulation of users. Hence, the state of an object not only depends on the operations issued on it, but also relies on the time when the operations are executed on it as well. It is to be noted, however, those objects with discrete media do not have such consistent issue. To solve this problem, we propose the delayed consistency model for the interactive object. The DCM model is superior to the models discussed in the literatures[21], since there are several advantages in DCM described as follows.

First, DCM does not require all the sites to have the same state at the same time. Unlike the model in Ref.[21] different sites share the same view of the objects at different time, which is specified at the designed stage. Second, having a good predicability results in tolerating long communication delay time in DCM model. A local site views the state of the other sites, and has the knowledge about what is the exact time associated with the states of other sites. Third, DCM is suitable for real-time network. Because there is a specified delay time for each object, and the specified delay time is an important parameter for the scheduling mechanism in real-time network. In real-time scheduling, the message with shorter delay time will be assigned a higher priority and be transferred earlier than messages with lower priorities[26].

This work represents our first and preliminary attempt to address the consistency issue in a complicated problem. There are a lot of open issues remaining to be studied. Our future work in this field could include:

(1) DCM assumes that the clocks in the system are synchronized by Network time Protocol or GPS clocks, and there is no clock drift. Based on DCM proposed in this paper, we will study a more sophisticated model that takes the clock synchronization and the clock drift into account.

(2) Another assumption of our work is that the ownership of each object is not changeable. In the future model, we will relax this assumption by letting the ownership be transferred from one site to another site, this will give rise to new consistency problems, and we will devise the new approach to solve the problems.

(3) The DCM model assumes that there is an upper bound on the communication delay for each network channel, and operations will be guaranteed to arrive at the destination within the specified delay time. We will relax this assumption in such a way that there is no such upper bound for message communication. In case that the operations arrive at the remote sites later than the specified time, the consistency can no longer be maintained. Therefore, a scheme must be facilitated to recover the false state of the objects under these circumstances.

Acknowledgement The author wants to thank Dr. Chengzheng Sun and David Chen for their helpful discussions and insightful suggestions.

References:

[1] Mills, D.L. Internet time synchronization: the network time protocol. IEEE Transactions on Communications, 1991,39(10): 1482~1493.

[2] Oliveira, J.C., Shirmohammadi, S., Georganas, N.D. Collaborative virtual environment standards: a performance evaluation. In: Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications. College Park, Maryland, 1999.

[3] Chen, J., Dede, C., Fu, X., *et al*. Distributed interactive learning environment. In: Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications. College Park, Maryland, 1999.

[4] Shen, X., Hage, R., Georganas, N. Agent-Aided collaborative virtual environments over HLA/RTI. In: Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications. College Park, Maryland, 1999.

[5] Fortino, G., Nigro, L. Simulation of multimedia systems based on actors and QoS synchronizers. In: Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications. College Park, Maryland, 1999.

[6] Pullen, J. Reliable multicast network transport for distributed virtual simulation. In: Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications. College Park, Maryland, 1999.

[7] Bieber, P., Siron, P. Design and implementation of a distributed interactive simulation security architecture. In: Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications. College Park, Maryland, 1999.

[8] Sun, C., Jia, X., Zhang, Y., *et al*. Achieving convergence, causality-preservation, and intention preservation in real-time cooperative editing systems. ACM Transactions on Computer-Human Interaction, 1998,5(1):63~108.

[9] Sun, C. Undo any operation at any time in group editors. In: Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work. Philadelphia, Pennsylvania, USA, 2000.

[10] Li, D., Zhou, L.M., Muntz, R.R. A new paradigm of user intention preservation in realtime collaborative editing systems. In: Proceedings of the 7th International Conference on Parallel and Distributed Systems. Iwate, Japan, 2000.

[11] Sun, C., Ellis, C.A. Operational transformation in real-time group editors: issues, algorithms, and achievements. In: Proceedings of the ACM Conference on Computer Supported Cooperative Work. Seattle, USA, 1998.

[12] Ellis, C.A., Gibbs, S.J. Concurrency control in groupware systems. In: Proceedings of the ACM SIGMOD Conference on Managements of Data. Portland, OR, USA, 1989. 399~407.

[13] Zhou, S., Cai, W., Lee, F.B., *et al*. Consistency in distributed interactive application. In: Proceedings of the 2001 European Simulation Interoperability Workshop. London, UK, 2001.

[14] Chen, D., Sun, C. A distributed algorithm for graphic objects replication in real-time group editors. In: Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work. Phoenix, USA, 1999. 121~130.

[15] Sun, C., Chen, D. A multi-version approach to conflict resolution in distributed groupware systems. In: Proceedings of the 20th IEEE International Conference on Distributed Computing Systems. 2000. 316~325.

[16] Strom, R., Banavar, G., Miller, K., *et al*. Concurrency control and view notification algorithms for collaborative replicated objects. IEEE Transactions on Computers, 1998,47(4):458~471.

[17] Frécon, E., Stenius, M. Dive: a scalable network architecture for distributed virtual environments. Distributed Systems Engineering Journal, 1998,5(3):91~100.

[18] Greenhalgh, C., Benford, S., Reynard, G. A QoS architecture for collaborative virtual environments. In: Proceedings of the ACM Multimedia. Oriaando, FL, USA, 1999. 121~130.

[19] Messina, P., Brunett, S., Davis, D., *et al*. Distributed interactive simulation for synthetic forces. In: Proceedings of the 6th Heterogeneous Computing Workshop. Geneva, Switzerland, 1997.

[20] Choukair, Z., Retailleau, D. Integrating QoS to collaborative distributed virtual reality applications. In: Proceedings of the 3rd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. 2000.

[21] Mauave, M. Consistency in replicated continuous interactive media. In: Proceedings of the ACM Conference on Computer Supported Cooperative Work. Philadelphia, PA, USA, 2000. 181~190.

[22] Diot, C., Gautier, L. A distributed architecture for multiplayer interactive applications on the Internet. IEEE Network, 1999,3(4):6~15.

[23] Mills, D.L. Network time protocol, version 3, specification, implementation and analysis. DARPA Network Working Group Report RFC-1305, University of Delaware, 1992.

[24] Lui, J., So, O., Tam, T. Deriving communication sub-graph and optimal synchronizing interval for a distributed virtual environment system. In: Proceedings of the IEEE International Conference on Multimedia Computing and Systems. Florence, Italy, 1999.

[25] Cristian, F., Aghili, H., Strong, R., *et al*. Atomic broadcast: from simple message diffusion to Byzantine agreement. Information and Computation, 1995,118(1):158~179.

[26] Zhang, S., Lee, E. Efficient global allocation of synchronous bandwidths for hard real-time communication with the timed token MAC protocol. In: Proceedings of the 6th International Conference on Real-Time Computing Systems and Applications. Hong Kong, 1999.

分布式交互系统中连续媒体的延迟一致性模型

秦 啸

(内布拉斯加-林肯大学 计算机科学与工程系,内布拉斯加 林肯 68588-0115,美国)

摘要: 随着多媒体和网络技术的发展,分布式交互系统被广泛应用.在这种系统中,多个客户端通过局域或广域网交互连接.为使响应时间短,本地节点产生的操作立即在本地执行,并广播到其他远程节点执行.在该系统中,一致性维护是一个关键问题,而文献中研究的一致性问题几乎都是基于不连续媒体的.通过一个实例,指出连续媒体中的一种不一致问题.虽然该问题可以通过绝对一致模型解决,但绝对一致模型应用在广域网中将导致长响应时间.为解决绝对一致模型中响应时间过长的问题,提出了延迟一致性模型(简称为 DCM 模型).在 DCM 中,如果节点 i 产生了作用于对象 x 上的操作,该操作到达远程节点后强行延迟一段时间并要求在统一规定的时间执行.通过该方法,对象 x 在其他远程节点上的状态将最终保持一致.DCM 很灵活,因为不同的对象可以有不同的强行延迟时间.如果分布式交互系统建立在实时网络上,这种强行延迟时间将成为实时通信中实时消息调度的重要参数.

关键词: 一致性维护;交互式对象;连续媒体;分布式系统

中图法分类号: TP391 文献标识码: A