Science in China Series F: Information Sciences

www.scichina.com
info.scichina.com
www.springerlink.com

Algorithm of simulation time synchronization over large-scale nodes

ZHAO QinPing[1,2], ZHOU Zhong[1, 2†] & LÜ Fang[1,2]

[1] State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing 100083, China;
[2] School of Computer Science and Engineering, Beihang University, Beijing 100083, China

In distributed simulation, there is no uniform physical clock. And delay cannot be estimated because of jitter. So simulation time synchronization is essential for the event consistency among nodes. This paper investigates time synchronization algorithms over large-scale distributed nodes, analyzes LBTS (lower bound time stamp) computation model described in IEEE HLA standard, and then presents a grouped LBTS model. In fact, there is a default premise for existing algorithms that control packets must be delivered via reliable transportation. Although, a theorem of time synchronization message's reliability is proposed, which proves that only those control messages that constrain time advance need reliability. It breaks out the default premise for reliability. Then multicast is introduced into the transmission of control messages, and algorithm MCTS (multi-node coordination time synchronization) is proposed based on multicast. MCTS not only promotes the time advance efficiency, but also reduces the occupied network bandwidth. Experiment results demonstrate that the algorithm is better than others in both time advance speed and occupied network bandwidth. Its time advance speed is about 50 times per second when there are 1000 nodes, approximately equal to that of similar systems when there are 100 nodes.

distributed simulation, time synchronization, multi-node coordination, LBTS, multicast

1 Introduction

Event disorders, such as the famous "fire-explosion" transposing problem, are common in networked games or simulations. Event consistency is an important issue in distributed simulation. The main reason is derived from the network communication of nodes. Because of the uncertainty of jitter as well as different computing resources of nodes, the event processing time spent

Received January 4, 2008; accepted May 7, 2008
doi: 10.1007/s11432-008-0118-x
†Corresponding author (email: zz@vrlab.buaa.edu.cn)
Supported by the National Natural Science Foundation of China (Grant No. 60603084) and the Hi-Tech Research and Development Program of China (Grant No. 2006AA01Z331)

by the node cannot be predicted. At the same time, owing to node scattering, it is difficult to conserve a uniform physical clock for their time synchronization. An efficient solution is to simulate the physical time and get the event consistency based on the simulation time. Therefore, time synchronization, also called "time management"[1], is one of the classical problems in the field of distributed simulation. It manages the message delivery time, coordinates different time advance manners, and guarantees the logic correctness during simulation[2,3].

There are several shortcomings in existing time synchronized algorithms, such as low efficiency of time advance and high occupied network bandwidth[4]. Aiming at these problems, this paper analyzes the node characteristics during simulation time advance, and presents a grouped LBTS model. Then a theorem of time synchronization message's reliability is proposed, which breaks the default premise for reliability and proved that only those control messages that constrain system advance need reliability. The algorithm of multi-node coordination time synchronization (MCTS) is put forward, which coordinates multi-node time synchronization based on IP multicast. MCTS is implemented in the time management service of BH RTI. Experiment result shows that MCTS is superior to other similar systems on time advance speed, occupied network bandwidth, and simulation scalability.

This paper is organized as follows. Section 2 introduces related works on simulation time synchronization; section 3 analyzes the characteristics of safe advance in the LBTS time management model, and presents a grouped LBTS model; section 4 proposes and proves a theorem of time synchronization message's reliability, and then put forward MCTS algorithm; section 5 presents the experiment and result analysis; section 6 makes the conclusion.

2 Related work

In distributed systems, time synchronization is used to get the event consistency. Lamport algorithm, which is among the earliest methods, guarantees the event order consistency in all nodes by increasing the logic time in event delivery. It is mainly effective for causality event consistency. The following vector clock algorithm extends Lamport algorithm to treat the causality event delivery of multiple nodes simultaneously, one logic clock for events of one node. However, they cannot ensure the delivery or process consistency of simultaneous events. In most distributed simulations, every node sends at least tens of events at the same time. Actually, we do not demand the consistency between simulation time and the physical time, but the accurately consistency of event order and global event process order. So it is not sufficient even when all nodes share an absolutely consistent physical clock because the event delivery and process time cannot be predicted. Consequently, in the field of distributed simulation, simulation time is ordinarily applied to attain this strict time synchronization currently. Time management derives from PDES (parallel discrete event simulation), which treats the consistency of simulation time with multi-processor machine in parallel. The time management of distributed simulation is developed on the time management of PDES. It synchronizes the time advance of distributed nodes based on network delivery[1,2].

In the early classical Chandy/Misra algorithm[5], messages are sent to other logic processors (LP) with a non-decreasing timestamp. LP stores messages in a queue, and messages with the smallest timestamp are first treated in message processing. This algorithm can guarantee the event consistency in a local process, but it takes the risk of dead-lock.

In order to solve the deal-lock problem, Chandy and Misra again brought forward Null Mes-

sage algorithm. Logic process LP_A sends Null Message to logic process LP_B, and declares that LP_A will not send a message whose timestamp is less than a specific future time. The Null Message algorithm introduces the concept of Lookahead[2], and has been used in most time synchronization algorithms.

Existing time management algorithms mainly attain the event consistency based on the range of time advance. This kind of algorithms decides the range of time advance on exchanging time information among LPs, and then guarantee that every node will not receive an outdated message. In this way, any disorder can be avoided. The computation of time advance interval which is called lower bound time stamp (LBTS) is the critical point. HLA standard defines LBTS and Lookahead in its time management service. LBTS is the largest future time for a safe time advance, and nodes will not receive a message, whose timestamp is less than current LBTS. Mattern [6] brought forward an LBTS computation method on distributed snapshot. A counter is maintained for each LP, whose value is equal to the number of received messages subtracted from the number of messages sent out. The computation of LBTS is launched when the counter comes to zero. Fredrick proposed to compute LBTS on the output time of simulation processes in different time advance state. Time advance interval can help manage the time consistency efficiently; and is widely accepted in the field of distributed simulation. It is also regulated in the HLA standard. But the computation of LBTS requires real-time maintaining all the simulation time of nodes. Thus, it has high complexity and is disadvantageous in scalability.

There are also some other time synchronization algorithms. The approximate time by Fujimoto[7] uses the time interval instead of time point to increase the parallel degree. This method boosts the time advance efficiency of those distributed simulations whose Lookahead is small or equal to zero. But one problem is that the value of time interval is difficult to decide. Normally, the time interval should be obtained according to specific simulation model. Lee[8] advanced a causal order-based time management model to improve the efficiency of time advance, but the non-causal event orders cannot be guaranteed.

In current time synchronization methods[6−10], there exists a default premise that control messages among nodes must be delivered without loss. Although the event consistency can be obtained, time advance has low efficiency and high occupied network bandwidth.

This paper puts forward MCTS algorithm based on the time advance interval method and Lookahead, and proves that just parts of control messages need reliability. MCTS is implemented in the time management service of BH RTI. Experiment results show that the algorithm is superior to others in time advance speed, occupied network bandwidth, and simulation scalability.

3 Grouped LBTS computation model

3.1 LBTS

In distributed simulation, LBTS is referred to the maximum simulation time during which the node can advance safely. In order to maintain the simulation time consistency, every node should not receive any outdated message whose simulation time is less than the node's current time. So, a node should advance the time no more than LBTS, otherwise it is likely to receive an outdated message in the future. The correctness of LBTS computation is one of the critical problems in time synchronization algorithms. The LBTS is defined as

$$\mathrm{LBTS}_i = \min\{T(j) + L(j)\}, \quad i \neq j. \tag{1}$$

In eq. (1), $T(j)$ is node j's current simulation time and $L(j)$ is its Lookahead. Lookahead represents the ability of simulation node to predict its further event, namely the node knows events to take place some time in the future and promises that new events will not be produced in the following period of Lookahead. In this way, a node can use Lookahead to publish its future events beforehand, and then all the nodes can advance the time simultaneously in parallel with the help of LBTS. Fujimoto et al.[11] proposed LBTS and eq. (1) in the draft of HLA time management design document under support of DMSO in 1996. Fujimoto et al.[12] also used eq. (1) in the design of RTI F.0.

3.2 LBTS characteristic analysis

LBTS-based time management demands that nodes coordinate to advance the non-backward logic time and outdated messages will be invalid. Hence, each node must guarantee that it will not send any message whose timestamp is less than the time it has requested to advance. Because every node has the ability to predict some future events, it can promise that any new event will not be produced with the timestamp in the following period of Lookahead. According to this promised time, every node can compute a maximum time for it to request to advance without inconsistency, and then it will not receive any outdated message. This maximum time is called LBTS. At the same time, the node should abide by the rule that the timestamp of a new event must be bigger than or at least equal to the current logic time plus Lookahead.

Suppose that there are simulation nodes $N_1, N_2, \ldots, N_n$, the current time of node N_i is t, and received event is E_i, whose timestamp is $E_i.\eta$. In order to analyze the relationship of a node's logic time and event timestamp, we have the following definitions:

1) Promised logical time (PLT): The node's current logic time plus its Lookahead, presented as PLT = t + Lookahead. PLT is the lower bound of valid timestamps of future events that the node will send out at logic time t. Then its following events will not have a timestamp less than PLT. So E_i, timestamp of a future event, must satisfy $E_i.\eta \geqslant$ PLT.

2) Safe event set (SES): A subset of all the received events in the node's TSO queue that is waiting for process. It is a function of current logic time t, presented as $\mathrm{SES}(t) = \{E_i \mid E_i.\eta \leqslant t\}$. All events in this set can be processed safely.

It can be proved that LBTS has the following properties.

Theorem 1. If the node's lower bound timestamp evaluates the minimum of other nodes' PLTs, this node will not receive any outdated messages.

Proof. Suppose that there are n simulation nodes, $N_1, N_2, \ldots, N_n$. let $N=\{N_i\}, j=1,\ldots, n$. Let their current logic time be $t_1, t_2, \ldots, t_n$ respectively. Let their PLTs be $\sigma_1, \sigma_2, \ldots, \sigma_n$ correspondingly, and the timestamps of their next events be $\eta_1, \eta_2, \ldots, \eta_n$.

According to the definition of PLT, we can get $\eta_1 > \sigma_1$, $\eta_2 > \sigma_2$, $\eta_3 > \sigma_3$, $\ldots$, $\eta_n > \sigma_n$. Considering that the proposition does not aim at a specific node, they can be arranged according to the ascending order of PLTs. Suppose $\sigma_1 < \sigma_2 < \sigma_3 < \cdots < \sigma_n$. Then there have $\sigma_1 < \eta_1$, $\sigma_1 < \eta_2$, $\ldots$, $\sigma_1 < \eta_n$. There are two situations of whether a node's PLT is the smallest or not.

1) For N_i, when $2 \leqslant i \leqslant n$, its PLT = σ_i, not the smallest. The minimum of others' PLTs is σ_1.

Let the node advances to time t_i', $t_i < t_i' < \sigma_1$. Then, there has $t_i' < \eta_1$, $t_i' < \eta_2$, $\ldots$, $t_i' < \eta_n$.

Then, it can be concluded that when a node has not the minimum PLT, it will not receive any outdated message if its LBTS evaluates the minimum PLT of other nodes.

2) For N_i, when i=1, its PLT is the smallest. The minimum of others' PLTs is σ_2.

Let the node advances to t_1', $t_1 < \sigma_1 < t_1' < \sigma_2$. Accordingly, its PLT becomes σ_1', $\sigma_1 < \sigma_1'$.

Now the timestamp of event that each node can send out is $\eta_1, \eta_2, \ldots, \eta_n$, respectively.

According to the definition of PLT, we can get $\eta_1 > \sigma_1'$, $\eta_2 > \sigma_2$, $\eta_3 > \sigma_3$, …, $\eta_n > \sigma_n$.

N_1 can receive events from other nodes and the events' timestamps is $\eta_2, \eta_3, \ldots, \eta_n$.

We have $t_1' < \sigma_2 < \eta_2$, $t_1' < \sigma_2 < \sigma_3 < \eta_3, \ldots, t_1' < \sigma_2 < \sigma_3 < \cdots < \sigma_n < \eta_n$.

So, when the node has the smallest PLT, it will not receive any outdated message if its LBTS evaluates the minimum PLT of other nodes.

From situations 1) and 2), Theorem 1 is true. QED

3.3 Grouped LBTS computation model

On the basis of Theorem 1, we can present the grouped LBTS computation model. Before the computation, nodes involved in the time advance are divided into several groups. If a node's LBTS changes, then other nodes should possibly have their LBTS changed. There are mainly three steps. Firstly, minimum time and the second minimum time are computed inside each group. Then global computation on all groups' minimum time and the second minimum time are done to get the global minimum time and the second minimum time from them. At last, every node's LBTS can be calculated out on its time advance state and the global time. The computation model is described below.

Computation model. Suppose that there are m coordination nodes in the system, respectively $\boldsymbol{C}=\{C_1, C_2, C_3, \ldots, C_m\}$. Every coordination node takes charge of one group's time synchronization including n_h ($1< h <m$) nodes. In other words, all the simulation nodes are divided into m groups and the node number in each group is n_h. Each node has simulation time t and PLT σ.

There are PLT sets $\Omega_1, \Omega_2, \ldots, \Omega_m$: $\Omega_1 = \{\sigma_{1,1}, \sigma_{1,2}, \sigma_{1,3}, \ldots, \sigma_{1,n1}\}$, $\Omega_2 = \{\sigma_{2,1}, \sigma_{2,2}, \sigma_{2,3}, \ldots, \sigma_{2,n2}\}$, …, $\Omega_m = \{\sigma_{m,1}, \sigma_{m,2}, \sigma_{m,3}, \ldots, \sigma_{m,nm}\}$.

For the node PLT set Ω_k, $1 \leqslant k \leqslant m$, coordination node C_k will have two variables (λ_k, δ_k) on all the PLTs σ in the group. λ_k and δ_k are the minimum one and second minimum one of the PLTs in the group. The computation formulae are represented as $\lambda_k = \sigma_{ki}$, where $(\sigma_{ki} \in \Omega_k) \wedge (\forall j(\sigma_{ki} \leqslant \sigma_{kj})$; $\delta_k = \sigma_{ki}$, where $(\sigma_{ki} \in \{\Omega_k - \{\lambda_k\}\}) \wedge (\forall j(\sigma_{ki} \leqslant \sigma_{kj})$.

Then, we get set Φ at $\Phi = \{\lambda_1, \delta_1, \lambda_2, \delta_2, \lambda_3, \delta_3, \ldots, \lambda_m, \delta_m\}$.

Let $\mu_k = \lambda_k$, $\mu_{k+m} = \delta_k$. Then $\Phi = \{\mu_1, \mu_2, \mu_3, \mu_4, \ldots, \mu_{2m-1}, \mu_{2m}\}$.

For set Φ, another two variables (ψ, ξ) can also be determined. ψ and ξ are the minimum one and second minimum one in set Φ. The computation is performed using formulae $\psi = \mu_k$, where $(\mu_k \in \Phi) \wedge (\forall j(\mu_k \leqslant \mu_j)$; $\xi = \mu_k$, where $(\mu_k \in \{\Phi - \{\psi\}\}) \wedge (\forall j(\mu_k \leqslant \mu_j)$.

After the global computation, we can get set $\Phi_{\min}$ of nodes whose PLT is the smallest and set $\boldsymbol{C}_{\min}$ of coordination nodes whose (λ_k, δ_k) is the smallest. $\Phi_{\min} = \{N_i \mid \sigma_i = \psi\}$, $\boldsymbol{C}_{\min} = \{C_k \mid \lambda_k = \psi\}$.

Variable ψ and ξ are the upper bound of time to advance of all nodes. For node with the smallest PLT, the upper bound is ξ. Those of others are all ψ. Consequently, every node's LBTS can be obtained

$$\mathrm{LBTS}_i = \begin{cases} \psi, & N_i \notin \Phi_{\min}; \\ \xi, & N_i \in \Phi_{\min}. \end{cases}$$

With the grouped LBTS computation model, it is easy to alleviate nodes' load by distributing the conservation and computation of time advance in different nodes. In this way, the time information exchange is mainly carried out in a smaller group, the traffic among nodes can also be reduced.

4 MCTS algorithm

In distributed simulation, nodes communicate via message delivery, and coordinate the time advance. Existing time management algorithms have a default premise that control messages in coordinating time advance should be delivered without loss. In these algorithms, any loss of control messages is not allowed. With the development of simulation scalability, this requirement demands much for network resources. The time advance of node will also slow down when there is an increase in node number. However, we find some characteristic of group communications in the exchange of time related control messages among nodes in time synchronization. Therefore, it will be beneficial if multicast can be introduced into time synchronization algorithm. Both the efficiency of time advance and network bandwidth occupation can be improved. Unfortunately, because of the premise of reliability requirement of control messages, IP multicast is not used in existing algorithms because of its unreliability. Whether messages should be delivered without loss to all the nodes that request during the time synchronization is investigated in this section.

4.1 Theorem of time synchronization message's reliability

In a time synchronization system based on grouped LBTS computation model, every node uses time synchronization messages to coordinate the time advance. In order to analyze the message reliability requirements, messages are categorized. As shown in Figure 1, there are n simulation nodes and one coordination node C in the system, and they send messages to each other so as to exchange the time related information. All these messages can be categorized into two kinds. One is of the messages sent out from the simulation nodes to the coordination node C. They contain the node's PLT σ and variable λ-δ computed on grouped computation model. Because λ and δ are computed from σ, we call this kind of message "σ message". The other kind is the messages sent out from the coordination node C to the simulation nodes. They contain ψ and ξ needed in LBTS calculation. We call them "ψ-ξ message". The delivery relationship between the two kinds of messages is shown in Figure 1.

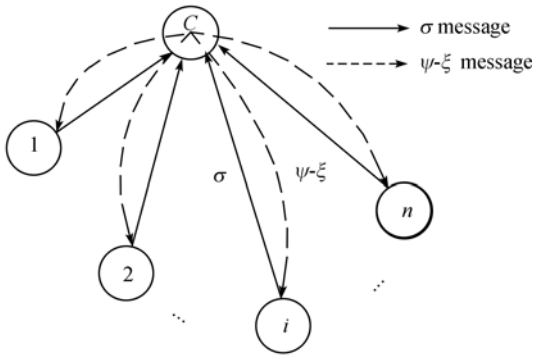

Figure 1 Category of messages.

For each node, its step pace of time advance and Lookahead can be different. The node PLT

can be the minimum, second minimum or others. In view of this, we categorize the nodes into min node (MIN), second min node (SEC) and general node (GEN). Then NodeType = enum{MIN, SEC, GEN}. The relationship between the message reliability and correct time advance is introduced below.

Theorem 2. The reliability of σ message is essential for correct time advance.

Proof. Suppose that there are n simulation nodes $N_1, N_2, \ldots, N_n$, whose current simulation time are $t_1, t_2, \ldots, t_n$ and promised time $\mathrm{PLT}_1, \mathrm{PLT}_2, \ldots, \mathrm{PLT}_n$ are $\sigma_1, \sigma_2, \ldots, \sigma_n$ respectively. For the convenience of expression, suppose the order of $t_1 < t_2 < t_3 < \cdots < t_n$, $\sigma_1 < \sigma_2 < \sigma_3 < \cdots < \sigma_n$.

Take the condition of four nodes as an example, as show in Figure 2(a). Three nodes are responsible for simulation logic and one node C takes charge of the coordination. NodeType (N_1) = MIN, NodeType (N_2) = SEC, NodeType (N_3) = GEN. Their time advance states, current logic time and promised time are shown in Figures 2(a)—(c). The three nodes send their σ to C in message. When no message is lost, C computes out the result of $\{\sigma_1, \sigma_2\}$. Next, we will discuss three conditions respectively when the node's NodeType is MIN, SEC or GEN.

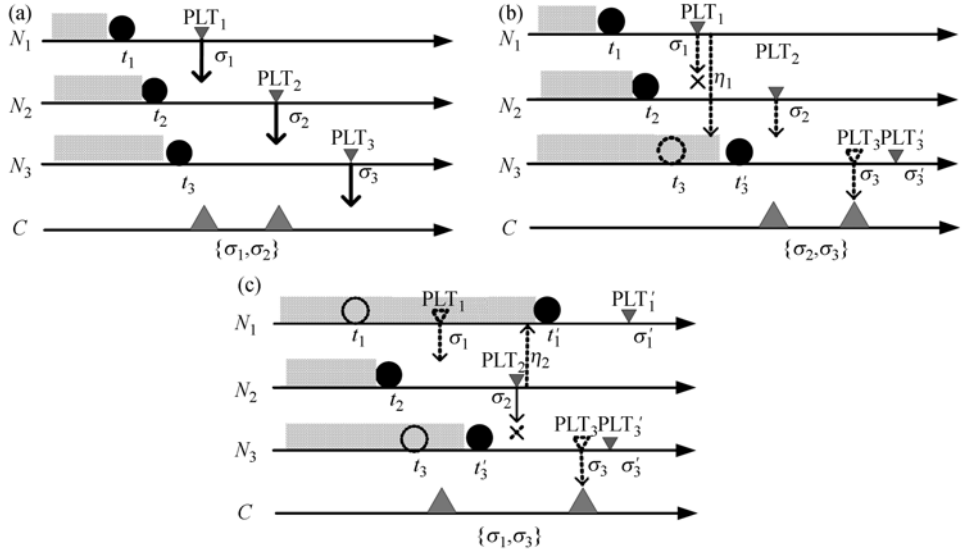

Figure 2 σ message lost. (a) Status of nodes' time advance before sending σ message; (b) σ message of node whose NodeType is MIN lost; (c) σ message of node whose NodeType is SEC lost.

Condition 1. σ message from node whose NodeType is MIN lost.

As shown in Figure 2(b), suppose that σ message sent from N_1 to coordination node C is lost. Then, C computes and returns the result $\{\sigma_2, \sigma_3\}$ to every node. At that time, N_3's upper bound of safe advance time $\mathrm{LBTS}_3 = \sigma_2$, which means it can advance to any time less than σ_2. Suppose that N_3 advances to t_3', $\sigma_1 < t_3' < \sigma_2$. Before N_1 makes the next advance, its PLT will keep σ_1. N_1 sends a message with timestamp η_1, $\sigma_1 < \eta_1 < t_3'$. N_3 receives a message from N_1, $\eta_1 < t_3'$, which is an outdated message for N_3. It means some inconsistency occurs. Therefore, σ message sent from a node whose NodeType is MIN must be guaranteed.

Condition 2. σ message from node whose NodeType is SEC lost.

As shown in Figure 2(c), suppose σ message that is sent from N_2 to coordination node C is lost.

Then C computes and returns the result $\{\sigma_1, \sigma_3\}$ to every node. Now, the LBTS of nodes are $LBTS_1 = \sigma_3$, $LBTS_2 = \sigma_1$, $LBTS_3 = \sigma_1$. Each node can advance to any future time that is less than its LBTS. Suppose N_1 advances to t_1', $\sigma_2 < t_1' < \sigma_3$. Correspondingly, its PLT changes to σ_1'. At that time, N_2 does not make an advance, so PLT_2 stays no change. N_2 sends a message with time-stamp η_2. $\sigma_2 < \eta_2 < t_1'$. Then N_1 will receive the message from N_2, $\eta_2 < t_1'$, which is an outdated message for N_1. It means some inconsistency occurs. Therefore, σ messages sent from a node whose NodeType is SEC must be reliable.

Condition 3. σ message from node whose NodeType is GEN lost.

When the message sent by the general node is lost, the events can be consistent at that time. But in the period of time advance, every node may become the min node or second min node. Before the time get consistency, each node can not judge whether it is min or second min by itself. So, σ messages sent from a node of GEN NodeType must also be reliable.

It follows from Conditions 1—3 that Theorem 2 is true. QED

From the proof above, during the period of time advance, messages bringing time information of nodes play an important role. It is impossible to make global time synchronization if any time information of one node cannot be obtained by the coordination node. Then we can deduce that σ messages should be delivered without loss in the grouped LBTS computation model.

Theorem 3. The MIN node's receiving ψ-ξ message is required to be reliable for time advance in time, while other nodes' receiving ψ-ξ message can be unreliable.

Proof. Suppose the state and attribute of each node as shown in Figure 3(a). There are three nodes N_1, N_2, N_3 currently, and the time requested to advance are t_1, t_2, t_3 respectively. The vertical line of LBTS indicates the upper bound of safe time advance of the nodes. Because $LBTS = \sigma < t_1 < t_2 < t_3$, the nodes are all in the pending state. NodeType (N_1) = MIN, NodeType (N_2) = SEC, NodeType (N_3) = GEN. Coordination node C receives the consistency message from every node and computes out $\{\lambda, \delta\} = \{\sigma_1, \sigma_2\}$. Then we will discuss possible loss of the following ψ-ξ message in three conditions respectively when the node's NodeType is MIN, SEC or GEN.

Condition 1. A node whose NodeType is MIN fails to receive ψ-ξ message.

As shown in Figure 3(b), coordination node C sends ψ-ξ message to each node and N_1 fails to receive it. N_2 and N_3 receive the message, conduct the computation, and then get the result $LBTS_2 = LBTS_3 = \sigma_1$. Because N_1 does not receive the message, its $LBTS_1$ keeps unchanged. Since $LBTS_2 > t_2$ and $LBTS_3 > t_3$, N_2 and N_3 are allowed to make the time advance. N_1 continues to keep pending state for $LBTS_1 = \sigma < t_1$ and its PLT_1 stays. N_2 and N_3 make a next advance and send out their PLTs. Their times to advance are t_2' and t_3'; and their promised times are σ_2' and σ_3'. Then coordination node C makes computation, gets the new result that $\{\lambda, \delta\} = \{\sigma_1, \sigma_2'\}$ and sends the ψ-ξ message again. If node N_1 loses the message again, it will keep the pending state. At this time, N_2 and N_3 compute and get the result of $LBTS_2 = LBTS_3 = \sigma_1 < \sigma_2' < \sigma_3'$. Then they will keep pending. Consequently, a dead-lock occurs. All the three nodes come into the pending state and cannot advance.

In order to make time advance in time, ψ-ξ messages must be received without loss in a node whose NodeType is MIN. Otherwise, other nodes can advance to σ_1 at most.

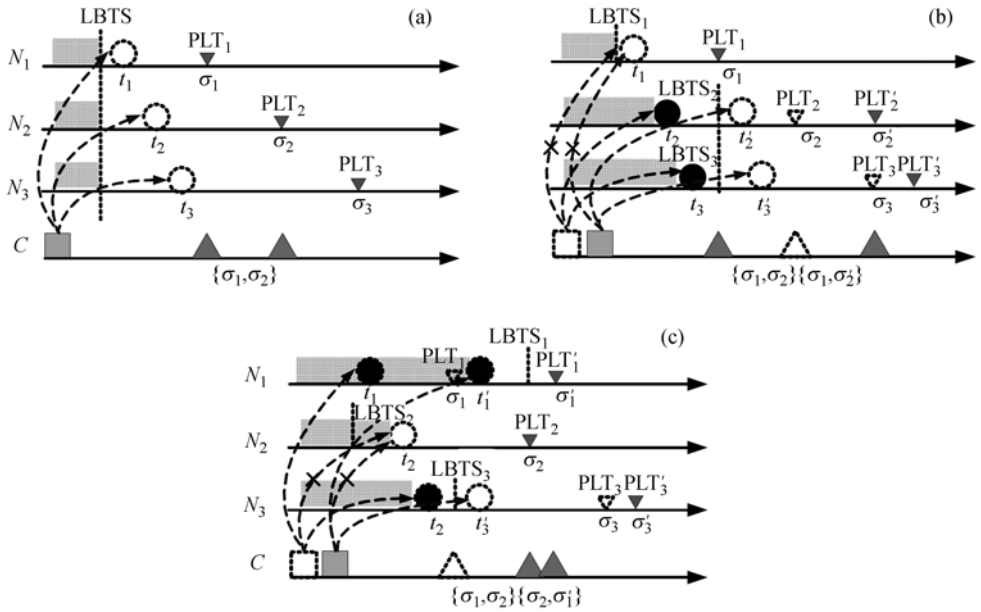

Figure 3 ψ-ξ message lost. (a) Current status of time advance of nodes; (b) ψ-ξ message of node whose NodeType is MIN lost; (c) ψ-ξ message of node whose NodeType is SEC lost.

Condition 2. A node whose NodeType is SEC fails to receive ψ-ξ message.

As show in Figure 3(c), coordination node C sends ψ-ξ message to each node and N_2 fails to receive the message. N_1 and N_3 receive the message, conduct the computation, and get the result $\mathrm{LBTS}_1 = \sigma_2$, $\mathrm{LBTS}_3 = \sigma_1$. Because N_2 lost the message, its LBTS_1 keeps unchanged. Since $\mathrm{LBTS}_1 > t_1$ and $\mathrm{LBTS}_3 > t_3$, N_1 and N_3 are allowed to make time advance. N_2 continues to keep pending for $\mathrm{LBTS}_2 < t_2$ and its PLT_1 has no change. N_1 and N_3 do a next time advance to t_1' and t_3'. Their promised times are σ_1' and σ_3', $\sigma_2 < \sigma_1' < \sigma_3'$. N_1 and N_3 send new PLTs to C. When C receives the new PLTs, it computes and gets $\{\lambda, \delta\} = \{\sigma_2, \sigma_1'\}$, and then sends messages to other nodes. At this time, N_2 becomes the latest node among all nodes, i.e., NodeType (N_2) = MIN.

If a node whose NodeType is SEC fails to receive the ψ-ξ message, its NodeType will eventually become MIN after several steps of time advance.

Condition 3. A node whose NodeType is GEN fails to receive ψ-ξ message.

From the analysis of Condition 2, we can get the similar conclusion of the node whose NodeType is GEN. If it cannot receive ψ-ξ message continuously, then after several time advances, its NodeType will become SEC and then become MIN.

It follows from Conditions 1—3 that Theorem 3 is true. If MIN node fails to receive ψ-ξ, a dead-lock will occur. If SEC or GEN nodes fails to receive ψ-ξ, not the consistency but the speed of time advance will be affected. QED

From the proof above, for the ψ-ξ messages the coordination node returns to nodes, not all the messages are required to be received without loss. In fact, only those nodes that constrain the time advance of the system (i.e., the later nodes) need to receive each of them. Consequently, it is essential for the MIN node to receive ψ-ξ messages in the grouped LBTS computation model for time advance in time. It is not strictly required that other nodes should receive each ψ-ξ message.

From Theorems 2 and 3, in the period of time synchronization, it is not required for all the nodes to receive control messages without loss. The time advance of simulation just lies mainly on the reliability of those messages of critical nodes. Hence, as long as those nodes that constrain the system's advance can receive the critical control messages in time, time advance of the whole simulation will continue.

Theorem of time synchronization message's reliability. In the time synchronization system based on grouped LBTS computation model, so long as σ message's delivery and MIN node's receiving ψ-ξ message are reliable, simulation will conduct time advance correctly.

Proof. In the simulation based on grouped LBTS computation model, if σ message is received without loss, the coordination node gets time of all nodes correctly. Then the coordination node can compute LBTS in time. Thus, the global time is obtained correctly and timely.

In the course of time synchronization, the time advance is constrained by the MIN node. If the MIN node keeps pending and cannot advance, the whole simulation will not make time advance in time. If the MIN node receives ψ-ξ messages without loss, it can advance all along and the simulation will continue.

For other nodes, when one fails to receive ψ-ξ messages, it cannot advance for some time and finally will become the MIN node. Thus, it can receive ψ-ξ message without loss, and continues to make time advance.

To sum up, if only both the σ messages and the MIN node's receiving ψ-ξ messages are with no loss, simulation can conduct time advance correctly. QED

4.2 Design of MCTS algorithm

The characteristics of LBTS, together with the requirements for the reliability of control messages in time synchronization, have been investigated. Then the algorithm MCTS is brought forward based on the grouped LBTS computation model and theorem of time synchronization control message's reliability. All nodes are categorized into three roles, and each node is only responsible for some part of the maintenance and computation related to time synchronization. According to the message type, the node uses multicast or reliable unicast in control message delivery. In this way, coordination groups get time synchronization.

4.2.1 Node-role category. Based on the grouped LBTS computation model, nodes are categorized into three types of roles: member node, local coordination node and global coordination node. Each node takes their work of time synchronization on its role. The relationship of nodes and their roles is shown in Figure 4.

Explanation of the three kinds of roles is as below.

1) Member node (MN) responsible for the simulation logic. Its attributes include Lookahead L, promised logic time PLT, upper bound of safe advance time LBTS and safe event set SES. The related concepts have been given in section 3.

2) Local coordinator node (LCN) responsible for the time synchronization of MNs under its charge. Its attributes include local minimum timestamp (LMTS), local second minimum timestamp (LSTS) and minimum node set (MNS). LMTS and LSTS are computed by the LCN with its MNs, and the two are corresponding to λ and δ respectively in LBTS computation model. For each local coordination node LCN_k, MNS is corresponding to $\Phi_{\min}$. In the set Ω_k containing PLTs of MNs in the group, $\Omega_k = \{\mathrm{PLT}_{k1}, \mathrm{PLT}_{k2}, \mathrm{PLT}_{k3}, \ldots, \mathrm{PLT}_{kn}\}$. Attributes of the LCN can be expressed as $\mathrm{LMTS}_k = \mathrm{PLT}_{ki}$, where $(\mathrm{PLT}_{ki} \in \Omega_k) \wedge (\forall \mathrm{j}\ (\mathrm{PLT}_{ki} \leqslant \mathrm{PLT}_{kj})$; $\mathrm{LSTS}_k = \mathrm{PLT}_{ki}$, where

$(PLT_{ki} \in \{\Omega_k - \{LMTS_k\}\}) \wedge (\forall j(PLT_{ki} \leqslant PLT_{kj})$.

3) Global coordinator node (GCN) responsible for the synchronization of LCNs in the simulation system. Its attributes include global minimum timestamp (GMTS), global second minimum timestamp (GSTS) and minimum local coordinator set (MLCS). GMTS and GSTS are computed by GCN with all the LCNs. The two are corresponding to ψ and ξ respectively in grouped LBTS model. For the GCN, MLCS corresponds to $\boldsymbol{C}_{\min}$ in the model. And also, set Φ consists of all the LMTS and LSTM. $\Phi = \{LMTS_1, LSTS_1, LMTS_2, LSTS_2, \ldots, LMTS_m, LSTS_m\}$. Let $LTS_i = LMTS_i$, $LTS_{m+i} = LSTS_i$, $1 \leqslant i \leqslant m$. Then set Φ can also be expressed as $\Phi = \{LTS_1, LTS_2, LTS_3, LTS_4, \ldots, LTS_{2m-1}, LTS_{2m}\}$. Each attribute of the GCN can be expressed as

$$GMTS = LTS_k, \text{ where } (LTS_k \in \Phi) \wedge (\forall j\ (LTS_k \leqslant LTS_j),$$

$$GSTS = LTS_k, \text{ where } (LTS_k \in \{\Phi - \{GMTS\}\}) \wedge (\forall j\ (LTS_k \leqslant LTS_j),$$

$$MLCS = \{LCN_k \mid LMTS_k = GMTS\}.$$

On the GCN's attributes, the attribute MNS of LCN is expressed as $MNS_k = \{MN_{ki} \mid PLT_{ki} = GMTS\}$.

Thereby, LBTS can be obtained as

$$LBTS_{ki} = \begin{cases} GMTS, \text{ when } MN_{ki} \notin MNS_k; \\ GSTS, \text{ when } MN_{ki} \in MNS_k. \end{cases}$$

The role's function and node's attributes have been introduced. It can be seen that the LCN and MN have a relationship of one to many. So is that of GCN and LCN. As shown in Figures 4 and 5, LCN and GCN are the coordination nodes which coordinate the time synchronization of nodes under their charge. MN sends σ messages to its LCN, and then LCN sends σ message to GCN. Then it is time for GCN to compute out the GMTS and GSTS. The ψ-ξ will be published in two levels. GCN multicasts ψ-ξ message to LCNs. There are two choices for LCNs to publish ψ-ξ to their MNs, similar multicast to MNs or calculating LBTS for each MN and sending individually.

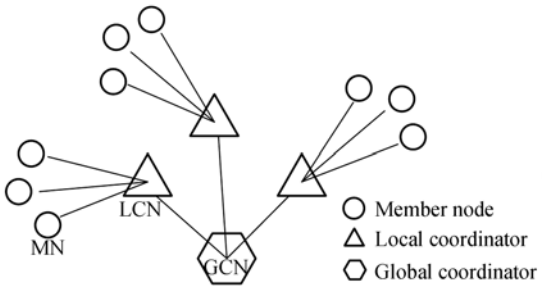

Figure 4 Node-role category.

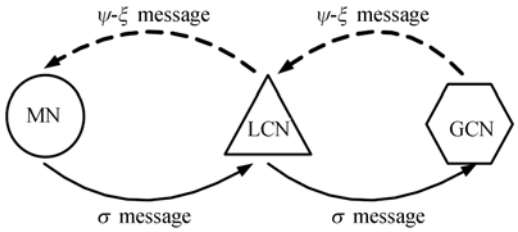

Figure 5 Time message type.

4.2.2 Introduce IP multicast to the delivery of time synchronization messages. In the distributed simulation, nodes coordinate time advance by sending time messages. Based on the theorem of time synchronization message's reliability proved above, different communications can be applied in the message delivery according to the relationship of messages and nodes. For messages that need reliability, reliable unicast is appropriate. When it comes to messages that need not definite reliability, IP multicast will be a better choice. Multicast can help much in lowing down the cost of network resources and improving the delivery efficiency.

According to the theorem of reliability, σ message need to be reliable. So MCTS algorithm

adopts TCP to deliver σ messages. ψ-ξ message only demands reliable transportation of the MIN node to receive. Then unreliable UDP multicast is adopted to send ψ-ξ messages, and extra copies will be sent to the MIN nodes via TCP. The extra copies were sent to avoid the possible loss of the message. The message delivery in the MCTS algorithm is illustrated in Figure 6, in which real line indicates the reliable unicast of TCP and dashed line indicates the unreliable UDP multicast.

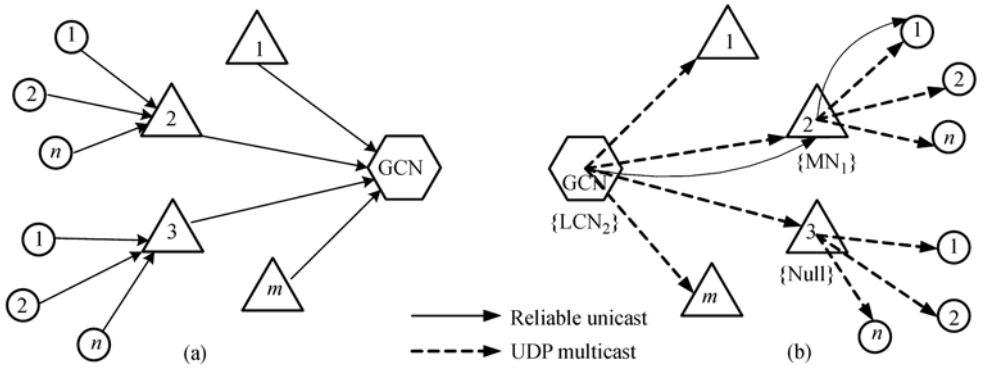

Figure 6 Delivery of time synchronization messages between nodes. (a) Delivery of σ message; (b) delivery of ψ-ξ message.

Figure 6(a) is a sketch map of σ message delivery of the nodes. Messages from MN to LCN and those from LCN to GCN are delivered by TCP, reliable unicast. Figure 6(b) is a sketch map of ψ-ξ message delivery of the nodes. The reliable unicast and multicast are used in ψ-ξ message delivery in different situations. For GCN's ψ-ξ to LCNs, GCN firstly multicasts to all the LCNs, and then sends again in TCP to those in its MIN node set. It can a similar process for LCN's sending ψ-ξ to MNs. LCN can also calculate an MN's latest LBTS and send to them individually.

According to Theorem 3, there are some nodes that should receive ψ-ξ messages without loss, which is called a reliable node set (RNS) here. GCN, or maybe LCN with similar multicast process, need to use RNS in publishing ψ-ξ. The set can be denoted by

$$\mathrm{RNS_{LCN}} = \{\mathrm{MN}_i \mid \mathrm{MN}_i \in \mathrm{MNSet} \wedge \mathrm{PLT}_i = \mathrm{GMTS}\},$$

$$\mathrm{RNS_{GCN}} = \{\mathrm{LCN}_k \mid \mathrm{LCN}_k \in \mathrm{LCNSet} \wedge \mathrm{LMTS}_k = \mathrm{GMTS}\}.$$

MN_i is the ith MN in some LCN, and MNSet is the set of all MNs under the charge of the LCN. LCN_k is the kth LCN in GCN, and LCNSet is the set of all LCNs.

4.2.3 Coordinate the simulation time advance. Based on the definitions above, the coordination of time advance among the three types of nodes can be divided into six steps for a round as follows.

Step 1. Member node MN_{ki} requests for a time advance. It computes the PLT out, and then sends σ message to local coordinator LCN_k to inform its current PLT.

Step 2. When LCN_k receives the time information from MN_{ki}, it updates the PLT of MN_{ki} in its local memory. Then the new PLT_{ki} should be investigated to find out whether it affects the value of (LMTS_k, LSTS_k) of LCN_k. If the two have no change, there will be nothing to do and the LCN_k waits for next message. Otherwise, new (LMTS_k, LSTS_k) will be computed out according to the promised logic times (PLT_{k1}, PLT_{k1}, …, PLT_{kn}) of the member nodes (MN_{k1}, MN_{k2}, …, MN_{kn}) under its charge. Then an σ message is sent out to GCN, which informs the new local time attribute.

Step 3. When GCN receives the time information from LCN_k, it updates the attributes of

LCN_k in its local memory. Then the new ($LMTS_k$, $LSTS_k$) will be investigated to find out whether it affects the (GMTS, GSTS) of GCN. If not, nothing will be done and the GCN waits for next message. Otherwise, new (GMTS, GSTS) will be computed out according to the {($LMTS_1$, $LSTS_1$), ($LMTS_2$, $LSTS_2$), …, ($LMTS_m$, $LSTS_m$)} of all local coordinator (LCN_1, LCN_2 , …, LCN_m).

Step 4. GCN gets new attribute value (GMTS, GSTS). Then it multicasts ψ-ξ message to all LCN nodes, announcing the latest time information. Then the set RNS_{GCN} is calculated out. For any node $LCN \in RNS_{GCN}$, GCN sends the ψ-ξ message again via TCP.

Step 5. For any $LCN_k \in RNS_{GCN}$, it may receive a ψ-ξ message twice by multicast and by TCP. So firstly, the node has to check whether there is a duplicate message. If duplicate, the latter will be discarded. There are two alternative ways for LCN_k to treat with the ψ-ξ message. The one is to multicast ψ-ξ message to MNs and then send again to $MN \in RNS_{LCN}$ via TCP. The other is to compute the LBTS of each MN according to the global (GMTS, GSTS), and then send to the MN individually. The first method has advantage in bandwidth requirements but it requires more dynamic multicast addresses. The second method is easy to be implemented and reduces the work of MNs. Either is practical and we select the second method here in implementation for its simplicity.

Step 6. Node MN_{ki} receives the coordination message from LCN_k, and updates its own LBTS. If the time to advance goes beyond the range of safe advance, i.e., is bigger than LBTS, the node will keep pending. Otherwise, the node conducts a time advance, processes its SES, and then a next round of time advance starts.

5 Experiment and result analysis

MCTS algorithm is applied in the time management service implementation of BH RTI (http://www.hlarti.com). The service implementation includes two parts. One is the time management server TMServer, corresponding to the GCN of the algorithm. The other is a module of the distributed RTI execution RtiExec, corresponding to the LCN. Several experiments have been conducted to evaluate the algorithm.

5.1 Experiment configuration

The experiment configuration of network, host and software is given in Table 1. Experiment was carried out on MCTS-based time management service of BH RTI. At the same time, some other RTIs were also tested in the same condition, including DMSO RTI 1.3NGv6, Mak RTI 3.0 and Pitch pRTI 1516LE. Among the computers used in the experiments, we took one as the server and the rest as simulation hosts. Since RTIs have various architectures, the deployment varies with different RTIs.

Table 1 Experiment configuration

Network	100M Ethernet, HuaWei Quidway S3928
Computer	CPU P4 3.40 GHz, Memory 1 GBytes, Windows XP. Total host number is 11.
Software	RTI software: BH RTI 2.3, DMSO RTI 1.3NGv6, Mak RTI 3.0, pRTI 1516LE DMSO Benchmark (RTI performance benchmark published by DMSO) Network traffic Analyzer: IRIS v4.07.1 (http://www.eeye.com/)

BH RTI 2.3: Service host starts BH_TMServer, and each simulation host starts BH_RtiExec

and Benchmark.

DMSO RTI 1.3NGv6: Service host starts DMSO RTI, and each simulation host starts Benchmark.

Pitch pRTI1516LE: Service host starts pRTI, and each simulation host starts Benchmark.

Mak RTI 3.0: Service host starts Mak RTI, and each simulation host starts Benchmark.

5.2 Issues to evaluation

Time advance speed (TAS) and vacant network bandwidth occupied (VNBO) are regular in performance evaluation for time synchronization. An important characteristic of MCTS is the introduction of multicast to the algorithm. In order to validate its feasibility, loss effect ratio (LER) is adopted to evaluate the influence of message loss on time advance speed. Furthermore, simulation member scale (SMS) is used to test the simulation scalability.

Time advance speed is referred to as the times of time advance per second. It is used to weigh the efficiency of time synchronization. The unit of TAS is times per second. Suppose that there are altogether n_timeAdvance times of time advance in the period from t_1 to t_2. Then, TAS can be expressed as TAS = n_timeAdvance/(t_2–t_1).

Vacant network bandwidth occupied is referred to as the network bandwidth that an RTI occupies per second in time synchronization when there is no attribute value updates or interactions. It is used to weigh the bandwidth cost of time synchronization. The unit is Mbps. Network traffic analyzer is used for this item. There can be a rough calculation. For time from t_1 to t_2, suppose that the number of message exchange of the host is n, denoted by $\{m_1, m_2, m_3, \ldots, m_n\}$ and the size of each message is $\{l_1, l_2, l_3, \ldots, l_n\}$. Then, $\text{VNBO} = \sum_{i=1}^{n} (m_i * l_i)/(t_2 - t_1)$.

Loss effect ratio is the ratio of TAS when there are some messages lost to that when no message is lost. It is used to find out the influence of multicast loss on time synchronization. The unit is percentage. Suppose that the TAS of the node is tas_lostX when LER is x%, and the time advance speed is tas_noLoss when LER is 0 % (namely there is no multicast message lost). Then, LER can be expressed as LER = tas_lostX / tas_noLoss.

SMS discloses the relationship between the number of MNs and the time advance speed. It is used to test the scalability of time management service of RTI.

5.3 Result and analysis

The experiment result of TAS is shown in Figure 7. For BH RTI, one host starts TMServer, and the rest 2/5/10 hosts start RtiExec and some Benchmark programs. For the other three kinds of RTIs, one host starts RTI server and the rest 2/5/10 hosts start some Benchmark programs. The node's Lookahead is set to 1, and the time step to advance is set to 10. We tested the TAS in the conditions of different numbers of hosts and nodes. As seen from Figure 7(a), the TAS of BH RTI is much higher than those of the other three. With the increase of node number, the TAS of each RTI is lower than that of 2 hosts (Figure 7(b)). The Mak RTI used in the experiment can only support 37 nodes to join in. BH RTI's TAS is also higher than those of other RTIs. In the experiment shown in Figure 7(c), the scale is even larger. Similarly, with the increase of node number, the TAS of each RTI decreases further. There is no data in some conditions. Mak RTI cannot have the experiments of 50 nodes or 100 nodes, and DMSO RTI cannot have the experiments of 100 nodes. BH RTI has a small decrease in its TAS and it can still keep a relatively higher TAS than others in the condition of 100 nodes.

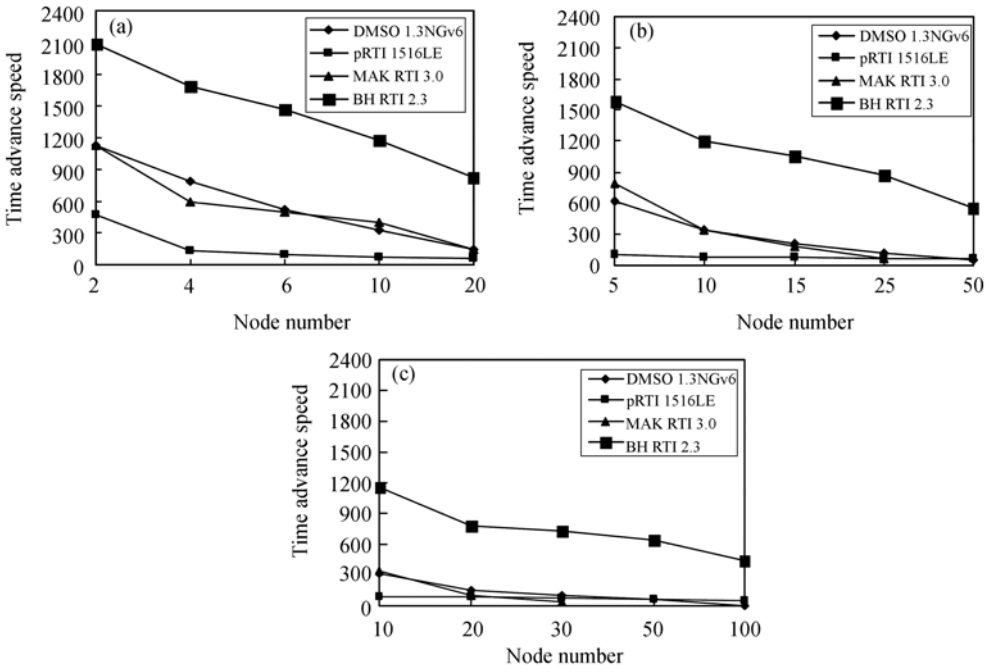

Figure 7 Test result of time advance speed. (a) TAS of 2 hosts; (b) TAS of 5 hosts; (c) TAS of 10 hosts.

While conducting the experiments of TAS shown in Figure 7, the VNBO of RTI time management is also tested in each experiment (Figure 8). The bandwidth analyzer used in the experiment is IRIS v4.07.1. Because the pRTI we used is an evaluation version which only provides 100 times of function call, the measure may not be so exact. So pRTI was not included in this experiment. The experiment result of 2 hosts is shown in Figure 8(a). Mak RTI occupies the highest bandwidth up to 28%. DMSO RTI occupies the second, and BH RTI occupies the least bandwidth. The experiment result of 5 hosts, as shown in Figure 8(b), is similar to that of Figure 8(a). Because Mak RTI and DMSO RTI did not have some experiments of 50 nodes or 1000 nodes, the corresponding VNBOs are also absent. The experiment result of 10 hosts is shown in Figure 8(c). In the condition of 10 hosts and 100 nodes, the VNBO of BH RTI is only about 3 percent of the total bandwidth.

Figure 9 shows the VNBO of the time management server of BH RTI 2.3, also in the same experiments above. It can be seen that as the GCN server for coordination, its VNBO is only a little higher than those of other hosts.

Figure 10 shows the experiment result of LER in the condition of 20, 50 and 100 nodes. The number of simulation hosts is 2, 5 or 10 respectively, and there are 10 member nodes in each host to join the time synchronization. The experiment simulated 11 instances of multicast loss ratio from 0—100% by discarding messages intentionally in the network layer of RTI. The Lookahead of a node evaluates the number index of its host, and the time step to advance values 1. The experiment results indicate that with an increase in node number, the impact of the loss ratio of multicast on the TAS is also increasing. When some messages via multicast are lost, the simulation can make time advance correctly. In the extreme conditions, for example, when the loss ratio is 100%, nodes' time advance speed of three kinds of scale is 79%, 76% and 57% respectively of the speed when no message is lost. Nodes in all the conditions can make time advance correctly,

but the TAS is some lower.

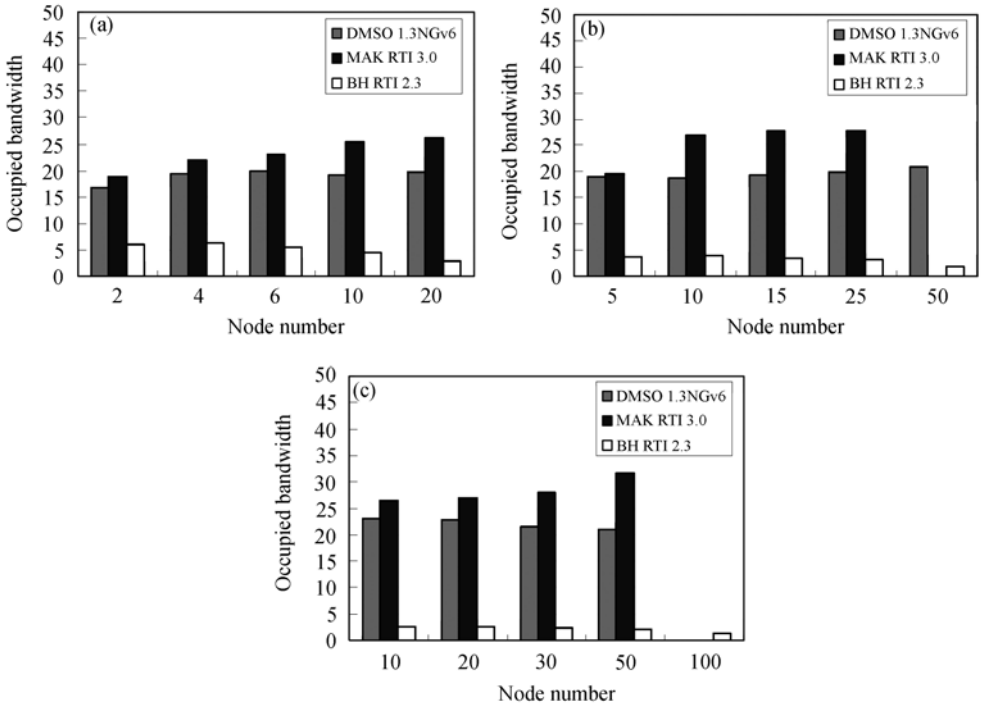

Figure 8 Test result of occupied network bandwidth. (a) VNBO of 2 hosts; (b) VNBO of 5 hosts; (c) VNBO of 10 hosts.

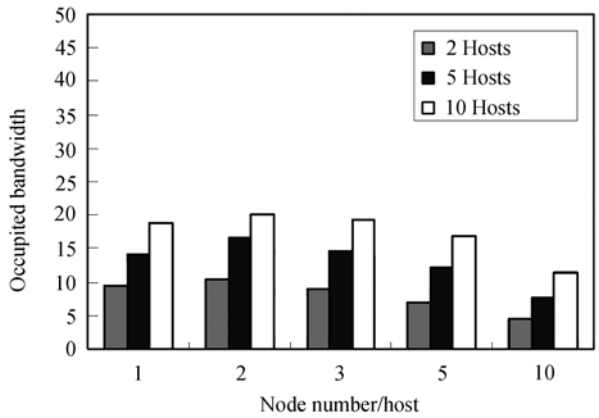

Figure 9 Test result of occupied network bandwidth of time management server.

Experiment result of SMS of BH RTI's time management is shown in Figure 11. Each node's TAS is tested in the conditions of 10 hosts and 100—1000 nodes. From Figure 11 we can see that with an increase in node number, TAS of the node will decline and go even gradually. When the node number is 1000, the advance speed is about 50 times per second. The speed is close to that of pRTI 1516LE of 100 nodes and that of DMSO RTI 1.3NGv6 of 50 nodes, and a little higher than that of MAK RTI 3.0 of 30 nodes. The experiment result indicates that the time management of BH RTI 2.3 can support a larger scale of simulation, whose TAS can attain 50 times per second at a scale of 1000 nodes.

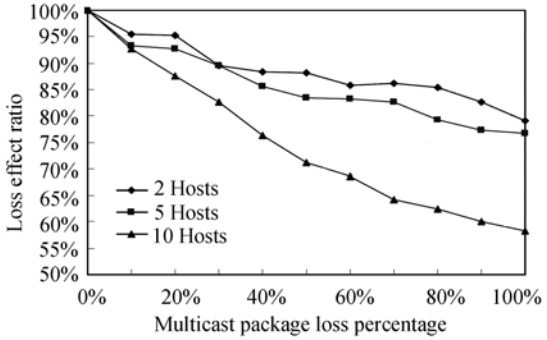

Figure 10 Test result of loss effect ratio.

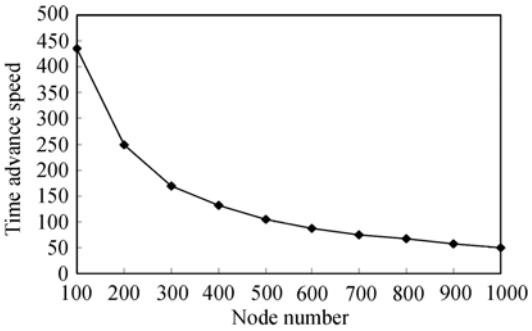

Figure 11 Test result of node scale of time management in BH RTI 2.3.

6 Conclusion

This paper investigates the time synchronization algorithm in distributed simulation, puts forward a time synchronization algorithm for large-scale distributed simulation. The analysis of the upper bound of time advance and characteristics of time coordination has been made, and a grouped LBTS computation model is given. Theorem of time synchronization message's reliability is proved. According to the characteristics of group communication among nodes, together with the reliability requirements of nodes for time synchronization messages, we introduced IP multicast to the delivery of time synchronization messages. At last, the algorithm is applied to the time management implementation of BH RTI. Experiment result demonstrates that the algorithm is superior to other similar systems in time advance speed, vacant network bandwidth occupied, and simulation scalability. Its time advance speed of 1000 nodes is about 50 times per second which is approximate to the speed of other systems of 100 nodes.

1 Fujimoto R M. Parallel simulation: parallel and distributed simulation systems. In: Proceedings of the 33rd Winter Simulation Conference. Virginia: IEEE, 2001. 147－157
2 Fujimoto R M. Parallel simulation: distributed simulation systems. In: Proceedings of the 35th Winter Simulation Conference. Louisiana: ACM, 2003. 124－134
3 Fujimoto R M. Time management in the high level architecture. SIMULATION, 1996, 71(6): 60－67
4 Cai W T, Turner S J, Lee B S, et al. An alternative time management mechanism for distributed simulations. ACM Trans Model Comput Simul, 2005, 15(2): 109－137
5 Chandy K M, Misra J. Distributed deadlock detection. ACM Trans Comput Syst, 1983, 1(2): 4－9
6 Mattern F. Efficient algorithms for distributed snapshots and global virtual time approximation. J Parallel Distr Comput, 1993, 18(4): 423－424
7 Fujimoto R M. Exploiting temporal uncertainty in parallel and distributed simulation. In: Proceedings of the 13th Workshop on PADS. Georgia: IEEE, 1999. 46－53
8 Lee B S, Cai W T, Zhou J L. A causality based time management mechanism for federated simulation. In: Proceedings of the 15th IEEE/ACM/SCS Workshop on PADS. California: IEEE, 2001. 83－90
9 Wang X G, Turner S J, Low M Y H, et al. Optimistic synchronization in HLA-based distributed simulation. In: Proceedings of the 18th IEEE/ACM/SCS Workshop on PADS. Austria: ACM, 2004. 123－130
10 Morillo P, Orduna J M, Duato J. A scalable synchronization technique for distributed virtual environments based on networked-server architectures. In: Proceedings of the 2006 International Conference on Parallel Proceeding. Ohio: IEEE, 2006. 74－81
11 Fujimoto R M, Weatherly R M. Time management in the DoD high level architecture. In: Proceedings of the 10th Workshop on PADS. Pennsylvania: IEEE, 1996. 60－67
12 Carothers C D, Weatherly R M, Fujimoto R M, et al. Design and implementation of HLA time management in the RTI version F.0. In: Proceedings of the 29th Winter Simulation Conference. Georgia: IEEE, 1997. 373－380