Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds.

TIME MANAGEMENT IN HIERARCHICAL FEDERATION USING RTI-RTI INTEROPERATION

Min-Wook Yoo ,
Chang Beom Choi,
Tag Gon Kim

Electrical Engineering
KAIST
Daejeon, KOREA

ABSTRACT

High Level Architecture (HLA) provides interoperation of federates, and hierarchical federation was proposed to extend interoperability to the federation level. In a hierarchical federation, several federations make a hierarchical structure using proxies which represent the behavior of the federations. Time synchronization of federates is essential for interoperation and should be accomplished in hierarchical federation. Previous research studies have suggested a time synchronization algorithm based on LITS (Least Incoming Time Stamp) but a deadlock problem remains in some cases. This paper proposes time management in hierarchical federation. We propose time synchronization algorithm to solve the deadlock problem and stipulates the time states of the proxy for representing federation. A proxy model is constructed based on the proposed algorithm and the algorithm is verified to work correctly in hierarchical federation.

1 INTRODUCTION

High Level Architecture (HLA) is an IEEE1516 standard for the interoperation of distributed simulators (IEEE Std. 1516.1, 2010). Run Time Infrastructure (RTI) is a middleware which provides HLA services. In HLA, a federation is created in an RTI and simulators called federates join to it for interoperation. Each federate simulates its own system and interacts with other federates through the RTI. Time synchronization is essential for interoperation of simulations and HLA specifies a time management (TM) service for it. TM services are used to synchronize the logical time of federates and exchange time stamp order (TSO) messages among federates.

When two or more federations perform a simulation together, the interoperation of federations is required. There are several methods for the interoperation of federations but only a proxy method can be applied without modifying the RTI. A proxy is composed of two agents, and each agent is used to represent the behavior of a federation. Because HLA defines only the interoperation of federates, there are some problems in the proxy method (Dingel et al. 2002), and several methods, like MOM service or Agent-User protocol, are proposed to solve problems (Yoo and Kim 2007). A proxy is used for the interoperation of two federations and a hierarchical federation can be constructed based on the proxy (Myjak and Sharp 1999). In hierarchical federation, several proxies are used to connect federations.

To interoperate federations completely, time management should be achieved in hierarchical federation. Compared to a federation, there are several RTIs in hierarchical federation and each RTI can recognize a part of federates and agents. Therefore, each agent should represent the time state of all the other federate which join the other federation. Previous research studies have used LITS (Least Incoming Time Stamp) for time synchronization (Chen et al. 2010) but there remain deadlock problems in some cases.

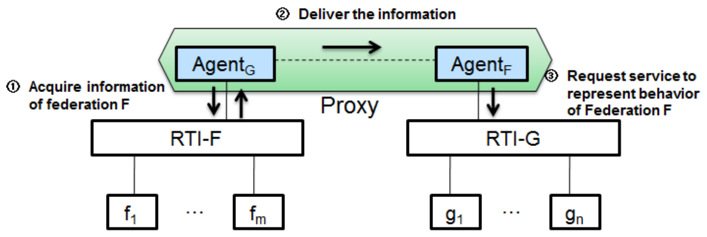
In this paper, we propose time management in hierarchical federation. We propose a modified time synchronization algorithm to solve the deadlock problem. The proposed algorithm also uses LITS to represent federation, and additional time advance requests are used to solve the problem. Agent model is constructed based on the proposed algorithm and the proposed algorithm is verified using it. In addition, we stipulates time states of agents and they are used in initialization and simulation.

The paper's structure is as follows. Section 2 introduces hierarchical federation. Section 3 defines the time states of agents. Section 4 describes the proposed time management algorithm in hierarchical federation and verifies it. Section 5 shows experimental results. Finally, Section 6 concludes the paper.

2 RELATED WORK

2.1 Hierarchical Federation by Proxy

Figure 1 shows the proxy-based interoperation of federations. A proxy consists of two agents and each agent joins a federation. Each agent interacts with RTI to represent the behavior of the other federation. In Figure 1, Agent$_G$ is the acquisition agent and Agent$_F$ is the representing agent for federation F. They changes roles for federation G. Interoperation is accomplished as follows. First, the acquisition agent acquires information about joined federation. It can use query or callback services for acquisition. During the simulation, user federates request services to progress and it causes callbacks to the other federates or state change of the federation. If the agent finds a change of state or receives information through a callback, it delivers the information to the representing agent. Representing agent translates the information and requests the appropriate service to represent behavior of the other federation. Then, RTI reacts to the request and the behavior of the other federation is represented in the other federation.


② Deliver the information
Agent$_G$
Agent$_F$
① Acquire information of federation F
Proxy
③ Request service to represent behavior of Federation F
RTI-F
RTI-G
f$_1$
…
f$_m$
g$_1$
…
g$_n$
Figure 1: Proxy-based interoperation of federations

To interoperate three or more federations, a hierarchical federation can be constructed using several proxies. Figure 2 shows an example of hierarchical federation. In each federation, a proxy represents all federates should be represented in a federation. Due to system hierarchy, proxy may acquire the federation information from the other proxy.

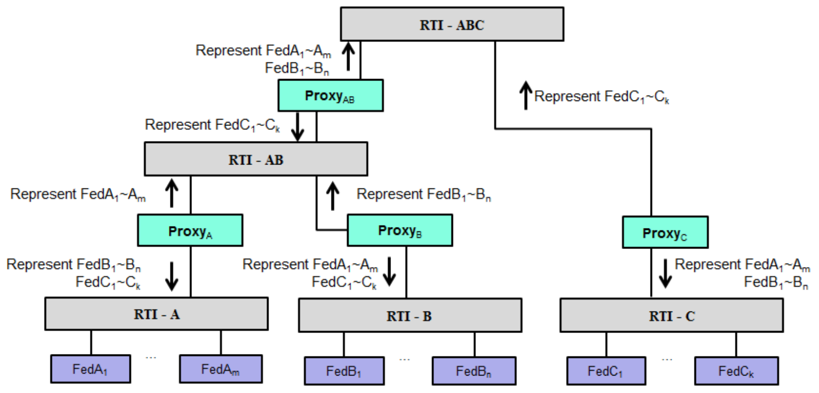


Figure 2: Hierarchical Federation

2.2 Time Synchronization in Hierarchical Federation

In HLA, the logical times of federates are synchronized by the TM service. RTI receives the time advance requests of federates and determines the time advance grants of each federates. Each federate has permission time and other federates are affected by permission time of the other federates. Federates can advance a time lower than GALT(Greatest Available Time) which is the smallest value among permission times and RTI grants a time advance when federates satisfy that condition.

To represent the time behavior, permission times are delivered to the other federation. An agent need not know the times of all the federates but should know the permission time which affects to advance of the other federates. The agent can use LITS (same with MinNextEventTime in HLA 1.3) which means earliest possible receiving TSO event time. The LITS of agent means that it will not receive messages earlier than the LITS. The agent requests TARA(Time Advance Request Available) service to LITS of the other federation. Then it means that agent guarantees that it will not generate TSO events earlier than LITS. RTI determines the time advance grant based on each federate's permission time. This algorithm generally works correctly. However, this algorithm may cause the deadlock problem as described in Section 4.

3 TIME STATE OF AGENT IN THE INTEROPERATION OF FEDERATIONS

In the proxy-based interoperation of federations, an agent represents the time state of the other federation. There are several time states for federates in HLA and the federates initialize and change them during simulation. Time-regulating and time-constrained means influence on time advance of the other federates. Asynchronous delivery describes the relation of a RO (Receive Order) message and time advance. According to these states of the joined federates, agents should determine the time states.

3.1 Time Regulation

In HLA, only time-regulating federates can affect the time advance of the other federates and send TSO messages. If there are no time-regulating federates in a federation, its representing agent need not regulates the time advance of the other federates. If one of the federates is regulating, federates of the other sides should be regulated by it and representing agent should be regulating. Therefore, the agent should check that there are time-regulating federates in the joined federation. However, there is no service that can check it. Therefore, we stipulate that the agent is always time-regulating. In this case, we consider the case in which there are no time-regulating federates. If there are no time-regulating federates, the LITS of the acquisition agent is not defined and queried LITS value is infinity. Then, representing agent requests time advance to infinity, which means that it cannot regulate the other federates.

Regulating federates has lookahead, which refers to a period that it will not generate TSO messages. Initially, a federate cannot have a negative current time and guarantees that it will not generate TSO messages lower than (current time + lookahead). The agent should guarantees that it will not generate TSO message lower than smallest value of (current time + lookahead). Therefore, the lookahead of the agent should be smaller than minimum lookahead of federates. To consider all cases including zero-lookahead, the lookahead of the agent should be set to zero.

3.2 Time Constrained

The time advance of time-constrained federates are affected by time-regulating federates and only time constrained federates can receive TSO messages in time stamp order. Because the agent does not perform a simulation, a time-constrained agent means that it will receive TSO messages in time stamp order. If there are even one time-constrained federates, the agent should be time-constrained to deliver TSO message to it. However, there is no service that can check it, like the time regulation. Therefore, we stipulates that agents are always time constrained and consider the case in which there are no time-constrained federates. Because non time-constrained federates receives all messages in receive order, they are not affected by message types sent from agents.

3.3 Asynchronous Delivery

In HLA, each federates can switch asynchronous delivery state. If switch is enabled, federates can receive messages, including RO messages, only in the time advancing state. Asynchronous delivery is not affected by sending federates and determined by receiving federates.

In the interoperation of federations, the agent always enables asynchronous delivery. Because asynchronous delivery is determined by the receiving federate, the agent should always deliver RO messages to the other federation regardless of its time advance. RTI, which includes receiving federates, will determine whether the message is delivered to them according to their states. If the agent disables asynchronous delivery, RO messages cannot be delivered to the other federation when the agent is time advancing state and it may causes different behavior with respect to a federation

4 TIME SYNCHRONIZATION IN HIERARCHICAL FEDERATION

In this section, we propose a time synchronization algorithm in a hierarchical federation. The proposed algorithm is based on LITS like previous research studies. First, we describes the problem of LITS-based algorithm and find a solution. Then, we propose modified algorithm using the solution. The agent model is constructed based on the algorithm and we verify that it works correctly in hierarchical federation.

4.1 Problem

In a LITS-based algorithm, agent uses “query LITS” to acquire permission time and the representing agent requests “TARA” to queried LITS. The agent requests “TARA” only when LITS is changed. A problem can be occurred when receiving TSO message time and the grant time of agent are equal. In this case, messages may remain in the queue of agents and they may not delivered. Figure 3 describes the problem. In a federation, a message is delivered directly to the receiver’s queue. On the other hand, the message is delivered to agent’s queue first and to the other federation later. Due to this progress, there may be some problems which cause deadlock.

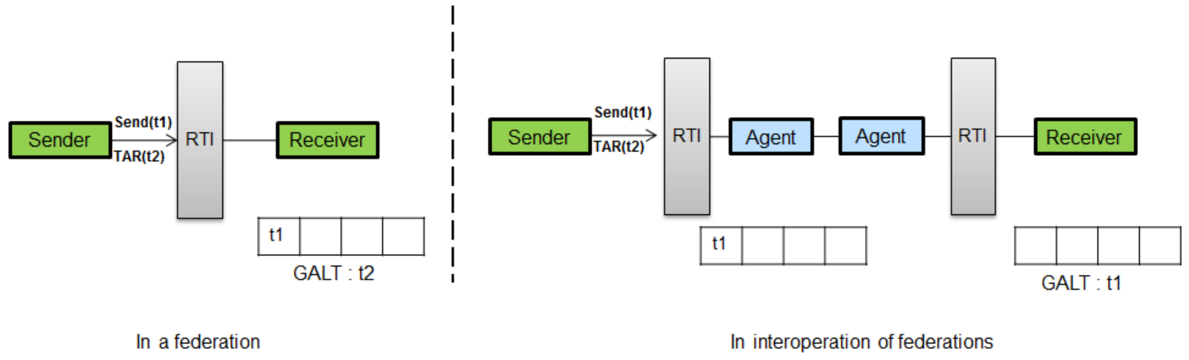


Figure 3: Message delivery problem in proxy-based interoperation of federations

Figure 4 shows detailed example. Initially, two federates requests TARA to 1, two agents deliver the requests, and the federates receive TAG(Time Advance Grant). Then, Fed1 send a message of time 1 and request TARA to 2. Fed2 requests NMRA(Next Message Request Available) to 2. If Fed1 and Fed2 are in a federation, Fed2 will receive TAG to 1 with message. In the interoperation of federation, agent queries LITS periodically and compare it to previous value. If LITS is changed, it delivers changed values to other agent and representing agent request TARA to LITS. However, LITS is not changed due to queued message in this case. Because the LITS is not changed, the agents do not request the time advance and simulation will not progress.

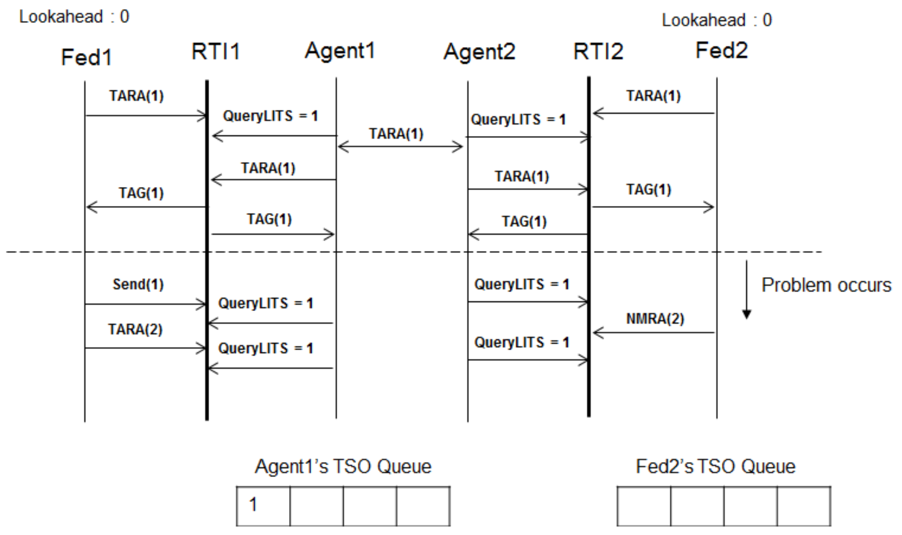


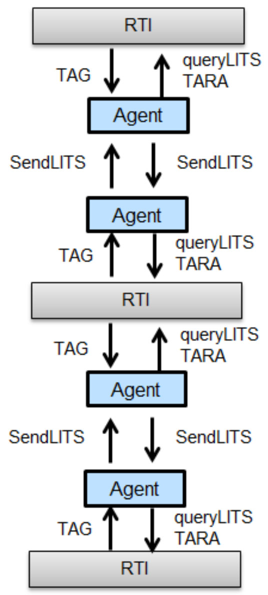
Figure 4: Example of message delivery problem

4.2 Proposed Algorithm

The problem occurs when TSO messages remain in agent's queue and it will not be delivered. To solve the problem, agent should receive queued messages and deliver it to the other agent. In addition, the representing federates should not modify their permission time. Therefore, we let the agent request TARA to current time(the last grant time) to receive queued message. Time advance request to current time does not change permission time and it will be grant simultaneously because it is already granted time. When it receives time advance grant, it receives TSO messages of the current time. Because a new TSO message may be sent to agent after grant, this process should be repeated until the LITS is increased.

Figure 5 shows the proposed time synchronization algorithm for hierarchical federation. All agents works independently and have the same algorithm regardless of its position. Time advance will be propagated through several agents in hierarchical federation. The algorithm is described as follows.

- Agent query LITS periodically until simulation is end. If queried LITS is changed, the agent sends it to the other agent and saves it. The the agent also checks its time advancing states. If it is in time granted state, it requests TARA to current time to receive queued messages.
- When the agent receives new LITS from the other, it request TARA to new LITS. If it is in time advancing state, it reserves the time requests.
- When the agent grants time advance, it sets the time grant flag and checks reserved time advance request. If there is a reserved request, the agent requests time advance to represent the other federation.
- When an agent requests time advance, agent of the same federation can recognize it and the request will be propagated to other federations.



```
Do
    LITS = queryLITS()
    If(prevLITS != LITS)
        SendLITS(LITS);
        prevLITS = LITS;
    End
    If(TimeGrant = true)
        TARA(currentTime);
    End
While(Simulation is end)

When receive SendLITS(t)
    If(time advance grant = true)
        TARA(t);
    Else
        NextRequestTime = t;
    End
End

When receive TAG(t)
    Time advance grant = false;
    If (NextRequstTime != -1)
        TARA(NextRequestTime);
        NextRequestTime = -1;
    End
End
```

Figure 5: Proposed algorithm for the agent in a hierarchical federation

4.2.1 Time Model of Agent

The time model of the agent are constructed based on proposed algorithm. Model is described by extended finite state machine. In this model, [] means condition, ? is input, ! is output and statements after "/" are operations.

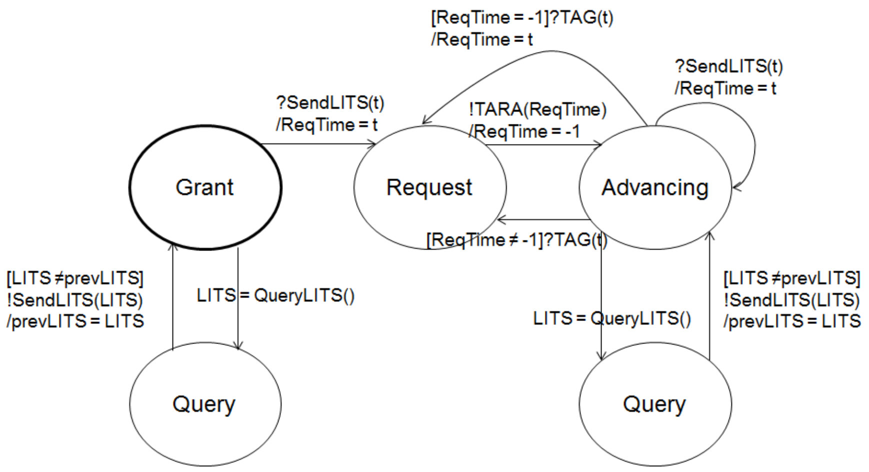


Figure 6: Time Model of Agent

4.2.2 Correctness of the proposed algorithm

In order to prove that the proposed algorithm works correctly, we define following notations. $GALT_i$ is GALT of federate i. $PermissionTime_i$ is permission time of federate i. It will be (current time + lookahead) for NMR and (request time + lookahead) for TAR. $GlobalMin$ is the minimum value for all the federations, $GlobalMin_{second}$ is the second minimum value and $GlobalMin_{Third}$ is the third minimum values. $LocalMin^A$ is the minimum value in the federation A and $LocalMin^A_{Second}$ is the second minimum value in the federation A. $Min(A, B)$ is smaller value for A and B.

The proposed algorithm is verified based on following assumptions.

Assumption 1. GALT is calculated by RTI and has following values

- $GALT_i$ = LocalMin(PermissionTime) for all i≠p (federate p has minimum permission time)
- $GALT_p$ = $LocalMin_{Second}$ (PermissionTime)

Assumption 2. LITS is calculated by RTI and LITS has following values

- LITS = Min(GALT, the minimum time of the queued messages)

In HLA, the time advance of a federate is determined by its GALT. If all the GALTs of hierarchical federation are same with those of a federation, time synchronization works correctly.

Theorem 1 *Each GALTs of hierarchical federation are same with those of a federation.*

Proof.
We show the correctness of the algorithm by mathematical induction.
Basis)
Initially, logical times of federates is zero and they have a non-negative lookahead. Let federate p has minimum lookahead and federate q has second minimum lookahead. Then, GALTs of federates have following values in a federation

$$GALT_i = GlobalMin(PermissionTime) = \text{Lookahead}_p \text{ for } i \neq p$$
$$GALT_p = GlobalMin_{second}(PermissionTime) = \text{Lookahead}_q$$

In the interoperation of federations, we assumes that federate p joins federation A and the other federation is federation B. Because there are no messages initially, the LITS of the agent is same with the GALT. Therefore, the agent of Federation A requests to $LocalMin^B(PermissionTime)$and the agent of Federation B requests to $LocalMin^A(PermissionTime)$. $LocalMin^A(PermissionTime)$ is Lookahead_p. $LocalMin^B(PermissionTime)$ is Lookahead_q if federate q is in federation B and is larger than Lookahead_q if federate q is in federation A. Therefore, federation A has following values

$$GALT_i = LocalMin^A(PermissionTime) = Lookahead_p \text{ for } i \neq p$$
$$GALT_p = LocalMin^A_{second}(PerimissionTime) = Lookahead_q$$

Federation B has following values

$$GALT_j = LocalMin^B(PermissionTime) = Lookahead_p \text{ for all } j$$

Therefore, we can see that all GALTs are same with those of a federation.

Inductive Assumption)
Each federate requests a time advance after initialization and that causes permission time of the federate. We assume that the GALTs of hierarchical federation is same with those of a federation after *n*th request. Let federate p has minimum lookahead, federate q has second minimum lookahead and federate r has third lookahead. The GALTs of a federation have following values.

$$GALT_i = GlobalMin(PermissionTime) = PermissionTime_p \text{ for } i \neq p$$
$$GALT_p = GlobalMin_{second}(PermissionTime) = PermissionTime_q$$

In the interoperation of federations, let federate p joins federation A and the other federation is federation B. The GALTs of the federation A have following values.

$$GALT_i = LocalMin^A(PermissionTime) = PermissionTime_p \text{ for } i \neq p$$
$$GALT_p = LocalMin^A_{second}(PerimssionTime)$$

The GALTs of the federation B have following values.

$$\text{GALT}_j = LocalMin^B(PermissionTime)$$

We assumed the GALTs are same with a federation, $LocalMin^A_{second}(PerimssionTime)$should be $PermissionTime_q$ and $LocalMin^B(PermissionTime)$ should be $PermissionTime_q$. Therefore, permission time of the agents are as follows.

$$PerimssionTime_{AgentA} \begin{cases} = PermissionTime_q, & \text{if federate q is in federation A} \\ \geq \ PermissionTime_q, & \text{if federate q is in federation B} \end{cases} \tag{1}$$

$$PermissionTime_{AgentB} = PermissionTime_p \tag{2}$$

Inductive Step)

We will shows that two values are same after *n+1*th request. We assume that federate k changes its permission time in *n+1*th step. We divides the cases according to requesting federates and its permission time.

(Case1~3 assumes that queued messages for which have a timestamp bigger than new permission time.)

Case 1. Federate k had minimum permission time in *n*th request. (k=p)

Case 1-1. Permission time of federate k is minimum value

In a federation, the GALTs has following values.

$$GALT_i = GlobalMin(PermissionTime) = PermissionTime_k \text{ for i} \neq \text{k}$$
$$GALT_k = GlobalMin_{second}(PermissionTime) = PermissionTime_q$$

Federate k still has minimum permission time but changes the value.

In the interoperation of federations, the agent of federation B requests TARA(changes its permission time) to LocalMin$^{A}(PermissionTime)$ because LITS of the federation A is changed. The GALTs have following values in federation A.

$$GALT_i = LocalMin^A(PermissionTime) = PermissionTime_k \text{ for i} \neq \text{k}$$
$$GALT_k = LocalMin^A_{second}(PermissionTime) = PermissionTime_q \quad \text{- from (1)}$$

The GALTs have following values in federation B.

$$GALT_j = LocalMin^B(PermissionTime) \text{ for all j}$$

Let's assume that there are the GALTs which have different in a federation. Then, the following equation is invalid for federation B

$$GALT_j = LocalMin^B(PermissionTime) = PermissionTime_k$$

However, the equation is valid because the permission time of the agent is LocalMin$^{A}(PermissionTime) = PermissionTime_k$ (from Assumption 1). Therefore, we shows GALTs are same with those of one federation in the case 1-1 by proof by contradiction.

Case 1-2. Permission time of federate k is the second minimum value

In this case, previous second minimum values becomes minimum value. In a federation, the GALTs has following values.

$$GALT_i = GlobalMin(PermissionTime) = PermissionTime_q \text{ for i} \neq \text{q}$$
$$GALT_q = GlobalMin_{second}(PermissionTime) = PermissionTime_k$$

In the interoperation of federations, The agent of federation B requests TARA to LocalMin$^{A}(PermissionTime)$ because LITS of the federation A is changed. The GALTs have following values in federation A.

$$GALT_i = LocalMin^A(PermissionTime) = PermissionTime_q \text{ for i} \neq \text{q} \quad \text{- from (1)}$$

The GALTs have following values in federation B.

$$GALT_i = LocalMin^B(PermissionTime)$$

The GALT of federate q, which have minimum permission time, have following value.

$$GALT_q = \begin{cases} LocalMin^A_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ A \\ LocalMin^B_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ B \end{cases}$$

Let's assume that there are the GALTs which have different in a federation. Then, at least one of the following equation is invalid.

$$GALT_i = LocalMin^B(PermissionTime) = PermissionTime_q \text{ for federation B}$$
$$GALT_q = PermissionTime_k$$
$$= \begin{cases} LocalMin^A_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ A \\ LocalMin^B_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ B \end{cases}$$

The first equation is valid due to Assumption 1. The second equation is valid due to (1) and the assumption that federate k has second minimum permission time. Therefore, we shows GALTs are same with those of one federation in the case 1-2 by proof by contradiction.

Case 1-3. Permission time of Federates is greater than the second minimum values

In this case, previous second minimum values becomes minimum value and previous third minimum becomes second minimum. In a federation, the GALTs has following values.

$$GALT_i = GlobalMin(PermissionTime) = PermissionTime_q \text{ for } i \neq q$$
$$GALT_r = GlobalMin_{second}(PermissionTime) = PermissionTime_r$$

In the interoperation of federations, The agent of federation B requests TARA to $\text{LocalMin}^A(PermissionTime)$ because LITS of the federation A is changed. The GALTs have following values in federation A.

$$GALT_i = LocalMin^A(PermissionTime) = PermissionTime_q \text{ for } i{\neq}q \text{ - from (1)}$$

The GALTs have following values in federation B.

$$GALT_i = LocalMin^B(PermissionTime)$$

The GALT of federate q, which have minimum permission time, have following value.

$$GALT_q = \begin{cases} LocalMin^A_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ A \\ LocalMin^B_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ B \end{cases}$$

Let's assume that there are the GALTs which have different in a federation. Then, at least one of the following equation is invalid.

$$GALT_i = LocalMin^B(PermissionTime) = PermissionTime_q \text{ for federation B}$$
$$GALT_q = PermissionTime_r$$
$$= \begin{cases} LocalMin^A_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ A \\ LocalMin^B_{second}(PermissionTime), & if\ federate\ q\ is\ in\ federation\ B \end{cases}$$

The first equation is valid due to Assumption 1. The second equation is valid due to the assumption that federate r has second minimum permission time. Therefore, we shows GALTs are same with those of one federation in the case 1-3 by proof by contradiction.

Case 2. Federate k had second minimum permission time in *n*th request. (k=q)

Case 2-1. Permission time of Federates is second minimum value

In this case, previous minimum values is still minimum value. In a federation, the GALTs has following values.

$$GALT_i = GlobalMin(PermissionTime) = PermissionTime_p\ for\ i \neq p$$
$$GALT_p = GlobalMin_{second}(PermissionTime) = PermissionTime_k$$

In the interoperation of federations, we should consider two cases. If federate k is in federation A, the agents does not request TARA because LITS of the federation A is still PermissionTime_p. If federate k is in federation B, the agent of federation A requests TARA(changes its permission time) to $\text{LocalMin}^B(PermissionTime)$ because LITS of the federation B is changed. The GALTs have following values in federation A.

$$GALT_i = LocalMin^A(PermissionTime) = PermissionTime_p\ for\ i \neq k$$
$$GALT_p = LocalMin^A_{second}(PermissionTime)$$

The GALTs have following values in federation B.

$$GALT_j = LocalMin^B(PermissionTime) \text{ for all j}$$

Let's assume that there are different GALTs which have different in a federation. Then, at least one of the following equation is invalid.

$$GALT_p = LocalMin^{A}_{second}(PermissionTime) = PermissionTime_k$$
$$GALT_j = LocalMin^{B}(PermissionTime) = PermissionTime_p \text{ for federation B}$$

When federate k is in federation A, the first equation is valid due to assumption that federate k has second minimum permission time and the second equation is valid due to (2). When federate k is in federation B, the first equation is valid due to Assumption 1 and the assumption that federate k has second minimum permission time and the second equation is valid due to (2). Therefore, we shows GALTs are same with those of one federation in the case 2-1 by proof by contradiction.

Case 2-2. Permission time of Federates is greater than the second minimum values

In this case, previous second minimum values becomes minimum value and previous third minimum becomes second minimum. In a federation, the GALTs has following values.

$$GALT_i = GlobalMin(PermissionTime) = PermissionTime_p \text{ for } i \neq p$$
$$GALT_p = GlobalMin_{second}(PermissionTime) = PermissionTime_r$$

In the interoperation of federations, we should consider two cases. If federate k is in federation A, this case works same with case 1-3. Therefore, we will consider the case in which federate k is in federation B. The agent of federation A requests TARA(changes its permission time) to LocalMin^{B}($PermissionTime$) because LITS of the federation B is changed. The GALTs have following values in federation A.

$$GALT_i = LocalMin^{A}(PermissionTime) = PermissionTime_p \text{ for } i \neq p$$
$$GALT_p = Local^{A}_{second}(PermissionTime)$$

The GALTs have following values in federation B.

$$GALT_i = LocalMin^{B}(PermissionTime)$$

Let's assume that there are the GALTs which have different in a federation. Then, at least one of the following equation is invalid.

$$GALT_p = Local^{A}_{second}(PermissionTime) = PermissionTime_r$$
$$GALT_j = LocalMin^{B}(PermissionTime) = PermissionTime_p \text{ for federation B}$$

The first equation is valid due to Assumption 1 and the assumption that federate r has second minimum permission time. The second equation is valid due to (2) Therefore, we shows GALTs are same with those of one federation in the case 2-2 by proof by contradiction.

Case 3. Federate k had the permission time greater than in *n*th step.

In this case, minimum and second minimum values of permission time is not unchanged. Results are same with *n*th stop. The agent works when federate k was local minimum permission time. However, time advance request of the agents are bigger than PermissionTime$_p$ and PermissionTime$_q$ and does not affect to GALTs.

Case 4. Agents have TSO messages which has smaller time stamp than new permission time

Let the time of the messages is t_m. Then, agent will request time advance to t_m first. Because t_m is smaller than GALT of the agents, it will be delivered later. Because the agent requests time advance until receiving all the messages and t_m is always smaller than GALT, it can receive the messages with time advance grant. After delivery, the LITS of the agents becomes same value with the GALT and this case becomes Case1~3 according to new permission time.

5 EXPERIMENT

To identify that the time management algorithm works correctly in actual environment, experiments are performed in two environment – in one federation and in a hierarchical federation. First, all the federates join a federation and perform a simulation. Second, federates are arranged to several federations they are interoperated in hierarchical federation. We make two hierarchical federation with different architecture.

In hierarchical federation A, Fed1~Fed3 is in federation A and Fed4~Fed6 are in federation B. In hierarchical federation B, Fed1~Fed4 are in federation A and Fed5~Fed6 are in federation B. If time management works correctly in hierarchical federation, the three results should be same, and there will be no deadlocks in hierarchical federation. pRTI1516e is used for the experiments.

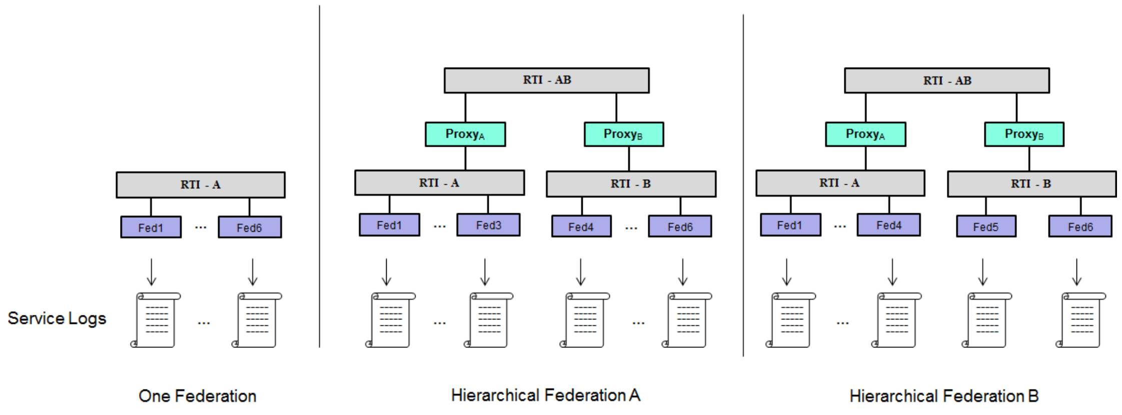
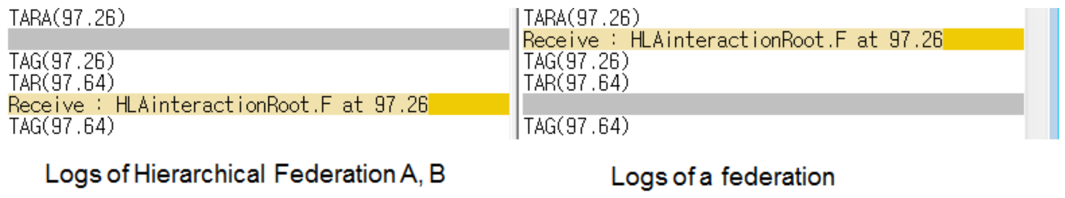


Figure 7: Experiment Environment

Each federate works as follows. All the federates are time-regulating and time-constrained. The federate's lookahead is set to 0.1 and requests time advance 1000 times. The federate decides time advance request service – TAR, TARA, NMR and NMRA – at random in each cycle. It may send an interaction or update an object with a time stamp. All the experiments uses same random seed.

Every federate writes a service log which includes request time, grant time and TSO message time. We compared all the logs in three experiments and we can identify most logs are same. The only different logs are shown in Figure 8. In HLA, receiver requests TARA(NMRA) and is granted, it may receive or may not receive the messages which have a time stamp with granted time and two cases are allowed. Therefore, we can consider that these results are same in HLA.



```
TARA(97.26)

TAG(97.26)
TAR(97.64)
Receive : HLAinteractionRoot.F at 97.26
TAG(97.64)
```

Logs of Hierarchical Federation A, B

```
TARA(97.26)
Receive : HLAinteractionRoot.F at 97.26
TAG(97.26)
TAR(97.64)

TAG(97.64)
```

Logs of a federation

Figure 8: Different result case

6 CONCLUSION

This paper proposed time management in hierarchical federation. We stipulated time states of the agents which are used in hierarchical federation. The agent are stipulated time-regulating and constrained and enables asynchronous delivery. Because we stipulates that the these states are not affected by the federates, it will be applied without change during simulation. A time synchronization algorithm is proposed for hierarchical federation. The proposed algorithm solves the previous problem using additional time advance requests. An agent model is constructed by the proposed algorithm, and algorithm is verified using that model.

Future work is improving performance of time synchronization. Because time advance should be propagated through several federations, time synchronization of hierarchical federation works slower than a federation. Control of querying period or finding appropriate structure may help to improve performance of time synchronization in hierarchical federation

ACKNOWLEDGMENTS

This work was partially supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract. (UD110006MD)

REFERENCES

Chen, D., S. J. Turner, W. Cai, G. K. Theodoropoulos, M. Xiong, and M. Lees. 2010, "Synchronization in Federation Community Networks," *Journal of Parallel and Distributed Computing*, vol. 70, no. 2, 144-159.

Dingel,J., D. Garlan, and C. Damon. 2002. "Bridging the HLA: Problems and Solutions," Sixth IEEE International Workshop on Distributed Simulation and Real Time Applications.

IEEE Std. 1516.1-2010. *IEEE standard for Modeling and Simulation : High Level Architecture - Federate Interface Specification*, IEEE Computer Society.

Myjak, M. D. and S. T. Sharp. 1999. "Implementations of Hierarchical Federations," Proceedings of I999 Fall Simulation Interoperability Workshop, 99F-SIW- 180, Orlando Florida, USA..

Yoo, M.-W., and T. G. Kim. 2009. "Design and Implementations of Surrogates for Interoperation of HLA Federations," *European Simulation Interoperability Workshop*., Istanbul, Turkey, 122 – 130..

AUTHOR BIOGRAPHIES

MIN-WOOK YOO is currently a Ph.D. candidate at the School of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology(KAIST). He received a B.S in Electrical Engineering in 2007 from KAIST and a M.S. in 2009 from KAIST. His research interests include methodology for modeling and simulation of discrete event systems, interoperation of federations for HLA/RTI. His e-mail address is mwyu@smslab.kaist.ac.kr

CHANG BEOM CHOI received his BS in computer engineering from Kyung Hee University and his MS in computer science from KAIST in 2005 and 2007, respectively. He is currently a PhD candidate at the Department of Electrical Engineering, at KAIST. His research interests include discrete event systems modeling/simulation, verification, validation, and accreditation (VV&A) and agent-based simulation. His e-mail address is cbchoi@smslab.kaist.ac.kr

TAG GON KIM is a Professor at the Department of Electrical Engineering Korea Advanced Institute of Science and Technology (KAIST). He was the Editor-In-Chief for Simulation: Transactions for Society or Computer Modeling and Simulation International (SCS). He is a co-author of the text book, Theory of Modeling and Simulation, Academic Press, 2000. He published about 200 papers in M&S theory and practice in international journals and conference proceedings. He is very active in defense modeling and simulation in Korea. His e-mail address is tkim@ee.kaist.ac.kr.