Science in China Series F: Information Sciences

© 2009 SCIENCE IN CHINA PRESS

Springer

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

LoI: Efficient relevance evaluation and filtering for distributed simulation

ZHOU Zhong[1,2†] & WU Wei[1,2]

[1] State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing 100191, China;
[2] School of Computer Science and Engineering, Beihang University, Beijing 100191, China

In large-scale distributed simulation, thousands of objects keep moving and interacting in a virtual environment, which produces a mass of messages. High level architecture (HLA) is the prevailing standard for modeling and simulation. It specifies two publish-subscribe mechanisms for message filtering: class-based and value-based. However, the two mechanisms can only judge whether a message is relevant to a subscriber or not. Lacking of the ability to evaluate the relevance, all relevant messages are delivered with the same priority even when congestion occurs. It significantly limits the scalability and performance of distributed simulation. Aiming to solve the relevance evaluation problem, speed up message filtering, and filter more unnecessary messages, a new relevance evaluation mechanism Layer of Interest (LoI) was proposed by this paper. LoI defines a relevance classifier based on the impact of spatial distance on receiving attributes and attribute values. An adaptive publish-subscribe scheme was built on the basis of LoI. This scheme can abandon most irrelevant messages directly. Run-time infrastructure (RTI) can also apply congestion control by reducing the frequency of sending or receiving object messages based on each objects' LoI. The experiment results verify the efficiency of message filtering and RTI congestion control.

distributed simulation, high level architecture (HLA), relevance evaluation, layer of interest (LoI), run-time infrastructure (RTI)

1 Introduction

In large-scale distributed simulation, thousands of objects keep moving and interacting in a virtual environment. It results in a data explosion because each object is producing messages nearly all the time. The data explosion cripples the simulation performance and restricts its scalability. Interest management is the key technology of large-scale distributed simulation, which provides only the relevant messages to participants by relevance filtering. Under interest management, a simulator expresses its message interests in terms of location or other attributes. Then messages that do not meet the requirements will be filtered. In this way the simulator will only receive relevant messages.

There are mainly three types of interest management according to their expressions: formula, cell and extent[1]. Area of interest (AOI)[2] is a typ-

Received February 28, 2008; accepted November 22, 2008
doi: 10.1007/s11432-009-0058-0
†Corresponding author (email: zz@vrlab.buaa.edu.cn)
Supported by the National Basic Research Program of China (Grant No. 2009CB320805), the National Natural Science Foundation of China (Grant No. 60603084), and the National High-Tech Research & Development Program of China (Grant No. 2006AA01Z331)

ical formula filtering, which delivers messages on the mathematical formula of distance between the sender and the receiver. The management overhead of formula filtering is high because continuous formula evaluation is costly. The space division technique used in NPSNET[3] is cell filtering. It requires little overhead, but the filtering efficiency is low. Extent-based filtering strikes a balance between formula-based and cell-based mechanisms. Researches on interest management technologies had prevailed over the course of 1990s when distributed interactive simulation (DIS) standard[4] dominated. Morse[1] surveyed ten typical systems, and presented taxonomy to classify interest management systems. High level architecture (HLA) has been accepted as the technical standard of modeling and simulation. It prescribes a run-time Infrastructure (RTI) to provide run-time services including interest management[5].

This paper focuses on the interest management of HLA systems. HLA standard specifies two kinds of publish-subscribe mechanisms for simulators to express their interests[6,7], and RTI is responsible for their realization. One is class-based publish-subscribe mechanism which filters messages on their object/interaction classes as well as attributes of object classes. This mechanism is defined in declaration management (DM) and object management (OM) services. The other one is value-based mechanism, which is based on extent filtering. It is defined in data distribution management (DDM) services. Three main terms, routing space (RS), update region (UR) and subscription region (SR), are defined in DDM services in HLA 1.3 standard[7]. RS is an abstract multidimensional rectangle space, and interest management filters messages on their dimensions. Either UR or SR is a set of extents. The sender declares that its object's position in the RS is UR. The receiver declares that its interest of receiving objects' messages in the RS is within SR. Suppose the receiver has subscribed the sender's object class attributes. RTI will deliver messages from the sender to the receiver when the UR of the sender object and the SR of the receiver are overlapped.

Although DDM has been successfully used in a series of large-scale military exercises such as STOW97, JSIMS and MC'02, the data explosion caused much trouble all along. Experiences in US military simulations illustrated that RTI-NG cannot support large-scale distributed simulations without modification[8]. And they also suggested that RTI-s serving in US army had better not to strictly conform to HLA DDM specification[9]. There are three reasons for this problem. First, existing publish-subscribe mechanisms can only judge whether a message is relevant to a subscriber or not after attribute set and region matching. Lacking of the ability to evaluate the relevance degree, all relevant messages are delivered with the same priority even when congestion occurs. That is, no messages can be abandoned to reduce the traffic. Second, HLA regulates only two transportation types, reliable and best-effort. It is insufficient and apt to be illegally used by the developers with this rough classification. In theatre-level simulations that may be disastrous for the reliable messages worsen the traffic heavily. In fact, reliable transportation type even was not used in US large-scale military simulations ever before[10]. Finally, a lot of attribute set matching is required in class-based filtering. Suppose a subscriber has subscribed m attributes of the object class, and a message contains other n attributes. A set matching of time complication $O(m^*n)$ will be executed before the message is ultimately found to be irrelevant to the subscriber. Something may be done to abandon most messages that are irrelevant according to their attribute relevance to optimize the set matching.

Aiming to solve the relevance evaluation problem, speed up message filtering, and filter more unnecessary messages, a new relevance evaluation mechanism Layer of Interest (LoI) is proposed in this paper. LoI defines a relevance classifier based on the impact of spatial distance on receiving attributes and attribute values. An extension to HLA is also provided by LoI. An adaptive publish-subscribe scheme is built on the basis of LoI. It uses LoI to denote the relevance of publishing, subscribing and messages. Most irrelevant messages can be abandoned directly by the sender or receiver on LoI comparison. Lots of attribute set matching could be avoided before precise class-based filtering in this way. RTI can also apply congestion control by reducing the frequency of sending or receiving object messages based on each object's LoI. LoI technique has been implemented in BH RTI, a distributed RTI software. Experiments show that LoI is efficient in message filtering and RTI congestion control. BH RTI has been applied in many military exercises from 2004. In a BH RTI supported exercise, nearly 50000 objects were successfully simulated, interacting in a virtual environment, using 50 PCs from three sites of university campus network.

The rest of the paper is organized as follows: Section 2 discusses related work. Sections 3 and 4 present the LoI and the extension to HLA. Sections 5 and 6 describe an adaptive publish-subscribe scheme based on LoI, together with its algorithm analysis. RTI congestion control model is presented in section 7. Section 8 introduces the LoI implementation in BH RTI. Performance evaluation of the LoI is presented in section 9, and we conclude the work in section 10.

2 Related work

Some efforts have been made to alleviate the relevance evaluation problem in distributed simulation systems. In general, two kinds of approaches are proposed oriented to the filtering types: class/attribute based and value-based.

In class/attribute based approaches, an object class or its attribute is associated with some parameters to specify the relevance of publisher-subscriber. Studies in refs. [11, 12] use some QoS requirements as the parameters, and then RTI or underling network can treat messages on their parameters. McLean implemented a HRT (Hard Real-time) RTI for better real-time performance. As a conclusion, he put forth that a better mechanism would be allowing RTI to determine a range of QoS requirements based upon publisher-subscriber pairs[11]. Zhao and Georganas[12] provided an extension to associate attributes with some priorities. The priorities are given based on the application level, operating system level and network infrastructure level QoS parameters, involving average unit rate, maximum unit rate, burst duration, maximum delay, loss ratio, priority level, security level, etc. They also suggested implementing RTI to invoke underlying QoS mechanisms based on the priorities.

The value-based approaches mainly classify the message relevance based on the distances between entities. SANDS uses active network technology to install content-based filters in intermediate active routers of IP multicast distribution tree[13]. This approach improves delivery efficiency by pruning messages from those portions of the multicast tree where they are not needed. Frequency filters are set up to control the receiving update frequency based on the entities' distances[14]. Cai et al.[15] proposed a multi-level threshold scheme for dead reckoning based on the concepts of area of interest and sensitive region. The levels of threshold are adaptively adjusted based on the relative distance between entities during the simulation. They also presented two adaptive algorithms for dead reckoning using the threshold adjustment and extrapolation equation selection. Zhou et al.[16] proposed a utility model to evaluate the importance of a simulation entity. The importance is determined by two factors: the number of entities that this entity may have influence on, and the distances between this entity and entities within its area of influence. The AOIs of different types of entities are divided into different number of levels by the distances. Based on the utility model, they devised some flexible update mechanisms of controlling the sending frequency to use the bandwidth more efficiently.

Our work is composing the two types of approaches. There are three main differences between our method and existing research. First, we conduct researches on the relevance among the publisher, the subscriber and messages, whereas previous work focused only on the relevance between the publisher and the subscriber. In fact, messages containing different attribute updates have different relevance to a subscriber. For example, there are two messages. One message contains none of the attributes that a subscriber is interested in. The other one contains some. It is obvious that they should be treated differently, although they may come from the same sender. We arrive at two important deductions revealing the relationship among LoIs of the three subjects in section 5. Second, LoI is good at supporting the publish-subscribe mechanisms of HLA. Existing research can hardly be applied in HLA systems with small modifications, especially those of the multi-level interest management. HLA extension of LoI requires small extensions to HLA and maintains federate code compatibility. We integrate HLA's class-based and value-based publish-subscribe mechanisms into an adaptive LoI-based one, which makes the transportation clearer and easier to use. Third, LoI-based RTI congestion control is more practical. The congestion control is decentralized in all the senders and receivers to cut down the total traffic by controlling the update/reflect frequency of each object on its LoI. The thought of layered priority, proposed in Spring Simulation Interoperability Workshop in 2003 by us, was actually the sparkling of LoI. The systematic method in this paper was formed after theory advancement, implementation and testing in the past four years.

3 LoI (layer of interest)

HLA standard does not take the differences of relevance into account. Existing publish-subscribe mechanisms only care about the relevance of publishers and subscribers. However, from related research about relevance evaluation, we can find that spatial distance has some influence on receiving object attributes and attribute values. Firstly, a sender sends different messages to the receivers on their distances, in frequency and content. For example, you find a person far away. But you can see his raising hand only when the distance is close enough. So messages about his hands are unnecessary to send to you. Generally speaking, receivers far away from the sender object get less frequent messages, and less content. That is to say, values of less attributes will be send to those receivers less frequently. Secondly, the receivers usually have only several types of interests in the sender object. Considering an n-attribute object, the possible subscription types are 2^n, the attribute combinations. Actually, the subscriptions usually have only several types. The reason is that there is always some relationship between the attribute subscriptions. If a subscriber subscribes one attribute X, he will subscribe attribute Y almost definitely.

In view of the features above, LoI was proposed to classify the relevance. It is based on some distinct layered varieties of the receiver's interest in objects in the abstract dimensions. The relevance is divided into six layers on the receiver interest in objects: NO_LAYER, LAYER_CRITICAL, LAYER_VISION, LAYER_ABOUT, LAYER_COMPONENT and LAYER_INSIDE. LoI represents the relevance of object class attributes and attribute values.

NO_LAYER. When UR of object A has no overlap with SR of subscriber B (Figure 1(a)), B will not receive messages of A. LoI define the relevance as NO_LAYER. B is not interested in A, and he would not receive any messages of A.

LAYER_CRITICAL. HLA standard defines reliable transportation type for important messages. These messages should not be dropped in transportation so long as the subscriber is interested in the object. We treat the relevance as LAYER_CRITICAL. Heart beaten messages that occur only once or once in a long time can also be treated as this special layer.

LAYER_VISION represents the relevance of lowest interest. It depicts the broad interest of receiving object's messages in the RS, similar to the scope of visual reach. It has the same subscription region as traditional AOI or SR in DDM. When A entered SR (Figure 1(b)), the LoI reaches LAYER_VISION.

LAYER_ABOUT is the relevance of regular interest. It is based on the fact that one can recognize his surroundings clearly. D_SR is the subscription region for LAYER_ABOUT, illustrated in Figure 1(c). It uses a variable EXTEND to define the connotative region of LAYER_ABOUT. The precondition is that the center of SR is where the subscriber lies. We prescribe EXTENT as the range multiple of LAYER_VISION over LAYER_

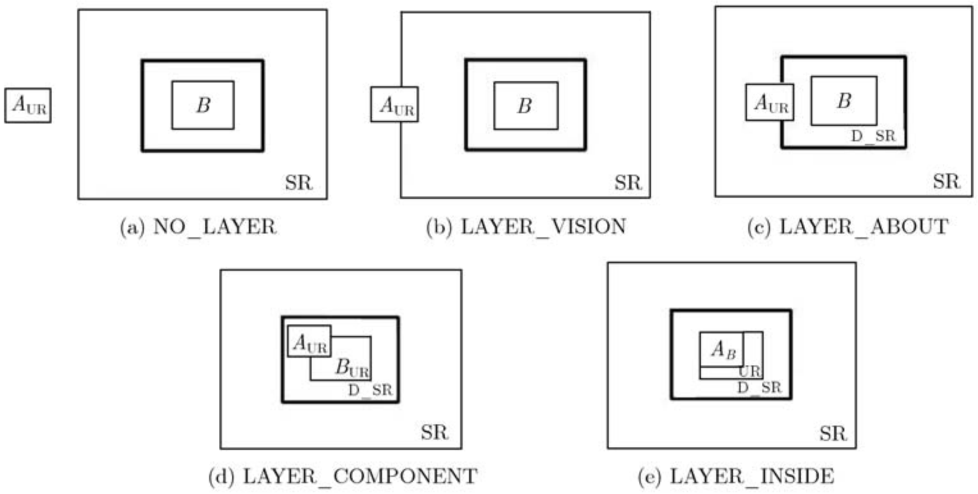


Figure 1 LoI illustration.

ABOUT. The default value is 1, its lower bound. When LAYER_VISION has the range $[b_{\text{lower}}, b_{\text{upper}})$ in a dimension, the corresponding range of LAYER_ABOUT will be

$$\begin{aligned}[a_{\text{lower}}, a_{\text{upper}}) = {} & [(b_{\text{upper}} + b_{\text{lower}})/2 - (b_{\text{upper}} \\ & - b_{\text{lower}})/2/\text{EXTEND}, \\ & (b_{\text{upper}} + b_{\text{lower}})/2 + (b_{\text{upper}} \\ & - b_{\text{lower}})/2/\text{EXTEND}) \\ = {} & [((\text{EXTEND} - 1) \times b_{\text{upper}} \\ & + (\text{EXTEND} + 1) \times b_{\text{lower}}) \\ & /2/\text{EXTEND}, \\ & ((\text{EXTEND} + 1) \times b_{\text{upper}} \\ & + (\text{EXTEND} - 1) \times b_{\text{lower}}) \\ & /2/\text{EXTEND}). \qquad (1)\end{aligned}$$

LAYER_COMPONENT is the relevance of detailed interest. It is suitable for component-level de-aggregation or collision detection. The collision detection impacts much on system performance. The reason is that each simulator has to compute the model intersection between the local object and remote objects. A smaller region for the detection would be better. The subscription region of LAYER_COMPONENT can be approximately looked on as update region (Figure 1(d)). And it is also feasible to define a variable PACE to assist with region overlap computation. No need to compute the region overlaps when PACE is set to zero.

$$[c_{\text{lower}}, c_{\text{upper}}) = [(b_{\text{upper}} + b_{\text{lower}})/2 - \text{PACE}, \\ (b_{\text{upper}} + b_{\text{lower}})/2 + \text{PACE}). \quad (2)$$

LAYER_INSIDE is special for it is the relevance of zero-distance interest. When an object resides right in UR of subscriber (Figure 1(e)), special treatment is required. This zero-distance interest is beyond the ordinary means where value-based filtering takes effect. This inside layer is designed for some special situation such as scene switch or extremely large objects.

The LoI classification is described above on the interest variation with distances. The layers no lower than LAYER_VISION stand for some relevance in best-effort transportation type. These layers have an important characteristic. Relevance in a layer embraces all the relevance in lower layers. Meanwhile, the subscription region of a layer, other than NO_LAYER and LAYER_CRITICAL, covers the regions of higher layers. As most interest management techniques, geography coordinate, which has attribute layer characteristics, is a typical dimension set for LoI.

4 HLA extension of LoI

LoI can be easily used to extend HLA standard for its basic concepts are accordant with HLA. The principles of HLA extension should be compliant with standard HLA based systems, and minimizing the modifications. We have also noted that the region related concepts have changed from HLA 1.3 to IEEE 1516. An attribute can be associated with a dimension set in IEEE 1516, but only can be with fixed routing space in older HLA 1.3. It is convenience to associate an attribute with a dimension in its LoI variable in IEEE 1516. When it comes to HLA 1.3, the attributes have to compromise on LoI variables if they want different EXTEND or PACE value in a dimension. The extension introduced in this section is based on HLA 1.3.

The HLA extension of LoI only involves HLA OMT, no modification to HLA framework and rules or interface specification. The routing space table adds two new columns EXTEND and PACE. And the attribute table adds column LoI. The three columns are optional in extended OMT. If an attribute is associated with none LoI, the LAYER_ABOUT is associated by default. So the extended HLA maintains backward compatibility with standard HLA based systems. Examples of extended routing space table and attribute table are as Tables 1 and 2. Only one routing space can be associated with LoI simultaneously in one federation object model (FOM).

Correspondingly, FED (Federation Execution Data) file should be extended. We add the following five productions to FED DIF (Data Exchange Format) BNF (Backus-Naur Form):

1) ⟨⟨LoI⟩⟩ ::= ⟨NameString⟩;
2) ⟨⟨ExtendValue⟩⟩ ::= ⟨Float⟩;
3) ⟨⟨PaceValue⟩⟩ ::= ⟨Float⟩;
4) ⟨Extend⟩ ::=“(Extend” ⟨⟨ExtendValue⟩⟩ “)”;
5) ⟨Pace⟩ ::= “(Pace” ⟨⟨PaceValue⟩⟩ “)”.

And the following two BNF productions are extended:

1) ⟨Dimension⟩ ::=“(dimension” ⟨⟨DimensionName⟩⟩ “)”;
2) ⟨Attribute⟩ ::=“(attribute” ⟨⟨AttributeName⟩⟩ ⟨⟨Transport⟩⟩ ⟨⟨Order⟩⟩ [⟨⟨SpaceName⟩⟩] “)”.

The extended BNF productions are

1) ⟨Dimension⟩ ::= “(dimension” ⟨⟨DimensionName⟩⟩ [⟨⟨Extend⟩⟩] [⟨⟨Pace⟩⟩] “)”;
2) ⟨Attribute⟩ ::= “(attribute” ⟨⟨AttributeName⟩⟩ ⟨⟨Transport⟩⟩ ⟨⟨Order⟩⟩ [⟨⟨SpaceName⟩⟩] [⟨⟨LoI⟩⟩] “)”.

FED DIF glossary shall add three terms in the extended HLA FED DIF BNF definition.

LoI: The LoI of an object-class attribute.

ExtendValue: The Extend value of a routing-space dimension.

PaceValue: The Pace Value of a routing-space dimension.

Table 1 Example of routing space table with LoI extension

Routing space	Dimension	Dimension type	⋯	EXTEND	PACE
PositionSpace	pos_x	Float	⋯	4	50.0
	pos_y	Float	⋯	4	50.0

Table 2 Example of attribute table with LoI extension

Object	Attribute	Data type	⋯	Routing space	LoI
DveTank	DESTROYED_LEVEL	enum	⋯	PositionSpace	LAYER_CRITICAL
	Position	vector type	⋯	PositionSpace	LAYER_VISION
	Color	enum	⋯	PositionSpace	LAYER_VISION
	Direction	vector type	⋯	PositionSpace	LAYER_ABOUT
	Velocity	vector type	⋯	PositionSpace	LAYER_ABOUT
	Acceleration	vector type	⋯	PositionSpace	LAYER_ABOUT
	Pedrail	vector type	⋯	PositionSpace	LEYER_COMPONENT
	⋯	⋯	⋯	⋯	N/A

5 Adaptive publish-subscribe scheme based on LoI

Early researches only use "object" or "entity" to indicate virtual objects. HLA induces the concepts of "object class" and "object instance". An object class is in fact an object type, and an object instance is an object. The terms "object class" and "object instance" are used in the following parts to be accorded with this.

To speed up message filtering, we should be able to determine the relevance among the publisher, the subscriber and the messages prior to sending messages. An adaptive publish-subscribe scheme would be practical if it can evaluate the message relevance to the publisher and subscriber in the dynamic changing publisher-subscriber environment. This would allow the system to manage delivery requirements across individual links and to optimize network utilization. Current publish-subscribe mechanisms mainly act in object class level. And publish-subscribe in object instance level is also valid in our scheme. If a subscriber wants data of a remote object instance, we call that the subscriber subscribes the object instance.

First of all, the data type of LoI is given in Definition 1.

Definition 1 (LoI Data Type). LoI is an enumerate data type. enum LoI {NO_LAYER=0, LAYER_CRITICAL, LAYER_VISION, LAYER_ABOUT, LAYER_COMPONENT, LAYER_INSIDE}.

The scheme comprises formulas of several LoI variables, thermos and deductions, and some filtering algorithms. The following five LoI variables are used to evaluate the relevance among the three publish-subscribe parts. We arrive at two important deductions on these variables. The two deductions reveal the relationship among the LoIs in publish-subscribe process. Then novel algorithms for sending and receiving messages are proposed based on the deductions.

(1) LoI of publisher over object class;
(2) LoI of local object instance;
(3) LoI of attribute update/reflect message;
(4) LoI of subscriber over object class;
(5) LoI of remote object instance.

5.1 Basic terms

Some basic terms and symbols will be used to indicate object class, object instance, attribute set and region in the context of a publish-subscribe service.

There are several handles including object class handle, attribute handle and object instance handle. The object classes and object instances are identified by unique handles, while an attribute is identified by the owner object class handle and a unique attribute handle. An attribute set refers to a set of attribute handles in the publish-subscribe process of object classes.

Suppose i is an object class handle, o is an object instance handle. Let m, j, k, l be the sizes of attribute sets, representing set size in publishing object class, attribute value update message, subscribing object class and subscribing object instance separately. For example, m-size attribute set means the set has m attribute handle elements.

Let function loi(int i, int hd) returns the LoI value of an attribute handled hd belonging to object class i.

Let $P_m^{(i)}$ be LoI of publisher over object class i with m-size attribute set.

Let $p_m^{(i,o)}$ be LoI of local object instance o of object class i with m-size attribute set.

Let $\eta_j^{(i,o)}$ be LoI of attribute update/reflect message with j-size attribute set of object instance o of object class i.

Let $S_k^{(i)}$ be LoI of subscriber over object class i with k-size attribute set. We define the maximum $S_k^{(i)}$ in all subscribers other than the publisher itself be $S_{\max}^{(i)}$ in the sender side.

Let $s_l^{(i,o)}$ be the LoI of remote object instance o of object class i with l-size attribute set. We define the maximum $s_l^{(i,o)}$ be $s_{l,\max}^{(i,o)}$.

There exist some region-related terms in DDM service. Although some have been narrated before, we give a complete glossary here.

Let the update region be UR and the update region of subscriber be UR′.

Let the subscription region be SR.

Let the region associated with LAYER_ABOUT be D_SR, and that with LAYER_COMPONENT be P_SR. UR′ also denotes the subscription region of LAYER_INSIDE.

When the range of one dimension in SR is $[b_{\mathrm{lower}}, b_{\mathrm{upper}})$, we can get that the corresponding range of D_SR is $[a_{\mathrm{lower}}, a_{\mathrm{upper}})$ and that of P_SR is $[c_{\mathrm{lower}}, c_{\mathrm{upper}})$. The computation methods are referred as formulas (1) and (2).

Figure 2 gives the illustration of attribute sets used in the publish-subscribe service. The attribute sets $\Phi_{i,m}$ and $\Phi_{i,k}$ are used in object class level publish-subscribe service. When DDM is used, the subscriber requests only a subset of $\Phi_{i,k}$ over an object instance. We define the subset as $\Phi_{i,l}$. The attribute set in registering object instance is not required in the scheme.

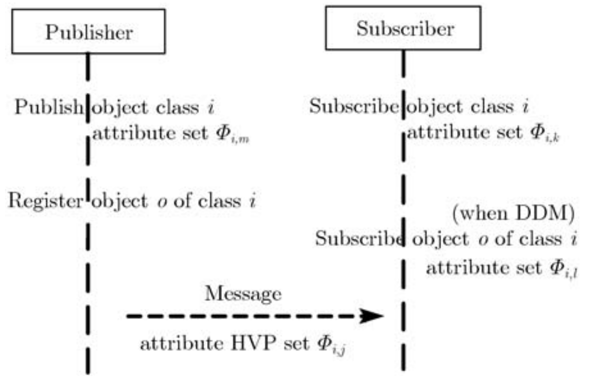

Publisher
Subscriber
Publish object class i attribute set $\Phi_{i,m}$
Subscribe object class i attribute set $\Phi_{i,k}$
Register object o of class i
(when DDM)
Subscribe object o of class i attribute set $\Phi_{i,l}$
Message
attribute HVP set $\Phi_{i,j}$

Figure 2 The attribute sets in publish-subscribe service.

5.2 LoI of publisher over object class

If a publisher publishes object class i with attribute set $\Phi_{i,m}$, we use $P_m^{(i)}$ to denote the detail relevance of publishing the object class.

Definition 2 (LoI of publisher over object class). If a publisher publishes object class i with attribute set $\Phi_{i,m}$, then $P_m^{(i)} = loi(i,h)$, where $h \in \Phi_{i,m}$ and $\forall x \in \Phi_{i,m}$, $loi(i,x) \leqslant loi(i,h)$.

We can obtain from Definition 2:

$P_m^{(i)}$ =LAYER_CRITICAL only when $\forall x \in \Phi_{i,m}$, $layer(i,x)$=LAYER_CRITICAL.

$\Phi_{i,m} = \phi$ means the publisher does not publish object class i, so $P_m^{(i)}$= NO_LAYER in this case.

5.3 LoI of local object instance

When a publisher registered local object instance o with attribute set $\Phi_{i,m}$, it is not determined to send all its messages (attribute updates) to all the subscribers. Messages usually bring value updates of different attributes. The sender had better send a message only to the interested subscribers. Messages with no interested subscribers can be abandoned directly in the sender side. Because DDM service further restricts the scope of receivers, the publishing of local object instance varies when it uses DDM service. The LoI of local object instance denotes the actual detail relevance of publishing object instance o, so it is decided by both $P_m^{(i)}$ and actual subscription LoI $S_k^{(i)}$ or $s_l^{(i,o)}$.

Definition 3 (LoI of local object instance). If a publisher publishes an object instance o of object class i, then

$$p_m^{(i,o)} = \begin{cases} \min(P_m^{(i)}, S_{\max}^{(i)}), o \text{ isn't associated with UR}, \\ \min(P_m^{(i)}, s_{\max}^{(i,o)}), o \text{ is associated with UR}. \end{cases}$$

In Definition 3, whether the DDM service is used in publishing object instance o is drawn by its UR association.

5.4 LoI of attribute update/reflect message

A message of object instance in publish-subscribe environment is often called attribute update (or attribute value update) in sender side and attribute reflect (or attribute value reflect) in receiver side. Attribute update/reflect message is composed of HVP (handle value pair) set where an HVP consists of an attribute handle and value of the attribute. When some attribute values of a local object instance change, the publisher will send its attribute update message. If $\Phi_{i,j}$ is the j-size attribute set of an attribute update, $\Phi_{i,j}$ should be embraced in $\Phi_{i,m}$ for only published attributes can be updated. Then we can get that

$$\Phi_{i,j} \subseteq \Phi_{i,m}. \tag{3}$$

The LoI of attribute update/reflect message denotes the fundamental relevance of messages.

Definition 4 (LoI of attribute update/reflect). If an attribute update message is of object class i and object instance o with attribute set $\Phi_{i,j}$, then $\eta_j^{(i,o)} = loi(i,h)$, where $h \in \Phi_{i,j}$ and $\forall x \in \Phi_{i,j}$, $loi(i,x) \geqslant loi(i,h)$.

Then it can be obtained that

$\eta_j^{(i,o)}$ =LAYER_CRITICAL when $\exists x \in \Phi_{i,j}$, $loi(i,x)$ =LAYER_CRITICAL.

LoI $\eta_j^{(i,o)}$ is calculated by the publisher and will be tagged into the message. When subscribers receive the message, they can pick up the LoI from it.

5.5 LoI of subscriber over object class

If a subscriber subscribes an object class i with attribute set $\Phi_{i,k}$, we use $S_k^{(i)}$ to denote the detail relevance of subscribing object classes. Let $\cup\Phi_{i,k}$ be the aggregated subscription set of $\Phi_{i,k}$ to a specific publisher, and $\Phi_{i,k}$ here should exclude the one of the publisher itself because the publisher will not send message to itself.

Each attribute in a message should have at least one subscriber, so

the subscriber can receive message of attribute set $\Phi_{i,j}$ without DDM if and only if $\Phi_{i,j} \cap \Phi_{i,k} \neq \phi$; (4)

and $\Phi_{i,j} \subseteq \cup\Phi_{i,k}$. (5)

Definition 5 (LoI of subscriber over object class)**.** If a subscriber subscribes object class i with attribute set $\Phi_{i,k}$, then $S_k^{(i)} = loi(i,h)$, where $h \in \Phi_{i,k}$, and $\forall x \in \Phi_{i,k}, loi(i,x) \leqslant loi(i,h)$.

Then the following results are obtained:

$S_k^{(i)}$ =LAYER_CRITICAL only when $\forall x \in \Phi_{i,k}$, $loi(i,x)$= LAYER_CRITICAL;

$\Phi_{i,k} = \phi$ means the subscriber does not subscribe object class i, so $S_k^{(i)}$=NO_LAYER in this case.

$S_{\max}^{(i)}$ denotes the actual subscription of object class i to the publisher. It is defined as follows:

$S_{\max}^{(i)} = \max\{S_k^{(i)}$ of all the subscribers other than the publisher itself$\}$. (6)

We let $S_k^{(i)}(\cup\Phi_{i,k})$ to refer to the $S_k^{(i)}$ with attribute set $\cup\Phi_{i,k}$.

Theorem 1. $S_{\max}^{(i)} = S_k^{(i)}(\cup\Phi_{i,k})$.

Proof. By Definition 5, $S_k^{(i)} = loi(i,h)$ where $h \in \Phi_{i,k}$, and $\forall x \in \Phi_{i,k}, loi(i,x) \leqslant loi(i,h)$, and $\cup\Phi_{i,k}$ is the aggregated set of all the $\Phi_{i,k}$ except the one of the publisher, $h \in \cup\Phi_{i,k}$, so $loi(i,h) \leqslant S_k^{(i)}(\cup\Phi_{i,k})$ and $\exists h' \in \cup\Phi_{i,k}, loi(i,h') = S_k^{(i)}(\cup\Phi_{i,k})$.

Then we obtain $S_k^{(i)} \leqslant S_k^{(i)}(\cup\Phi_{i,k})$ and $\exists S_{k'}^{(i)}, S_{k'}^{(i)} = S_k^{(i)}(\cup\Phi_{i,k})$.

According to the $S_{\max}^{(i)}$ definition (Formula (6)), $S_{\max}^{(i)} = S_k^{(i)}(\cup\Phi_{i,k})$.

5.6 LoI of remote object instance

When a subscriber discovers remote object instance o, it subscribe object instance o with the same attribute set $\Phi_{i,k}$ in subscribing object class i. It is similar to the LoI of local object instance in section 5.3. The publisher's attribute set $\Phi_{i,m}$ of publishing object class i is also unsuitable for the object instance subscription. Let $\Phi_{i,l}$ be the actual j-size attribute set that the subscriber subscribes object instance o with. Then $\Phi_{i,l}$ should be a subset of both of them for the publish-subscribe relationship.

Each attribute of a message should have been published, so

$$\Phi_{i,l} \subseteq \Phi_{i,m}. \tag{7}$$

A subscriber should subscribe an object instance in the scope of subscribing its object class, so

$$\Phi_{i,l} \subseteq \Phi_{i,k}. \tag{8}$$

A subscriber can receive an attribute update with attribute set $\Phi_{i,j}$ only when it has subscribed any attribute in the set, so

receiving attribute update with set $\Phi_{i,j}$

if and only if $\Phi_{i,j} \cap \Phi_{i,l} \neq \phi$. (9)

Let $\cup\Phi_{i,l}$ be the aggregate set of $\Phi_{i,l}$. Here the publisher's $\Phi_{i,l}$ is note excluded because $\Phi_{i,l}$ is the actual subscription attribute set of object instance o. Any attribute update message sent out shall have at least one subscriber, so

$$\Phi_{i,j} \subseteq \cup\Phi_{i,l}. \tag{10}$$

Definition 6 (LoI of remote object instance)**.** If a subscriber subscribes object instance o with attribute set $\Phi_{i,l}$, then $s_l^{(i,o)} = loi(i,h)$, where $h \in \Phi_{i,l}$, and $\forall x \in \Phi_{i,l}, loi(i,x) \leqslant loi(i,h)$.

Theorem 2. $s_l^{(i,o)} \leqslant S_k^{(i)}$.

Proof. By Definition 5, $S_k^{(i)} = loi(i,h)$, where $h \in \Phi_{i,k}$, and $\forall x \in \Phi_{i,k}, loi(i,x) \leqslant loi(i,h)$ and condition in formula (8) $\Phi_{i,l} \subseteq \Phi_{i,k}$, so $\forall x \in \Phi_{i,l}, loi(i,x) \leqslant S_k^{(\mathrm{i})}$.

It can be obtained from Definition 6, $s_l^{(i,o)} = loi(i,h')$, $h' \in \Phi_{i,l}$, then we conclude $s_l^{(i,o)} = loi(i,h') \leqslant S_k^{(i)}$, that is $s_l^{(i,o)} \leqslant S_k^{(i)}$.

$s_{\max}^{(i,o)}$ denotes the actual subscription of object instance o. It is defined as follows:

$$s_{l,\max}^{(i,o)} = \max\{\text{all the } s_l^{(i,o)} \text{ of subscribers}\}. \tag{11}$$

We let $s_l^{(i,o)}(\cup\Phi_{i,l})$ to refer to the $s_l^{(i,o)}$ with attribute set $\cup\Phi_{i,l}$.

Theorem 3. $s_{\max}^{(i,o)} = s_l^{(i,o)}(\cup\Phi_{i,l})$.

Proof. The proof is similar to that of Theorem 1, so here we omit it.

We can see $s_l^{(i,o)} = S_k^{(i)}$ by Definition 6 when object instance o is not associated with a region.

Because the restriction of object instances' attribute set is owing to DDM service, we should get attribute set $\Phi_{i,l}$ based on region overlap. Definition 6 needs another statement. Let symbol $\cap$ be the computation operator of region overlap. Let symbol $\subseteq$ be the logical inclusion expression. Let symbol ϕ be the empty region representing the $\cap$ result of two non-overlap regions. We define the region related symbols according to LoI definition.

Let $R1$ = UR$\cap$SR$\rightarrow \neg\phi$;

Let $R2$ = UR $\cap$ D_SR$\rightarrow \neg\phi$;

Let $R3$ = UR $\cap$ P_SR$\rightarrow \neg\phi$;

Let $R4$ = UR$\subseteq$UR$'$.

Definition 7 (LoI of remote object instance, restatement)**.** If a subscriber subscribes object instance o with region, then

$$s_l^{(i,o)} = \begin{cases} S_k^{(i)}, & \text{UR or SR does not exist,} \\ \text{NO\_LAYER}, & \neg R1 \\ \text{LAYER\_CRITICAL}, & R1 \text{ and } S_k^{(i)} = \text{LAYER\_CRITICAL}, \\ \text{LAYER\_VISION}, & R1 \text{ and } \neg R2 \text{ and } S_k^{(i)} \geqslant \text{LAYER\_VISION}, \\ \text{LAYER\_ABOUT}, & R2 \text{ and } \neg R3 \text{ and } S_k^{(i)} \geqslant \text{LAYER\_ABOUT}, \\ \text{LAYER\_COMPONENT}, & R3 \text{ and } \neg R4 \text{ and } S_k^{(i)} \geqslant \text{LAYER\_COMPONENT}, \\ \text{LAYER\_INSIDE}, & R4 \text{ and } S_k^{(i)} = \text{LAYER\_INSIDE}. \end{cases}$$

We make a simplification here. The region overlap of LAYER_COMPONENT and LAYER_ INSIDE are dependent on specific application requirements, so it will not be taken into account. Then Definition 7 is simplified into the following form.

Definition 8 (LoI of remote object instance, simplified statement)**.** The simplified form of $s_l^{(i,o)}$ is defined as follows:

$$s_l^{(i,o)} = \begin{cases} S_k^{(i)}, & (\text{UR or SR does not exist}) \text{ or } (R2 \text{ and } S_k^{(i)} \\ & \geqslant \text{LAYER\_ABOUT}), \\ \text{NO\_LAYER}, & \neg R1, \\ \text{LAYER\_CRITICAL}, & R1 \text{ and } S_k^{(i)} = \text{LAYER\_CRITICAL}, \\ \text{LAYER\_VISION}, & R1 \text{ and } \neg R2 \text{ and } S_k^{(i)} \geqslant \text{LAYER\_VISION}. \end{cases}$$

5.7 Update/reflect theorems of publish-subscribe LoIs

We obtain the following two important theorems by the definitions and theorems given above.

Theorem 4 (update rule)**.** $p_m^{(i,o)}$ is the least upper bound of $\{\eta_j^{(i,o)}\}$ of the publisher's sending attribute updates.

Proof. If $p_m^{(i,o)}$=NO_LAYER, the object instance will not update. So $\eta_j^{(i,o)}$ does not exist.

If $p_m^{(i,o)} \geqslant$LAYER_CRITICAL, $\eta_j^{(i,o)}$ exists.

Then by Definition 4, $\exists h \in \Phi_{i,j}$, such that $\forall x \in \Phi_{i,j}$, $loi(i,h) \leqslant loi(i,x)$, and by Definition 2, $\exists h' \in \Phi_{i,m}$, such that $\forall x \in \Phi_{i,m}$, $loi(i,x) \leqslant loi(i,h')$.

Observe that $\Phi_{i,j} \subseteq \Phi_{i,m}$ by formula (3), and $\forall x \in \Phi_{i,j}$, $loi(i,x) \leqslant loi(i,h')$.

So $loi(i,h) \leqslant loi(i,h')$, i.e. $\eta_j^{(i,o)} \leqslant P_m^{(i)}$.

For similar reasons, we have

$\eta_j^{(i,o)} \leqslant S_{\max}^{(i)}$ (following formula (5) and Theorem 1),

$\eta_j^{(i,o)} \leqslant s_{\max}^{(i,o)}$ (following formula (10) and Theorem 3).

Then from Definition 3, $\eta_j^{(i,o)} \leqslant p_m^{(i,o)}$.

If there is only one subscriber and $m = k = l = 1$, it is obvious that $loi(i,h') = loi(i,h) = loi(i,x)$ and $\eta_j^{(i,o)} = P_m^{(i)}$.

And by Definition 5, Theorem 1, Definition 6, and Theorem 3, we have $\eta_j^{(i,o)} = S_k^{(i)} = S_{\max}^{(i)}$, $\eta_j^{(i,o)} = s_l^{(i,o)} = s_{\max}^{(i,o)}$.

So $\eta_j^{(i,o)} = p_m^{(i,o)}$ in this case.

Then we arrive at the conclusion that $p_m^{(i,o)}$ is the least upper bound of $\{\eta_j^{(i,o)}\}$.

We can now get a deduction from the Update Rule.

Deduction 1 (update rule, restatement)**.** A publisher can only send attribute updates of LoI

$\eta_j^{(i,o)} \leqslant p_m^{(i,o)}$.

Theorem 5 (reflect rule). $s_l^{(i,o)}$ is the least upper bound of $\{\eta_j^{(i,o)}\}$ of the subscriber's receiving attribute reflects.

Proof. If $s_l^{(i,o)}$=NO_LAYER, the subscriber will not accept reflects of the object instance. So $\eta_j^{(i,o)}$ does not exist.

If $s_l^{(i,o)} \geqslant$ LAYER_CRITICAL, $\eta_j^{(i,o)}$ exists.

Then by Definition 4, $\exists h \in \Phi_{i,j}$ such that $\forall x \in \Phi_{i,j}$, $loi(i,x) \geqslant loi(i,h)$.

By formula (9), receiving attribute reflects of $\Phi_{i,j}$ if and only if $\Phi_{i,j} \cap \Phi_{i,l} \neq \phi$.

Let $\varphi = \Phi_{i,j} \cap \Phi_{i,l}$, then $\varphi \neq \phi$.

$\exists y \in \varphi$ such that $\forall x \in \varphi$, $loi(i,x) \geqslant loi(i,y)$.

Because $\varphi \subseteq \Phi_{i,j}$, $loi(i,y) \geqslant \eta_j^{(i,o)}$.

By Definition 6, $\forall x \in \Phi_{i,l}$, $s_l^{(i,o)} \geqslant loi(i,x)$.

Because $\varphi \subseteq \Phi_{i,l}$, $s_l^{(i,o)} \geqslant loi(i,y)$.

So $s_l^{(i,o)} \geqslant \eta_j^{(i,o)}$.

And we can see that if $j = l = 1, loi(i,y) = s_l^{(i,o)} = \eta_j^{(i,o)}$.

Then we have the conclusion that $s_l^{(i,o)}$is the least upper bound of $\{\eta_j^{(i,o)}\}$.

We can now get a deduction from the reflect rule.

Deduction 2 (reflect rule, restatement). A subscriber can only receive attribute reflects of LoI $\eta_j^{(i,o)} \leqslant s_l^{(i,o)}$.

5.8 Algorithms for sending and receiving messages

The LoIs of the publisher, the subscriber and messages are described above. And HLA's class-based and value-based publish-subscribe mechanisms are unified in it. We drew two important deductions about the LoI relationship among the three parts. It can help simplify the attribute set matching and speed up the message filtering. In the new publish-subscribe environment, the publisher works with $P_m^{(i)}$ of object class i and $p_m^{(i,o)}$ of local object instance o, and a subscriber works with $S_k^{(i)}$ of object class i and $s_l^{(i,o)}$ of remote object instance o. The four LoIs denote the dynamic detail relevance in publish-subscribe sides. Messages will be tagged with the value $\eta_j^{(i,o)}$. LoI $\eta_j^{(i,o)}$ represents the fundamental relevance of messages.

According to Deductions 1 and 2, we can get adaptive algorithms for sending and receiving messages in dynamic publish-subscribe process (Algorithm 1 and Algorithm 2).

Algorithm 1 (Algorithm for sending data) LoI_UAV

```
For each attribute update of local object instance $o$
  int $l = \eta_j^{(i,o)}$, compute according to Definition 4.
  If ($l \leqslant p_m^{(i,o)}$)
    //attach $l$ to the update packet;
    update.loi=$l$;
    multicast the update packet to the subscriber group;
End for
```

Algorithm 2 (Algorithm for receiving data) LoI_RAV

```
For each attribute reflect of remote object instance $o$
  int $l$=reflect.loi; //get $\eta_j^{(i,o)}$
  If($l \leqslant s_l^{(i,o)}$)
    //the reflect packet is wanted by the subscriber
    accept the reflect packet;
    callback the corresponding user function;
End for
```

The sending and receiving algorithms become very simple. This scheme evaluates the relevance of the publisher, the subscriber and messages, and then performs relevance filtering. It speeds up the filtering and guarantees the publish-subscribe requirements. Precise attribute set matching also can be performed as Algorithm 3 in receiver side.

Algorithm 3 (Algorithm for enhanced receiving data) PreciseLoI_RAV

```
For each attribute reflect $\{\mathrm{HVP_j}\{\langle \mathrm{attr_i}, \mathrm{value_i}\rangle\}, \mathrm{loi}\}$
of remote object instance $o$
  int $l$=reflect.loi; //get $\eta_j^{(i,o)}$
  If ($l \leqslant s_l^{(i,o)}$)
    //perform precise matching
    While (int $h = 1; h <= j; h++$)
      If $\mathrm{attr_h} \notin$ subscription attribute set $\{\mathrm{c_i}, \langle \mathrm{attr_i}\rangle\}$
        remove $\langle \mathrm{attr_h}, \mathrm{value_h}\rangle$ from $\mathrm{HVP_j}$ $\{\langle \mathrm{attr_i},$
        $\mathrm{value_i}\rangle\}$;
    If sizeof $\mathrm{HVP_j}$ $\{\langle \mathrm{attr_i}, \mathrm{value_i}\rangle\} == 0$
      break;
    //the reflect packet is wanted by the subscriber
    accept the reflect's valid data $\mathrm{HVP_j}\{\langle \mathrm{attr_i}, \mathrm{value_i}\rangle\}$;
    callback the corresponding user function;
End for
```

6 Algorithm analysis

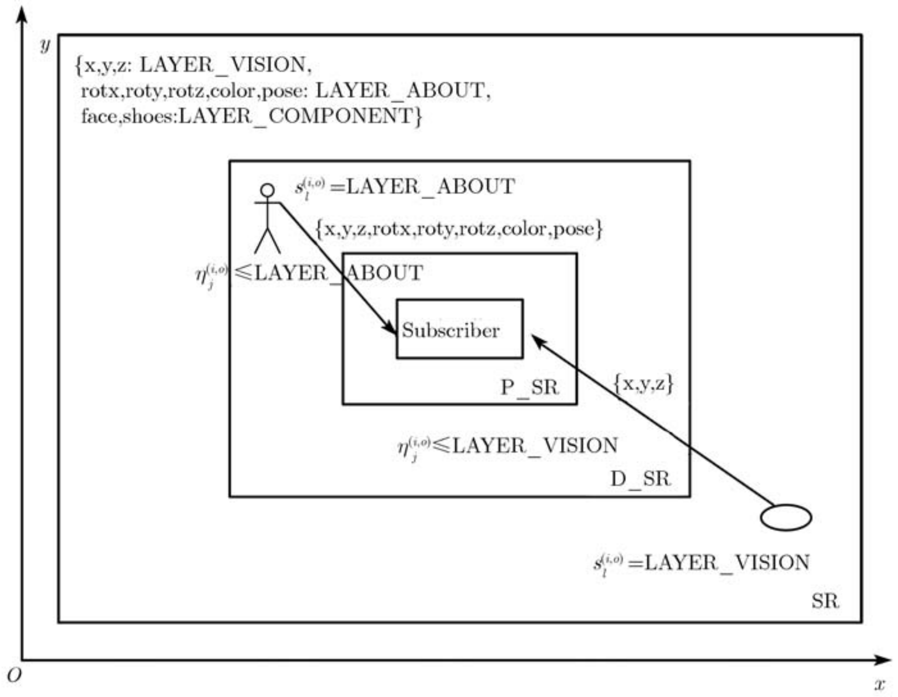
Message filtering computation cost in distributed simulation consists mainly of attribute set matching and value-based interest filtering. Algorithms in the two aspects affect to a great extent the message delivery efficiency and RTI performance.

Attribute set matching is the base method of publish-subscribe. Publish-subscribe implementations contain some or all the set matching between HVPs and publishing sets, publishing sets and subscription sets, HVPs and subscription sets. Two sets are matched when they have same attributes. Sending a message from the sender to the receiver usually needs multiple times of set matching. They usually include set matching in the sender and each receiver. Some dynamic ones and set decomposing may exist in different implementation.

Let the size of message HVP set be m and size of the subscription set be n. Then the time complication of set matching is $O(t) = O(m^*n)$. Each element requires traversal in the worst situation. Sorting the attribute handles in inserting can lower the time complication from $O(m^*n)$ to $O(n)$. The new time complication will be $O(n)$ plus time complication of the insert sort. Algorithms using other data structures than linear set will be much more complicated because the publish or subscription structures need transportation.

When it comes to the adaptive LoI-based publish-subscribe scheme, LoI simplifies the sort and matching of attribute set into a value comparison. That is comparing $\eta_j^{(i,o)}$ with $p_m^{(i,o)}$ in sender side and with $s_l^{(i,o)}$ on receiver side. This scheme can achieve fast matching without any insert sort or traversal. The time complication is $O(t) = O(1)$ plus that of $\eta_j^{(i,o)}$ computation. It is far lower than that of regular algorithms. Because most publish-subscribe sets have limited number of combinatorial attributes, the LoI classification is practical for the simulation requirements. The same effect can be attained with correct LoI extension, comparing with regular publish-subscribe process. Certainly, we can further perform precise matching in receiver side.

A sample of LoI based publish-subscribe scheme is shown in Figure 3, where the subscriber subscribes the class “person”. The outer rectangle labeled with “SR” is the bound of LoI “LAYER_VISION”. The inner one labeled with “D_SR” is the bound of “LAYER_ABOUT”. The HLA is extended according to the method introduced in section 4, including extended Tables 1 and 2. In Figure 3, persons in the D_SR rectangle send abundant data including position, rotation, color, pose, etc.



y
{x,y,z: LAYER_VISION,
rotx,roty,rotz,color,pose: LAYER_ABOUT,
face,shoes:LAYER_COMPONENT}
$s_l^{(i,o)}$ =LAYER_ABOUT
{x,y,z,rotx,roty,rotz,color,pose}
$\eta_j^{(i,o)}$≤LAYER_ABOUT
Subscriber
P_SR
{x,y,z}
$\eta_j^{(i,o)}$≤LAYER_VISION
D_SR
$s_l^{(i,o)}$ =LAYER_VISION
SR
O
x

Figure 3 Sample of LoI based publish-subscribe scheme.

to the subscriber. The subscriber can see them exactly. On the other hand, the subscriber gets less data, mainly about position, of the persons out of D_SR. In this way, most irrelevant messages can be abandoned without attribute set matching. Object instances in different LoI can also be treated separately.

7 RTI congestion control model based on LoI

The data explosion in large-scale distributed simulation cripples the performance of simulation and restricts the scalability. Because there is no relevance evaluation before, RTI cannot conduct congestion control on specific simulator features. LoI evaluates the message relevance, and then provides a way for RTI to control low-level transportation service.

To alleviate the congestion problem, it is necessary to cut down irrelevant messages as possible. We have implemented RTI congestion control by controlling the update frequency of each object instance based on its LoI. The congestion discovery and control algorithm are presented in another paper[17]. Here we further present the model of RTI congestion control as shown in Figure 4.

Above all, each object instance has been attached with LoI, according to the former scheme. Then each host node reduces sending and receiving messages while congestion occurs. This reducing is based on the publish/subscribe requirements. Two fundamental methods are adopted in sender/receiver side. The sender adjusts update frequency of local object instances based on their LoI. Object instances in lower LoI will send out less frequent messages than those in higher LoI. Here the update frequency is the frequency in which the sender actually sending out the object messages. Fewer messages are sent out, although the sender may still update in each simulation cycle. Similarly, RTI selectively drops unimportant messages and provides only necessary messages for the receiver. Then the receiver will deal with less messages. Because the reflect frequency and important messages are guaranteed, the system can run correctly although more messages are dropped.

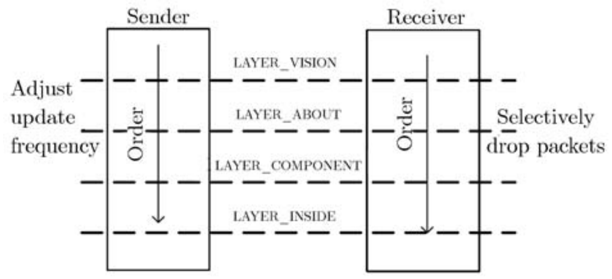


Figure 4 RTI congestion control model based on LoI.

Now several definitions in the model are given for the user to regulate according to specific application requirements. To obtain the real-time update frequency of object instance, we need to know how long it takes for the last several updates. We define the number of messages for statistics as a period. Let τ_s be the frequency statistics period, and we can obtain the continuous update frequency according to the time span every τ_s messages. Let τ be the filtering period that congestion control algorithm is applied cyclically.

Definition 9 (update frequency). The update frequency of an object instance is defined as $v = \frac{\tau_s}{\Delta t} = \frac{\tau_s}{t-t_0}$, where t_0 is the start time of the period and t is the end time of the period.

Users should provide standard update frequencies of LoI according to application requirements. It is acceptable for an object instance to be reflected in the standard update frequency. It should be noted that there may be several settings for federates in a simulation. So a distributed RTI has more advantages to preserve a specific setting for federates on an RTI.

Definition 10 (standard update frequency of LoI). Standard update frequency of LoI is marked as $f(l)$, where l is LoI of the object instance.

Then we can get $f(l) = f(p_m^{(i,o)})$ in sender host and $f(l) = f(s_l^{(i,o)})$ in receiver host.

Suppose there are n object instances in the simulation. These instances are of m types of object classes $C_1, C_2 \cdots C_m$, and there are n_i object instances of class C_i, where $i \in [1, m]$ and $\sum_{i=1}^{m} n_i = n$. In a simulation time, there are N_0 instances of LoI LAYER_CRITICAL, N_1 instances of LoI LAYER_VISION, N_2 of LAYER_ABOUT, N_3 of LAYER_COMPONENT and N_4 of LAYER_IN-

SIDE, where $\sum_{i=0}^{4} N_i = n$.

Let $U(I_j)$ be the required update frequency of object instance I_j for other host nodes to process correctly. When lack of relevance evaluation for the update messages, the total data traffic produced in the host node will be $\sum_{j=1}^{n} U(I_j) = n^*U(I_j) = n^*U_0$, where U_0 is the update frequency required in the simulation and $U_0 >= f(l)$.

As to the LoI based sender, the total produced data traffic will be

$$\begin{aligned}
\sum_{j=1}^{n} f(l_j) &= \sum_{j=1}^{n} f(p_m^{(i,o)}) \\
&= N_0U_0 + \sum_{j=1}^{N_1} f(\text{LAYER\_VISION}) \\
&\quad + \sum_{j=1}^{N_2} f(\text{LAYER\_ABOUT}) \\
&\quad + \sum_{j=1}^{N_3} f(\text{LAYER\_COMPONENT}) \\
&\quad + \sum_{j=1}^{N_4} f(\text{LAYER\_INSIDE}) \\
&= N_0U_0 + N_1 f(\text{LAYER\_VISION}) \\
&\quad + N_2 f(\text{LAYER\_ABOUT}) \\
&\quad + N_3 f(\text{LAYER\_COMPONENT}) \\
&\quad + N_4 f(\text{LAYER\_INSIDE}).
\end{aligned}$$

Suppose there are n' object instances in the receiver host and corresponding LoI instance counts are N_0', N_1', N_2', N_3' and N_4', where $\sum_{i=0}^{4} N_i' = n'$, the total data traffic received in the host node will be $\sum_{j=1}^{n'} U(I_j) = n'U(I_j) = n'U_0$.

As to the LoI based receiver, the total received data traffic will be

$$\begin{aligned}
\sum_{j=1}^{n'} f(l_j) &= \sum_{j=1}^{n'} f(s_l^{(i,o)}) \\
&= N_0'U_0 + \sum_{j=1}^{N_1'} f(\text{LAYER\_VISION}) \\
&\quad + \sum_{j=1}^{N_2'} f(\text{LAYER\_ABOUT}) \\
&\quad + \sum_{j=1}^{N_3'} f(\text{LAYER\_COMPONENT}) \\
&\quad + \sum_{j=1}^{N_4'} f(\text{LAYER\_INSIDE}) \\
&= N_0'U_0 + N_1' f(\text{LAYER\_VISION}) \\
&\quad + N_2' f(\text{LAYER\_ABOUT}) \\
&\quad + N_3' f(\text{LAYER\_COMPONENT}) \\
&\quad + N_4' f(\text{LAYER\_INSIDE}).
\end{aligned}$$

It can be seen from the above that the bandwidth used by the object instances is divided into four groups plus bandwidth for critical messages. Those critical update/reflect messages are not dropped. Object instances of higher LoI can be put more bandwidth on. The setting task assigned to end users makes the model more flexible. In this way, we can control the total traffic by regulating update/reflect messages according to the specific object instance requirements.

8 LoI implementation in RTI

We developed an RTI software, and the adaptive publish-subscribe scheme and RTI congestion control are implemented in it. This section briefly introduces the implementation.

RTI is the software infrastructure of HLA systems, which is similar to a distributed operating system for simulation applications from the view of function. Federates interoperate based on RTI, using the HLA-defined programming interface provided by LRC (local RTI component). We have developed BH RTI in the last few years, adopting a distributed run-time structure (Figure 5). A federation can be supported by several peer to peer BH RTIs without CRC (central RTI component). Each BH RTI serves some federates and maintains only interested data via multicast communication. So it can also act as a centralized RTI when no other BH RTI exists in the federation. BH RTI has been released at http://www.hlarti.com since August 2006.

The main modules of BH RTI consist of two parts: LRC and rtiexec (Figure 6). LRC provides the HLA programming library conforming to the standard interface specification. The rtiexec implements the HLA services including federation management, object management, declaration management, ownership management, time management, data distribution management and MOM (management object model). Because the adaptive LoI-based publish/subscribe scheme unifies the publish/subscribe mechanism of OM, DM and DDM, Module ObjectManager and DataManager implement OM, DM and DDM services.

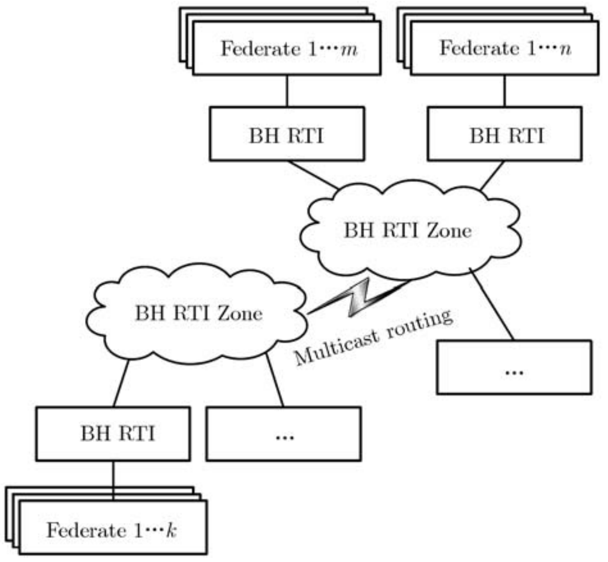


Figure 5 Distributed run-time structure of BH RTI.

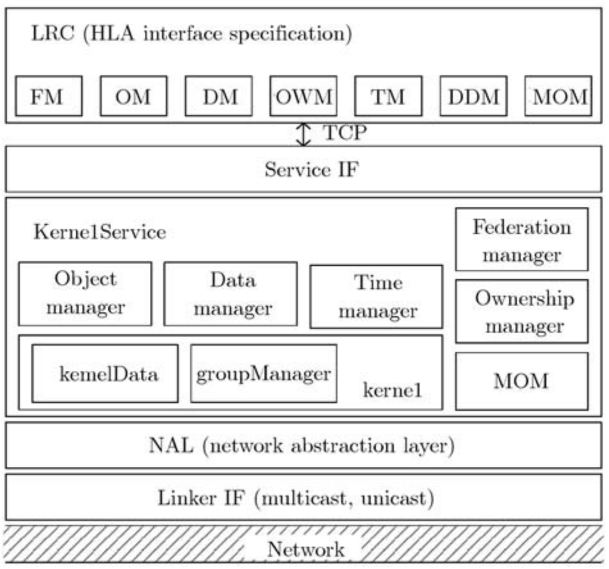


Figure 6 BH RTI main modules.

8.1 Adaptive LoI-based publish/subscribe process

The LoI extension and adaptive publish-subscribe scheme are applied into the work mechanism of BH RTI. Figure 7(a) shows the process of publishing object class, registering object instance and sending attribute update values. Figure 7(b) shows the process of subscribing object class, discovering object instance and receiving attribute reflect values. The two figures give the concrete publish-subscribe processes using LoI. Here four processes are included: HLA application (federate), LRC, BH RTI and other BH RTIs. RTI in the figure means rtiexec of BH RTI. BH RTI provides services for all the HLA applications via their LRCs, each LRC for one federate. In the whole federation, the BH RTI communicates with other BH RTIs in multicast. In short, BH RTI gets the LoI collections of local object instances, publishes them to other BH RTIs, discovers remote object instances that are subscribed and delivers their data to required federates.

8.2 RTI congestion control

RTI congestion control is implemented in BH RTI's NAL (network abstraction layer) module (Figure 8). It is used to control the traffic of PDU (protocol data unit) packets that contain object instance update/reflect. According to the congestion control model, when congestion occurs, NAL will get the object instance handle of data PDU and perform local LoI filtering. Correspondingly, when a data PDU is received, NAL will get the remote object instance handle and perform remote LoI filtering. The local and remote filtering are of the same filtering mechanism that controls the update/reflect frequency in the way described in section 7. The earlier LoI filtering algorithm is introduced in our previous paper[17], and we are still working on an update algorithm[18].

9 Performance evaluation

In this section, we investigate the efficiency of adaptive publish-subscribe scheme, RTI congestion control and some practice in large-scale distributed simulations.

9.1 LoI-based adaptive publish-subscribe scheme

Here is an instance of study on filtering efficiency of publish-subscribe scheme. Assume there is an

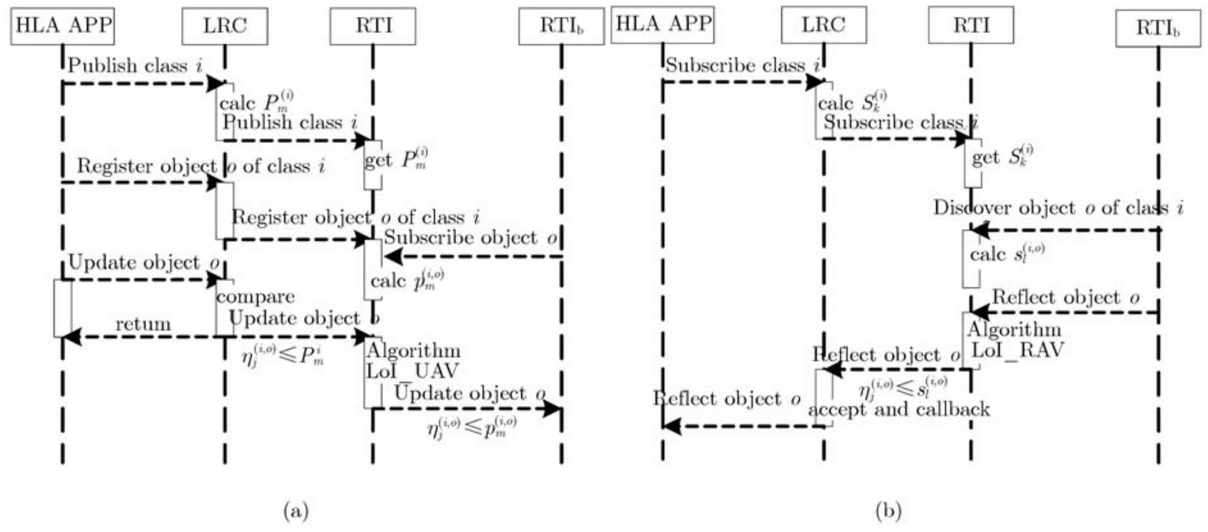


Figure 7 Adaptive LoI-based publish/subscribe process in BH RTI. (a) Publish process; (b) subscribe process.

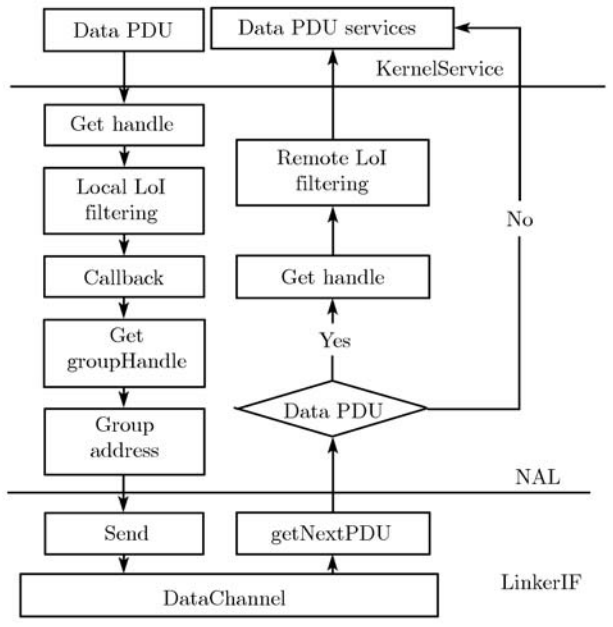


Figure 8 Congestion control implementation in BH RTI.

object which has 35 attributes. The attribute numbers of LoIs is as Table 2.

Table 2 Object attribute number of LoI

LoI	Attribute number
LAYER_CRITICAL	0
LAYER_VISION	5
LAYER_ABOUT	10
LAYER_COMPONENT	10
LAYER_INSIDE	10

Then the subscriber will receive attribute updates of different attributes according to their $S_k^{(i)}$. The attribute receiving is as Table 3.

Table 3 Attributes to receive in Federate subscriber

LoI $S_k^{(i)}$	Attribute number
LAYER_CRITICAL	0
LAYER_VISION	5
LAYER_ABOUT	5+10 = 15
LAYER_COMPONENT	5+10+10 =25
LAYER_INSIDE	5+10+10+10 = 35

When a packet is composed, the packet attribute set and subscription attribute set are required to match during publish-subscribing. Table 4 is the matching count of LoI-based scheme and existing standardized scheme. From the result we can see that LoI-based scheme is efficient in publish-subscribe set matching. The efficiency of existing scheme varies much according to different situations.

The efficiency of publish-subscribe scheme is one of the major factors of RTI's one-trip delay. The delay comparison of some commonly used RTIs is performed. The experiments are conducted in network using Huawei Switch Quidway S3050. The detailed setup is shown in Table 5.

The setup for experiments is shown in Figure 9(a) for traditional DMSO RTI like RTI experiments and Figure 9(b) for distributed RTI experiments. The delay is tested using the RTI Benchmark 1.3 published in DMSO website.

Table 4 Matching count of publish-subscribe set

LoI $\eta_j^{(i,o)}$	Attribute number of packet	LoI $S_k^{(i)}$	LoI-based scheme	Existing scheme
LAYER_VISION	5	LAYER_VISION	1	1
LAYER_VISION	5	LAYER_ABOUT	1	5
LAYER_VISION	5	LAYER_COMPONENT	1	5
LAYER_VISION	5	LAYER_INSIDE	1	5
LAYER_ABOUT	10	LAYER_VISION	1	1
LAYER_ABOUT	10	LAYER_ABOUT	1	1
LAYER_ABOUT	10	LAYER_COMPONENT	1	10
LAYER_ABOUT	10	LAYER_INSIDE	1	10
LAYER_COMPONENT	10	LAYER_VISION	1	1
LAYER_COMPONENT	10	LAYER_ABOUT	1	1
LAYER_COMPONENT	10	LAYER_COMPONENT	1	1
LAYER_COMPONENT	10	LAYER_INSIDE	1	10
LAYER_INSIDE	10	LAYER_VISION	1	1
LAYER_INSIDE	10	LAYER_ABOUT	1	1
LAYER_INSIDE	10	LAYER_COMPONENT	1	1
LAYER_INSIDE	10	LAYER_INSIDE	1	1

Table 5 Host setup for delay experiment

Host Id	CPU (GHz)	RAM (MB)	OS	NetworkCard (Mbit)
A1	P4 2.8	512	winXP	10/100
A2	P4 3.0	512	winXP	10/100
A3	P4 2.4	768	winXP	10/100

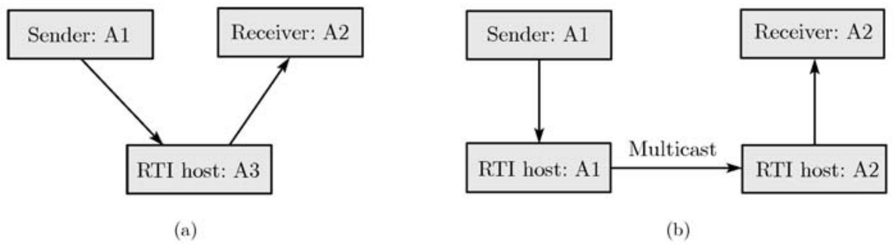


Figure 9 Setup for RTI experiments. (a) Setup for cnetral RTI experiment; (b) setup for distributed RTI experiment.

The RTI delay experiment results are as Figure 10, in which BH RTI 2.2 (central mode), DMSO RTI 1.3NGv6, pRTI 1516v2.3 is using the setup of Figure 9(a) and BH RTI 2.2 is using the setup of Figure 9(b). The central mode of BH RTI indicates that several federates connect to a unique BH RTI for RTI services. From the experiments results, we can see that BH RTI 2.2 has comparatively smaller delay. The distributed mode of BH RTI 2.2 is premier in the experiments. The central model of BH RTI 2.2 is relatively smaller in small payload, and there is an increase when the payload adds. Because the publish-subscribe efficiency has nothing to do with payload, the increase should be owing to other factors in RTI implementation. From the experiment, we can see that LoI can improve message delivery performance considerably.

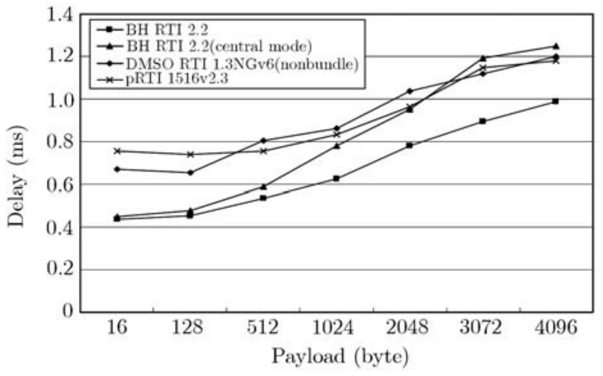


Figure 10 Delay Comparison of RTIs.

9.2 RTI congestion control

Experiments are conducted to evaluate the RTI congestion control. We use four computers to conduct the experiment. The computer setup is shown in Table 6. The sender computers, S1 and S2, are simulating 500 object instances individually based on BH RTI. Each receiver computer, R1 or R2, starts a BH RTI receiving all the update messages of total 1000 object instances. To investigate the congestion control independently in the receiving computers, congestion control is disabled in the sender computers. In this way, we can see what ability the receivers have to treat the traffic. All the incoming packets are looked on as "original data", and packets after congestion control are looked on as "data after congestion control". Then we can see the congestion control efficiency in computer R1 and R2.

Table 6 Host setup for RTI congestion control experiment

Host Id	CPU (GHz)	RAM	OS
S1	P4 3.2	1 GB	winXP
S2	P4 3.2	1 GB	winXP
R1	P4 3.0	512 MB	winXP
R2	P4 3.0	512 MB	winXP

The experiments are carried out in 100M Ethernet LAN using Huawei Switch Quidway S3050. According to the system requirements of R1 and R2, $f(l)$ is set for them. Here are the settings.

Packet size: 220 bytes, including 10 attribute values, 8 bytes for each.

S1, S2: no congestion control. 500 objects in LAYER_ABOUT

R1: 400 objects in LAYER_ABOUT and 600 objects in LAYER_VISION; f(LAYER_ABOUT)=20, f(LAYER_VISION) = 5.

R2: 400 objects in LAYER_ABOUT and 600 objects in LAYER_VISION; f(LAYER_ABOUT) = 10, f(LAYER_VISION) = 2.

The bandwidth usage results are shown in Figure 11. From the figure we can see that R1 and R2 have endured the total bandwidth more than 70 Mbps. But after congestion control, the bandwidth is cut down to about 20 and 10 Mbps respectively. Then we select two object instances from R1 and R2 to check the update fidelity. From Figure 12 we can see that each object instance maintains a successive update frequency. The total update frequency of object instance in LAYER_ABOUT is frequency of LAYER_VISION; plus that of LAYER_ABOUT plus that of LAYER_CRITICAL, none in this experiment. The update frequencies of object instances in LAYER_ABOUT are high enough in each LoI, LAYER_VISION and LAYER_ABOUT, for the application to treat with. The update frequencies of object instances in LAYER_VISION are also successive. The successive update in each LoI guarantees real-time update for each attribute with the help of LoI.

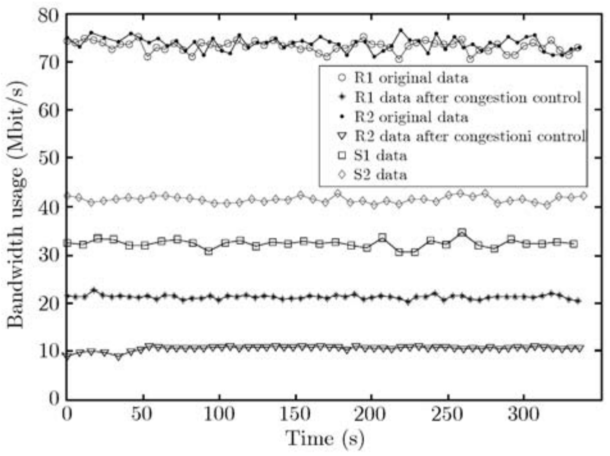


Figure 11 Bandwidth usage.

9.3 Practice in large-scale distributed simulation

BH RTI has been applied in several large-scale distributed exercises. In a battlefield demonstration,

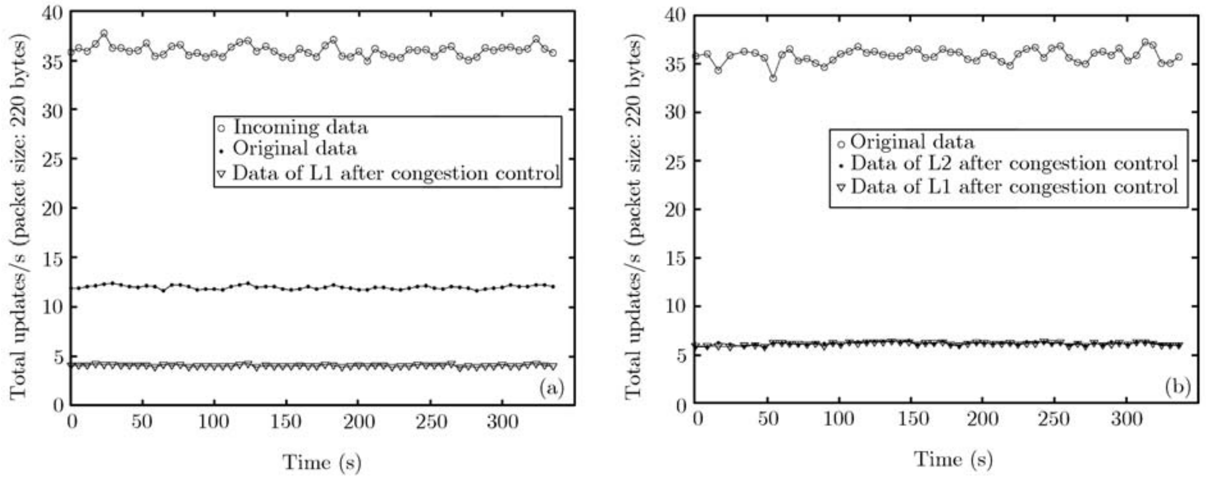


Figure 12 Object reflect results. (a) Host R1's object in LAYER_VISION; (b) Host R2's object in LAYER_ABOUT.

47200 object instances are simulated under the hybrid DDM implementation, using 50 PCs. The network topology is as Figure 13. Three subnets are distributed in the 1st floor and 3rd floor of building RuXin and the 4th floor of building YiFu in Beihang University. They are connected by fiber and routers. Extreme Switch Summit 24 and two Switch Summit 48 are used, supporting PIM multicast routing protocol. The average delay between two neighboring subnets are illustrated in the figure, using "ping" to test. All PCs are about P4 1.5 GHz to P4 2.8 GHz, 512 MB memory. Because they can only render up to 200 to 300 tank or helicopter 3D models used in the exercise, most simulators rendered 2D display in their output. The viewer federates had the 3D view output.

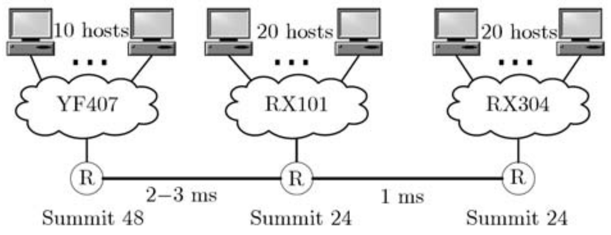


Figure 13 Network ropology.

Each host launched a BH RTI and simulated 800 to 1400 tank or helicopter object instances. Dynamic bullet/shot entities were not counted in it. The total environment is divided into 30×30 blocks, and each block in use is assigned to a multicast address. In the simulation we set Extend at 2. Object instances of LAYER_VISION will update/reflect 5 times per second. Object instances of higher LoIs will update/reflect 20 times per second. Those BH RTIs, which serve radars and viewers, had another setting of 5 reflects per second for all the object instances. The lower frequency is enough for their requirements. With the help of multicast and LoI-based technologies, we succeed in running the large-scale distributed simulations. A scene of the simulation supported by BH RTI is as Figure 14.

Figure 14 3D scene in large-scale distributed simulation supported by BH RTI.

10 Conclusion

In this paper, aiming at the relevance evaluation problem, a LoI technique is proposed. We made the following work.

(1) LoI is presented, which makes a message relevance classification based on the impact of spatial distance on receiving attributes and attribute values. Its extension to HLA requires small modifications and maintains federate code compatibility.

(2) On the basis of LoI, we arrived at two important deductions, and proposed an adaptive publish-subscribe scheme. The scheme can abandon most irrelevant messages directly before precise class/attribute publish-subscribe matching.

(3) RTI congestion control model is proposed. It can use the bandwidth more efficiently by controlling the update frequency of object instances based on LoI.

LoI related techniques support HLA-based systems. Some important issues in its implementation are also introduced. We have evaluated LoI techniques in both experiments and applications. Matching count computation and RTI delay experiments showed that LoI can speed up the message filtering considerably. The experiment result of RTI congestion control showed that it cuts down the traffic a lot. A simulation exercise of using BH RTI to support nearly 50000 object instances is introduced, including its network topology, configurations and settings.

1 Morse K L, Bic L, Dillencourt M. Interest management in large-scale virtual environments. Presence: Teleoperat Virtual Environ, 2000, 9(1): 52–68

2 Abrams H A. Extensible interest management for scalable persistent distributed virtual environments. Dissertation for the Doctoral Degree, Monterey: Naval Postgraduate School, 1999. 17–26

3 Zyda M, Brutzman D, Darken R, et al. NPSNET——Large-scale virtual environment technology testbed. In: Proceedings of the International Conference on Artificial Reality and Tele-Existence. 1997. 18–26

4 IEEE Std 1278.1, Standard for Information Technology——Protocols for Distributed Interactive Simulation Applications, 1995

5 IEEE. 2000. IEEE Std 1516-2000: IEEE standard for modeling and simulation (M&S) High Level Architecture (HLA)–Framework and rules

6 IEEE. 2000. IEEE Std 1516.1-2000: IEEE standard for modeling and simulation (M&S) High Level Architecture (HLA)–Federate interface specification

7 U.S. Department of Defense. High Level Architecture (HLA) – Federate Interface Specification, Version 1.3, April 1998. available through the Internet: http://www.dmso.mil

8 Torpey M, Wilbert D, Helfinstine B, et al. Experiences and lessons learned using RTI-NG in a large-scale, platform-level federation. In: Proceedings of the Spring Simulation Interoperability Workshop. 2001. Paper 00F-SIW-031

9 Helfinstine B, Wilbert D, Torpey M, et al. Experiences with data distribution management in large-scale federations. In: Proceedings of the Fall Simulation Interoperability Workshop. 2001. Paper 01F-SIW-032

10 Hyett M, Wuerfel R. Connectionless mode and user defined DDM in RTI-NG V6. In: Proceedings of the Spring Simulation Interoperability Workshop, 2003. Paper 03S-SIW-102

11 McLean T, Fujimoto R, Fitzgibbons B. Middleware for real-time distributed simulations. Concurr Comput Pract Exper, 2004, 16(15): 1483–1501

12 Zhao H, Georganas N D. HLA real-time extension. Concurr Computa Pract Exper, 2004, 16(15): 1503–1525

13 Zabele S, Stanzione T, Kurose J, et al. Improving distributed simulation performance using active networks. (Invited Paper) In: Proceedings of the World Multi Conference, 2000

14 Zabele S, Dorsch M, Keaton M, et al. Dynamic interest filtering for optimal state update messaging. In: Proceedings of the I/ITSEC (Interservice/Industry Training, Simulation & Education Conference), 2001

15 Cai W T, Lee F B S, Chen L. An auto-adaptive dead reckoning algorithm for distributed interactive simulation. In: Proceedings of the 13th Workshop on PADS (Parallel and Distributed Simulation), 1999: 82–89

16 Zhou S P, Turner S J, Cai W T, et al. A utility model for timely state update in distributed wargame simulations. In: Proceedings of the 18th workshop on PADS (Parallel and Distributed Simulation). 2004. 105–111

17 Zhou Z, Zhao Q P. Research on RTI congestion control based on the layer of interest (in Chinese). J Software, 2004, 15(1): 120–130

18 Zhou Z, Zhao Q P. LoI-based flow control on low-bandwidth federates. In: Proceedings of the 2nd International Conference for E-Learning and Games, Edutainment 2007, Hongkong, 11-13 June, 2007. Springer LNCS vol. 4469: 904–915