A time-space consistency solution for hardware-in-the-loop simulation system

Zexin Jiang[a]

Electric Power Research Institute of Guangdong Power Grid Co.Ltd., Guangzhou, P.R.China.
CSG KEY Laboratory of Power Grid Automation, Guangzhou, P.R.China.

Abstract. Time-Space consistency is critical while designing hardware-in-the-loop simulation (HILS) system, due to the HILS system's Real-Time performance need. We find out that devices in HILS system can be classified into two types -- positive device and passive one. And only the positive device would generate Time-Space inconsistency problem. Then, we deeply analyses three different phenomena of Time-Space inconsistency generated by the positive device. A device state mapping solution is proposed and used in practise HILS project.
Keywords: Hardware-In-the-Loop Simulation (HILS); Time-Space inconsistency; device access proxy mechanism (DAPM); device state mapping (DSM) solution.

1 Introduction

Hardware-in-the-loop simulation (HILS) means devices can access to the simulation system and interact with it. Compared with the pure mathematical simulation, HILS is more credible and closer to the real world. HILS is widely used to test and verify new system. Ref [1] and Ref [2] introduce various HILS applications.

From time slot perspective, the essential characteristic between HILS system and non-hardware-in-the-loop simulation (non-HILS) system is that: HILS system needs to guarantee its real-time simulation (RTS) capability while the non-HILS does not. RTS means the time in the simulation system (termed simulation time or logical time) needs to synchronize with the time in the real world (termed wall clock time). Because devices work in the real world and we are unable to change the real world's clock. On the contrary, models work in the virtual simulation world and we can regulate their time advance speed.

Viewed from the relationship between the simulation time and the wall clock time, there are two types of simulation system – faster than real-time simulation (FRTS) and slower than real-time simulation (SRTS). FRTS defines when the simulation time runs faster than the wall clock time, and SRTS defines the simulation time runs slower than the wall clock time. Particularly, when the simulation time runs synchronize with the wall clock time, we called real-time simulation (RTS).

Let us define r = simulation time / wall clock time, which is the ratio of simulation time - wall clock time. This ratio represents the sizes of the simulation time lapse in a unit wall clock time. Hence, we has r=1 for RTS, SRTS for $0<r<1$, and FRTS for $r>1$.

[a] Corresponding author : jiangzexin-08@tsinghua.org.cn

In FRTS system, we can slow down simulation time advance speed and guarantee events' causal relationship order to make r=1. Oppositely, SRTS system is usually difficult to accelerate its simulation time unless to optimize the simulation system's performance.

Hence, RTS system denotes the simulation time can runs equal with or faster than the wall clock time. That includes two aspects capability: one is the simulation time advance efficiency in the simulation system, the other is the capability to real-timely deal the messages come from devices out of the simulation system.

As to the first aspect, references [3-6] are mostly aiming to solve this problem. Their mainly ideas are giving an efficient time management algorithm. Especially in the distributed HILS system, there are two types of time management algorithm: Termed conservative time synchronization algorithm and optimistic time synchronization algorithm.

As to the second aspect, the capability of real-timely dealing the messages from devices is ignored in these proposed works. This real-timely dealing message capability depends on simulation system's messages tackle throughput and devices' messages generate throughput. Our main contribution is to deeply analyse these phenomena and proposes a proper solution to tackle it.

First of all, we found out that devices can be divided into two types: -- positive device and passive one. Time-space inconsistency problem is prone to be happened when positive devices access to the simulation system. On the contrary, it is easy to keep time and space consistency when passive devices access to the simulation system. Due to passive devices never send messages to simulation system before simulation system sends messages to them.

2 Device access proxy mechanisms

Device access proxy mechanism (DAPM) is proposed in reference [1], which is with the "virtual-reality interchange" principle. Assume simulation system contains three modules A, B, C. Each module runs with a simulation model or a device. With DAPM, it is transparent to model in module A whether Access proxy in module B connects to model or device. That is to say, models in the simulation system cannot feel whether they are communicating with real device or virtual model, just like Turing intelligent testing. DAPM provides a mechanism for various hardware devices to access to simulation system.

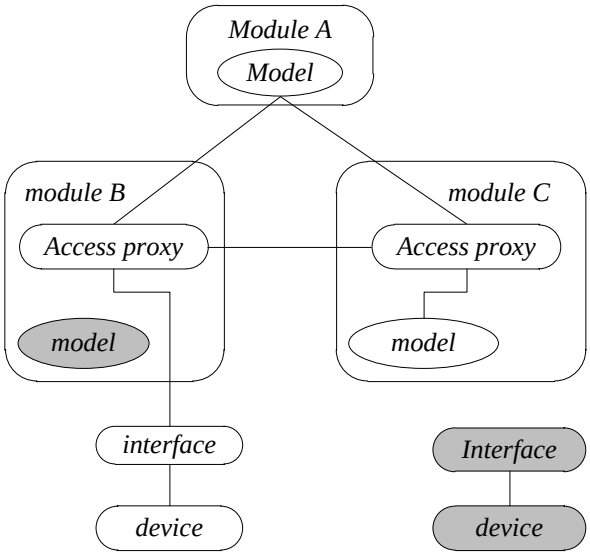

Figure 1. Diagram of DAPM

DAPM presents a good approach to implement HILS. But the Time-Space inconsistency problem in HILS still exists. And we are focus on solving the Time-Space inconsistency problem by giving a detail inner implement of device access proxy.

3 Time-space inconsistency problem

Due to devices work in the real world and run with the wall clock time coordinate, the state in devices keep simultaneously along the wall clock time. Simulation models in the virtual world are running with the virtual simulation time coordinate. In HILS system, we need to synchronize the wall clock time with the virtual simulation time (also termed logical time), and hence the simulation time advance speed can keep consistent with the wall clock time, so the HILS system works.

In HILS system, devices send messages to the simulation system and they would receive messages from the simulation system as well. Message contains a timestamp attribute. As to message sent to the simulation system by devices, the timestamp can only be referenced to the wall clock time. In the contrast, as to message from the simulation system (especially in HLA), its timestamp can only be referenced to the simulation time.

Normally, in RTS system, simulation time advance speed in simulation system equals with the speed of wall clock time lapse, and so messages sent from devices can be real-timely received and deal by simulation system. In this case, the state and its corresponding wall clock timestamp of devices keep consistent between real world and simulation system. However, when a simulation process in simulation computer occurs to suddenly pause or temporary stop, the simulation virtual time pause while the wall clock real one continue. The state and its corresponding timestamp in devices could not keep consistent between real world and simulation system.

In order to analyse this problem deeply, we would like to classify devices into two types shown as Fig.2.

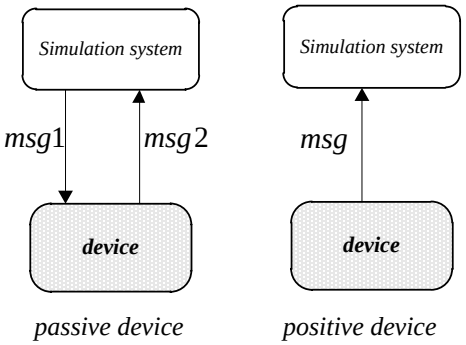

Figure 2. Classification of devices

Passive devices refer to those devices that only send response message msg_2 to simulation system when they received request message msg_1. Such as motors.

Positive devices refer to those devices that automatically periodically send message to the simulation system without any motivate out from the simulation system, such as same active sensors.

As to passive devices, whether simulation system processes run fast or not, the timestamp of msg_1 and the timestamp of msg_2 (t_1 for msg_1 and t_2 for msg_2) always be $t_1 < t_2$. Msg_2 could be considering being the state of devices when simulation time is t_2. And devices' state and its corresponding timestamp can be perceived with little deviation in the simulation system.

As to Positive devices, assume devices actively periodicity send *msg* to report their stat every Δt. However, simulation system can feel the latest stat of devices only when he received the *msg* and deal it. Suppose devices send message msg_i (i=1,2,…,n) over a period of time, and their corresponding wall clock timestamp in the real world is t_1, $t_1+\Delta t$, $t_1+2\Delta t$, … $t_1+(n-1)\ \Delta t$, while their corresponding simulation system receive time (simulation time) in the virtual world is t_{si} (i=1,2,…,n) . In normal RT

simulation situation $t_{s(i+1)}$-t_{si}=Δt , in other word, devices' state in simulation system keep synchronized with it in the real word. In abnormal situation, if simulation system processes occur to suddenly pause with wall clock time interval δ or temporary stop at t_{si}+Δt, three phenomena of time-space inconsistent appear.

3.1 Device messages lost phenomenon

Assume no messages reception buffer in device access interface, messages from device could be lost in the period of simulation system pause of δ. The simulation system misses those messages and simulation experiment fails.

Take $\delta = 2\Delta t$ as an example, the wall clock time - simulation time relationship diagram shown as Fig.3. Msg_3 and msg_4, which are essential to the simulation such as tele control commands or key stat messages, were lost during δ, and the simulation experiment cannot simulate the true results even the opposite results.

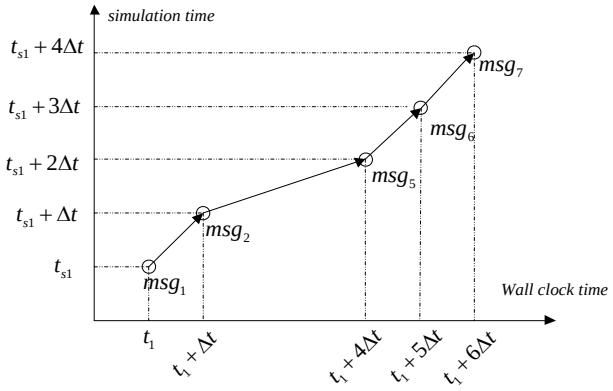

Figure 3. Message loss phenomenon diagram

3.2 Device messages lag phenomenon

Assume messages reception buffer (FIFO Queue Buffer) was setup by device access interface in simulation system and devices outside simulation system send messages period with an interval time of Δt. In the real world, messages were sent with Δt wall clock time interval, that is msg_{n+1} @$t_{w(n+1)}$, msg_n @ t_{wn} and $t_{w(n+1)}$ - t_{wn} =Δt.

In the simulation world, messages were received and labelled with t_{si} separately, t_{si} means the simulation time at the moment when the messages were received. And we could not simply let t_w = t_{si}, due to the causal sequence of events would be confused in the simulation system [3-6].

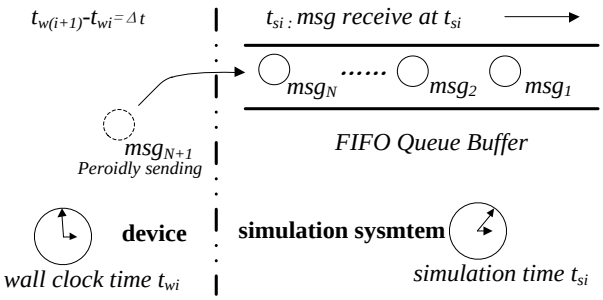

Figure 4. FIFO in simulation system diagram

Assume simulation timestamp of messages are increased by Δt from the timestamp of the latest one. This case is shown as Fig.5, during some reason of pause with δ, simulation system does nothing in $t_1+\Delta t \sim t_1+3\Delta t$. Unfortunately simulation system at $t_1+4\Delta t$ (the wall clock time) would deal msg_5 and it deals msg_3, so the simulation system lag to recognize the real one with $2\Delta t$. Once the sudden pause moment occur frequently, the expenditure of wall clock time with doing nothing would accumulate and time-space inconsistent phenomenon appear.

In fact, it is unwise to mark the simulation timestamp of msgi+1 with the simulation timestamps of $msg_i+\Delta t$ (that is $t_{si+1}=t_{si}+\Delta t$). Once the simulation system performs faster than the real world, it will receive past messages and its simulation causal relationship would be confused. However, if $t_{si+1}\neq t_{si}+\Delta t$, the third time-space inconsistent phenomenon appear.

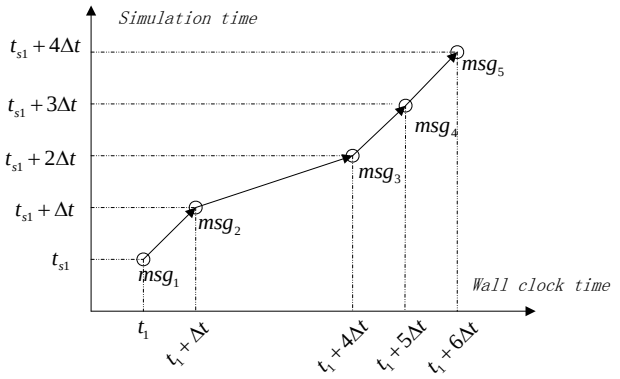

Figure 5. Message lagged deal scenario diagram

3.3 Simulation cannot advance phenomenon

Assume messages reception buffer (FIFO Queue Buffer) was setup by the device access interface in simulation system and devices outside simulation system send messages arbitrarily. In the real world, messages were sent with disorder wall clock time.

In the simulation world, messages were received and we could only label them with t_{si} separately, t_{si} means simulation time at the moment when the message was received. In normal circumstances, due to simulation time advances synchronously with wall clock time, HILS performs well.

This case is shown as Fig.6, during some reason of pause with δ, simulation system does nothing in $t_1+\Delta t \sim t_1+3\Delta t$. Unfortunately the simulation system at $t_1+4\Delta t$ (the wall clock time) would receive and deal msg_5 , it receives and deals messages in the FIFO Queue buffer orderly. Once the simulation system cannot finish all messages in the buffer before $t_1+5\Delta t$, messages in the queue buffer may increase. They will be labelled with the current simulation timestamp and so the simulation system has no opportunity to advance its simulation time. The simulation system would deals messages continually but simulation time pauses.

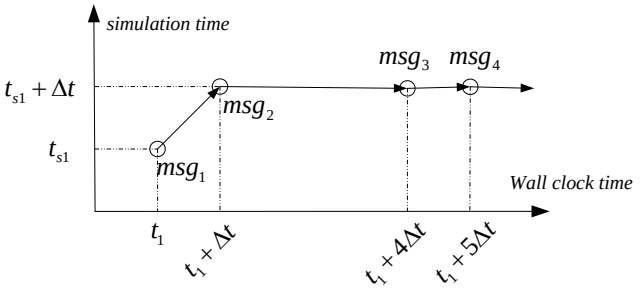

Figure 6. Simulation time pausing phenomenon diagram

4 Device state mapping solution

The critical reason of the three time-space inconsistency phenomena above is the lack capability to deal messages from devices. With the framework of DAPM, we propose a device state mapping solution to reduce messages piling.

4.1 Device state mapping table

We define a device state mapping table as Tab.1, contains three fields: state variable name, state variable value, changed flag. Simulation system treats the table as device outside, while a thread named Device State Daemon runs background to receive messages from device and change the state variable value correspondingly.

Table 1. DEVICE state mapping table

variable name	variable value	changed flag
X1	…	0/1
X2	…	0/1
X3	…	0/1
…	…	…
Xn	…	0/1

4.2 Inner architecture with DAPM

We design the inner architecture with DAPM shows as Fig.7. In device access proxy, the device state daemon thread receives messages from device interface through socket and updates the device state mapping table. In the simulation system, the simulation system thread inserts events to query the device state mapping table at regular intervals, the event firstly checks the table's changed flag field and tackles all the changed states.

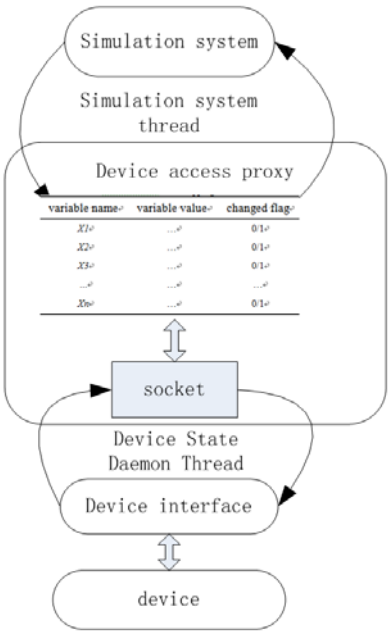

Figure 7. inner architecture with DAPM diagram

With this solution, once simulation system processes in the simulation computer occurs to suddenly pause or temporary stop, the device state daemon thread could continue work and the device state mapping table is changing and updating when messages come in. When simulation system process recover working, it can deal one event but lots of messages all completed. There is no message loss, message lag, message pile, and so no time-space inconsistency problem.

The device state mapping solution is effective only to those systems related to states but to processes. Because during the pause of the simulation system process, the device state may change several times and the last state can be recognized to the simulation system.

5 Conclusions

The device state mapping solution to solve three time-space inconsistent problem is used in several applications and the results show that it works and performs excellent.

References

1. H.DONG. Design and Implementation on Software Framework of Hardware-In-the-Loop Simulation System for High-Speed Railway [D], Beijing: Tsinghua University (2009)
2. Z.JIANG. Design and Implementation on a HLA-based Run Time Infrastructure for Hardware-In-the-Loop Simulation [D], Beijing: Tsinghua University (2011)
3. H.Zhong. Research on Time-Space Consistency in Large-Scale Distributed Simulations [D], Changsha: National University of Defense Technology (2005)
4. X.Wang. Research on the Time Management Technology in Parallel and Distributed Simulation Systems [D], Changsha: National University of Defense Technology (2006)
5. Y.Yao. Research on the key technologies of high performance runtime infrastructure of distributed interactive simulation [D], Changsha: National University of Defense Technology, (2003)
6. Z.JIANG, W.DONG, Y.JI. A conservative time management model with optimal degree of parallelism for distributed simulation [C]. *Performance Evaluation of Computer & Telecommunication Systems (SPECTS) 2011, The Hague*, 241 – 246(2011)