CoSe: Collaborative Simulation Environment for Heterogeneous Models

Huilong Chen[1], Yiping Yao[1,2], Laibin Yan[1], Feng Zhu[1], Jin Li[1]

[1] College of Information System and Management, National University of Defense Technology, Changsha 410073, China

[2] State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha 410073, China

email: chenhuilong@nudt.edu.cn

Keywords: Multidisciplinary Modeling; Distributed Computing Environment; Hierarchical Modeling Method; Synchronous Control Algorithm

Abstract. Modeling and simulation of complex systems in nature and society usually involves multidisciplinary studies. In many simulation systems, it is necessary to develop models using different modeling languages and tools. As a result, it will be a big challenge in the modeling of the whole system and collaborative calculation of the heterogeneous models. In this paper, we propose a collaborative simulation environment for heterogeneous models (CoSe) to support modeling and simulation for complex systems. Based on our early work of a hierarchical modeling approach, we improve the early synchronous control algorithm and present a hybrid synchronous algorithm. Experiment results and the case study illustrates that CoSe supports both hierarchical modeling for complex systems and efficient collaborative simulation of heterogeneous models with a good scalability.

Introduction

Collaborative simulation organizes heterogeneous models to work together to analyze and evaluate complex systems in nature and society. In many cases, the heterogeneous models are located in different geographical areas, calculated on different computing systems, or developed with different modeling languages and modeling tools [1] [2]. The motivation of collaborative simulation is to solve simulation problems within the course of designing complex system [3]. Although some existing simulation tools, to some degree, are able to solve most analysis and evaluation issues of complex system in single domain, they lack the support for complex system simulation in multi-domain [4]. With the increasing complexity of simulated systems, it is not sufficient to model and simulate complex systems using only one kind of simulation tools [5]. Therefore, collaborative simulation for modeling and simulation of complex systems in multi-domain becomes an important trend [6] [7].

In modeling and simulation of many complex systems, computational models may be developed based on multidisciplinary knowledge, such as hydro-mechanics, aerodynamics, mechanical control and many other disciplines. Not only that, computational models may be developed using different languages and tools of modeling and simulation, such as Matlab, standard C/C++, Fluent, Adams and so on. Therefore, a collaborative modeling and simulation system usually involves many computational models based on multidisciplinary knowledge [8], which is characterized of a heterogeneous and hierarchical structure [9] [10]. As a result, it brings a great challenge in modeling of the complex systems based on multidisciplinary knowledge and collaborative simulation of the heterogeneous models. In this paper, we propose a collaborative simulation environment for heterogeneous models (CoSe) to support efficient modeling and simulation for complex systems based on our former work [11]. Experiment results and the case study illustrates that CoSe supports both hierarchical modeling for complex systems and efficient collaborative simulation of heterogeneous models with a good scalability.

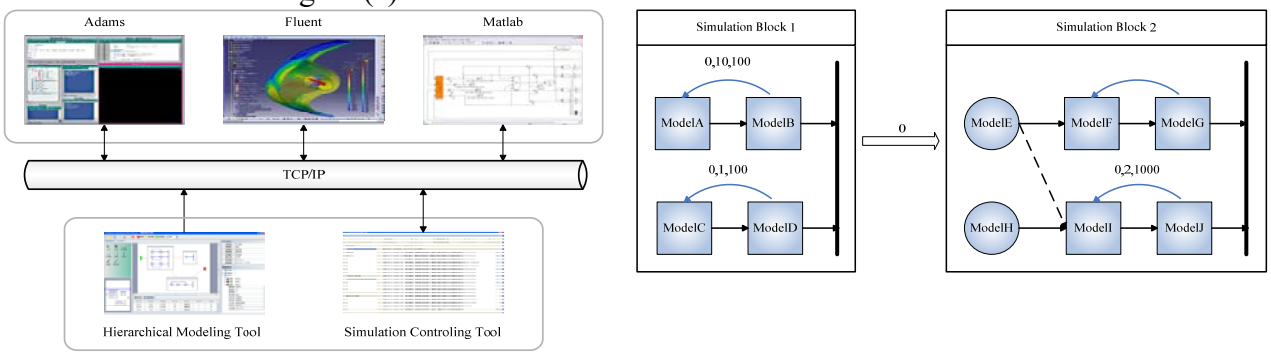
The remainder of this paper is structured as follows: We firstly introduce the background and related works. Then the hybrid synchronous algorithm is presented in detail. After that, we give the

analysis with experiment. Then we present a case study using CoSe to build an aerospace engineering simulation. Finally, our conclusion will be made with an indication of the future work.

Background and Related Works

Background.

In our previous article, a collaborative simulation framework is proposed, including the hierarchical modeling approach and a synchronous control algorithm [11]. Computer nodes are classified as a simulation control node and several computing nodes in our framework. Each computing node contains at least one computational model. Computational models on computing nodes pass output parameters, status information to the simulation control node though TCP/IP protocol. The simulation control node transmits loading command, initiating command and computing command to computing nodes. In the running process, the simulation control node receives output parameters and model status from the previous computational model, then passes input parameters to next computational model and schedules it to run. The collaborative simulation framework shows as Fig. 1. (a).



(a) collaborative simulation architecture (b) hierarchical modeling approach

Fig.1. Overview of the collaborative simulation framework

Collaborative simulation often combines hydromechanics, aerodynamics and mechanical control models to achieve united simulation. It involves various computational models with multidisciplinary knowledge, such as CAE model, engineering algorithm model and unified physical model. In order to implement the distributed developing, debugging and packaging and ensure those models fine-grained and high quality of reliability. Those models are built by experts from different research domains using different modeling languages and methods. For instance, CAE models are usually developed using Fluent, CFX, Ls_Dyna, Nastran, Abaqus etc. Engineering algorithm models are usually developed using Matlab, Fortran, C and so on. Unified physical models are usually developed using Modelica. Those models can be assembled hierarchically to construct a complex simulation system model.

The establishment of a precise system model is a hard and time-consuming work. To achieve the flexibility and generality, and to support constructing heterogeneous and hierarchical system model, we use graphic elements to represent computing models. The input-output relationships are described by the lines connecting between these elements, as shown in Fig.1. (b). According to the order of execution, the models are grouped into different simulation blocks. Each simulation block contains several loop branches whose starting time and end time is the same. Each loop branch contains multiple computing models. Because of the differences of executive mode, the model can be divided into once model or iterative model. Once model is the model that exits immediately after it finishes one step, represented by circle. Iterative model is the model that without exiting until to the end of this loop branch, represented by pane. We use one-dimensional unidirectional solid arrow to represent the scheduling relationship between computing models. One-dimensional dotted unidirectional arrow only indicates the input-output relationship while two-dimensional unidirectional solid lines represent the order of execution between different simulation blocks. Each loop branch is denoted by the starting time, time step and the end time. Vertical thick lines indicate the synchronization time. Because of the execution order between computing models, and the

difference of time steps, it is necessary to synchronize among loop branches, to ensure the validity of model execution and parameter transmission. The establishment of the entire simulation system model uses mouse dragging. It is so convenient as to improve the flexibility of building the complex system simulation model.

Related Work.

With the deep development of simulation applications, complex system oriented simulation technology changes to multi-domain from single domain gradually. After years of development and improvement, lots of simulation software and collaborative simulation specification have been emerged both at home and abroad, such as Adams, Fluent, Matlab, CoSim, HLA.

Adams, Fluent and Matlab is becoming more and more mature in the application of collaborative CAX, engineering computing technology, such as Adams is applied to the development of dynamic model, Fluent is applied to the development of fluid model, Matlab is applied to the development of numerical computational model. Although these tools can assistant to finish the simulation mission with single function, but they are lack of supporting for multi-domain complex system simulation. There are several simulation software that can achieve collaborative simulation though interface. For instance, Fluent fluid computational model work together with Matlab numerical computational model though interface. However, as a result of this collaborative simulation approach use point to point communication to achieve software interface, the simulation scale is limited and the flexibility and compatibility are the existing problems. It is difficult to deal with the problem of complex system simulation involving multidisciplinary interaction.

HLA has advantages in flexibility and versatility, which has become an important supporting technology of multidisciplinary collaborative simulation development. CoSim [12] has been developed for multidisciplinary simulation platform, which is used to resolve multi-domain collaborative simulation problem for complex productions. Collaborative modeling and simulation environment DEVS/HLA has been developed to research distributed heterogeneous simulation [13]. However, with the deep development of heterogeneous simulation application, the limitation of HLA-based distributed heterogeneous simulation is becoming obvious. The HLA specification originated from the military field, mainly suitable for process simulation and demonstration class, such as the combat process simulation and training, many research results and application experience of difficult to be used in other fields. Multi-domain collaborative simulation most involves lots of commercial software, such as Adams, Fluent and Matlab, which can’t provide interface yet, so it can not applied to HLA distributed simulation.

Hybrid synchronous algorithm

This paper further studies the synchronous control algorithm and a hybrid synchronous algorithm is proposed. The synchronous control algorithm includes two heterogeneous algorithms, one is a time-step synchronous control algorithm, the other is an event-based synchronous control algorithm.

Time step synchronous algorithm.

The collaborative simulation synchronous control algorithm assembles multithreads, a synchronization points array and a waiting queue to control the execution of the simulation system model. A thread for each loop branch will be created to schedule computational models to run. In order to reduce the times of synchronization, only when it runs to a synchronous model (i.e. the last model of a loop branch, such as ModelB, ModelD, ModelG and ModelJ shown in Fig.1 (b), the simulation needs to synchronize.

First, we establish a synchronous points array, where the array elements are the multiples of the second small time step of all the loop branches in a same simulation block. (E.g., suppose that a simulation block includes six loop branches, and the time steps of each loop branches are 3, 1, 3, 6, 12 and 1. The end time of all the loop branches in this simulation block is set to 120. Thus the elements of the synchronous points array are (3, 6, 9, 12...120)). Each element of the synchronous points array is a synchronization time. All the loop branches need to synchronize at each synchronization time.

Next, we set up a waiting queue. The element of this queue is the number of a loop branch whose current simulation time plus the time step exceeds the current synchronization time. We use the waiting queue to control the execution of each loop branch. When the simulation time advanced, if the simulation time of a loop branch in the waiting queue plus the time step less than the current synchronization time, the loop branch will be resume.

Last, we build synchronous thread, each circle branch i of simulation block corresponding to synchronous thread Si to handle the synchronization of a circle branch. When loop branch i is in synchronous queue and waiting queue, synchronous thread Si are in waiting state; when all of loop branches are in the synchronous and waiting queue, rouse all of the synchronous threads in synchronous queue. The synchronous protocol of collaborative simulation is shown in **Algorithm 1**.

Algorithm 1:time step synchronous algorithm

1 When the synchronous model in loop branch *i* finish step execution, compare the simulation time of loop branch *i* with synchronous time, and then continue (2, 3, 4, 5) to handle.
2 When the simulation time of loop branch *i* plus step stamp is less than synchronous time, advance the simulation time of loop branch *i*, restart the begin model of loop branch *i* (just like ModelA, ModelC, ModelF, ModelI Fig.1 (b) to execution.
3 When the simulation time of loop branch *i* plus step stamp equal to synchronous point, insert loop branch *i* to synchronous queue, and then suspend the synchronous thread corresponding to loop branch.
4 When all of loop branches in simulation block are in synchronous or waiting queue, arouse all of the synchronous threads in synchronous queue, clear all of the elements of synchronous queue and then advance synchronous point time, start the begin model of each loop branches to execution.
5 When advanced to synchronous point time, traversal waiting queue. If loop branch *i* of waiting queue plus step stamp equal to synchronous point, then insert loop branch *i* to synchronous queue, and then delete the loop branch *i* from waiting queue.

Event synchronous algorithm.

After simulation block k executes, it is needed to end iterative models in the simulation block, and end all of the threads corresponding to loop branches. Clear synchronous queue and waiting queue. If there is next simulation block, set new simulation block parameter and start the new simulation block to execute, otherwise, simulation end. The event synchronous mechanism of collaborative simulation is shown in **Algorithm 2**.

Algorithm 2: events synchronous algorithm

1 The simulation control module send executive end message to k iterative model.
2 Clear synchronous queue TB and waiting queue WTB;
3 Finish the threads corresponding to all of the loop branches;
4 Simulation block add 1; (k=k+1)
5 If k<SK then
6 Get the number of loop branches in simulation block k;
7 Else
8 Set the initial value to simulation time for each loop branch in simulation block k;
9 Simulation end;

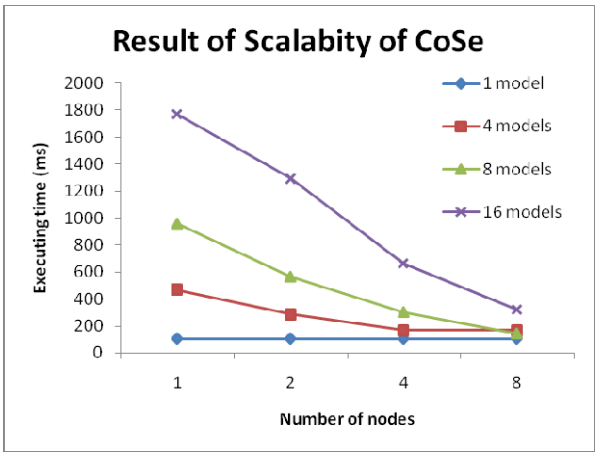
Test results

Our previous article tests the run performance evaluation of CoSe [11]. Here we mainly focus on the scalability of CoSe. We use the simplified models to simulate real models built by different developing tools such as Fluent or Matlab. To add the communication traffic among computing models, we set 15 parameters to transmit between two computing models on average. The data type of these parameters is double precision float. To add computation load into each model, we set each loop branch to be 100 iterations of model calculations. Also we can add different computation loads into model calculation process to vary the model's computation load. Here we use "the rate of computation to communication" to present the quantitative relationship between computation and communication load. At first, we run simulation several times to get the average communication

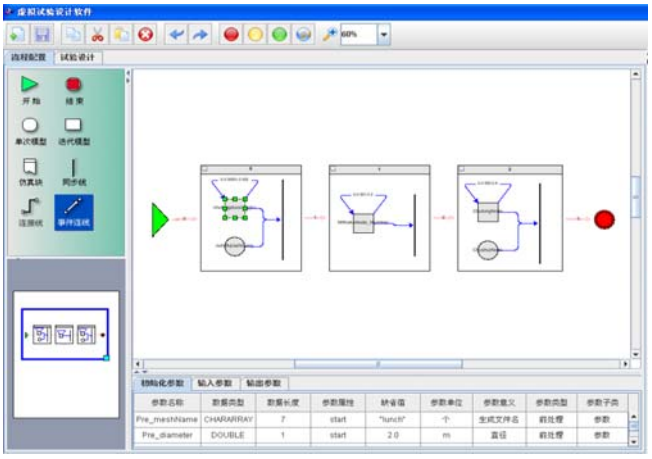
cost for a single iteration of model process.

All tests will run in a distributed computing environment composed of 9 computer workstations. Each workstation contains a 2.93 GHz Intel Core2 Duo CPU E7500 and 4G DDR3 1333 memories. The operating system is Windows XP. The collaborative simulation controlling module runs on one computer node, and the models run on other computer nodes.

Scalability is an important criterion for parallel approaches. In order to evaluate the scalability of CoSe under different number of heterogeneous models and computation nodes, we have performed the experiments with the same computational overhead. Here we set the rate of computation to communication as 1:1. There are 4 scenarios in this experiment. The number of models is set to 1, 4, 8 and 16 respectively. For each scenario, simulation is run on 1, 2, 4 and 8 computation nodes. The result in Fig.2 (a) shows that performance promotes almost linearly when more computation nodes are used.



(a) Scalability of CoSe



(b) Example of an aerospace engineering simulation

Fig.2. The experimental results and case study

We take the Out-Water simulation of an aerospace engineering project which involving the multidisciplinary field of fluid, dynamics and mathematics as the application background to testify the multi-domain complex system collaborative simulation. According to the demand of simulation, we build five domain computational model using Fluent, Adams and Matlib. They are in three simulation blocks and five loop branches. The collaborative simulation application is built as shown in Fig.2. (b).

By using hierarchical system model construction approach, the cost time for developing complex system collaborative simulation is decreased. By using collaborative simulation synchronous control protocol, the collaborative execution for multi-domain computational models supported by simulation control module is achieved, which provide data support for complex system analysis as soon as possible.

Conclusion

With the development of simulation application, complex system oriented collaborative simulation technology expended from single domain to multi-domain. This paper firstly studies the key points of collaborative simulation, then introduced a graphic heterogeneous and hierarchical modeling approach, and designed a hybrid synchronous control algorithm composed with time step and event-driven algorithms. We implemented a collaborative simulation environment (CoSe) for heterogeneous models based on distributed computation environment. The results of the experiment illustrate that CoSe offers good scalability with performance promotes almost linearly when more computation nodes are used. The case study shows that CoSe is able to support collaborative simulation development and execution for complex systems with multidisciplinary models.

In the future, we plan to design various experiments for our simulation framework on high performance computation environment, and utilize the results to improve it. Another interesting line of investigation would be the research on the efficiency of communication in our collaborative

simulation framework.

Acknowledgement

The authors appreciate the support from National Natural Science Foundation of China (no. 61170048) and Research Project of State Key Laboratory of High Performance Computation of National University of Defense Technology (no.201303-05).

References

[1] R.M. Fujimoto, Parallel and Distributed Simulation Systems, JOHN WILEY&SONS, INC, 2000.

[2] Xiao Tian-yuan, Fan Wen-hui. HLA based Integrated Platform for Collaborative Design, Simulation and Optimization [J]. Journal of System Simulation, 2008, 20(13): 3542-3547.

[3] Maghnouji R, Berbraeck A, et al. Collaborative simulation modeling: experiences and lessons learned [C]. Proceedings of the 34th Hawaii International Conference on System Sciences, Hawaii, U.S.A, 2001, 88-97.

[4] Van Beek D A. Multi-domain modeling, simulation, and control [C]. Proceedings of 4th International Conference on Automation of Mixed Processes. Dortmund, Germany, 2000, 139-146.

[5] Miller V T. Hybrid Heterogeneous Hierarchical Model for System Simulation [J]. International Journal in Computer Simulation, Vol 5, 1995.

[6] Schweiger, Wolfgang; Schoefmann, Werner; Vacca, Andrea. Gerotor Pumps for Automotive Drivetrain Application: a Multi Domain Simulation Approach [C]. SAE International Journal of Passenger Cars Mechanical Systems, 2011, 4(3): 1358-1376.

[7] Faruque, M.Q.; Dinavahi, V.; Steurer, M.; Monti, A.; Strunz, K.; Martinez, J.A.; Chang, G.W.; Jatskevich, J.; Iravani, R.; Davoudi, A.. Interfacing Issues in Multi-Domain Simulation Tools [J]. IEEE Transactions on Power Delivery, 2012, 27(1): 439-448.

[8] Chai Xudong, Li Bohu, Xiong Guangleng, et al. Research and Implementation on Collaborative Simulation Platform for Complex Product [C]. Computer Integrated Manufacturing Systems. 2002, 8(7):580-584.

[9] Deng Rui, Zhao Wen, Wang Weiping. Study on Collaborative Modeling Framework for Constructing Distributed Heterogeneous Simulation [J]. Computer Simulation 2005, 22(9):76-80.

[10] Manuel González, Javier Cuadrado. Mbslab: a New Collaborative Simulation Environment of Multi-body System Analysis [J]. Proceedings of ACMD 2004, 383-389.

[11] Feng Zhu, Yiping Yao, Tengfei Hu, Laibin Yan. A Collaborative Simulation Framework based on Distributed Computing Environment [C]. 2012 AASRI Conference on Modeling, Identification and Control, 2012, 3: 571-576.

[12] Tingyu Lin, Xudong Chai, Bohu Li. Top-level modeling theory of multi-discipline virtual prototype [J]. System Engineering and Electronics, 2012, 3: 425-437.

[13] Han Shou-peng, Zhang Peng, Qiu Xiao-gang, Huang Ke-di. Research of Time Advancing Mechanism for DEVS/HLA Distributed Simulation [J]. Journal of System Simulation, 2008, 20(7): 1744-1748.