Model Checking of Software Development in Distributed Animation Rendering System

Zhiguo Hong[1,a], Yongbin Wang[1,2,3], Minyong Shi[1]

[1]School of Computer, Faculty of Science and Engineering, Communication University of China, Beijing 100024, China

[2]Key Laboratory of Acoustic Visual Technology and Intelligent Control System, Ministry of Culture, Beijing 100024, China

[3]Beijing Key Laboratory of Modern Entertainment Technology, Beijing 100024, China

[a]hongzhiguo@cuc.edu.cn

Keywords: model checking; animation rendering; SMV; the state machine

Abstract: Distributed animation rendering systems are used to speed up of rendering i.e the computational-intensive phase of producing animation. How to develop softwares with high availability and efficiency is a hot topic in distributed animation rendering systems. In this paper, based on model checking method for system modeling, three modules (i.e. rendering management server, rendering nodes and storage system) are modeled and analyzed. Firstly, by analyzing the system's typical infrastructure, messages-driven finite state machines of such three modules are studied. Then, the properties of such three modules are described using CTL(Computational Tree Logic). Furthermore, safety and livenesses are verified with SMV (Symbolic Model Verification). The methodology of this paper offers important theoretical guide to software development.

Introduction

With the improvement of computers and clusters' performance, distributed animation rendering systems containing various heterogeneous computing nodes are increasingly becoming an efficient solution to overcome the bottleneck of rendering. For instance, Distributed Rendering Environments (DREs) [1, 2, 3] are used to accelerate the process of rendering by interconnecting heterogeneous PCs and clusters. The phase of rendering is computation-intensive, for it would cost a lot of CPU, RAM resource. By processing 3D models in scenes via rendering computation, AVI files or sequences of images are generated. Therefore, 3D film producers or animation fans can conduct further production on the basis of those rendering output. Two factors they care about are the quality of rendering effect and time delay of rendering. For a typical distributed animation rendering system, softwares are programmed to satisfy users' demand of rendering. Obviously, the performance and efficiency of softwares providing rendering services have great impact on users' experiences. Consequently, how to develop softwares with high availability and efficiency is an important issue to be solved in distributed animation rendering systems. Commonly, white box testing and black box one are utilized as effective methods to find bugs of software. Undoubtedly, such testing methods can detect faults of programs. However, due to the partial coverage of all paths some hidden bugs are hardly invoked sometimes. Additionally, it would cost a lot of time to discover all bugs by testing on the scope of full coverage.

On the basis of mathematic theory, model checking method has been widely used to guarantee the time sequence logic between the modules. Subsequently, it offers theoretical foundation of improving the robustness and usability of softwares. Model checking is a verification technique for concurrent systems with finite states and it is a very important method in formal verification. In 1980s, Clarke and Emerson et al. proposed the CTL (Computation Tree Logic) to describe the properties of concurrent systems, designed the algorithm to detect whether a finite state system satisfies a given CTL formula, and implemented a prototype system named EMC[4]. This work has offered a new way for the automatic verification of concurrent systems. It has become a hot

research topic in computer science in recent years[5]. At present, the model checking technique is applied to the hardware design [6]and software development [7] etc.
SMV(Symbolic Model Verification) [8] tool is a popular model testing and analysis software, which is used to describe the specification language of the finite state concurrent system, and supports the verification of the related properties of the model.

The rest of this paper is organized as follows. In section 2, the software architecture for typical distributed animation rendering system is analyzed. In section 3, by focusing on the system's three key modules (i.e. rendering management server, rendering nodes and storage system), messages-driven finite state machines of these modules are studied. Additionally, the description of these modules' properties are given using CTL(Computational Tree Logic). Furthermore, safety and livenesses are verified with SMV. Section 4 concludes this paper.

Software Architecture for Distributed Animation Rendering system

High availability and user-friendly interface are key principles of designing distributed animation rendering systems. From the viewpoint of user-friendly interface, Browser/Server-based portal is developed to offer the status of current rendering task for users. From the prospect of rendering system's high availability, socket-based communication modules are programmed to warrant the connections among render portal, rendering management server and rendering nodes. Therefore, we use the infrastructure to specify the flows of control and data as Fig. 1.

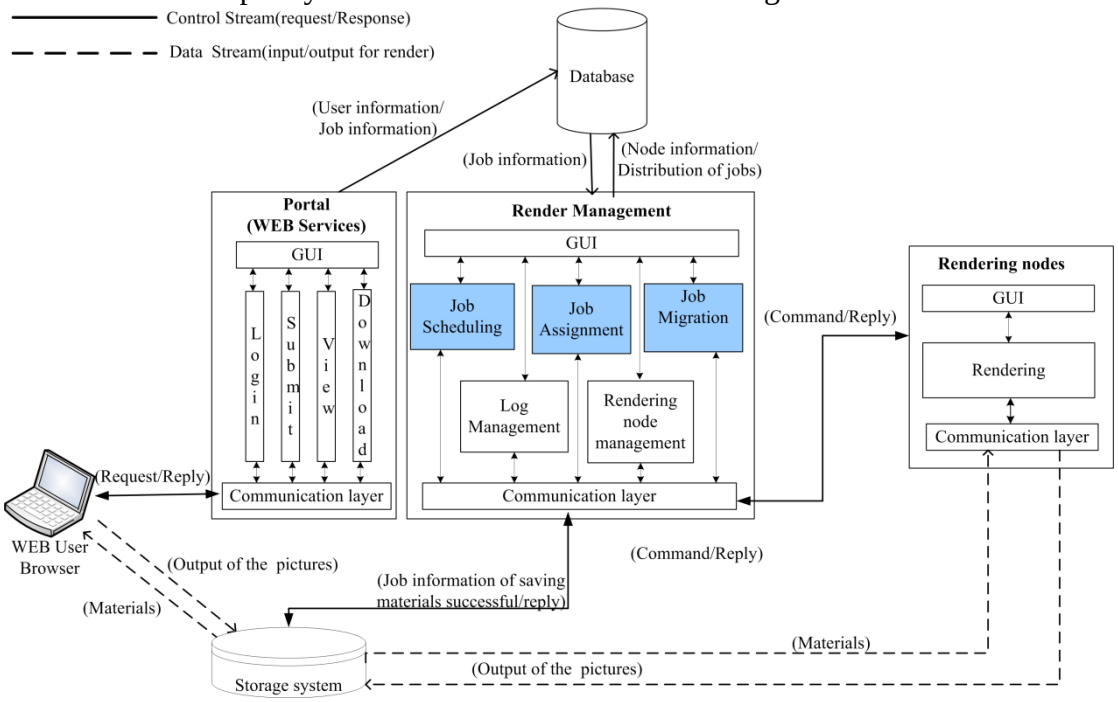

Fig. 1 Software architecture for distributed animation rendering system

As Fig. 1 illustrates, there exist two kinds of streams which are control stream and data one. In order to distinguish from each other, the streams of control and data are plotted with solid line and dotted line respectively. The control stream is implemented with request/response mechanism. Users can visit the web portal to login, submit their 3D scenes, view the status of rendering and download output files. Meanwhile, the related information of users and jobs are stored in database. Storage system is in charge of saving 3D scenes, corresponding maps and render output files. Sometimes, the volume of such media files is becoming larger with the increase of users' jobs. Therefore, FTP (File Transfer Protocol)-based storage system are constructed to offering sending and receiving data with fast speed and high reliability. Rendering management server functions as the manager of rendering jobs and deals with job scheduling, job assignment and job migration. It sends instructions to rendering nodes via socket-based communication.

Model Checking

Thus, the modules of rendering management server, rendering nodes and storage system are key components in distributed animation rendering systems. To code the related applications to offer rendering services with high availability, the temporal constraints among different modules should be considered in the design procedure. For instance, some processes are executed in parallel. However, some processes have to be synchronized under certain condition. Runtime errors would probably be triggered without such constraints.

By taking three modules (i.e. rendering management server, rendering nodes and storage system) of distributed animation rendering system as research objects, we investigate SMV tool-based modeling and analysis. Finite state machines of these modules driven by messages are constructed as Fig. 2, Fig. 3 and Fig. 4 respectively.

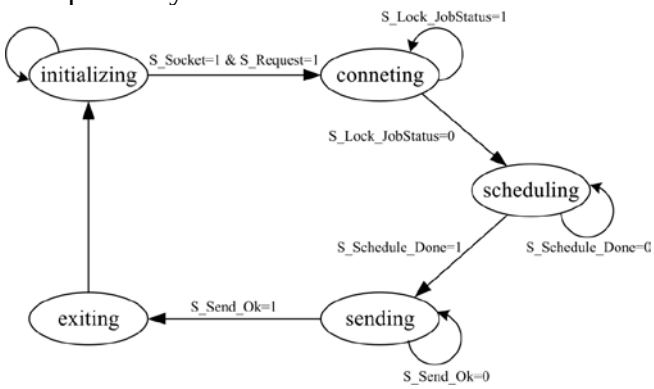

Fig. 2 Finite state machine of rendering management server module driven by messages

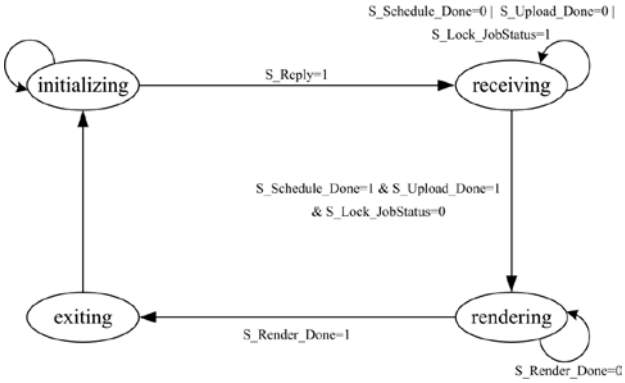

Fig. 3 Finite state machine of rendering node module driven by messages

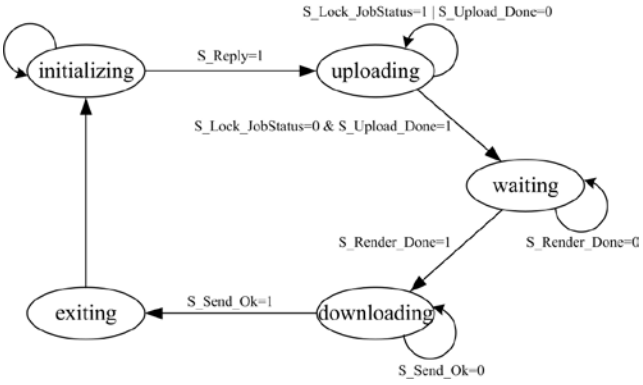

Fig. 4 Finite state machine of storage system module driven by messages

Definition of Finite State Machines' Messages. As Fig.2 shows, the set of rendering management server module's messages is {initializing, connecting, scheduling, sending, exiting}. As Fig.3 demonstrates, the set of rendering node module's messages is {initializing, connecting,

scheduling, sending, exiting}. As Fig.4 signs, the set of storage system module's messages is {waiting, initializing, rendering, exiting}.

Analysis of Processes Driven by Messages. From Fig.2, Fig.3 and Fig.4, we can observe that different states in each module can be transferred to each other when some condition satisfies. Otherwise, except for "exiting" states, other states could probably be in the status of self-waiting.

The related messages responding to rendering management server module, rendering node module and storage system module are given as follows.

(1) S_Socket: boolean, signing the status of socket-based communication;

(2) S_Request: boolean, signing whether there exists any request from rendering nodes;

(3) S_Lock_JobStatus: boolean, signing the semaphore of exclusive operation on information of job's status. "S_Lock_JobStatus=1" means one of three modules has gained the right of accessing information of job's status. Other modules cannot update job's status until "S_Lock_JobStatus=0", i.e. the right of exclusive operation is released.

(4) S_Schedule_Done: boolean, signing the completion of scheduling process. When the process of scheduling is done, S_Schedule_Done is set to be "true" or "1".

(5) S_Send_Ok: boolean, signing the result of downloading files from storage system. When the result is success, S_Send_Ok is set to be "true" or "1".

(6) S_Reply: boolean, symbolizing whether rendering management server replies rendering nodes according to their requests;

(7) S_Upload_Done: boolean, symbolizing the result of uploading scene files to storage system. When the result is success, S_Upload_Done is set to be "true" or "1".

(8) S_Render_Done: boolean, symbolizing the progress of rendering by rendering nodes. When the progress reaches 100%, S_Render_Done is set to be "true" or "1".

CTL Properties of the software. In order to conduct SMV-based model checking on the software of distributed animation rendering system, the related specifications of CTL properties are given as follows.

(1) safety1 : assert G ~(M_RMS_Process.State=scheduling & M_RN_Process.State=rendering);

(2) Liveness1 : assert G (M_RMS_Process.State=waiting -> F M_RN_Process.State=waiting);

(3) Liveness2 : assert G (M_RMS_Process.State=scheduling -> F M_SS_Process.State=downloading);

Here M_RMS_Process, M_RN_Process and M_SS_Process represent the modules of rendering management server, rendering node and storage system respectively. "safety1" states the safety. "Liveness1" and "Liveness2" demonstrate the liveness. Besides, "G", "F", "->" and "&" are symbols of CTL. "AG" expresses all the states in all paths. "AF" illustrates the final state in all paths. "->" formulates the logic implication, i.e. some conclusion can be inferred.

The detailed meanings of these properties are described as follows.

(1) "safety1" means that the state of M_RMS_Process being "scheduling" and that of M_RN_Process being rendering won't occur at the same time. Consequently, it guarantees exclusive use of related shared resource and avoid the related abnormal occurrence of runtime.

(2) "Liveness1" means that the state of M_RMS_Process being "waiting" would eventually lead to the state of M_RN_Process being "waitng".

(3) "Liveness2" means that the state of M_RMS_Process being "scheduling" would ultimately result in the state of M_RN_Process being "downloading".

SMV-based verification. Based on the finite state machines as seen in Fig.2, Fig.3 and Fig. 4, the SMV-formatted codes are written and implemented by SMV tool with the version of "experimental release 10-11-02P46" under the operating system of "Windows 7 Home 64bits". The results of verification are shown in Fig. 5. We can observe that properties of "safety1", "Liveness1" and "Liveness2" are verified to be "true". Therefore, it manifests the safety and correctness of logic constraints among the modules of rendering management server, rendering node and storage system in distributed animation rendering system.

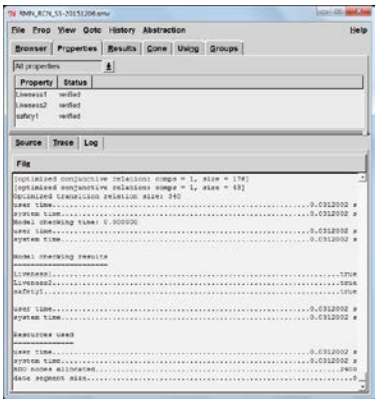

Fig. 5 The results of SMV-based model checking

Conclusion

In this paper, model checking technology has been introduced to design software in distributed animation rendering system. Three key modules (i.e. rendering management server, rendering nodes and storage system) are modeled and investigated. Firstly, by analyzing the system's typical infrastructure, messages-driven finite state machines of such three modules are studied. Then, the properties of safety and liveness are defined with the specification of CTL. Furthermore, codes are implemented by SMV tool and the properties are verified. Thereby, it can be concluded that the three modules satisfy the constraints of correctness and safety.On the basis of the work, further model checking-based methodology can be conducted to research software problems of integrating different modules in distributed animation rendering system.

Acknowledgements

This work is supported by the National Science and Technology Supporting Program (Project numbers: 2012BAH37F02), the Excellent Young Teachers Training Project (the second level, Project number: YXJS201508), Engineering Project of Communication University of China (Project number: 3132015XNG1504).

References

[1] S. L. Gooding, L. Arns, P. Smith, et al.: Implementation of a Distributed Rendering Environment for the TeraGrid, Proceedings of IEEE Challenges of Large Applications in Distributed Environments(CLADE). Paris, France, (2006), p.13-21.

[2] A. Chong, A. Sourin and K. Levinski: Grid-based computer animation rendering, in Proc. of the 4th international conference on computer graphics and interactive techniques. Australasia, Southeast Asia, (2006), p.39-47.

[3] C. Glez-Morcillo, D. Vallejo and J. Albusac et al. A New Approach to Grid Computing for Distributed Rendering. Proceedings of 2011 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC 2011), (2011), p. 9-16.

[4] E.M.Clarke, E.A. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications [C]. In ACM Transactions on Programming Languages and Systems, vol. 8, no.2(1986) , p.244- 263.

[5] LIN Hui-min,ZHANG Wen-hui. Model Checking: Theories, Techniques and Applications [J], Acta Electronica Sinica, vol. 30, no.12A(2002), p.1907-1912. (in chinese).

[6] YANG Ying, WANG Yong-bin, YAN Jian, YU Ning. Develop the Digital Hard-disk Using SMV[J]. JOURNAL OF SYSTEM SIMULATION, vol. 17, no.z1(2005): p.190-192. (in chinse).

[7] Ma Ling. Distributed On-line Animation Rendering and Model Checking[D].Master's thesis of Communication University of China, June 2010. (in chinse).

[8] K L McMillan. The SMV language [M]. Cadence Berkeley Labs. 1999.