Research Journal of Applied Sciences, Engineering and Technology 4(21): 4453-4457 , 2012
ISSN: 2040-7467

Submitted: May 01, 2012 Accepted: June 08, 2012 Published: November 01, 2012

Towards the Availability of the Distributed Cluster Rendering System: Automatic Modeling and Verification

[1,2]Kemin Wang, [1]Zhengtao Jiang, [1]Yongbin Wang, [3]Youshan Yang and [4]Wei Jiang
[1]School of Computer Science, Communication University of China, Beijing 100024, China
[2] Informatics Department, Technical University of Denmark, 2800 Lyngby, Denmark
[3] Television Station of HeiLongJiang, Haerbin 150090, China
[4]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731

Abstract: In this study, we proposed a Continuous Time Markov Chain Model towards the availability of n-node clusters of Distributed Rendering System. It's an infinite one, we formalized it, based on the model, we implemented a software, which can automatically model with PRISM language. With the tool, whenever the number of node-n and related parameters vary, we can create the PRISM model file rapidly and then we can use PRISM model checker to verify ralated system properties. At the end of this study, we analyzed and verified the availability distributions of the Distributed Cluster Rendering System while the node number-n varying under different repair modes.

Keywords: Availability model, CTMC, distributed cluster rendering system, infinite markov chain, PRISM

INTRODUCTION

PRISM (PRISM, 2010) is a probabilistic model checker, which can support the formal verification of stochastic systems. In study Wang *et al*. (2011), we have completed the formal modeling and verification of an availability model of a Distributed Rendering System, for a one-node system and a two-node system. But when the node number increases, it is inefficient for the modeling, for there are so many state transitions need to describe, how to model it automatically for an arbitrary n-node model? This is a modeling problem of model checking towards a certain Infinite State Markov Chain.

PRISM cannot support the model checking of Infinite State Markov Chain. Towards the availability model of an arbitrary n-node Distributed Rendering System (DRS), we proposed a model suitable to be described by PRISM language, which is all labeled with numbers and then realized an automatic modeling tool, with the tool, we can rapidly create the PRISM model file, even if the node number-n and related parameters varying and then, we can analyze and verify related system properties on the models we have created. Hong *et al*. (2011) shows Markov-based availability analysis of clusters for distributed rendering system. Hahn *et al*. (2009) have a research of the INFAMY: an infinite - state markov model checker, in 21 st International Conference on Computer Aided Verification. Hahn *et al*. (2009) study the time-bounded model checking of infinite-state continuous-time markov chains. Zhang *et al*. (2008) study the time-bounded model checking of infinite-state continuous-time markov chains. Remke *et al*. (2005) have a research of the model checking infinite-state Markov chains. in 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Remke and Haverkort (2007) study the CSL model checking algorithms for infinite-state structured Markov chains. Ben Mamoun and Pekergin (2008) study the model checking of infinite state space Markov chains by stochastic bounds.

This study is based on an Availability Model of Distributed Rendering System. The model is extendable, we formalized it and then implement an automatic modeling tool. Through the tool, we can generate the PRISM model file of the n-node system, even if n varies; and then we can use PRISM model checker to verify related system properties. Moreover, we analyzed its steady state possibility, i.e., the availability. Based on our automatic tool and the availability model of the system, we propose a rapid method for analyzing the availability of Distributed Rendering System, the tool chains. This study solved an automatic modeling problem of a certain Infinite State Markov Chain.

Corresponding Author: Kemin Wang, School of Computer Science, Communication University of China, Beijing 100024, China

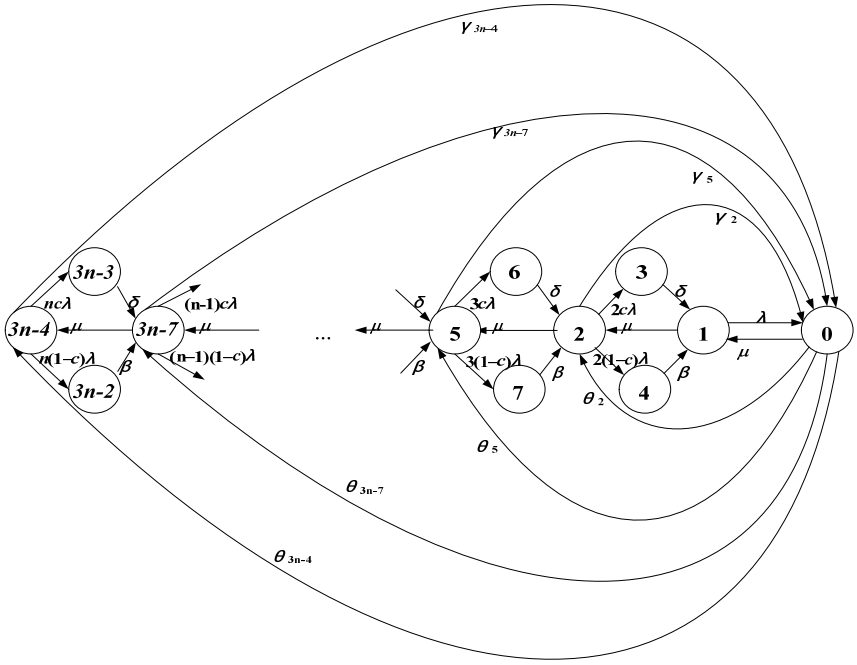

Fig. 1: N-node availability model (CTMC, as n≥ = 3)

METHODOLOGY

Availability model of the distributed rendering system and its formula: Based on the Availability Model of the Distributed Rendering System proposed in study Hong *et al*. (2011), which is a Infinite States Continuous Time Markov Chain, we proposed a new one which is suitable for being described with PRISM language, as shown in Fig. 1- an n-node system's Availability Model-an CTMC (n≥ = 3). PRISM uses number to label the states, but the model proposed in study (Hong *et al*., 2011) is not all labeled by numbers, as ours is, however actually they are the same model.

Here, we will introduce the model simply and some details you can reference these studies (Wang *et al*., 2011; Hong *et al*., 2011).

Suppose the node number of the Rendering System is n:

- The state number of the model is 3 * n-2 and the Main States are labeled as: 0, 1, 2, 5 ,…, 3 * n-4, which represents the available nodes number of current Distributed Rendering System are 0, 1, 2, 3, 4, …., n, respectively.
- Totally Damage Mode and Totally Repair Mode: For a cluster with n-node, when the k nodes of it are totally damaged at the same time, the cluster can reboot, then after some time, can recover available via reset operations. As, γi (i = 2, 5, …, 3*n-7, 3*n-4, n≥ = 3) denotes the totally damage rate of i nodes, θj (j = 2, 5,…, 3*n-7, 3*n-4, n≥ = 3) denotes the totally repair rate of j nodes
- Repair One-by-One Mode, is to recover the availability of one node at one time, the repair rate is mu
- Totally Repair Mode means Repair All-by-Once.
- The possible damages, which can been forcast, make the state transfer from（3*n-4） to （3*n-3） and then to （3*n-7）, the transfer rate as shown in Fig. 1.
- The damages, which have not been forcast, make the state transfer from（3*n-4） to（3*n-2） and then to（3*n-7, the transfer rate as shown in Fig. 1. No matter the damages can be forecast or cannot be forecast, we need to repair the system, through reboot and reconfigure, to recover the availability of the whole system

For the availability model we proposed in which all states are labelled with integer numbers, as shown in Fig. 1, we can formulate it as follows:

Definition 1: based on the availability model for the Distributed Cluster Rendering System, as shown in Fig. 1, we define the expression P(n) as:

$$P(n) = \sum_{m=1}^{n} Q(m)$$

For which, Q (m) is an expression formula related to the transitions of the Main State m, which will be added in to P (m-1):

- when n = 1, P (1) = Q (1), which is the transition formula of a one-node availability model and Q (1) = q = 0-> mu: (q' = 1) + q = 1> lamda: (q' = 0)
- when n=2, P (2) = Q (1)+Q (2), which is the transition formula of a two-node availability model and;

Q (2) = q = 1 ->mu: (q' = 2)
+ q = 2 -> to_c_lamda: (q' = 3)
+ q = 3 -> diata: (q' = 1)
+ q = 2 -> to_1_c_lamda: (q' = 4)
+ q = 4 -> beta: (q' = 1)
+ q = 2 -> gama_2: (q' = 0)
+ q = 0 -> thita_2: (q' = 2)

when n> = 3;

Q(m) = A1(m) + A2(m) +A3(m) + A4(m)
+A5(m) + A6(m) + A7(m)

and;

A1(m) = (q = 3* m-7) -> mu: (q' = 3* m-4)
A2 (m) = (q = 3* m-4) ->m_c_lamda:(q' = 3* m-3)

A3(m) = (q = 3*m-3) ->diata: (q' = 3* m-7)
A4 (m) = (q = 3*m-4) ->m_1_c_lamda: (q' = 3* m-2)
A5 (m) = (q = 3* m-2) ->beta: (q' = 3* m-7)
A6 (m) = (q = 3* m-4) ->gama_3m_4: (q' = 0)
A7 (m) = (q = 0) ->thita_3m_4: (q'= 3* m-4)

- q is a variable which labels the current state and q' labels the next state
- + denotes the simple connection of the states
- -> denotes the transition action of state
- mu,lamba,to_c_lamda,diata,to_1_c_lamda,beta, gama_3m_4, thita_3m_4, etc., are the related variables which represent the transfer rate between states, actually they can be array elements

Using Mathematical Induction Method can easily prove the correctness of the formula P (n). The proof is omitted here.

Based on the formula P (n), we can implement a tool which can refine the model, then for an arbitrary n, we can refine the formula P (n), create the corresponding PRISM model file and then we can use PRISM model checker to verify the related properties of the system.

The windows interface of the automatic modeling tool: The automatic modeling tool is implemented based on .net framework 3.5, the programming language is c++. net and the application is a windows form application, the window interface as shown in Fig. 2. When the tool is in runtime, we can fill the number of nodes of the distributed system, as shown in Fig. 2, Nodes of Distributed System, Repair Mode and a series of parameters and then can press 'Create Model' to generate the PRISM model file.

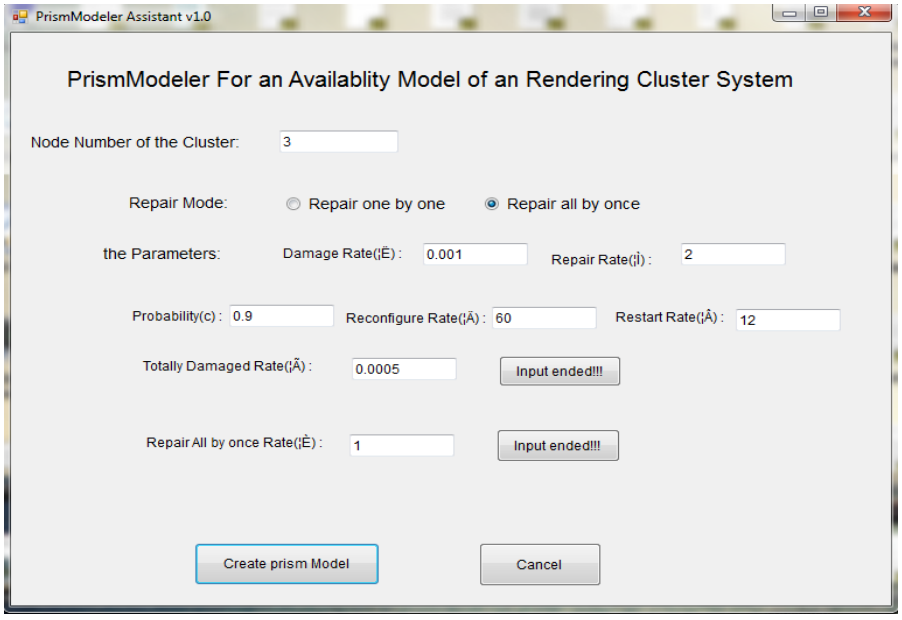

Fig. 2: The windows interface of the tool

Table 1: The availability distribution while n varies under different modes

Node no	Repair one by one mode	Repair totally mode
1	0.9995002499	0.9995002499
2	0.9997496041	0.9998331747
3	0.9997500169	0.9998750085
4	0.9997500233	0.9999000093
5	0.9997500291	0.9999166764
6	0.9997500349	0.9999285814
7	0.9997500407	0.9999375102
8	0.9997500465	0.9999444548
9	0.9997500524	0.9999500105
10	0.9997500582	0.9999545561
50	0.9997502893	0.9999902075
100	0.9997505743	0.9999950610
300	0.9997516702	0.9999983503
500	0.9997526945	0.9999990132
800	0.9997540912	0.9999993868
1000	0.9997549239	0.9999995113

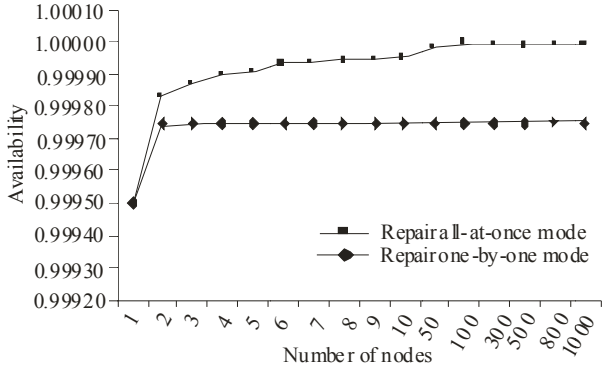

Fig. 3: The availability distribution under different modes

A 3-node showcase of the model which generated by the tool as shown below, for simplicity, we just show the description of the transitions of the states.

ANALYSES AND VERIFICATIONS

we can use the automatic modeling tool to create the model file via inputing some parameters and then we can use PRISM model checker to analyze the availability of the current model.

Suppose, the parameters' values are as follows:

$$\lambda = 0.001,\ \mu = 2,\ c = 0.9,\ \delta = 60,\ \beta = 12$$

For convenience, we iniate the two arrays- gama and thita with 0.0005 and 1.

So with the same series of parameters, we can analyze the availability distributions while the number of nodes-n increases.

As shown in Table 1, while node number-n increases, n = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 50, 100, 300, 500, 800, 1000, with the corresponding availability. From Fig. 3, we can easily get, for a certain system, the availability under Repair Totally Mode is higher than

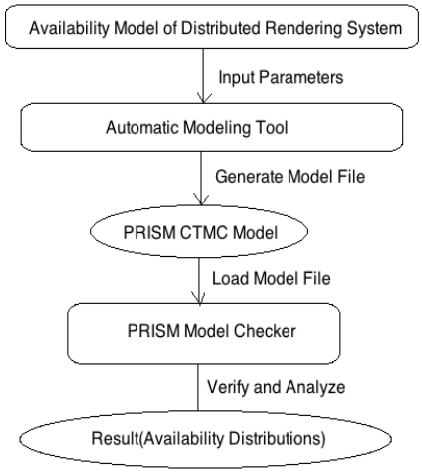

Fig. 4: Tool chains for analyzing the availability of the DRS

that under Repair One-by-One Mode; and, no matter the repair mode is, the availability will increase while node number-n increase.

The result is exactly same with the general knowledge, when the system under Repair Totally Mode, it can recover all of the damaged nodes at one time, so the available possibility definitely higher than another mode; and when n increases, for every node's damaging possibility is same, so the system with one or more nodes which node can be available, the possibility will increase.

CONCLUSION AND FUTURE WORK

This study is based on an Availability Model of Distributed Rendering System, the model is extendable, we formalized it and then implement an automatic modeling tool. Through the tool, we can generate the PRISM model file of the n-node system, even if n varies; and then we can use PRISM model checker to verify related system properties, in this study, we analyzed its steady state possibility, i.e., the availability. Based on our automatic tool and the availability model of the system, we propose a rapid method for analyzing the availability of Distributed Rendering System, the tool chains as shown in Fig. 4.

This study solved an automatic modeling problem of a certain Infinite State Markov Chain, the certain infinite one is also equipped with many different parameters, although INFAMY (Hahn *et al*., 2009) and Geobound (Zhang *et al*., 2008) are excellent model checkers towards infinite state markov chain, for our certain one, it does not fit very well.

However, our method is restricted by the PRISM model checker, when we verify the availability of a 1500-node model, PRISM will violate when parsing the model file. So, next work, we will study on the model checking algorithm towards the infinite state markov chain, especially for the Continuous Time Markov

Chain, just as the works of Remke and Haverkort (2007) and Remke and Haverkort (2007).

ACKNOWLEDGMENT

This study is supported partly by the scholarship of CSC (China Scholarship Council) and CUC (Communication University of China). Particularly, supported by National Natural Science Foundation of China (61103199, 61003244, 60903180, 61072140), Beijing Municipal Natural Science Foundation (4112052), Engineering Program Project of CUC, the research project of the State Administration of Radio Film and Television of China.

REFERENCES

Ben Mamoun, M. and N. Pekergin, 2008. Model checking of infinite state space Markov chains by stochastic bounds. Proceedings of the 15th International Conference on Analytical and Stochastic Modeling Techniques and Applications, (ASMTA '08), Springer-Verlag Berlin, Heidelberg, pp: 264-278, ISBN: 978-3-540-68980-5.

Hahn, E.M., H. Hermanns, B. Wachter and L. Zhang, 2009a. INFAMY: An infinite-state markov model checker. Proceeding of 21st International Conference on Computer Aided Verification (CAV '09), Springer, of LNCS, 5643: 641-647.

Hahn, E.M., H. Hermanns, B. Wachter and L. Zhang, 2009b. Time-bounded model checking of infinite-state continuous-time markov chains. Fundamenta Informaticae, 21(2001): 1001-1027.

Hong, Z.G., W. Yongbin and S. Minyong, 2011. Markov-based availability analysis of clusters for distributed rendering system. J. Commun., 2011-04, DOI: CNKI:SUN:TXXB.0.2011-04-022.

PRISM manual, 2010. Retrieved from: http://www.prismmodelchecker.org/manual/.

Remke, A., B.R. Haverkort and L. Cloth, 2005. Model checking infinite-state Markov chains. I Proceedings of the 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, (TACAS '05), Springer-Verlag Berlin, Heidelberg, pp: 237-252, ISBN: 3-540-25333-5 978-3-540-25333-4.

Remke, A. and B.R. Haverkort, 2007. CSL model checking algorithms for infinite-state structured Markov chains. Proceedings of the 5th International Conference on Formal Modeling and Analysis of Timed Systems, (FORMATS '07), Springer-Verlag Berlin, Heidelberg, pp: 336-351, ISBN: 3-540-75453-9 978-3-540-75453-4.

Wang, K. and Y. Wang, 2011. A stochastic model checking method for the usability of clusters about distributed rendering system. International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011.

Zhang, L., H. Hermanns, E.M. Hahn and B. Wachter, 2008. Time-bounded model checking of infinite-state continuous-time markov chains. Proceeding of 8th International Conference on Application of Concurrency to System Design (ACSD'08), Department of Computer Science, University of Oxford, pp: 98-107.