Software Rejuvenation in Cloud Systems Using Neural Networks

Chapram Sudhakar
Department of CSE,
National Institute of Technology, Warangal, INDIA.
chapram@nitw.ac.in

Ishan Shah, T. Ramesh
Department of CSE,
National Institute of Technology, Warangal, INDIA,
ishan.shah@gmail.com, rmesht@nitw.ac.in

***Abstract*—Virtual Machine Monitor (VMM) is very important for the cloud and data center environment. VMM runs continuously for a long time and hence encounters the problem of software aging. VMM experiences failure because of software aging. In order to prevent the VMM failure caused by software aging, a proactive fault management approach called software rejuvenation is used. There are various software rejuvenation approaches existing in literature that can be broadly categorized into two categories namely model based approaches and measurement based approaches. Time to failure is predicted in measurement based approaches by monitoring the resource usage statistics. There can be any non-linear relationship between resource usage statistics and the time to failure. Such a non-linear function can be approximated using Artificial Neural Networks (ANN). The change in the value of attributes of resources is given as input to ANN and new value of time to failure is generated as output. Experiments shows that if there is some pattern in the arrival and departure of the VMs, then the prediction is more accurate.**

Keywords—Software Aging, Virtual Machine Monitor, Artificial Neural Networks, Cloud Computing

I. INTRODUCTION

In software engineering, software aging refers to progressive performance degradation or a sudden hang/crash of a software system due to exhaustion of operating system resources, fragmentation, and/or accumulation of errors [16]. A Virtual Machine (VM) is a software implementation of a computing environment in which an operating system (OS) or a program can be installed and run. Typically a virtual machine emulates a physical computing environment. VMM processes the requests generated by VM for CPU, memory, network, hard disk and other resources. VMM runs continuously for a long time and encounters the problem of software aging.

Software rejuvenation is a cost effective technique for dealing with software faults. It consists of a proactive fault management technique. Its aim is to clean up the system internal state in order to prevent the occurrence of more severe faults, such as crashes or software malfunctioning. Cleaning up the internal state includes removal of accumulated error conditions, free the resources of the operating system, garbage collection, flushing operating system kernel tables, reinitializing the internal data structures etc. [14].

Rest of the paper is organized as follows. In Section II, related work is presented. In Section III, the proposed neural network based approach is described. Experimental setup and results are discussed in Section IV and Section V respectively. Conclusions and future work are given in Section VI.

II. RELATED WORK

The software aging problem can be handled either by creating analytical models or analyzing the software based on measurements. In measurement based approach, the attributes related to resources, that are subject to software aging (aging indicators) are monitored. ANN is used to produce adequate forecasting of swap space, physical memory, response time of the apache web server [1] and in general can be used to predict any non-linear function.

The following subsections describes different software rejuvenation approaches.

A. Model Based Approach

In this approach, a model is created with some assumptions about the system. The model is analyzed to find the appropriate time for rejuvenation. Generally the models adopted are Markov-based and Petri-nets based [2,4,9,14] models. Using model based approaches, experiments are conducted on the telecommunication and transactional systems[14].

B. Measurement Based Approach

In this approach, analysis of the software aging is performed based on the measurements of the real systems. In [5], Huang et al. used measurement based approach for software rejuvenation. In this approach, system variables, called aging indicators, are monitored directly to predict the failure. Aging indicators can provide the onset of the software aging. Several approaches have been proposed to perform prediction of time to failure [1,3,6,8,10,11,12] based on aging indicators.

In the present work, ANN is used to produce the time to failure of the VMM, which can be used in machine independent way. Two types of neural networks are created in which, one takes single time slot as an input and another takes a window of time slots.

978-1-4799-7683-6/14/$31.00©2014 IEEE

III. Proposed Neural Network based Approach

This section explains in detail about the input data set generation, creation and training of the neural network.

A. Input Data Generation

Six different resources are considered here, namely number of processes, number of threads, CPU utilization, number of open TCP connections, available physical memory and available swap space. While generating the data, following points are considered.

- The time attribute is considered as a discrete attribute.
- There are two events regarding virtual machine namely arrival event and departure event.
- At a particular time, only one of the two events, arrival or departure, can be triggered.
- There is a one to one mapping between the arrival event and the VM.
- There is a one to one mapping between the departure event and the VM.
- The probability of occurrence for either of the two events is same.
- Resources are allocated to VM, when it arrives. These resources exist until the departure event occurs for that VM. Between the arrival event and the departure event of a VM, the allocated resources to that VM remains constant.
- Each VM can have 1 to 10 application processes,1 to 25 threads, 500MB to 2GB physical memory, 1GB to 4 GB swap space and 2 to 5 open TCP connections.
- The aging rate caused by a VM remains constant throughout the execution of that VM. The aging rate of two VMs need not be same.
- After the departure event of VM, it will not contribute to the aging of the VMM.
- The aging phenomenon of the VMM follows the Weibull failure distribution.

Following flow chart Fig. 1, explains the steps of generating the resource usage statistics.

B. Training of the Neural Network

In order to implement the neural network, ENCOG machine learning library is used. BasicNetwork class of ENCOG library is extended to implement feed forward neural network.

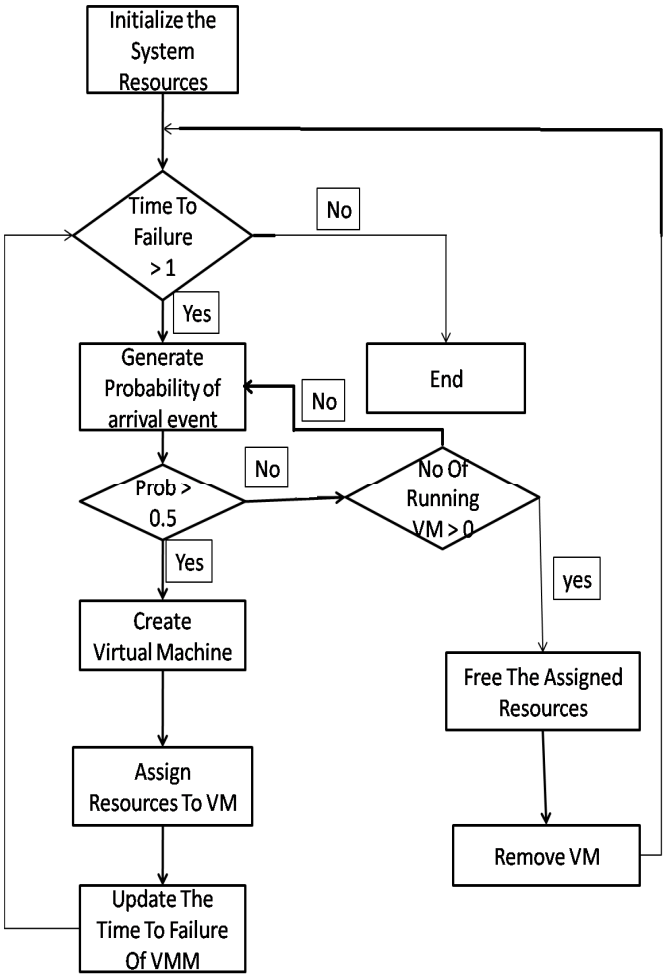

Fig. 1. Flowchart for Test Data Generation

The data generated, as explained in the previous subsection, is used to train the neural network. The change in resource utilization is computed for successive events and it is given as input to the neural network. The rate at which a VM affects the aging phenomenon depends on the change of resources utilized due to that VM, but not on the absolute quantity of resources available. As it is not dependant on absolute quantity of resources, the approach is to some extent machine independent.

Two kinds of feed forward neural networks are created. One neural network takes the 'data set at a particular time' and another takes the 'data sets in an interval of time' (Time Window) as inputs. Both neural networks generate percentage of change in time to failure as output.

C. Finding the Suitable Neural Network

The main problem in the machine learning algorithm is to find the right structure of the feed forward neural network. The following iterative procedure can be used to find the suitable neural network.

1. Create the neural network with some initial structure
2. Train the neural network with the input data

3. Find the error of the neural network
4. Change the structure of the neural network and repeat the steps 1-3, until the error is minimized.

IV. Experimental Setup

Two datasets have been generated. One data set follows a pattern for arrival and departure, and another without any pattern. Data for a total period of 10000 time units is generated. Experiments are conducted with and without a time window. Two different neural networks are used, one takes data set at a particular instant of time as input and another takes data set for a time window. The prediction accuracy of both kinds of neural networks has been compared in all the experiments. All programs are written in Java language.

V. Results And Analysis

In the first dataset, the probability of the arrival and departure event at a particular time is same. There is no pattern for the occurrence of the arrival and departure events. In this experiment a time window of size five is used. Prediction times using both neural networks is shown in the Fig. 2, in the form of a graph and percentage of error in prediction is shown in Table 1. Percentage of error in the predicted time to failure ranges from 0.02% to 5.37 for time instant case. For the case of time window of length 5 the range of percentage of error is 0.43 to 5.96. Maximum and minimum percentage of errors are shown in bold face in the table. The neural network that takes data of one time instant as input performs slightly better than the neural network that takes one time window as input.

In the second dataset some regular pattern is introduced for arrival and departure events. Time window sizes of six and eight are considered in the experimentation. The observed results are shown in Fig. 3. Time window based neural networks performed better than time instant based neural network. Among the two time window based neural networks, the neural network with time window length of eight performed better than the one with time window length of six. By observing the results of both experiments it can be concluded that the neural network that takes data of longer time window as input performs better, when there is a regular pattern of arrivals and departures. However when there is no regular pattern in arrival and departure events then simple time instant based neural network predicts slightly better than time window based neural network.

Experiments show that the length of the time window is important in the prediction of the time to failure, for the system with regular patterns for arrival and departure events. If that pattern of events can be covered in the time window and such a time window is given as input to the neural network then it can learn for the pattern and predict the time to failure more accurately.

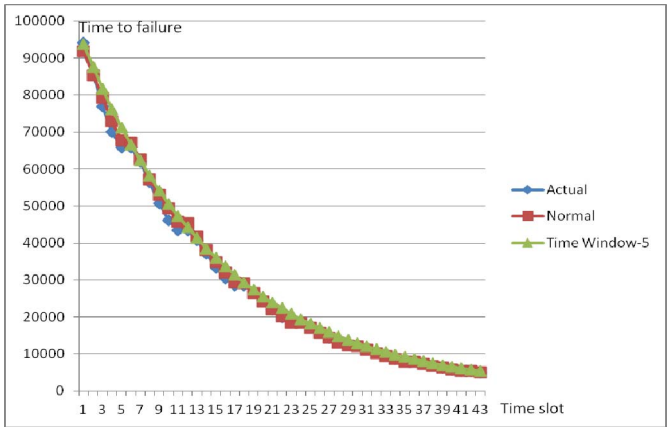

Fig. 2. Prediction of Time to Failure Using Neural Networks with Time Instant and Time Window of Length-5

Table 1. Percentage of Error for Time Instant and Time Window Methods.

Time Instant	Time Wind-5	Time Instant	Time Wind-5	Time Instant	Time Wind-5
2.31	**0.43**	5.25	5.64	4.62	3.5
0.4	2.26	2.2	5.09	3.86	4.29
3.08	4.39	1.52	2.69	1.84	10.6
3.94	4.8	1.61	1.76	1.03	1.09
2.68	3.9	1.39	4.15	3.69	4.16
1.99	1.06	0.61	2.81	4.32	5.05
0.92	0.55	0.62	1.74	5.14	4.52
1.44	2.9	2.15	3.81	4.31	4.98
4.79	**5.96**	2.79	3.26	0.03	1.2
5.02	5.47	3.52	2.4	0.84	1.81
5.37	6	3.1	5.24	**0.02**	1.33
4.66	2.01	1.25	2.87	0.67	1.06
2.27	1.17	1	2.45	2.61	5.19
2.5	3.46	1.73	2.42		
4.72	5.65	2.37	3.93		

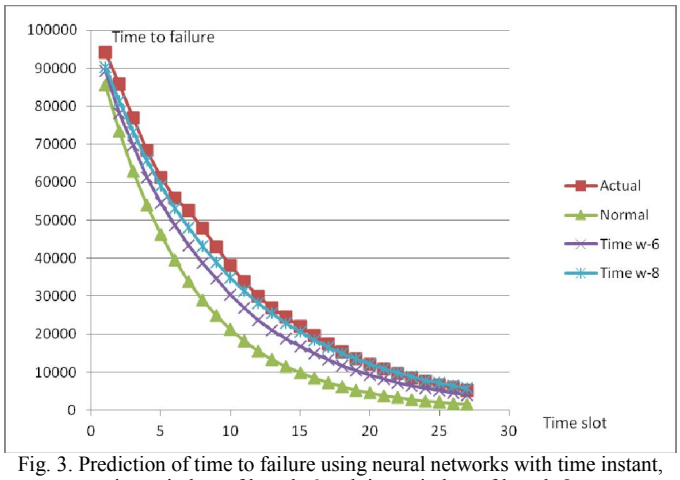

Fig. 3. Prediction of time to failure using neural networks with time instant, time window of length-6 and time window of length-8

VI. Conclusions And Future Work

By analyzing aging indicators, time to failure of the software system can be predicted. In this work, time to failure of the VMM is be predicted by considering different againg indicators like number of processes, number of threads, physical memory, swap space, number of TCP connections and CPU utilization using Artificial Neural Networks.

The number of inputs, to the Artificial Neural Network, can also affect the prediction accuracy. Artificial Neural Network, that takes time window as input, predicts the more accurately if that data contains some pattern in the arrival and departure events of VMs.

In future work, pattern recognition techniques can be applied to the dataset to identify more complex regularity in the dataset. Optimal window size for the ANN can be decided, so that it can predict time to failure of the software more accurately.

References

[1.] Shishiny, sally, Omar, "Mining Software Aging: A Neural Network Approach", IEEE, 2008

[2.] Garg, S., Kintala, C., Huang, Y., And Trivedi, K., "Minimizing completion time of a program by check pointing and rejuvenation", Performance Evaluation Review 24, 1, 252–261,1999

[3.] Grottke, M., Li, L., Vaidyanathan, K., And Trivedi, K., "Analysis Of Software Aging In A Web Server". Reliability, IEEE Transactions on 55, 3. 2006.

[4.] Bernstein, L., "Innovative technologies for preventing network outages", AT & T TECH J. 72, 4, 4–10.,1993

[5.] Huang, Y., Kintala, C., Kolettis, N., And Fulton, "Software rejuvenation: analysis, module and applications", In Fault-Tolerant Computing, 1995. FTCS-25. Digest of Papers, Twenty-Fifth Int. 1995.

[6.] Cotroneo, D., Natella, R., Pietrantuono, R., And Russo, S., "Software Aging Analysis Of The Linux Operating System", In Software Reliability Engineering (ISSRE), 2010 IEEE

[7.] Machida, F., Kim, D. S., And Trivedi, K. "Modeling And Analysis Of Software Rejuvenation In A Server Virtualized System", In Software Aging and Rejuvenation (WoSAR), 2010 IEEE Second Int'l. Workshop.

[8.] Alonso, J., Belanche, L., And Avresky, D., "Predicting Software Anomalies Using Machine Learning Techniques", Proceedings - 2011 IEEE International Symposium on Network Computing and Applications, NCA 2011, 163–170.

[9.] Bao, Y., Sun, X., And Trivedi, K., "A Workload-Based Analysis Of Software Aging, And Rejuvenation", Reliability, IEEE Transactions on 54, 3, 2005

[10.] Magalhaes, J. And Silva, L., "Prediction Of Performance Anomalies In Web-Applications Based-On Software Aging Scenarios", In Software Aging and Rejuvenation (WoSAR), 2010 IEEE Second Int'l. Workshop.

[11.] Matias Jr., R., F. P., "An Experimental Study On Software Aging And Rejuvenation In Web Servers", Proceedings - International Computer Software and Applications Conference 1, 189–196. 2006

[12.] Araujo, J., Matos, R., Maciel, P., Vieira, F., Matias, R., And Trivedi, K., "Software Rejuvenation In Eucalyptus Cloud Computing Infrastructure: A Method Based On Time Series Forecasting And Multiple Thresholds", In Software Aging and Rejuvenation (WoSAR), 2011 IEEE Third International Workshop on. 38 –43. 38

[13.] Vaidyanathan, K. And Trivedi, K., "A Comprehensive Model For Software Rejuvenation", Dependable And Secure Computing, IEEE Transactions On 2, 2, 2005

[14.] D. Contoneo, R. Natella, R. Pietrantuono, and S. Russo, "A Survey on Software Aging and Rejuvenation Studies", IACM Journal on Emerging Technologies in Computing Systems (JETC), Vol 10, Issue 1, 8:1-8:34, 2014.

[15.] K. Vaydianathan and K.S. Trivedi., "Extended Classification of Software Faults Based on Aging", Dept. of ECE, Duke University, Durham, USA, 2001.

[16.] "Software Aging", http://en.wikipedia.org/wiki/Software_aging, 2011