Journal of Engineering and Applied Sciences 12 (3): 644-652, 2017
ISSN: 1816-949X
© Medwell Journals, 2017

Probabilistic Reliability Prediction Models for Task Scheduling in Distributed Systems: A Review

[1]Faruku Umar Ambursa, [1, 2]Rohaya Latip, [1]Azizol Abdullah and [1]Shamala Subramaniam
[1]Departement of Communication Technology and Network,
Faculty of Computer Science and Information Technology,
[2]Institute for Mathematical Research (INSPEM), Universiti Putra Malaysia, Serdang, Malaysia

Abstract: In service-oriented distributed systems, beside time and cost, reliability is the most important concern to both service users and the service providers. Although, this has been many decades problem, the existence of large number of service systems on the internet today has rendered the problem more difficult. This is because the distributed environment of today is more complex with numerous uncertainties and chances of failure at all levels. Therefore, selection of reliable service poses a serious challenge. To combat this problem, over the years, huge number of reliability researches has been reported in literature. These researches have been categorized and analysed in many survey and review studies. However, most of these studies focus on the architecture-based reliability mechanisms and pay little attention to the advances in the popular probabilistic reliability prediction methods which are based on quantitative reliability measurements. These methods which are sometimes called 'black box' techniques are of great importance to both service designers and service clients such as brokers and other proprietary schedulers, for evaluating reliability of services or service components. Therefore, in this study the previous survey and review studies are extended by analyzing these methods and their recently proposed variants. In the end the study reveal some of the current issues that need further research.

Key words: Reliability prediction, fault tolerant, distributed, systems, method

INTRODUCTION

Today, distributed systems such as grid and cloud, offer services to millions of users in diverse applications such as business, scientific and social networking. Enormous number of these services, rendered by various service-oriented distributed platforms is available on the internet. However, these services differ in terms of their ability to render the required QoS guarantee to users. Reliability is one of the most important QoS attributes that users require for their applications. It is a service characteristic upon which depend many other important QoS features of services such as monitory cost, availability, reputation, etc. Therefore, in scheduling user applications, one of the challenging aspects is selecting reliable composite services, i.e., how to ensure the selected services can guarantee the expected QoS delivery such as the deadline constraint. This is because many of these services exhibit dynamic QoS behavior at runtime. Also, service-oriented distributed environments are complicated and complex with numerous uncertainties and chances of failures at various levels. For example, some workflows are composed of thousands of tasks with various execution times which are interdependent; these workflows are executed on distributed services many of which are prone to failures due to long unning tasks, particularly data intensive workflows. For instance, Google reported on average 5 permanent failures in form of machine crashes per MapReduce workflow during March 2006 (Dean, 2006) and at least one disk failure in every run of MapReduce workflow with 4000 tasks. In addition, in concrete workflow execution, failure of one of the services can lead to failure of the entire workflow. As a result, many subareas in reliability research have been born (Nandagopal and Uthariaraj, 2010; Wang *et al*., 2014, 2011; Conejero *et al*., 2014; Hirales-Carbajal *et al*., 2012; Garg and Singh, 2014; Amoon, 2012; Ding *et al*., 2014; Ma *et al*., 2016; Tang *et al*., 2015; Calheiros *et al*., 2015; Li *et al*., 2014; Kianpisheh and Charari, 2014; Cicotti *et al*., 2015; Wang *et al*., 2014).

Researches in reliability have been broadly categorized into two sub-areas, namely: reactive and proactive failure management. Reactive methods are fault-tolerant schemes that are otherwise known as

Corresponding Author: Faruku Umar Ambursa, Departement of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang, Malaysia

knowledge-free techniques. They do not consider information about task and use approaches such as checkpointing, replication, retry, etc., to ensure task execution is completed. These techniques are slow and results in severe waste of resources. For example: in checkpointing, large storage space is needed to keep the checkpoints; in replication, task is duplicated and submitted to a number of service nodes, only the one that finishes first is useful, others are discarded. On the other hand, proactive methods which are known as knowledge-based schemes, use historical data about task and service characteristics to find a reliable task-to-service match. In other words, based on proactive method, finding reliably good task-to-node matches requires predicting the reliability of the match, typically based on the estimated duration of task which is normally provided by the submitting user and the success/failure history of the corresponding service node. To obtain historical behavior of services, performance monitoring is typically used which involves recording successes and failures of the monitored node. Reliability monitoring and prediction based methods provide viable solution for producing good and reliable schedules. By using reliability prediction techniques, it is possible for schedulers to forecast how tasks in an application will behave on distributed services and thus make decisions on how and where to run them (Yu and Buyya, 2005a, b). They are simpler and more efficient mechanisms against the dynamic nature of services at runtime.

In the literature, researches in reliability prediction are numerous and can be categorized based on service architecture level and service level. The service architecture level methods are comprehensive testing schemes conducted on the distributed systems to collect failure data and to make sure that the reliability threshold has been achieved before making the service available to the end users (Immonen and Niemela, 2008). These methods mostly involve state-based models such as Markov chain and Bayesian techniques. On the other hand, the service level schemes mostly involve quantitative models used to evaluate reliability of services. They are user-oriented models and are the focus of this research. These reliability models include success rate, failure rate/count, etc. The models are used either on the service provider side or service client side. Most of the current service schedulers and brokers, in one way or the other, make use of these models for evaluating reliability of services. Although, some of these models have been used for many decades, their variations have been proposed recently. Most of the current survey and review reports did not: either cover the most recent of these mechanisms or adequately analyse them. Therefore, the aim of this study is to explore some of the recent advances in this research area and analyse the models and their variants to reveal some of the current issues for further research.

Literature review: Numerous literature survey and review efforts have been made to explore and analyse the various studies conducted and several mechanisms proposed in the area of reliability prediction in distributed systems.

The research by Immonen and Niemela (2008) presented a survey of architecture based reliability and availability prediction methods. In their report reliability prediction approaches are classified, in the high level, into qualitative and quantitative methods. Quantitative methods are defined as system measurement based methods and reliability growth based models which focus on failures and down times and statistical testing. The methods are involve architecture based mechanisms such as state-based that consider the system's internal structure in reliability prediction, computing the system level reliability based on the reliabilities of its components. Salfner *et al.* (2010), a survey about the failure prediction methods is reported. This report covers proactive fault management approaches, specifically, online failure prediction techniques. The study by Pandey and Gayal (2013a, b) presented a review of the reliability prediction researches. In this report, models for reliability measurement are categorized as failure rate model, fault count model, software reliability growth models, etc. Also, reliability prediction approaches are classified into Bayesian models, architecture-based models and early software reliability prediction models. Sharma *et al.* (2016), a survey and taxonomy of reliability in cloud computing systems is reported. They discussed the reactive and proactive failure management classifications. Resource failures and fault tolerance mechanisms are covered in this study. The research by Ahmed and Wu (2013) reported a survey on reliability in distributed systems. The researchers also categorized reliability mechanisms based on fault-tolerance and failure prediction where the prediction approaches are further categorized into user-centric, architecture-based and state-based.

However, most of these survey and review reports focused more on the architecture based reliability mechanisms, giving less concern to exploring and anlysing the recent studies in the facet of quantitative reliability measurement and prediction approaches. Therefore, this study extends the past studies by focusing on providing the reader with the review and analysis of the popular probabilistic prediction models

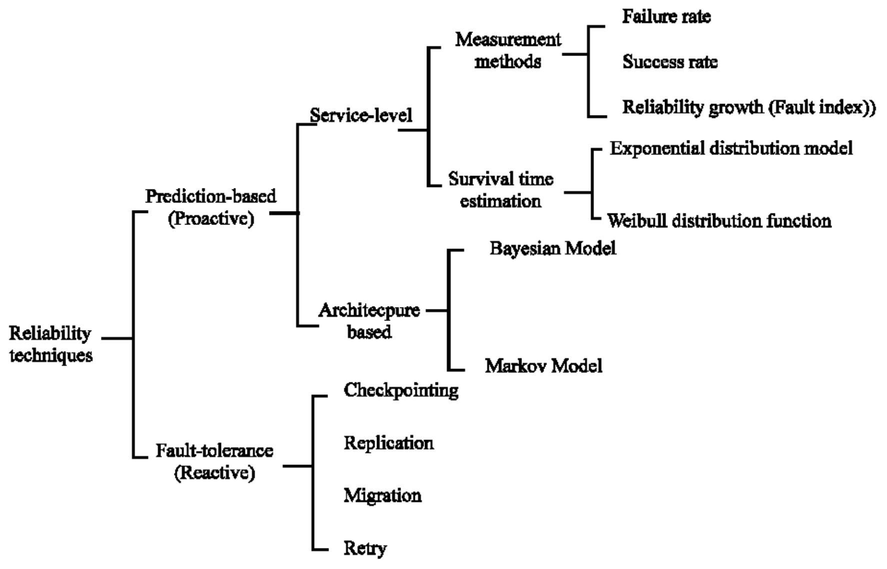

Fig. 1: Taxonomy of reliability techniques from user perspective

and their variants in service-oriented distributed systems. In the end, the paper demonstrates the common weakness of these models and the need for further research Fig. 1.

MATERIALS AND METHODS

Reliability approaches for task scheduling (An overview): One of the challenging aspects of task scheduling is how to ensure the selected services can guarantee the expected QoS delivery. This is because many of these services exhibit dynamic QoS behavior at runtime. Ahmed and Wu (2013) examined the reliability models in the literature and revealed that reliability can be ensured either by: predicting reliability early at architectural level to better analyze the system before actual application development or providing a fault tolerant system to ensure seamless environment for end users in case if any unanticipated or unpredicted scenario has occurred during execution in real environment. In this study, the overview of these reliability dimensions is discussed. The taxonomy of the reliability techniques is presented in Fig. 1.

Fault-tolerance-based reliability: This is reactive approach to providing reliability. It deals with handling failures as they occur during live execution of an application and provides the system with the ability to continue execution until completion (Ahmed and Wu, 2013). Some runtime fault-tolerant scheduling mechanisms such as checkpointing, replication, migration restart, etc. have been used for ensuring reliability to the end-user (Khan *et al*., 2010). Checkpointing is a technique that permits a running task to periodically store snapshots of its status somewhere in the system. If the node in which it runs fails during the execution of the task, another worker can resume the execution from the last available check-point. On the other hand, Task replication consists of executing several simultaneous replicas of the same task in different nodes. If one of them fails, the execution can hopefully succeed in one or more of the remaining replicas. This mechanism improves system responsiveness from the point of view of the users submitting tasks. However, these approaches waste a lot of resources which render them less desirable for both users and service owners in service-oriented environments. For example with checkpointing the CPU cycles used since the last snapshot are still thrown away. Additionally, this technique requires space to store the snapshots. For replication, the overhead to pay in terms of wasted resources is severe.

Prediction-based reliability: Prediction-based approach is used to address some of the issues associated with the reactive fault management techniques. This is to predict how tasks of an application will behave at runtime. It is important to application scheduling because tasks and services often show dynamic performance at runtime. It is powerful method for ensuring quality and reliability of schedules. By using performance prediction techniques, it is possible for schedulers to forecast how tasks in an application will behave on distributed services and thus make decisions on how and where to run them (Yu and

Buyya, 2005a, b). Reliability prediction, therefore, serves as a proactive mechanism against the dynamic nature QoS of services at runtime. It is considered more effective and efficient approach than the reactive approaches such as checkpoing and task replication highlighted above.

Reliability prediction of services can be classified based on service architectural level and service level. The service architectural level prediction methods are mostly used on the service provider side, typically at the design phase, for controlling the system development process for cost-effectiveness (Khan *et al*., 2010; Gokhale, 2007; Reussner *et al*., 2003; Pandey and Goyal, 2013a; Dai *et al*., 2003; Singh *et al*., 2001). These are comprehensive testing schemes conducted on the distributed systems to collect failure data and to make sure that the reliability threshold has been achieved before use by end users. On the other hand, service methods are quantitative system measurement based methods which focus on failures and down times used for analyzing systems already in use and for making predictions on implemented systems that are usually run and tested in a lab. This category of reliability evaluation methods is of great importance in service-oriented distributed environment. This is because the models are used in both service side and client side. In the service side, the quantitative models are used in some states/phases of architectural or component-based reliability analysis where the components of the service architecture are tested quantitatively based on 'black-box' approach. On the client side, since services are perceived as 'black-box', the methods are mostly employed to measure reliability of services. Client side systems include service brokers and other proprietary schedulers. The prediction methods based on the quantitative measurement models are usually simple and efficient for evaluating services and forecasting future reliability behavior of the services (Nandagopal and Uthariaraj, 2010; Sonnek *et al*., 2007; Wang *et al*., 2011; Amoon, 2013). To forecast reliability of an execution, probabilistic models are typically employed. Particularly, for estimating survival time of a service node, the popular probabilistic distribution models, i.e., the weibull probability distribution and exponential distribution are mostly used.

Existing probabilistic models: Predicting reliability of task on a service involves two aspects: service reliability measurement and task-service reliability estimation. Several studies have been reported on the reliability prediction of task mappings to available services. However, very few of these mechanisms pay adequate attention to how to accurately evaluate reliability of services (Wang *et al*., 2011; Amoon, 2013) which is fundamental and essential for successful prediction. To evaluate reliability of services various approaches have been proposed. Numerous analytical models have been proposed in literature to tackle the issue of reliability measurement in distributed systems. These approaches are based mainly on the failure history of a service node. The various models for reliability measurement can be categorized in to fault index, success rate and failure rate. In the next subsections we give overview of these models.

Fault index-based: One of the approaches and used in many previous works for reliability evaluation found in literature is based on using fault index. This fault index is maintained by taking into consideration the fault occurrence history information of service. The fault index of a service is incremented every time the service does not complete the assigned job within the expected deadline. The fault index is decremented whenever the service completes the assigned job within deadline (Amoon, 2012). This approach is used by Nandagopal and Uthariaraj (2010) and in several other researches Chang *et al*. (2009), Chtepen *et al*. (2006), Chunlin *et al*. (2009) Huang *et al*. (2009), Srinivasa *et al*. (2010). Nandagopal and Uthariaraj (2010) the value of the fault index is decremented only if its value is greater than or equal 1. So, the minimum value of the resource fault index is 0. Amoon (2013) studied this approach and observed that in most cases, it is not a suitable indicator for the resource failure history. For example, if we have two resources R1 and R2 and the total numbers of jobs assigned to each one are 100 and 1000, respectively, R1 completes 95 and does not complete 5. So, it has a fault index of 0. R2 completes 900 and does not complete 100. So, it has also a fault index of 0. Both resources have fault index = 0. This is because the value of the fault index is decremented only if its value is ≥ 1. Which resource will be selected in this case R1 or R2. Thus, the fault index is not an effective factor in choosing the most reliable resource for executing a job.

Success rate-based: A traditional and one of the most common approaches used to evaluate reliability of services is based on ratio of successfully completed tasks (Sonnek *et al*., 2007; Ambursa *et al*., 2016; Song *et al*., 2006; Tao *et al*., 2011; Kamvar *et al*., 2003). It is computed by calculating the ratio of the number of successful service executions and the total number of the service executions. This method is used in many scheduling models. Wang *et al*. (2011) studied this approach and

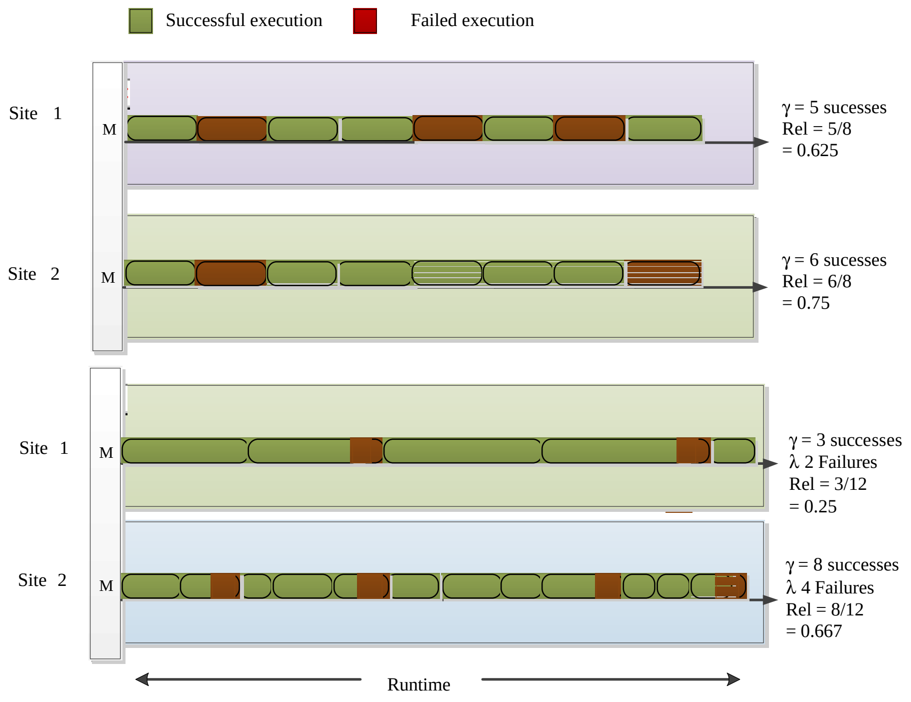

Fig. 2: a) Reliability calculation based on success counts with identical number of tasks; b) Reliability calculation based on success counts with variable number of tasks

discovered that two problems arise when it is used to capture reliability of services. First, from the service perspective, the influence of the task runtime (size) is not considered. For example, peer A has a higher task failure rate (task failures per unit time) than peer B, so peer B should have a better reputation. But traditional reputation models will instead predict a better reputation for peer A when peer A executes more tasks with short runtime and peer B executes more tasks with long runtime. This is because peer A may successfully complete more short tasks than peer B. sec, from the task perspective, existing reputation models assign the same reliability (success probability) to all tasks on a service based on the reputation of the service. However, the longer a task runs on an unreliable service, the lower success probability it should have (Wang *et al*., 2011). The weakness of success rate method is demonstrated in Fig. 2.

RESULTS AND DISCUSSION

Time-dependent: Wang *et al*. (2011) argued that real-time task failure rate of service is better way of capturing the reliability of the service. They proposed a time-dependent reliability-driven reputation mechanism which calculates failure rate of a service based on task runtime. The algorithm monitors and calculates reliability of each service based on successive time intervals, each lasting a time window. During each time interval, the monitoring server maintains the reliability statistic, in terms of failure rate, for each service. By so doing, the dynamic changes in real-time reliability of every service is observed and recorded. They used exponential function to compute reliability in terms of success probability of task-service mapping as:

$$R_j^i = e^{-t_j^i} \times rdr_j \tag{1}$$

Where:

i and j	=	Task and service
t_j^i	=	The time to complete task
i	=	On service
j and rdr_j	=	The reputation of service
j	=	Calculated in terms failure rate and a time decay factor

Simulation results shows that the approach can improve the reliability of a workflow application with more accurate reliability compared to approaches based on ratio of successfully completed tasks. Figure 3 demonstrate how failure rate addresses the problem of invariable number of tasks.

Fault rate: Amoon (2012) proposed a method based on resource fault rate to address problem of fault index. In

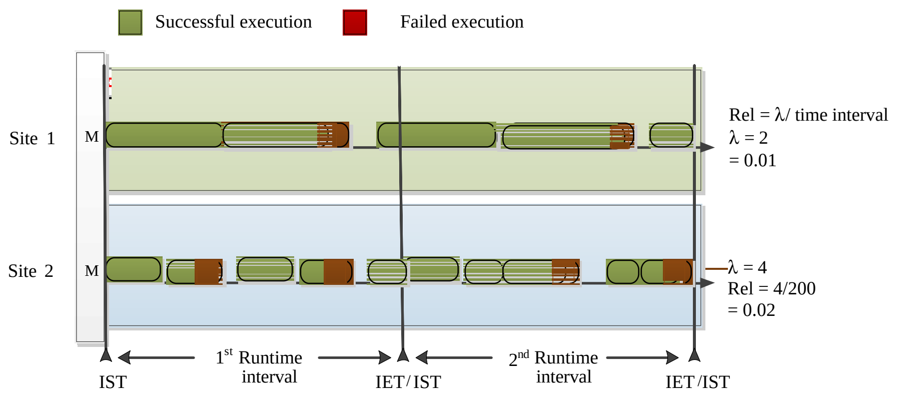

Fig. 3: Reliability calculation based on failure rate with variable number of tasks

this method it is argued that the resource with the lower failure rate will have a lower tendency to fail. Let N_f denotes the number of times the resource has failed to complete the jobs assigned to it, N_s represents the number of times the resource has completed jobs successfully. The fault rate P_{fj} of resource j is defined as:

$$P_{fj} = \frac{N_f}{N_s + N_f} \tag{2}$$

Each time a resource fails to complete a job the value of is increased by 1 and the jobs assigned to that resource will be distributed to other suitable resources. Otherwise, the value of the is increased by 1. Therefore, according to, the above example. R1 will have fault rate = 5% and R2 will have fault rate = 10%. So, R1 is selected because it has the lowest fault rate than R2 and consequently, the lowest tendency to fail (Amoon, 2012). Experimental results of the proposed failure-rate based system compared to the fault-index based system showed that; it achieved better throughput; it improves the turnaround time and the unavailability and its failure tendency is far better. The research adopted this strategy in his further research by Amoon (2013) to determine the number of checkpoints and the checkpoint intervals for each job.

Best-fit: The researchers proposed another scheduling polices based on resource failure rate approach. In their research, to characterize reliability of a task-service mapping, two functions, called score functions are introduced. The first function is based on the probability of the node surviving enough time to complete the task. Similar to Wang *et al*. (2011), exponential distribution function is used to calculate score of mappings. But the failure rate is computed as λ_i = 1/mean alive time. The first score function is used to determine if a resource is suitable to complete a given task. The sec function takes into consideration the normalized gap between the length of task and the expected lifetime of the resource. It is defined as:

$$f_2(i,j) = \begin{cases} MAX_{SCORE} & \text{if } E[X_i] = t_j, \\ \dfrac{e^{-\lambda_i t_j}}{|\lambda_i . t_j - 1|} & \text{otherwise} \end{cases} \tag{3}$$

Where:

$E[x_i] = 1/\lambda_i$	=	The expected lifetime of resource i
MAX_{SCORE}	=	A certain value considered as maximum possible score
$e^{-\lambda_i t_j}$	=	The first score function
$\lvert\lambda_i . t_j - 1\rvert$	=	The normalized gap between the length of task and the expected lifetime of the resource

In general, the second function is used to determine if the task length suits the expected lifetime of the resource. The idea of using the two functions is to favour the execution of long tasks in stable nodes, using the unstable ones for short tasks as a way to increase node utilization and system throughput. The mechanism achieves better performance compared to the reactive scheduling algorithms based on checkpointing and replication only. It however shows similar performance, in terms of makespan and time wastage when compared to the Amoon (2012) approach discussed above.

Summary and analysis of the existing Models: Table 1 provides summary of comparison of the various schemes in terms of reliability prediction. The analysis of each aspect is detailed in the following.

The related works on reliability prediction are categorized based on resource reliability state evaluation

Table 1: Comparison of the related reliability prediction approaches

Reference	Underlying idea	Node reliability model	Reliability prediction model	Weakness
Sonnek *et al.* (2007) and Tao *et al.* (2011)	Success rate	$\lambda=\frac{N_S}{N'}$	-	Low reliability under variable sizes of tasks
Nandagopal and Uthariaraj (2010)	Fault index	$\lambda=\begin{cases}\lambda+1, \text{if fail}\\ \lambda-1, \text{if succ}\end{cases}$	-	Low reliability under variable number of tasks
Amoon (2012)	Fault rate	$\lambda=\frac{N_f}{N_f+N_s}$	$f=e^{-\lambda_i t_j}$	High makespan and high waste of resources
Amoon (2012)	Failure rate	$\lambda=\frac{N_f}{\text{run\_time}}$	$f=e^{-\lambda_i t_j}$	High makespan and high waste of resources
Alexander and Jose (2015)	Best fit	$\lambda=\frac{1}{\text{alive\_times}}$	$f=\frac{e^{-\lambda_i t_j}}{\lvert\lambda_i, t_j-1\rvert}$	High makespan and waste of time in stable but high-response-time environment

N = Total execs, N_s = Successes, N_f = Total failures

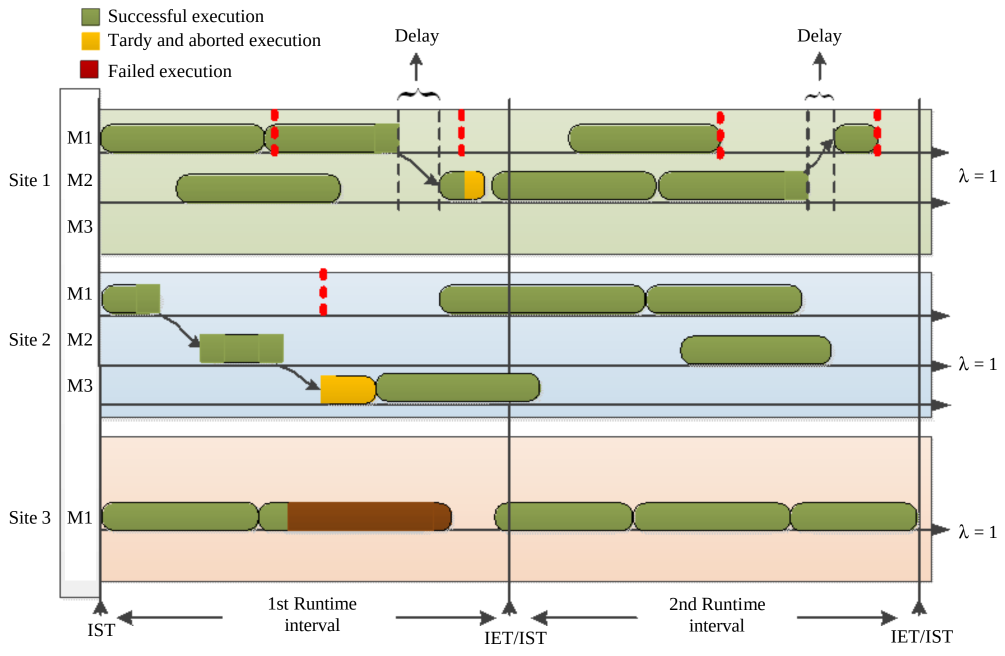

Fig. 4: Weakness of reliability calculation based on failure rate only

(Fig. 4) methods: fault index, success ratio and failure rate. Nandagopal and Uthariaraj (2010) monitored and computed reliability of a resource is based on fault index which decrements with successful execution and increments with failure. However, the method gives incorrect failure rating to resources when the resources execute variable number of tasks. Success ratio (Sonnek *et al.*, 2007; Tao *et al.*, 2011) is one of the most common techniques used to evaluate reliability of a resource. The reliability is computed in terms of ratio of number of successfully completed task executions and total number of task executions. This method however does not give correct estimation when the sizes of the tasks are variable (Wang *et al.*, 2011). Wang *et al.* (2011) computed reliability of resources based on time-dependent failure rate which is defined as the mean time to failure divided by runtime. Amoon (2012) proposed another mechanism that computes reliability of a resource based on fault rate. The fault rate is monitored and computed in terms number of failed executions divides by the total executions. This addressed the problem of reliability evaluation based on fault index. However, the approach has common problem with the scheme based on success ratio that is it does not perform well when resources execute tasks with variable sizes. Alexander used failure rate based technique and showed that compared to Amoon's work, it exhibits similar performance in terms of throughput and compared to non-failure-aware techniques, it achieves significantly better throughput and less wasted resources. On the other hand, the researcher however pointed out that selecting resource based on only their failure rate leads to underutilization of the resources because short tasks are sometimes send to resources with longer lifetime. In view of this problem, the researchers extended the reliability model and proposed BFGC algorithm that assigns tasks to resources looking for the best fit between the expected lifetime of the resource and the task length. This approach has succeeded in assigning long tasks to stable nodes that can guarantee failure free execution and short tasks to unstable nodes. As a result, the wasted time is reduced. However, the reliability model leads to high abortion of tasks which results to high makespan and resource wastage (high re-executions). This is because of inadequate mechanism to predict the ability of the resources to finish executions before their given deadlines. In other words, although the current reliability model can determine resource ability to survive until completion of an execution, it does not guarantee that the resource would finish the job before its deadline. Figure 4 demonstrates how tasks executions are delayed due to internal fault tolerance operations which are transparent to the schedulers. Therefore, to address this shortcoming, new algorithm needs to be developed that can, more appropriately, estimate reliability of resources taking into cognizance their actual failure behavior and tardiness behavior.

CONCLUSION

In this research, the overview of reliability prediction approaches is presented. Unlike the previous survey and review works which focus more on exploring and analyzing the server-side reliability techniques this research concentrates on the recent advances in the popular quantitative reliability models, from the user-side perspective. It provides overview and description of these models and analysis the different approaches reported in the literature. In the end, it reveals some of the current issues that need further attention of researchers in this field.

REFERENCES

Ahmed, W. and Y.W. Wu, 2013. A survey on reliability in distributed systems. J. Comput. Syst. Sci., 79: 1243-1255.

Ambursa, F.U., R. Latip, A. Abdullah and S. Subramaniam, 2016. A particle swarm optimization and min-max-based workflow scheduling algorithm with QoS satisfaction for service-oriented grids. J. Supercomputer, 1: 1-34.

Amoon, M., 2012. A fault-tolerant scheduling system for computational grids. Comput. Electr. Eng., 38: 399-412.

Amoon, M., 2013. A job check pointing system for computational grids. Open Comput. Sci., 3: 17-26.

Calheiros, R.N., E. Masoumi, R. Ranjan and R. Buyya, 2015. Workload prediction using ARIMA model and its impact on cloud applications QoS. IEEE. Trans. Cloud Comput., 3: 449-458.

Chang, R.S., J.S. Chang and P.S. Lin, 2009. An ant algorithm for balanced job scheduling in grids. Future Generat. Comput. Syst., 25: 20-27.

Chtepen, M., B. Dhoedt, F. Cleays and P. Vanrolleghem, 2006. Evaluation of replication and rescheduling heuristics for gird systems with varying resource availability. Proceedings of the 18th International Conference on Parallel and Distributed Computing Systems, November 13-15, 2006, ACTA Press, Anaheim, California, USA., ISBN:9780889866386, pp: 622-627.

Chunlin, L., Z.J. Xiu and L. Layuan, 2009. Resource scheduling with conflicting objectives in grid environments: Model and evaluation. J. Netw. Comput. Appl., 32: 760-769.

Cicotti, G., L. Coppolino, S. D'Antonio and L. Romano, 2015. Runtime model checking for SLA compliance monitoring and QoS prediction. J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl., 6: 4-20.

Conejero, J., B. Caminero, C. Carrion and L. Tomas, 2014. From volunteer to trustable computing: Providing QoS-aware scheduling mechanisms for multi-grid computing environments. Future Gener. Comput. Syst., 34: 76-93.

Dai, Y.S., M. Xie, K.L. Poh and G.Q. Liu, 2003. A study of service reliability and availability for distributed systems. Reliab. Eng. Syst. Saf., 79: 103-112.

Dean, J., 2006. Experiences with map reduce an abstraction for large-scale computation. Proceedings of the 15th International Conference on Parallel Architectures and Compilation Techniques (PACT), September 16-20, 2006, ACM, New York, USA., ISBN:1-59593-264-X, pp: 1-1.

Ding, S., S. Yang, Y. Zhang, C. Liang and C. Xia, 2014. Combining QoS prediction and customer satisfaction estimation to solve cloud service trustworthiness evaluation problems. Knowl. Based Syst., 56: 216-225.

Garg, R. and A.K. Singh, 2014. Fault tolerant task scheduling on computational grid using check pointing under transient faults. Arabian J. Sci. Eng., 39: 8775-8791.

Gokhale, S.S., 2007. Architecture-based software reliability analysis: Overview and limitations. IEEE Trans. Dependable Secure Comput., 4: 32-40.

Hirales-Carbajal, A., A. Tchernykh, R. Yahyapour, J.L. Gonzalez-Garcia, T. Roblitz and J.M. Ramirez-Alcaraz, 2012. Multiple workflow scheduling strategies with user run time estimates on a grid. J. Grid Comput., 10: 325-346.

Huang, P., H. Peng, P. Lin and X. Li, 2009. Static strategy and dynamic adjustment: An effective method for grid task scheduling. Future Gener. Comput. Syst., 25: 884-892.

Immonen, A. and E. Niemela, 2008. Survey of reliability and availability prediction methods from the viewpoint of software architecture. Software Syst. Model., 7: 49-65.

Kamvar, S.D., M. T. Schlosser and H. Garcia-Molina, 2003. The eigentrust algorithm for reputation management in p2p networks. Proceedings of the 12th International Conference on World Wide Web, May 20-24, 2003, Budapest, Hungary, pp: 640-651.

Khan, F.G., K. Qureshi and B. Nazir, 2010. Performance evaluation of fault tolerance techniques in grid computing system. Comput. Electr. Eng., 36: 1110-1122.

Kianpisheh, S. and N.M. Charkari, 2014. A grid workflow quality-of-service estimation based on resource availability prediction. J. Supercomputing, 67: 496-527.

Li, J., X. Luo, Y. Xia, Y. Han and Q. Zhu, 2014. A time series and reduction-based model for modeling and QoS prediction of service compositions. Concurrency Comput. Pract. Experience, 27: 146-163.

Ma, Y., S. Wang, P.C. Hung, C.H. Hsu and Q. Sun *et al*., 2016. A highly accurate prediction algorithm for unknown web service QoS values. IEEE. Trans. Serv. Comput., 9: 511-523.

Nandagopal, M. and V.R. Uthariaraj, 2010. Fault tolerant scheduling strategy for computational grid environment. Intl. J. Eng. Sci. Technol., 2: 4361-4372.

Pandey, A.K. and N.K. Goyal, 2013a. Background: Software Quality and Reliability Prediction. In: Early Software Reliability Prediction, Pandey, A.K. and N.K. Goyal (Eds.). Springer, New Delhi, India, ISBN:978-81-322-1175-4, pp: 17-33.

Pandey, A.K. and N.K. Goyal, 2013b. Introduction. In: Early Software Reliability Prediction, Pandey, A.K. and N.K. Goyal (Eds.). Springer, New Delhi, India, ISBN:978-81-322-1175-4, pp: 1-16.

Reussner, R.H., H.W. Schmidt and I.H. Poernomo, 2003. Reliability prediction for component-based software architectures. J. Syst. Software, 66: 241-252.

Salfner, F., M. Lenk and M. Malek, 2010. A survey of online failure prediction methods. ACM. Comput. Surv., 42: 1-68.

Sharma, Y., B. Javadi, W. Si and D. Sun, 2016. Reliability and energy efficiency in cloud computing systems: Survey and taxonomy. J. Netw. Comput. Appl., 74: 66-85.

Singh, H., V. Cortellessa, B. Cukic, E. Guntel and V. Bharadwaj, 2001. A bayesian approach to reliability prediction and assessment of component based systems. Proceedings of the 12th International Symposium on Software Reliability Engineering, Nov. 27-30, Washington, DC, USA, pp: 12-21.

Song, S., K. Hwang and Y.K. Kwok, 2006. Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling. IEEE Trans. Comput., 55: 703-719.

Sonnek, J., A. Chandra and J. Weissman, 2007. Adaptive reputation-based scheduling on unreliable distributed infrastructures. IEEE. Trans. Parallel Distrib. Syst., 18: 1551-1564.

Sonnek, J., A. Chandra and J. Weissman, 2007. Adaptive reputation-based scheduling on unreliable distributed infrastructures. IEEE. Trans. Parallel Distrib. Syst., 18: 1551-1564.

Srinivasa, K.G., G.M. Siddesh and S. Cherian, 2010. Fault-tolerant middleware for grid computing. Proceedings of the 12th IEEE International Conference on High Performance Computing and Communications (HPCC), September 1-3, 2010, IEEE, New York, USA., ISBN:978-1-4244-8335-8, pp: 635-640.

Tang, M., W. Liang, B. Cao and X. Lin, 2015. Predicting quality of cloud services for selection. Intl. J. Grid Distrib. Comput., 8: 257-268.

Tao, Q., H.Y. Chang, Y. Yi, C.Q. Gu and W.J. Li, 2011. A rotary chaotic PSO algorithm for trustworthy scheduling of a grid workflow. Comput. Oper. Res., 38: 824-836.

Wang, S., C.H. Hsu, Z. Liang, Q. Sun and F. Yang, 2014. Multi-user web service selection based on multi-QoS prediction. Inf. Syst. Front., 16: 143-152.

Wang, X., C.S. Yeo, R. Buyya and J. Su, 2011. Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm. Future Gener. Comput. Syst., 27: 1124-1134.

Yu, J. and R. Buyya, 2005a. A taxonomy of scientific workflow systems for grid computing. ACM. Sigmod Rec., 34: 44-49.

Yu, J. and R. Buyya, 2005b. A taxonomy of workflow management systems for grid computing. J. Grid Comput., 3: 171-200.