This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TCC.2021.3055119

3D Analytical Modelling and Iterative Solution for High Performance Computing Clusters

Yonal Kirsal, Yoney Kirsal Ever, Glenford Mapp, and Mohsin Raza

Abstract—Mobile Cloud Computing enables the migration of services to the edge of Internet. Therefore, high performance computing clusters are widely deployed to improve computational capabilities of such environments. However, they are prone to failures and need analytical models to predict their behaviour in order to deliver desired quality-of-service and quality-of-experience to mobile users. This paper proposes a 3D analytical model and a problem-solving approach for sustainability evaluation of high-performance computing clusters. The proposed solution uses an iterative approach to obtain performance measurements to overcome the state space explosion problem. The availability modelling and evaluation of master and computing nodes are performed using a multi-repairman approach. The optimum number of repairmen is also obtained to get realistic results and reduce the overall cost. The proposed model is validated using discrete event simulation. The analytical approach is much faster and in good agreement with the simulations. The analysis focuses on mean queue length, throughput and mean response time outputs. The maximum differences between analytical and simulation results in the considered scenarios of up to a billion states are less than 1.149%, 3.82%, and 3.76% respectively. These differences are well within the 5% of confidence interval of the simulation and the proposed model.

Index Terms—analytical modelling, iterative solution approach, mobile cloud computing, high performance computing clusters, state space explosion problem, large-scale clusters.

1 Introduction

MOBILE cloud computing (MCC) is a combination of wireless and mobile technologies and cloud computing to allow convenient, on-demand access to shared computing resources in the local environment. These can include storage servers, networking applications and services that can be effectively provisioned by network operators as well as mobile users [1]. However, in order to provide better quality of service (QoS) and seamless service to users, it is necessary to provide a low latency, high bandwidth, end-to-end service environment.

High performance computing (HPC) has therefore emerged as an appealing MCC service, especially with the proliferation of big data in a mobile environment [2] because it can be used to provide reduced latency in an MCC environment [4], [5] and thus results in high performance. In addition, HPC uses supercomputers and parallel processing techniques to solve complex computational problems [6]–[9]. However, computations for parallel processing are very expensive and thus, in some cases, are not affordable for MCC. This leads to new architectures such as cluster computing because the necessary work can be shared among cluster nodes. HPC clusters (HPCCs) allow large amount of computing nodes for parallel and/or simultaneous processing at a much lower cost [10]. HPCCs essentially provide access to large amounts of data and resources through different interfaces. HPCCs are the emerging paradigm that has been dominating the processing and visualization of huge amounts of web data. HPCCs can also be employed in various applications such as high throughput applications, Monte Carlo calculations, statistical simulations, grid/fog computing and software defined networks (SDN) [12]–[20]. In last few years, big data usage, grid/fog computing and SDN have significantly increased and many related applications have been developed. Hence, modeling and simulation of such systems became a new research focal point.

In many practical systems, failures are expected and they can significantly affect the system performance. Therefore, when the number of nodes is significantly large, focusing purely on performance without taking into account the possibility of node failures would lead to a significant over-estimation in how an actual system would perform. The typical HPPC is based on a master/slaves architecture [21]. It has a master node (head node) and the identical computing nodes (slave) [22]. The main responsibility of the master node is to distribute user tasks among the computing nodes; however these nodes cannot serve the tasks if the master node is not operational [23]. Hence, due to the single master node formation, such systems are particularly vulnerable because master node failures will limit access to the computing nodes. Thus, this significantly affects the availability of these clusters. In other words, the master node failures makes QoS analysis even more interesting and essential.

Analytical models are useful to understand the nature of systems and derive relevant conclusions by analysing the mathematical relations of parameters on a specific measure of interest [12]–[14], [24]–[26], [29]–[33]. So, there is a need to extend the analytic models for such systems with several assumptions in order to use the analytic equations [34]–

Y. Kirsal is with the Department of Electrical and Electronics Engineering, Faculty of Engineering, European University of Lefke, Lefke, Northern Cyprus, TR-10 Mersin, e-mail:ykirsal@eul.edu.tr.
Y.K. Ever is with the Software Engineering Department, Near East University, 99138 Nicosia, Cyprus
G. Mapp is with Middlesex University, London NW4 4BT, UK.
M. Raza is with Department of Computer Science, Edge Hill University, Ormskirk, UK.

[40]. Therefore, this study presents 3D analytical modelling and iterative solution for HPCC, considering the failures of master as well as a large number of computing nodes, together with optimum repair- men analysis. The proposed iterative solution approach has been shown to overcome the state space explosion problem for steady-state queuing systems while giving good approximate results. The state space explosion is a major problem for such modelling because it limits the size of such systems that can be evaluated [24], [29], [32], [33], [35]–[38], [40], [41]. Using this method, all steady state probabilities can be calculated. These probabilities can then be used to calculate QoS measures such as mean queue length (MQL), throughput (THRP) and mean response time (MRT) in this paper. In addition, the QoS results obtained from the analytical model are compared to the discrete event simulation (DES) results in order to show the accuracy and effectiveness of the proposed work. The DES [42] is mainly used for the validation purposes however it can also be used for the QoS evaluation of such systems as it simulates the actual scenario rather than the Markov models presented in this paper. DES therefore forms an important comparative baseline for this work.

1.1 Problem Statement

Many analytical approaches are presented in the literature for large-scale networks [5], [14], [28], [34], [35], [45] as well as HPPCs [3], [4], [8], [23], [25], [36], [38], [39]. However, the well-known solution approaches such as Markov Reward Model [33], the Matrix Geometric method [43], the spectral expansion approach [24], [26], [29], [36], [40], and/or open queuing networks for QoS analysis with failure suffer from the state space explosion problem due to the large number of states. Various approaches have also been proposed in the literature [29], [41] to minimize the state explosion problem for multi-dimensional Markov models, however none of them considered master/slaves architecture as well as the failure of master node and proposed a solution approach for QoS analysis. On the other hand, a single repairman strategy does not give the desired QoS output parameters for such systems, especially for large-scale complex systems [35]. In addition, solving complex systems through simulation resulted in prohibitively long computational times for such systems. Therefore, it is of great importance to develop an analytical method and a solution approach to overcome these problems for such systems.

1.2 Research Contributions

The main contributions of this paper can be summarized as follows:

- Analytical performance and availability models have been developed for large-scale HPCCs in a MCC environment. The multi-repairman scenario is also considered to obtain more realistic QoS results considering MCC characteristics. The optimum multi-repairman solution is obtained and used for each experiment which reduces the cost of such HPCCs.
- An iterative 3D analytical modelling solution approach is also proposed for HPCCs. A large number of nodes and availability issues for QoS evaluation in the MCC environment are addressed. The proposed model uses an iterative solution approach and a large number of nodes with failure of master and computing nodes can be handled without causing the state space explosion problem.
- Even-though the proposed iterative solution gives approximate results, findings obtained from the analytical model and an iterative solution approach for large-scale HPCCs show good agreement with DES in MCC. Findings obtained for processing time from DES for an extreme case is 22.46 hours however it is less than 5 hours for the proposed solution. Thus, this clearly shows the computational efficiency of the proposed solution.

1.3 Paper Organization

The rest of the paper is organized as follows: Section 2 presents the related studies and covers the motivation for this work. Section 3 describes the system model and the analytical solution approach. Section 4 presents the numerical results and discussions for the proposed model. Finally, Section 5 concludes this paper.

2 Related Work

For many years, researchers in modelling and simulation designed efficient and effective models for different applications in MCC. Similarly, HPCC and its application areas have emerged and progressed. This leads to the design and development of analytical models and solution approaches to understand and to predict of complex system behavior in order to ensure best QoS for such systems. Distributed computing platforms, such as Hadoop [44] and Spark [45] have been adopted to support such systems. In addition, SDN [13], [17], [18], grid/fog computing [15], [16], [19], [20] and Beowulf clusters [24]–[28] have also been widely used for QoS analysis, improving performance, data storage and processing solution of clustering.

In [13], a new analytical model was presented for investigating the performance of SDN for bursty and correlated arrivals modelled by the Markov Modulated Poisson Process (MMPP). This analytical model is for traffic characteristics of multimedia applications. The QoS performance metrics in terms of average latency and average network throughput of the SDN networks are derived based on the developed analytical model. The validity of the proposed model was demonstrated through extensive simulation experiments. A novel mobility management architecture was proposed in [17] for seamless mobility management in 5G heterogeneous networks when the users move from one SDN controller coverage to another. The authors used the concept of Distributed Hash Table (DHT) to track users movement. The proposed solution has been introduced in the OMNET++ simulator. Findings showed that the proposed solution reduces handover latency by around 50% compare to the existing ones. In [18] an adaptive update mechanism was presented based on the QoS-aware traffic classification and real-time network status in order to improve network resource utilization and system performance.

On the other hand, a new Fog-2-Fog (F2F) collaboration model was proposed in [19] to improve fog computing performance. A formal mathematical model which enables fog traffic management via service offloading in fog-based architecture was explored. The results showed that offloading significantly impacts the overall latency of services. Similarly, in [20] a fog collaboration approach was presented for simple and complex multimedia service delivery to cloud subscribers. The proposed work explored a learning mechanism that relies on online and offline simulation results to build guaranteed work-flows for new service requests. The obtained results demonstrated significant improvements on performance in terms of service delivery success rate, service quality, reduced power consumption etc. However, availability issues have not been considered in any of the studies listed above. In [31], the necessity of availability was shown and the availability modelling and evaluation of HPCC systems were presented. The authors in [31] have developed a novel solution approach using an object oriented Markov model which provides availability modelling for typical HPCC. Numerical results presented in [31] demonstrated that availability modelling and evaluation need to be considered at the system design stage for typical HPCCs.

The Beowulf type cluster system is a good example of HPCCs. This is due to a single head node with a possibility to have a backup node for the head node. These types of systems are called highly available Beowulf systems [27], [28]. As stated in [28] the role of the HPC service providers must be to give guaranteed QoS by offering highly available services with dynamically scalable resources. In [28] authors used HA-OSCAR, which is an open source High Availability (HA) solution for HPC/cloud that offers failure detection and component redundancy. It is assumed that any task being addressed by the cloud/cluster center is served via a suitable node called a facility node [28], [34]. This facility node may contain different computing resources. In addition, when the task is served, it leaves the center. However, the authors in [34] emphasised that cloud centers differ from traditional queuing systems in a number of important aspects. More importantly, a cloud center can have a large number of orders, summing up to hundreds and thousands requests, which traditional queuing analysis rarely considers. Hence, the analytical approaches were presented for master/slave architectures for such systems. However, large-scale clusters could not be considered due to the state space explosion problem. Presently, cloud computing, web hosting and cluster computing provide a total of 256, 372, 512 or even 1000 nodes [30], [31], [34]–[36], [38]–[40], [46]–[49]. For such systems, analytical models and solution approaches were proposed in [35] and [36]. However, a master/slave architecture was not considered for both studies. The proposed models in [35] and [36] are two-dimensional Markov chains. However, the proposed model is three dimensional due to the master/slave architecture. In addition, the failure and repair behavior of the master node is more essential in such systems. Considering availability issues of master node together with slaves and obtaining a solution approach is necessary for such systems to get the best QoS parameters. In addition, a single-repairmen analysis was applied in [4], [7], [13], [24], [26], [33], [36], [38], [39], [44] where the importance of multi-repairmen analysis was clearly demonstrated in [35]. Therefore, an optimum repairmen analysis performed in [35] is also adopted and used in this paper to get more realistic results.

The proposed analytical model is based on assumptions and an approximation of initial conditions by considering the behavior of the system in terms of performance and availability issues individually. The assumptions and the initial value of all unknown elements can be obtained from the practical systems as well as the theoretical studies in the literature. The most important assumptions in this paper are the arrival, service, failure/repair rates of master and slaves of the system which have been considered for all states. The inter-arrival and the service times are assumed to be exponentially distributed in this paper similar to previous studies in the literature [30], [32], [46], [47], [50]. For instance in [50] an open Jackson queuing model was proposed to characterize the service components in content-delivery-as-a-service (CoDaaS). A proposed model was formed with a network of queues. In this model, the arrival rates for each queue are modeled as a Poisson process whereas the service times follow the exponential distribution. In [46], a classic *M/M/m* open network was proposed to get the response time distribution of a cloud system assuming an exponential distribution of the inter-arrival and service times. Similarly, in [47], a *M/M/m/m+r* queuing model was considered for performance evaluation of a cloud system. The inter-arrival and service time distributions were modeled as an exponential with finite queue capacity (*m+r*). In [30], a queuing performance model consisting of a cloud architecture and a service center such as a data center was studied. The database server was used as the service center. Thus, the inter-arrival and the service times followed an exponential distribution with means $1/\lambda$ and $1/\mu$, respectively with multiple servers in [30]. The time between failure and repair times were also assumed to be exponentially distributed. The availability models mostly considered in the literature assume exponentially distributed time between failures [29], [36], [37]. However, it should be noted that alternative distributions such as Weibull and Gamma distributions for failures and Lognormal distribution for repairs have also been reported [51]–[53]. In addition, a number of nodes, repairmen, and buffer capacity are the important inputs for the QoS evaluation of the proposed system. It is difficult to assume and obtain input parameters for all possible situations. For example, the time between failures considered are taken as 100, and 1000 hours for each node (ξ is 0.01/hr, and 0.001/hr, respectively). On the other hand, the time between repairs considered is taken as 2 hours for each node (η is 0.5/hr). This is the case mostly for the software based failures and repairs [24], [26], [29], [31], [33], [35]–[37], [39], [40], [47]. This paper therefore provides a comprehensive analysis of HPCC clusters in master/slave configurations for MCC environments. A 3D analysis is used to address issues of availability and performance and a full set of results are obtained.

3 The Proposed HPCC and The Iterative Solution Approach

In this section, the proposed HPCC is presented along with an iterative solution approach for performability evaluation of systems containing one head node and a large number of computing nodes. Please note that, the head node and the computing nodes represent the master node and the slaves respectively, in this paper. The proposed HPCC architecture is shown in Fig. 1. The main responsibility of the head node is to distribute tasks to computing nodes; it can also serve requests from clients. Identical computing nodes provide computation.

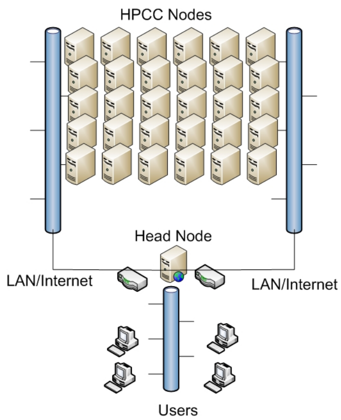


Fig. 1: High performance computing cluster, master-slave architecture

If the head node is not working, the computing nodes cannot serve the tasks. Due to the head node failure, the identical computing nodes are vulnerable. The failure of the head node limits access to the computing nodes. As stated before, the proposed system is three-dimensional (3D) as shown in Fig. 2 and consists of two planes. The notations used in this model is summarized in Table 1. The behaviour

TABLE 1: Notations Used

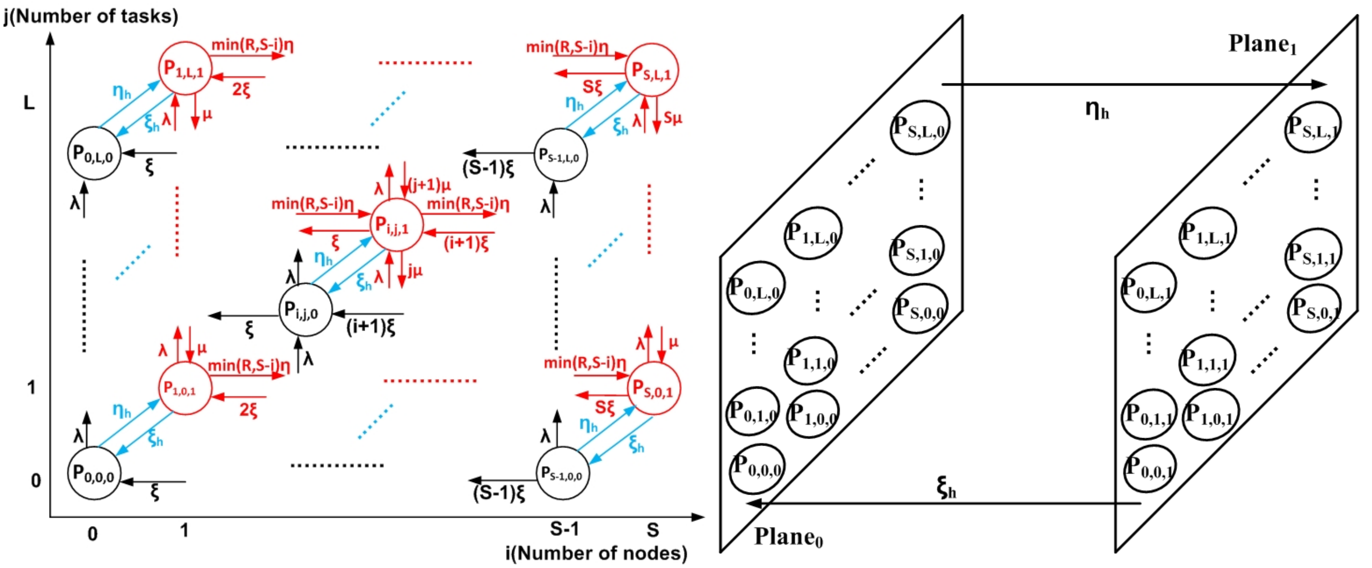
Notations	Descriptions
S	Number of nodes
L	Maximum number of requests in the system
λ	Total arrival rate
T	Service time of requests serviced by computing nodes
T_h	Service time of requests serviced by head node
μ	Service rate of computing nodes
μ_h	Service rate of head node
ξ	Failure rate of computing nodes
ξ_h	Failure rate of head node
η	Repair rate of computing nodes
η_h	Repair rate of head node
R	Number of repairman
ρ	Traffic intensity
$P_{(i,j,n)}$	State probabilities
n	Status of the head node

and modelling of the planes are presented in the following section. The proposed system consists of *S* number of nodes including a head node labelled as (1) and *S*-1 identical computing nodes, numbered 2,. . .,S-1,S on the x-axis. As shown in the Fig 2a, *L* is the capacity of the proposed system where L≥S. In other words, L is the maximum number of tasks that can be accommodated in the system including the one being served. Tasks arrive to the system in a Poisson stream at a mean rate of λ, and join the queue [12], [13]. Tasks are homogeneous and the service rates of the computing nodes are equal since same specifications of the computing nodes are assumed [23]–[26], [29]. The service times of requests serviced by the computing node (T) k (k=1,. . .,S) and the head node (T_h) are distributed exponentially with mean $1/\mu$ and $1/\mu_h$, respectively [31], [32], [34], [46]. T is the time taken to service a request and T_h is the time the head node takes to assign a task to be done. The head node distributes work amongst the nodes with rate μ_h. In addition, the head node also serves requests. It does so at the same rate of the other servers which is μ. If the head node participates in computations, it generally has the same service rate as that of the identical computing nodes (μ=μ_h). Since the random variables T and T_h have exponential distribution, the service times of a task is exponentially distributed with mean [23]–[26], [29], [31], [32], [34]–[37], [46]; $E[T] = 1/\mu = E[min(T, T_h)] = 1/(\mu + \mu_h)$.

ξ_h and ξ are failure rates of the head node and the computing nodes, respectively. Thus, $1/\xi_h$ and $1/\xi$ are operative periods of head node and computing nodes, respectively, and the means are also distributed exponentially [33], [35], [36], [38]–[40], [44], [47]. At the end of the node failure time, an exponentially distributed repair time is needed with mean $1/\eta$. On the other hand, if the head node fails, the repair facility is first provided to head node with mean repair time $1/\eta_h$. This is because, the computing nodes cannot serve without of the head node. If there are tasks waiting to be served, the operative computing nodes cannot be idle. Services that are interrupted by failures are eventually resumed from the point of interruption or repeated with resampling. In the case of head node failure, tasks continue to arrive with the same rate, λ, and tasks in the queue remain in the queue without being serviced.

3.0.1 Modelling Proposed HPCC

In this section, the proposed analytical model and the iterative solution method are introduced for large-scale HPCCs. It is possible to represent the proposed system with *S* computing nodes, including the head node, by using a Quasi Birth and Death (QBD) process with finite state space. Since in a HPCC, none of the computing nodes can operate without the head node, the relation of the failure and the repair rate of the head node lead us to model the proposed system in two planes. The transitions highlighted in blue in Fig. 2a shows relationships between two planes. For a clearer view, Fig. 2b indicates relationships between two planes. The first plane, which is the plane highlighted in red in Fig. 2a is used in states where the head node is always available, which is indicated as $Plane_1$ in Fig. 2b. On the other hand, the $Plane_0$ is the second plane as represented in Fig. 2b and is highlighted in black in Fig. 2a, which is used to represent the states, where the head node is broken. Hence, the proposed system is 3D and has two planes as shown in Fig. 2. Thus, $P_{i,j,n}$s are all steady state probabilities of the proposed system. It can be seen in the Fig. 2, that values of *i* and *j* indicate the number of computing nodes and number of tasks in the system, respectively and *n* indicates the mode



(a) The state transition diagram of the proposed model

(b) General transitions between two planes

Fig. 2: Three dimension (3D) feature of the proposed high performance computing clusters

of a head node. When *n=0* the head node is not operational which represents the $Plane_0$. Whereas, when *n=1* the head node is operational and these states are represented in $Plane_1$.

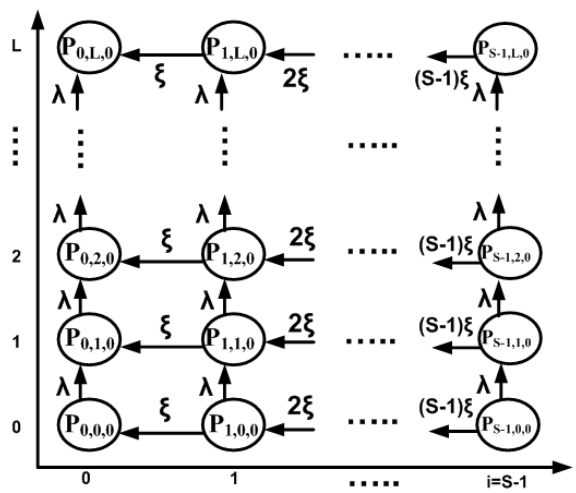


Fig. 3: State diagram of the $Plane_0$ where the head node is not operational

Figs. 3 and 4 show the state diagram of the $Plane_0$ and $Plane_1$, respectively of the proposed 3D system. There are *S-1* computing node configurations, i=0, 1,. . ., S-1 in Fig. 3. These *S-1* configurations indicate the possible states of the $Plane_0$. As described before, L represents the number of tasks in the system for both figures. The $Plane_0$ describes the case where the head node is not operational and the whole system does not provide service as shown in Fig. 3. Therefore, the downward transitions with service rate μ are not available. The repair priority is given to the head node; hence, the computing nodes cannot be repaired before the head node becomes alive. In other words, there are no repair transitions for the identical computing nodes in $Plane_0$, since the only repair transition which can take place is the transition to $Plane_1$. On the other hand, in Fig. 4, there are *S* computing node configurations, i=1,. . ., S and they are used to represent possible operational sates similar to Fig. 3. However, the number of computing nodes start from 1 because the head node is operational and thus requests are being served. Hence, downward transitions are possible since the system is active. In addition, since the head node is operational, it is possible to use the repair facility to deal with computational node failures.

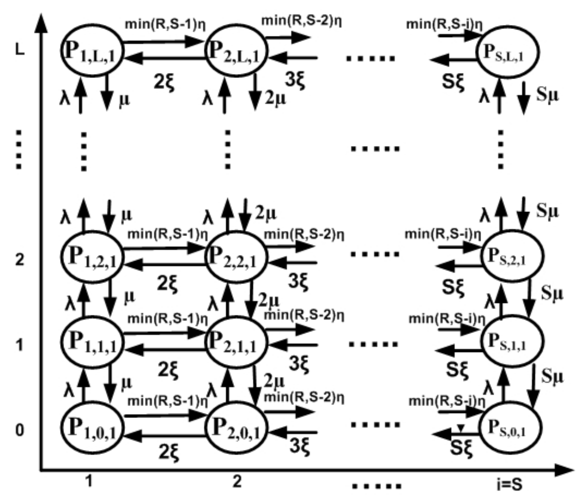


Fig. 4: State diagram of the $Plane_1$ where the head node is operational

Available identical computing nodes can provide services with a service rate of μ and the broken nodes can be repaired with rate of min(R,S-i)η. *R* is the number of repairmen considered in the proposed model. When large-scale systems are considered, the effect of R is more evident on QoS evaluation since the number of nodes increase and thus the probability of having node failures increases as well. Therefore, as stated in [35] obtaining the optimum number of repairmen is essential and the relationship between the number of nodes and repairmen is required for maximizing the QoS for such systems. The detailed analysis can be found in [35] in order to obtain the optimum R value. A similar evaluation technique is applied in our proposed model. As previously stated and clearly seen in Fig. 4, ξ and η are the failure and repair rates of each nodes in the proposed system. However, since the case of multi-repairmen is being considered in this paper, the repair rate of the nodes depends on the minimum number of R and inoperative nodes in the system. Once the condition R$\eta \geq S\xi$ is satisfied, the optimum number of repairmen can be found to obtain more realistic QoS outputs.

3.1 The Iterative Solution Approach

This section explains and represents the proposed iterative solution approach for large-scale HPCC along with availability issues and optimized repairmen. The analytical evaluation is considered and relevant equations are generated for the proposed 3D model. An approximate decomposition is applied to a multi-node cluster in order to enable fast convergence of the steady state probabilities. Then, all balance equations are produced considering the proposed system for a large number of parallel homogeneous computing nodes. All required performance measurements can be obtained by generating the balance equations for steady state probabilities of the system and solving them by using an iterative method for both planes. Moreover, in order to have a faster convergence of the iterative solution, the state probabilities of $P_{i,j,1}$s can be computed by analytical decomposition. These state probabilities of $P_{i,j,1}$ may not give very close approximations to real steady state probabilities, however the balance equations are still required in order to take all possible transitions into account for fast convergence. Thus, to find all $P_{i,j,n}$, the sum of all probabilities in both planes of the computing nodes should be considered individually. Two planes, $Plane_0$ and $Plane_1$, can then be analyzed separately. Every single plane has its own states and hence the sum of all these state probabilities in each planes is equal to 1, as is the case in single/multi server queue system. Thus, it is necessary to compute the sum of all probabilities in each phase. The sum of the overall probabilities ($Plane_0$ + $Plane_1$) should be 1 as shown in equation 1.

$$\sum_{i=0}^{S-1}\sum_{j=0}^{L} P_{i,j,0} + \sum_{i=1}^{S}\sum_{j=0}^{L} P_{i,j,1} = 1 \tag{1}$$

However, in order to obtain a general solution for the sum of all probabilities can be generated as $\xi_h/(\eta_h+\xi_h)$ and $\eta_h/(\eta_h+\xi_h)$ for $Plane_0$ and $Plane_1$, respectively. The derivation of both expressions can be found in Appendix A. Both expressions clearly indicate that the head node failure and repair rates are essential to find overall probabilities for such systems. Therefore, the following actions are taken to analyze performability of the HPCCs. The state probabilities of $Plane_1$ are analyzed where the head node is operational. Thus, the sum of all possible probabilities of $Plane_1$ is taken as $\eta_h/(\eta_h+\xi_h)$ as it can be derived in Appendix A. Thus, equation 2 can be written as follows:

$$\sum_{i=1}^{S}\sum_{j=0}^{L} P_{i,j,1} = \frac{\eta_h}{\eta_h+\xi_h} \tag{2}$$

It is required to obtain the $P_{i,j,1}$ values in $Plane_1$. However, it is clear that these probabilities cannot be obtained directly by using a product form solution. Therefore, at first, the sums of all probabilities are considered for each operational state of the system in $Plane_1$. Figure 5 shows the overall operative states of computing nodes for $Plane_1$.

Fig. 5: General lateral transitions of the system for a $Plane_1$

Equation 3 can then be used to derive to calculate $P_{i,j,1}$ for all possible values of i.

$$\sum_{j=0}^{L} P_{i,j,1} = \frac{\frac{\eta_h}{\eta_h+\xi_h}\frac{1}{i!}\left(\frac{min(R,S-i)\eta}{\xi}\right)^{i-1}}{\sum_{i=0}^{S}\frac{1}{i!}\left(\frac{min(R,S-i)\eta}{\xi}\right)^{i-1}} \tag{3}$$

where i=1,2,3, ...S. Since the sum of all $P_{i,j,1}$ in $Plane_1$ is known, it is easy to compute the overall probabilities for each operational computing node. $P_{i,j,1}$ can then be calculated in terms of $P_{i,0,1}$. Hence, equation 4 is derived using product form solution. By these set of equations all $P_{i,j,1}$ can then be expressed in terms of $P_{i,0,1}$.

$$P_{i,j,1} = \begin{cases} \frac{\rho^j}{j!}\cdot P_{i,0,1} & 0 \leq j \leq i \\ \frac{\rho^j}{j!i^{j-i}}\cdot P_{i,0,1} & i+1 \leq j \leq L \end{cases} \tag{4}$$

where i=1,2,3, ...S and $\rho = \lambda/\mu$. Then, equation 4 can be generalized for each column as follows:

$$\sum_{j=0}^{L} P_{i,j,1} = \left[\sum_{j=0}^{i}\frac{\rho^j}{j!} + \sum_{j=i+1}^{L}\frac{\rho^j}{i!i^{j-i}}\right] P_{i,0,1} \tag{5}$$

where i=0,1,..., S. Hence, $P_{i,0,1}$ can be computed as in equation 6 using the equations 3-5 with some simplifications.

$$P_{i,0,1} = \frac{\left(\frac{min(R,S-i)\eta}{\xi}\right)^i}{i!\sum_{k=0}^{S}\frac{\left(\frac{min(R,S-i)\eta}{\xi}\right)^k}{k!}\left[\sum_{j=0}^{i}\frac{\rho^j}{j!} + \frac{(i^{L-i}\rho^{i+1})-\rho^{L+1}}{i!i^{L-i}(i-\rho)}\right]} \tag{6}$$

where i=0,1,..., S. Since $P_{i,0,1}$s have been obtained, it is easy to find all $P_{i,j,1}$ by using the above equations. Thus,

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TCC.2021.3055119

the general expression of $P_{i,j,1}$ can be written as follows:

$$P_{i,j,1} = \begin{cases} \dfrac{\left(\frac{min(R,S-i)\eta}{\xi}\right)^i \frac{\rho^j}{j!}}{i! \sum\limits_{k=0}^{S} \frac{\left(\frac{min(R,S-i)\eta}{\xi}\right)^k}{k!} \left[\sum\limits_{j=0}^{i} \frac{\rho^j}{j!} + \frac{(i^{L-i}\rho^{i+1}) - \rho^{L+1}}{i! i^{L-i}(i-\rho)}\right]} \\ where \qquad j = 0, 1, 2, \cdots, i \\ \dfrac{\left(\frac{min(R,S-i)\eta}{\xi}\right)^i \frac{\rho^j}{j! i^{j-i}}}{i! \sum\limits_{k=0}^{S} \frac{\left(\frac{min(R,S-i)\eta}{\xi}\right)^k}{k!} \left[\sum\limits_{j=0}^{i} \frac{\rho^j}{j!} + \frac{(i^{L-i}\rho^{i+1}) - \rho^{L+1}}{i! i^{L-i}(i-\rho)}\right]} \\ where \qquad j = i+1, i+2, \cdots, L \end{cases} \tag{7}$$

Hence, all $P_{i,j,1}$ can be calculated using equation 7 which are not exact. Please note that, the state probabilities obtained for $Plane_1$ are used to get faster computations and more accurate results. However, it is not possible to follow a similar approach for $Plane_0$. This is mainly because in $Plane_0$ the system does not serve.

3.2 Balance Equations and Iterative Procedure

In this section, the main balance equations are derived for each plane individually in order to obtain all $P_{i,j,n}$s. In addition, the balance equations obtained are also used to consider the transitions between these planes to take all possible transitions into account.

3.2.1 Balance Equations for $Plane_1$ and $Plane_0$

In order to accumulate the effect of transitions together for the $Plane_0$ and $Plane_1$, all possible balance equations can be derived from Fig. 3 and Fig. 4, respectively. Thus, as an example, the generalized balance equations which give an idea how to obtain all balance equations for the $Plane_1$ and the $Plane_0$ when $2 \le i < S$ and $0 < j < L$ are given in equations 8 and 9, respectively. However, the more specific transitions and all balance equations for $Plane_1$ as well as $Plane_0$ can be found in Appendix B.

$$P_{i,j,1} = \frac{min(i,(j+1))\mu P_{i,j+1,1} + (i+1)\xi P_{i+1,j,1} + \lambda P_{i,j-1,1}}{\lambda + min(R,S-i)\eta + min(i,j)\mu + i\xi} + \frac{min(R,S-i)\eta P_{i-1,j,1}}{\lambda + min(R,S-i)\eta + min(i,j)\mu + i\xi} \tag{8}$$

$$P_{i,j,0} = \frac{(i+1)\xi P_{i+1,j,0} + \lambda P_{i,j-1,0}}{\lambda + i\xi} \tag{9}$$

Therefore, the final values of $P_{i,j,0}$ and $P_{i,j,1}$ can also be obtained using all balance equations in Appendix B. Please note that the general balance equations are derived for both planes separately. However, the relation between both planes and balance equations are also required for the proposed system in order to obtain more accurate and correct results which are presented in the next section.

3.2.2 Essential Balance Equations

In order to obtain correct steady state probabilities of the proposed system, the essential balance equations have to be considered. Figure 6 indicates the relation between two planes in the proposed model.

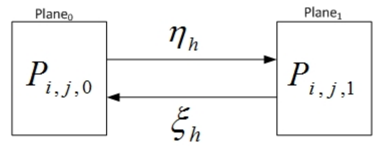


Fig. 6: The relation between two planes in the proposed model

Hence, as in Fig. 6, $\eta_h P_{i,j,0} = \xi_h P_{i,j,1}$. Since this applies to all states on both planes, the relation between two planes can be written as follows:

$$\eta_h Plane_0 = \xi_h Plane_1 \tag{10}$$

Hence, the relation between two planes with essential balance equations for each state can be obtained by using given relation in equation 10. Thus, similar to previous balance equations, the generalized balance equations for plane $Plane_1$ and $Plane_0$ taking into account the effect of $Plane_0$ and $Plane_1$ when $2 \le i < S$ and $0 < j < L$ are given in equations 11 and 12, respectively. However, the more specific essential balance equations obtained for $Plane_1$ and $Plane_0$ taking into account the effect of $Plane_0$ and $Plane_1$ for all value of *i* and *j* can be found in Appendix C.

$$P_{i,j,1} = \frac{min(i,(j+1))\mu P_{i,j+1,1} + (i+1)\xi P_{i+1,j,1} + \lambda P_{i,j-1,1}}{\lambda + min(R,S-i)\eta + min(i,j)\mu + i\xi + \xi_h} + \frac{min(R,S-i)\eta P_{i-1,j,1} + \eta_h P_{i-1,j,0}}{\lambda + min(R,S-i)\eta + min(i,j)\mu + i\xi + \xi_h} \tag{11}$$

$$P_{i,j,0} = \frac{(i+1)\xi P_{i+1,j,0} + \lambda P_{i,j-1,0} + \xi_h P_{i+1,j,1}}{\lambda + i\xi + \eta_h} \tag{12}$$

Thus, in order to obtain all accurate $P_{i,j,n}$s, the iterative procedure is applied and given in algorithm 1. First S, L, R, λ, μ_h, μ, η_h, η, ξ_h, ξ, Δ, and the number of iterations can be defined as input. Using Equation 7 the approximate steady state probabilities of the $Plane_1$ can be evaluated to get faster computations and more accurate results. Equations in Appendix B and C can then be used to calculate rough steady state probabilities of the system considered for both planes to have a faster convergence when the essential balance equations are employed. Please note that $QoS_{converge}$ and QoS_{approx} are general terms for calculating MQL, THRP and MRT in algorithm 1. For example, $MQL_{converge}$ can be obtained by using mathematical relations in application as shown in steps 4 to 9. Then, the correct steady state probabilities, $P_{i,j,n}$ can also be calculated by using the equations in Appendix B and C. Steps 10 to 11 are repeated until the normalization condition is satisfied. Once the sum of all $P_{i,j,n}$s converges to 1, all QoS outputs can be calculated. Thus, as an example, MQL calculation can be obtained by checking Δ, where Δ is taken as 0.001 in this paper, from steps 14 to 27. Thus, the iteration will be ended when $|MQL_{approx} - MQL_{converge}| \le \Delta$, or the iterative approach will assign the recent values of performability measures to previous values i.e, $MQL_{converge} = MQL_{approx}$ and continue from step 14 separately for both planes. At the end the QoS output parameters MQL, $MQL = MQL_0 + MQL_1$, THRP, $THRP = THRP_0 + THRP_1$ and MRT, $MRT = MRT_0 + MRT_1$ can be calculated using the recent $P_{i,j,n}$ as:

$$MQL = \sum_{i=0}^{S-1} i \sum_{j=0}^{L} P_{i,j,0} + \sum_{i=1}^{S} i \sum_{j=0}^{L} P_{i,j,1} \tag{13}$$

$$THRP = \sum_{i=0}^{S-1} \sum_{j=0}^{L} j\mu P_{i,j,0} + \sum_{i=1}^{S} \sum_{j=0}^{L} j\mu P_{i,j,1} \tag{14}$$

$$MRT = \frac{MQL}{THRP} \tag{15}$$

Algorithm 1: Iterative solution procedure

Input: S, L, R, λ, μ_h, μ, η_h, η, ξ_h, ξ, Δ
Output: MQL, THRP, MRT

```
1  Evaluate the approximate steady state probabilities by
     using 7.;
2  Obtain P_{i,j,1} of the Plane_1 to get faster computations
     and more accurate results.;
3  Equations in Appendix B and C are then used to
     calculate rough P_{i,j,1} ;
4  QoS_converge = 0.0;
5  for (i=0; i≤ S; i++) do
6    | for (j=0; j≤ L; j++) do
7    |   | QoS_converge = QoS_converge +j * P_{i,j,n}
8    | end
9  end
10 Calculate correct P_{i,j,n}s by using the equations in
     Appendix B and C;
11 Apply normalization;
12 Steps 10 to 11 are repeated until the normalization
     condition is satisfied;
13 The sum of all P_{i,j,n}s converges to 1;
14 while |QoS_approx − QoS_converge| ≤ Δ do
15   | QoS_converge = 0.0;
16   | for (i=0; i≤ S; i++) do
17   |   | for (j=0; j≤ L; j++) do
18   |   |   | QoS_converge = QoS_converge +j * P_{i,j,n}
19   |   | end
20   | end
21   | if |QoS_approx − QoS_converge| = 0,001 then
22   |   | Assign the recent values of QoS measures to
     |   |   previous values;
23   |   | i.e, QoS_converge = QoS_approx;
24   | else
25   |   | go back to the beginning of step 14;
26   | end
27 end
28 Evaluate the QoS output parameters MQL, THRP, MRT.
```

4 Numerical Results and Discussions

This section presents numerical results of large-scale HPCC considering multi-repairmen analysis along with availability of the head node as well as the computing nodes. Numerical results are presented for mean queue length, throughput, and mean response time in order to show the capabilities of the proposed analytical model and the solution approach.

Please note that the rest of the analysis presented in this paper considers the optimum number of repairmen in order to get more realistic QoS results for each experiment. The assumptions and parameters used in [30], [31], [34]–[36], [38]–[40], [46]–[49] are also employed in this paper for consistency. However, the proposed analytical model and solution approach can be easily applied to any similar system. For instance, the failure and repair times of the computing nodes are taken from studies [35] and [36] taking into account the availability of HPC clusters as well as cloud computing systems for consistency. Therefore, time between failures for head node and computing nodes can be taken as 250 (ξ=ξ_h= 0.004/h), 500 (ξ=ξ_h= 0.002/h), and 1000 (ξ=ξ_h= 0.001/h) hours. The parameters used for computations are μ=0.25 requests/sec, η=η_h=0.5/hr, ξ=ξ_h=0.001/hr and the λ rate per user, which varies from 10 requests per second unless stated otherwise. In addition, all numerical results presented in this paper are obtained using PCs with Intel(R) Core(TM) i7-363QM CPU @ 2.40GHz, 16GB RAM, running a 64-bit operating system.

For system reliability, an important parameter of large scale HPCC is the number of repairmen. As shown in [35] the number of R affects the system performance significantly. Thus, the multi-repairmen analysis is also included in this paper in order to find the optimum number of R to obtain desired QoS outputs for such systems. To illustrate the effect of multi-repairmen on the system performance, the throughput and mean queue length results are presented as a function of arrival rate in Table 2, Figs. 7 and 8.

TABLE 2: Throughput results as a function of mean arrival rate to obtain optimum multi-repairmen for the given scenario. S=200, L=600, $\mu = \mu_h$=1 req/sec, $\eta = \eta_h$=0.5/h, $\xi = \xi_h = 0.001/h$

λ	R=1	R=2	R=3	R=4	R=5	R=6	R=50
10	20	20	20	20	20	20	20
20	39.96	40	40	40	40	40	40
30	49.92	60	60	60	60	60	60
40	50	79.99	80	80	80	80	80
50	50	96.84	100	100	100	100	100
60	50	99.97	119.99	120	120	120	120
70	50	100	139.15	140	140	140	140
80	50	100	149.04	159.99	160	160	160
90	50	100	149.98	179.49	180	180	180
100	50	100	149.99	188.03	194.53	195.63	195.72

As shown in Table 2 and Figs. 7, 8, R=1 gives the worst QoS results in terms of MQL and THRP when S=200 and S=512. The system performance increases when R increases to 2, 3 and 4 in Table 2. However, when R is increased further, it can be observed that the system performance does not change, rather it stays constant. It is clear from Table 2 that THRP results for R=6 and R=50 are same whereas negligibly different from results when R=5. Considering R as 5, 6 and 50, the THRP results are 194.53 req/sec, 195.63 req/sec and 195.72 req/sec for λ=100 in Table 2, respectively. The results from Table 2 and Figs. 7, 8 show that there is a threshold value of R based on η, ξ and the number of computing nodes. Hence, R$\eta \geq$ Sξ is one of the essential conditions to obtain more realistic QoS results in HPCCs. In addition, when the condition: R$\eta \geq$ Sξ is satisfied and the threshold

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TCC.2021.3055119

value is obtained, the differences of the performance results are very small or negligible. Thus, the optimal number of repairmen for the HPCC considered in Table 2 is 5.

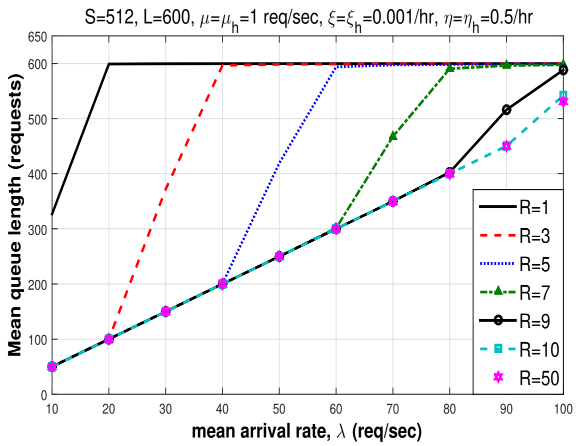


Fig. 7: Mean queue length results as a function of λ with different R

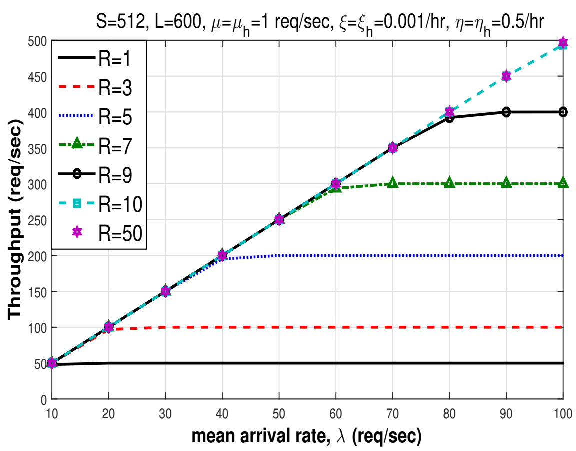


Fig. 8: Throughput results as a function of λ with different R

Similarly, MQL and THRP results are shown in Figs. 7 and 8, respectively in order to optimize R for a large-scale HPCC system (S=512). In this experiment, the number of computing nodes is increased from 200 to 512. In order to do a fair comparison, other parameters are kept at the same values. As shown in from figures, R=10 is the optimum number of repairmen to get desired QoS outputs. Obviously, in order to achieve high service quality in large-scale HPCCs, it is necessary to increase the number of repairmen when the number of computing nodes increases until it reaches the threshold value.

Figs. 9-11 present MQL, THRP, and MRT results as a function of λ for proposed analytical model and DES, respectively with different computing node failure rates. A system with S=500, and L=1000 is considered. The figures clearly show that the effect of failure of computing nodes on the QoS of the system is quite significant in large scale fault-tolerant HPCCs. The system quickly reaches full utilization when a higher value of ξ is considered. However, the considered HPCC can serve the arriving requests due to the large

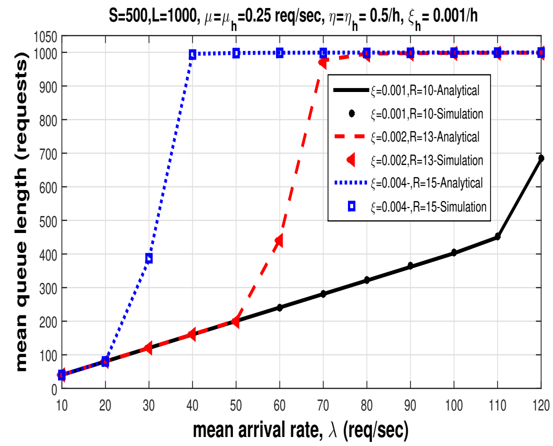


Fig. 9: Mean queue length results as a function of λ with different values of ξ

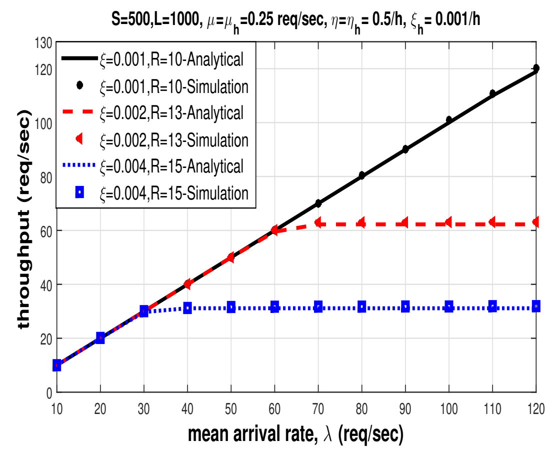


Fig. 10: Throughput results as a function of λ with different values of ξ

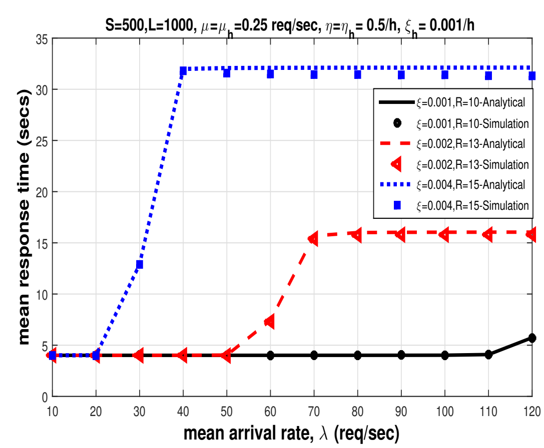


Fig. 11: Mean response time results as a function of λ with different values of ξ

number of computing nodes for lower values of ξ. In other words, increasing the time between failures of computing nodes decreases the MQL as shown in Fig. 9. For instance, in Fig. 9, the MQL value is 280.924 when ξ=0.001 and $\lambda = 70$. However, increasing the failure rate of computing nodes to 0.002 and 0.004 when $\lambda = 70$, the MQL values increase to 976.147 and 999.188, respectively. In addition, the average number of requests in the system reaches the maximum capacity of the system, L, when the failure rate increases. It can be clearly observed in Fig. 10 that THRP of the system increases as λ increases. However, the THRP reaches saturation after certain values of λ, depending on the failure rates of the computing node. Thus, the system cannot serve the requests efficiently and requests start to queue up to be served in the system especially for the systems that are highly loaded. Higher THRP values are obtained for the systems with lower failure rates due to the average value of operative computing nodes. On the other hand, similar behavior is observed for MRT in Fig. 11. The MRT increases when the failure rate of computing node increases as expected. Moreover, the optimum R value is also obtained in order to get best QoS results for each experiment as the failure rate of computing nodes changes in Figs. 9-11. Hence, optimum R is obtained as 10, 13, 15 for ξ=0.001, ξ=0.002, and ξ=0.004, respectively. The effects of the failure and the repair rates of head node are given in Figs. 12 and 13, respectively. Figure 12 shows the THRP results as a function of arrival rate for different head node failure rates. The THRP results decrease when ξ_h=0.01; this is due to the increased failure rate of the head node since the computing nodes directly depend on the head node. Even though a good repair facility is provided to the system ($\eta = \eta_h$=0.5/h), the QoS degrades due to the failure rate of the head node. In addition, the MQL results are presented in Fig. 13 as a function of arrival rate for different repair rates.

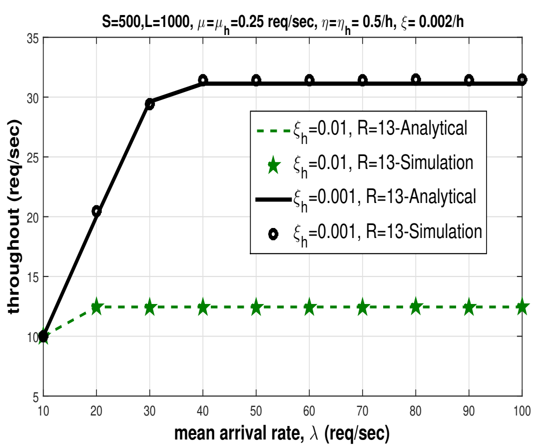


Fig. 12: Throughput results as a function of λ with different values of ξ_h

The significance of the head node repair rates is clearly shown in the figure in terms of the mean queue length. In other words, when the system has a moderate traffic ($\lambda = 70$), the MQL is 281.849. However, for the same situation the MQL is almost full (999.189) due to the increased failure of the head node. In the case of frequent failure of the head node, the rest of the computing nodes are no longer able to serve. Hence, the MQL of the system increases rapidly and the THRP of the system decreases because the system does not serve. If a head node is unable to serve (fails), the system stops serving the requests. The head node

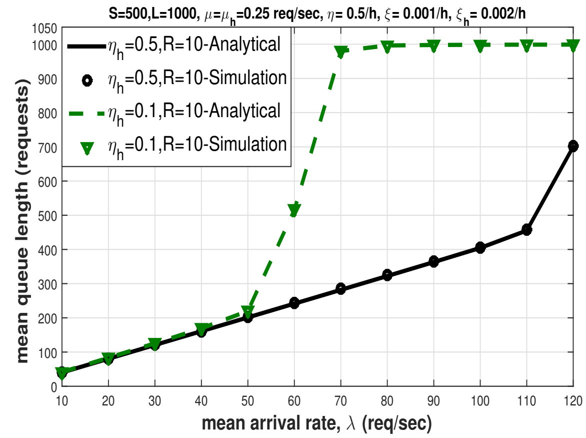


Fig. 13: Mean queue length results as a function of λ with different values of η_h

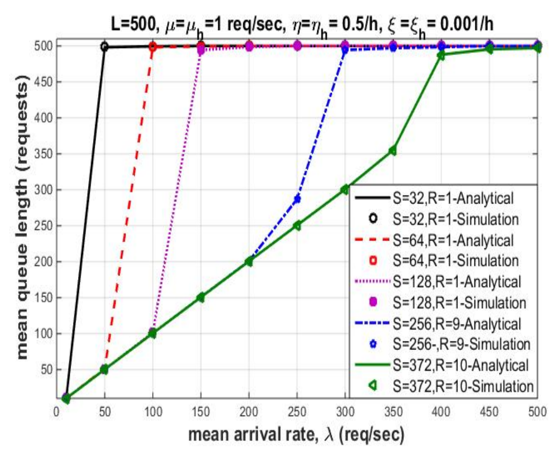


Fig. 14: Mean queue length results as a function of λ with different number of nodes

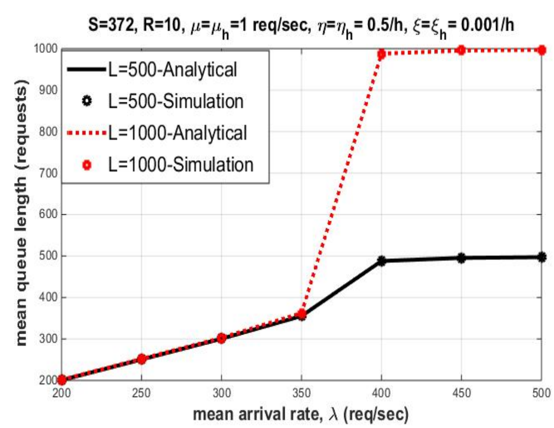


Fig. 15: Mean queue length results as a function of λ with different queue capacities

may regain connectivity and rejoin the system after some time. The results for this situation are shown in Fig. 13. Hence, as a summary, the repair and failure rates of the head node have a notable effect on the system performance.

Figure 14 shows the MQL results as a function of arrival rate for different number of computing nodes. The various

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TCC.2021.3055119

TABLE 3: Comparison of MQL, THRP and MRT results of proposed model and DES for S=500, L=1000, $\mu = \mu_h$=0.25 requests/sec, $\eta = \eta_h$=0.5/h, $\xi = \xi_h = 0.001/h$, D=Discrepancy

	Mean Queue Length			Throughput			Mean Response Time		
λ	Analytical	DES	$D(\%)$	Analytical	DES	$D(\%)$	Analytical	DES	$D(\%)$
10	40.12	40.45	0.83	9.99	10.01	0.10	4.01	4.04	0.73
20	80.23	80.42	0.22	20	20.12	0.59	4.01	3.99	0.37
30	120.36	121.76	1.14	29.99	30.05	0.17	4.01	4.05	0.97
40	160.48	160.82	0.21	39.99	40.10	0.25	4.01	4.01	0.04
50	200.60	200.10	0.25	49.99	50.02	0.04	4.01	4.00	0.29
60	240.75	240.15	0.24	59.99	60.12	0.20	4.01	3.99	0.44
70	280.92	281.25	0.11	69.99	70.01	0.01	4.01	4.01	0.10
80	321.15	322.56	0.43	79.99	80.42	0.52	4.01	4.01	0.09
90	361.51	365.16	0.99	89.99	90.12	0.13	4.01	4.05	0.85
100	402.24	404.48	0.55	99.98	100.92	0.92	4.02	4.01	0.37
110	449.53	452.32	0.61	109.97	110.72	0.67	4.08	4.08	0.06
120	682.03	684.75	0.39	118.87	120.18	1.09	5.73	5.69	0.69

TABLE 4: Comparison of MQL, THRP and MRT results of proposed model and simulation for S=1000, L=2000, $\mu = \mu_h$=0.25 requests/sec, $\eta = \eta_h$=0.5/h, $\xi = \xi_h = 0.001/h$, D=Discrepancy

	MQL			MRT			THRP		
λ	DES	Analytic	D(%)	DES	Analytic	D(%)	DES	Analytic	D(%)
50	200.599	200.607	0.004	4.01	4.01	0.038	50.016	49.999	0.033
100	402.304	402.399	0.023	4.022	4.024	0.044	100.02	99.999	0.02
150	1993.03	1994.709	0.084	13.284	13.803	3.763	150.031	144.507	3.822

TABLE 5: Comparison of the CPU times of proposed model and DES for S=500, L=1000, $\mu = \mu_h$=0.25 requests/sec, $\eta = \eta_h$=0.5/h, $\xi = \xi_h = 0.001/h$

	$\xi = 0.001$		$\xi = 0.002$		$\xi = 0.004$	
λ	Analytical(sec)	DES(sec)	Analytical(sec)	DES(sec)	Analytical(sec)	DES(sec)
10	3006.295	6012.59	3056.45	7513.74	3389.355	6268.16
20	3343.078	11217.54	3417.078	13733.21	3309.25	13094.34
30	3200.668	16381.27	3564.879	16773.36	3455.56	15938.84
40	3207.955	16415.91	3407.955	18724.66	3745.23	16102.76
50	3556.91	17113.82	3656.91	19567.83	3856.963	16231.36
60	3679.825	17359.65	3629.825	17562.31	3880.412	16437.25
70	3737.205	17474.41	3837.205	18720.11	3996.45	17076.91
80	3788.445	17576.89	3988.445	112262.7	4215.53	17587.06
90	3602.6	17565.46	3745.433	113542.6	4422.64	17586.19
100	3835.69	17671.38	3935.69	19358.46	4424.52	19682.15
110	3261.104	17826.65	3344.438	19600.47	4439.09	20043.66
120	3379	33793	38752	202842.4	4459.36	35186

TABLE 6: Comparison of the CPU times of proposed model and DES for S=1000, L=2000, $\mu = \mu_h$=0.25 requests/sec, $\eta = \eta_h$=0.5/h, $\xi = \xi_h = 0.001/h$

λ	Analytical (sec.)	DES(sec.)
50	18320.877	76930.53
100	15860.703	80878.871
150	16440.566	79505.016

numbers of computing nodes (S=32,64,128,256 and 372) are taken from [35], [36], [48], [49] as presented in Fig. 14. It can be seen that the proposed model and solution can easily handle the large amount of computing nodes: up to several hundreds. Fig. 15 shows the MQL results as a function of arrival rate for different queue capacities. It is clearly seen from the figure that queue capacity is the limiting factor of large scale HPCCs.

A comparative study is further performed in order to show a certain degree of accuracy of the proposed solution approach and DES. Tables 3 and 4 present MQL, THRP and MRT results comparatively with the simulation results for S=500, L=1000 and S=1000, L=2000, respectively.

The differences between the proposed analytical model and DES are also presented in all the figures. The maximum discrepancies for MQL, THRP and MRT are less than 1.149%, 3.82%, and 3.76% in Tables 3 and 4, respectively. It is clearly seen that discrepancies are well within the 5% confidence interval of the simulation. In this paper, the DES is mainly used for the validation purposes, however it can also be used for the QOS evaluation of such systems as it simulates the actual scenario rather than the Markov models presented. The DES model is implemented in C++ and adopted for the scenario considered in [42]. In addition, the CPU times of the analytical approach and DES for the computations are also presented comparatively in Tables 5 and 6. The proposed 3D analytical model uses an iterative approach to obtain steady state probabilities based on (S)x(L+1) number of equations for both $Plane_0$ and $Plane_1$. For instance, the number of states is considered in Table 6 is (1000 x 2001) x (1000 x 2001).

Thus, the processing time of analytical models are also presented with the DES processing time for comparison purposes. Tables 5 and 6 show the CPU times of systems with S=500, L=1000, and S=1000, L=2000, respectively. The computational efficiency of the proposed solution approach with the DES is clearly given in both tables in terms of

CPU times. For instance, in Table 6, 80878.871 seconds (22.46 hours) is the longest CPU time for DES for S=1000, L=2000. However, the longest CPU time of the analytical approach is less than 5 hours for an extreme case. Thus, the proposed analytical model and an iterative solution approach are efficient in QoS evaluation of HPCCs with a large number of nodes and an optimum R.

5 Conclusion

This paper has proposed an analytical model and an iterative solution which obtains more realistic QoS results by finding optimum number of repairmen for large-scale HPCCs in the MCC environment without encountering a state explosion problem. The performability models were developed which considered both performance and availability models for large-scale HPCCs. In addition, the optimum number of repairman is obtained for more realistic QoS in such systems. The QoS results obtained from the analytical model are compared with DES results.

The main focus of this work is performability output parameters such as MQL, THRP and MRT. The obtained numerical results clearly show that, the proposed model and proposed solution approach can easily handle a large number of states (i.e, several billion states such as (1000 x 2001) x (1000 x 2001)) in HPCCs without encountering the state explosion problem. Findings show that the analytical models and approximate solution approach presented in large-scale HPCCs are in good agreement with DES. The comparative results show that the discrepancy between the DES and the analytical model is less than 5% for all cases. Hence, the analytical models and the solution approach used in this paper can be adopted and mapped to many other practical, fault-tolerant large-scale systems. In addition, the computation time of the proposed model is significantly reduced in comparison to DES, especially for loaded and large-scale cases.

A major hindrance to using this method is the large number of equations. In the proposed model, the balance equations depend on each other and are chained together for obtaining steady state probabilities. An iterative technique was used to solve the steady state probabilities. This technique increases the computation times for results. Furthermore, increasing the queue capacity, the number of computing nodes, or mean arrival rate force a significant increase in computation times. Therefore, it is essential that programming techniques are effectively used to further reduce computation times. Although the computational speed is an issue, the proposed method is superior to DES under all circumstances. Results show that proposed method works with large queue capacities and large numbers of computing nodes such as as 500 to 1000 for both effectively while giving accurate results. A potential future extension of this work would involve the incorporation of heterogeneous nodes which will result in multiple service rates that would affect the performance of the overall system. In addition, the effect of user mobility in MCC environment is another future concern.

References

[1] X. Yang, et al. "Cloud computing in e-Science: research challenges and opportunities," *The Journal of Supercomputing*, vol.70, no.1, pp. 408-464, 2014.

[2] Y. Wang, I.R. Chen, and D.C. Wang, "A Survey of Mobile Cloud Computing Applications: Perspectives and Challenges," *Wireless Personal Communications*, vol.80, no.4, pp. 1607-1623, 2015.

[3] A. Ahmad, A. Paul, S. Din, M. M. Rathore, G. S. Choi, and G. Jeon, "Multilevel data processing using parallel algorithms for analyzing big data in high-performance computing," *International Journal of Parallel Programming*, vol.46, no.3, pp. 508-527, 2018.

[4] A. Boukerche, R. Al-Shaikh and M. S. M. Notare, "Towards building a highly-available cluster based model for high performance computing," *Proceedings 20th IEEE International Parallel and Distributed Processing Symposium*, pp. 1-8, 2006, doi: 10.1109/IPDPS.2006.1639633.

[5] J. O. Fajardo, et al., "Introducing mobile edge computing capabilities through distributed 5G cloud enabled small cells," *Mobile networks and applications*, vol.21, no.4, pp. 564-574, 2016.

[6] A. Botta, W. Donato, V. Persico, and A. Pescapé, "Integration of Cloud computing and Internet of Things," *Future Generation Computer Systems*, vol.56, pp. 684-700, March 2016.DOI: https://doi.org/10.1016/j.future.2015.09.021

[7] D. Bruneo, et al., "A stochastic model to investigate data center performance and qos in iaas cloud computing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol.25, no.3, pp. 560-569, 2014.

[8] E. Constantinos, and N. Hill, "Cloud Computing for parallel Scientific HPC Applications: Feasibility of Running Coupled Atmosphere-Ocean Climate Models on Amazon's EC2," *In The 1st Workshop on Cloud Computing and its Applications*, Chicago, IL. vol.34, pp. 20-38, 2008.

[9] D. Eddelbuettel, "CRAN Task View: High-Performance and Parallel Computing with R," 2019.

[10] R. Buyya, et. al., "High Performance Cluster Computing: Architectures and Systems," Prentice Hall PTR, Upper Saddle River, NJ, USA, 1999.

[11] C. Z. Qin, "A strategy for raster-based geocomputation under different parallel computing platforms," *International Journal of Geographical Information Science*, vol. 28. no.11, pp. 2127-2144, 2014.

[12] C. Grelck, E.N. Szynkiewicz, M. Aldinucci, A. Bracciali, E. Larsson, "Why High-Performance Modelling and Simulation for Big Data Applications Matters," *High-Performance Modelling and Simulation for Big Data Applications. Lecture Notes in Computer Science*, vol. 11 no. 400. Springer, Cham, 2019.

[13] W. Miao, G. Min, Y. Wu, H. Wang, and J. Hu, "Performance Modelling and Analysis of Software-Defined Networking under Bursty Multimedia Traffic," *ACM Trans. Multimedia Comput. Commun. Appl*, vol. 12, no. 77, pp. 1-19, 2016.

[14] J. Bhimani, N. Mi, M. Leeser and Z. Yang, "FIM: Performance Prediction for Parallel Computation in Iterative Data Processing Applications,"*IEEE 10th International Conference on Cloud Computing (CLOUD)*, Honolulu, CA, pp. 359-366, 2017.

[15] R. Mahmud, R. Kotagiri, and R. Buyya,"Fog computing: A taxonomy, survey and future directions," *Internet of Everything. Springer*, Singapore, pp. 103-130, 2018.

[16] Y. Yan, and W. Su,"A fog computing solution for advanced metering infrastructure," *In IEEE/PES Transmission and Distribution Conference and Exposition*, pp.1-4, May 2016.

[17] A. S. D. Alfoudi, S. H. S. Newaz, R. Ramlie, G. M. Lee and T. Baker, "Seamless Mobility Management in Heterogeneous 5G Networks: A Coordination Approach among Distributed SDN Controllers", IEEE 89th Vehicular Technology Conference, Kuala Lumpur, Malaysia, pp. 1-6, 2019.

[18] C. Yu, J. Lan, Z. Guo, Y. Hu and T. Baker, "An Adaptive and Lightweight Update Mechanism for SDN," *IEEE Access*, vol. 7, pp. 12914-12927, 2019, doi: 10.1109/ACCESS.2019.2893058.

[19] M. Al-khafajiy, T. Baker, "H. Al-Libawy, Z. Maamae, M. Aloqaily, Y. Jararweh, Improving fog computing performance via fog-2-fog collaboration," *Future Gener. Comput. Syst.*, 100, pp. 266-280, 2019.

[20] I. A. Ridhawi, Y. Kotb, M. Aloqaily, Y. Jararweh and T. Baker, "A Profitable and Energy-Efficient Cooperative Fog Solution for IoT Services," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3578-3586, May 2020.

[21] D. U. Junping, et al., "Pro visioning a cluster of distributed computing platform based on placement strategy," U.S. Patent No 9, 268, 590, 2016.

[22] R. Buyya, "High Performance Cluster Computing: Architectures and Systems", 1999, Prentice Hall PTR, Upper Saddle River, NJ, USA.

[23] J. Ekanayake, and F. Geoffrey, "High performance parallel computing with clouds and cloud technologies," *Cloud Computing*, Springer Berlin Heidelberg, pp. 20-38, 2010.

[24] E. Ever, O. Gemikonakli, and R. Chakka, "A Mathematical Model for Performability of Beowulf Clusters," *In Proceedings of* 39^{th} *Annual Symposium on Network Simulation*, pp. 118-126, 2006.

[25] J. Adams, and D. Vos, "Small-College Supercomputing: Building A Beowulf Cluster At A Comprehensive College," *In Proceedings of* 33^{rd} *Technical symposium on computer science education*, pp. 411-415, 2002.

[26] E. Ever, O. Gemikonakli, and R. Chakka, "Analytical modelling and simulation of small scale typical and highly available Beowulf clusters with breakdowns and repairs," *Simulation Modelling Practice and Theory*, vol. 17, pp. 327-347, 2009.

[27] I. Haddad, C. Leangsuksun, S. L. Scott, HA-OSCAR: the birth of highly available OSCAR, Linux Journal, 1-115, 2003.

[28] T. Thanakornworakij, High availability on cloud with HA-OSCAR, Parallel Processing Workshops. Springer Berlin Heidelberg, 2012.

[29] E. Ever, O. Gemikonakli, A. Kocyigit, and E. Gemikonakli, "A hybrid approach to minimize state space explosion problem for the solution of two stage tandem queues," *J. Network and Computer Applications*, vol.36, no. 2, pp. 908-926, 2013.

[30] J. Vilaplana, "The cloud paradigm applied to e-Health. BMC medical informatics and decision making," vol. 13, no. 1, 2013.

[31] H., Song, C. Leangsuksun, and R. Nassar, "Availability Modeling and Evaluation on High-Performance Cluster Computing Systems," *Journal of Research and Practice in Information Technology*, vol. 38. no. 4, pp. 317-335, 2006.

[32] J. Vilaplana, F. Solsona, I. Teixidó, J. Mateo, F. Abella, and J. Rius, "A queuing theory model for cloud computing," *The Journal of Supercomputing*, pp. 1-16, 2014.

[33] Y. Ma, J. Han, K. S. Trivedi, "Composite performance and availability analysis of wireless communication networks," *IEEE Trans. Veh. Technol.*, 50(5), pp. 1216-1223, 2001.

[34] H. Khazaei, J. Misic, and B. M. Vojislav, "Performance Analysis of Cloud Computing Centers Using M/G/m/m + r Queuing Systems," *IEEE Transactions on Parallel And Distributed Systems*, vol. 23, no. 5, pp. 936-943, 2012.

[35] Y. Kirsal, "Analytical modelling and optimization analysis of large-scale communication systems and networks with repairmen policy," *Computing*, vol. 100, no.5, pp. 503-527, 2018.

[36] E. Ever, "Performability analysis of cloud computing centers with large numbers of servers," *The Journal of Supercomputing*, pp. 1-27, 2016.

[37] Ever, E., Fault tolerant two stage open queuing systems with server failures at both stages, *IEEE Communication Letters*, 2014, 18, 1523-1526.

[38] E. Ever, O. Gemikonakli, R. Chakka, "A Mathematical Model for Highly Available Clusters with One Head and Several Identical Computing Nodes," In Proceedings of the 9^{th} International Conference on Computer Modelling and Simulation, pp. 32-37, 2006.

[39] A. Song, W. Wang, and J. Luo, "Stochastic modeling of dynamic power management policies in server farms with setup times and server failures," *International Journal of Communication Systems*, vol. 27, pp. 680-703, 2014.

[40] O. Gemikonakli, T. V. Do, R., Chakka., and E. Ever, "Numerical Solution to the Performability of a Multiprocessor System with Reconfiguration and Rebooting Delays," In Proceedings of ECMS, pp. 766-773, 2005.

[41] M., Tschaikowski and M. Tribastone, "Tackling continuous state space explosion in a markovian process algebra," *Theoretical Computer Science*, 2014, vol. 517, 1-33.

[42] J. Banks, J. Carson, and B. Nelson, Discrete-Event System Simulation. Prentice Hall Englewood Cliffs. NJ. USA. 2000.

[43] I, Mitrani, R. Chakka "Spectral expansion solution for a class of markov models: application and comparison with the matrix–geometric method," *Performance Evaluation*, 23(3), pp. 241-260, 1995.

[44] Z. Chen, G. Xu, V. Mahalingam, L. Ge, J. Nguyen, W. Yu, and C. Lu, "A cloud computing based network monitoring and threat detection system for critical infrastructures," *Big Data Research*, vol. 3, pp. 10-23, 2016.

[45] J. Wang, T. Wang, Z. Yang, N. Mi, and B. Sheng, "eSplash: Efficient speculation in large scale heterogeneous computing systems," *In 2016 IEEE 35th International Performance Computing and Communications Conference (IPCCC)*, pp. 1-8, December 2016.

[46] K. Xiong, and H. Perros, "Service performance and analysis in cloud computing, In Proceedings of IEEE World Conference Services," pp. 693-700, 2009.

[47] B. Yang, F. Tan, Y. Dai, and S. Guo, "Performance Evaluation of cloud service considering fault recovery," In Proceedings of the First International conference on cloud computing, pp. 571-576, 2009.

[48] Academic (UB-HPC) Compute Cluster Hardware Specs, The University of Buffalo, $https://www.buffalo.edu/ccr/support/research\_facilities/general\_compute/cluster-hardware-specs.html$

[49] High Performance Computing using Beowulf clusters $http://www2.hawaii.edu/~zinner/101/students/MitchelBeowulf/cluster.html$

[50] Y. Jin, Y. Wen, and W. Zhang, "Content Routing and Lookup Schemes Using Global Bloom Filter for Content-Delivery-as-a-Service," *IEEE Systems Journal*, vol. 8, no. 1, pp. 268-278, 2014.

[51] R. Chakka, *Performance and Reliability Modelling of Computing Systems Using Spectral Expansion*, Ph.D. Thesis, University of Newcastle, Upon Tyne, UK, 1995.

[52] B. Schroeder, and G. Gibson, "A large-scale study of failures in high performance computing systems," *IEEE Transactions on Dependable and Secure Computing*, 2010, 7(4), 337-350.

[53] R. K. Sahoo, A. Sivasubramaniam, M. S. Squillante, and Y. Zhang, "Failure data analysis of a large-scale heterogeneous server environment," International Conference on Dependable Systems and Networks, 2004, 772-781.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TCC.2021.3055119

YONAL KIRSAL received his MSc and PhD degrees from Middlesex University, London, UK in 2008 and 2013 respectively, after graduating from Electrical and Electronic Engineering Department of Eastern Mediterranean University (EMU), Fagamusta, Cyprus in 2006. He joined the Electrical and Electronic Engineering Department, European University of Lefke, as an Assistant Professor in 2014. He has been elected Head of the Electronics & Communication Engineering department since October 2017. He was appointed as Associate Professor in 2019. His main research is in the field of modelling of heterogeneous wireless networks, performance evaluation, queueing theory, network design and evaluation, Intelligent Transportation Systems (ITS).

YONEY KIRSAL EVER received the B.Sc. degree from Eastern Mediterranean University, Famagusta, Cyprus, in 2002, the M.Sc. degree in internet computing from University of Surrey, Guilford, Surrey, U.K., in 2003, and the Ph.D. degree from School of Engineering and Information Sciences in Middlesex University, London, U.K.,Also, in 2012, she completed the Post Graduate Certificate in higher education. Her research is on development of security strategies using Kerberos in wireless networks. She has worked as a part-time Lecturer while she was pursuing the B.Sc. and Ph.D. degrees, and as a Lecturer with the Department of Computer and Communications Engineering, Middlesex University London. She is currently an Assoc. Prof. Dr. at Near East University, Nicosia, Turkey. She has authored or co-authored international conference papers with various awards including IEEE best paper for promising research. Her research interest include network security, authentication protocols, and formal verification methods. Dr. Kirsal Ever has been a member of ACM since 2007. She reviews papers for various journals, mainly on network security.

GLENFORD MAPP received his BSc (First Class Honours) from the University of the West Indies in 1982, a MEng (Distinction in Thesis) from Carleton University in Ottawa, Canada in 1985 and a PhD from the Computer Laboratory, University of Cambridge in 1992. He then worked at AT&T Cambridge Laboratories for ten years before joining Middlesex University in London in 2003, where he is currently an Associate Professor. He was also a Visiting Research Fellow at the Computer Laboratory between 2003 and 2010 where he worked on several projects. His primary expertise is in the development of new technologies for mobile and distributed systems including vehicular networks for Connected and Autonomous Vehicles (CAVs). He has published over 100 papers in refereed journals and conferences. Glenford worked on Y-Comm, a new architecture for building future mobile networks. He is currently doing research on Future Internet using Y-Comm, SDN and NFV. He is also the Head of the Cooperative Intelligent Transport Systems (C-ITS) Research Group at Middlesex University and has been involved in building a number of Connected Vehicle testbeds in the UK.

MOHSIN RAZA is a senior lecturer at Department of Computer Science, Edge Hill University, UK. He completed his PhD at Math, Physics and Electrical Engineering Department, Northumbria University (NU), UK and BS (Hons) and MS degrees in Electronic Engineering from Mohammad Ali Jinnah University (MAJU), Pakistan. Prior to his work at Edge Hill University, he worked as a Lecturer (2019-20) at Northumbria University, UK, as a post-doctoral fellow (2018-19) at Middlesex University, UK, as a Demonstrator/Associate-Lecturer and Doctoral Fellow (2015-17) at Northumbria University UK, Junior lecturer (2010-12) and later as Lecturer (2012-15) in Engineering department at Mohammad Ali Jinnah University, Pakistan, and Hardware Support Engineer (2009-10) at Unified Secure Services, Pakistan. He served as a technical committee member for ICET 2012, SKIMA 2015, SKIMA 2017, WSGT 2017, CSNDSP 2018, SKIMA 2018, ICT 2019 and CSoNet 2019. He has also been a guest editor to special issue on Heterogenous Internet of Medical Things in Int. Journal of Distributed Sensor Networks and reviewer of several journals including IEEE Access, IEEE Comm. Letters, MDPI Sensors, Elsevier Vehicular Comms. and Springer AJSE. His research interests include IoT, 5G and wireless networks, autonomous transportation systems, machine learning, Industry 4.0 and digital twins.