Dynamic Hardware Redundancy Approaches Towards Improving Service Availability in Fog Computing

Sara Alraddady[1], Ben Soh[2], Mohammed AlZain[3], Alice Li[4]
Department of Computer Science and Information Technology-School of Computing, Engineering and Mathematical Sciences, La Trobe University, Australia[1, 2]
Department of Information Technology-College of Computers and Information Technology, Taif University, Kingdom of Saudi Arabia[3]
Department of Management & Marketing-La Trobe Business School, La Trobe University, Australia[4]

***Abstract*—The distributed nature of fog computing is designed to alleviate bottleneck traffic congestion which happens when a massive number of devices try to connect to more powerful computing resources simultaneously. Fog computing focuses on bringing data processing geographically closer to data source utilizing existing computing resources such as routers and switches. This heterogeneity nature of fog computing is an important feature and a challenge at the same time. To enhance fog computing availability with such nature, several studies have been conducted using different methods such as placement policies and scheduling algorithms. This paper proposes a fog computing model that includes an extra layer of duplex management system. This layer is designated for operating fog managers and warm spares to ensure higher availability for such a geographically disseminated paradigm. A Markov chain is utilized to calculate the probabilities of each possible state in the proposed model along with availability analysis. By utilizing the standby system, we were able to increase the availability to 93%.**

Keywords*—Fog computing; fault tolerance; Markov chain; hardware redundancy

I. INTRODUCTION

The main idea that led researchers to introduce fog computing was to increase systems availability and delivered quality of service to overcoming cloud computing limitations. Fog computing focuses on distributing data processing instead of relying on centralized computing resources which is cloud data centers. It was introduced in 2012 by Cisco and defined as "an extremely virtualized environment that delivers networking, storage, and compute resources between outdated cloud computing information centers, usually, but not entirely situated at the network edge [1]." The reference structure of fog computing was introduced later in 2017. The structure consists of three layers. Highest layer is for cloud data centers, lowest layer is where end users reside, and between these two layers, a layer designed for fog node devices [2]. Fog nodes are computing devices with limited capabilities compared to data centers. These nodes can be routers, switches, access points, vehicles, or personal computers. On the other hand, end users can be mobile devices, sensors, actuators, or vehicles. The name fog reflects that fog in weather is closer to the ground than clouds. Fog computing brings the computation process closer to the end user by leveraging all available computing resources at the periphery of the network. The decentralised nature of fog computing reduces the amount of data that needs to be sent to the cloud. Thus, higher quality of service (QoS) can be achieved. Given the nature of fog computing which includes heterogeneity and end users mobility, numerous studies were conducted by researchers to explore the potential of this new computing paradigm aiming to support users mobility and design context aware fog computing paradigms. However, fog systems availability did not receive much attention in cases of faults occurrence. fog computing systems availability must be properly addressed to mitigate service disruption impact since such interruption can be very financially expensive and, in some cases, can lead to fatalities.

Availability of fog computing paradigm is concerned with ensuring that the system is reachable to end users as much as possible. In such highly heterogeneous paradigm, system availability is divided into two main parts. The first part is related to the availability of cloud layer, which is the sustainability of cloud data centers and their networking. This part has been investigated by scholars over the years as surveyed in [3], [4], and [5]. Different mechanisms were incorporated to improve the availability of cloud computing such as hot migration, load balancing, resource management, and traffic management. The second part is concerned with fog nodes and their communication. Researchers have done some work related to enhance the overall performance of fog computing by resource management mechanisms, load balancing algorithms, designing mobility, energy, and context aware fog environment, reviewing different fog architectures, and only a few incorporated fault tolerance techniques into fog computing.

Fault tolerance FT concept is defined as systems survival attribute in which it can operate with the existence of faults in any part of it. FT is a mechanism used to achieve high availability, scalability, resilience, and reliability and can be found in many fields including aviation, military, telecommunication, and space missions. Faults can be results of several factors - some are external factors and other internal ones like incorrect algorithms. Various Fault Tolerance Techniques FTT have been introduced to minimize faults

manifestation in computing systems such as checkpointing, watchdog, and redundancy [6]. Fig. 1 summarizes fault tolerance techniques which include software and hardware techniques.

Aiming to increase fog computing availability, two fault tolerance techniques FTT are used. First, a management layer as a software FTT. This layer should be able to function independently from main cloud where contacting main cloud in the proposed work should be minimal. Main cloud communication is allowed to perform certain tasks such as long-term storage, history analysis, or complex computation. The efficiency of this layer is discussed in detail in [7]. Second, a standby system deployed at management layer as a hardware FTT since the management layer is considered as a backbone in the proposed model.

The remaining of the paper is structured as follows: Section II discusses related studies. Section III briefly, describes the software FTT in the proposed model followed by a duplex management system as a hardware FTT. Section IV provides a quantitative analysis for the proposed model using an example followed by an availability analysis in Section V. The comparison of proposed model with other models is given in Section VI and Section VII concludes the paper and highlights future work.

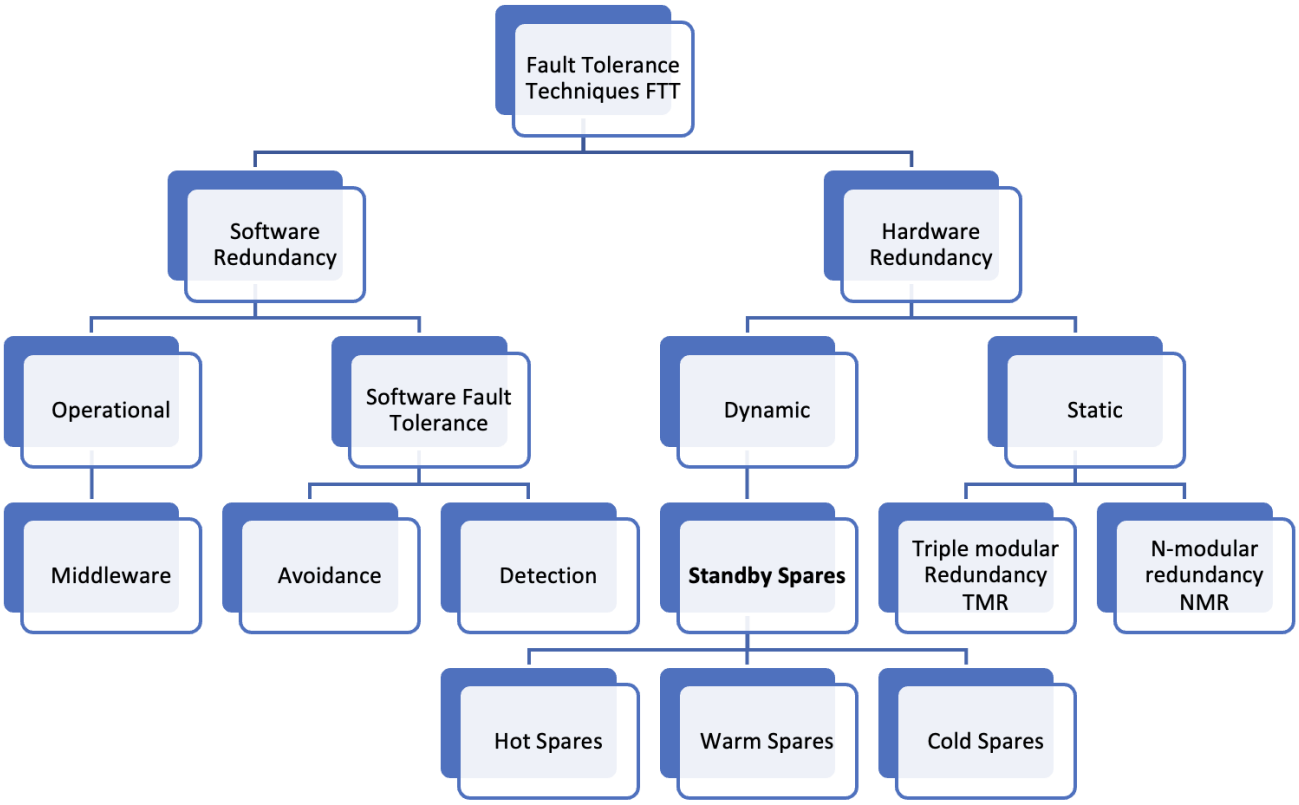

Fig. 1. Fault tolerance techniques FTT.

II. Literature Review

Fog computing is becoming popular since it provides a computing experience with low latency at low cost, and it is applicable to be deployed in various domains like healthcare, industries, and smart cities. To reach its potentials, researchers have been exploring several aspects of this new computing paradigm including: first, placement policies as in [8], [9], and [10]. Placement polices are responsible for deciding the most suitable computing node to process each single request. These policies can be context aware, energy aware or mobility aware. Second, several studies have been conducted on fog computing architecture as in [11], [12], [13], [14], and [15]. Some researchers increased the number of layers to reach six layers aiming to enhance the performance and robustness of fog computing. Third, scholars have been investing in fog resources management. Because of the heterogeneous nature of fog computing, balancing resources utilization and achieving the required quality of service must be addressed. Hence, authors in [16] propose load balancing algorithm to maximize resource utilization. However, searching for studies that focus on fault tolerance in fog computing, showed that service replication was the only FTT investigated by researchers. The main idea of service replication is to create replicas of data at the edge of the hierarchy. These replicas can be used in cases of fault occurrence. Authors in [17], propose a multi-tiers fog model, in which data replicas are generated at the same tier. The proposed model includes mobile agents that can roam all tiers to investigate failures when occurred, facilities communication between devices in different tiers and most importantly, fetch jobs with high priorities to be assigned. Simulating this model showed better performance and fault tolerance. Additionally, Javed et al. in [18], propose a fault tolerant architecture for edge applications which can save data when connection to cloud is lost. This architecture consists of

three layers which are: application isolation, data transport, and management layer.

They tested the performance of the proposed architecture on surveillance cameras and found that the model was able to tolerate losing two nodes out of five. Furthermore, Authors in [19] developed a service replication scheme that can sense what services need to be replicated and chooses the most suitable node to perform and store the replica.

After reviewing all the listed studies, we found out that hardware redundancy did not receive much attention. One of the reasons behind this goes to the fact that hardware redundancy involves extra cost. Also, incorporating hardware redundancy increase the level of complexity of any system. However, integrating extra hardware ensures higher availability and higher fault tolerance level because faults are inevitable whether internal or external ones. Accordingly, the idea of the proposed model in this paper arose, which addresses heterogeneity and resource management in fog computing and improve system availability using hardware FTT as the next section illustrates.

III. Proposed Model Design

Fog computing architecture mainly consists of three layers as mentioned before. In this paper, one more layer is added to the hierarchy that resides between fog node layer and cloud layer. The purpose of this layer is to governs fog nodes and their communications among each other and with cloud data centres. It consists of fog nodes that are called fog managers. Each fog manager has three modules: processing module, tracking module, and allocating module. It also, has a sending and receiving units for communication purposes as depicted in Fig. 2 [7].

Since this management layer is considered the backbone of the proposed hierarchy, any fault that occurs at this layer will lead to expensive consequences. Therefore, a standby system for this layer is designed. Generally, standby systems were designed for extremely high availability systems such as aircraft where service disruptions are not acceptable. Standby systems consist of three components: active node, standby node, and a switching unit as illustrated in Fig. 3. Techniques to implement a standby system fall into three categories: hot, warm, and cold. Tradeoffs between cost, energy consumption, and availability rate must be considered during design stage. When switching time is critical as the case in aircrafts, hot standby is the most suitable technique. In this technique, a completely fired up node is always ready to take over the failing node. Energy consumption for this technique is very high, yet higher availability is guaranteed. On the other hand, cold standby requires higher switchover time, consumes less energy since the standby node is not fully functioning and in sometimes is not fired up, and provides less availability than hot standby. It is suitable for industrial use and weapon systems. Lastly, warm standby which falls between hot and cold standby technique [20]. In this paper, a standby system is designed at management layer to increase availability in fog computing environment. To the best of our knowledge, designing a highly available fog computing environment using this fault tolerance technique was not investigated yet.

Accordingly, at management layer in the proposed model, a group of fog nodes reside in charge of managing fog nodes connected to it. Each cluster of fog nodes in a relatively small area is managed by a single fog node. In this proposed model, a number of standby nodes is added to optimize the availability as described in Fig. 4.

The proposed model is an active/passive standby system which denotes that only one node is in control. Passive/standby node steps into failover the active node. The switching technique between nodes is the common switching policy as illustrated in Fig. 5.

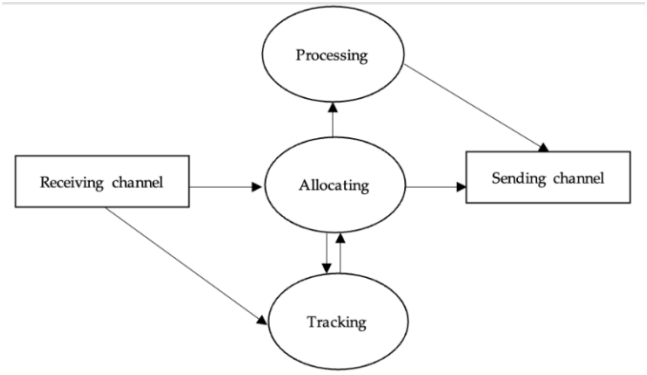

Fig. 2. Fog manager components.

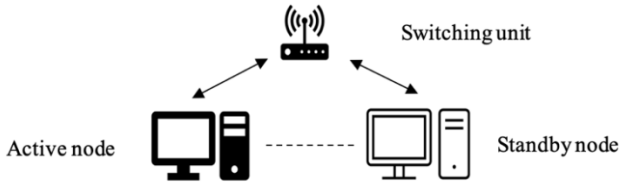

Fig. 3. Standby systems components.

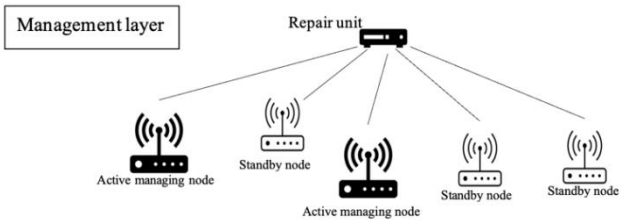

Fig. 4. Proposed standby system at management layer.

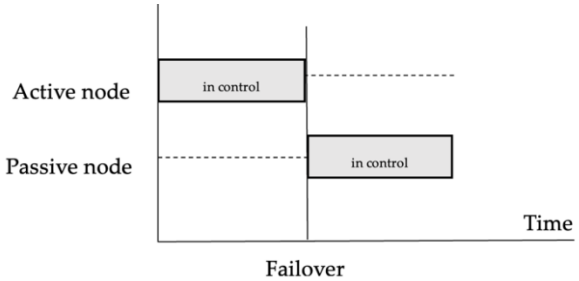

Fig. 5. Common switching policy for standby systems.

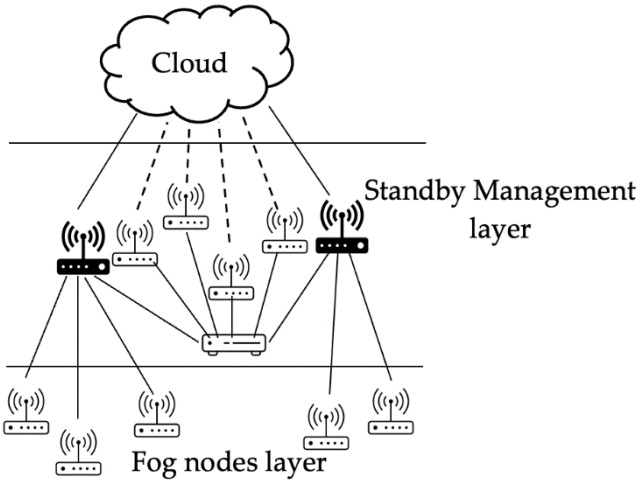

Fig. 6. Network connectivity in the proposed model.

Both the active/operating manager O_m and warm manager W_m are identical hardware components but differ in the failure rate. The failure rate of O_m is λ_o while $\lambda_o > 0$, and the W_m failure rate is λ_w where $0 < \lambda_w < \lambda_o$. Each standby manager has a unit with a Boolean value either up or down. When the unit value is up (one) the manager can be in an active mode or idle mode. A down or (zero) value indicates that the manager is not operating. Network connectivity assumptions for the proposed standby system are depicted in Fig. 6 as follows:

- Both active managers and standby mangers are securely connected the repair unit.
- Active managers are securely connected to cloud data centers.
- Standby managers have an established connection to cloud centers which will activated once the standby manager status switched to active.
- Active fog managers are connected to fog nodes at the next lower level of the models.
- Fog managers can establish secure connections among each other for higher resource utilization.

To illustrate and evaluate the switching mechanism between active and standby managers at the management layer, a Colored Petri net is designed. Petri nets PN are used to evaluate and analyze embedded systems performance, and they consist of places, transitions, and arcs. Places, which are in a circle, represent the states of a system; transitions, symbolized as rectangles, exist between places; and arcs demonstrate the workflow [21]. Usually, some places contain tokens that demonstrates the dynamics of the system. Even though Petri nets were invented at an earlier time, it is still used for its proficiency in designing sophisticated distributed computing systems as in [22]. Fig. 7 describes how the proposed standby system works in case an active manager fails. The red places represent failed managers and blue places signifies the completion of a process. Additionally, green transitions indicate the start of the moving process.

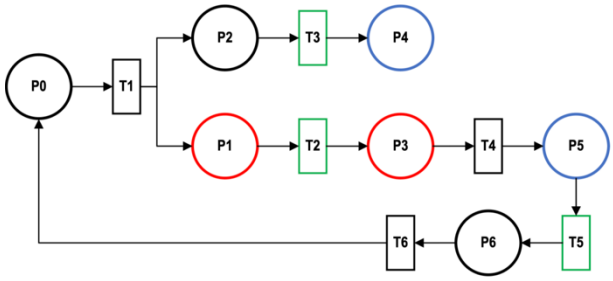

Fig. 7. PN design of the proposed standby system.

PN notations 1: There are six states for the proposed system components as follow:

P0: initial state

P1: failed fog manager moving to repair unit

P2: setting W_m in full operating mode with failure rate of λ_o instead of λ_w

P3: failed managers are moving to repair unit

P4: fog manager is active with a failure rate of λ_o

P5: failed managers are being fixed and restored as new components

P6: the manager is ready to use as a warm standby manager

PN notations_2:

T1: is enabled when an active manager has failed

T3: start moving the activated manager into fully operating manager

T2: start moving the failed manager to the repair unit

T4: end moving the failed manager to the repair unit

T5: enabled when the failed manager has been fixed

T6: start using fixed manager as a warm standby manager with failure rate of λ_w which takes the system to its initial state.

The initial state of the proposed model is represented by P0 where the active manager is in a fully functioning state with a failure rate of λ_o and the standby manager in warm state and with a failure rate of λ_w. When the active manager fails, which is represented by T1, the system performs two steps. First, it moves the failed manager to the repair unit (P1) and sends an order to the standby manager to be fully operating (P2). P3 represents the system state during moving the failed manager to the repair unit, and in P4, standby manager is now fully active with failure rate of λ_o and referred to as the active manager. States P3 and P4 happen at the same time as well as P1 and P2. Next, P5 represents the system state when the failed manager is being fixed or replaced depending on its condition. When P5 is completed, T5 is enabled. T5 transits the system to P6, in which the failed manager is fixed and ready to be active again. Subsequently, T6 is activated by P6 which takes the system back to its initial state.

IV. Qualitative Analysis of the Proposed Model

The proposed model consists of cloud data centers, operating fog manager O_m, warm fog managers W_m, fog manager, and one repair unit. In the next lower layer, fog nodes reside to provide the required services to end users. Operating fog managers manage the fog nodes with a failure rate of λo, and warm fog managers are in standby mode with a failure rate of λw. There are two chains of circles representing all possible states as illustrated in Fig. 8. Each circle has a number inside of it representing the number of failed managers in that state. The values on the arcs flowing in and out is associated with the status of the devices (failed/fixed) that changes the system state. Two chains of states coded as 1 and 2 to represent the possible state transitions for operating managers O_m and warm managers W_m where μ is the repair rate of the repair unit. Chain 1 represents the system states with 0, 1, …, O_m failed operating managers, and chain 2 represents the system states with failed warm managers starting from O_m+1 until $O_m + W_m$, which represents the total number of managers T.

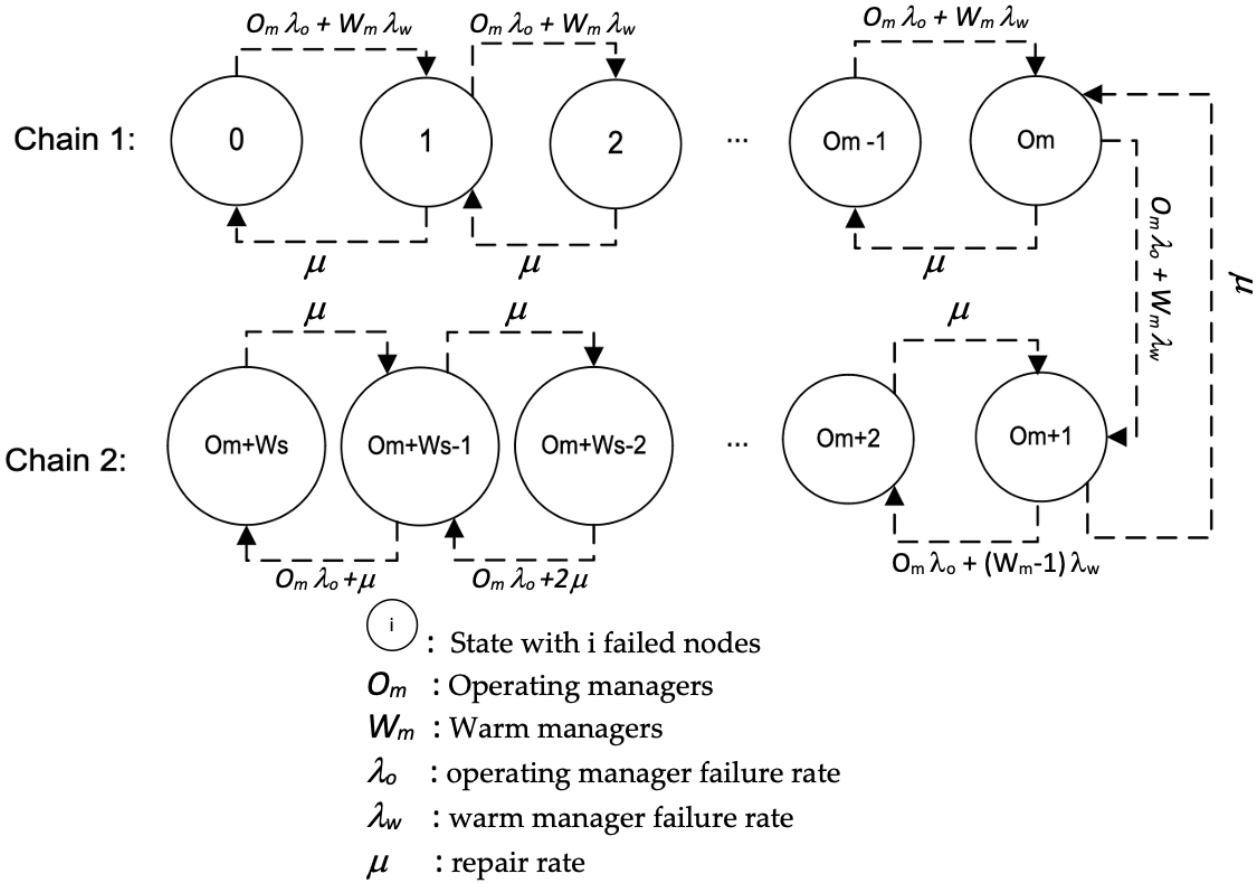

Fig. 8. Markov chain of the proposed model.

Let T represent the total number of fog managers, and the states i, $i \in [0, T]$, represents the number of failed devices in a state. In the initial state P(0), there is a state with zero failed devices, and P(i) represents the probability of i failed devices. The states probability can be derived from the following equation:

$$\text{Chain (1): } 0 \le i \le O_m$$

$$i = 0 \text{ (initial state): } O_m \lambda_o + W_m \lambda_w P(0) = \mu P(1)$$

$$i = 1: (O_m \lambda_o + W_m \lambda_w + \mu) P(1) = (O_m \lambda_o + W_m \lambda_w) P(0) + \mu P(2)$$

$$i = 2: (O_m \lambda_o + W_m \lambda_w + \mu) P(2) = (O_m \lambda_o + W_m \lambda_w) P(1) + \mu P(3)$$

$$i = O_m - 1 : (O_m \lambda_o + W_m \lambda_w + \mu) P(O_m - 1) = (O_m \lambda_o + W_m \lambda_w) P(O_m - 2) + \mu P(O_m)$$

$$i = O_m : (O_m \lambda_o + W_m \lambda_w + \mu) P(O_m) = (O_m \lambda_o + W_m \lambda_w) P(O_m - 1) + \mu P(O_m + 1)$$

A general expression for P(i) in chain 1 can be calculated by:

$$(O_m \lambda_o + W_m \lambda_w + \mu) P(i) = (O_m \lambda_o + W_m \lambda_w) P(i-1) + \mu P(i+1) \quad (1)$$

$$\text{where } i \in [1, O_m]$$

$$\text{Chain (2): } (O_m + 1) \le i \le (O_m + W_m)$$

$$i = O_m + 1 : [O_m \lambda_o + (W_m - 1) \lambda_w + \mu] P(O_m + 1) = [O_m \lambda_o + (W_m - 1) \lambda_w] P(O_m) + \mu P(O_m + 2)$$

$$i = O_m + 2 : [O_m \lambda_o + (W_m - 2) \lambda_w + \mu] P(O_m + 2) = [O_m \lambda_o + (W_m - 2) \lambda_w] P(O_m + 1) + \mu P(O_m + 3)$$

$$i = (O_m + W_m - 1): [O_m \lambda_o + \lambda_w + \mu] P(O_m + W_m - 1) =$$

$$[O_m \lambda_o + 2\lambda_w] P(O_m + W_m - 2) + \mu P(O_m + W_m)$$

$$i = O_m + W_m : [O_m \lambda_o + \mu] P(O_m + W_m) = [O_m \lambda_o + \mu] P(O_m + W_m)$$

A general expression for P(i) in chain 1 can be calculated by:

$$i = O_m + W_m : [O_m \lambda_o + \mu] P(O_m + W_m) = [O_m \lambda_o + \mu] P(O_m W_m - 1) + \mu P(O_m + W_m + 1) \quad (2)$$

$$[O_m \lambda_o + (O_m + W_m - i) \lambda_w + \mu] P(i) = [O_m \lambda_o + ((O_m - W_m - i) + 1) \lambda_w P(i-1) + \mu P(i+1)$$

$$\text{where } i \in [(O_m + 1), (O_m + W_m)] \quad (3)$$

Given the fact that $O_m + W_m$ is equal to T and based on equation (2), the final state of the system is given by:

$$i = T \text{ (final state)} : [O_m \lambda_o + \mu] P(T) = [O_m \lambda_o + \mu] P(T - 1) + \mu P(T+1) \quad (4)$$

Consequently, the utilization of operating managers and warm managers can be denoted as ρ_o and ρ_w respectively. The utilization parameter can be calculated by dividing the failure rate by the repair rate. By solving Eqs. (1) to (4) P(i) can be calculated as follows:

$$\text{Chain 1} : P(i) \left(Om + Wm \right) i \; P(0) \, , 0 \leq i \leq Om) \quad (5)$$

$$\text{Chain 2} : P(i) \left(Om + Wm \right) om \; x \prod_{j=om}^{i-1} [\, 0m \rho o + (T - j) \rho w \,] P(0) \; , Om \leq i \leq T \quad (6)$$

V. Availability Analysis for the Proposed Model

Since the proposed model is designed to enhance availability in fog computing, it is crucial to construct related metrics such as expected number of failed operating and warm fog managers, which is the focus of this paper. Generally, systems availability can be defined as the system probability of being in a functioning state at any time. Fog computing availability includes two parts: the availability of cloud data centers and the availability of fog nodes. Cloud computing availability has been receiving a decent amount of attention from scholars compared to fog computing. The following characteristics has been defined for the proposed model:

L = represents the expected number of failed managers

E[O] = the expected number of failed operating managers O_m

E[W] = the expected number of warm managers W_m

A = the availability of standby management layer

From Eq. (5) to (6), the following expressions can be driven:

$$L = \sum_{i=0}^{T} i \times P(i) \quad (7)$$

$$E[O] = \sum_{i=1}^{Om} (T - i) \times P(i) \quad (8)$$

$$E[W] = \sum_{i=(Om+1)}^{T} (T - i) \times P(i) \quad (9)$$

$$A = \frac{T - L}{T} = 1 - \frac{L}{T} \quad (10)$$

In order to illustrate the theoretical implication of the proposed model, an example with comprehensive calculation is provided. In this example, there are two operating managers with failure rate λ_o of 0.1, three warm managers with failure rate λ_w of 0.024, and one repair unit with repair rate μ of 0.8. Accordingly, ρ_o and ρ_w are equal to 0.125 and 0.03 respectively.

Based on Eq. (5) and the given values of:

$$O_m = 2, \rho_o = 0.125, W_m = 3, \rho_w = 0.03, \text{ and } \mu = 0.8$$

chain 1 yields to:

$$\text{Chain 1: } P(i) = (O_m \rho_o + W_m \rho_w)^i P(0), 0 \leq i \leq O_m \quad (5)$$

$$i = 1, P(1) = (O_m \rho_o + W_m \rho_w) P(0) = (0.25 + 0.09) P(0) = 0.34 P(0)$$

$$i = 2, P(2) = (O_m \rho_o + W_m \rho_w)^2 P(0) = (0.25 + 0.09)^2 = (0.34)^2 P(0) = 0.1P(0)$$

And chain 2 based on Eqs. 6 leads to

$$\text{Chain 2 : } P(i) \left(O_m + W_m \right)^{om} \text{ x } \prod_{j=om}^{i-1} [\, 0m \rho o + (T - j) \rho w \,] P(0) \; , O_m \leq i \leq T \quad (6)$$

$$i = 3, P(3) = (0.34)^2 \text{ x } \prod_{2}^{2} [\, 0.25 + (5 - j)\, 0.03 \,] P(0)$$

$$\rightarrow (0.34)^2 \text{ x } [0.25 + (5\text{-}2)\, 0.03]$$

$$P(3) = 0.1 \times 0.34 \, P(0) = 0.034 \, P(0)$$

$$i = 4, P(4) = (0.34)^2 \times \prod_{2}^{3} [0.25 + (5 - j)\, 0.03] P(0)$$

$$\rightarrow (0.34)^2 \text{ x } \{ [0.25 + (5\text{-}2)\, 0.03] \text{ x } [0.25 + (5\text{-}3)\, 0.03] \}$$

$$P(4) = 0.1 \times 0.1 \, P(0) = 0.01P(0)$$

$$i = 5, P(5) = (0.7)^2 \times \prod_{2}^{4} [0.25 + (5 - j)\, 0.03] P(0)$$

$$\rightarrow (0.34)^2 \text{ x } \{ [0.25 + (5\text{-}2)\, 0.03] \text{ x } [0.25 + (5\text{-}3)\, 0.03] \text{ x } [0.25 + (5\text{-}4)\, 0.03 \}$$

$$P(5) = 0.1 \times 0.03 \, P(0) = 0.003 \, P(0)$$

The five state probabilities expressed in terms of P(0) are as follows:

$$[P(1), P(2), P(3), P(4), P(5)] P(0) = [0.34, 0.1, 0.034, 0.01 \; 0.003] P(0) \rightarrow 0.49 \, P(0)$$

To calculate the value of P(0), the following normalization condition is used,

$$\sum_{i=0}^{T} P(i) = 1$$

$$\sum_{i=0}^{5} P(0) + P(1) + P(2) + P(3) + P(4) + P(5) = 1$$

$$\Rightarrow 0 \; .51 + P(0) = 1 \Rightarrow \; \therefore P(0) = 0.51$$

All the values above were calculated with respect to P(0). After calculating P(0), P(i) values are as follows:

P(1) = 0.17, P(2) = 0.05, P(3) = 0.017, P(4) = 0.005, and P(5) = 0.0015

Accordingly, solving Eq. (7) to (10) to calculate the proposed model metrics as follows:

$$L = \sum_{i=0}^{5} i \times P(i) = [P(1) + 2\,(P2) + 3\,(P3) + 4\,P(4) + 5\,P(5)] \Rightarrow$$

$$= (0.34 + 0.2 + 0.1 + + 0.04 + 0.015) = 0.7 \times 0.51 = 0.357$$

$$E[O] = \sum_{i=1}^{2} (5 - i) \times P(i) = 4\,P(1) + 3\,P(2) \Rightarrow (1.36 + 0.3) \times 0.51 = 0.85$$

$$E[W] = \sum_{i=3}^{5} (5 - i) \times P(i) = 2\,P(3) + P(4) \Rightarrow (0.7 + 0.01) \times 0.51 = 0.36$$

$$A = 1 - \frac{L}{T} = 1 - \frac{0.36}{5} \Rightarrow 93\%$$

Fig. 9 illustrates the probabilities of a fog management system of two operating managers and three warm managers with failure rates of 0.1 and 0.024 respectively and a repair rate of 0.8. The vertical axis represents the steady state probability, and the horizontal axis denotes the number of failed managers. The curve starts with the probability of losing one operating manager with a probability of 0.17. As the number of failed managers increases, the failure probability decreases until it reaches 0.0015. With the mentioned failure and repair rates, the model was able to reach 93% availability and the relevant measures such as E[O] and E[W] are 0.7 and 1.5 independently.

Additionally, Fig. 10 depicts the improvement in availability percentage of the proposed model using redundancy. The figure compares availability percentages of two models. First, the redundant model consists of two operating mangers, three warm mangers, and a repair unit. Second, the non-redundant model consists of only two operating managers. Failure rates are identical in both models. As the figure shows, the redundant model availability is 93% while the non-redundant model availability is 87%. Accordingly, the redundant model increases availability by 6% compared to a non-redundant model.

Further experiments were conducted to increase the proposed model availability over 98%. These experiments included changing the failure and repair rates. However, increasing the repair rate and decreasing failure rate can be very expensive. A different experiment was conducted which focused on the ratio of operating managers to warm managers.

As Fig. 11 represents, availability percentage increases as the number of warm managers increase. The figure depicts different configurations with correlations of operating managers O_m to warm managers W_m as follows:

A: $W_m = O_m$

(The number of W_m equals to the number of O_m e.g., O_m = 2 and W_m = 2).

B: $W_m = O_m^2$

(The number of W_m equals to the number of O_m to the power of 2 e.g., O_m = 2 and W_m = 4).

C: $W_m = O_m^3$

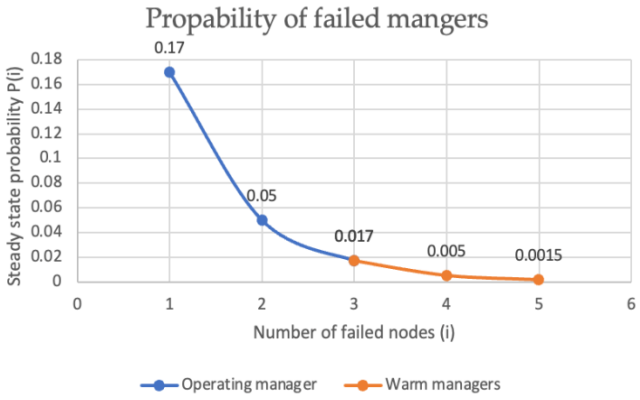

Fig. 9. Probabilities of failed managers in fog management system.

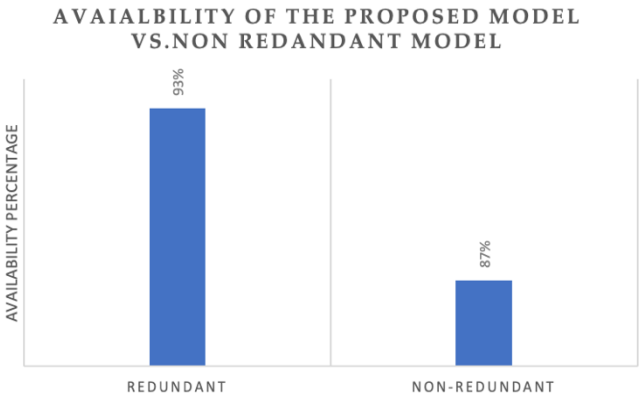

Fig. 10. Availability of the proposed model compared to a non-redundant model.

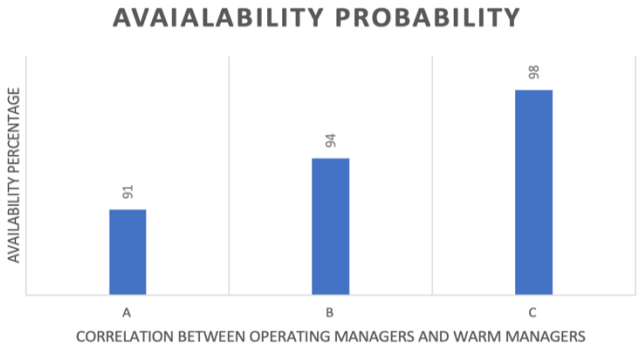

Fig. 11. Availability probability for different numbers of operating/warm nodes.

(The number of W_m equals to the number of O_m to the power of 3 e.g., O_m = 2 and W_m = 8).

Failure and repair rates are fixed for all models. As the figure shows, an availability rate of 98% can be reached when using configuration C, which is a high rate and defiantly it comes with expenses. In the case of configuration C, the extra

cost is in the form of extra hardware which is more practical than unreasonable repair and failure rates.

VI. Comparison of the Proposed Model with other Models

The proposed model in this paper is a standby system consisting of operating nodes and warm nodes. The system is designed to reside in a separate layer between fog nodes and cloud data centers, which is called the management layer. This layer is discussed in detail in [7]. The mechanism used in this model is hardware redundancy. As mentioned in Section II, the main fault tolerance technique that have been incorporated into fog computing is data/application replication as in [18] and [19]. Availability probability was not calculated in these studies. The research in [18], did not include any evaluation metrics for the proposed software, On the other hand, authors in [19] presented a proactive scheme for service replication in IoT computing. The evaluation process of the scheme included service drop rate, response time, and service completion time. Even though this study covered aspect of availability, it was designed for qusai adhoc scenarios in general which may include fog computing. Also, it did not include hardware redundancy to be suitable for comparison with the presented model in this paper. Further research can be conducted to explore other models or techniques to improve availability in fog computing.

VII. Conclusion and Future Works

To enhance performance and availability in fog computing, researchers have been focusing on several aspects of fog computing such as resource management, load balancing algorithms, placement policies, and service replication. However, deployment in fault tolerance techniques has not received much attention. In this paper, a fog model is proposed with a standby management layer. The model is designed to tolerate losing fog managers at the management layer. Qualitative analysis of the proposed model is presented using a Markov chain. For further studies, we aim to study the limitations of the proposed model and enable periodic switching for the duplex system to avoid exhausting fog nodes. Furthermore, a cost effectiveness study along with extensive comparison of the proposed model with other models designed to improve availability in fog computing can be conducted.

References

[1] R. M. J. Z. S. A. Flavio Bonomi, "Fog computing and its role in the internet of things," in The first edition of the MCC workshop on Mobile cloud computing, 2012. (pp. 13-16). New York, NY, United States: Association for Computing Machinery., 2012.

[2] O. C. A. Working, "OpenFog Reference Architecture for Fog Computing," February 2017. [Online]. Available: https://site.ieee.org/denver-com/files/2017/06/OpenFog_Reference_ Architecture_2_09_17-FINAL-1.pdf. [Accessed June 2022].

[3] T. Welsh and E. Benkhelifa, "On Resilience in Cloud Computing: A Survey of Techniques across the Cloud Domain," ACM Computing Surveys, vol. 35, no. 3, pp. 1-36, 2021.

[4] O. H. M. Heberth F. Martine, H. A. Rubio and J. Marquez, "Computational and Communication Infrastructure Challenges for Resilient Cloud Services," Computers , vol. 11, no. 8, 2022.

[5] W. Wang, H. Chen and X. Chen, "An Availability-Aware Approach to Resource Placement of Dynamic Scaling in Clouds," in IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, USA, 2012.

[6] I. Koren and C. Krishna, Fault-Tolerant Systems, 2007.

[7] B. S. M. A. A. L. Sara Alraddady, "Fog Computing: Strategies for Optimal Performance and Cost Effectiveness," Electronics, vol. 11, no. 21, 2022.

[8] P. Maiti, H. K. Apat, B. Sahoo and A. K. Turuk, "An effective approach of latency-aware fog smart gateways deployment for IoT services," Internet of Things, vol. 8, 2019.

[9] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana and M. Parashar, "Mobility-Aware Application Scheduling in Fog Computing," IEEE Cloud Computing , vol. 4, no. 2, pp. 26 - 35, 2017.

[10] I. Lera, C. Guerrero and C. Juiz, "Availability-Aware Service Placement Policy in Fog Computing Based on Graph Partitions," IEEE Internet of Things Journal , vol. 6, no. 2, pp. 3641 - 3651, 2018.

[11] M. Aazam and E.-N. Huh, "Fog Computing Micro Datacenter Based Dynamic Resource Estimation and Pricing Model for IoT," in 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, Gwangju, Korea (South), 2015.

[12] T. Zhang, J. Jin, X. Zheng and Y. Yang, "Rate adaptive fog service platform for heterogeneous iot applications," IEEE Internet of Things Journal , vol. 7, no. 1, pp. 176 - 188, 2020.

[13] V. Karagiannis and S. Schulte, "Comparison of Alternative Architectures in Fog Computing," in 2020 IEEE 4th International Conference on Fog and Edge Computing (ICFEC), Melbourne, VIC, Australia, 2020.

[14] M. Aldossary and H. A. Alharbi, " Towards a Green Approach for Minimizing Carbon Emissions in Fog-Cloud Architecture," IEEE Access, no. 9, pp. 131720 - 131732, 2021.

[15] L. Benchikh and L. Louail, "Task scheduling approaches for fog computing," in 2021 30th Wireless and Optical Communications Conference (WOCC), Taipei, Taiwan, 2021.

[16] A. J. Kadhim and J. I. Naser, "Proactive load balancing mechanism for fog computing supported by parked vehicles in IoV-SDN," China Communications, vol. 18, no. 2, pp. 271 - 289, 2021.

[17] J. Grover and R. M. Garimella, "Reliable and Fault-Tolerant IoT-Edge Architecture," in IEEE SENSORS, New Delhi, India, 2018.

[18] A. U. E. F. Asad Javed Department of Computer Science, K. Heljanko, A. Buda and K. Främling, "CEFIoT: A fault-tolerant IoT architecture for edge and cloud," in IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, 2018.

[19] B. Choudhury, S. Choudhury and A. Dutta, "A Proactive Context-Aware Service Replication Scheme for Adhoc IoT Scenarios," IEEE Transactions on Network and Service Management , vol. 16, no. 4, pp. 1797 - 1811, 2019.

[20] T. Zhang, M. Xie and M. Horigome, "Availability and reliability ofk-out-of-(MCN):G warm standby systems," Reliability Engineering and System Safety, vol. 91, no. 4, p. 381–387, 2006.

[21] M. Naedele and J. W. Janneck, "Design patterns in Petri net system modeling," in Proceedings. Fourth IEEE International Conference on Engineering of Complex Computer Systems, Monterey, CA, USA, 1998.

[22] N. S. Noori and T. I. Waag, "Application of Hierarchical Colored Petri Nets for Real-Time Condition Monitoring of Internal Blowout Prevention (IBOP) in Top Drive Assembly System," in IEEE International Systems Conference (SysCon), Orlando, FL, USA, 2019.