INTERNATIONAL JOURNAL OF NATURAL AND APPLIED SCIENCES (IJNAS), VOL. 12, Special Edition (2019); P. 141 – 144, 0 TABLES, 1 FIGS.

A model of an improved fault-tolerant system for wireless network

D.O. Egete [1], B.I. Ele[*1], E.E. Umoh[2] and D.U. Ashishie[1]

ABSTRACT

Fault-tolerant system has become intricately linked with everyday life, being discovered in our society and it is used in industries, health, and the banking sector for service delivery. Traditional techniques of fault tolerance exploit redundancy to mask the faults that will improve the system availability but the cost of deploying these systems has relegated it to be used only in critical cases. Therefore; it is imperative to develop a model of fault tolerant system for a wireless network that can ensure availability and to reduce cost. The proposed model will use a combination of triple modular redundancy scheme and the fuzzy logic to solve the problem of excessive cost and to ensure that the system dependability in evaluating fault tolerance in systems can effectively reduce cost. In this study, an improved model of a fault-tolerant system for wireless network was developed. Hence; the combination methods use above should be applied in checking or removing the fault in the systems.

INTRODUCTION

Systems used in critical applications such as health, commerce, transportation, utilities, and national security must be highly reliable. Global use of computing systems and other electronic systems in these critical areas calls for computing systems with high reliability. High reliability is achieved by designing the systems to be fault-tolerant (Koren & Krishna, 2017).

Therefore; one of the goals for adaptive design is to allow flexible use of available resources to cover a much wider range of different kinds of environmental variables than could be covered by a fixed, worst-case design, the demand for availability in computing are those that cannot be overemphasized and the advent of cloud computing and internet of things have brought us closer to devices which places more focus on fault tolerant systems. Fault tolerance is a system's ability to perform designated functions in the presence of faults, and a fault-tolerant system should be able to handle faults in individual hardware and software components, power failures, other kinds of unexpected problems and still meet its specification (Dubrova, 2013). Fault tolerance is important because it's practically impossible to build a perfect system and the reliability of a system decreases as its complexity increases unless some reparation is considered and all fault tolerance is an exercise in manipulating and managing *redundancy*. Redundancy is the property of having more of a resource than is minimally necessary to do the job at hand.

As failures happen, redundancy is manipulated to mask or otherwise work around these failures, thus maintaining the desired level of functionality (Koren & Krishna, 2017).

Redundancy and System Performance

Many design-based techniques have been outlined for both detection and mitigation of faults in systems. The detection techniques include; Software redundancy, Hardware redundancy, Time redundancy, and Information redundancy. Mitigation techniques consist of multiple redundancies with voting (Khorasani, *et al*, 2019), triple modular redundancy (TMR), and Error Detection and Correction coding (EDAC).

Hardware redundancy

This provides the extra hardware design that will detect or override the effects of a failed component by incorporating the Static Hardware Redundancy that will initiate the immediate masking of a breakdown with the active component with a higher operating cost. Therefore; it is mostly advisable that this scheme is always applied to critical systems where the overhead cost can offset the cost of failure (Dubrova, 2013).

Information redundancy

Is provided through error detection and correction coding. Were the extra check-up bits are added to the original data bits, to check an error in the data bits, which will detect or correct. The codes are extensively used today in memory units and various storage devices to protect against non-malignant failures (Dubrova, 2013).

*[1]Corresponding author. Email: mydays2020@gmail.com
[1]Department of Computer Science, University of Calabar, Calabar, Nigeria
[2]Department of Computer Science, Cross River University of Technology, Calabar Nigeria

Time redundancy

Is against *transient* faults computing nodes that exploit time redundancy through re-execution of the same program on the same hardware and if the code used for data communication is not capable to detect the faults that have occurred without correcting them, then it can be retransmitted as required by employing the time redundancy. A fault-tolerance ultimate aim is to develop dependable systems that have formed an integral part of our everyday lifecycle which has become important for society since computing has become more demanded dependability (Dubrova, 2013).

Origins of Faults

Faults can be caused by external factors such as human actions which could be accidental or deliberate or environmental disturbances. Problems of not meeting its specification requirements, implementation, or fabrication stages of a system design also result in faults. Faults lead to errors, and errors lead to failure, these are categorized into; Incorrect specifications from incorrect algorithms, architectures, or requirements. Therefore, incorrect implementation of the specification requirements leads to defects imperfections from the manufacturers, random device and components wear-outs, and external faults arise from an outside system that will limit the user/operator environment (Alwan, & Agarwal, 2009).

Fault Types

There are three types of faults: permanent, intermittent, and transient. A permanent fault does not die away with time but remains until the affected unit is repaired or replaced. This is an intermittent fault cycle between the fault–active and faults benign states. A transient fault dies away after some time (Persya & Nair, 2008).

RELATED WORKS

Triple Modular Redundancy (TMR)

Multiple copies are executed and error checking is achieved by comparing results after completion. In this scheme, the overhead is always on the order of the number of copies running simultaneously (Behzadi & Azad, 2014)

Primary/Backup (*PB)*: The tasks are known to be periodic and two instances of each task are primary and a backup, are scheduled on a uni-processor system. One of the restrictions of this approach is that the period of any task should be a multiple of the period of its preceding tasks. It also assumes that the execution time of the backup is shorter than that of the primary.

Primary/Exception (PE): This performs the same function as PB Method except that it uses the exception handlers to execute its task instead of backup programs.

Triple Modular Redundancy Fault Tolerance* (*TMR): The weakest component/subsystem dictates the *reliability* of a series system while the Triple Modular Redundancy (TMR) is the most popular static redundant architecture for large complex systems with the assurance of no single point of failure but it is week. The features of TMR are as follows: (i). It is designed for applications that demand the highest reliability (ii) It cannot eliminate all single points of failure by providing redundant control paths for all communications (Persya & Nair, 2008).

Fuzzy Control Systems: Fuzzy control systems produce actions according to fuzzy rules based on fuzzy logic. The basic units of the fuzzy logic controller are fuzzifier, fuzzy rule base, fuzzy inference engine, and defuzzifier (Gajjar, Sarkar, & Dasgupta, 2014). In the fuzzifier, crisp input values are mapped to fuzzy sets using membership functions. Fuzzy rule base contains the IF-THEN rules which specify the behavior of the system. Fuzzy inference engine maps input fuzzy sets to output fuzzy sets using a rule base. Defuzzifier maps the fuzzy output sets to the crisp output value. Rule-based fuzzy inference step along with the centroid norm for defuzzification are used in this voter. Statistically, selecting the fuzzy parameter values, in this voter, the variation of the fuzzy parameter values is varied by the performance of the voter (Devi & Ramaiah, 2015).

Voting is an important operation in realizing ultra-reliable systems based on a multi-channel computation paradigm and it needed even if multiple computation channels for redundant hardware units, diverse program modules executed on same basic hardware, identical hardware, and software with diverse data or other possible hardware/program/data redundancy and diversity combinations. The TMR method is suitable for realizing this but there is a need for automatic dynamic selection of values for parameters for any dynamically varying input dataset. The fuzzy voter computes the voter output as a weighted average and weights are determined by the fuzzy inference engine. The fuzzy voter computes the voter output as a weighted average and weights are determined by the fuzzy inference engine (Khan, Daachi, & Djouani, 2012). A Neural-Fuzzy System (NFS) is designed to

realize the process of fuzzy reasoning where the connection weights of the network relate to the parameters of fuzzy reasoning. Using the backpropagation type learning algorithms, the NFS identifies fuzzy rules and learns membership functions of the fuzzy reasoning and it is easier to start a one-to-one correspondence between the network and the fuzzy system. The outputs from the redundant blocks are passed through the NFS where the level of their availability is determined. The defuzzified result is then passed to the voter for the voting process. Based on our analysis above the combination of hybrid triple modular redundancy scheme and fuzzy logic used in solving the problem of excessive cost, to ensure the system dependability has enhanced the fault tolerance and also reduced the cost in the execution of tasks as require (Imad, Shawi, & Lianshan, 2012).

Also, the model has independent control systems and voters that are reliable to increase between one to two orders of magnitude. Reliability, R(t), is a distribution function, R(t) = 1 –P{t≤t}, a function that describes the probability of a system operating at time t.

METHODOLOGY

An important system used in industries like health and manufacturing needs to be operational as failure can occur and it will lead to drastic consequences. Different systems commit for time to respond and function correctly even in the presence of faults. This implies that safety-critical applications with stringent time and cost constraints should be implemented to tolerate faults and satisfy constraints. The Earliest deadline first (EDF) for real-time scheduling method was employed for soft real-time systems since in soft real-time systems it is more important to economically detect a fault as soon as possible rather than to mask a fault. From the design, the redundant blocks which served as a backup are automatically removed from memory when the primary blocks finish executing.

Therefore; the novel algorithm for increasing faults tolerance in wireless network using triple modular redundancy scheme and fuzzy logic with neighbor sensors to cover the area affected by the faulty sensor to eliminate the redundant sensors, which is more economical and the algorithm knows that all network sensors are the same and are randomly dispersed in a two-dimensional rectangular environment. The network is a cluster of sensors with each sensor being a member of the cluster. The election process here is to allow the neighbor sensor with the highest priority to cover the missed area by moving towards that area.

In this study, Reliability is linked to Mean Time Between Failure (MTBF) as follows:

MTBF is defined as a function of the reliability distribution function

$$MTBF = \int_0^\infty t \frac{d}{dt}(1 - R(t)) \cdot dt. \quad (1)$$

From above, when h(t) = λ is a constant, $R(t) = e^{-\lambda t}$, and MTBF is then $1/\lambda$. This simple relationship between λ and MTBF makes MTBF an easy to calculate and popular metric of anticipated system performance. Figure 1 is the triple modular redundancy architecture.

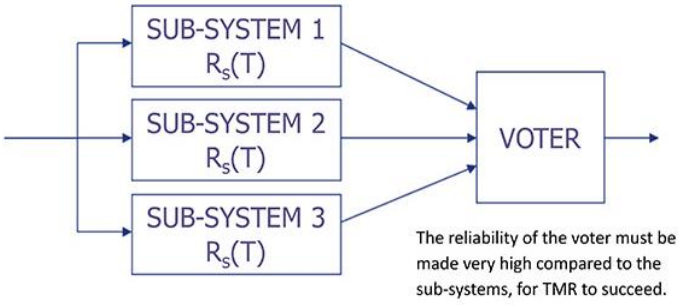

Fig. 1. Triple Modular Redundancy Architecture

ALGORITHM FOR FAULT TOLERANCE MODEL

Each fault-tolerant format will have several unique candidates based on their configurations and characteristics such as response time, resources like computing power, and storage. Used for, replication fault tolerance model for storage and compute power-intensive while the process migration model is time-intensive. A fault-tolerance model, F_i, can have I candidates F_{ij} for k=1 to n.

Algorithm for the proposed fault tolerance model

Input: Array of FT candidate values, value of a configuration

Output: Best FT candidate for the given configuration

1. Function of best FT candidate(t[i], $C_{tk,}$ while t[i]- FT value is an interaction value
2. x = 0
3. For I = 1 to n do
4. If t[i]<= C_{tk} then
5. S[x] = t[i]
6. x = x+ 1
7. end if

```
8.    end for
9.   min=S[1]
10.  for k = 2 to x do
11.      if min >S[k]
12.          min = S[k]
13.          FT(Ct) = k
14.      end if
15.    end for
16.  end function
```

RESULT AND DISCUSSION

In this study, an improved model of a fault tolerant system for the wireless network was developed that is capable of detecting and fixing faults in wireless network efficiently in less time and at reduced cost. Based on the findings of this study, the combination of the Triple Modular Redundancy and Fuzzy logic Control method has proven to be very efficient and significantly reduces the cost of executing the tasks and maintenance of redundant system blocks of the faulty components easily identified the level of its dependability to determine both the election of TMR voter. The NFS has incorporated into the fault-tolerant modules of its working lifecycle of the various modules which are used to determine the faulty modules and redeem it before it scratches the entire system.

CONCLUSION

The combination of the Triple Modular Redundancy and fuzzy logic approaches was used to develop an improved model of a fault tolerant system for the wireless network and this constitutes a key component of soft computing. The Neural Fuzzy System (NFS) which is a subsystem in the proposed model helps in troubleshooting the entire system and enables the operator to know how dependable the system is. From the design, the lifecycle of modules is used to monitor the dependability of the system to ascertain the maintenance scheduled with ease. Thus, the hybrid approach is recommended for use in wireless network fault tracing and detection.

REFERENCES

Alwan, H., and Agarwal, A. (2009). A survey on fault tolerant routing techniques in wireless sensor networks. *International Conference on Sensor Technologies and Applications*. 366-371.

Behzadi, S., and Azad, M. (2014). Fault-tolerant in wireless sensor networks using fuzzy logic. International Research Journal of Applied and Basic Sciences, 8 (9): 1276-1282.

Devi, D., and Ramaiah, P. (2015). A Neuro-Fuzzy Approach Based Voting Scheme for Fault-Tolerant Systems Using Artificial Bee Colony Training. World Academy of Science, Engineering and Technology, International Journal of Computer and Information Engineering, 9(3).

Dubrova, E. (2013). Fault-tolerant design. Springer Science+Business Media, New York.

Gajjar, S., Sarkar, M., and Dasgupta, k. (2014). Cluster Head Selection Protocol using Fuzzy Logic for Wireless Sensor Networks. *Computer Application*, 97:38-43.

Imad, S., Shawi, A. and Lianshan, Y. (2012). Lifetime Enhancement in Wireless Sensor Networks Using Fuzzy logic system Approach and A-Star Algorithm. *IEEE Sensors Journal*, 12:3010-3018.

Jhawar, R., Piuri, V., and Santambrogio, M. (2012). Fault tolerance management in cloud computing: a system-level perspective.

Khan, S. A., Daachi, B., and Djouani, K. (2012). Application of fuzzy inference systems to the detection of faults in wireless sensor networks. *Neurocomputing*, 94:111-120.

Khorasani, A., Vahdat, B., and Mortazavi, C. (2019). Analysis of 32-bit Fault-Tolerant ALU Methods.

Koren, I., and Krishna C. (2007). Fault-Tolerant Systems. Morgan Kaufmann Publishers.

Persya, A., and Nair, T. (2008). Fault-tolerant real-time systems. International Conference on Managing Next Generation Software Application (MNGSA-08), Coimbatore, 2008