A novel probabilistic bit voter using genetic algorithm for fault-tolerant systems

Manizheh Mirsaeidi [1], Abbas Karimi [2]

**[1] Department of Computer Engineering, Faculty of Engineering, Arak Branch, Islamic Azad University
Arak 38181-46775, Iran**

**[2] Department of Computer Engineering, Faculty of Engineering, Arak Branch, Islamic Azad University
Arak 38181-46775, Iran**

Abstract

Fault-tolerant systems are systems that can continue their operations even in the case of an error. These systems are used in the real-time systems, embedded business systems, e-commerce, transport systems (including rail, air and car), nuclear energy, electronic Ad-hoc system, space and military networks and industrial environments. Voting algorithms[13,14,15] are used in a wide area of control systems from real-time and safety-critical control systems to pattern recognition, image processing and human organization systems in order to arbitrating among redundant results of processing in redundant hardware modules or software versions .Bit voting algorithm is a voting algorithm used for fault-tolerant control systems. In this paper, a new bit voting based on a probabilistic algorithm is proposed. Some of voters do not provide good availability play an important role in proper implementation of the systems. Experimental results show that the proposed voter improve availability considerably.

***Keywords**: Fault-tolerant systems, bit voting algorithms, probabilistic voting algorithm, genetic algorithms.*

1. Introduction

Redundancy is used as one of the basic techniques in the implementation of fault-tolerant algorithms result of voting algorithms for choosing the most appropriate voting level among redundancy modules and likely false results. Redundancy can be implemented in three forms: static, dynamic and hybrid. Also, the redundancy in the system can be done in one of following four ways: 1 redundancy in hardware 2. Redundancy in software 3- redundancy in data 4- redundancy in time [6]. NMR and NVP are two main methods of static redundancy used in hardware and software . TMR is the simplest form of NMR, where N is equal to 3 [5.8]. Redundancy systems such as NMR require a voting to judge about inputs. Several strategies is used for voting fault-tolerant systems such as majority algorithm, relative majority algorithm, the weighted average algorithm, Midian algorithms, predictive algorithms, etc.

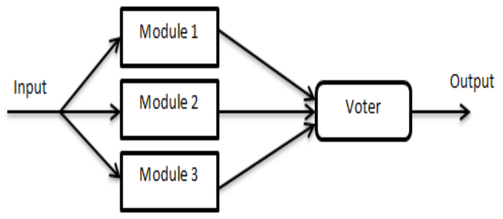

Fig. 1 TMR SYSTEM

In this paper is presented a new probabilistic voting bit algorithm with high availability, low faults and high safety and make simulate and evaluate by genetic algorithm [7, 9,10].
In Section 2, we refer to an earlier work, Section 3 describes the basic concepts of genetic algorithm and the proposed algorithm is introduced in Section 4. In Section 5, the new proposed system will be evaluated and this research conclusion will be present in the study.

2. RELATED WORKS

Major headings are to be column centered in a bold font without underline. They need be numbered. "2. Headings and Footnotes" at the top of this paragraph is a major heading.

2.1 Smoothing voter

Smoothing voter [12] algorithm is a developed algorithm of majority voter that an acceptance test has been added to it. Acceptance test based on the assumption that the distance between consecutive versions results is indicate of an error. This assumption is valid in many real time control applications where there is feedback and consecutive calculations.

In smoothing voter, when there is disagreement between variables, the closest result to the previous voter output is considered as a possible result for voting cycle.

If the distance measured is less than the predicates known as smoothing threshold, the result is the same output voter. Otherwise, the voter cannot respond.it supposed he is successful in the first algorithm cycle. The answer to the next cycle and its output can be used the next on the basis of the correct answer. The selection method of Smoothing threshold is very important.

However, although choosing value is arbitrary, but will be improved if the selected value would be on the basis of information about the potential difference between consecutive results during the mission system.

2.2 New weighted algorithm using neural network

As we know, from one other voting algorithms are based on two groups: 1- on the basis of agreement like majority voter and plurality provided according to output agreement and 2- without considering agreement in which redundancy variables results provided without considering agreement like average voter and weighted average used for high availability systems among which weighted average is more reliable. In addition, average voter simply select results average. Weighted average gives weight to input according to predetermined priority or their distance to inputs. So, it is possible that participation of more reliable inputs will be increased to voter than fault inputs. In this section we introduce new weighted algorithm using neural network for increasing reliability.

2.3 Neural network voter

For designing weighted average voter, we must instruct neural network according to previous instruction so allocate weight to each input and provide appropriate output for each input. Weighted average algorithm flowchart is as follow on the basis of neural network. For facilitating in the instruction process, we consider the limits of neural network inputs- output. This voter algorithm is important for three reasons: 1- neural network was used for designing and implementing voter algorithm. 2- this algorithm is useful according to above diagrams and median and weighted average voter applications in the high availability systems which sometimes we need high reliability and safety cases because in comparison to median and weighted , the number of correct result is high and the number of false result is low. 3- if we consider the weight equal to 2, so we need more calculations and as a result memory space and time complexity will be increased. Although instructing neural network will be prolonged but it will be done before implanting algorithm one time and so there isn't time complexity. Therefore we need not complex mathematics calculations[11]. Fig.2 shows the flowchart of designing neural weighted average vote useful according to above diagrams and median and weighted average voter applications in the high availability systems which sometimes we need high reliability and safety cases because in comparison to median and weighted , the number of correct result is high and the number of false result is low. 3- if we consider the weight equal to 2, so we need more calculations and as a result memory space and time complexity will be increased. Although instructing neural network will be prolonged but it will be done before implanting algorithm one time and so there isn't time complexity. Therefore we need not complex mathematics calculations[11]. Fig.2 shows the flowchart of designing neural weighted average vote designing neural weighted average voter.

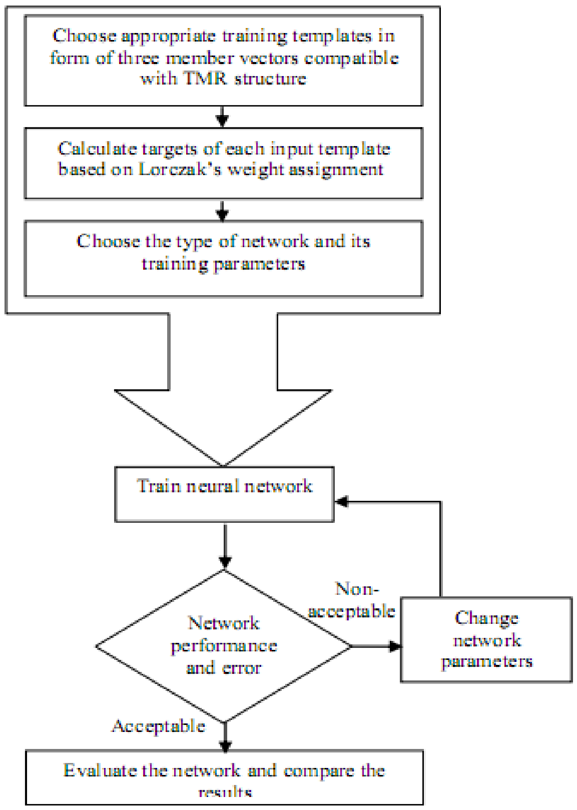
Choose appropriate training templates in form of three member vectors compatible with TMR structure
Calculate targets of each input template based on Lorczak's weight assignment
Choose the type of network and its training parameters
Train neural network
Network performance and error
Non-acceptable
Change network parameters
Acceptable
Evaluate the network and compare the results

Fig. 2 flowchart of designing neural weighted average voter

2.4 Majority voter algorithm

Output values of function f(x1,x2,x3) is shown in the following table[5] :

table 1:the truth table of majority voter

X_1	X_2	X_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

2.5 Probabilistic voting

Set parts {0, 1} are output symbols of redundancy modules. E1 is fault probability of symbol {1} in the modular output. E0 is the probability of fault of symbol {0} in the modular output. For redundancy modular with n bite input, logical function of modular with n2 condition is shown in the truth table. The number of zero in the f is equal to N0 and The number of 1 in the f is equal to is N1.

$$E_1 = \frac{N_0}{2^n} \tag{1}$$

$$E_0 = \frac{N_1}{2^n} \tag{2}$$

The truth table of Probabilistic voting for TMR is shown in the table 2:

Table 2: The truth table of Probabilistic voting for TMR

y_1	y_2	y_3	Cost for 0 C_0	Cost for1 C_1	y
0	0	0	$E_0/3$	∞	0
0	0	1	$E_0/2$	E_1	X
0	1	0	$E_0/2$	E_1	X
0	1	1	E_0	$E_1/2$	X
1	0	0	$E_0/2$	E_1	X
1	0	1	E_0	$E_1/2$	X
1	1	0	E_0	$E_1/2$	X
1	1	1	∞	$E_1/3$	1

For define logical function, 'X' values are obtained in the general truth table with the following equation:

$$X = \begin{Bmatrix} 1 & C_1 \le C_0 \\ 0 & C_0 < C_1 \end{Bmatrix} \tag{3}$$

Here C0 is cost function for symbol {0} and C1 is cost function for symbol {1}.

$$C_0 = \frac{E_0}{V_0} \; and \; C_1 = \frac{E_1}{V_1} \tag{4}$$

3. GENETIC ALGORITHM

Genetic algorithm introduced by john Holland works on principle of natural evolution. Further the efficiency of genetic algorithm has been mathematically established.
Solution to problems using genetic approach involves encoding and evaluation. The parameters of interest are converted into codes and combined together to form a chromosome. Further the solution is evaluated for a fitness function (which has to be minimized/maximized) for each solution is calculated, the solution with better fitness function has a higher probability of survival. A simple genetic algorithm that yields good results is composed of three operations:
1. Reproduction
2. Crossover
3. Mutation
Reproduction is a process in which individual solutions are selected according to their objective function values, F (fitness function). Selecting solutions according to their fitness values ensures that they have a higher probability of contributing one or offspring in the next generation. After reproduction, simple crossover may proceed in two steps. First members of the newly reproduced solutions are selected randomly.second, each pair of solutions undergoes swapping of information at certain position/s. Mutation is a genetic operator that alters one or more gene value in a chromosome (solution) from its initial state.this can result in an entirely new chromosome added in next generation.with this new gene values, the genetic algorithm would be able to arrive at better solution compared to the previous cycle.[1,16]

4. PROPOSED ALGORITHM

As indicated before, majority voter is the most important algorithm used in the NMR systems for covering fault in the digital systems. The most basic bit voting algorithm is majority algorithm that select input majority as output. Table 1 shows the accuracy table of customary bit voting for TMR.in this table, y1, y2 and y3 are modular outputs and y is voter output. The general architect of TMR is shown in the figure

1. In the previous studies, majority voter could cover one fault when there was one of three module was faulty. The customary bit majority voter selects more reliable results on the basis of output majority of redundancy modular. This voter doesn't consider the information based on symbol probability which resulted from modular performance. In this section, we introduce bit voting algorithm with the high availability which is aware of modular function. This knowledge is depending on the probability of obtained fault results in the modular output. Therefore, we call voting algorism as probabilistic voting algorithm. This probabilistic voting algorithm increase availability and decrease the complexity of voter cycle. Now, we describe the proposed algorithm as follows: We consider input function as A.

2- We calculate the number of input state and the number of 1 and number of 0.

3- We obtain C0 and C1 according to 4 formulas.

4- We receive the number of input states.

5- We calculate C0 and C1 on the basis of the formula of cost function.

6- We calculate y based on 'X'.

7- We get pe value (decimal) between 0 to 1from input.

8- We get repetition number from input.

9- We calculate decimal value on the basis of logical state of y1y2y3 inputs.

10- We obtain y function for logical state of y1y2y3 (known as f)

11- We provide three number of decimal random between 0 to 1.

12- For each bit for example y1, we compare this value with pe, if it is less than pe, its bit value will be reversed.

13- We repeat the above work for y2 y3.

14- We call this new value as HJL.

15- We calculate the decimal value of HJL.

16- We obtain y function value for this decimal value.

17- If this value will be 1 in the function, we put 1 otherwise we put 0 (known as N).

18- If F is equal to N, we put 1 and it's correct but if N is reverse of F, we put 0 and it's false (we consider the result as 0)

19- Now we sum 0 columns. (Sum 0)

20- We obtain sum ((0/2000)*100), this value is availability. As this value is not more than 50%, so we add 21 step.

21- Now we correct codes converse to each other as inaccurately. For example, if one code with one fault converts to codes which its output is one, and the output of its code is zero, we convert output to one.

Now we repeat this work to reach a highest availability value. With doing this, the availability will be to 80% it means increase to 30%.

5. Evaluation

As we refer before, the problem of cost function is that the number of 1 and 0 of logical function is important and the position of 1 and 0 is not important while 1 and 0 positions effect on the availability. The fault is depending on 1 and 0 position. By increasing function input, obtaining states convert to each other and this subject that which states cover the fault in the best way is not possible. So we use genetic algorithm. Here the function is availability. We consider chromosome as function y. our gens is y function bits. Preliminary population is 100 and it is shown with the matrix 100*8. The selection method is elitist that selects the best availability. The cross over is one point. At the first, we show simulation results o without genetic algorithm and previous algorithm.

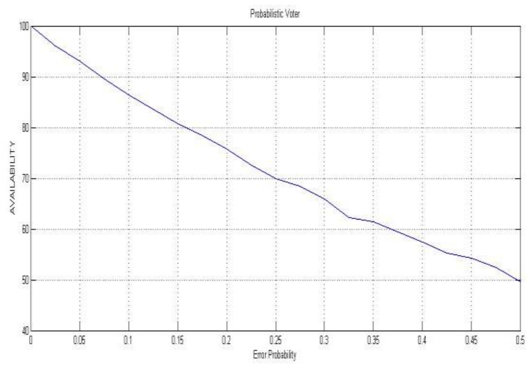

Fig.3 availability of Probabilistic algorithm

In the fig. 3 we observe that the availability is 50%. This algorithm was tested for 2000 random data and on different functions and showed that the availability will be maximum 50%, while in the proposed algorithm, the availability will increased to 80%.

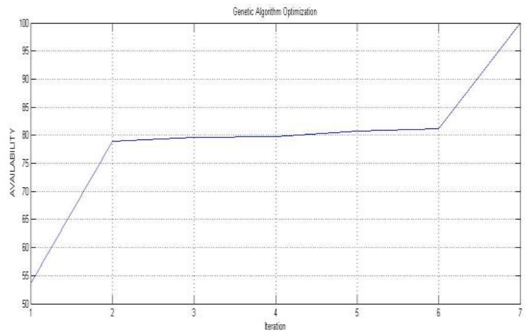

Fig. 4 the availability of proposed algorithm

In the fig. 4 following criteria are considered.

Table 3: the criteria of genetic algorithm

Value	***Topic***
The number of inputs function	16
The number of 1	7
Logical states	4
Error probability	0.5
Repeat ion	100
The number of iteration of genetic algorithm	100
The population of genetic algorithm	100
The rate of rest generation in the each algorithm repeat ion	0.5

Fig. 5 compares the availability of three voters (majority voter, probabilistic voter with proposed voter) for four functions.

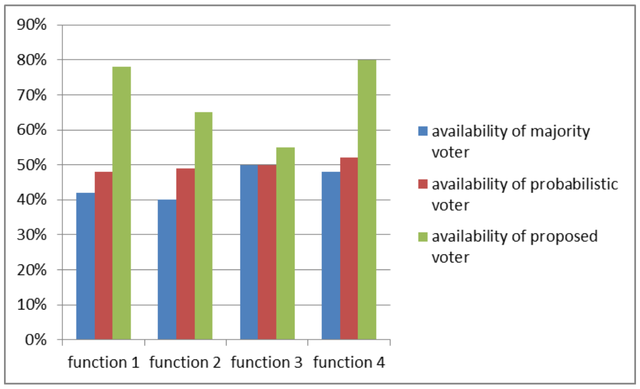

Fig.5 comparing the availaility of three voters

.

4. Conclusions

In this paper we propose a novel probabilistic bit voter using genetic algorithm for fault tolerant systems.

The criteria influenced on the voter performance are reliability, fault number, availability and safety. Another problem may be occur in the voting algorithm is their performance time influence on the system efficiency. Also, their implementing may occupy large space that has negative affect on the system.

In this study, we use probabilistic voting algorithm for improving availability. The algorithm has been simulated using MATLAB and its performance with respect

To safety and availability has been found to be better than the existing voting techniques. This algorithm notice to symbol probabilities. Another important point of this voter is that its complexity is less than majority voter circuit. This voter can be implemented for systems requiring high availability; however for time critical applications better genetic search techniques would be required to ensure time bound response of the system.

References

[1] T. Agarwal, A. Pathak, and A. Mohan, "A Novel Hybrid Voter Using Genetic Algorithm and Performance History," INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE), p. 117. [2] B. B. Alagoz, "Adaptive fault masking with incoherence scoring," arXiv preprint arXiv:0811.3816, 2008.
[3] B. B. Alagoz, "Fault Masking By Probabilistic Voting," arXiv preprint arXiv:0901.1181, 2009.
[4] B. B. Alagoz, "Hierarchical Triple-Modular Redundancy (H-TMR) Network For Digital Systems," arXiv preprint arXiv:0902.0241, 2009.
[5] S. Almukhaizim and O. Sinanoglu, "Novel hazard-free majority voter for N-modular redundancy-based fault tolerance in asynchronous circuits," Computers & Digital Techniques, IET, vol. 5, pp. 306-315, 2011.

[6] M. Amiri and V. Přenosil, "DESIGN OF A SIMPLE RELIABLE VOTER FOR MODULAR REDUNDANCY IMPLEMENTATIONS

[7] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," Dependable and Secure Computing, IEEE Transactions on, vol. 1, pp. 11-33, 2004.

[8] H. Aysan, I. Bate, P. Graydon, and S. Punnekkat, "Improving Reliability of Real-Time Systems through Value and Time Voting," 2013.

[9] T. Ban and L. A. de Barros Naviner, "A simple fault-tolerant digital voter circuit in TMR nanoarchitectures," in NEWCAS Conference (NEWCAS), 2010 8th IEEE International, 2010, pp. 269-272.

[10] T. Ban and L. A. Naviner, "Optimized Robust Digital Voter in TMR Designs," in Proceedings of Colloque National GdR SoC-SiP, 2011.

[11] F. Zarafshan, G. Latif-Shabgahi, and A. Karimi, "A novel weighted voting algorithm based on neural networks for fault-tolerant systems," in Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, 2010, pp. 135-139.

[12] G. Latif-Shabgahi, S. Bennett, and J. M. Bass, "Smoothing voter: a novel voting algorithm for handling multiple errors in fault-tolerant control systems," Microprocessors and Microsystems, vol. 27, pp. 303-313, 2003.

[13] P. B. Saheb, M. k. S. , and D. S. P. k. , "a survey on voting algorithms used in safety critical systems," International Journal Of Engineering And Computer Science, vol. 2, pp. 2272-2275 2013.

[14] M. Aggarwal, "Hierarchal Object Oriented Fault Tolerant Secured and Atomic Mobile Agent Model," 2011.

[15] A. Bala and I. Chana, "Fault tolerance-challenges, techniques and implementation in cloud computing," *IJCSI International Journal of Computer Science Issues,* vol. 9, pp. 1694-0814, 2012.

[16] C. Sharma, S. Sabharwal, and R. Sibal, "Applying genetic algorithm for prioritization of test case scenarios derived from UML diagrams," *arXiv preprint arXiv:1410.4838,* 2014.

Manizheh Mirsaeidi received her B.Sc. degree in Computer Hardware Engineering from Khayyam university of Mashhad in 2008.and she is a master student in computer Architecture engineering in Islamic Azad University, Arak .Her area of research Includes computer architecture,
Fault tolerant systems and VLSI.

Dr Abbas Karimi was born in Ahwaz, Iran. He received his B.S. degree in computer hardware engineering and M.S. degree in computer software engineering. He received his Ph.D. degree in computer system engineering (Neuro-fuzzy, AI, and Fault-tolerant) from UPM, Malaysia. He is currently working as a senior lecturer and a faculty member with the department of computer engineering, I.A.U-Arak Branch. Apart of lecturing and supervising students, he also plays an active role in research and development. He was the leader of multiple research projects, and the author of three textbooks, many journals and actively participated in seminars and conferences conducted at national and international level. He is a senior member of IACSIT (International Association of Computer Science and Information Technology), member of IEEE (the Institute of Electrical and Electronics Engineers), IAENG (International Association of Engineers), SDIWC (the Society of Digital Information and Wireless Communications), and WASET (World Academy of Science, Engineering and Technology) and a reviewer of multiple cited and ISI journals, such as JEST (Journal of Electronic Science and Technology), IAJIT (the International Arab Journal of Information Technology), Engineering Letters, Entertainment ACM (Association of Computer Machinery), IJACSA (International Journal of Advanced Computer Science and Applications), and IJCSIS (International Journal of Computer Science and Information Security). His research interests include load balancing algorithms, real-time, distributed, parallel and fault-tolerant systems.