ijecs open access www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 7 (July 2013), Page No. 2272-2275

A Survey on Voting Algorithms Used In Safety Critical Systems

P.Babul Saheb[1], Mr. k.Subbarao[2] , Dr. S.Phani kumar[3]

[1]M.tech, Information Technology, Lakireddy Bali reddy college of engineering
Mylavaram, India
Saheb.babul@gmail.com

[2] Assoc.Professor, Information Technology, Lakireddy Bali reddy college of engineering
Mylavaram, India
ksubbarao_22@yahoo.co.in

[3]Professor&HOD, Dept.of Computer Science, Geetham's University
Hyderabad, India
phanikumar.s@gmail.com

Abstract - ***Several voting algorithms have been described to arbitrate the results of redundant modules in fault-tolerant systems. A voting scheme based on fuzzy set theory was introduced which softens the harsh behavior of the inexact majority voter in the neighborhood of the 'voter threshold' and handles uncertainty and some multiple error cases in the region defined by the fuzzy input variables. A set of fuzzy rules determines a single fuzzy agreeability value for each individual input which describes how well it matches the other inputs. Automatic fuzzy parameter selection based dynamic fuzzy voter for safety critical systems with limited system knowledge. Existing fuzzy voters for controlling safety critical systems and sensor fusion are surveyed and safety performance is empirically evaluated. The major limitation identified in the existing fuzzy voters is the static fuzzy parameter selection. Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges. Dynamic voter is designed in such a way that it can be plugged in and used in any safety critical system without having any knowledge regarding the data produced and their ranges.***

Keywords- ***fault tolerant system, fuzzy set theory, voter threshold, fuzzy input variables, automatic fuzzy parameter, dynamic fuzzy voter***

1. Introduction

Increasing reliability and safety is one of the primary concerns in many real-time systems. Safety critical systems are the systems which may lead to hazards, loss of lives or great damage to the property if they fail. There are different domains in which safety critical control systems are used (automotives) drive-by-wire systems, brake by wire systems used in cars; (medicine) infusion pumps, cancer radiation therapy machines, etc.; (military and space applications) rocket launchers, satellite launchers, etc.; and (industrial process control) robotics and consumer electronic appliances. There is a need to increase the reliability, availability and safety in all these applications. Faults that occur in these applications may lead to hazardous situations. If a single module or channel is used and when it becomes faulty environment [1] due to some noise the system may fail and hazard may occur. The inexact majority and weighted average voters are widely used in control and safety-critical applications. Inexact majority voters require an application-specific 'voter threshold' value to be specified, whereas weighted average voters are unable to produce a benign output when no agreement exists between the voter inputs. A major difficulty with inexact majority voters is the need to choose an appropriate threshold value, which has a direct impact on the voter performance. The problem of all documented weighted average voters is their inability to produce a benign output in cases of complete disagreement between the voter inputs. Both types of voters are unable to cope with uncertainties associated with voter inputs originated from erroneous software, noisy environment, or noisy hardware modules. The voter is experimentally evaluated from the point of view safety and availability and compared with the inexact majority voter in a Triple Modular Redundant structured framework. It is predicted that the fuzzy voter can be invaluable in at least two cases 1)as a substitute for the inexact majority voter in applications in which a small degradation in the safety performance of the system is acceptable at the cost of a large increase of its availability and a considerable decrease of its benign outputs 2) when arbitrating between the responses of dynamic channels of control systems which may include some uncertainty.

1.1 Voting algorithm

Voting algorithms have been extensively applied in situations where choosing an accurate result out of the outputs of several redundant modules is required. Voting algorithms are used to provide an error masking capability in a wide range of highly dependable commercial &research applications. These applications include N-modular redundant hardware systems and diversely designed software system based on N-version programming. The algorithms can be implemented in hardware or software depending on the characteristic of application and the type of voter selected. Generalized voters including majority, plurality, median and weighted average have been first introduced. The voting unit will be referred as voter . The outputs of redundant modules provide the voter inputs. voting algorithms and experimental evaluation of fault-tolerant mechanisms are described. We classify voter into three categories generic, hybrid and purpose build voters .Selected algorithms of each category are described for illustrative purposes and application areas proposed.

1) *SOFTWARE VOTER VS HARDWARE VOTER*

Low level bit wise voting with high frequency requires a hardware voter, whereas high level voting on the results of complex computations can be best performed in software[2].the flexibility of software enables a wide range of voting methods to be used with the incorporation of additional redundancy management procedures within the voter. The disadvantage of the software voter is that the voting may require more time to perform, simply because the processor cannot execute instructions as rapidly as a dedicated hardware voter. Depending on the data volume, the frequency of input & performance requirements working environment, system nature and the number of voters which must be provided hardware or software schemes can be selected.

1.2 Classification of voting algorithms

Voting algorithms can be grouped from various view points. They may be classified according to

- The implementation method - Software or Hardware voters
- The type of agreement – Exact or Inexact voter
- The output space cardinality size – small space or large output space
- The nature of working environment – synchronous or asynchronous voter

2) *EXACT (BIT-BY-BIT) VOTING*

In exact voting, agreement means that the redundant results are exactly same. Voting on the results of redundant modules with discrete values is straightforward, and is referred to as exact voting. A variant result is selected by the voter as an output if and only if it is in exact agreement with a Majority [(n+1)/2] of the other variant results. Another approach to voter implementation in hardware is presented. Here, a voter is attached to each of the communication channels in a parallel computing system. This approach is intended to provide application independent fault-tolerance. There are two obvious limitations with the exact (bit-by-bit) voting approach where design diversity is used, to minimize software faults, correct results may not have identical values and further effort may be desirable to ensure that the voter produces a sensible output where a Majority does not exist.

3) *INEXACT VOTING*

In Inexact voting agreement means that the results are not exactly the same but their difference from each other is smaller than a predefined application specific threshold In redundant sensor output which are read by digital computers or the output of diversely implemented software programs which handle floating point calculations, in such applications inexact voting is required [3].However, the voter threshold must be carefully selected. Because there is no mathematically precise way to define this value,. Inexact voting means several approaches to voting on correct results with similar but not identical results have been published.

4) *OUTPUT SPACE CARDINALITY*

In some applications, the output cardinality of redundant modules is finite. For example, a redundant program which performs a yes-no decision algorithm, has cardinality two. Similarly, a variant which can produce only five output values has cardinality five. In contrast, other application programs in a NMR configuration may have output spaces with very large cardinality that is they can produced any value within the computational range of the underlying hardware. Each of these groups needs a specific voting strategy because a voter which is appropriate for small output space redundant modules may not be effective when employed with large or infinite output space redundant modules.

5) *SYNCHRONOUS VOTING VERSUS ASYNCHRONOUS VOTING*

In an environment where redundant modules work synchronously by means of a common clock, voting is a straight forward result-to-result comparison with relatively low complexity. However, an application in which redundant modules work by their own clocks requires an asynchronous voting algorithm. Such algorithms are more complex because of the skew in the associated times of the redundant results and they require extra mechanisms (such as waiting loops, timing checks, or extrapolation of redundant results to a common time to be compared) which make them complex.

1.3 CLASSIFICATION OF VOTERS BASED ON THEIR FUNCTIONALITY

From the view point of functionality, voting algorithms can be classified into three main categories

- Generic voting algorithms(the result is generated by amalgamation or selection)
- Hybrid voting algorithms(generic plus additional information about variants)
- Purpose-built voters

1) *GENERIC VOTING ALGORITHMS:*

Algorithms which arbitrate only between the variant output results to produce a final output are classified as generic voters. The voter may either select one of the variant results or amalgamate them to produce a new, distinct value from the individual variant results. From this category unanimity, majority, plurality, 2-out-of-n, median, predictor, smoothing and vector voters are discussed.

- Result selection voters

Voter	Output	Problem
unanimity voter	All of variant results are in agreement.	Does not mask any variant fault. Reaching agreement on all the variant results is strictly necessary.
majority voter	produces an output among n variant results where at least n+1/2 variant results agree	
plurality voter	Implements m-out-of-n voting where m is less than strict majority. Have an odd number of variants. so, that a tie does not occur	
An inexact voter	output space with large cardinality	
median voter	produces a correct result up to maximum n-1/2 faulty inputs	no inherit fault detection capability
Consensus voting	introduced for multi version software with small output space If a voter is a yes/no decision maker its output space is binary and if the output of a voter can be any value its output space is infinite.	
2-out-of-n voter	output space cardinality is large	problem with this voter are twofold with small output space incorrect variant results increases

Table 1: Result selection voters

- Result amalgamation voters: The average voter output the average of its variant results..A weighted average voter calculates the weighted mean of the variant results. The weights can be predetermined or can be adjusted dynamically.

2) HYBRID VOTING ALGORITHMS

A group of voters which differ from generic voter ,use extra information such as the reliability level of variants on-line diagnosis information of modules or various probabilistic information to improve voting performance. This type of voter is called a hybrid voter.

- Voting algorithms incorporating prediction and smoothing: Embedded control applications are typically cyclic systems in which there exists some relationship between the result in one cycle & the result in next. Knowledge of this relationship between successive results is used in predictive voters [4] to produce results in cases of disagreement. The prediction-threshold value is application specific. Different methods of prediction have been used in the implementation of these voters..The smoothing voter extends the majority voter by adding an acceptance test which is based on the assumption that an excessive discontinuity between consecutive variant results is an indication of an error. In the smoothing voter when there is no agreement between variant results the closest result to the previous voter output is selected as the probable output for this cycle.

- Voting algorithms supplemented with diagnostic information on variant:
The benefits of integration of voting algorithms & self diagnostic elements in a TMR configuration of self-diagnosing elements has been addressed[5].A range of integrated voters which handles diagnosis information have been introduced. It has been shown that the appropriate use of diagnosis information in a fault masking system enables the voter to select more correct results than voting on element result values alone. Three general categories of integrated voters have been proposed which have ability to mask certain self-diagnosing element common error modes. The integrated voters capture certain advantages of both fault masking and fault detection & isolation features.

2. Existing system

Existing fuzzy voters for controlling safety critical systems and sensor fusion are surveyed and safety performance is empirically evaluated. The fuzzy parameter values are statically selected in this voter and the performance of the voter varies with variation of these fuzzy parameter values. Static selection of fuzzy threshold parameter values is a major limitation in this voter. Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges.

3. Proposed method

A dynamic or automatic fuzzy parameter selection method for fuzzy voters is proposed based on the statistical parameters of the local set of data in each voting cycle. Proposed dynamic fuzzy voter is dynamically self configurable. This dynamic fuzzy voter can be used for any systems with little system knowledge and for any input data ranges. The dynamic fuzzy voter can configure itself for any kind of dynamically changing data and it is the first attempt to our knowledge to automate a fuzzy voter. Proposed Dynamic fuzzy voter is giving safety if two of the three modules of the TMR System are error free.

3.1 PROBLEM DEFINITION

Designing automatic fuzzy parameter selection based dynamic fuzzy voter for safety critical systems with limited system knowledge. Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges. A dynamic or automatic fuzzy parameter selection method for fuzzy voters is proposed based on the statistical parameters of the local set of data in each voting cycle.

4. Conclusion& Future work

A self configurable dynamic fuzzy voter using statistical parameters is designed and safety performance is compared with the existing static fuzzy voter. In the existing static fuzzy voter, optimal fuzzy parameters are selected globally which are fixed throughout all the voting cycles, which is not a better idea. This static method is useful only in the situations where the data ranges are known based upon which optimal fuzzy parameters are selected to decide the fuzzy bandwidth. Dynamic fuzzy voter can dynamically configure itself for any data of any ranges as it decides the fuzzy parameters based upon the local data of a particular voting cycle, using statistical parameters like mean and standard deviation. This dynamic fuzzy voter can be used in any safety critical system without having much knowledge about the system, data and ranges of data. The safety performance of the static and dynamic fuzzy voters are compared empirically by running for 10000 voting cycles on a TMR simulator system. The dynamic fuzzy voter is given almost100% safety if two modules are error free and giving better safety performance than the static fuzzy voter if one module is error free and two modules have errors. Though there is no great improvement in the safety performance with the dynamic method, it is useful method since it automates the fuzzy voting technique.

One limitation identified with the dynamic fuzzy voter is the approach used for outlier detection. If two modules of TMR system, which have errors, wrongly or coincidentally satisfy the majority consensus and then the other module output which is actually correct is considered as an outlier. Hence there is a need to consider the module reliability history also apart from the statistical parameters, in the calculation of the fuzzy parameters and this remains the future work. This work can be extended to design Interval type fuzzy voter using fuzzy sets and proposed automatic parameter selection method may be applied to increase the safety performance. There is also scope for designing neuro-fuzzy voting system.

5. References

[1]Laprie, J.-C. (1985). Dependable computing and fault-tolerance: concepts and terminology. In Proceedings of the IEEE 15th Annual International Symposium on Fault-Tolerant Computing Systems (pp. 2-11).

[2] B. Parhami, "Voting Algorithms," IEEE Transactions on Reliability, vol. 43, pp. 617 629, 1994.

[3J.H.lala and R.E.harper,"Architectural principles for safety critical real systems applications,"proc.IEEE ,vol .82,no.1,pp.25-39,jan.1994

[4]"Threshold voting is fundamentally simpler than plurality voting"Int.J.Reliab.Qual.saf.Eng .vol.1,no.1,pp.95-102,1994

[5] G. Latif-Shabgahi, Performance analysis of software implemented inexact voting algorithms, Ph.D. Thesis, Department ofAutomatic Control and Systems Engineering, The University ofSheKeld, SheKeld, UK, 1999.

[6]G.Latif-shabgahi,"performance analysis of software Implemented Inexact voting algorithms ,"Ph.D thesis ,Depatment of automatic control and system engineering,the university of Sheffield,uk,1999.

[7]M.A.Vouk,A.M.Paradkar and D.F.Mcallister "modelling execution Time of multi stage N-version fault –tolerant software ",in proc.IEEE computer software and application conf .1990,pp.505-511.

[8] C.-L. Lei and H.-T. Liaw, "Efficient Parallel Algorithms for Finding the Majority Element," Journal of Information Sceince and Engineering, vol. 9, pp. 319-334,1993.

[9] B. Parhami, "Parallel Threshold Voting," The Computer Journal, vol. 39, pp. 692-700, 1996.
[10] G. Latif-Shabgahi, J.M. Bass, S. Bennett, A taxonomy for software voting algorithms used in safety-critical systems, IEEE Trans. Reliability (2004), in press.
[11] G. Latif-Shabgahi, S. Bennett, A.J. Hirst, Soft threshold voting scheme for safety-critical computer control applications, Proc. WRTP'03: 27th IFAC/IFIP/IEEE Workshop on Real-Time Programming, 14–17 May, Lagow, Poland, 2003.