Distributed Interval Voting with Node Failures of Various Types

Behrooz Parhami
Department of Electrical and Computer Engineering
University of California
Santa Barbara, CA 93106-9560, USA
parhami@ece.ucsb.edu

Abstract

Intervals constitute one of the most important tools for dealing with uncertainty in computations. Researchers in the fields of interval arithmetic and constraint propagation have devised elaborate methods for computing with interval variables. In this interpretation, an interval represents the proposition: "I don't know what the correct value is, but it cannot be outside this range." However, intervals also have another use, which is captured in the statement: "Any value in this range would be fine with me." In devising voting schemes for data fusion and fault-tolerant distributed computation, these two meanings, and a number of other lesser known variations, must be completely understood in order to design and implement meaningful voting strategies. Irregularities and paradoxes in voting schemes, extensively studied by mathematicians and social scientists, must also be taken into account to avoid serious pitfalls. In this paper, we discuss the two interpretations of interval voting, along with their practical implications, and show how voting strategies differ in their time and communication complexities, performance, and resilience according to the meaning intended and the types of failure assumed.

Keywords — Approval voting, Benign failure, Byzantine failure, Consensus, Data fusion, Dependable computing, Distributed computer system, Fault tolerance, Majority, Multichannel computation, Plurality, Uncertainty.

1. Introduction

Voting is used as a data fusion tool in connection with unreliable, incomplete, or inaccurate data collection (e.g., to accommodate failure-prone or low-precision sensors) and for realizing ultrareliable systems based on multichannel computation [20]. The sources of data on which voting is performed are various: identical hardware circuits, diverse software modules, multiple specialized sensors, and so on. Voting research in these contexts dates back half a century [24]. In the sociopolitical domain, voters provide inputs (cast their votes) and a vote tabulation system manipulates the input data to obtain the results [6, 23]. In fault-tolerant computing, the term "voter" has been used not for the entity that supplies an input (casts a vote) but for the hardware or software module that derives the result from the inputs provided by computation channels, alternates (in the case of software redundancy), or sensors. To avoid any confusion, particularly in light of the fact that any serious research on voting does need results from the social sciences, we use the following terminology:

Opinion: an input to the voting process

Participant: person or entity that supplies an opinion

Alternative: one of the entities about which opinions are expressed by participants

Fusion: combining the opinions according to the rules
of the voting system

Fuser: the hardware or software module that carries out the fusion algorithm

Outcome: result(s) of the voting process

An opinion may carry an integer or real weight, leading to a weighted voting system. However, our discussions in this paper are limited to unweighted voting. Note that contrary to current practice, an opinion need not be a simple indication of preference for one alternative. In approval voting, an opinion is a subset of all alternatives. In interval voting [18], alternatives are assumed to form an ordered set and an opinion is an interval of values specified by its least and greatest members. Finally, in yes/no voting [4], an opinion may include approved and disapproved subsets, with intervals used to specify the two sets in the case of totally ordered alternatives and contiguous approved and disapproved subsets.

Before proceeding further, it is necessary to clarify why the unconventional voting schemes just enumerated are relevant in the distributed computing context. Previously published research on voting in distributed systems has considered an opinion to be a single value, thus defining the fusion process as the task of coming up with a consistent choice (at all nonfailed sites) from among the opinions offered. Usually, a default value is built into the fusion process so as to ensure termination when some opinions are missing or excessively delayed. Distributed voting with set-valued opinions, and its special case of the sets being represented by intervals, is useful for the same reasons as centralized voting of these types: nonunique answers to a problem or uncertainties in the solution process [17, 20]. In the first instance, each opinion consists of a set of preferred or approved values that are used in reaching agreement. In the second case, bounds on the magnitude of a solution may be received by the fusion process, which uses them to derive bounds of higher quality or dependability.

Our failure model contains the standard failure types considered in distributed systems [10, 16]: benign (dormant) failures, which cause omission of messages or delays in sending them, and malicious (arbitrary, Byzantine) failures, which may cause a node to send conflicting messages to different nodes.

2. Preference Intervals

In the context of modern fault-tolerant digital systems, whether centralized or distributed, there is much more to voting than simple majority or plurality [14, 18, 19]. The choice of voting algorithm (or fusion process) has significant effects on system reliability, safety, and performance [8, 12, 13, 17]. Consider an ordered (finite or infinite) list of alternatives a_1, a_2, a_3, . . . and a finite set $\{p_1, p_2, . . . , p_n\}$ of participants in the voting process. Each participant p_i specifies an interval $[l_i, u_i]$ of preferred alternatives as an input to the fusion process, which then determines an outcome comprising of an alternative or a set of alternatives as the winner(s). Even though we show each input interval as a segment on the real number line in our graphical representations, intervals of interest to us have a finite set of discrete (integer or rational) values in almost all cases.

The simplest fusion process for the voting arrangement outlined above is that of approval voting with plurality selection rule: the outcome consists of a set of alternatives that are preferred by the largest number of participants [1, 3, 7]. In a safety-critical computer system, for example, the preference intervals may represent proposed safe set of values for a particular control system parameter, with differing opinions resulting from diverse evaluation criteria and/or algorithms. In expressing preferences, there is no *correct* or *best* alternative. So, even in the absence of faulty participants, there may be no subinterval in which all n input intervals overlap. Hence, a faulty participant (one that has used flawed hardware or software to reach its decision) may be indistinguishable from a participant that has an unconventional opinion. Figure 1 depicts an example for this type of voting.

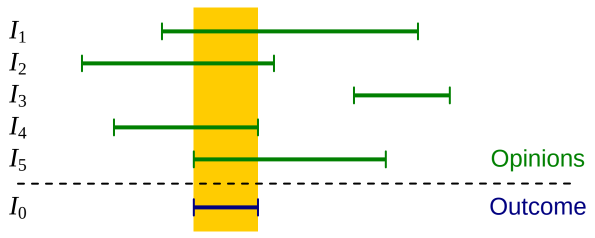

Fig. 1. Voting with preference intervals and plurality selection rule.

In practice, the situation with this voting strategy isn't always as clean as that shown in Fig. 1, and a number of complications may arise. Obviously, the fuser must determine areas of overlap between the largest possible number of intervals, which can be done using a fairly simple algorithm [18]. However, there may be multiple disjoint subintervals with identical approval levels, which cannot be combined into a single interval. Additionally, it is not clear whether the width of the intervals should in any way influence the fusion process. In other words, if one considers the possibility of a malicious participant, could it affect the outcome in an undesirable way by presenting a very wide or a very narrow interval as its opinion? We leave discussion of the latter problem to Section 5. To ensure uniqueness of the result of the fusion process when multiple disjoint subintervals have the same approval level, we take the lowest subinterval (the one with the smallest start and end points) among those having the same approval as the voting outcome.

3. Uncertainty Intervals

Intervals used to denote uncertainty represent the proposition: "I don't know what the correct value is, but it cannot be outside this range." Such intervals may arise from separate lower-bound and upper-bound calculations or be derived from approximate calculations or measurements, along with guaranteed error bounds. The latter intervals are of the form $[x - \varepsilon, x + \varepsilon]$ and are sometimes denoted as $x \pm \varepsilon$. By definition, such an interval is guaranteed to contain the correct value. Disregarding, for a moment, the possibility of a faulty participant, the fusion process is trivial, as the n intervals are guaranteed to overlap in at least one alternative (the correct one). The fused result based on n intervals is their intersection, which is frequently narrower than those offered by the participants as inputs. Thus, the diversity of interval opinions leads naturally to a type of refinement and increase in precision in the course of data fusion.

Now, allowing for faulty participants, the situation becomes more complicated. Any faulty participant that presents an interval containing one or more "correct" value(s) will not cause a problem, because its interval will still overlap with those of the nonfaulty participants. Thus, erroneous reduction in the lower bound, or increase in the upper bound, is a less serious form of error than the complementary events of erroneously larger lower bound or smaller upper bound. The latter type of error may cause the erroneous interval to have no overlap with one or more of the error-free intervals. Figure 2 depicts one such situation in which interval I_3 has no overlap with I_2 or I_4. Whether the error resides in the lower bound of I_3, or in the upper bounds of both I_2 and I_4, is subject to interpretation, with the single error being more likely than the double one in most practical cases.

Assuming equal failure probability for all participants, choosing the overlap of I_1, I_2, I_4, and I_5, that is, the subinterval $[l_5, u_4]$, as the voting outcome may appear reasonable. On the other hand, the shaded region shown in Fig. 2 is more likely to contain the correct result and may be the appropriate outcome in many application contexts. Defining the extended-union of k input intervals $[l_i, u_i]$, with $1 \le i \le k$, as the interval $[min_i\ l_i, max_i\ u_i]$, a safe voting strategy would set the plurality threshold according to the application context and then would choose the extended union of all qualifying result subintervals for which the number of supporting opinions is at or above the threshold as the voting outcome. In the case of Fig. 2, we have the following subintervals enjoying majority support: $[l_1, l_5]$ with 3 votes, $[l_5, u_4]$ with 4 votes, $[u_4, u_2]$ with 3 votes, and $[l_3, u_5]$ with 3 votes. The extended union of these qualifying subinterval is $[l_1, u_5]$, which is the appropriate voting outcome under this interpretation.

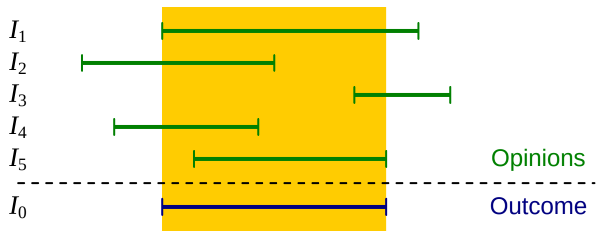

Fig. 2. Voting with uncertainty intervals and majority selection rule.

A possible voting algorithm, that is quite efficient, might work as follows. Rather than identify all of the qualifying subintervals, followed by the formation of their extended union, the algorithm eliminates all nonqualifying subintervals from either end, stopping once a qualifying interval has been encountered. For example, beginning at the left end of Fig. 2 and proceeding rightward, we increment a counter (initialized to 0) whenever the low end of an interval is encountered and decrement it for each high end. When the counter reaches 3, which represents a majority with 5 inputs, we have the lower end of the result interval. A similar process finds the upper end, scanning from the right end.

A subtle point regarding the example in Fig. 2 is worth mentioning. If, as is customary in 3-out-of-5 majority voting, we disqualify two suspect input intervals and use the extended union of the remaining three intervals as the outcome, the resulting interval would be wider than the shaded area shown. In other words, after disqualifying I_2 and I_3 which have the most extreme bounds, the outcome would be $[l_4, u_1]$ based on the three intervals I_1, I_4, and I_5. However, it is readily seen that this level of pessimism is unnecessary if failures are limited to at most 2.

4. Interval-Based Agreement

Consider a 3-node system, with the maliciously failed node 3 presenting two different intervals to the fusion processes at nodes 1 and 2 (solid and broken lines, respectively, in Fig. 3). It is clear from Fig. 3 that the voting outcome at nodes 1 and 2 will be different under the preference interval fusion scheme that uses the plurality rule to rank the candidate subintervals and then selects the subinterval with the highest support, breaking ties by favoring lower subintervals. What is particularly alarming in this example is the fact that the two fusion outcomes do not even overlap.

Similarly, the voting outcome will disagree under the uncertainty view of intervals, although in this case the resulting intervals, that is, $[l_1, u_1]$ and $[l_2, u_3]$ for versions 1 and 2 of I_3, are different from those obtained under the preference scheme. The nature of difficulty here is intrinsic to interval voting, rather than being due to the well-known requirements of distributed agreement [5, 22], or the equivalent Byzantine generals problem [10]. In this example, each of the two sites 1 and 2 has received a correct majority of opinions. Had we been dealing with scalar values rather than intervals, 2 out of 3 correct opinions would have been sufficient to draw correct and consistent conclusions at sites 1 and 2.

Let *Fuse*(•) denote the result of fusion on the set of intervals specified within the parentheses, where the fusion scheme can be any member of set of voting or fusion strategies applicable to the particular interval type. The interval-based agreement problem is formulated as follows. Given n nodes or sites, each having a local interval $[l_i, u_i]$, $1 \le i \le n$, use information exchange among the sites along with local decision processes to derive a global or outcome interval such that the termination, agreement, and integrity conditions are satisfied:

Termination – Every healthy site eventually reaches a decision

Agreement – All healthy sites agree on the same global or outcome interval I_G

Integrity – If all healthy sites propose intervals that overlap in the subinterval I, then $I \subseteq I_G$

The termination and agreement conditions are self-explanatory. The integrity condition is needed to prevent *Fuse*(•) from yielding a constant value, which would ensure agreement but is clearly undesirable because it disregards the input opinions.

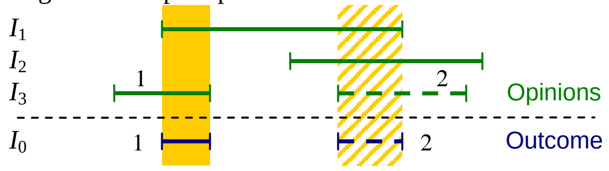

Fig. 3. Voting with preference intervals and one malicious participant.

We can define the notion of weak integrity for intervals in a manner similar to weak interactive consistency [11], by replacing the last condition above by the following requirement:

Weak integrity – If all sites propose intervals that overlap in the subinterval I, then $I \subseteq I_G$

Note that with weak integrity, even a single failed site among many can prevent a “correct” decision from being reached, although agreement must still be guaranteed in all cases. This weaker consistency requirement may be appropriate in some cases if it leads to significantly lower computational and communication overheads.

There are really two sets of issues to investigate in connection with interval-based distributed agreement. The first of these, that is, ensuring that all healthy sites eventually arrive at the same global interval (termination and agreement) is identical to the problem of distributed agreement on a single value, given that an interval is completely specified by its two endpoints. Any valid interactive consistency algorithm [2, 9], with its messages carrying two values, can be used for ensuring the termination and agreement conditions. During the algorithm’s execution, each site i keeps, and incrementally updates, a set of $2n$ values l_{ij} and u_{ij}, for $1 \le j \le n$, which represent the view of site i regarding the endpoints of the interval proposed by each site j. Once the interactive consistency algorithm has run its course, the voting or fusion process is executed at each site, producing identical results at all healthy sites. So, in addition to termination and agreement, the weak integrity condition is also satisfied with no additional effort.

The second set of issues pertains to whether maliciously failed sites can affect the agreement process in a manner that is more serious than the corresponding effect in interactive consistency with scalar values. In other words, we may ask whether a malicious participant with knowledge of the fusion algorithm can influence the fusion outcome in an undesirable way. We turn to this problem in the next section.

5. Extent of Failure Tolerance

We adopt the hybrid failure model of Meyer and Pradhan [16] which postulates the presence of b benign and m maliciously failed sites out of a total of n sites. Benign (dormant) failures cause omission of messages or delays in sending them, while malicious (arbitrary, Byzantine) failures may cause a node to send conflicting messages to different nodes.

Theorem 1: The correct outcome of interval voting with preference intervals cannot be undermined by b benign and m maliciously failed participants (among n), with $b + m = f$, iff $n \geq 3f + 1$ and the plurality threshold in the fusion process is set to $T = f + \lceil (n - f)/2 \rceil$.

Proof sketch: The minimum number $3f + 1$ of sites, being no less than $b + 3m + 1$, is such that a healthy site p_i is guaranteed to arrive at the same set of intervals $[l_{ij}, u_{ij}]$, for $1 \leq j \leq n$, following the interactive consistency phase of the fusion process. At this point, the opinion of any maliciously failed site that presented inconsistent opinions to different sites will have been replaced by the default interval. So, the worst case to be dealt with in regard to the m maliciously failed sites is when they present consistent opinions, but try to choose them to influence the voting outcome in an undesirable way. Furthermore, the absolute worst case (in the sense of increasing the vote count for an inappropriate subinterval) is when these m opinions happen to coincide with those of the b benignly failed sites. So, assume that $f = b + m$ sites provide the interval $[l, u]$ as their inputs to the fusion process. Unless at least $\lceil (n - f)/2 \rceil$ of the healthy sites, that is, one-half or more of the $n - f$ healthy sites, also present opinions that overlap with $[l, u]$, the latter will not affect the fusion outcome, given the chosen threshold $T = f + \lceil (n - f)/2 \rceil$. Furthermore, if all $n - f$ opinions from the healthy nodes overlap in the subinterval I, this subinterval will have a vote tally of at least $n - f$, which is easily shown to equal or exceed the threshold $T = f + \lceil (n - f)/2 \rceil$. Note that when $n = 3f + 1$, that is, with the minimum required number of sites, the threshold simplifies to $2f + 1$. □

Theorem 1 indicates that requirements for agreement with preference intervals are more stringent than those for agreement with scalar values (at least $3b + 3m + 1$ sites, compared with $b + 3m + 1$ sites). In other words, benign failures are no easier to deal with than malicious failures. On the positive side, with $n \geq 3f + 1$ sites, there is no way that malicious participants, acting alone or in concert, can force an inappropriate outcome by presenting extremely narrow or wide opinions.

An immediate consequence of Theorem 1 is that majority voting with preference intervals is never safe, regardless of the number of malicious participants. Note that the most appropriate default value for use with preference-interval voting is the empty interval, which is represented as $[x, y]$, with $x > y$; the interval $[x, x]$ is not empty, but contains a single point.

Considering now the case of voting with uncertainty intervals, we can prove the following result.

Theorem 2: The correct outcome of interval voting with uncertainty intervals cannot be undermined by b benign and m maliciously failed participants (among n) with $b + m = f$, iff $n \geq b + 3m + 1$, the plurality threshold is set at $T = n - f$, and the default opinion replacing missing inputs is the universal interval $[-\infty, +\infty]$.

Proof sketch: The minimum number of sites is such that any healthy site i is guaranteed to arrive at the same set of intervals $[l_{ij}, u_{ij}]$, for $1 \leq j \leq n$, following the interactive consistency phase of the fusion process. At this point, the opinion of any maliciously failed site that presented inconsistent opinions to different sites will have been replaced by the default interval $[-\infty, +\infty]$, which overlaps with all intervals from the healthy sites. Because the threshold is chosen as $n - f$, the overlap between all intervals from the $n - f$ healthy sites will be included in the result interval. The worst that can happen due to erroneous intervals presented by faulty participants is to widen the result interval, which does not affect the property that it includes the correct result. In other words, whereas faulty participants can degrade the precision of the outcome, they will not affect its correctness. □

The requirements of Theorem 2 for voting with uncertainty intervals are less stringent than those pertaining to preference intervals of Theorem 1 primarily because healthy sites are guaranteed to provide overlapping opinions. Theorem 2 postulates the same requirements as those for ordinary agreement using scalar values, with the only new element being a specification of the default interval. As a special case of Theorem 2, one may conclude that unweighted majority voting with uncertainty intervals is always safe when the malicious participants are in the minority.

It is instructive to study the worst-case behavior of voting with uncertainty intervals. In other words, how would malicious participants present their opinions so as to have the worst possible effect on the precision of the fusion result? Note that the worst case for preference intervals was exposed within the proof of Theorem 1.

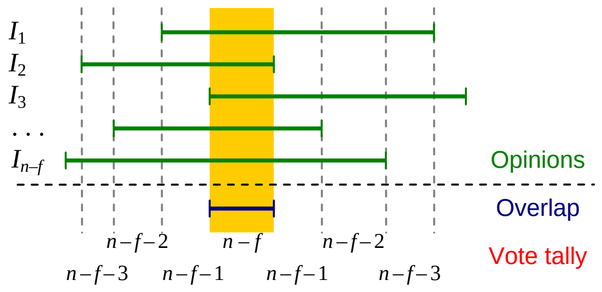

Fig. 4. Vote tallies for subintervals using only the opinions of healthy participants.

Consider the uncertainty intervals associated with the $n - f$ healthy participants (Fig. 4) and let there be no benignly failed site, that is, assume $b = 0$ and $f = m$. The aforementioned $n - f$ intervals will overlap in some nonempty subinterval, having a vote tally of $n - f$, which is flanked on both sides by subintervals of gradually decreasing vote tallies, as depicted in Fig. 4. When we include d default intervals $[-\infty, +\infty]$ which might have been inserted by the fusion process, the vote tallies will range from a minimum of d to a maximum of $n - m + d$, where $d \le m$. It is readily seen that for the remaining $m - d$ malicious participants to maximize the width of the outcome interval, they should present $[-\infty, +\infty]$ as their opinions, leading to vote tallies ranging from m to n. Because the threshold $n - f$ is at least equal to $2m + 1$ in this case, only subintervals that are contained within the opinions of a majority of the healthy participants are potentially included in the fusion outcome.

6. Some Extensions

We can extend the preceding observations and results to yes-no voting with intervals [4]. In yes-no voting, each participant specifies an approved interval and a disapproved interval, the idea being that the participant has some likes and some dislikes, but is indifferent to other alternatives. Yes-no voting, in the context of social choice, is useful when the alternatives can be placed in a total order (such as a political spectrum from extreme left to extreme right), with a participant indicating a range of acceptable alternatives and a range of unacceptable ones. This is a very useful paradigm in data fusion and dependable computation in that it makes it less likely for malicious participants to influence the fusion outcome in an undesirable way. This is because certain dangerous or unsafe values can be ruled out by healthy participants, thus reducing their eventual vote tallies.

By way of example, consider the situation depicted in Fig. 5. Each participant has specified an approved interval $I_j = [l_j, u_j]$ and a disapproved interval $I'_j = [l'_j, u'_j]$, where for $l'_j > u'_j$, a wraparound interval is assumed. A natural strategy in this case is to tally the level of support for each of the finite number of subintervals defined by the endpoints of the input intervals, counting each approval as +1 and each disapproval as –1, using positive and negative thresholds to decide on the fused version of the approved and disapproved results. Note that the approval and disapproval intervals need not be treated symmetrically. For example, Fig. 5 depicts a scheme in which any negative tally is viewed as overall disapproval. Clearly, the range of choices here is much wider than those of Figs. 1 and 2 and, thus, greater caution is required to ensure the reasonableness of the fusion strategy under all failure scenarios.

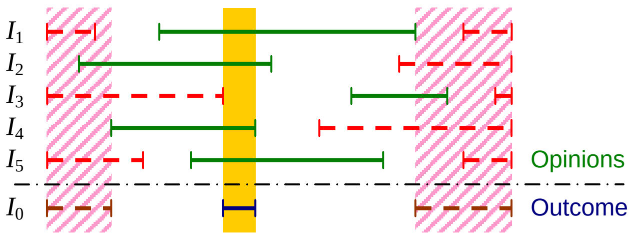

Fig. 5. Voting with approval (solid) and disapproval (dashed) intervals.

Weighted approval voting [15] provides a way of incorporating a-priori knowledge about the reliability of the various participants into account. An orthogonal notion to that of weights for various participants is to take degrees or levels of approval from each participant into account. For example, it may be desirable to associate a greater level of approval with the center of an interval than with its edges [20]. Further exploration is also possible with fuzzy intervals (derived from fuzzy sets) and with rough intervals, with the latter consisting of two nested intervals that define a “rough set” [21].

Another possible extension is to consider the combined effect of preference and uncertainty in the participants’ inputs. One way to accomplish this is to use the rough-set paradigm mentioned above. Participant i

presents the nested intervals $[l_j, u_j]$ and $[l'_j, u'_j]$, with $l'_j \leq l_j$ and $u'_j \geq u_j$. The interpretation of this opinion is that the values in $[l_j, u_j]$ are preferred and that those in $[l'_j, l_j]$ and $[u_j, u'_j]$ are neutral or uncertain. This is readily seen to be a particular form of yes-no voting.

7. Conclusions

In this paper, we have shown that distributed interval voting shares some of the difficulties of distributed voting with scalar values and that it presents a number of additional problems arising from its particular semantics. Two interpretations of intervals, those of preference and uncertainty, were discussed. Preference intervals are appropriate when there are multiple correct answers to a particular question. Uncertainty intervals arise, for example, from imprecise computations with guaranteed error bounds (interval arithmetic). other interpretations, including combined preference and uncertainty, are possible. Further research can proceed in many different directions, as outlined in Section 6. Extensions to other failure models are also possible.

References

[1] Alos-Ferrer, C., "A Simple Characterization of Approval Voting," *Social Choice and Welfare*, Vol. 27, pp. 621-625, 2006.

[2] Barborak, M., and M. Malek, "The Consensus Problem in Fault-Tolerant Computing," *ACM Computing Surveys*, Vo1. 25, No. 2, pp. , June 1993.

[3] Brams, S. J., and P. C. Fishburn, "Approval Voting," *American Political Science Review*, Vol. 72, pp. 831-847, 1978.

[4] Brams, S. J., and P. C. Fishburn, "Yes-No Voting," *Social Choice and Welfare*, Vol. 10, pp. 35-50, 1993.

[5] Castro, M., and B. Liskov, "Practical Byzantine Fault Tolerance and Proactive Recovery," *ACM Trans. Computer Systems*, Vol. 20, No. 4, pp. 398-461, November 2002.

[6] Dasgupta, P. and E. Maskin, "The Fairest Vote of All," *Scientific American*, Vol. 290, pp. 92-97, March 2004.

[7] Hodge, J. K., and R. E. Klima, *The Mathematics of Voting and Elections: A Hands-on Approach*, American Mathematical Society, 2005.

[8] HoseinNejad, R., A. Bab-Hadishahr, and P. Harding, "Fusion of Brake Pedal Sensors in by-Wire Cars: A Fuzzy Logic Approach," *Proc. 3rd IFAC Symp. Mechatronic Systems*, September 2004, pp. 639-644.

[9] Krol, T., "Interactive Consistency Algorithms Based on Voting and Error-Correcting Codes," *Proc. 25th Int'l Symp. Fault-Tolerant Computing*, 1995, pp. 89-98.

[10] Lamport, L., R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Trans. Programming Languages and Systems*, Vol. 4, No. 3, pp. 382-401, July 1982.

[11] Lamport, L., "The Weak Byzantine Generals Problem," *J. ACM*, Vol. 30, No. 3, pp. 668-676, July 1983.

[12] Latif-Shabgahi, G., Julian M. Bass, and Stuart Bennett, "A Taxonomy for Software Voting Algorithms Used in Safety-Critical Systems," *IEEE Trans. Reliability*, Vol. 53, No. 3, pp. 319-328, September 2004.

[13] Levitin, G., "Weighted Voting Systems: Reliability versus Rapidity," *Reliability Engineering & System Safety*, Vol. 89, pp. 177-184, 2005.

[14] Lorczak, P. R., A. K. Caglayan, and D. E. Eckhardt, "A Theoretical Investigation Generalized Voters for Redundant Systems," *Proc. International Symp. Fault-Tolerant Computing*, 1989, pp. 444-451.

[15] Masso, J., and M. Vorsatz, "Weighted Approval Voting," Unpublished manuscript, version of September 4, 2006.

[16] Meyer, F.J. and D.K. Pradhan, "Consensus with Dual Failure Modes," *IEEE Trans. Parallel and Distributed Systems*, Vol. 2, No. 2, pp. 214-222, April 1991.

[17] Parhami, B., "Threshold Voting is Fundamentally Simpler than Plurality Voting," *International Journal of Reliability, Quality, and Safety Engineering*, Vol. 1, No. 1, pp. 95-102, March 1994.

[18] Parhami, B., "Voting Algorithms," *IEEE Trans. Reliability*, Vol. 43, No. 4, pp. 617-629, December 1994.

[19] Parhami, B., "A Taxonomy of Voting Schemes for Dependable Multi-Channel Computations," *Reliability Engineering and System Safety*, Vol. 52, pp. 139-151, May 1996.

[20] Parhami, B., "Voting: A Paradigm for Adjudication and Data Fusion in Dependable Systems," in *Dependable Computing Systems: Paradigms, Performance Issues, & Applications*, ed. by H.B. Diab and A.Y. Zomaya, Wiley, 2005, pp. 87-114.

[21] Pawlak, Z., J. Grzymala-Busse, R. Slowinski, and W. Ziarko, "Rough Sets," *Communications of the ACM*, Vol. 38, No. 11, pp. 89-95, November 1995.

[22] M. Pease, R. Shostak, and L. Lamport, "Reaching Agreement in the Presence of Faults" *J. ACM*, Vo1. 27, No. 2, pp. 228, 1980.

[23] Saari, D. G., *Chaotic Elections: A Mathematician Looks at Voting*, American Mathematical Society, 2001.

[24] von Neumann, J., "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components," in *Automata Studies* (Annals of Mathematics Studies, No. 34), Ed. by C.E. Shannon and J. McCarthy, Princeton Univ. Press, pp. 43-98, 1956.