# Take Intelligent Risk and Optimize Decision Based on Time, Available Resources and Risk Tolerance Limits

Yue Yu*, Shangping Ren*‡
*Department of Computer Science*
*Illinois Institute of Technology*
*{yyu8, ren}@iit.edu*

Kevin A. Kwiat
*Information Directorate*
*Air Force Research Laboratory, Rome, NY*
*kwiatk@rl.af.mil*

## Abstract

*In real-time environment, data usually has a lifespan associated with it. The semantics and the importance of the data depend on the time when data is utilized. Hence, the process of getting a consensus data from a group of replicated units must not take longer time than the lifespan of the data. However, in real environment, every unit, faulty or non-faulty, may encounter delays when processing and sending their data which inevitably increases the time of acquiring a consensus. The latency for obtaining a valid data hence depends not only on the time when individual replicas make their votes, but also on the accuracy and credibility of the votes. Thus, a new metric, i.e. a credibility function, needs to be taken into account when evaluating expected time and deciding upon data replications. This paper presents analytical solutions for the expected time when dependable data can be obtained under different voting schemes. We show that if not all replicas are truthful, increasing replication does not reduce the time for obtaining valid results. When different types of resources are used to ensure the quality of the data, we show that the allocation of the resource plays an important role in satisfying both data availability and consistency constraints. We further demonstrate that that when point-based constraints may be intrinsically impossible to satisfy, a more general interval-based constraint can be used to obtain statistical solutions.*

## 1. Introduction

In presence of hardware or software failures, which may be caused by intentional attacks or unintentional human errors, replicating the functional units and then getting majority consensus of output data from these replicated units is a widely used approach to prevent the propagation of erroneous information to the ultimate end-users. To distinguish our use of replication from other methods for achieving fault tolerance, such as stand-by sparring, we note that our replicas submit their results that serve as votes for tabulation of consensus. Therefore, we refer to individual replicas as *voters*. Real-time data usually has a lifespan ($\Delta$) associated with it [13]. In other words, the semantics and the importance of the data depend on time. Data becomes stale, and using it beyond its intended lifespan can be catastrophic. Hence, the process of getting the consensus of the data from a group of replicated units and delivered it to the data client must not take longer than the lifespan of the data.

However, as contended by Dr. Lee [15] that though ironical, the advances in computer architecture and software have made it difficult or impossible to estimate or predict the execution time of software in a networked and embedded system. Every embedded unit, faulty or non-faulty, may encounter delays when processing and sending their voting data which inevitably increases the time to a consensus. Most voting schemes use a deadline to mark the end of the data's lifespan [13]. If a deadline is reached before the corresponding consensus is obtained, the data is discarded and a new round of data solicitation is initiated. This approach though guarantees data safety, it does not guarantee data availability.

The precise execution time of software in a networked and embedded system is difficult to predict; yet aggravating the difficulty are potential malicious attacks to the system. Although precise predictions are unobtainable, the statistical behavior of software and the network is, nevertheless, generally attainable. The paper presents our use of statistical data to increase real-time data availability based on: 1) expected time, and 2) how resource availability may impact the time a decision is made. In addition, we apply more generalized timing constraints - *interval-based timing constraints* [14, 26, 27] - that refine the timing

* Supported in part by NSF under grant CNS 0431832.
‡ Supported in part by Air Force Summer Faculty Fellowship.

constraints (where feasible resource allocations exist), whereas the point-based timing constraints are intrinsically infeasible.

The rest of the paper is organized as follows: Section 2 presents the background in voting mechanisms. Section 3 first gives formal definitions and terms that our analysis is based on. Then the analytical results on the expected time for obtaining valid votes in different voting protocols are given. Assuming all voters are truthful, we show how the number of replicas as well as their voting probability and credibility affect the data safety and availability in real-time environment. Moreover, in the situation where not all voters are truthful, we show that adding homogeneous resources does not improve much on the time of getting valid voting results. Section 4 presents how to adjust resource allocation to satisfy data consistency constraints while maintaining dependability and data availability in heterogonous environments. We show that point-based timing constraints are sometimes insufficient and therefore generally unsuitable for describing constraints in a networked and embedded system. We then identify the need for the more natural and general interval-based timing constraints. Section 5 gives an introduction to the interval-based timing constraints. Section 6 shows how to use interval-based timing constraint to describe constraints where feasible resource allocations exist. Related work is discussed in Section 7. Section 8 summarizes our conclusions and future work.

## 2. Background

In embedded systems, data sensed from the environment may have a timeliness parameter ($\Delta$). The timeliness pertains to how soon a data should be delivered at the user since the occurrence of reference datum it represents. It depicts that the data has a life time after the expiry of which it is of no use [12].

Consider an example presented in [13], the detection of an enemy plane flying at azimuthal location 35.0°. A radar unit may report detection at a reasonable close azimuth 35.1°. This report should be delivered to the Command and Control center (C2) within a few seconds of the presence of enemy plane at the reported azimuth. With such tolerances in reporting, a missile fired at the enemy plane by C2 can still be within intended hit range. However, a faulty radar unit may report the plane to be at, say, 55.0° azimuth to prevent the plane from being hit or send an accurate azimuth but so late that the plane has left the hit range. To avoid single point of failure, multiple radar systems are deployed and we use voting protocol to decide the correct data.

The boolean expression ($T(d) < \Delta(d)$) tests if the time $T(d)$ for the data $d$ to reach its client meets the timing constraint of $\Delta(d)$. A voting protocol should validate $d$ for reasonable accuracy and for timely delivery with respect to $\Delta(d)$, in the presence of possible failures. For data safety reasons, if the decision unit cannot decide on $d$ with reasonable assurance within the data delivery deadline $\Delta(d)$, it discards the data $d$ and initializes a new round of data collection. This approach guarantees the data safety with close to 100% assurance (at least from the decision unit perspective), but the data availability is not unrivaled especially when unexpected delays occur at sensing, processing or transporting units.

As argued by Dr. Lee in his invited talk [15] that precise timing estimation of software execution time in embedded networked systems is impossible. Instead, what we may know is a statistic time within a range. For instance, upon an enemy plane has emerged in the region at time 0, it usually takes a non-faulty radar $t_1$ to $t_2$ seconds to detect it and transmit the information to the control center. In other words, normally, the command and control center should receive the plane information within the $[t_1, t_2]$ time interval. However, the exact time may only be known statistically even for non-faulty units. Thus, knowing expected time when valid data will arrive prepares the data end user for appropriate actions if the expectation is not realized. A further observation is that under non-faulty circumstance, if data are only statistically certain, increasing the number of replicas (sensors in our example) will increase probabilistic guarantees.

## 3. Expected Time for Obtaining a Valid Vote in Different Voting Protocols

In this section, our discussion is based on the assumption that all the $n$ sensor units provide datum $D_i$ to the decision unit(s) and the inherently correct data value is $D$. The information credibility may not be at the fixed 100% level, that is, $D_i$ may not always be the same as $D$. Instead, it may be time dependent. We use a credibility function $C_i(t)$ to describe the probability that $D_i$ is the same as $D$ at time $t$.

The following voting schemes are discussed here:

- **1-out-of-n scheme.** Under truthful assumption, we have that $D_i = D$, that is, every sensor unit provides correct data and $C_i(t) = 1$. In this case, once the decision unit gets a datum $D_i$ from any sensor, it can deliver $D_i$ to the user without waiting for data from other sensors.
- **k-out-of-n scheme.** In the presence of faulty voters, a datum $D_i$ given by a faulty voter may not

be in agreement with the data of non-faulty voters. However, a datum $D_i$ given by a non-faulty voter will be in close agreement with (or simply the same as) the data $D$ of all the other non-faulty voters. We assume that the inherently correct data $D$ is in the majority so that $D$ can be determined by majority voting protocols. The credibility function $C_i(t)$ is given to be monotonic with bound of [0, 1]. The monotonicity indicates that with more time, we would get more trustworthy data.

We further assume that the probability distribution function for the time a sensor $i$ takes to obtain and transmit data is given as $V_i(t)$. In other words, the probability that the decision unit get a datum from a sensor $i$ by time $t$ is given by $V_i(t)$.

To formulate the problem, let $X_i$ be the random variable representing if the decision unit get a vote from the $i$th sensor

$$X_i = \begin{cases} 1, & \textit{if } \textbf{the vote of the i'th sensor is given} \\ 0, & \textit{otherwise} \end{cases} \tag{1}$$

Thus, $P\{X_i = 1\} = V_i(t)$, $P\{X_i = 0\} = 1 - V_i(t)$

Moreover, we interpret data credibility as the probability that a given data $D_i$ agrees with the inherently correct data $D$. Let $Y_i$ be the random variable representing whether the data $D_i$ agrees with $D$, that is

$$Y_i = \begin{cases} 1, & \textit{if } \textbf{the vote given by the i'th sensor is } D \\ 0, & \textit{otherwise} \end{cases} \tag{2}$$

Thus, $P\{Y_i = 1 | X_i = 1\} = C_i(t)$, $P\{Y_i = 0 | X_i = 1\} = 1 - C_i(t)$. Therefore, the probability that the decision unit get a correct vote from the $i$th sensor is

$$\begin{aligned} p_i &= P\{Y_i = 1 \cap X_i = 1\} \\ &= P\{Y_i = 1 \mid X_i = 1\} \times P\{X_i = 1\} = C_i(t)V_i(t) \end{aligned} \tag{3}$$

and the probability that the decision unit cannot get a correct vote (either the vote is not given, or the given vote is incorrect) from the $i$th voter is

$$\begin{aligned} q_i &= P\{Y_i = 0 \cup X_i = 0\} \\ &= P\{\overline{Y_i = 1 \cap X_i = 1}\} = 1 - p_i = 1 - C_i(t)V_i(t) \end{aligned} \tag{4}$$

When all sensors are homogeneous, i.e., their $C_i(t)$ and $V_i(t)$ are identical, the probability that at least $k$ similar (or the same as $D$) votes are collected is the summation of binomial distributions

$$\begin{aligned} & P\left\{\sum_{i=1}^{n}(X_i \wedge Y_i) \geq k\right\} = \sum_{i=k}^{n}\binom{n}{i}p^i(1-p)^{n-i} \\ & \textit{where} \\ & p = p_1 = \cdots = p_n = C(t)V(t) \end{aligned} \tag{5}$$

Note that $p$ is a function of $t$, it follows that (5) is the probability that at least $k$ similar votes are collected before time $t$. Let random variable $T$ represent the time at which enough similar votes (at least $k$) are collected, i.e., the decision time, we have

$$P\{T \leq t\} = \sum_{i=k}^{n}\binom{n}{i}p^i(1-p)^{n-i} \tag{6}$$

*and*

$$P\{T > t\} = 1 - \sum_{i=k}^{n}\binom{n}{i}p^i(1-p)^{n-i} = \sum_{i=0}^{k-1}\binom{n}{i}p^i(1-p)^{n-i}$$

Therefore, the expected time that at least $k$ same/similar votes are collected by the decision unit is

$$\begin{aligned} E[T] &= \int_0^\infty P\{T > t\}dt \\ &= \int_0^\infty \sum_{i=0}^{k-1}\binom{n}{i}\left(C(t)V(t)\right)^i\left(1 - C(t)V(t)\right)^{n-i}dt \end{aligned} \tag{7}$$

Note that in (7), different $k$'s are used in distinct voting schemes. In *1-out-of-n* scheme where all sensors are truthful, we have that $k = 1$. Whereas in *k-out-of-n* scheme, we have $k = \lceil (n+1)/2 \rceil$ in majority voting protocols and $k = \lceil 2n/3 \rceil$ in the more stringent Byzantine voting protocols. In the following subsections, we discuss these schemes separately, assuming $C(t)$ and $V(t)$ are given.

## 3.1. Truthful Voters

Under this scheme, we have $k = 1$ and $C(t) = 1$ in (7). We further assume that $V(t)$ is uniformly distributed over the interval $[0, T_1]$, i.e.,

$$V(t) = \begin{cases} \dfrac{t}{T_1}, & \textit{if } t \in (0, T_1) \\ 1, & \textit{otherwise} \end{cases} \tag{8}$$

Substitute $k$, $C(t)$, and $V(t)$ in (7), we have

$$E[T] = \int_0^{T_1}\left(1 - \frac{t}{T_1}\right)^n dt + \int_{T_1}^{\infty}(1-1)^n dt = \frac{1}{n+1}T_1 \tag{9}$$

Equation (9) indicates that as $n$ increases, $E[T]$ decreases. In other words, under truthful assumption, resource availability positively impact data availability and system dependability. More careful observation reveals that the voting subsystem under truthful assumption is in fact a parallel system where the probability that the decision unit get at least one correct data from $n$ sensors is

$$\begin{aligned} P\{\sum_{i=1}^{n}(X_i \wedge Y_i) \geq 1\} &= 1 - P\{\sum_{i=1}^{n}(X_i \wedge Y_i) = 0\} \\ &= 1 - \prod_{i=1}^{n} q_i = 1 - \prod_{i=1}^{n}(1 - C_i(t)V_i(t)) \end{aligned} \tag{10}$$

in which $\prod q_i$ characterizes a parallel system. In such a

system, sensor units work in a "co-operative" way. Therefore, adding resources (more homogenous sensor units) to the subsystem improves its performance and thus reduces the expected decision time.

Similarly, consider a situation in which the data coming from the sensors are at constant rate ($\lambda$) for any unit interval, i.e., the number of data within a unit time is constant over time. Based on probability theory, we know that such event probability distribution can be modeled as exponential distribution, with probability distribution function given below

$$V(t) = 1 - e^{-\lambda t}, \quad t \geq 0 \tag{11}$$

Substitute $k$, $C(t)$, and $V(t)$ in (7), we have

$$E[T] = \int_0^{\infty} e^{-n\lambda t} dt = \frac{1}{\lambda} \cdot \frac{1}{n} \tag{12}$$

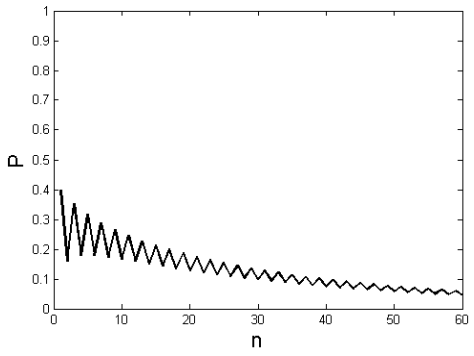
Therefore, though the probability distribution functions for voting time are different, if all the sensors are truthful, increasing $n$, i.e., the number of resources, reduces the expected time to obtain assured votes.

### 3.2. Untruthful Voters

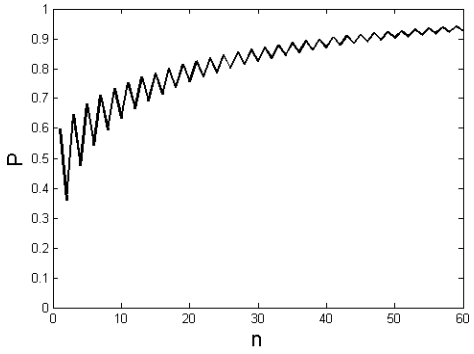
Under untruthful voter scenario, $k$ is determined by the specific majority voting protocol (we use $k = \lceil (n+1)/2 \rceil$ in the following discussions). We further assume that $C(t)$ is uniformly distributed over the interval $[0, T_2]$ and $V(t) = 1$[1]. From (5), we can derive the probability of getting a valid data before time $t$:

$$P(t) = \sum_{i=\lceil (n+1)/2 \rceil}^{n} \binom{n}{i} \left(\frac{t}{T_2}\right)^i \left(1 - \frac{t}{T_2}\right)^{n-i} \quad (t \in [0, T_2]) \tag{13}$$

Figure 1 shows the relationships between $P(t)$ and $n$ under different $t$. As can be seen, when $t = 0.4T_2$, which means that the probability of getting a valid vote from an individual voter by time $t$ is less than 50%, adding more homogeneously untruthful resources only makes it harder to get a consensus within given time. Intuitively, if over 50% chance a voter is to lie, adding more such voters only reduce the probability of getting valid votes within a given time. However, when $t = 0.6T_2$, which means that the probability of getting a valid vote from an individual voter by time $t$ is greater than 50%, adding more homogeneous resources facilitates the decision process, thus resulting in an increasing probability of obtaining a valid vote. The question now is: how does the resource availability influence the average decision time and thus the data availability?



(a) $t = 0.4\ T_2$



(b) $t = 0.6\ T_2$

**Figure 1. The relationship between *P*(*t*) and *n***

Substitute $C(t)$, and $V(t)$ in (7), we have

$$E[T] = \int_0^{T_2} \sum_{i=0}^{k-1} \binom{n}{i} \left(\frac{t}{T_2}\right)^i \left(1 - \frac{t}{T_2}\right)^{n-i} dt + \int_{T_2}^{\infty} \sum_{i=0}^{k-1} \binom{n}{i} (1)^i (1-1)^{n-i} dt$$
$$= \sum_{i=0}^{k-1} \binom{n}{i} \int_0^{T_2} \left(\frac{t}{T_2}\right)^i \left(1 - \frac{t}{T_2}\right)^{n-i} dt \tag{14}$$

Make the substitution $x = t/T_2 \Rightarrow dx = (1/T_2)dt$ in (14), we have

$$E[T] = \sum_{i=0}^{k-1} \binom{n}{i} \int_0^1 x^i (1-x)^{n-i} T_2 dx \tag{15}$$

Integrate by parts, we have

$$\int_0^1 x^i (1-x)^{n-i} dx = \frac{1}{i+1} \left[ x^{i+1} (1-x)^{n-i} \Big|_{x=0}^{1} - \int_0^1 x^{i+1} d(1-x)^{n-i} \right]$$
$$= \frac{n-i}{i+1} \int_0^1 x^{i+1} (1-x)^{n-i-1} dx \tag{16}$$

[1] Although it is unreasonable to assume $V(t) = 1$, i.e., a sensor is constantly giving out vote to the decision unit, we do this to simplify calculations and because not $V(t)$ alone but $C(t) \times V(t)$ characterizes the possibility that the decision unit gets a vote valued $D$, which is the inherently correct data.

Use mathematical induction on (16), we can prove that

$$\int_0^1 x^i (1-x)^{n-i}\,dx = \frac{i!(n-i)!}{(n+1)!} \tag{17}$$

Therefore, from (15) and (17), we have that

$$E[T] = T_2 \sum_{i=0}^{k-1} \binom{n}{i} \frac{i!(n-i)!}{(n+1)!} = T_2 \sum_{i=0}^{k-1} \frac{1}{n+1} = \frac{k}{n+1} T_2 \tag{18}$$

Given that $k = \lceil (n+1)/2 \rceil$ and $n$ is large, we have

$$E[T] = T_2/2 \tag{19}$$

Therefore, in an open hostile environment where not all voters are truthful, adding homogeneous resource does not impact the expected time of getting a valid vote. The intuitive explanation for this result is that the integrated effects of Figure 1(a) and (b) are neutralized.

Similarly, when the credibility function $C(t)$ is exponentially distributed over the interval $[0,\infty)$ with average rate $\lambda$, that is,

$$C(t) = 1 - e^{-\lambda t},\ t \in [0,\infty) \tag{20}$$

Using (7), we have

$$E[T] = \sum_{i=0}^{k-1} \binom{n}{i} \int_0^\infty \left(1-e^{-\lambda t}\right)^i \left(e^{-\lambda t}\right)^{n-i} dt \tag{21}$$

Make the substitution where $x = e^{-\lambda t} \Rightarrow dx = -\lambda e^{-\lambda t} dt = -\lambda x dt$, we have

$$\begin{aligned} E[T] &= \sum_{i=0}^{k-1} \binom{n}{i} \int_1^0 (1-x)^i x^{n-i} \frac{1}{-\lambda x} dx \\ &= \frac{1}{\lambda} \sum_{i=0}^{k-1} \binom{n}{i} \int_0^1 (1-x)^i x^{n-i-1} dx \end{aligned} \tag{22}$$

Integrate by parts and use mathematical induction, we can prove that

$$\int_0^1 (1-x)^i x^{n-i-1} dx = \frac{i!(n-i-1)!}{n!} \tag{23}$$

Therefore, from (22) and (23), we have that,

$$\begin{aligned} E[T] &= \frac{1}{\lambda} \sum_{i=0}^{k-1} \binom{n}{i} \frac{i!(n-i-1)!}{n!} \\ &= \frac{1}{\lambda} \sum_{i=0}^{k-1} \frac{n!}{i!(n-i)!} \frac{i!(n-i-1)!}{n!} = \frac{1}{\lambda} \sum_{i=0}^{k-1} \frac{1}{n-i} \end{aligned} \tag{24}$$

where $k = \lceil (n+1)/2 \rceil$. The relationship between $E[T]$ and $n$ in case of exponential distribution is illustrated in Figure 2. As can be seen, when the number of working sensors are small, increasing the number of sensors generally decreases expected decision time. However, since

$$\lim_{n\to\infty} \sum_{i=0}^{\lceil (n+1)/2 \rceil - 1} \frac{1}{n-i} = \ln n - \ln \frac{n}{2} = \ln 2 \approx 0.6931 \tag{25}$$

The expected decision time converges at $\ln 2/\lambda$ and no further decrease can be achieved by adding more resources. For example, with 11 sensors, the expected decision time is $0.7365/\lambda$, while with 23 sensors, the expected decision time is $0.7144/\lambda$ — a 3.0% time gain is at the cost of more than twice the resources.

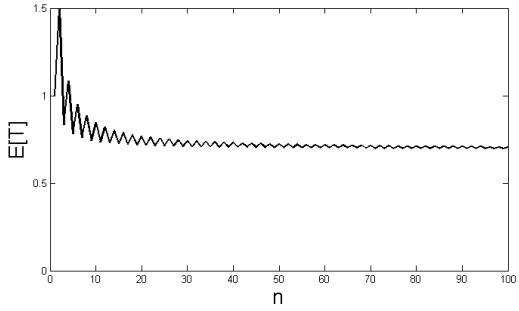



**Figure 2. Expected Decision Time with $\lambda$=1**

## 4. Data Availability and Its Consistency Constraints

In heterogonous environments, data come from different sources. However, they need to be semantically coherent if such data are to be delivered to the end users. Consider a setting in which two types of sensors are deployed in a region to monitor potential targets. One type is infrared (*IR*) sensors used for producing thermo graphic images. Another type is radio wave (*RW*) sensors used for measuring speed of targets. The *IR* sensors produce clear and reliable data. However, due to electromagnetic interferences, *RW* sensors produce less reliable data and hence it is necessary to get a consensus from other *RW* sensors deployed in the region. Furthermore, in order for a soldier or Command and Control Center to take critical actions, the data from two different sources (*IR* and *RW*) must be coherent — not only they provide the correlated information, but also the information from two different sources must arrive at the requester within a limited time frame $\Delta$, or these two sets of information may not be related.

Now, assume that the external requests come at time $t_0$, and a valid datum (already checked for majority) from group *IR* becomes available at $t_1$ and a valid datum from group *RW* becomes available at $t_2$. Given the assumptions and results in Section 3, together with the relative span requirement $\Delta$, we have the following timing constraints

$$\begin{cases} t_1 - t_0 = 1/\left(\lambda_1 |IR|\right) \le \Delta_1 \\ t_2 - t_0 = \left(\ln 2\right)/\lambda_2 \le \Delta_2 \\ |t_1 - t_2| \le \Delta \end{cases} \tag{26}$$

The first two equations come from (12) and (24) where $|IR|$ and $|RW|$ ($|RW|$ is not reflected in the equation because of (25)) are the number of sensors under group *IR* and *RW* [2]. $\Delta_1$, $\Delta_2$ are the individual deadline requirements for the two data, respectively, and $\Delta$ is the required maximum time span of the two replies. We further convert the constraints into the form $x_1 - x_2 \le d$ and instantiate a timing constraint graph as shown in Figure 3.

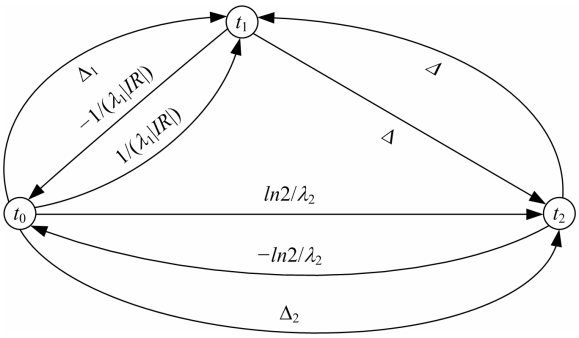



**Figure 3. Timing Constraint Graph**

With such a constraint graph, the Bellman-Ford algorithm is used to detect if the graph has negative-weight cycles. The existence of negative-weight cycle in the constraint graph indicates that the constraints are unsatisfiable. The cycle $\overrightarrow{t_0t_1}(1/(\lambda_1 | IR|))$, $\overrightarrow{t_1t_2}(\Delta)$, $\overrightarrow{t_2t_0}(-\ln 2/\lambda_2)$ is negative if

$$\frac{1}{\lambda_1 | IR|} + \Delta - \frac{\ln 2}{\lambda_2} < 0 \qquad (27)$$

which indicates that the time-consistency constraint between the two data is infeasible. This means that group *IR* is so fast that group *RW* cannot match with it. In this case, the system designers must reconsider the specification or declare exception handling to relocate resources (e.g., cut down the number of sensors in group *IR*). However, such resource reduction must not be at the cost of reduced individual data availability level, that is, if $1/(\lambda_1|IR|)$ becomes too large because of decrease of $|IR|$, there will be another negative cycle $\overrightarrow{t_0t_1}(\Delta_1), \overrightarrow{t_1t_0}(-1/(\lambda_1 | IR|))$ if

$$-\frac{1}{\lambda_1 | IR|} + \Delta_1 < 0 \qquad (28)$$

which means that group IR cannot meet the individual deadline requirement and thus dependability requirement is violated.

Therefore, to adjust the cardinality of group *IR* to satisfy the time-consistency constraint of data and retain data availability, both (27) and (28) need to be taken into account:

$$\begin{cases} \frac{1}{\lambda_1 | IR|} + \Delta - \frac{\ln 2}{\lambda_2} \ge 0 \\ -\frac{1}{\lambda_1 | IR|} + \Delta_1 \ge 0 \end{cases} \Rightarrow |IR| \in \left[ \frac{1}{\lambda_1 \Delta_1}, \frac{\lambda_2}{\lambda_1} \frac{1}{\ln 2 - \lambda_2 \Delta} \right] \qquad (29)$$

However, (29) may be intrinsically infeasible when

$$\frac{1}{\lambda_1 \Delta_1} > \frac{\lambda_2}{\lambda_1} \frac{1}{\ln 2 - \lambda_2 \Delta} \Rightarrow \frac{\ln 2}{\lambda_2} > \Delta + \Delta_1 \qquad (30)$$

Equation (30) implies another negative cycle in the constraint graph. This intrinsic infeasibility comes from the fact that when not all voters are truthful, any attempt to shorten the expected time of getting a valid vote by adding homogeneous resource will be futile.

Under soft-deadline settings, with the probability distribution function in (5), it is possible to relax the timing constraints as in (26) by adopting the *interval-based timing constraints* [14, 26, 27]. In the next two sections, we show that when timing constraints are relaxed from point-based to interval-based, we are able to obtain a feasible solution to the problem of deciding necessary number of voters.

## 5. Interval-based Constraint Model

Generally speaking, real-time constraints can be categorized into two classes, namely, deadline constraints or delay constraints. More specifically, a deadline constraint between two events with timestamps $\sigma$ and $\gamma$ is modeled as $\sigma + d \ge \gamma$, while a delay constraint is modeled as $\sigma + d < \gamma$, where $d \ge 0$ is a constant representing a deadline or a delay.

For self-completeness, we quote the related definitions (Definition 1 through 3) from [14] in the following.

**Definition 1 (Timestamp)** A timestamp *I* consists of a pair of time points: [*min_time, max_time*] where *min_time* and *max_time* are the earliest and latest time point at which an event may occur, respectively. Moreover, given a timestamp *I*, we assume the probability density function of *X* representing the time point at which the event may occur is $f(x)$.

**Definition 2 (Function *min, max,* and *len*)** Given a timestamp *I* = (*min_time, max_time*), the *min*, *max* and *len* functions of a timestamp *I* are defined as follows:

[2] Although the actual decision times may deviate from the expected decision times, we use expectation here to simplify the discussion. However, the same discussion follows if deviations, e.g., standard deviations, are added. Also note that since we use expected values, the constraint specification is in fact soft. A hard real-time specification which requires that data be delivered with probability 1 will not be appropriate here since it would take arbitrarily long time to make the probability reasonably close to 1 under exponential distribution.

$$min(I) = min\_time$$
$$max(I) = max\_time$$
$$len(I) = max(I) - min(I)$$

For the sake of brevity, we use $min_k$, $max_k$, and $len_k$ to denote $min(I_k)$, $max(I_k)$, and $len(I_k)$, respectively, where $I_k$ is a timestamp.

**Definition 3 (Interval-based timing constraint)** An interval-based deadline constraint with a confidence threshold is given by:

$$c^+ : I_2 - I_1 \le d \quad with\ P \tag{31}$$

where $I_1$ and $I_2$ are timestamps, $d \ge 0$ is a constant representing a deadline, and $P$ is a confidence threshold ranging from 0% to 100%.

Given the definitions above, we present Theorem 1 [27] that calculates the satisfaction probability of a deadline constraint and under arbitrary probability density functions. If the calculated satisfaction probability is less than the confidence threshold, we know at compile time that the specified constraint is not satisfiable.

**Theorem 1** Given a deadline constraint $c^+$: $I_1 + d \ge I_2$, where $d \ge 0$, and $f(x)$, $g(y)$ are the probability density functions of independent event occurrences on interval $I_1$ and $I_2$, respectively, the satisfaction probability of $c^+$, $SP|_{c^+}$ is given by the expression:

$$SP|_{c^+} = \frac{\int_{x=MAX(\min_1,\min_2-d)}^{\max_1} f(x)\int_{y=\min_2}^{MIN(x+d,\max_2)} g(y)dydx}{\int_{\min_1}^{\max_1} f(x)dx \cdot \int_{\min_2}^{\max_2} g(y)dy} \tag{32}$$

Proof:

Let $X \in I_1$, $Y \in I_2$ be two continuous random variables with density functions $f(x)$ and $g(y)$. Since the two random variables are mutually independent, the joint density function $z(x, y)$ is simply the product of their individual density functions, as shown in Figure 4.

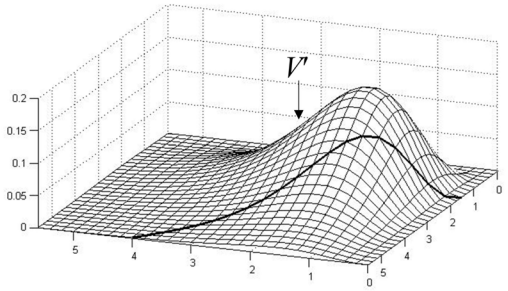



**Figure 4. Joint density function of independent events on two intervals.**

Furthermore, the joint cumulative distribution over $I_1 = [min_1, max_1]$ and $I_2 = [min_2, max_2]$, denoted as $V$, is:

$$V = \int_{x=\min_1}^{\max_1}\int_{y=\min_2}^{\max_2} z(x,y)dydx = \int_{\min_1}^{\max_1} f(x)dx \cdot \int_{\min_2}^{\max_2} g(y)dy$$

The satisfiable region, denoted as $V'$, is the intersection between the region $y \le x+d$ and $V$, as shown in Figure 4. The bold line represents the intersection between the joint density function and the plane $y = x+d$.

The satisfaction probability is thus the ratio between the satisfiable region $V'$ and the joint cumulative distribution $V$.

To calculate $V'$, we project the plane $y = x+d$ and the surface $z(x, y)$ onto the X-Y plane and consider the relationship between the line $y = x+d$ and the four points $(min_1, min_2)$, $(min_1, max_2)$, $(max_1, min_2)$, and $(max_1, max_2)$. There are only six possible relationships which correspond to the six permissible configurations in [14]. Two of the six cases are trivial:

- $(min_1, max_2)$ is below the line $y = x + d$, that is, $max_2 \le min_1+d$, which implies a 100% satisfaction probability;
- $(max_1, min_2)$ is above the line $y = x + d$, that is, $min_2 > max_1+d$, which implies a 0% satisfaction probability.

The four non-trivial configurations are:

- $\alpha\beta$ configuration, where $min_1+d \le min_2 \wedge min_2 < max_1+d \le max_2$
- $\alpha\gamma$ configuration, where $min_1+d \le min_2 \wedge max_2 < max_1+d$
- $\beta\beta$ configuration, where $min_2 < min_1+d \le max_2 \wedge min_2 < max_1+d \le max_2$
- $\beta\gamma$ configuration, where $min_2 < min_1+d \le max_2 \wedge max_2 < max_1+d$

Figure 5 gives graphical view for the four configurations. For simplicity and consistency, we adopt the same naming scheme for the configurations as in [14] and name the four configurations as $\alpha\beta$, $\alpha\gamma$, $\beta\beta$, and $\beta\gamma$, respectively.

The satisfiable region $V'$ in each of them is as following:

- $\alpha\beta$ configuration

$$V' = \int_{x=\min_2-d}^{\max_1} f(x)\int_{y=\min_2}^{x+d} g(y)dydx$$

- $\alpha\gamma$ configuration

$$V' = \int_{x=\max_2-d}^{\max_1} f(x)\int_{y=\min_2}^{\max_2} g(y)dydx + \int_{x=\min_2-d}^{\max_2-d} f(x)\int_{y=\min_2}^{x+d} g(y)dydx$$

- $\beta\beta$ configuration

$$V' = \int_{x=\min_1}^{\max_1} f(x)\int_{y=\min_2}^{x+d} g(y)dydx$$

- $\beta\gamma$ configuration

$$V' = \int_{x=\max_2-d}^{\max_1} f(x)\int_{y=\min_2}^{\max_2} g(y)dydx + \int_{x=\min_1}^{\max_2-d} f(x)\int_{y=\min_2}^{x+d} g(y)dydx$$

Equation (32) then follows.

□

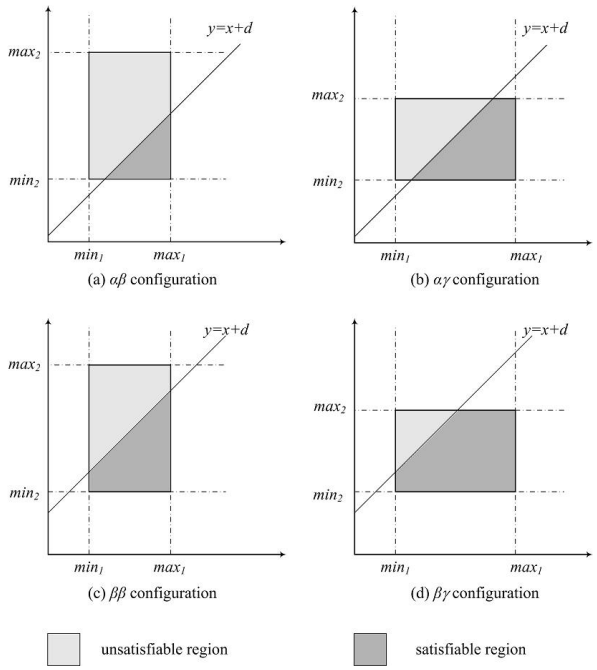



**Figure 5. Four non-trivial configurations.**

## 6. Interval-Based Timing Constraints in Distributed Voting Problem

Now, we modify our problem in Section 4 by relaxing point-based timing constraints to interval-based timing constraints. Assume that the external requests come at time $t_0 = 0$, the individual deadline requirements for the two data are $\Delta_1$ and $\Delta_2$, respectively. The required maximum time span of the two replies is $\Delta$. We further assume that the individual deadline (data availability) is more important than the relative span requirement (data consistency). In the interval-based timing constraint framework, we can assign the confidence thresholds of individual deadlines to be 65% and the confidence threshold of relative span to be 30% — the more important the constraint, the higher the confidence threshold. The confidence level, 65%, 30% are chosen for illustration purpose. They are application dependent. According to the data availability and consistency constraints shown in Section 4, we may have the following timing constraints:

$$\begin{cases} [t_0,t_0]+\Delta_1 \geq [t_0,t_1] & with \quad 65\% \\ [t_0,t_0]+\Delta_2 \geq [t_0,t_2] & with \quad 65\% \\ \left|[t_0,t_1]-[t_0,t_2]\right| \leq \Delta & with \quad 30\% \end{cases} \tag{33}$$

where $[t_0,t_0]$ is the timestamp of the request, $[t_0, t_1]$ and $[t_0, t_2]$ are the timestamps of the replies from group *IR* and group *RW*, respectively. From (5), we know that the cumulative distribution functions of getting valid data under *IR* and *RW* are:

$$P_1(x) = 1 - e^{-|IR|\lambda_1 x} \tag{34}$$

and

$$P_2(y) = \sum_{i=\lceil (|RW|+1)/2 \rceil}^{|RW|} \binom{|RW|}{i} (1-e^{-\lambda_2 y})^i e^{-(|RW|-i)\lambda_2 y} \tag{35}$$

respectively. Thus, the probability density functions over the two intervals are:

$$f(x) = \frac{\mathrm{d}P_1(x)}{\mathrm{d}x} \; and \; g(y) = \frac{\mathrm{d}P_2(y)}{\mathrm{d}y} \tag{36}$$

respectively. The curves of $f(x)$ and $g(y)$ are shown in Figure 6(a) and (b).

Since we use exponential distributions in this example, we let $t_1$ and $t_2$ to be arbitrarily large to guarantee that valid votes can be obtained within the interval of the timestamp. To facilitate discussion, we further assume the following parameters: $\lambda_1 = 1$, $\lambda_2 = 1$, $\Delta_1 = 0.35$, $\Delta_2 = 0.85$, and $\Delta = 0.3$. Under this set of parameters, (29) fails to hold (because $\ln 2/\lambda_2 > \Delta+\Delta_1$), which means that the set of point-based timing constraints is not feasible. However, as can be seen, the set of interval-based timing constraints in (33) is feasible under certain resource allocations.

### 6.1. Data Availability (Individual Deadline) Constraints

From (34) and (35), we know that the probabilities of getting valid data before $\Delta_1$ under *IR* and $\Delta_2$ under *RW* are $P_1(\Delta_1)$ and $P_2(\Delta_2)$, respectively. Under the parameter settings, making $|IR| = 3$ and $|RW| = 7$ will satisfy the first two constraints in (33), with satisfaction probability $P_1(\Delta_1) = 65.01\%$ and $P_2(\Delta_2) = 65.55\%$, respectively.

### 6.2. Data Consistency (Relative Time Span) Constraint

Convert the constraint $|[t_0, t_1]-[t_0, t_2]| \leq \Delta$ *with* 30% in (33)into the form of (31):

$$-\Delta \leq [t_0,t_2]-[t_0,t_1] \leq \Delta \quad with \quad 30\% \tag{37}$$

We now consider the joint density function of $f(x)$ and $g(y)$, as shown in Figure 6(c) and (d). The satisfiable region, which is shaded in Figure 6(c), is the region between $y \leq x+\Delta$ and $y \geq x-\Delta$. Let $SP|_{c^+}(x+\Delta)$ and $SP|_{c^+}(x-\Delta)$ denote the satisfaction probabilities of the constraints $[t_0, t_2] - [t_0, t_1] \leq \Delta$ and $[t_0, t_2] - [t_0, t_1] \leq -\Delta$, respectively. Thus, we have that the satisfaction probability of the constraint in (37) is

$$SP\big|_{c^+}=SP\big|_{c^+}(x+\Delta)-SP\big|_{c^+}(x-\Delta) \tag{38}$$

Substitute $f(x)$ and $g(y)$ in (32) with $dP_1(x)/dx$ and $dP_2(y)/dy$, respectively, and substitute corresponding bounds, we have that

$$SP\big|_{c^+}(x+\Delta)=\frac{\int_{x=MAX(0,-\Delta)}^{t_1}\frac{dP_1(x)}{dx}\int_{y=0}^{MIN(x+\Delta,t_2)}\frac{dP_2(y)}{dy}dydx}{P_1(t_1)\cdot P_2(t_2)}$$

$$=\frac{\int_{x=MAX(0,-\Delta)}^{t_1}P_2(MIN(x+\Delta,t_2))dP_1(x)}{P_1(t_1)\cdot P_2(t_2)} \tag{39}$$

Since $t_1$ and $t_2$ are large enough to guarantee valid data deliveries (with probabilities $P_1(t_1)\to 1$, $P_2(t_2)\to 1$), the satisfaction probability of can be simplified to

$$SP\big|_{c^+}(x+\Delta)=\int_{x=0}^{\infty}P_2(x+\Delta)dP_1(x) \tag{40}$$

Similarly, we have

$$SP\big|_{c^+}(x-\Delta)=\int_{x=\Delta}^{\infty}P_2(x-\Delta)dP_1(x) \tag{41}$$

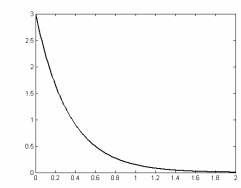
Therefore, the satisfaction probability of (37) is

$$SP\big|_{c^+}=\int_{x=0}^{\infty}P_2(x+\Delta)dP_1(x)-\int_{x=\Delta}^{\infty}P_2(x-\Delta)dP_1(x) \tag{42}$$

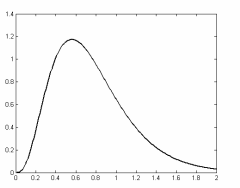
From (34), (35), and (42), we can calculate the value of the satisfaction probability of the data consistency (relative time span) constraint (37). Using numerical integration, under the parameter settings, we have $SP\big|_{c^+}(x+\Delta)$ = 38.85%, $SP\big|_{c^+}(x-\Delta)$ = 6.78%, and $SP\big|_{c^+}$ = 32.07%, which satisfy the specified confidence threshold.

From the comparison between the examples presented in this section and Section 4, we can conclude that:

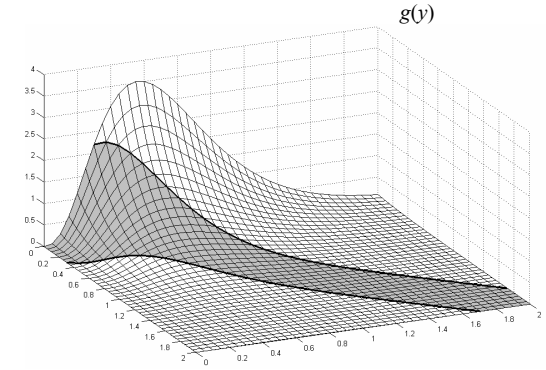
- Interval-based timing constraints are more realistic than point-based timing constraints as the precise time estimation of software execution time in embedded networked system is impossible [15]. Instead, what we may know is a statistical time within a range.
- When a confidence threshold is associated to a corresponding constraint, the priority of the constraint is also explicitly specified by the threshold.
- While certain point-based timing constraints may be intrinsically infeasible to satisfy, relaxing timing constraints from point-based to interval-based, we are able to find feasible solutions with appropriate resource allocations.



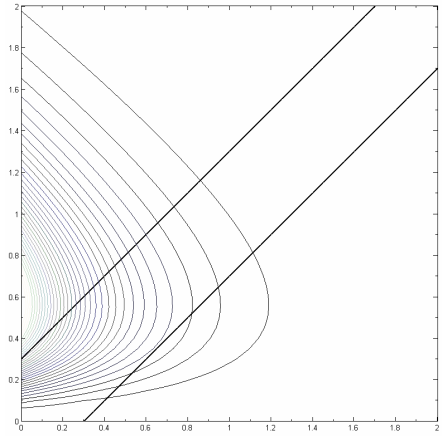
(a) *IR* group with probability density function $f(x)$



(b) *RW* group with probability density function $g(y)$



(c) Joint density function of $f(x)$ and $g(y)$. The bold lines represent the intersections between the joint density function and the planes $y = x+\Delta$ and $y = x-\Delta$, respectively. The shaded region which lies between the two planes represents the satisfiable region.



(d) Contour map of $f(x)$ $g(y)$. The bold lines are projections of the planes $y = x+\Delta$ and $y = x-\Delta$ on to the X-Y plane. The satisfiable region lies between the two lines.

**Figure 6. Individual and joint probability density functions of the two groups.**

## 7. Related Work

It is important for a fault-tolerant distributed computing system to reach agreement on data values from non-faulty processes in the presence of faulty ones. Therefore, voting is widely used in consistency and agreement algorithms in distributed systems [23]. Many voting protocols have been studied elsewhere by the research community under various application

settings and environments [6, 10, 25]. [20] gives a very good summary of various issues in voting. The four main components of a voting algorithm, namely input data, output data, input votes, and output votes, can be used to impose a binary 4-cube classification scheme, leading to 16 classes [19]. Although we only consider the expected decision time of the exact consensus threshold voting, our methodology can be applied to other voting classes.

One of the most important performance parameters in evaluating voting schemes is latency. Latency is defined as the length of the time interval between the availability of the last input and the production of the voter output. In most cases, the dominant factor of the latency of a voting algorithm is not the computational part of the algorithm but rather the multiple rounds of communication [5, 11]. [22] discusses the possibility to strike a balance between the overhead of tight synchronization and the algorithmic complexity of fully asynchronous operation via an intermediate approach.

The idea that diagnostic decisions in dynamic environments often require trade-offs between decision accuracy and timeliness comes from [7]. Thus, the time required to obtain a correct vote in a distributed system not only depends on the communication latencies but also the time-dependent accuracy. The vote accuracy is actually reflected in this paper by the credibility function monotonically increasing with time. That is, with more time, we would get more trustworthy and accurate data.

As for the timing constraints parts, Chodrow et al. [2] presents a constraint-graph-based algorithm for detecting violations of timing constraints. To check the satisfiability for a set of constraints, a constraint graph is instantiated from the current event histories with an all-pair shortest path algorithm run on the instantiated graph. A negative cycle indicates unsatisfiability of the constraint set.

In [21] and [9], the authors extend the timing constraint specification and violation detection algorithm to distributed real-time systems. They indicate that the derivation of implicit constraints is essential for catching timing violations at an earlier time since it is possible that an implicit constraint is violated before an explicit delay or deadline becomes unsatisfiable at run-time. They also prove that for constraint violation detection, the problem of minimizing the amount of information to be exchanged between processors is NP-hard.

In [17], Mok and Liu provide a more expressive specification language based on Real Time Logic to define timing constraints. To reduce time complexity of the event monitoring algorithm at run-time, they resolve most of the shortest path information of the instantiated constraint graph from the uninstantiated constraint graph at compilation time. Thus, only small modifications of the graph are needed during run-time.

Two new timing constraints based on time intervals: *certain* and *possible* are proposed in [18] to specify the desired degree of certainty whether a timing violation has occurred. The authors indicate that this interval-based timing constraint: $I_1 + d \geq I_2 U$, where $I_1$ and $I_2$ denote the time intervals in which the corresponding events may occur, is satisfied either possibly ($P$) or certainly ($C$). The authors further extend the monitoring algorithm in [17] to monitor interval-based timing constraints with probabilities.

Lee et al. [14] propose interval-based timing constraints with a confidence threshold model. The concept of *Earliest Expiration Time* (EET) is also introduced. The knowledge of EET enables the event monitor to announce the violation of timing constraints even before the actual deadlines.

Yu et al. [27] extend Lee's work by considering more general cases of the interval-based timing constraint model where events are exponentially or normally distributed. They use the results from [27] to study the timing constraints between a pair of faults in a distributed embedded system in [26].

Examples of other stochastic approaches in real-time applications can be found in [1, 3, 4, 8, 16, 24].

## 8. Conclusion

In this paper, we have studied the expected decision times under two different voting schemes. We assume that the latency for obtaining valid data depends not only on the time that a sensor gives a vote, but also on the time-dependent accuracy of the vote. Assuming all voters are truthful, we show that increasing the number of resources reduces the expected time of obtaining assured votes. Our analytic study has also shown that in an environment where not all voters are truthful, adding homogeneous resources may increase the trustworthiness of the voting, but it offers little improvement on the expected delivery time of the voting.

In addition, we have demonstrated that the timing information of groups of homogenous sensors can be used to balance the trade-off between data consistency constraints and data availability requirements in a heterogonous environment.

There are situations where point-based timing constraints are not only difficult to accurately specified,

but also not satisfiable. In contrast, interval-based timing constraint not only provides a more general format for timing constraints and is more realistic, but also it allows us to obtain statistical solutions when guaranteed solutions are not available.

## 9. References


[1] L. Abeni and G. Buttazzo, "Qos guarantee using probabilistic deadlines," in *Proc. of the 11th Euromicro Conference on Real-Time Systems*, 1999, pp. 242–249.

[2] S. Chodrow, F. Jahanian, and M. Donner, "Run-time monitoring of real time systems," in *Proc. of the 12th IEEE Real-Time Systems Symposium*, 1991, pp. 74–83.

[3] L. David and I. Puaut, "Static determination of probabilistic execution times," in *Proc. of the 16th Euromicro Conference on Real-Time Systems*, 2004, pp. 223–230.

[4] J. L. D´ıaz, D. F. Garc´ıa, K. Kim, C.-G. Lee, L. L. Bello, J. M. L´opez, S. L. Min, and O. Mirabella, "Stochastic analysis of periodic real-time systems," in *Proc. of the 23rd IEEE Real-Time Systems Symposium*, 2002, pp. 289–300.

[5] D. Dolev, N. A. Lynch, S. S. Pinter, and E. W. Stark, "Reaching approximate agreement in the presence of faults," In *ACM Journal*, vol. 33, no. 3, pp. 499-516, July 1986.

[6] S. Hariri, A. Choudhary, and B. Sarikaya, "Architectural support for designing fault-tolerant open distributed systems," In *IEEE Computer*, pp. 50-62, June 1992.

[7] M. Hildebrandt and J. Meyer, "When to act? Managing time-accuracy trade-offs in a dynamic belief updating task," In *Proc. the 49th Annual Meeting of the Human Factors and Ergonomics Society*, 2005.

[8] X. S. Hu, T. Zhou, and E. H.-M. Sha, "Estimating probabilistic timing performance for real-time embedded systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 833–853, 2001.

[9] F. Jahanian, R. Rajkumar, and S. Raju, "Run-time monitoring of timing constraints in distributed real-time systems," University of Michigan, Technical Report CSE-TR 212–94, 1994.

[10] P. Jalote and et al, "Atomic actions on decentralized data," Chap. 6, *Fault-tolerant Systems*, John-Wiley Publ. Co., 1995.

[11] R. M. Kieckhafer and M. H. Azadmanesh, "Reaching approximate agreement with mixed-mode faults," In *IEEE Trans. Parallel and Distributed Systems*, vol. 5, no. 1, pp. 53-63, January 1994.

[12] H. Kopetz and P. Verissmo, "Real time dependability concepts," Chap. 16, *Distributed Systems*, S. Mullender, Addison-Wesl. Co., 1993.

[13] K. A. Kwiat, K. Ravindran, and P. Hurley, "Energy-efficient replica voting mechanisms for secure real-time embedded systems," In *Proc. of the 6th IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*, 2005.

[14] C.-G. Lee, A. Mok, and P. Konana, "Monitoring of timing constraints with confidence threshold requirements," in *Proc. of the 24th IEEE Real Time Systems Symposium*, 2003, pp. 178–187.

[15] E. A. Lee, "Building unreliable systems out of reliable components: the real time story," *Abstract of Invited Plenary Talk, Monterey Workshop, Laguna Beach, California*, September 22, 2005.

[16] S. Manolache, P. Eles, and Z. Peng, "Optimization of soft real-time systems with deadline miss ratio constraints," in *Proc. of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2004, pp. 562–570.

[17] A. K. Mok and G. Liu, "Efficient run-time monitoring of timing constraint," in *Proc. of the 3rd IEEE Real Time Technology and Applications Symposium*, 1997, pp. 252–262.

[18] A. K. Mok, C.-G. Lee, H. Woo, and P. Konana, "The monitoring of timing constraints on time intervals," in *Proc. of the 23rd IEEE Real Time Systems Symposium*, 2002, pp. 191–200.

[19] B. Parhami, "A taxonomy of voting schemes for data fusion and dependable computation," In *Reliability Engineering and System Safety*, vol. 52, no. 2, pp. 139-151, May 1996.

[20] B. Parhami, "Voting: a paradigm for adjudication and data fusion in dependable systems," Chap. 4, *Dependable Computing Systems*, edited by H. B. Diab and A. Y. Zomaya, John-Wiley Publ. Co., 2005.

[21] S. Raju and R. Rajkumar, "Monitoring timing constraints in distributed real time systems," in *Proc. of the 13th IEEE Real-Time Systems Symposium*, 1992, pp. 57–67.

[22] K. G. Shin and J. W. Dolter, "Alternative majority-voting methods for real-time computing systems," In *IEEE Trans. Reliability*, vol. 38, no. 1, pp. 58-64, April 1989.

[23] M. Spasojevic and P. Berman, "Voting as the optimal static pessimistic scheme for managing replicated data," In *IEEE Trans. Computers*, vol. 24, no. 5, pp. 525-533, May 1975.

[24] S. Wang, J. R. Merrick, and K. G. Shin, "Component allocation with multiple resource constraints for large embedded real-time software design," in *Proc. of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2004, pp. 219–226.

[25] H. Y. Youn, J. Y. Lee, and A. D. Singh, "Adaptive unanimous voting scheme for distributed self-diagnosis," In *IEEE Trans. Computers,* pp. 730-735, 1995.

[26] Y. Yu, S. Ren, and O. Frieder, "Prediction of timing constraint violation for real-time embedded systems with known hardware failure model," In *Proc. of the 27th IEEE Real-Time Systems Symposium*, 2006.

[27] Y. Yu, W. Guan, S. Ren, O. Frieder, "Satisfaction Probabilities of Interval-based Timing Constraints" (Submitted to IEEE Transactions on Computers)