Expected Time for Obtaining Dependable Data in Real-Time Environment

Yue Yu[1], Shangping Ren[1,2]
Department of Computer Science
Illinois Institute of Technology
{ yyu8, ren}@iit.edu

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 environments, 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 voters make their votes, but also on the accuracy and credibility of the votes. Thus, a new metric, i.e. credibility function, need be taken into account in evaluating expected time and deciding replications. This paper presents analytical solutions for expected time under different voting schemes when dependable data can be obtained. We also show that if not all voters are truthful, adding more replications does not improve much on the time of obtaining valid results.

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. Different from other methods used in achieving fault tolerance, such as stand-by sparring, we treat values from different replicas as votes to tabulate the consensus. Therefore, we refer to individual replicas as *voters*. Real-time data usually has a lifespan associated with it [10]. 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 delivering it to the data client must not take longer than the lifespan of the data.

However, as contended by Dr. Lee [12] 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 reach a consensus. Most voting schemes use a deadline to mark the end of the data's lifespan [10]. 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) the expected values of decision time and, 2) how resource availability may impact the time a decision is made.

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 data safety and availability in

[1] Supported in part by NSF under grant CNS 0431832.
[2] Supported in part by Air Force Summer Faculty Fellowship.

1-4244-0910-1/07/$20.00 ©2007 IEEE.

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. Related work is discussed in section 5. Section 6 summarizes our conclusions and future work.

2. Background

In embedded systems, data sensed from the environment may have a timeliness parameter. The timeliness pertains to how soon 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 [7].

Consider an example presented in [10], 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 $\Delta(d)$, even 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 [12], a precise timing estimation of software execution time in embedded networked system is impossible. Instead, what we may know is a statistical time within a range. For instance, upon an enemy plane has emerged in the region at time, 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, exactly *when* 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. Further observation is that under non-faulty circumstance, if data are only statistically certain, increasing the number of replicas 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 voters 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 voter provides correct data and $C_i(t) = 1$. In this case, once the decision unit gets a datum D_i from any voter, it can deliver D_i to the user without waiting for data from other voters.

- **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 voter 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 voter i by time t is given by $V_i(t)$.

Given the variables above, our goal is to estimate the expected time for the decision unit to get truthful data from the voters.

To formulate the problem, let X_i be the random variable representing if the decision unit get a vote from the ith voter

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

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 inherent correct data D. Let Y_i be the random variable representing whether the data D_i agrees with D, that is

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

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 ith voter is

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

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 ith 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 voters 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 equation (5) is the probability that at least k similar votes are collected before time t. Let random variable T represent the time point at which enough similar votes (at least k) are collected, i.e., the decision time, we have,

$$\begin{aligned} &P\{T \leq t\} = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} \\ &\textit{and} \\ &P\{T > t\} = 1 - \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} = \sum_{i=0}^{k-1} \binom{n}{i} p^i (1-p)^{n-i} \end{aligned} \tag{6}$$

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 voters 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). Moreover, to get the probability distribution function $V(t)$ for voting time, we consider a situation in which the data coming from the voters are at constant rate (λ) 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:

$$V(t) = 1 - e^{-\lambda t}, \quad t \geq 0 \tag{8}$$

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

$$E[T] = \int_0^{\infty} e^{-n\lambda t} dt = \frac{1}{\lambda} \cdot \frac{1}{n} \tag{9}$$

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.

Similarly, if we assume that $V(t)$ is uniformly distributed over the interval [0, T_1], i.e.,

$$V(t) = \begin{cases} \dfrac{t}{T_1}, & \textit{if } t \in (0, T_1) \\ 1, & \textit{otherwise} \end{cases} \tag{10}$$

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

$$E[T]=\int_{0}^{T_1}\left(1-\frac{t}{T_1}\right)^{n}dt+\int_{T_1}^{\infty}\left(1-1\right)^{n}dt=\frac{1}{n+1}T_1 \qquad (11)$$

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

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 voters is

$$\begin{aligned}P\{\sum_{i=1}^{n}(X_i \wedge Y_i)\geq 1\} &= 1-P\{\sum_{i=1}^{n}(X_i \wedge Y_i)=0\}\\ &=1-\prod_{i=1}^{n}q_i=1-\prod_{i=1}^{n}(1-C_i(t)V_i(t))\end{aligned} \qquad (12)$$

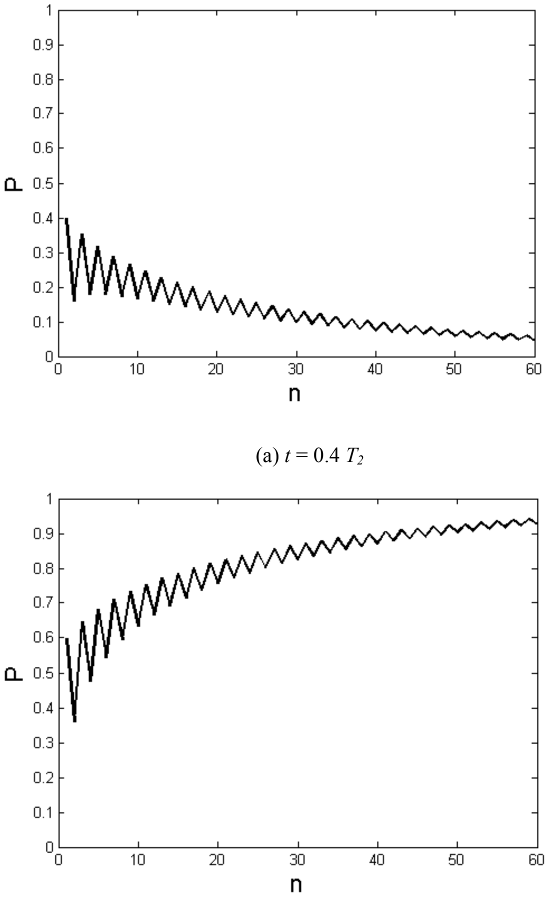
in which $\prod q_i$ characterizes a parallel system. In such a system, voters work in a "co-operative" way. Therefore, adding resources (more homogenous voters) to the subsystem improves its performance and thus reduces the expected decision time.

3.2. Untruthful Voters

Under untruthful voter scenario, we have k determined by the specific majority voting protocol (where $k = \lceil (n+1)/2 \rceil$ in majority voting protocols and $k = \lceil 2n/3 \rceil$ in the more stringent Byzantine voting protocols, and we use the former in the following discussions). We further assume that $C(t)$ is uniformly distributed over the interval $[0, T_2]$ and $V(t) = 1$[3]. From (5), we can derive the probability of getting a valid data before time t:

$$P\{T\leq t\}=\sum_{i=k}^{n}\binom{n}{i}\left(\frac{t}{T_2}\right)^{i}\left(1-\frac{t}{T_2}\right)^{n-i} \qquad \left(t\in[0,T_2]\right) \qquad (13)$$

The following figures show the relationships between $P\{T \leq t\}$ and n under different t:



(a) $t = 0.4\ T_2$

(b) $t = 0.6\ T_2$

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

As can be seen that when $t = 0.4T_2$, which means that $C(t)V(t)$, i.e., 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?

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

[3] Although it is unreasonable to assume $V(t) = 1$, i.e., a voter 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.

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

Make the substitution $x = t/T_2 \Rightarrow \mathrm{d}x = (1/T_2)\mathrm{d}t$ in (14), we have,

$$
E[T]=\sum_{i=0}^{k-1}\binom{n}{i}\int_0^1 x^i(1-x)^{n-i}\,T_2dx \tag{15}
$$

Integrate by parts, we have,

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

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,

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

Given that $k = \lceil (n+1)/2 \rceil$ and n is large, we have that $E[T]=T_2/2$

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

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

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

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

Using equation (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:

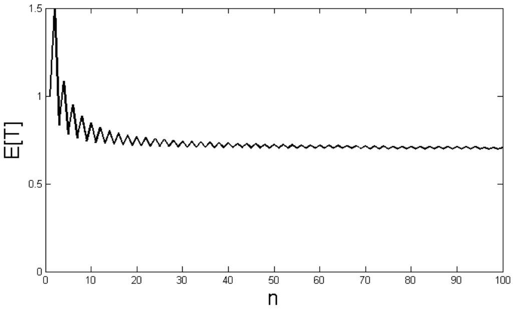


Figure 2. Expected Decision Time with λ=1

As can be seen, when the number of working voters are small, increasing the number of voters 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 $\ln2/\lambda$ and no further decrease can be achieved by adding more resources. For example, with 11 voters, the expected decision time is $0.7365/\lambda$, while with 23 voters, the expected decision time is $0.7144/\lambda$ — a 3.0% time gain is at the cost of more than twice the resources.

4. Data Availability and Its Consistency Constraints

In heterogonous environments, data comes from different sources. However, they need to be coherent if such data are different views of the same object. 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 Δ, 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 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 Δ, we have the following timing constraints:

$$\begin{cases} t_1 - t_0 = \dfrac{1}{\lambda_1 |IR|} \pm d_1 \le \Delta_1 \\ t_2 - t_0 = \dfrac{\ln 2}{\lambda_2} \pm d_2 \le \Delta_2 \\ |t_1 - t_2| \le \Delta \end{cases} \tag{26}$$

where the first two equations come from (9) and (24), respectively, in which $|IR|$ and $|RW|$ are the number of sensors under group *IR* and *RW*; d_1 and d_2 are the deviations (e.g., the standard deviations or the maximum possible deviations) from the expected decision times[4]; Δ_1 and Δ_2 are the individual deadline requirements for the two data, respectively, and Δ is the required maximum time span of the two different types of replies. We further convert the constraints into the form $t_j - t_i \le d$ and construct the corresponding constraint graph G = (*V*, *E*) as shown in Figure 3, where

$$E = \{(v_i, v_j) \mid t_j - t_i \le d_k \in S\} \; and \; w(v_i, v_j) = d_k \tag{27}$$

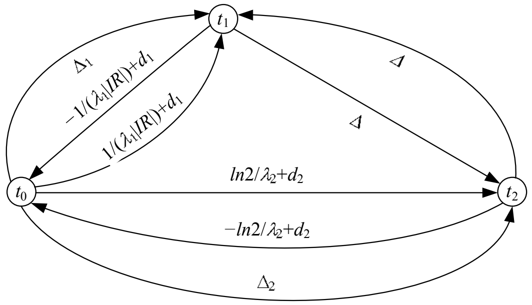


Figure 3. Timing Constraint Graph

With such a constraint graph, the Bellman-Ford algorithm can be used to detect if the graph has negative-weight cycles. The existence of negative-weight cycles in the constraint graph indicates that the constraints are unsatisfiable [15]. For instance, the cycle $\overrightarrow{t_0t_1}(1/(\lambda_1 \mid IR \mid) + d_1)$, $\overrightarrow{t_1t_2}(\Delta)$, $\overrightarrow{t_2t_0}(-\ln 2/\lambda_2 + d_2)$ is negative if

$$\frac{1}{\lambda_1 \mid IR \mid} + d_1 + \Delta - \frac{\ln 2}{\lambda_2} + d_2 < 0 \tag{28}$$

which indicates that the time-consistency constraint between the two types of 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 (such as reduce 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 $\overrightarrow{t_0t_1}(\Delta_1)$, $\overrightarrow{t_1t_0}(-1/(\lambda_1 \mid IR \mid) + d_1)$ if

$$-\frac{1}{\lambda_1 \mid IR \mid} + d_1 + \Delta_1 < 0 \tag{29}$$

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 (28) and (29) need to be taken into account:

[4] 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.

$$\begin{cases} \dfrac{1}{\lambda_1 |IR|} + d_1 + \Delta - \dfrac{\ln 2}{\lambda_2} + d_2 \geq 0 \\ -\dfrac{1}{\lambda_1 |IR|} + d_1 + \Delta_1 \geq 0 \end{cases} \qquad (30)$$

$$\Rightarrow |IR| \in \left[\frac{1}{\lambda_1(\Delta_1 + d_1)}, \frac{\lambda_2}{\lambda_1} \frac{1}{\ln 2 - \lambda_2(\Delta + d_1 + d_2)} \right]$$

Therefore, since the satisfaction of data availability and consistency constraints depends solely on the selection of parameter |*IR*|, i.e., the number of sensors in group *IR*, (30) gives the tunable range of |*IR*| where a feasible resource allocation strategy exists.

5. 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 [18] Many voting protocols have been studied elsewhere by the research community under various application settings and environments [3, 5, 19]. [14] gives a profound 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 [13]. 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 [2, 6]. [17] 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 [4]. 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.

6. Conclusion

In this paper, we study the expected decision times under two different voting schemes. We assume that the latency for obtaining a valid data depends not only on the time that a voter gives a vote, but also on the time-dependent accuracy of the vote. Assuming all voters are truthful, we show how the number of replicas and their voting probability affect the data safety and availability. Moreover, we show that in an environment where not all voters are truthful, adding homogeneous resources may increase the trustworthy of voting results, but it does not improve much on the average time of getting valid voting results. We also study how to use timing information of groups of homogenous sensors to deal with the trade-off between data consistency constraints and data availability requirements in a heterogonous environment.

Our future work targets the following directions:

- Apply the methodologies presented in this paper to more complicated voting schemes, such as EDEC(explicit dissent explicit consent) and ICED(implicit consent explicit dissent) voting protocols [16] and algorithms for collaborative target detection in a sensor network that are efficient in terms of communication cost, precision, accuracy, and number of faulty sensors tolerable [1].
- To make the ideas in this paper more intuitive, data credibilities are modeled by independent uniform/exponential random variables. In real scenarios, however, more complicated random variables are used to model faults, such as in sensor networks [8, 9]. With (7), we can derive expected decision time for practical fault models. Some experiments will also be conducted to assess the theoretical results.
- Note that in Section 4, Eq. (30) could be intrinsically infeasible when

$$\frac{1}{\lambda_1(\Delta_1 + d_1)} > \frac{\lambda_2}{\lambda_1} \frac{1}{\ln 2 - \lambda_2(\Delta + d_1 + d_2)}$$
$$\Rightarrow \frac{\ln 2}{\lambda_2} > \Delta + \Delta_1 + 2d_1 + d_2$$
$$\Rightarrow \frac{\ln 2}{\lambda_2} > \Delta + \Delta_1 + d_2 \qquad (31)$$

Equation (31) 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 point-based timing constraints as in (26) by adopting the *interval-based timing constraints* [11, 20]. With interval-based timing constraints, we can also explicitly specify priorities of different constraints so that in case not all constraints can be satisfied, constraints with higher priorities are satisfied with higher probability.

Moreover, in [20], the authors consider two event occurrences to be monitored. If one considers a distributed system, monitoring event pairs is not sufficient. However, if more events are to be considered, the complexity of the described approach would increase in magnitude. The results in this paper show how to give group statistical information from that of individual components. Thus, monitoring interval-based timing constraints in a distributed system may be decomposed into two phases to reduce complexity: (1) group homogenous components and derive group statistical information; (2) study the interval-based timing constraints with the group statistical information.

References

[1] T. Clouqueur, K. K. Saluja, and P. Ramanathan. Fault tolerance in collaborative sensor networks for target detection. In *IEEE Trans. Computers*, vol. 53, no. 3, pp. 320-333, March 2004.

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

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

[4] 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.

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

[6] 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.

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

[8] F. Koushanfar, M. Potkonjak, A. Sangiovanni-Vincentelli. On-line Fault Detection of Sensor Measurements. In *Proceedings of IEEE Sensors*, 2003.

[9] B. Krishnamachari, S. Iyengar. Distributed Bayesian Algorithms for Fault-Tolerant Event Region Detection in Wireless Sensor Networks. In *IEEE Trans. Computers*, vol. 53, no. 3, pp. 241-250, March 2004.

[10] 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.

[11] 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, pp. 178-187, 2003.

[12] 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.

[13] 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.

[14] 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.

[15] S. C. V. Raju, R. Rajkumar, and F. Jahanian. Monitoring timing constraints in distributed real-time systems. In *Proc. of the 13th IEEE Real-Time System Symposium, RTSS*, pp 57-67, Phoenix, AZ, Dec. 1992.

[16] K. Ravindran, K. A. Kwiat, and A. Sabbir. Adapting distributed voting algorithms for secure real-time embedded systems. In *Proc. the 24th International Conference on Distributed Computing Systems Workshops*, 2004

[17] 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.

[18] 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.

[19] 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.

[20] 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.