IPSJ Journal Vol. 49 No. 6 2092–2105 (June 2008)

Regular Paper

Analysis of Probabilistic Trapezoid Protocol for Data Replication

Tabito Suzuki,[†1] Mamoru Ohara,[†1] Masayuki Arai,[†1]
Satoshi Fukumoto[†1] and Kazuhiko Iwasaki[†1]

Maintaining replicated data between nodes can improve the dependability of data. We propose a probabilistic trapezoid protocol for replicated data that combines the trapezoid protocol with the concept of a probabilistic quorum system. We analyzed the read availability, the latest version's read availability and the average number of nodes accessed for the protocol. Our numerical evaluations demonstrated that it improves not only the read availability but also the latest version's read availability. Furthermore, when the number of nodes is greater than 100, it could effectively reduce the system's load. We designed and implemented a file transfer protocol to replicate data. Experimental results proved that the trapezoid protocol could achieve a better throughput than the voting system or the grid protocol. Despite node failure, the probabilistic trapezoid protocol also achieved a relatively better throughput.

1. Introduction

With the advancement of computer networks, techniques to support the dependability of networking and distributed data have become more and more important [1),2)]. Data replication techniques are promising to improve data dependability. A cluster system consisting of many inexpensive PCs can be extremely dependable and have an outstanding performance [4)]. The consistency of replicated data in the distributed nodes must be maintained. That is, we must consider the cost of maintaining consistent data in order to improve the dependability [5)].

One data replication technique is the Read One Write All (ROWA), where data are read from one node and written to all nodes [3)]. The main advantage of this technique is its low read overhead. However, the disadvantage is that the write overhead is proportional to the number of nodes in the system. In addition, if a node is not available, data cannot be updated, resulting in low reliability on write operations.

The Quorum theory was proposed to overcome these difficulties [6),7)]. The data in a quorum system (refferred to as QS in the following), are read from a set of nodes called a quorum and written to it. The read and write operations can be mutually excluded, by applying read and/or write lock techniques. The QS, however, has to access $O(N)$ nodes, resulting in high read/write operation overheads, where N is the number of nodes in the system. Peleg, et al. evaluated the probe complexity, that is, the number of nodes accessed in the worst case, of QS [8)].

A few data replication protocols have been proposed that can reduce the system's load by utilizing a logical structure for the nodes. A node with the grid protocol was arranged on a logical mesh to reduce the number of nodes accessed [9)]. The tree quorum protocol applies a logical tree structure [10),11)]. Jimenez-Peris, et al. evaluated the effectiveness of some types of quorum systems, including the grid and the tree quorum protocol [12)]. Youn, et al. proposed a hybrid data replication protocol that combines the concepts of the grid protocol and the tree quorum protocol [13)]. With this protocol, it was possible to read/write data only by accessing a small number of nodes when node failures did not occur. The trapezoid protocol (referred to as TP in the following) has also been proposed [14)]. The TP has a higher level of data availability and it also has a throughput better than that of the quorum system or the grid protocol.

Another approach to improve the data availability has been presented, which proposed a probabilistic quorum system (probabilistic QS) [15)]. Not only can this technique reduce the system's load, it also increases the data availability. However, it could not always ensure the consistency of data. The probabilistic QS is considered useful in systems where the latest data is not always required, but a high level of data availability is Refs. 16), 17). The write quorum and the read quorum do not necessarily intersect for the probabilistic QS.

We propose a probabilistic TP that combines the TP with the concept of the probabilistic QS. By relaxing the intersection requirement on the TP, the technique we propose is able to provide a better read availability and to reduce the

†1 Tokyo Metropolitan University

average number of the nodes accessed. We can prevent write conflicts by applying the write lock technique. We theoretically analyzed the read availability, the latest version read availability (LV read availability), and the expected number of nodes accessed. We also did experiments and obtained results for various replication protocols.

This paper is organized as follows. Section 2 briefly reviews replication protocols. We then propose the probabilistic TP in Section 3. The theoretical analysis of the availabilities and the number of accessed nodes is shown in Section 4. The experimental results follow in Section 5, and Section 6 concludes the paper.

2. Overview of Data Replication Protocols

2.1 Definitions

A read quorum (RQ) denotes a set of nodes from which data are read. $|RQ|$ denotes the size of the RQ. A write quorum (WQ) denotes a set of nodes in which data are written. $|WQ|$ denotes the size of the WQ. In a QS, for an RQ and a WQ, at least one common node is included in the RQ and the WQ [6),7)]. That is,

$$RQ \cap WQ \neq \emptyset. \quad (1)$$

Also, for two WQs, WQ_1 and WQ_2,

$$WQ_1 \cap WQ_2 \neq \emptyset \quad (2)$$

holds to inhibit generating multiple latest versions. For a read operation, the RQ is locked. Similarly, the WQ is locked for a write operation. Consequently, access conflicts can be prevented.

The probabilistic QS does not guarantee that the obtained data is the latest version [15)]. That is, the system might return old data even when it is regarded as available for a read operation. Therefore, distinguishing the read availability for the latest version (LV) of data from the one for any version, we define the availability for read/write operations as follows.

[Definition 1]

(1) Read availability is defined as the probability that the data can be read from the system.
(2) LV read availability is defined as the probability that the data can be read and is the latest.
(3) Write availability is defined as the probability that the data can be written into the system.

N denotes the number of nodes in the quorum system and p denotes the probability that a node is available, i.e., the node availability. We assumed an independent fail-stop model for the nodes.

2.2 Voting System

As a simple case of quorum system, we explain the voting system (VS), which is a special case of the weighted voting system where all nodes have a single vote [7)]. Equation (3) holds and ensures that the RQ and the WQ have at least one common node in a voting system to read the latest data. In other words, the RQ and the WQ intersect.

$$|RQ| + |WQ| > N. \quad (3)$$

Moreover the following equation ensures that the latest version is unique because the write operation must lock more than half of the nodes.

$$2|WQ| > N. \quad (4)$$

2.3 Grid Protocol

Each node for a grid protocol is arranged on an $I \times J$ logical mesh structure [9)]. The WQ consists of I nodes in a column as well as $(J-1)$ nodes, which have been selected from all the other columns. The RQ is a set of J nodes, which have been selected from each column. **Figure 1** has a 3×5 grid protocol. An example of RQ is $RQ = \{A_{0,0}, A_{2,1}, A_{1,2}, A_{1,3}, A_{2,4}\}$. An example of WQ is $WQ = \{A_{1,0}, A_{0,1}, A_{1,1}, A_{2,1}, A_{0,2}, A_{2,3}, A_{1,4}\}$.

2.4 Trapezoid Protocol

As we can see in **Fig. 2**, the nodes for the TP are arranged as a logical trapezoid that has a height of $(h+1)$ [14)]. The top level consists of b nodes and the l-th level $(1 \leq l \leq h)$ consists of s_l $(= al + b)$ nodes, where a is a non-negative integer and b is a positive integer. For TP, RQs and WQs are redefined as follows, to ensure the consistency in read/write operations [14)]. Here, the parameter w is defined as the number of nodes to which the data is written at each level except for the top level.

(1) RQ: the majority of nodes residing at the top level, or $(s_l - w + 1)$ nodes residing at the l-th level $(1 \leq l \leq h)$.
(2) WQ: the majority of nodes residing at the top level, and w arbitrarily chosen nodes in each level.

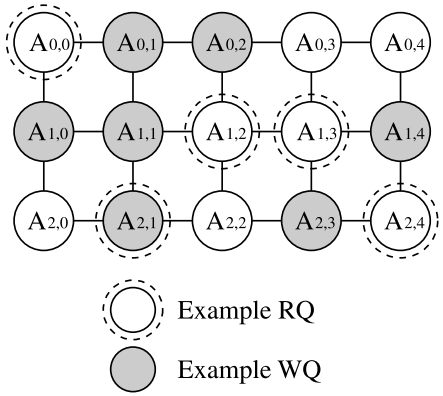


Fig. 1 Example of Grid Protocol with 3×5 nodes. Examples of RQ and WQ are the set of nodes circled with dotted line and painted gray, respectively.

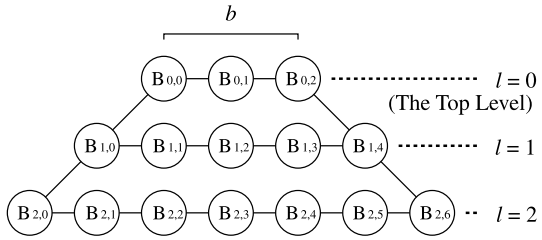


Fig. 2 Example of Trapezoid Protocol ($s_l = 2l + 3$, $a = 2$, $b = 3$, $h = 2$, $N = 15$). Nodes are arranged by logical trapezoid with 3 upper base and 7 lower base nodes.

If more than half of the nodes are available for the top level, the latest data is readable. For the l-th level, if more than $(s_l - w + 1)$ nodes are available, the latest data is readable from the l-th level. In this paper, RQ_l means the quorum for the l-th level.

We adopted a mechanism to balance the load [13),14)]. The parameter f ($0 \leq f \leq 1$, f is real) is introduced to distribute read operations over the logical levels. Using f, the probability that a read request will first be sent to the l-th level, $F(l)$, is derived as

$$F(l) = \begin{cases} (1-f)^l \cdot f, & (l < h) \\ (1-f)^h. & (l = h) \end{cases} \quad (5)$$

A larger f causes a heavier load at the top level, resulting in bottlenecks.

All nodes in at least one WQ have to be available for the write operation to be successful.

Figure 2 has an example of a TP, where $s_l = 2l + 3$, $h = 2$ ($0 \leq l \leq 2$). For $w = 1$, there are 5 RQs ; $RQ_0 = \{B_{0,0}, B_{0,1}\}$, $\{B_{0,0}, B_{0,2}\}$, $\{B_{0,1}, B_{0,2}\}$, $RQ_1 = \{B_{1,0}, B_{1,1}, B_{1,2}, B_{1,3}, B_{1,4}\}$ and $RQ_2 = \{B_{2,0}, B_{2,1}, B_{2,2}, B_{2,3}, B_{2,4}, B_{2,5}, B_{2,6}\}$. An example of WQ is $WQ = \{B_{0,1}, B_{0,2}, B_{1,2}, B_{2,6}\}$. If node $B_{1,3}$ is not available, possible RQs are $RQ_0 = \{B_{0,0}, B_{0,1}\}$, $\{B_{0,0}, B_{0,2}\}$, $\{B_{0,1}, B_{0,2}\}$ and $RQ_2 = \{B_{2,0}, B_{2,1}, B_{2,2}, B_{2,3}, B_{2,4}, B_{2,5}, B_{2,6}\}$.

2.5 Probabilistic Quorum System

For a probabilistic QS, the condition for a quorum is relaxed, i.e., an RQ and a WQ do not necessarily intersect [15)]. As a result, this technique ensures a high degree of read availability in exchange for a slightly lower level of consistency. In the ϵ-intersecting quorum system [15)] which is one of the probabilistic QS, the following equation holds.

$$\Pr(WQ_{selected} \cap RQ_{selected} \neq \emptyset) \geq 1 - \epsilon, \quad (6)$$

where $0 \leq \epsilon \leq 1$, and $WQ_{selected}$ and $RQ_{selected}$ are the quorums selected by a given strategy to write to and read from, respectively. For example, for $N = 100$, $|RQ| = |WQ| = 30$, and the node availability is $p = 0.9$, the ϵ-intersecting quorum system provides better than 0.99999 LV read availability as well as a read availability that is very close to 1.0.

3. Probabilistic Trapezoid Protocol

We propose a probabilistic TP which has a logical trapezoid structure with the probabilistic QS. The proposed protocol attains a higher read availability than the TP in Section 2.4. This technique can also detect write conflicts by applying the write lock technique.

In the probabilistic TP, smaller sets of nodes than those of TP can be used as RQs, and they do not necessarily intersect each other. The parameter γ ($0 \leq \gamma \leq 1$, γ is real) indicates the degree of relaxation. The maximum number

of relaxed nodes for the probabilistic TP, t_l, is expressed as follows for the l-th level:

$$t_l = \begin{cases} 0, & (l = 0) \\ \lfloor s_l \cdot \gamma \rfloor. & (1 \le l \le h) \end{cases} \quad (7)$$

In other words, the size of the quorum for the l-th level ($1 \le l \le h$) is derived as Eq. (8).

$$|RQ_l| \ge s_l - w + 1 - t_l. \quad (8)$$

This kind of relaxation is not applied to the top level. Therefore, if more than half of the nodes in the top level are available, it is said to be readable for the top level. For the l-th level, a set of more than $(s_l - w + 1 - t_l)$ nodes can be used as an RQ. In other words, the l-th level is readable. For $\gamma = 0$, the probabilistic TP is exactly the same as the TP shown in Section 2.4. Hereafter, we call this protocol the TP with $\gamma = 0$.

The read operation for the probabilistic TP is described by the following steps:

Step 1. Select the l-th level according to Eq. (5).
Step 2. Check if the l-th level is readable by applying steps A, B, and C (to be described later).
Step 3. If readable, the data will be read and the read operation will terminate successfully.
Step 4. If not, try the $(l+1)$-th level. At the bottom level, $l = h$, try the top level.

If no levels are readable, the read operation fails. The following describes the procedure for checking whether the l-th level is readable.

Step A. Select a node randomly and test whether it is available.
Step B. If the following conditions hold, the l-th level is determined to be readable:
  case 1. There are $(s_l - w + 1)$ available nodes.
  case 2. There are at least $(s_l - w + 1 - t_l)$ available nodes and all the nodes in the l-th level have been checked.
Step C. The l-th level is not readable: if the number of available nodes found plus the number of unchecked nodes is less than $(s_l - w + 1 - t_l)$.

Figure 3 describes this procedure.

```
Definition:
  U, V: set of nodes ;
  |U|, |V|: size of U, size of V ;
  u: chosen node ;
Initialization:
  U = {u_1, u_2, u_3, ··· , u_{s_l}} from l-th level ;
  V = {∅} ; // available nodes
Checking:
  repeat_forever{
Step A:
    u = selectOneNodeRandomlyFrom(U) ;
    U = U − u ;
    if ( u is available ){
      V = V ∪ {u} ;
    }
Step B:
    if ( |V| ≥ s_l − w + 1
        or ( |V| ≥ |RQ_l| and |U| == 0 ) ){
      Exit as readable ;
    }
Step C:
    if ( |U| + |V| < |RQ_l| ){
      Exit as not readable ;
    }
  }
```

Fig. 3 Procedure for checking whether l-th level is readable.

Let us look at an example of a probabilistic TP for $w = 1$, $\gamma = 0.2$ by using Fig. 2. We assume that the node $B_{1,3}$ is unavailable. The number of nodes in the second level is 5 and $\gamma = 0.2$, $(s_l - w + 1 - t_l) = 4$. Therefore, $RQ_1 = \{B_{1,0}, B_{1,1}, B_{1,2}, B_{1,4}\}$ as well as $RQ_0 = \{B_{0,0}, B_{0,1}\}$, $\{B_{0,0}, B_{0,2}\}$, $\{B_{0,1}, B_{0,2}\}$ and $RQ_2 = \{B_{2,0}, B_{2,1}, B_{2,2}, B_{2,3}, B_{2,4}, B_{2,5}, B_{2,6}\}$. When the latest data is written in $B_{1,3}$, RQ_1 does not provide the latest data.

4. Analysis of Probabilistic Trapezoid Protocol

In this section, we analyze the read availability, the LV read availability and the expected number of nodes accessed, for the probabilistic TP. The write availability for the probabilistic TP is exactly the same as the one derived in Ref. 14). To simplify the analysis, we assumed the following conditions.

1. Each node would fail independently.
2. A node will stop on failure (fail-stop).
3. Communication links would always be available.

For readability, we defined the following expression, which shows the probability that at least j nodes out of i nodes would be available.

$$\Psi(i,j) \equiv \sum_{k=j}^{i} \binom{i}{k} \cdot p^k (1-p)^{i-k}, \tag{9}$$

where p denotes the node availability.

By applying the above expression, the write availability P_{write} for the TP [14] can be expressed as

$$P_{write} = \Psi(b, \lfloor b/2 \rfloor + 1) \cdot \prod_{l=1}^{h} \Psi(s_l, w). \tag{10}$$

The write availability for the probabilistic TP is exactly the same as above.

4.1 Read Availability

The probability that the top level will be readable is equal to the probability that more than half of the b nodes will be available in the probabilistic TP. If $(s_l - w + 1 - t_l)$ nodes are available for the l-th level, then it is readable. That is, the probability that the l-th level will be readable is

$$Q_l = \begin{cases} \Psi(b, \lfloor b/2 \rfloor + 1), & (l = 0) \\ \Psi(s_l, s_l - w + 1 - t_l). & (1 \le l \le h) \end{cases} \tag{11}$$

In the probabilistic TP, if one or more levels are readable, read operations are successful. Therefore, the read availability P_{read} is derived as

$$P_{read} = 1 - \prod_{l=0}^{h} (1 - Q_l). \tag{12}$$

Next, let us derive the LV read availability, G_l, for the l-th level. According to Eq. (7), the latest data is obtained from the top level. This means $G_0 = Q_0$. For the lower level, the probability that the latest data will be available is obtained by

$$G_l = \sum_{k=s_l-w+1-t_l}^{s_l} \xi(k), \quad (1 \le l \le h) \tag{13}$$

where $\xi(k)$ means the probability that exactly k nodes are available and at least one of these k nodes has the latest data. $\xi(k)$ is expressed as

$$\xi(k) = \binom{s_l}{k} \cdot p^k (1-p)^{s_l - k} \cdot \left\{ 1 - \frac{\binom{s_l - w}{k}}{\binom{s_l}{k}} \right\}. \tag{14}$$

This is because $\left\{ 1 - \frac{\binom{s_l - w}{k}}{\binom{s_l}{k}} \right\}$ means the probability that k nodes will have at least one common node with w nodes to which the data are written.

The LV read availability G_{read} can be expressed as

$$G_{read} = \sum_{l=0}^{h} \left(F(l) \cdot P_g(l) \right), \tag{15}$$

where $P_g(l)$ is the probability that the latest data will be read when the read operation starts from the l-th level. Since such a successful read operation is attained somewhere in all the levels, the following equation holds:

$$\begin{aligned} P_g(l) = {} & G_l \\ & + \overline{Q}_l \cdot G_{l+1} \\ & + \overline{Q}_l \cdot \overline{Q}_{l+1} \cdot G_{l+2} \\ & \quad \vdots \\ & + \overline{Q}_l \cdot \overline{Q}_{l+1} \cdot \overline{Q}_{l+2} \cdots \overline{Q}_{h-1} \cdot G_h \\ & + \overline{Q}_l \cdot \overline{Q}_{l+1} \cdot \overline{Q}_{l+2} \cdots \overline{Q}_{h-1} \cdot \overline{Q}_h \cdot G_0 \\ & + \overline{Q}_l \cdot \overline{Q}_{l+1} \cdot \overline{Q}_{l+2} \cdots \overline{Q}_{h-1} \cdot \overline{Q}_h \cdot \overline{Q}_0 \cdot G_1 \end{aligned}$$

$$
\begin{aligned}
&\vdots \\
&+\overline{Q}_l \cdot \overline{Q}_{l+1} \cdot \overline{Q}_{l+2} \cdots \overline{Q}_{h-1} \cdot \overline{Q}_h \\
&\qquad \cdot \overline{Q}_0 \cdot \overline{Q}_1 \cdots \overline{Q}_{l-2} \cdot G_{l-1} \\
&= \sum_{k=l}^{h} \left(\prod_{j=l}^{k-1} \overline{Q}_j \cdot G_k \right) \\
&\quad + \sum_{k=0}^{l-1} \left(\prod_{j=l}^{h} \overline{Q}_j \cdot \prod_{j=0}^{k-1} \overline{Q}_j \cdot G_k \right). \qquad (16)
\end{aligned}
$$

4.2 Average Number of Nodes Accessed

We analyzed the average number of nodes accessed to evaluate the system's load with the probabilistic TP. It was derived separately for read/write operations. The number of the nodes accessed depends on the procedures that check whether the l-th level is readable. In this section, we assume the procedure described in Section 3.

If the l-th level is readable, the average number of nodes accessed with the procedure is derived as

$$
U_l = \begin{cases} \dfrac{U'(b, \lfloor b/2 \rfloor + 1, \lfloor b/2 \rfloor + 1, 0)}{\Psi(b, \lfloor b/2 \rfloor + 1)}, & (l = 0) \\ \dfrac{U'(s_l, s_l - w + 1, s_l - w + 1 - t_l, 0)}{\Psi(s_l, s_l - w + 1 - t_l)}. & \\ & (1 \le l \le h) \end{cases} \qquad (17)
$$

We derived U_l as the conditional expected value under the condition that the level is readable. Thus, the denominator in Eq. (17) is the probability that $(s_l - w + 1 - t_l)$ of s_l nodes will be available. The numerator is recursively derived as

$$
U'(n, j, k, c) = \begin{cases} c, & (j = 0) \\ c, & (n = 0 \text{ and } k \le 0) \\ 0, & (n < k) \\ p \cdot U'(n-1, j-1, k-1, c+1) & \\ \quad +(1-p) \cdot U'(n-1, j, k, c+1), & \\ & (\text{otherwise}) \end{cases} \qquad (18)
$$

where

- n : the number of unchecked nodes;
- j : the number of nodes to form a quorum for the TP with $\gamma = 0$;
- k : the minimum number of nodes to form a quorum for the probabilistic TP;
- c : the parameter to count the number of nodes accessed

However, if the l-th level is not readable, the average number of nodes accessed during the procedure can be derived as

$$
V_l = \begin{cases} \dfrac{V'(b, \lfloor b/2 \rfloor + 1, \lfloor b/2 \rfloor + 1, 0)}{1 - \Psi(b, \lfloor b/2 \rfloor + 1)}, & (l = 0) \\ \dfrac{V'(s_l, s_l - w + 1, s_l - w + 1 - t_l, 0)}{1 - \Psi(s_l, s_l - w + 1 - t_l)}. & \\ & (1 \le l \le h) \end{cases} \qquad (19)
$$

V_l is also the conditional expected value, and thus the denominator in Eq. (19) is the probability that $(w + t_l)$ of s_l nodes will not be available. As well as using Eq. (17), the numerator can be recursively derived by

$$
V'(n, j, k, c) = \begin{cases} 0, & (j = 0) \\ 0, & (n = 0 \text{ and } k \le 0) \\ c, & (n < k) \\ p \cdot V'(n-1, j-1, k-1, c+1) & \\ \quad +(1-p) \cdot V'(n-1, j, k, c+1). & \\ & (\text{otherwise}) \end{cases} \qquad (20)
$$

where

- n : the number of unchecked nodes;

- j : the number of nodes to form a quorum for the TP with $\gamma = 0$;
- k : the minimum number of nodes to form a quorum for the probabilistic TP;
- c : the parameter to count the number of nodes accessed

With the probabilistic TP, the average number of nodes accessed for read operations, C_{read}, is obtained by

$$C_{read} = \sum_{l=0}^{h} \Big(F(l) \cdot C_r(l) \Big), \tag{21}$$

where $C_r(l)$ is the summation of the probability that data will be read starting from the l-th level times the average number of nodes accessed during this trial. That is,

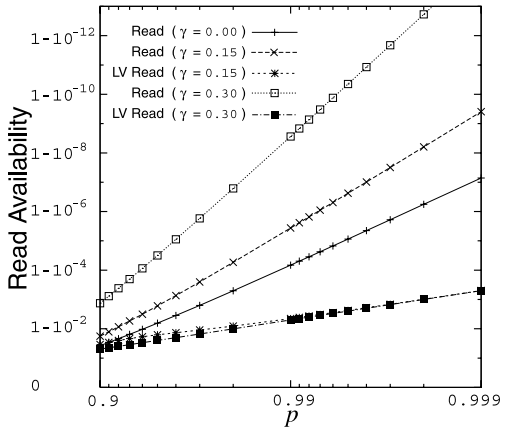
$$\begin{aligned} C_r(l) = & \sum_{k=l}^{h} \left\{ \left(\prod_{j=l}^{k-1} \overline{Q}_j \cdot Q_k \right) \cdot \left(\sum_{j=1}^{k-1} V_j + U_k \right) \right\} \\ & + \sum_{k=0}^{l-1} \left\{ \left(\prod_{j=l}^{h} \overline{Q}_j \cdot \prod_{j=0}^{k-1} \overline{Q}_j \cdot Q_k \right) \cdot \left(\sum_{j=l}^{h} V_j + \sum_{j=0}^{k-1} V_j + U_k \right) \right\} \\ & + \prod_{k=0}^{h} \overline{Q}_k \cdot \sum_{k=0}^{h} V_k. \end{aligned} \tag{22}$$

The write operation checks each level from top to bottom. Therefore, the average number of nodes accessed for write operations, C_{write}, is the summation over the average numbers of nodes accessed for all levels. That is,

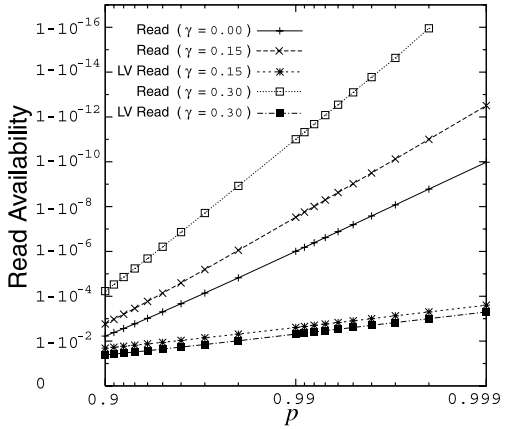
$$\begin{aligned} C_{write} = & U'(b, \lfloor b/2 \rfloor + 1, \lfloor b/2 \rfloor + 1, 0) \\ & + V'(b, \lfloor b/2 \rfloor + 1, \lfloor b/2 \rfloor + 1, 0) \\ & + \sum_{k=1}^{h} \Big(U'(s_k, w, w, 0) + V'(s_k, w, w, 0) \Big). \end{aligned} \tag{23}$$

4.3 Numerical Evaluations

Figure 4 plots the read availability for the probabilistic TP, as a function of the node availability p. Figure 4 (a) plots the results for $s_l = 8l + 4$, $h = 1$, $N = 16$, $f = 0.5$, and $w = 1$. Figure 4 (b) plots the results for $s_l = 2l + 3$, $h = 2$, $N = 15$, $f = 0.5$, and $w = 1$. For example, consider Fig. 4 (a), where the



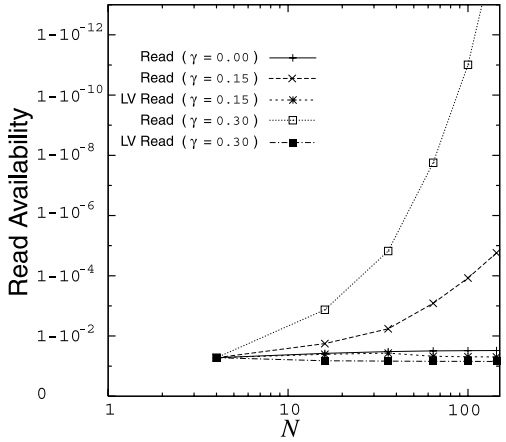
(a) $s_l = 8l + 4$, $h = 1$, $N = 16$, $f = 0.5$, and $w = 1$.



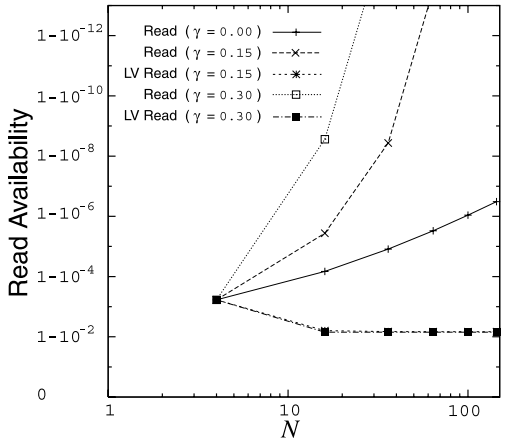
(b) $s_l = 2l + 3$, $h = 2$, $N = 15$, $f = 0.5$, and $w = 1$.

Fig. 4 Read availability for probabilistic TP as a function of the node availability p.

read availability for $\gamma = 0.3$, $p = 0.99$ is about $(1 - 10^{-9})$. Similarly, the read availability for $\gamma = 0.15$ is about $(1 - 10^{-6})$. The read availability for the TP with $\gamma = 0$ is about 0.9999 and the LV read availability for $\gamma = 0.15$ is 0.99. The read availabilities increase as the node availability increases. For a larger γ, the



(a) $s_l = 8l + 4$, $p = 0.9$, $f = 0.3$, and $w = 1$.



(b) $s_l = 8l + 4$, $p = 0.99$, $f = 0.3$, and $w = 1$.

Fig. 5 Read availability for probabilistic TP, as a function of the number of nodes N.

read availability improves quickly, but the LV read availabilities improve much slower.

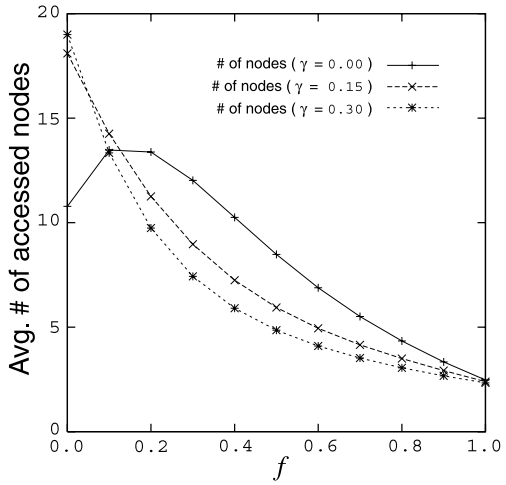
Figure 5 plots the read availability for the probabilistic TP, as a function of the number of nodes N. Figure 5 (a) plots the results for $s_l = 8l+4$, $p = 0.9$, $f = 0.3$, and $w = 1$. Figure 5 (b) plots the results for $s_l = 8l + 4$, $p = 0.99$, $f = 0.3$, and $w = 1$. For example, consider Fig. 5 (a), where the read availability for $\gamma = 0.3$ and $N = 100$ is $(1 - 10^{-11})$. Similarly, the read availability for $\gamma = 0.15$ is about $(1 - 10^{-4})$. The increased N does not contribute to any increased read availability for the TP with $\gamma = 0$. When $p = 0.9$, the read availability for the TP with $\gamma = 0$ is less than 0.99. Figures 5 (a) and (b) indicate that applying γ effectively increases the read availability.

Figure 6 plots the average number of nodes accessed for the probabilistic TP, as a function of the parameter f. Figure 6 (a) plots the results for $s_l = 2l + 3$, $h = 8$, $N = 99$, $p = 0.9$, and $w = 1$. Figure 6 (b) plots the results for $s_l = 2l + 3$, $h = 8$, $N = 99$, $p = 0.99$, and $w = 1$. For example, consider Fig. 6 (a), where the average number of nodes accessed for $\gamma = 0.0$, $f = 0.4$ is almost 10. Similarly, the average number of nodes accessed for $\gamma = 0.3$ is equal to 6. When $p = 0.9$, the increase in γ decreases the average number of nodes accessed. The growth of f can decrease the average number of nodes accessed. However, the increase in f may concentrate read operations on the top level.

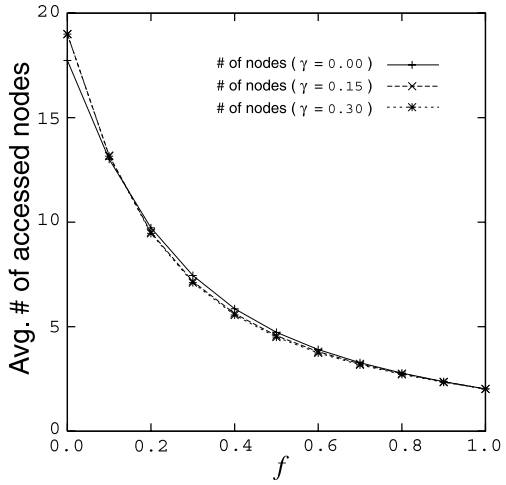
Figure 7 plots the read availability for the probabilistic TP, as a function of the parameter γ. Figure 7 (a) plots the results for $s_l = 8l+4$, $p = 0.9$, and $w = 1$. Figure 7 (b) plots the results for $s_l = 8l + 4$, $p = 0.99$, and $w = 1$. For example, consider Fig. 7 (a), where the read availability for $h = 1$, $\gamma = 0.3$ is about 0.999. The read availability for $h = 4$, $\gamma = 0.1$, is almost 0.999. LV read availabilities do not decrease when $\gamma \geq 0.1$. However, read availabilities greatly increase as γ increases.

Figure 8 plots the average number of nodes accessed for the probabilistic TP, as a function of the parameter γ. Figure 8 (a) plots the results for $s_l = 2l + 3$, $p = 0.9$, $f = 0.3$, and $w = 1$. Figure 8 (b) plots the results for $s_l = 2l + 3$, $p = 0.99$, $f = 0.3$, and $w = 1$. For example, consider Fig. 8 (a), where the average number of nodes accessed for $h = 2, 0 < \gamma < 0.5$, is 5.5. Similarly, the number for $h = 8$, $0.2 < \gamma < 0.5$, is almost 7.5. The larger the h, that is, the larger the N, the more effective the increase in γ was in reducing the load.

In **Table 1**, we can see the read availability for the probabilistic TP and the probabilistic QS. Table 1 lists the results for the probabilistic TP for $s_l = 2l + 3$,

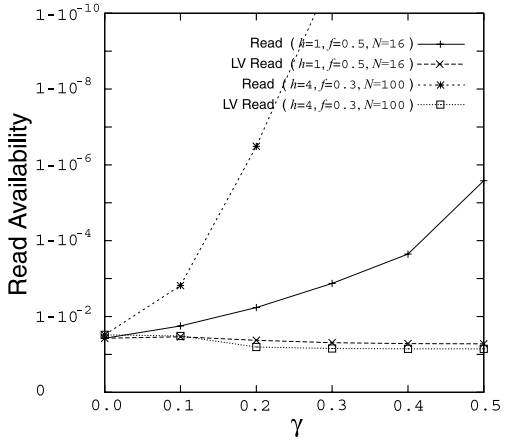


(a) $s_l = 2l + 3$, $h = 8$, $N = 99$, $p = 0.9$ and $w = 1$.

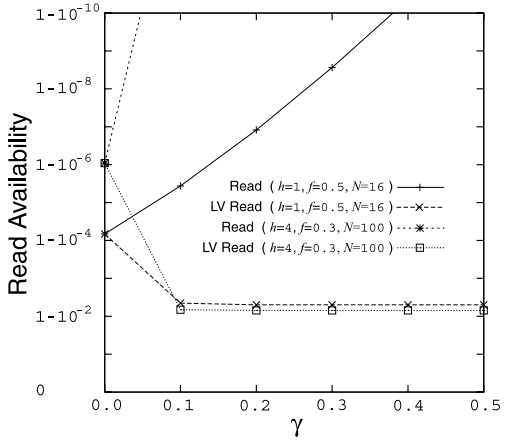


(b) $s_l = 2l + 3$, $h = 8$, $N = 99$, $p = 0.99$ and $w = 1$.

Fig. 6 Average number of nodes accessed for the probabilistic TP, as a parameter f.



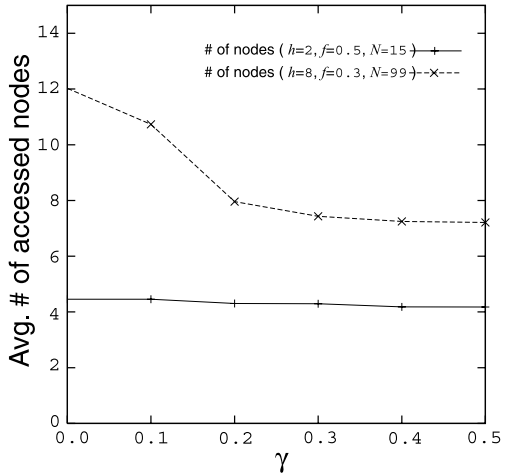
(a) $s_l = 8l + 4$, $p = 0.9$, and $w = 1$.



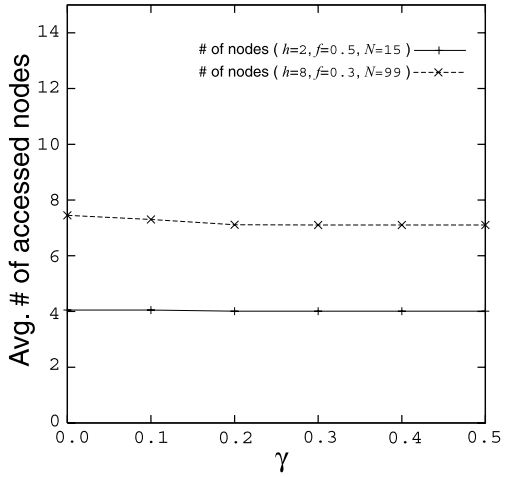
(b) $s_l = 8l + 4$, $p = 0.99$, and $w = 1$.

Fig. 7 Read availability for probabilistic TP, as a function of the parameter γ.

$h = 8$, $N = 99$, $p = 0.99$, $f = 0.3$, $w = 1$, and $\gamma = 0.1$ and the probabilistic QS for $N = 100$, $p = 0.99$, and $|RQ| = |WQ| = 8$. While the average number of accessed nodes for the probabilistic TP is about 7.3, we set the quorum size of



(a) $s_l = 2l + 3$, $p = 0.9$, $f = 0.3$, and $w = 1$.



(b) $s_l = 2l + 3$, $p = 0.99$, $f = 0.3$, and $w = 1$.

Fig. 8 Average number of nodes accessed for probabilistic TP, as a function of the parameter γ.

Table 1 Read availability when $p = 0.99$, probabilistic TP: $s_l = 2l + 3$, $h = 8$, $N = 99$, $f = 0.3$, $w = 1$, $\gamma = 0.1$, probabilistic QS: $N = 100$, $|RQ| = |WQ| = 8$.

	# of accessed nodes	LV read availability	read availability
TP	7.3	0.9978	≈ 1.0
QS	8.0	0.4998	≈ 1.0

Table 2 Read availability when $p = 0.9$, probabilistic TP: $s_l = 2l + 3$, $h = 8$, $N = 99$, $f = 0.3$, $w = 1$ and $\gamma = 0.1$, probabilistic QS: $N = 100$ and $|RQ| = |WQ| = 11$.

	# of accessed nodes	LV read availability	read availability
TP	10.7	0.9851	0.9999
QS	11.0	0.7421	≈ 1.0

the probabilistic QS to 8. Both protocols achieved a read availability very close to 1.0. With the probabilistic QS, the LV read availability was less than 0.5. The LV read availability with the probabilistic TP was about twice that with the probabilistic QS.

Table 2 lists the read availability for the probabilistic TP and the probabilistic QS. It also lists the results for the probabilistic TP with $s_l = 2l + 3$, $h = 8$, $N = 99$, $p = 0.9$, $f = 0.3$, $w = 1$, and $\gamma = 0.1$ and for the probabilistic QS with $N = 100$, $p = 0.9$ and $|RQ| = |WQ| = 11$. Both protocols achieved a high read availability. With the probabilistic QS, however, the LV read availability was 0.7421. This means that the probabilistic QS could not obtain the latest data with a probability of about 25%.

5. Experiments of Data Replication Protocols

In this section, we describe the design and the implementation of the data replication protocol. We also report the measured results of the throughput for VS, grid protocol, and TP on our experimental system. We evaluated the throughput under conditions where the node availability p was set to 1 or 0.

5.1 Experimental Setup

We implemented file transfer protocols based on VS, grid protocol, and TP in C++. The source codes were about 2500 lines. We set up 17 Linux computers (Celeron 2 GHz, 256 MB RAM) for the measurements. Both the clients and the

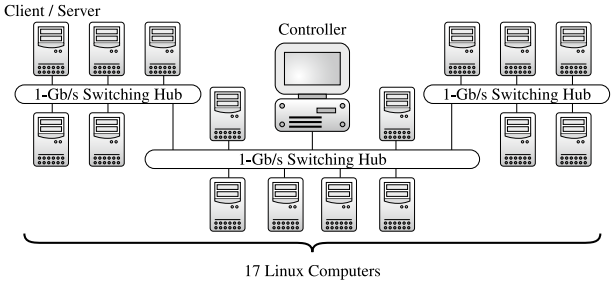


Fig. 9 Experimental setup for data replication protocols.

servers for file transfer were running on 16 computers. These were remotely controlled through a controller. As we can see from **Fig. 9**, the computers and the controller were connected to one another via 1 Gb/s switching hubs.

5.2 Specifications of The File Transfer Protocol

We implemented the file transfer protocol by expanding a simple protocol, TFTP [18]. TFTP assumes that a client will connect to one server node. In this study we designed a version management mechanism for data by adding version information to the data written to servers. We also implemented read and write lock operations for the target data to avoid concurrent executions of conflicting operations.

Figure 10 is a conceptual diagram of the file transfer protocol implemented in this study. Before a transmission, the client sent lock requests to target servers. Each server maintained a queue to store lock requests, and executed lock operations in the order of arrival when this was possible. Then, the servers sent back lock acknowledgements with version information on the data stored. If the client received lock acknowledgements from all servers to which lock requirements had been sent within the time-out interval, T_1, or if it received acknowledgements from the minimum number of required servers within the time-out interval, T_2, it started file transmission based on the received version information. However, if the client failed to receive acknowledgements from the minimum number of required servers within T_2, the write operation would terminate unsuccessfully. For a read operation, the client tried to request the next level. After the file had been transmitted, the client sent unlock requests, and the operation completed.

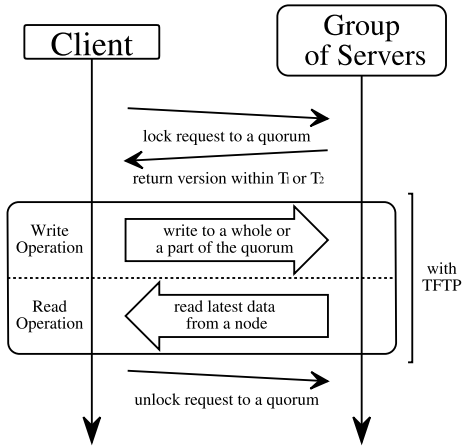


Fig. 10 Overview of the file transfer protocol designed.

Note that file transmissions for multiple servers were performed sequentially.

5.3 Experimental Results and Remarks

Here, we report on the measured results of the throughput in terms of processed requests/s, which were obtained by implementing the file transfer protocol. Throughout all the experiments, the request frequency λ for the operations on each client was set to $0.5 \leq \lambda \leq 2.5$ [the number of requests/s]. The rate at which read operations occurred with respect to write operations was set to 1. The time to measure each result was 1,000 seconds. Time-out intervals were set to $T_1 = 0.1$ [s] and $T_2 = 1.0$ [s]. We also used a text file of 10 KB as the transmitted data.

Figure 11 plots the measured throughputs for TP with $\gamma = 0$, grid protocol, and VS. We applied the same node arrangements as in Fig. 2 to the TP with $\gamma = 0$, where $s_l = 2l + 3$, $h = 2$, $N = 15$, $f = 0.5$, $w = 1$, and $\gamma = 0$. We applied the same 4×4 arrangement as in Fig. 1 to the grid protocol. We set the parameters to $N = 15$, $|WQ| = 8$, and $|RQ| = 8$ for VS. The throughputs for every protocol decreased as the request frequency λ increased. TP had the highest throughput. This can be attributed to the fact that the size of quorums for the TP was smaller than the one for the other protocols.

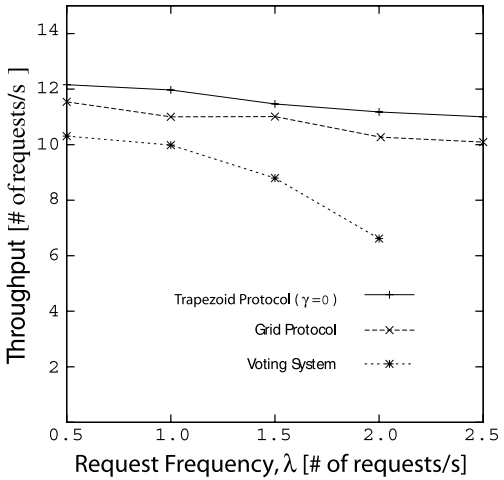


Fig. 11 Average throughput on data replication protocols ($N \approx 15$ and $p = 1.0$), TP: $s_l = 2l + 3$, $h = 2$, $N = 15$, $f = 0.5$, $w = 1$, $\gamma = 0.0$ and grid protocol: 4×4, VS: $N = 15$, $|RQ| = 8$, $|WQ| = 8$.

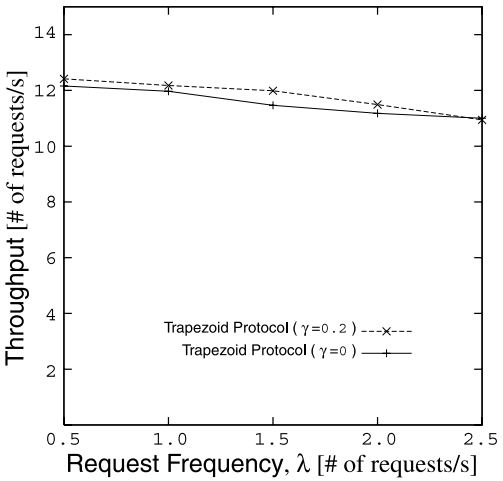


Fig. 12 Average throughput for probabilistic TP using $s_l = 2l + 3$, $h = 2$, $N = 15$, $p = 1.0$, $f = 0.5$, and $w = 1$.

Figure 12 plots the measured throughput for TP using $s_l = 2l + 3$ and $h = 2$ similar to Fig. 2, under different γ conditions, i.e., 0 and 0.2. In all ranges of λ, TP with $\gamma = 0.2$ had a higher throughput, although the differences were not significantly large. They were considered to have been caused when time-outs occurred while the request was kept in the queue.

Table 3 lists the measured throughput for the grid protocol and the TP with $\lambda = 2.0$ [requests/s]. We applied the same node arrangements as in Fig. 2 to the TP where $s_l = 2l + 3$, $h = 2$, $N = 15$, $f = 0.5$, $w = 1$, and $\gamma = 0$ or 0.2. Here, we measured throughput under two conditions, that is, all node were always available (case A), and one node was always unavailable, i.e., $p = 0$ (case B). The unavailable node was selected as $A_{1,2}$ for the grid protocol, and $B_{1,2}$ for the TP. For case A, the difference between the grid protocol and the TP, with $\gamma = 0$, was about 15%, while TPs, with $\gamma = 0$ and $\gamma = 0.2$, had smaller differences. For case B, where one node was unavailable, the grid protocol had the lowest throughput. However, the probabilistic TP with $\gamma = 0.2$ suffered a moderate decrease in throughput compared with the other protocols or conditions.

Table 3 Average throughput [the number of requests/s] on data replication protocols($N \approx 15$, $\lambda = 2.0$ [requests/s]), TP: $s_l = 2l + 3$, $h = 2$, $f = 0.5$, $w = 1$, grid protocol: 4×4.

	case A	case B
# of unavailable nodes	0	1
Grid Protocol	10.01	0.2707
TP ($\gamma = 0.0$)	11.42	0.4751
probabilistic TP ($\gamma = 0.2$)	11.68	1.7570

6. Conclusions

We proposed a probabilistic TP that combined a TP with the concept of probabilistic QS. The proposed technique was able to provide a higher read availability than the one with TP alone. We theoretically analyzed the read availability, the latest version read availability (LV read availability), and the expected number of nodes to be accessed. Our numerical evaluations substantiated the probabilistic TP's improved data availability. As we can see from Fig. 4, read availability is dramatically improved. Figure 8 indicates that when the number of the nodes increases, the average number of nodes accessed can be decreased. Furthermore,

the probabilistic TP achieves greater LV read availability than the probabilistic QS, which consists of almost the same size quorum. We implemented various replication protocols on a system consisting of 17 personal computers and found the probabilistic TP had a better dependability in terms of the availability of data.

A high availability with improved performance is obtained from a combination of factors, including the load balancing and parameter setups in the experiments. A further experimental evaluation, including a discussion about the confidence intervals of the result and a wider range of parameters, will be a part of our future work.

References

1) Avizienis, A., Laprie, J.C., Randell, B. and Landwehr, C.: Basic Concepts and Taxonomy of Dependable and Secure Computing, *IEEE Trans. on Dependable and Secure Computing*, Vol.1, No.1, pp.11–33 (Jan.-Feb. 2004).
2) Aguilera, M.K., Janakiraman, R. and Xu, L.: Using Erasure Codes Efficiently for Storage in a Distributed System, *Int'l Conf. on Dependable Systems and Networks*, pp.336–345 (June 2005).
3) Bernstein, P.A., Hadzilacos, V. and Goodman, N.: *Concurrency Control and Recovery in Database Systems*, Addison-Wesly (1987).
4) Barroso, L.A., Dean, J., Holzle, U. and Google: Web Search for a Planet: The Google Cluster Architecture, *IEEE Micro*, Vol.23, No.2, pp.22–28 (Mar.-Apr. 2003).
5) Carey, M.J. and Livny, M.: Conflict Detection Tradeoffs for Replicated Data, *ACM Trans. on Database System*, Vol.16, No.4, pp.703–746 (Dec. 1991).
6) Naor, M. and Wool, A.: The Load, Capacity and Availability of Quorum Systems, *SIAM Journal on Computing*, Vol.27, No.2, pp.423–447 (Apr. 1998).
7) Gifford, D.K.: Weighted Voting for Replicated Data, *Proc. 7th ACM Symposium on Operating System Principles*, pp.150–162 (Dec. 1979).
8) Peleg, D. and Wool, A.: How to Be an Efficient Snoop, or the Probe Complexity of Quorum Systems, *SIAM Journal on Discrete Mathematics*, Vol.15, No.3, pp.416–433 (Mar. 2002).
9) Cheung, S.Y., Ammar, M. and Ahamad, M.: The Grid Protocol: A High Performance Scheme for Maintaining Replicated Data, *IEEE Trans. on Knowledge and Data Engineering*, Vol.4, No.6, pp.582–592 (Dec. 1992).
10) Agrawal, D. and Abbadi, A.E.: The Tree Quorum Protocol: An Efficient Approach for Managing Replicated Data, *Proc. 16th Very Large Database Conference*, pp.243–254 (Aug. 1990).
11) Agrawal, D. and Abbadi, A.E.: The Generalized Tree Quorum Protocol: An Efficient Approach for Managing Replicated Data, *ACM Trans. on Database System*, Vol.17, No.4, pp.689–717 (Dec. 1992).
12) Jimenez-Peris, R., Patino-Martinez, M., Alonso, G. and Kemme, B.: Are quorums an alternative for data replication?, *ACM Trans. on Database Systems*, Vol.28 No.3, pp.257–294 (Sep. 2003).
13) Youn, H.Y., Lee, D., Lee, B., Choi, J.S., Kim, H.G., Park, C.W. and Su, L.H.: An Efficient Hybrid Replication Protocol for Highly Available Distributed System, *Proc. IASTED Int'l Conf. on Communications & Computer Networks*, pp.508–513 (Nov. 2002).
14) Arai, M., Suzuki, T., Ohara, M., Fukumoto, S., Iwasaki, K. and Youn, H.Y.: Analysis of Read and Write Availability for Generalized Hybrid Data Replication Protocol, *Proc. IEEE Pacific Rim International Symposium on Dependable Computing*, pp.143–150 (Mar. 2004).
15) Malkhi, D., Reiter, M.K. and Wool, A.: Probabilistic Quorum Systems, *Information and Computation*, Vol.170, No.2, pp.184–206 (2001).
16) Luo, J., Hubaux, J.P. and Eugster, P.T.: PAN: Providing Reliable Storage in Mobile Ad Hoc Networks with Probabilistic Quorum Systems, *Proc. ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pp.1–12 (2003).
17) Luo, J., Eugster, P.T. and Hubaux, J.P.: Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks, *Proc. 22nd Annual Joint Conference of the IEEE Computer and Communications Societies* (2003).
18) Sollins, K.: The TFTP Protocol (Revision 2), RFC 1350 (July 1992).

(Received October 5, 2007)
(Accepted March 4, 2008)
(Original version of this article can be found in the Journal of Information Processing Vol.16, pp.50–63.)

Tabito Suzuki received B.E. and M.E. degrees from Tokyo Metropolitan University in 2003 and 2005, respectively. He currently works for Ricoh Corp, and develops multifunction printer firmware.

Mamoru Ohara received his B.E., M.E, and Ph.D. degrees from Tokyo Metropolitan University in 2001, 2003, and 2006, respectively. Currently he is a researcher at Tokyo Metropolitan Industrial Technology Research Institute. His research areas include reliability of embedded systems and distributed systems. He is a member of IEICE.

Masayuki Arai received his B.E., M.E, and Ph.D. degrees from Tokyo Metropolitan University in 1999, 2001, and 2005, respectively. Currently he is an assistant professor of Tokyo Metropolitan University. His research areas include dependable computing and VLSI testing. He is a member of IEEE and IEICE.

Satoshi Fukumoto received his B.S.E., M.S, and Ph.D. degrees from Hiroshima University in 1987, 1989, and 1992, respectively. From 1992 to 2000 he worked in the Department of Information Network Engineering, Aichi Institute of Technology, Japan. Since 2000 he has been working at Tokyo Metropolitan University, where he is now an associate professor in the Faculty of System Design. His research areas include dependable computing, distributed systems, and VLSI testing. He is a member of the Operating Research Society in Japan, IEICE, IEEE, and ACM.

Kazuhiko Iwasaki received a B.E. degree in 1977, an M.E. degree in 1979, and a Ph.D. degree in 1988, all in information and computer sciences, from Osaka University. He joined Hitachi's Central Research Laboratory in 1979, where he researched and developed VLSI processors. From 1990 to 1995 he was an associate professor at Chiba University. Presently, he is a professor at Tokyo Metropolitan University. His research interests include dependable networking and VLSI testing. He is a senior member of the IEEE and a member of the ACM and the IEICE.