社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－DPS－118 (7)
2004／6／3

# Hierarchical Protocol for Byzantine Agreement

**Hiroyuki Yoshino, Satoshi Kawanami, Tomoya Enokido, and Makoto Takizawa**
**Tokyo Denki University, Japan**
**E-mail {yos, kawa, eno, taki}@takilab.k.dendai.ac.jp**

**Abstract**

Peer-to-peer information systems are composed of large number of peer processes interconnected in networks like the Internet. In addition to omission faults and stop faults, we have to discuss how a system is tolerant of Byzantine faults of processes to realize fault-tolerant systems. Even Byzantine agreement protocols with message authentication imply large communication overhead $O(n^t)$ for the number $n$ of processes and the maximum number $t$ of faulty processes. In order to reduce the overhead, we consider a hierarchical group composed of subgroups. Each subgroup shares at least one correct process with some subgroups in order to be tolerant of faults of leaders in each subgroup. Even if a leader process of a subgroup is faulty, all the correct processes can make an agreement on the correct value in a whole group. We evaluate the protocol in terms of number of messages and rounds.

# 階層型ビザンティン合意プロトコル

吉野 宏征　河浪 悟士　榎戸 智也　滝沢 誠
東京電機大学大学院理工学研究科情報システム工学専攻
E-mail {yos, kawa, eno, taki}@takilab.k.dendai.ac.jp

プロセスとネットワークに障害が発生する環境で、複数のプロセス間での合意プロトコルを論じる。Peer-to-Peer 型のアプリケーションのように数千から数万のプロセスが協調動作を行う環境では、停止障害に加えてビザンティン障害を考える必要がある。ビザンティン合意プロトコルは、合意に達するための通信と処理負荷が大きく、実現が困難である。本論文では、階層化したグループを用いて、プロセスのビザンティン障害に対処し、合意に必要なメッセージ数と時間を減少させる方法を提案する。

## 1. Introduction

In peer-to-peer (P2P) applications [9], a collection of multiple processes are cooperating to achieve some objectives. The collection of the cooperating peer processes is referred to as *group*. Processes in a group are exchanging messages with each other in a network. For example, messages are required to be reliably delivered to multiple processes in group communications [3–5]. A group of multiple processes make an agreement to realize the cooperation of the processes.

In this paper, we assume the underlying network is reliable and synchronous [2]. Processes may suffer from Byzantine fault [5]. We discuss how to realize the agreement among processes in a scalable group in presence of Byzantine faults of processes. Let $n$ be the number of processes and $t$ ($\leq n$) be the maximum number of faulty processes in a group of processes. In Byzantine agreement protocols [5], a leader process distributes a value $v$ to all the member processes. On receipt of the value $v$, each member process forwards the value to the other member processes. A process receives values forwarded by other processes. After exchanging the values in $t$ rounds, each correct process takes a majority value out of the values received. All correct processes can make an agreement only if $n \geq 2t + 1$. Here, $(n-1)^t$ messages are transmitted. Even in Byzantine agreement protocols with message authentication [3], the communication overhead is $O(n^t)$. Due to the large overhead, it is dif?cult, maybe impossible to realize the Byzantine agreement protocol. In this paper, we introduce a hierarchical group which is tolerant of Byzantine faults of processes to reduce the overhead of the Byzantine agreement protocol. In each subgroup, the Byzantine agreement protocol is adopted to make an agreement among all the correct processes. If a leader process of a subgroup is faulty, correct processes cannot agree on the correct value while the correct processes can agree on some value in the subgroup. In order to resolve this problem, each subgroup shares at least one correct process named *gateway* process with at least one subgroup. Even if the leader is faulty in a subgroup, at least one gateway process can forward the correct value obtained in another subgroups to all the correct processes in the subgroup.

In section 2, we overviews Byzantine agreement protocols. In section 3, we discuss the hierarchical Byzantine agreement protocol. In section 4, we evaluate the protocol.

## 2. Models of Faults

### 2.1 Types of faults

A system is composed of processes interconnected in a network. We assume the network to be reliable and synchronous [2]. In addition, the maximum de-

lay time between every pair of processes is bounded. A process is *correct* if and only if (*iff*) the process behaves only according to the speci?cation. Otherwise, the process is *faulty*. There are following types of faults [3] from which a process suffers:

1. Crash (or stop) fault: A faulty process stops prematurely and does nothing.
2. Omission fault: A faulty process stops prematurely, or intermittently omits to send or receive messages, or both.
3. Byzantine (arbitrary) fault: A process can exhibit arbitrary behavior.

Models of distributed systems are classi?ed in terms of types of faults to occur. A model $A$ is more *severe* than another model $B$ ($A \leftarrow B$) if a set of faulty behaviors allowed by $B$ is a proper subset of the set of those allowed by $A$ [3] where a directed edge shows the severe relation $\rightarrow$. Thus, an algorithm which tolerates faults in the model $A$ also tolerates faults of the model $B$. Crash $\rightarrow$ Omission $\rightarrow$ Byzantine. Thus, an algorithm tolerant of Byzantine faults of processes is also tolerant of every type of fault.

## 2.2 Byzantine agreement protocols

A group of $n$ ( $> 1$) processes $p_0$, $p_1$, ..., $p_{n-1}$ are required to make an agreement on a value in presence of process faults. The agreement protocol is brie?y described as follows:

1. At ?rst, a process $p_0$ sends some value $v$ to all the other processes $p_1$, ..., $p_{n-1}$. The process $p_0$ is referred to as *leader* one.
2. Every correct member process receiving a value $v$ from the leader process checks if all the other correct processes receive the same value $v$.

Here, not only a member process but also a leader process may be faulty. A faulty process may send different values to different processes and may not send any value to some process. A group of multiple processes are referred to as make a *Byzantine agreement* iff both of the following conditions are satis?ed [5]

IC1. All correct processes agree on a same value.

IC2. If a leader process is correct, a every correct process agrees on the value which the leader sends.

The second condition IC2 is required to hold since even a leader process may be faulty. If a leader process is faulty, all the correct processes agree on some *bottom* value $\perp$. If the leader process is correct, every correct process is required to make an agreement on the value which the leader process sends. In the agreement protocols, a leader process sends a value to all the member processes at the ?rst round. On receipt of a value from the leader process, each process forwards the value to all the other processes. This is the second round. At the ($t$ +1)th round, a process takes a majority value among values which the process has so far received. Each process $p_i$ accumulates values which the process $p_i$ has received in a set $V_i$. Let *majority*($V$) be a function which takes a majority value $v$ in a set $V$ of values. If there is no majority value in $V_i$, *majority*($V_i$) = $\perp$. Let $t$ be the maximum number of faulty processes in a group of $n$ processes ($t < n$). The Byzantine agreement can be realized in a group of $n$ processes only if $n > 2t + 1$ [5]. The number $O(n^t)$ of messages are exchanged and it takes $t + 1$ rounds to make an agreement among all the correct processes.

In order to reduce the overhead, the Byzantine agreement protocols with message authentication (BAMA Protocol) [5] are discussed. Each process signs a message with its unforgeable signature for sender authentication and then sends the message. Let $x{:}i$ denote the value $x$ signed by a process $p_i$. $x{:}i{:}...{:}k{:}j$ stands for $(x{:}i{:}...{:}k){:}j$. Suppose a faulty process $p_i$ sends a value $v'$ to a process $p_k$ after receiving a value $v{:}j$ from a process $p_j$. On receipt of a value $v'{:}i$ from $p_i$, the process $p_k$ detects that $p_i$ is faulty since the $p_j$'s sign on the value $v'$ is forged. On receipt of a value, each process $p_i$ accumulates the value to a variable $V_i$ if the value is properly signed. The BAMA protocol is brie?y presented as follows:

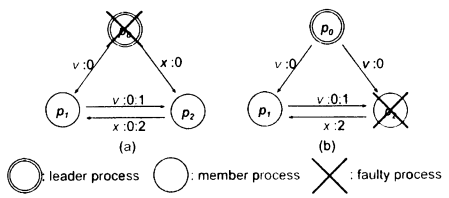


**Figure 1. Byzantine agreement with message authentication (BAMA).**

1. A leader process $p_0$ signs a value $v$ and sends the signed value $v{:}0$ to all the member processes $p_1$, ..., $p_{n-1}$.
2. On receipt of a value $v{:}j_0{:}j_1{:}...{:}j_k$ from a process $p_{j_k}$ ($k \geq 0$, $j_0 = 0$, $j_h \in \{1, ..., n\text{-}1\}$ ), if the value $v$ is proper, i.e. $v$ is not forged.
   i. a process $p_i$ adds the value $v$ to $V_i$;
   ii. if $k < t$, $p_i$ sends the signed value $v{:}0{:}j_1{:}...{:}j_k{:}i$ to every other process than $p_0, p_{j_1}, ..., p_{j_k}$.

   else ($v$ is not correct ) $p_i$ perceives $p_j$ to be faulty.
3. If $p_i$ receives no more message, a single element $v$ (= *majority*($V_i$)) is obtained from $V_i$.

At step 3, each correct process takes the majority value in the set $V_i$ by using the function *majority*. Figure 1 shows three processes $p_0$, $p_1$, and $p_2$ where one

of the processes is faulty. In case (a), the leader process $p_0$ is faulty. The leader $p_0$ sends different values $v$ and $x$ to a pair of member processes $p_1$ and $p_2$, respectively. On receipt of the values, the member processes $p_1$ and $p_2$ forward the values $v$:0:1 and $x$:0:2 to $p_2$ and $p_1$, respectively. The process $p_1$ receives values $v$:0 from $p_0$ and $x$:0:2 from $p_2$. Here, $p_1$ ?nds the leader $p_0$ to be faulty since the $p_0$'s signature on the value $x$:0:2 is not forged by $p_2$. Here, $p_1$ takes a value $\perp$. The other correct process $p_2$ takes $\perp$ in a same way.

In case (b), a member process $p_2$ is faulty. The leader process $p_0$ sends a signed value $v$:0 to the processes $p_1$ and $p_2$. Then, the faulty process $p_2$ sends a value $x$ to $p_1$. Here, $p_1$ ?nds $p_2$ to be faulty. If at most one process is faulty, the process $p_1$ can take the value $v$ because $p_0$ must be correct. If values are not signed, $p_1$ cannot decide which process $p_0$ or $p_2$ is faulty even if at most one process is faulty. Thus, the BAMA protocol is required to satisfy the following property for a group of $n$ processes:

A1. At least two correct processes exist in a group for reaching an agreement, i.e. $n > t + 1$.

In the BAMA protocol, each process sends a value to ($n$ - $i$) processes at the $i$th round. Hence, it takes $t$ +1 rounds and ($n$ -1)($n$ -2)...($n$ - $t$ -1) messages are transmitted in a group. The computation and communication overheads are too large to realize the Byzantine agreement protocol in a large-scale group.

## 3. Hierarchical Byzantine Agreement

### 3.1 Hierarchical group

A group $G$ is a collection of $n$ peer processes $p_0$, $p_1$, ..., $p_{n-1}$. Every pair of processes are assumed to reliably communicate with one another with the same bandwidth in a network. A process $p_0$ is a leader process and the other processes are members in the group $G$. A set of $n$ -1 member processes $p_1$, ..., $p_{n-1}$ are partitioned into *subgroups* $G_1$, ..., $G_s$ ($s > 1$). Here, each subgroup $G_i$ includes processes $p_{i0}$, $p_{i1}$, ..., $p_{i,k_i-1}$ where $p_{ij} \in \{p_1, ..., p_{n-1}\}$ ($j$ = 0, 1, ..., $k_i$-1). A *root subgroup* $G_0$ is a collection of the processes $p_0$, $p_{10}$, ..., $p_{s0}$ where $p_{i0}$ is the leader process of a subgroup $G_i$ ($i$ = 1, ..., $s$). A Byzantine agreement protocol with message authentication (BAMA protocol) is used for making a Byzantine agreement among $k_i$ processes in a subgroup $G_i$.

The leader process $p_0$ ?rst sends a value $v$ to all the member processes $p_{10}$, ..., $p_{s0}$. In each subgroup $G_i$ ($i$ = 1, ..., $s$), a leader process $p_{i0}$ forwards a value to all the member processes $p_{i1}$, ..., $p_{i,k_i-1}$. On receipt of a value $v$ from another process, a process $p_{is}$ forwards the value to other processes to make an agreement on the value $v$ sent by $p_0$. The communication and computation overheads for making a Byzantine agreement depend on the total number $n$ of processes. Every subgroup includes the same number of processes so that the communication and computation overheads are uniformly distributed to all the processes. Each subgroup $G_i$ can be furthermore composed of subgroups $G_{i1}$, ..., $G_{is_i}$. In this paper, we discuss a two-layered group for simplicity, i.e. a root subgroup $G_0$ with subgroups $G_1$, ..., $G_s$.

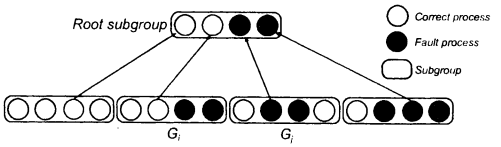


**Figure 2. Two-layered group.**

First, a leader process $p_0$ sends a value $v$ to all the member processes $p_{10}$, ..., $p_{s0}$ in a root subgroup $G_0$. By using the BAMA protocol, each correct process $p_{i0}$ obtains some value $v$ on which all the correct processes agree in the root subgroup $G_0$. Here, "$s > t$ +1" is required to hold to make an agreement for maximum number $t$ of faulty processes in the group $G$. If a process $p_{i0}$ is correct, $p_{i0}$ noti?es all the processes of the agreed value $v$ as a leader process in a subgroup $G_i$. All correct processes agree on the value $v$ by using the Byzantine agreement protocol with message authentication in each subgroup $G_i$.

Next, suppose that a leader process $p_{i0}$ is faulty in a subgroup $G_i$. Here, a subgroup whose leader process is correct is referred to as *correct*. Otherwise, the subgroup is *faulty*. The leader process $p_{i0}$ may send another value or may not send any value to each member process in the subgroup $G_i$ even if every correct leader process makes an agreement on a value $v$ in a root subgroup $G_0$. All the correct processes in a faulty subgroup $G_i$ make an agreement on a bottom value $\perp$ although the correct processes agree on the value $v$ in every correct subgroup. We have to discuss how every correct process makes an agreement on the value $v$ even in a faulty subgroup.

### 3.2 Shared subgroups

In order to resolve the inconsistent agreement in a hierarchical group, each subgroup shares at least one correct process named *gateway* process with at least one subgroup. Suppose a process $p$ is included in both subgroups $G_i$ and $G_j$ as shown in Figure 3. Suppose the process $p$ obtains an agreed value $v$ in one subgroup $G_i$. In another group $G_j$, a leader process $p_{j0}$ is faulty and sends a value $v'$ ($\neq v$) to the gateway process $p$. Through the Byzantine agreement protocol with message authentication (BAMA protocol) in the

subgroup $G_j$, the gateway process $p$ ?nds the leader process $p_{j0}$ to be faulty and takes a value $\perp$. Here, every correct process including the process $p$ knows which process is correct in the subgroup $G_j$. The gateway process $p$ is referred to as *inconsistent* if one of the gateways is faulty and $p$ takes different values in the subgroups. Here, the process $p$ sends the value $v$ on which $p$ agreed in the subgroup $G_i$ to all the correct processes in $G_j$. Then, every correct process makes an agreement on the value $v$ by the BAMA protocol. Each subgroup $G_i$ is required to include more number of processes than $t+1$, i.e. $k_i > t+1$.

In addition, if a leader process of every subgroup is faulty, the Byzantine agreement cannot be realized. Hence, at least one subgroup is required to be correct, i.e. $s \geq t+1$.

If a pair of subgroups $G_i$ and $G_j$ are sure to share at least one correct process, $G_i$ and $G_j$ are referred to as *directly connected* ($G_i \Leftrightarrow G_j$). Here, the leader process of $G_i$ is faulty. Hence, the maximum number of faulty member processes in all the subgroups is $(t-1)$. Hence, if $|G_i \cap G_j| \geq t$, $G_i$ and $G_j$ are directly connected ($G_i \Leftrightarrow G_j$). In addition, $G_i$ and $G_j$ are *connected* ($G_i \leftrightarrow G_j$) if $G_i \Leftrightarrow G_j$ or $G_i \leftrightarrow G_k \leftrightarrow G_j$ for some subgroup $G_k$. If a pair of subgroups $G_i$ and $G_j$ share some number of processes, $G_i$ and $G_j$ are referred to as *intersect*. $G_i$ and $G_j$ are *related* ($G_i \rightleftharpoons G_j$) if $G_i$ and $G_j$ intersect but are not directly connected. In Figure 3, $G_i \Leftrightarrow G_j$ if $t = 2$.

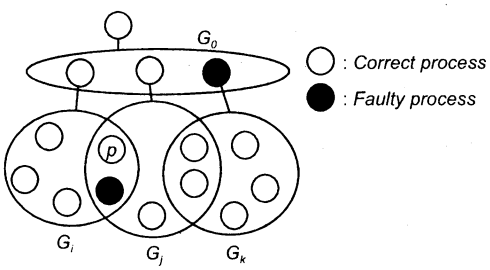


**Figure 3. Shared subgroups (t=2).**

In this paper, we make following assumptions:

**[Assumptions]**

1. Each process can send signed messages directly to every other process in a subgroup.
2. A faulty process can be detected by the message authentication.
3. No communication fault occurs.

### 3.3 Agreement protocol

We discuss how to make a Byzantine agreement on subgroups $G_1$, ..., $G_s$ for a collection of $n$ processes $p_0$, $p_1$, ..., $p_{n-1}$. A process $p_0$ is a leader, i.e. $p_0$ ?rst sends a value to every leader process $p_{i0}$ of every subgroup $G_i$ ($i$ = 1, ..., $s$). Here, let $G_0$ be a root subgroup which is a collection of the leader process $p_0$ and the leader processes of subgroups, $G_0 = \{p_0, p_{10}, ..., p_{s0}\}$.

Processes in a group $G$ make a Byzantine agreement by the following protocol:

**[Hierarchical agreement protocol]**

1. A leader process $p_0$ sends a signed value to a process $p_{i0}$ of every subgroup $G_i$ ($i$ = 1, ..., $s$).
2. The leader processes $p_{10}$, ..., $p_{s0}$ exchange values according to the BAMA protocol. Every correct leader process $p_{i0}$ of $G_i$ agrees on a value $v$.
3. A leader process $p_{i0}$ sends the value $v$ to all the processes in a subgroup $G_i$.
4. The processes $p_{i1}$, ..., $p_{i,k_i-1}$ make a Byzantine agreement on $v$ by the BAMA protocol in $G_i$.
5. If a process $p_{it}$ ?nds the leader $p_{i0}$ to be correct, $p_{it}$ agrees on $v$ and then terminates.
6. The process $p_{it}$ takes $\perp$ and $G_i$ is faulty if $p_{i0}$ is faulty. The process $p_{it}$ waits for a correct value from another correct member process.
7. If $p_{it}$ is an inconsistent gateway process among the faulty subgroup $G_i$ and another correct subgroup $G_j$, $p_{it}$ distributes the value $v$ on which $p_{it}$ agrees in $G_j$ to all the processes in the faulty subgroup $G_i$ by the BAMA protocol.
8. A correct process $p_{it}$ agrees on the value $v$ in $G_i$. If $p_{it}$ is still an inconsistent gateway with another faulty $G_j$, $p_{it}$ forwards the value $v$ to $G_j$ at step 7.

**[Theorem]** A group $G$ is composed of subgroups $G_1$, ..., $G_s$. The Byzantine agreement is realized in the group $G$ if the following conditions are satis?ed:

1. $s > t$.
2. $k_i > t+1$ for every subgroup $G_i$.
3. Every pair of subgroups $G_i$ and $G_j$ are connected ($G_i \leftrightarrow G_j$).
4. Every gateway process is not a leader process in each subgroup.

### 3.4 Design of shared subgroups

We discuss how to construct a hierarchical group for a collection $G$ of $n$ processes $p_0$, $p_1$, ..., $p_{n-1}$ and maximum number $t$ of faulty processes. We assume each subgroup $G_i$ includes the same number $k$ of processes so that every process spends a same amount of computation resource to make a Byzantine agreement. Subgroups $G_0$, $G_1$, ..., $G_s$ are constructed for the group $G$ as follows:

1. A root subgroup $G_0$ is a collection $\{p_0, p_1, ..., p_s\}$ of processes where $p_0$ is a leader process.

2. A subgroup $G_i$ is a collection $\{p_i, p_{s+1+h(i-1)}, ..., p_{s+h(i-1)+k-1}\}$ of processes ($i$ = 1, ..., $s$) where $p_i$ is a leader process. Here, the suf?x "$s$ +1 +$x$" stands for "$s$ +1 + $x$ mod ($n$ -$s$ -1)".
3. For a subgroup $G_s$, $s$ +1 +$h$($s$ -1) $\leq n$ and $s$ +$h$($s$ -1) +$k$ − 1 $\geq n$, i.e. ($n$ +$h$ -1) / ($h$ +1) $\geq s \geq$ ($n$ +$h$ -$k$ +1) / ($h$ +1). If every subgroup includes the same number of processes, $k = s$. ($n$ +$h$ +1) / ($h$ +1) $\geq s \geq$ ($n$ +$h$ +1) / ($h$ +2).

The total number of *effective* processes in the group $G$ with subgroups $G_1$, ..., $G_s$ is the summation of numbers of processes in subgroups, i.e. ($s$ +1) +$s$·$k$ = $s$($k$ +1) +1. The redundancy factor $r_G$ for a group $G$ is [($s$ +1) +$s$·$k$]/$n$. In a non-hierarchical ?at group $G$, $r_G$ = 1. The redundancy factor shows additional overhead of each process in a hierarchical group. A pair of subgroups $G_i$ and $G_j$ include [$k$ -1 + $h$($i$ - $j$)] common processes $p_{s+1+h(j-1)}$, ..., $p_{s+h(i-1)+k-1}$ if $h(i - j) + k - 1 > 0$.

**[Theorem]** A pair of subgroups $G_i$ and $G_j$ are directly connected ($G_i \Leftrightarrow G_j$) if $h(i - j) + k - 1 \geq t$.

That is, if a leader process is correct in one of subgroups $G_i$ and $G_j$, every correct process in each of $G_i$ and $G_j$ can make an agreement on the same value. Let us consider a group of 17 processes ($n$ = 17). Figures 4 and 5 show hierarchical groups which include four subgroups $G_1$, $G_2$, $G_3$ and $G_4$ ($s$ = 4) for $t$ = 1 and $t$ = 2, respectively. Each subgroup includes ?ve ($k$ = 5) and six ($k$ = 6) processes, respectively. The redundancy factors are $r_G$ = 1.47 in Figure 4 and $r_G$ = 1.56 in Figure 5. In the groups shown in Figures 4 and 5, $h$ = 3. The Byzantine agreement is realized for $t$ = 1 and $t$ = 2, respectively. $G_1 \Leftrightarrow G_2 \Leftrightarrow G_3 \Leftrightarrow G_4$. $G_1$ and $G_2$ shares two processes in Figures 4 and 5, respectively.

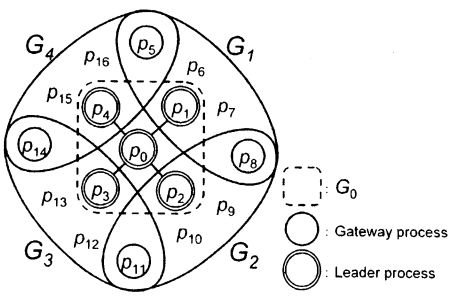


**Figure 4. Shared subgroups (t = 1).**

## 4. Evaluation

We evaluate the Byzantine agreement protocol for a shared hierarchical group $G$ compared with the traditional Byzantine agreement protocol with message authentication in terms of how many messages are transmitted and how long it takes to make a Byzantine agreement among processes. Suppose the number of messages exchanged in each subgroup of $k$ processes is given in a function $N(k, t)$ (= ($k$ -1)($k$ -2)...($k$ -$t$ -1)) for the maximum number $t$ of faulty processes in the group $G$.

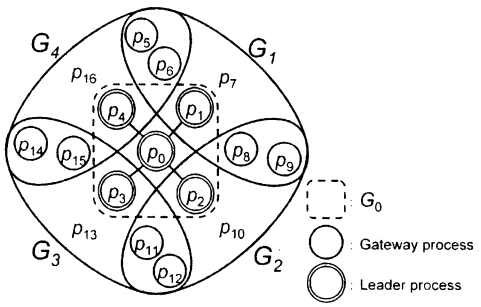


**Figure 5. Shared subgroups (t = 2).**

The minimum number of messages exchanged in the hierarchical group is $N(s, t)$ + $s$·$N(k, t)$. Here, a leader process of every subgroup is correct. It takes ($t$ +1) rounds in the root subgroup $G_0$ and ($t$ +1) rounds in each subgroup. Hence, it takes 2($t$ +1) rounds to make an agreement.

In the worst case, only one subgroup has a correct leader and the other subgroups are faulty. In addition, each subgroup has at least one and at most two subgroups which are directly connected with the subgroup, i.e. linearly chained as shown in Figure 6. Here, only a subgroup $G_1$ is correct and the other subgroups $G_2$, ..., $G_s$ are faulty. Every correct process agrees on a value $v$ in a subgroup $G_1$. First, a correct gateway process between $G_1$ and $G_2$ forwards the value $v$ in the subgroup $G_2$. Then, a gateway process of $G_2$ and $G_3$ distributes the value $v$ in $G_3$ and ?nally in $G_s$. Thus, ($s$ -1)$N(k, t$ -1) messages are transmitted. Totally, $N(s, t)$+$s$·$N(k, t)$+($s$ -1)·$N(k, t$ -1) messages are transmitted. Here, it takes 2($t$ +1) + ($s$ -1)$t$ rounds. The worst case is shown in Figure 6.

The redundancy factor [($s$ +1) + $s$·$k$]/$n$ shows the total processing overhead. We assume the processing overhead of each process in a subgroup is proportional to the number of messages transmitted, i.e. $N(k, t)$. The total processing overhead is given ($s$ +1)·$N(s, t)$ + $s$($k$ +1)·$N(k, t)$ in the best case. ($s$ +1)·$N(s, t)$ + $s$·($k$ +1)·$N(k, t)$ + ($s$ -1)·($k$ +1)·$N(k, t$ -1) in the worst case.

First, we assume $k = s$, i.e. every subgroup includes the same number of processes. For each $n$ and $t$, a hierarchical group where the numbers of messages and

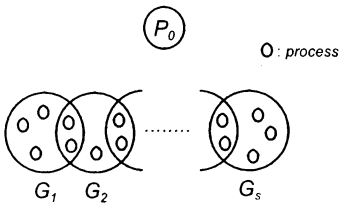


**Figure 6. Chained subgroups.**

rounds are minimized is found. Figures 7 and 8 show the numbers of messages and rounds in a each minimum subgroup for the total number $n$ of processes and ratio of the number of faulty processes $t/n$ ($r = t/n$). In the hierarchical protocol, the number of messages and rounds can be reduced.

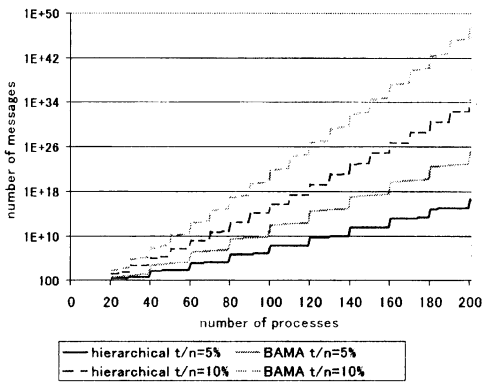


**Figure 7. Number of messages.**

## 5. Concluding Remarks

In this paper. we discussed how to make a Byzantine agreement on the delivery and orderly of messages by using the hierarchical group. We showed how many messages can be reduced to make a Byzantine agreement in the evaluation.

## References


[1] K. P. Birman and R. V. Renesse. *Reliable Distributed Computing with the Isis Toolkit.* IEEE Computer Society Press, 1993.

[2] M. Fischer, N. Lynch, and M. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985.

[3] V. Hadzilacos and S. Toueg. Fault-Tolerant Broadcasts and Related Problems. In *Distributed Systems*, chapter 5, pages 97–145. Addison Wesley, 1993.


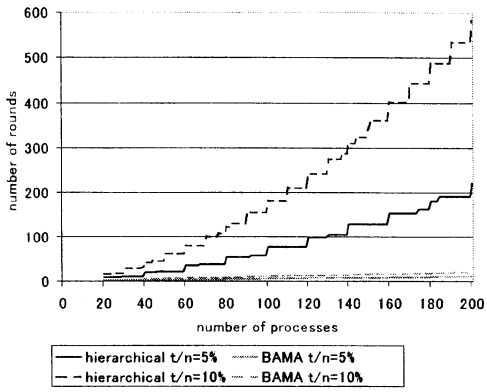


**Figure 8. Number of rounds.**


[4] L. Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. *Comm. ACM*, 21(7):558–565, 1978.

[5] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, Vol. 4(No. 3):382–401, 1982.

[6] F. Mattern. Virtual Time and Global States of Distributed Systems. *Parallel and Distributed Algorithms*, pages 215–226, 1989.

[7] A. Nakamura and M. Takizawa. Priorotiy-Based Total and Semi-Total Ordering Broadcast Protocols. In *Proc. of the 12th IEEE Int'l Conf. on Distributed Computing Systems (ICDCS-12)*, pages 178–185, 1992.

[8] A. Nakamura and M. Takizawa. Causally Ordering Broadcast Protocol. In *Proc. of the 14th IEEE Int'l Conf. on Distributed Computing Systems (ICDCS-14)*, pages 48–55, 1994.

[9] A. Oram. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies.* O'Reilly & Associates, 2001.

[10] T. Tachikawa, H. Higaki, and M. Takizawa. Group Communication Protocol for Realtime Applications. In *Proc. of the 18th IEEE Int'l Conf. on Distributed Computing Systems (ICDCS-18)*, pages 40–47, 1998.

[11] T. Tojo, T. Enokido, and M. Takizawa. Noti?cation-Based QoS Control Protocol for Multimedia Group Communication in High-Speed Networks. In *Proc. of the 24th IEEE International Conf. on Distributed Computing Systems (ICDCS-2004).*