The Generalized Consensus Problem in a Cloud Computing Environment

S.C. Wang S.S. Wang K.Q. Yan* C.P. Huang
(corresponding author)
scwang@cyut.edu.tw sswang@cyut.edu.tw kqyan@cyut.edu.tw s9714625@cyut.edu.tw
Chaoyang University of Technology

***Abstract*—The new concept, cloud computing uses low-power hosts to achieve high reliability and will ensure the ability to be better. Cloud computing increases the number of user's applications on the Internet. In this research, the cloud computing environment can provide better reliability and fluency is focused. The consensus problem is fundamental to fault-tolerant distributed systems, but previous studies of the consensus problem are not suitable for a cloud computing environment. To enhance fault tolerance, the consensus problem in a cloud computing environment is revisited in this study. The Generalized Consensus Protocol of Cloud Computing (GCPCC) that we proposed can solve the consensus problem with a minimal number of rounds of message exchange and tolerates a maximal number of faulty components. The GCPCC attempts to solve the consensus problem and makes all correct components in a cloud computing environment achieve stable results without any influence from faulty components.**

***Keywords* — Consensus, Cloud Computing, Distributed System, Fault Tolerant.**

I. Introduction

Today, network bandwidth and hardware technology must continuously advance to keep pace with the vigorous development of the Internet. The new concept of cloud computing allows for more applications for internet users [1,3,7,10,15]. In the real world of technology, the distributed system has to provide better reliability and fluency with service applications. Cloud computing is currently used many commodity computers that can cooperate to perform a specific service together. In addition, the Internet applications are continuously enhanced with multimedia, and vigorous development of the device quickly occurs in the network system [1,15]. As network bandwidth and quality outstrip computer performance, various communication and computing technologies previously regarded as being of different domains that can integrated, such as telecommunication, multimedia, information technology, and construction simulation. Thus, applications associated with network integration have gradually attracted considerable attention. Similarly, cloud computing facilitated through distributed applications over networks has also gained increased recognition. In a cloud computing environment, users have access to faster operational capability with internet application [19], and the computer systems must have high stability to handle where many users to execute in the environment. In a distributed computing system, components allocated to different places or in separate units are connected, so that they may collectively be used to greater advantage [4]. Cloud computing can ensure increased ability to use low-power hosts to achieve high reliability. In addition, cloud computing has greatly encouraged distributed system design and application to support user-oriented service applications [8]. Furthermore, many applications of cloud computing are increase user convenience, such as YouTube [13]. Component reliability is one of the most important aspects of cloud computing as it ensures overall reliability and fluency. Thus, the tasks in a distributed system must be synchronously completed and components must achieve common agreement. To ensure the cloud computing environment is reliable, a mechanism to ensure that a set of transmission media and nodes reaches an agreed value is thus necessary.

The concept of cloud computing uses many commodity computers in conjunction to complete a specific service for users. The requisite large number of computers will necessarily introduce faulty components into the system. However, the system has to allow for the toleration of faults while maintaining functionality. In the computer system,

each node has to pass messages through transmission media to other nodes to cooperate completed user requests. Many users in the cloud computing environment where have to execute application services simultaneously. Therefore, the high fault-tolerant capability of a cloud computing environment needs to be respected. However, the symptoms of faulty components can influence the normal operation of a system. Cloud computing of distributed systems tolerates the faulty components in the service environment because the system should respond to user requests quickly and completely the user requests as service. The requisite large number of computers maybe meet some computers will be fault to introduce faulty components into the system. However, components in the cloud computing environment cannot occur too much failure, otherwise should influence system provided application services for users, so that, the high fault-tolerant capability of a cloud computing environment needs to be respected. Simultaneously, in the cloud computing environment, nodes that receive user requests maybe influence by the failure transmission media and nodes. Hence, to remove the affect of faulty nodes is need to be mitigated. In a cloud computing environment, achieving perfect reliability must be accomplished by allowing a given set of transmission media and nodes to reach a common agreement even in the presence of faulty components.

The Byzantine Agreement (BA) problem has been studied in the literature [2,5,6,9,11] and the consensus problem is a closely related sub-problem of BA problem. The BA problem was first introduced in 1982 by Lamport who proposed a protocol to solve the fault tolerance problem in computer systems [5]. The consensus problem is one of the most important issues for designing a fault-tolerant distributed system [9]. Solving the consensus problem, many applications can be achieved [11]. Therefore, the consensus problem in a cloud computing environment is revisited to discuss its solution with malicious faulty components in this paper. The proposed protocol is named Generalized Consensus Protocol of Cloud Computing (GCPCC) can lead to a consensus between each correct node in a cloud computing environment.

The remainder of this paper is organized as follows. Section 2 discusses the related works. The basic assumption of consensus problem is illustrated in Section 3. The proposed protocol is shown in Section 4. Examples are given in Section 5. Section 6 gives the correctness and complexity of the GCPCC. Section 8 concludes this paper.

II. Related Work

Cloud computing is a new distributed system concept that has been implemented by businesses such as Google and Amazon [13]. Google provides various applications on their internet platform such as Gmail and YouTube [14]. In addition, Google provides free storage capacity with gigabytes for each user. The big and powerful Google search engine allows users to find multiple results from different file types on the Internet. In previous literature, the consensus problem has been solved in various network topologies. However, previous studies of the consensus problem [11] are not specifically address cloud computing to order the application of Internet. Hence, in this paper, the topology of a cloud computing environment is applied. Subsequently, the consensus problem with failure components in the topology of a cloud computing is discussed. Cloud computing is a new distributed system computing concept in which nodes are interconnected with the Internet; the network is assumed reliable and synchronous. Previous studies with network topology in the consensus problem that have not cloud computing environment suitable. Fig. 1 is the topology of cloud computing used in our study.

(1)The nodes in an A-Level group must receive the request from users of different types of applications. Therefore, the nodes of an A-Level group have higher computational capability than the nodes in a B-Level group. In addition, nodes in an A-Level group must compute enormous amounts data and can communicate with other nodes in the same group directly through transmission media (TM).

(2)Some nodes form groups in the B-Level group, where each group provides a specific application service. According to the properties of nodes, the nodes are clustered to group B_i where $1 \le i \le c_n$ and

c_n is the total number of groups in a B-Level group.

(3)For the reliable communication, multiple inter transmission media (ITM) are used to connect the nodes between an A-Level group and a B-Level group. In A-Level group, each node must forward the message to all nodes in the corresponding group of B-Level group.

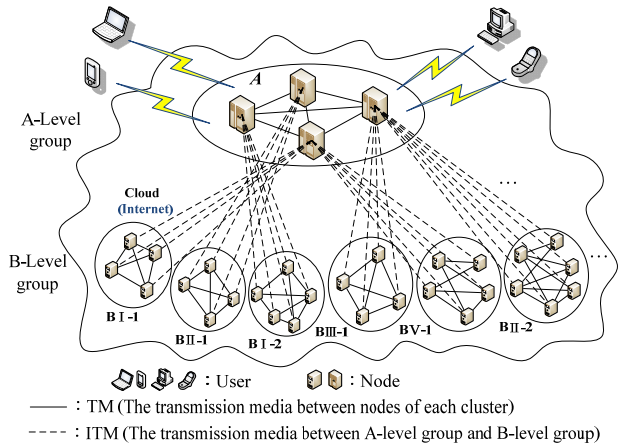


Fig. 1. Example of topology of cloud computing

Cloud Computing is a style of computing where massively scalable IT-related capabilities are provided to multiple external customers "as a service" using internet technologies [19]. The cloud providers have to achieve a large, general-purpose computing infrastructure; and virtualization of infrastructure for different customers and services to provide the multiple application services. The ZEUS Company has developed several types of software that can create, manage, and deliver exceptional online services from physical and virtual datacenters or from any cloud environment, such as ZXTM and ZEUS Web Server (ZWS) [17]. A cloud infrastructure virtualizes large-scale computing resources and packages them up into smaller quantities [18]. Furthermore, the ZEUS Company develops software that can let the cloud provider easily and cost-effectively offer every customer a dedicated application delivery solution [20]. The ZXTM software is much more than a shared load balancing service and it offers a low-cost starting point in hardware development, with a smooth and cost-effective upgrade path [20].

The ZEUS provided network framework can be utilized to develop new cloud computing methods, and is utilized in the current work. In this network composition that can support the network topology of cloud computing used in our research. According to the ZEUS network framework can testify the construction of network topology that we proposed in this paper, which let the company has to be considered. Hence, the proposed network topology of cloud computing that has the trustworthy example with the company provided to support our research.

III. The Basic Assumption Of Consensus Problem

In this paper, the consensus problem with faulty nodes and transmission media in cloud computing topology is revised. It requires a number of independent correct nodes to reach consensus when some of those components might be faulty. A distributed system can attain stable results without any influence from faulty components. However, in many cases, the faulty components will influence the system to reach agreement. Thus, the BA has been achieved if the following conditions are met [5]:

Agreement: All correct nodes agree on a common decision value.

Validity: If the source node is correct, then all correct nodes agree on the initial value sent by the source node.

The BA problem was defined as follows [5]:

(1) Of the n nodes (n>3) at most one-third of the total number of nodes can fail without breaking down a network system;
(2) Nodes communicate with each other through message exchange in a fully connected network;
(3) The message sender is identifiable by receiving nodes;
(4) A node is chosen as a source, and its initial value v_s is broadcasted to other nodes and itself to execute the protocol;
(5) In the system, faulty nodes only are considered.

A closely related sub-problem of the BA problem, the consensus problem, has been studied extensively in the literature [11]. In this paper, we will revisit the consensus problem in a cloud computing environment. The consensus problem requires a protocol to allow the components to exchange messages then the correct components are to achieve consensus agreement. Hence, the proposed protocol

we refer to as the Generalized Consensus Protocol of Cloud Computing (GCPCC). It can lead to an agreement of each correct component in a cloud computing topology. Lamport argues the consensus problem under the assumption of synchronous behavior in the BA problem [5] so that assumptions of the BA problem can be used to explain the consensus problem. The consensus problem has k initial values and subsequently achieves a common value even if certain transmission media and nodes fail. Therefore, the consensus problem is similar to the BA problem; such as in executing k copies of BA nodes. In this paper, the consensus problem for malicious faulty components in a cloud computing environment is solved. The proposed protocol of consensus defined with solutions, the goal of solving a consensus problem is to develop an optimal algorithm can use the minimal number of rounds and tolerate the maximal number of allowable faulty components to achieve consensus[11].

Achieving consensus on a same value in a distributed system, the protocols are required to provide that systems be executed correctly while achieving consensus on an identical value in a distributed system. In the consensus problem, many cases are based on the assumption of node failure in the general network [5]. Therefore, it is important to propose a protocol to solve the consensus problem [9] in a topology of cloud computing existence faulty components. In a cloud computing system all components have to execute a protocol that ensures that correct components reach consensus with the result that the system provides stable and reliable service for users. All nodes reach consensus, fault-tolerance capacity is enhanced even if there are faults with components in the topology. In this research, we solve the consensus problem for malicious fault transmission media and nodes in a cloud computing topology. However, the proposed protocol DFCCP is used to solve the problem of malicious fault components and allow all correct components to achieve consensus agreement. Fault-tolerance capacity is thus enhanced even if faulty transmission media and node in chorus exist in the topology.

IV. GCPCC For Cloud Computing

Cloud computing environment must be able to provide multiple services [10]. In this research, the consensus problem is revisit in cloud computing where faulty transmission media and nodes may influence normal operation in the system. In this paper, a new protocol called Generalized Consensus Protocol of Cloud Computing (GCPCC) is proposed to solve the consensus problem when caused by faulty components that may send incorrect messages that may in turn influence how the system reaches consensus in a cloud computing environment. If the number of faulty components can be known then the number of rounds required can be estimated to solve the consensus problem. For instance, if the faulty component is a node, then GCPCC can save some rounds required to remove the influence of faulty transmission media [11]. Conceptually, GCPCC removes the influence of faulty transmission media during exchange message by using a majority voting scheme because the major transmission media of the topology are assumed correct [11]. In the cloud computing topology, the main work of an A-Level group's nodes is collecting user requests. Each node in an A-Level group has to receive the various requests from users, while the nodes in a B-Level group's cluster provide many services for users. Hence, all nodes may receive different initial values different two level groups. The protocol GCPCC is executed by nodes in the A-Level and B-Level groups.

In a cloud computing environment, each node in A-Level group receives the various requests from users, and the nodes in B-Level group provide services to users. Each node of an A-Level group may receive a different service request from users. Each node in an A-Level group that uses the service request as the initial value executes the GCPCC to obtain the common vector DEC_A. Then, each node of the A-Level group forwards the element of vector DEC_A to the nodes in the B-Level group. However, the specific service request is to be conformed by the nodes of same group. Each node in the same cluster of a B-Level group receives the element from the nodes of the A-Level group. In the B-Level group, nodes may receive the fake value by the faulty transmission media and failure node in the A-Level

group. There the nodes in the B-Level group is receives the fake value from failure nodes of A-Level group through correct transmission media. Therefore, a failure node of the A-level group for nodes of a B-Level group is a fault transmission media. Therefore, the number of A-Level group's failure nodes and transmission media must be less than half with those components. Sequentially, each node in the same cluster of B-Level group has to take majority value of the received element values (DEC_A). Hence, the initial value for each node can be obtained in the same cluster in the B-Level group. Nodes in the same cluster of a B-Level group must exchange and receive the initial value with other nodes by executing the *Consensus Process*. Finally, each node takes a majority value of the vector value (DEC_B). Then the consensus value can be obtained by the GCPCC. GCPCC is invoked to solve the consensus problem with faulty transmission media and nodes in cloud computing. Based on the network topology of cloud computing, GCPCC can allow each node to transmit messages to other nodes without influence from faulty transmission media and nodes, the proposed protocol is shown in Fig. 2.

GCPCC

(1) The nodes of the A-Level group execute the *Consensus Process* (for the node i in the A-Level group with initial value v_i; $1 \le i \le n_A$) where n_A is the total number of nodes in the A-Level group.

(2) Each node of the cluster in the A-Level group sends the specific element of DEC_A to the nodes of a specific application having the cluster of the B-Level group.

(3) Each node k in the same cluster of a B-Level group takes a majority value MAJ_k ($1 \le k \le n_{Bj}$) of the received element, then the initial value v_k of each node k can be obtained.

(4) The nodes of the B-Level group's cluster execute the *Consensus Process* (for the node i in the cluster j of the B-Level group with common value v_i; $1 \le i \le n_{Bj}$).

(5) Each node of the same cluster in a B-Level group takes a majority value of DEC_B, and then the consensus value v is obtained.

Consensus Process(i, n, *X-Level group*)

Pre-Execute.

Compute the number of rounds required $\sigma = \lfloor (n-1)/3 \rfloor + 2$

***Message Gather Stage*:**

$r = 1$, do:

A) Each node i parallel broadcasts its initial value v_i to other nodes in the cluster of an X-Level group.

B) Each node receives and stores the n values sent from n nodes of the cluster in an X-Level group in the corresponding root of each mg-tree.

for $r = 2$; $r \le \sigma$;$\sigma + 2$;

C) Each node parallel transmits the values at level r - 1 in the corresponding mg-tree to other nodes in the cluster of an X-Level group.

D) Each node receives values from other nodes and stores them in level r of n corresponding mg-trees.

E) Call MAJ function.

***Decision Making Stage*:**

Step 1: Reorganize each mg-tree into a corresponding ic-tree by deleting the vertices with repeated node names.

Step 2: VOTE(i, n) function is paralleled to apply to the root of each corresponding ic-tree, then a vector DEC_X as a common value with n elements has been obtained.

Function VOTE(*i, n*)

1. val(i), if i is a leaf.
2. The majority value in the set of {VOTE (αi, n)|$1 \le i \le n$, and vertex αi is a child of vertex α}, if such a majority value exists.
3. A default value ϕ is chosen otherwise.

Function MAJ

Step 1: Count the received values and take the majority, then set a majority value x.

Step 2: If the majority value is not existed, then output a majority value ϕ.

Step 3: Otherwise, output a majority value x, where $x \in \{0,1\}$.

Fig. 2. The proposed protocol GCPCC

The node in B-Level group's cluster receives the initial value through the GCPCC. The *Consensus Process* of GCPCC requires $\sigma = \lfloor (n-1)/3 \rfloor + 2$ rounds to receive sufficient messages for A- and B-Level groups' nodes. In the first round of Message Gather Stage, each node parallel transmits its initial value to

other nodes in the same cluster and then receives the value and stores it at the r - 1 level of its mg-tree. The mg-tree is a tree data structure that is used to store the received messages [11]. Subsequently, each node in the same cluster transmits the received messages to other nodes and stores it at level r in its mg-tree. The function MAJ is applied on every two rounds in the leaf of mg-tree to take majority values to banish influence of the fault transmission media. In the Decision Making Stage of *Consensus Process*, each node reorganizes its mg-tree into a corresponding ic-tree. The ic-tree is a tree structure that is used to store a received message without repeated node names [11]. The function VOTE is applied to the root of each corresponding ic-tree to take the majority value, and then a vector DEC_A is obtained. Each element of DEC_A is mapped to a specific application that will be executed in the corresponding cluster of a B-Level group. Each node of the same cluster in the B-Level group takes a majority value as the initial value from the vector. Sequentially, each node in the same cluster which after execute the Message Gather Stage of *Consensus Process* and applied function MAJ to banish the fault transmission media, then each node reorganizes its mg-tree into a corresponding ic-tree and apply function VOTE to obtain the consensus value. Finally, all correct nodes in the same cluster are achieved the consensus value to reach agreement.

V. Example Of Executing GCPCC

An example of executing the GCPCC based on a cloud computing environment is shown in Fig. 3. In addition, an example of an A-Level group is shown in Fig. 4-1. The nodes in the A-Level group receive service requests. The protocol, for this example, requires four rounds ($\sigma = \lfloor (n_A - 1)/3 \rfloor + 2 = \lfloor (7-1)/3 \rfloor + 2 = 4$, where n_A is the number of nodes in the A-Level) to exchange the messages. Each node can obtain the initial value in the A cluster as shown in Fig. 4-2. The different requests are received from different users by each node, such as A1 receives the video service request and A5 receives the blog service request, etc. In every round of Message Gather Stage, there each node parallel transmits the initial value to all nodes in the same cluster and stores the received values in the corresponding root of the level 2 to 4 each mg-tree as shown in Figs. 4-3 to 6-6. Furthermore, the function MAJ is executed in level 2 and 4 as shown in Figs. 4-4 and 4.6. Subsequently, in the Decision Making Stage, the mg-tree is reorganized into the ic-tree by deleting the vertices with repeated node names as shown in Fig. 4-7. The function VOTE is applied on each corresponding ic-tree of all nodes and then taking the majority value of level 3 to 1. Eventually, the common vector value DEC_A is obtained for all nodes in the A-Level group as shown in Fig. 4-8.

All nodes in the cluster of the B-Level group receive the element DEC_A from the nodes of the A-Level group by multiple transmission media that the example as shown in Fig 5. All nodes in cluster B Ⅱ-1 of the B-Level group receive the element value DEC_A that transmits from the nodes in the A-Level group for the specific applications needing to be serviced. If the nodes in the A-Level group send the E-mail service request with elements of DEC_A to all nodes in cluster B Ⅱ-1. Subsequently, the elements of DEC_A can receives with each node in cluster B Ⅱ -1 receives the elements of DEC_A, and then takes a majority value as shown in Fig. 6.

The example of cluster B Ⅱ-1 in a B-Level group is presented in Fig. 7. In this example, there are eight nodes in cluster B Ⅱ -1 requiring four rounds ($\sigma = \lfloor (n_{Bj} - 1)/3 \rfloor + 2 = \lfloor (8-1)/3 \rfloor + 2 = 4$, where n_{Bj} is the number of nodes in cluster B Ⅱ-1 of the B-Level group). Fig. 8-1 presents each node's initial value. In the first round of Message Gather Stage, the node sends the initial value (=1) to other nodes and receives it from other nodes in same cluster as shown in Fig. 8-2. The node B3 to execute the second to four rounds of Message Gather Stage and the function MAJ is executed in level two to four as shown in Figs. 8-3 and 8-5. In the Decision Making Stage, the node B3's mg-tree is reorganized into the corresponding ic-tree as shown in Fig 8-6; and the function VOTE is applied on the ic-tree's root to take the majority value, and DEC_B (=1) is obtained as shown in Fig. 8-7. Hence, the consensus value has been obtained and all correct nodes reach consensus.

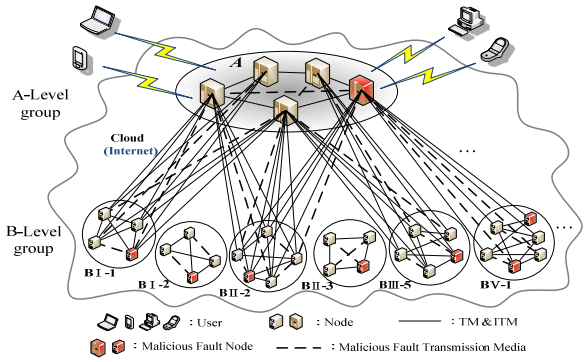


Fig. 3. An example of cloud computing environment

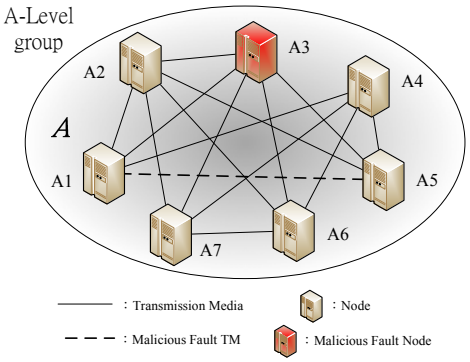


Fig.4-1.Example cluster A in A-Level group

A1	A2	A3	A4	A5	A6
0	1	0	1	0	1

Fig. 4-2. The initial value of each node in A cluster

level 0 *Root*	*level 1*	
A1	1	0
	2	1
	3	***1***
	4	1
	5	***1***
	6	1

level 0 *Root*	*level 1*	
A2	1	0
	2	1
	3	***1***
	4	1
	5	0
	6	1

level 0 *Root*	*level 1*	
A3	1	0
	2	1
	3	***0***
	4	1
	5	0
	6	1

level 0 *Root*	*level 1*	
A4	1	0
	2	1
	3	***1***
	4	1
	5	0
	6	1

level 0 *Root*	*level 1*	
A5	1	***1***
	2	1
	3	***1***
	4	1
	5	0
	6	1

level 0 *Root*	*level 1*	
A6	1	0
	2	1
	3	***0***
	4	1
	5	0
	6	1

Fig. 4-3. The mg-tree of each node in the A cluster at the 1st round

level 0 *Root*	*level 1*	*level 2*	
A1	Val(1)=0 MAJ ↑ (0,0,***1***,0,***1***,0)	11	0
		12	0
		13	***1***
		14	0
		15	***1***
		16	0
	Val(2)=1 MAJ ↑ (1,1,***0***,1,1,1)	21	1
		22	1
		23	***0***
		24	1
		25	1
		26	1
	Val(3)=***1*** MAJ ↑ (1,1,***0***,1,1,0)	31	1
		32	1
		33	***0***
		34	1
		35	1
		36	0
	Val(4)=1 MAJ ↑ (1,1,***0***,1,1,1)	41	1
		42	1
		43	***0***
		44	1
		45	1
		46	1
	Val(5)= 0 MAJ ↑ (***1***,0,***1***,0,0,0)	51	***1***
		52	0
		53	***1***
		54	0
		55	0
		56	0
	Val(6)=1 MAJ ↑ (1,1,***0***,1,1,1)	61	1
		62	1
		63	***0***
		64	1
		65	1
		66	1
	⋮	⋮	⋮

Fig. 4-4. The mg-tree of node A1 at the 2nd round

level 0 *Root*	*level 1*	*level 2*	*level 3*	
A1	Val(1)=0 MAJ ↑ (0,0,***1***,0,***1***,0)	Val(11)=0	111	0
			112	0
			113	***1***
			114	0
			115	***1***
			116	0
		Val(12)=0	121	1
			122	1
			123	***0***
			124	1
			125	1
			126	1
		Val(13)=1	131	1
			132	1
			133	***0***
			134	1
			135	1
			136	0
		Val(14)=0	141	1
			142	1
			143	***0***
			144	1
			145	1
			146	1
		Val(15)=0	151	***1***
			152	0
			153	***1***
			154	0
			155	0
			156	0
		Val(16)=0	161	1
			162	1
			163	***0***
			164	1
			165	1
			166	1
		⋮	⋮	⋮

Fig. 4-5 The mg-tree of node A1 at the 3rd round

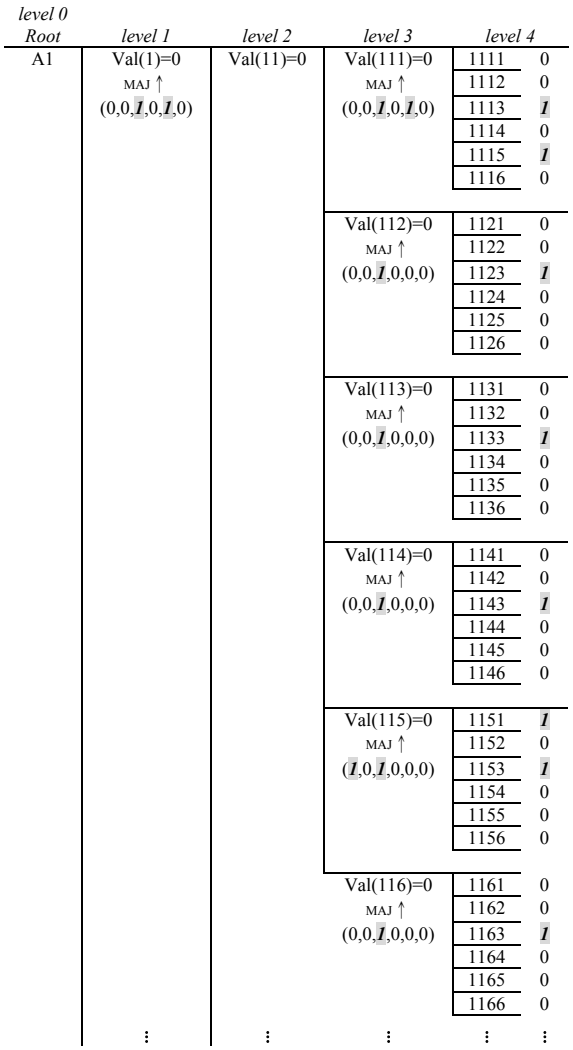
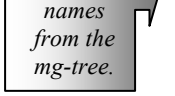


Fig. 4-6 The mg-tree of node A1 at the 4th round



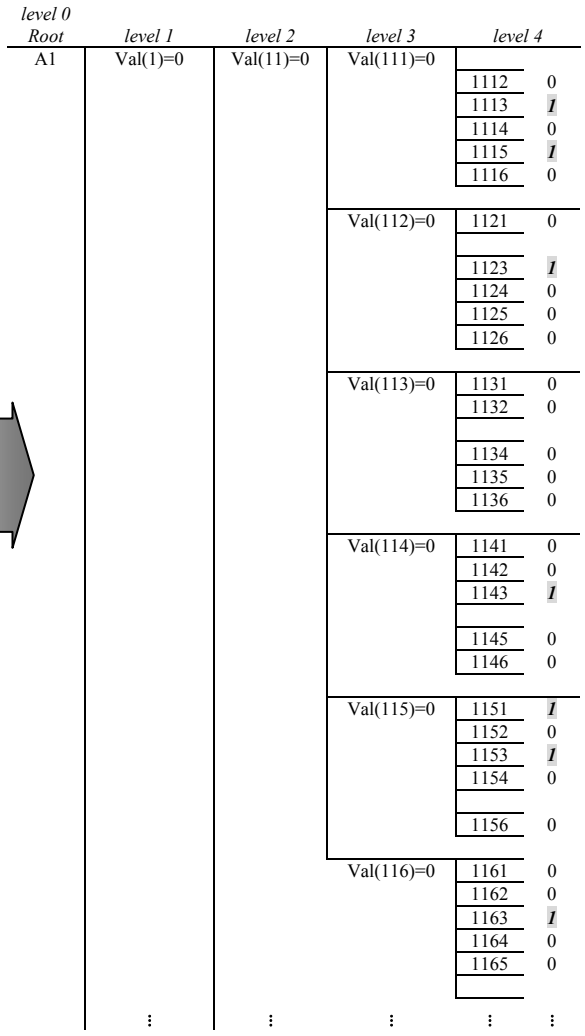


Fig. 4-7 The ic-tree of node A1 by the Decision Making Stage

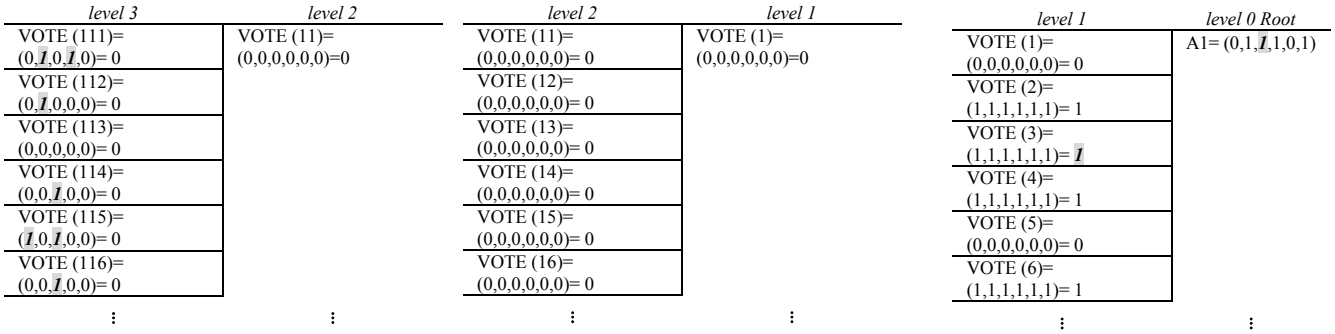


Fig. 4-8. The consensus value VOTE(γ) by node A1

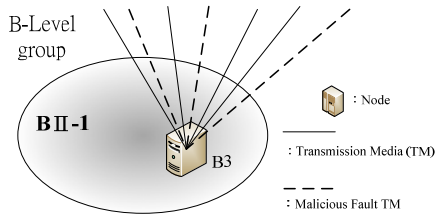


Fig. 5. The example for A-Level group nodes forward value to B-Level group's cluster

BⅡ-1-B1		BⅡ-1-B2		BⅡ-1-B3		⋯	BⅡ-1-B8	
A1	0	A1	0	A1	0	⋯	A1	0
A2	1	A2	1	A2	1	⋯	A2	1
A3	0	A3	0	A3	1	⋯	A3	0
A4	0	A4	0	A4	0	⋯	A4	0
A5	1	A5	1	A5	1	⋯	A5	1
A6	1	A6	1	A6	1	⋯	A6	1
A7	1	A7	1	A7	1	⋯	A7	1
MAJ_1=1		MAJ_2=1		MAJ_3=1		⋯	MAJ_8=1	

Fig. 6. Each node of BⅡ-1 cluster receive element of DEC_A from A-Level group node

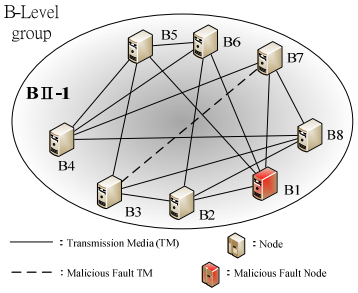


Fig. 7. Example of BⅡ-1 cluster in B-Level group

B1	B2	B3	B4
1	1	1	1

B5	B6	B7	B8
1	1	1	1

Fig. 8-1. The initial value of each node in BⅡ-1 cluster

level 0 Root	level 1	
Val(B1) =1	1	**0**
	2	1
	3	1
	4	1
	5	1
	6	1
	7	1
	8	1

level 0 Root	level 1	
Val(B2) =1	1	**0**
	2	1
	3	1
	4	1
	5	1
	6	1
	7	1
	8	1

level 0 Root	level 1		
Val(B3) =1	1	**0**	⋯
	2	1	⋯
	3	1	⋯
	4	1	⋯
	5	1	⋯
	6	1	⋯
	7	**0**	⋯
	8	1	⋯

Fig. 8-2. The mg-tree of each node in BⅡ-1 cluster at the 1st round

level 0 Root	level 1	Level 2	
B3	Val(1)=1 MAJ ↑ (**0** … 1)	11	**0**
		12	1
		13	1
		14	1
		15	1
		16	1
		17	1
		18	1
	Val(2)=1 MAJ ↑ (**0** … 1)	21	**0**
		22	1
		23	1
		24	1
		25	1
		26	1
		27	1
		28	1
	⋮	⋮	⋮

Fig. 8-3. The mg-tree of node B3 at the 2nd round

level 0 Root	level 1	Level 2	level 3	
B3	Val(1)=1 MAJ ↑ (**0** … 1)	Val(11) =0	111	**0**
			112	1
			113	1
			114	1
			115	1
			116	1
			117	1
			118	1
		Val(12) =1	121	**0**
			122	1
			123	1
			124	1
			125	1
			126	1
			127	1
			128	1
	⋮	⋮	⋮	⋮

Fig. 8-4.The mg-tree of node B3 at the 3rd round

level 0 Root	level 1	level 2	Level 3	level 4	
B3	Val(1)=1 MAJ ↑ (**0** … 1)	Val(11)=0	Val(111)=1 MAJ ↑ (**0** … 1)	1111	**0**
				1112	1
				1113	1
				1114	1
				1115	1
				1116	1
				1117	1
				1118	1
			Val(112)=1 MAJ ↑ (**0** … 1)	1121	**0**
				1122	1
				1123	1
				1124	1
				1125	1
				1126	1
				1127	1
				1128	1
	⋮	⋮	⋮	⋮	⋮

Fig. 8-5.The mg-tree of node B3 at the 4th round

The ic-tree erased the vertices with repeated names from the mg-tree.

level 0 Root	level 1	level 2	Level 3	level 4	
B3	Val(1)=1	Val(11)=0	Val(111)=1	1112	1
				1113	1
				1114	1
				1115	1
				1116	1
				1117	1
				1118	1
			Val(112)=1	1121	**0**
				1123	1
				1124	1
				1125	1
				1126	1
				1127	1
				1128	1
	⋮	⋮	⋮	⋮	⋮

Fig. 8-6. The ic-tree of node B3

level 3	level 2
VOTE (111)= (1,1,1,1,1,1,1)= 1	VOTE(11) = (1,1,1,1,1,1,1,1)=1
VOTE (112)= (0,1,1,1,1,1,1)= 1	
VOTE (113)= (0,1,1,1,1,1,1)= 1	
⋮	⋮

level 2	level 1
VOTE(11)= (1,1,1,1,1,1,1,1)=1	VOTE (1) =(1,1,1,1,1,1,1,1)=1
VOTE(12)= (1,1,1,1,1,1,1,1)=1	
VOTE(13)= (1,1,1,1,1,1,1,1)=1	
⋮	⋮

level 1	level 0 Root
VOTE (1)= (1,1,1,1,1,1,1,1)= 1	VOTE (B3)= (1,1,1,1,1,1,1,1)=1
VOTE (2)= (1,1,1,1,1,1,1,1)= 1	
VOTE (3)= (1,1,1,1,1,1,1,1)= 1	
⋮	⋮

Fig. 8-7. The consensus value by node B3

VI. THE CORRECTNESS AND COMPLEXITY

According to the literature, a protocol is obtained and the following proofs for the agreement and validity property are given in this section. The following lemmas and theorems are used to prove the correctness and complexity of the GCPCC. The notations and parameters of GCPCC are shown as follows:

- n: The number of nodes in the cloud computing environment.
- TM_{ij}: The transmission medium between node i and node j.

- ITM: The transmission media between A-level group and B-level group.
- c: The connectivity of network topology.
- n_A: The number of nodes in an A-Level group.
- n_{Bj}: The number of nodes in cluster j of a B-Level group.
- C_n: The total number of clusters in a B-Level group.
- TM_{mA}: The number of allowable malicious faulty transmission media in an A-Level group.
- TM_{mBj}: The number of allowable malicious faulty transmission media in the cluster j of a B-Level group.
- N_{mA}: The number of allowable malicious faulty nodes in an A-Level group.
- N_{mBj}: The number of allowable malicious faulty nodes in the cluster j of a B-Level group.
- TM_m: The number of allowable malicious faulty transmission media in the cloud computing environment.
- N_m: The number of allowable malicious faulty nodes in the cloud computing environment.
- T_f: The total number of allowable faulty nodes.
- ITM_c: The connectivity with each node of a B-Level group between an A-Level group.
- σ: The number of rounds required in the *Consensus Process*.

A. Correctness of GCPCC

To prove that vertex α is common, the term common frontier [2] is defined as follows: When every root-to-leaf path of the mg-tree contains a common vertex, the collection of the common vertices forms a common frontier. In addition, the constraints, Agreement and Validity, can be rewritten as:

- *Agreement*: Root i is common
- *Validity*: VOTE(i) = v_i for each correct node, if the node i is correct

Every correct node has the same values collected in the common frontier if a common frontier does exist in a correct node's mg-tree. Subsequently, using the same function VOTE to compute the root value of the tree structure, every correct node can compute the same root value because the same input (the same collected values in the common frontier) and the same computing function will produce the same output (the root value). Since GCPCC can solve the BA problem, the correctness of GCPCC should be examined in the following two ways [2].

(1) Correct vertex: Vertex αi of a tree is a correct vertex if node i (the last node name in vertex αi's node name list) is correct. In other words, a correct vertex is a place to store the value received from a correct node.

(2) True value: For a correct vertex αi in the tree of a correct node, val(αi) is the true value of vertex αi if TM_{ij} is fault-free. In other words, a correct vertex is a place to store the value received from a correct node. In other words, the stored value is called the true value of a vertex if the value stored in such a vertex is correct from the influence of a faulty TM.

Theorem 1. Each node can receive values without influences of malicious faulty nodes between the sender node via GCPCC in each round, then $n_A \geq 3N_{mA}+1$ in A-Level group and $n_{Bj} \geq 3N_{mBj}+1$ in cluster j of B-Level group.

Proof: We can ignore the influences of the malicious faulty nodes between any pairs of nodes in each round of value exchange and $n_A \geq 3N_{mA}+1$ in A-Level group; and $n_{Bj} \geq 3N_{mBj}+1$ in cluster j of B-Level group. The reason is that the correct sender node n_A (n_{Bj}) copies of a value to correct destination nodes. In the worst case, a correct destination node can receive n_A-N_{mA} values transmitted via the correct sender node in A-Level group; receive n_{Bj}-N_{mBj} values transmitted via the correct sender node in the cluster j of B-Level group.

Lemma 1. In an ic-tree, all correct vertices are common.

Proof: The tree structure has conversed from mg-tree to ic-tree. At the level σ or upon of ic-tree, the correct vertex i has at least 2σ-1 children, in which at least σ children are correct. The real value of these σ correct vertices is common, and the majority value of vertex α is common. For this reason, all correct vertices of the ic-tree are common.

Lemma 2. The common frontier exists in the ic-tree.

Proof: There are σ vertices along each root-to-leaf path of an ic-tree, so that though most σ-1 nodes have failed, at least one vertex is correct along each

root-to-leaf path of the ic-tree. The correct vertex is common, and the common frontier exists in each correct node ic-tree by Lemma 1.

Lemma 3. Let α be a vertex, and α is common if there is a common frontier in the sub-tree rooted at *i*.

Proof: When the height of α is 0, and the common frontier exists, α is common. If the height of α is σ, the children of α are all in common by induction hypothesis with the height of the children at σ-1. Then the vertex α is common.

Corollary. If the common frontier exists in the ic-tree, then the root is common.

Theorem 2. The root of a correct node's ic-tree is common.

Proof: By Lemmas 1, 2, 3 and the Corollary, the theorem is proved.

Theorem 3. Protocol GCPCC solves the consensus problem in a cloud computing environment.

Proof: Inasmuch as the theorem must describe that GCPCC meets the constraints *Agreement'* and *Validity'*.

Agreement: Root *i* is common, and by Theorem 3, *Agreement'* is satisfied.

Validity: VOTE(i) = v_i for each correct node, if the initial value of the node i is v_i.

Whereas most of nodes are correct, the nodes use GCPCC to communicate with all others. The message of correct vertices for all correct nodes' mg-trees is v_i. When the tree structure has converted from mg-tree to ic-tree, the correct vertices still exist. Therefore, every correct vertex of the ic-tree is common (refer to Lemma 1), and its true value is v_i. This root is common by Theorem 2. The computed value VOTE(i)=v_i is stored in the root of the ic-tree for all correct nodes. (Validity') is satisfied.

B. Complexity of GCPCC

The complexity of GCPCC is evaluated in terms of: 1) the maximum number of allowable faulty components; and 2) the minimum number of rounds to exchange messages. Theorems 4 and 5 show that the optimal solution is reached.

Theorem 4. The number of allowable faulty nodes N_m is maximal and $c \geq 2TMij+1$.

Proof: Over the past literature of the agreement problem for node faults only as $n \geq 3N_m+1$ and $c \geq 2TMij+1$. However, we also assume that components are faulty. Hence, the constraint of the maximum number of allowable faulty nodes can be applied to our study.

In a cloud computing environment, GCPCC can tolerate $N_{mA}+TM_{mA} \leq (n_A-1)/3$ faulty components in the A-Level group and the fault tolerant capability of the B-Level group is $\max_{j=1}^{C_n}(3N_{mBj}+TM_{mBj} \leq (n_{Bj}-1)/3)$. The connectivity with the A-Level group and each cluster of B-Level group is $c \geq 2TMij+1$; the node in the same cluster of B-Level group with the connectivity is $c \geq 2TMij+1$ and the malicious fault ITM at most $\lceil c/2 \rceil-1$. The total number of allowable faulty nodes by GCPCC is $T_f \leq [(N_{mA}+TM_{mA})+ \max_{j=1}^{C_n}(3N_{mBj}+TM_{mBj})+ (\lceil ITM_c/2 \rceil-1)$, and the number of faulty components is maximal in topology of cloud computing.

Theorem 5. GCPCC requires $[(\lfloor (n_A-1)/3 \rfloor+2)+ (\max_{j=1}^{C_n} \lfloor (n_{Bj}-1)/3 \rfloor+2)]+1$ rounds of message exchange to solve the consensus in a cloud computing environment.

Proof: The value passing is required only in the Message Gather Stage, it is time consuming. Yan et al. [11] pointed out that $\lfloor (n-1)/3 \rfloor+2$ rounds are the minimum number of rounds to send sufficient values to achieve consensus in an n-nodes distributed system [11]. In a cloud computing environment, each node needs to exchange messages with other nodes. Therefore, the constraint of the minimum number of rounds can be applied in the study. However, in a cloud computing environment, $\lfloor (n_A-1)/3 \rfloor+1$ rounds of exchange messages in A-level group is required and $\max_{j=1}^{C_n} \lfloor (n_{Bj}-1)/3 \rfloor+1$ rounds of exchange messages in B-level group is required. In addition, each node in the same cluster of the B-Level group needs to receive messages from A-Level group's nodes; therefore, one extra round is required. In conclusion, the minimum number of rounds is $[(\lfloor (n_A-1)/3 \rfloor+1)+ (\max_{j=1}^{c_n} \lfloor (n_{Bj}-1)/3 \rfloor+1)]+1$. Moreover, the number of rounds required is optimal.

VII. Conclusion

Cloud computing is a new concept of distributed systems [1,10]. It has greatly encouraged distributed system design and practice to support user-oriented services with application [1,16,18]. Fault-tolerance is an important research topic in the study of distributed systems and it is a

fundamental problem in distributed systems; there are many relative literatures in the past [2,5,9,11]. According to previous studies, network topology plays an important role in the consensus problem, but the results cannot cope with a cloud computing environment and the consensus problem thus needs to be reinvestigated. Moreover, in this paper, the consensus problem with faulty nodes and transmission media in a cloud computing topology has been solved by the proposed protocol. The proposed GCPCC ensures that all correct nodes in the topology of a cloud computing can reach a common value. Moreover, the new protocol GCPCC is adapted to the cloud computing environment and can derives the bound of allowable faulty components. GCPCC uses the minimum number of rounds of message exchange and tolerates the maximum number of allowable malicious fault transmission media and nodes in a cloud computing environment. Furthermore, the fault-tolerance capacity is enhanced by GCPCC.

ACKNOWLEDGMENT

This work was supported in part by the Taiwan National Science Council under Grants NSC96-2221-E-324-021 and NSC97-2221-E-324–007 -MY3.

REFERENCE

[1] F.M. Aymerich, G. Fenu and S. Surcis, "An Approach to a Cloud Computing Network," the First International Conference on the Applications of Digital Information and Web Technologies, pp. 113-118, Aug. 2008.

[2] M. Fischer and N. Lynch, "A Lower Bound for the Assure Interactive Consistency," Information Processing Letters, Vol. 14, No.4, pp. 183-186, 1982.

[3] R.L. Grossman, Y. Gu, M. Sabala and W. Zhang, "Compute and Storage Clouds Using Wide Area High Performance Networks," Future Generation Computer Systems, Vol. 25, No. 2, pp. 179-183, Feb. 2009.

[4] F. Halsall, Data Links, Computer Networks and Open Systems. 4th ed., Addison-Wesley Publishers, pp. 112-125, 1995.

[5] L. Lamport, et al., "The Byzantine General Problem," ACM Transactions on Programming Language and Systems, Vol. 4, No. 3, pp. 382-401, Jul. 1982.

[6] H.S. Siu, Y.H. Chin and W.P. Yang, "A Note on Consensus on Dual Failure Modes," IEEE Transactions on Parallel and Distributed Systems, Vol. 7, No. 3, pp. 225-230, 1996.

[7] M.A. Vouk, "Cloud Computing- Issues, Research and Implementations," Information Technology Interfaces, pp. 31-40, Jun. 2008.

[8] L.H. Wang, J. Tao and M. Kunze, "Scientific Cloud Computing: Early Definition and Experience," the 10th IEEE International Conference on High Performance Computing and Communications, pp. 825-830, 2008.

[9] S.C. Wang, K.Q. Yan, S.S. Wang and G.Y. Zheng, "Reaching Agreement Among Virtual Subnets in Hybrid Failure Mode," IEEE Transactions on Parallel and Distributed Systems, Vol. 19, No. 9, pp. 1252-1262, Sep. 2008.

[10] A. Weiss, "Computing in The Clouds," netWorker, Vol. 11, No. 4, pp. 16-25, 2007.

[11] K.Q. Yan, Y.H. Chin and S.C. Wang, "Optimal agreement protocol in malicious faulty processors and faulty links," IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 3, pp. 266-280, Jun. 1992.

[12] "Amazon.com: Online Shopping for Electronics, Apparel, Computers, Books, DVDs & more," http://www.amazon.com/, Jul. 2009.

[13] "Application Delivery Networking, Application Acceleration, Internet Traffic Management System : Zeus.com," http://www.zeus.com/, Jul. 2009.

[14] "Application Traffic Management, Application Security," http://www.zeus.com/ products/zxtm/index.html, Jul. 2009.

[15] "Cloud Computing," http://www.zeus.com/ cloud_ computing/, Jul. 2009.

[16] "Gartner Says Cloud Computing Will Be As Influential As E-business," http://www.gartner. com/it/page.jsp? id=707508, Jul. 2009.

[17] "Load Balancing, Load Balancer," http://www.zeus.com/products/zxtmlb/ index. html, Jul. 2009.

[18] "More Google Product," http://www.google. com/options/, Jul. 2009.

[19] "What is Cloud Computing?," http://www. zeus.com/cloud_computing/cloud.html, Jul. 2009.

[20] "ZXTM for Cloud Hosting Providers," http://www.zeus.com/cloud_computing/for_cl oud_providers.html, Jul. 2009.