Pakistan Journal of Applied Sciences 3 (7): 493-509, 2003
ISSN 1607-8926


# Generalized Agreement Underlying a Multicasting Environment

S.C. Wang and [1]K.Q. Yan
Department of Information Management, [1]Department of Business Administration,
Chaoyang University of Technology, 168 Gifeng E. Rd., Wufeng, Taichung County,
Taiwan, 413, R.O.C.

**Abstract:** The BA problem is solved in a multicasting environment. The proposed protocol uses the minimum number of message exchanges to reach an agreement while tolerating the maximum number of faulty components in the distributed system. It makes all the fault-free processors reach a common value to keep the system from the influences of processor failures and transmission medium failures.

**Key words:** Byzantine agreement, distributed system, distributed architecture, fault-tolerance and multicasting network

## Introduction

Traditionally, if some information is to be delivered from a processor to other n processors in a Fully Connected Network (FCN), the source needs to send out n copies of it. The FCN is considered inefficient because such a way of delivery costs too much of the bandwidth. In a broadcasting Network (BCN), the message is spread through the local bus. Although the BCN consumes less bandwidth, regardless of the messages' destinations, all the processors connected to the bus have to be alert for any messages on-line. It brings a heavy load onto the CPU resources and thus decreases the processors' efficiency. In addition, the scope of information dissemination is limited to a local bus. In a multicasting network, a frame can be sent to an individual, broadcasters, or a group address in a LAN (Halsall, 1995). Distinct from the FCN, only one copy of the message needs to be sent over the Internet. Therefore, the multicasting network is more powerful and applicable than the traditional FCN or the BCN. Some well-known network systems are in fact applications (Wang ***et al.***, 1982) of the multicasting network; e.g. a modern advanced avionic network may include a fire control system, a flight path control system, a communication/navigation control system, a mission planning system and a radar system. In this complex distributed system, each subsystem allocates a number of CPUs or computers to achieve the stability and reliability. That is, each subsystem is equipped with 3 or 4 processors to increase the reliability of the software/hardware. In addition, each subsystem (or called the group of processors) needs to carry out an individual task, based on some certain common initial value, such as the start time and the initial geographical location. In these applications, if some faulty components exist, the system may fail due to the influence of the faulty components. It is a critical problem and to the best of our knowledge, there is yet no solution to this problem. In this paper, a novel approach for solving this unanimity problem is to be proposed.

Reaching an agreement in the presence of system errors plays an important role in designing a fault-tolerant distributed system. Such a problem is called the Byzantine Agreement (BA) (Dolev, 1982, Dolev ***et al.***, 1985; Lamport ***et al.***, 1982; Lamport ***et al.***, 1984; Pease ***et al.***, 1980; Reischuk 1982, Strong ***et al.***, 1982) problem. The goal of the BA is to reach a common value among fault-free processors even if certain components fail. The source has its initial value $v_s$ to start with. The agreement is reached if all the fault-free processors agree upon a common value v. That is, the BA problem is solved when the following constraints are satisfied:

($BA_1$): All fault-free processors agree on a common value v.

($BA_2$): If the initial value of the source is $v_s$ and the source is fault-free, then all fault-free processors shall agree on the value $v_s$; i.e., $v = v_s$.

With the agreement, many applications can be achieved, such as a two-phase commitment to be made in a distributed database system (Molina, 1986), a task of locating the whereabouts of a replicated file in a distributed environment (Gifford, 1979) and a landing task controlled by the processors in a flight control system (Yan ***et al.***, 1999).

Previously, the agreement problem has been discussed in the FCN, the BCN, or the Generalized Connected Network (GCN). Lamport ***et al.*** (Lamport ***et al.***, 1982) first brought up the BA problem with the following assumptions: (1) In an n-processor system, it tolerates at most $f_p$ faulty processors. (2) The processors communicate with each other directly through the transmission medium; that is, the network architecture is fully connected. (3) The message sender is always identified; in other words, the receiver is aware of the source of the message. (4) Only the symptom of some processor failure is discussed. It requires $f_p+1$ ($f_p \leq \lfloor (n-1)/3 \rfloor$) rounds of message exchange to reach a common value and the system tolerates $\lfloor (n-1)/3 \rfloor$ faulty processors. Fischer (Fischer ***et al.***, 1985) further proved that $\lfloor (n-1)/3 \rfloor$ faulty processors with $f_p+1$ rounds is the optimal solution. Bar-Noy (Bar-Noy ***et al.***, 1987) proposed a data structure to solve the BA problem. In a generalized case (where both processor failures and transmission medium failures may exist), (Yan ***et al.***, 1999) showed an optimal solution to making the fault-free processors reach an agreement by $f_p+2$ rounds of message exchange in the generalized FCN. In the BCN, Babaoglue (Babaoglue ***et al.***, 1985) has proved that the agreement can be reached at the cost of only 2 rounds if the number of faulty processors is less than half of n.

To improve the applicability of the traditional FCN and the BCN, Wang ***et al.*** (1985) and Siu ***et al.*** (1996) have revised network models respectively. Wang ***et al.***, introduced a GCN model, in which the concept of a group was first employed. It partitions n processors into g groups and each one of them contains p processors. Each group communicates with the others directly. That is, Wang's GCN requires a group-to-group fully connected environment. On the other hand, Siu ***et al.*** (1996) gave another GCN model, similar to the FCN, which allows the network with a bounded connectivity, called the c-connectivity (c is a constant). It is connected in a pure processor-to-processor manner and each pair of processors does not need to be connected.

The GCN models proposed by Wang or Siu were elegant, but they are still not generalized due to some limitations. For instance, Wang's GCN must be built on a fully connected environment. It is hardware wasting and inapplicable. On the other hand, Siu's GCN states the processor-to-processor relationship; it does not seem to go with the LAN or WAN environment. To offer a better solution, in this paper, we shall review the BA problem in a popular multicasting network.

As for the faulty components, they are classified as either processor failures or transmission medium failures. The types of processor failures include crash fault, omission fault and malicious fault; the types of transmission medium failures are crash fault, stuck-at fault and malicious fault (Yan ***et al.***, 1999). Generally, both faulty processors and faulty transmission media may exist. In the generalized case, as many as nine failure types may exist because of the interaction between the three types of processor failures and the three types of transmission medium failures. The nine possible types of faults are: crash-crash, crash-sa (Stuck-at), crash-malicious, omission-crash, omission-sa, omission-malicious, malicious-crash, malicious-sa and malicious-malicious. The part of the word before the hyphen indicates the faulty processor type and the part of the word after the hyphen denotes the faulty transmission medium type. For example, the term "crash-malicious" indicates that the processor is in the symptom of crash fault and the transmission medium is in the failure type of malicious fault. The symptom of a malicious fault is unpredictable and the behaviors of the other failure types can be treated as special cases of a malicious fault. If the malicious-malicious case, which is the thorniest case, can be solved, then the other types can surely be solved.

In short, the goal of the BA, based on the assumption that the source has its own initial value, is to make all the fault-free processors reach a common value. The proposed protocol solves the BA problem in a multicasting network in the presence of generalized component failures. The rest of this paper is organized as follows. Section 2 will serve to define the multicasting network model. Then, in Section 3, we shall illustrate out MAP (Multicasting Agreement Protocol) protocol in detail. Finally, the conclusion and direction of future work will be presented in Section 4.

**The SGCN model**

In this section, a multicasting network model, called the SGCN (Super Generalized Connected Network), is to be defined. It combines group-to-group connection (Cristian, 1991) and loose constraints on the connectivity: (1) A system with n processors can be partitioned into g groups, where $1 \leq g \leq n$. The number of processors, $1 \leq p \leq n$, in each of the groups can be varied. (2) The transmission medium is freely connected. The connectivity does not necessarily have to reach each pair of groups. In other words, the connectivity c can be varied. Fig. 1 illustrates an SGCN model, in which 15 processors are partitioned into 6 groups. There are 2 processors in the group $G_1$, 3 in $G_2$, 1 in $G_3$ and so on. As for the connectivity, for example, $G_1$ is connected to $G_3$, $G_4$ and $G_5$, but there is no direct link between $G_1$ and $G_2$ as well as $G_6$. $G_2$ is connected to $G_3$, $G_4$ and $G_5$, but it has no direct link to $G_1$ and $G_6$. Consequently, c = 3.

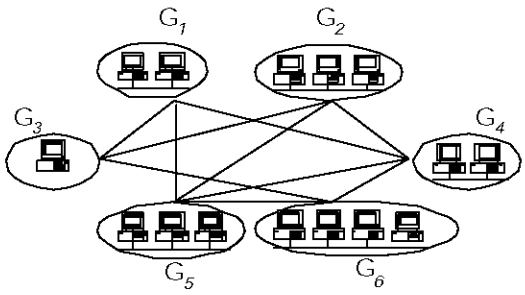



Fig. 1: An example of an SGCN (n = 15, g = 6, p = {1, 2, 3, 4} and c = 3)

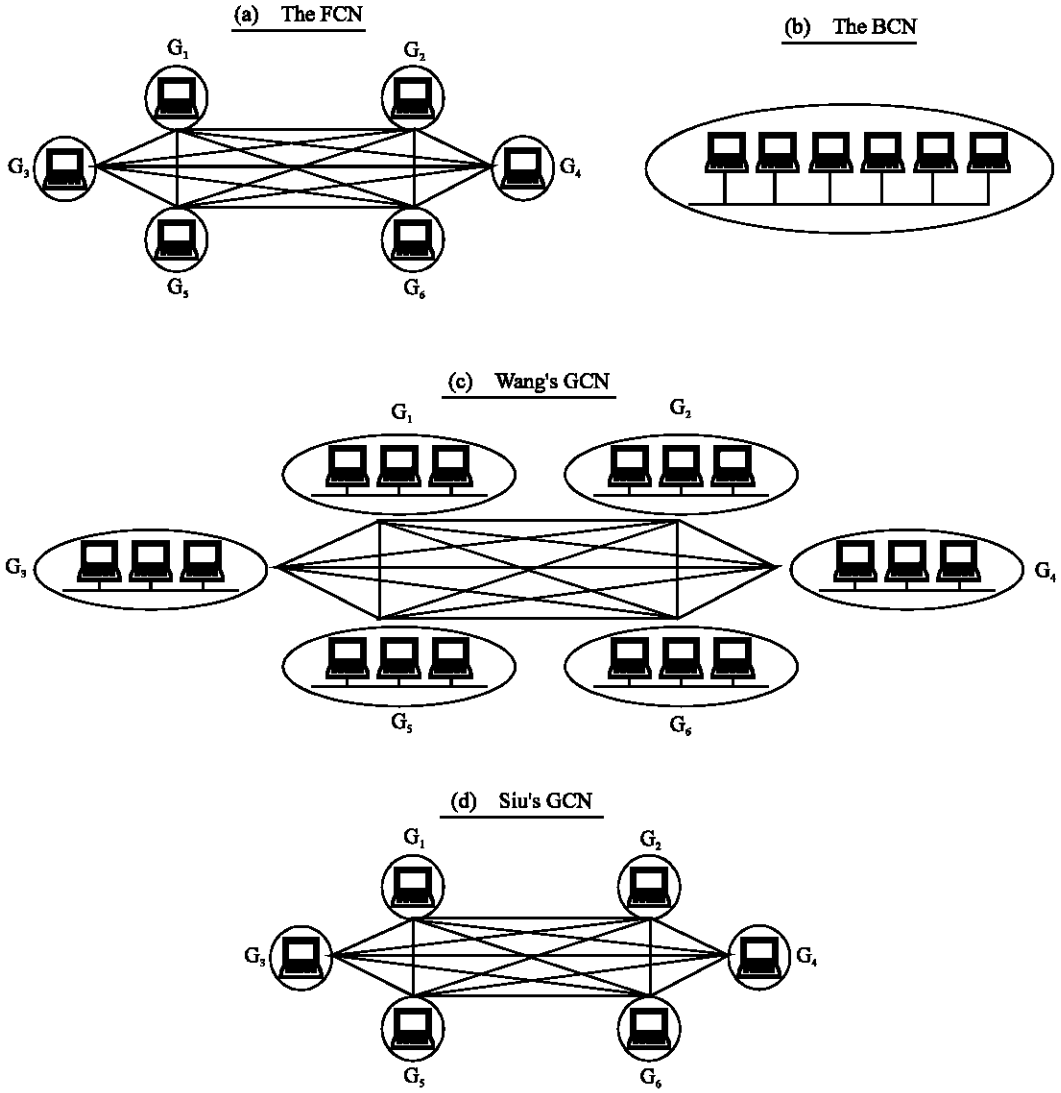



Fig. 2: Various SGCN models

The traditional FCN, the BCN and Wang or Siu's GCN are special cases of the SGCN: (1) An n-processor FCN can be treated as an n-group SGCN where g = n, p = 1 and c = n-1. Fig. 2(a) shows a 6-processor FCN, which is in fact an SGCN with n = 6, g = 6, p = 1 and c = 5. (2) The BCN includes packet terrestrial radio, satellite network, bus local network and ring local network (Lamport ***et al.***, 1984). For simplicity, Fig. 2(b) illustrates an example bus local network as well as an SGCN with n = 6, g = 1, p = 6 and c = 1. Due to the fact that the processors communicate with each other through a local bus, the BCN can be regarded to as a 1-group SGCN with c = 1. (3) Wang's GCN works as an SGCN with g groups while c = g-1. Fig. 2© is an example of Wang's GCN and is also an SGCN with n = 18, g = 6, p = 3 and c = 5. (4) In addition, an n-group, c-connectivity SGCN coincides with Siu's GCN presented in Fig. 2(d), in which n = 6, g = 6, p = 1 and c = 4. According to these observations, if the BA problem can be solved in the SGCN, then the BA problem can surely be reached in the FCN and the BCN as well as the GCNs proposed by Wang and Siu separately.

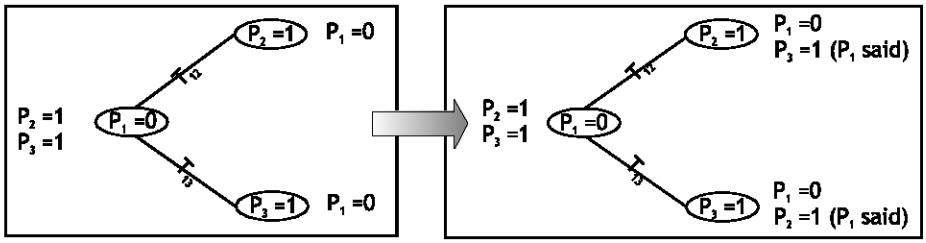
To the best of our knowledge, no currently existing protocols are capable of solving the BA problem in an SGCN. The SGCN has loose restrictions on the network architecture, is generalized and can practically be widely applied to many fields. The agreement problem mostly happens in the network environment and leads to a critical issue. If the BA is reached in an SGCN, then our purposed protocol MAP for the SGCN can take care of the problem for the FCN, the BCN and the GCN as well since the traditional FCN, BCN and GCN are in fact subsets of the SGCN. The concept and protocol to solve the BA problem in the generalized case will be discussed right next.

**Concept and protocol**

Conceptually, there are two phases in this protocol: the message exchange phase and the decision making phase. In order for all the fault-free processors to reach an agreement, each of them needs to collect enough agreement values from all the others if they are fault-free. As a result, exchanging the received values helps fault-free processors to collect enough agreement values. The received messages are stored in a tree structure called the message-gathering tree (mg-tree), which is similar to what Bar-Noy ***et al.*** (1987) proposed. This is the basic concept of the message exchange phase. Besides, a local majority (LMAJ) at the end of each round can also help reduce some following rounds since the processors are partitioned into groups in an SGCN. The LMAJ of the set $\{v_1, ..., v_n\}$ is v if the number of v's presented in the set is greater than n/2; otherwise, a default value $\phi$ is chosen. For example, the LMAJ of the set {0, 0, 0, 1, 1} is 0 and that of the set {1, 0, 1, 0, 0, 1} is $\phi$. In addition, the influence of a faulty transmission medium is removed during the message exchange phase by applying a global majority function, called the MAJ. After removing the influence of the faulty transmission medium, the mg-tree is reorganized into an information-collecting tree (ic-tree) and another global majority, called the VOTE, is applied for removing the influence of a faulty processor. The details of such functions as the MAJ and the VOTE will be given later.

Since all the fault-free processors reach a common value, the first step of the protocol is to determine the number of rounds of message exchange required. If the faulty component is identified, then the protocol can decide the minimum number of rounds required to solve the

BA problem. For example, if the faulty component is a processor, then the protocol can save some rounds required for removing the influence of a faulty transmission medium because there is not any. However, if there are both a faulty processor and a faulty transmission medium, then the number of rounds required will be bigger. The protocol works efficiently. During each round of message exchange, all the fault-free processors communicate with each other through the transmission media directly. In an SGCN, one may not be fully connected to all the others, but it can still communicate with them smoothly. For instance, in Fig. 3, processor $P_2$ is not directly connected to $P_3$. During the first round of message exchange, $P_1$ communicates with $P_2$ and $P_3$ through the transmission media $T_{12}$ (connected to $P_1$ and $P_2$) and $T_{13}$ (connected to $P_1$ and $P_3$) separately. $P_1$ gets $P_2$'s value as well as $P_3$'s value. $P_2$ gets $P_1$'s value, but not $P_3$'s. Similarly, $P_3$ gets $P_1$'s value, but not $P_2$'s. In the next round, $P_1$ asks $P_2$ what $P_3$'s value is and $P_2$ gets $P_3$'s value through $P_1$. Enough information helps making correct decisions. Then, the decision is made by a function VOTE executed by each processor in the decision making phase.




(Though $P_2$ is not directly connected to $P_3$, $P_2$ can still get $P_3$'s value by message exchanging.)
Fig. 3: The concept of message exchange

The message exchange phase is time consuming. Therefore, to reduce the number of required rounds is the major concern in designing the protocol. If the condition of the network is given, the number of required rounds is a minimum. For a given network healthy condition, there are four possibilities: (1) all components are fault-free; (2) transmission medium failure only; (3) processor failure only; (4) both processor failures and transmission medium failures exist. They are elaborated in the following passages.

**Case 1: All components are fault-free.**

Such a case is obviously the simplest one. The processors are fault-free and the messages are not forged. The transmission media are perfect and the messages are delivered correctly. Without facing any obstacle, each processor reaches the agreement in the first round. With only one round, each processor receives one value and agrees on the value at the end of the decision making phase while the network topology is fully connected; otherwise, two rounds of message exchange are needed in a c-connectivity ($c < n-1$) network. If it is not fully connected, some processors may not receive the value in the first round and thus another round of message exchange is needed. Since each processor will definitely receive values in the second round, the

agreement can surely be reached in two rounds. To sum up, in case all the components are fault-free, an agreement can be reached in one round of message exchange if the network is a fully connected one; otherwise, two rounds will be needed.

**Case 2: Transmission medium failure only**

In this case, no fault-free processor knows exactly which transmission medium is failed. In order to keep away from the influence of a faulty transmission medium, collecting all the processors' messages is necessary. If the number of the faulty transmission media is not a dominant one, then the agreement can be determined through the MAJ. This way, only two rounds are needed in this case and at most $f_t$ ($\leq \lfloor (c+1)/2 \rfloor - 1$) transmission medium failures can be tolerated (the number of faulty transmission media is denoted as $f_t$).

In the first round of message exchange, where r = 1, the source multicasts its initial value $v_s$ through the connected transmission media. Each processor stores the value at the root of its mg-tree. Based on the assumption that the message sender is always identified, the processors are aware of the absence of messages. If there is no connection, replace it with λ. In the second round of message exchange, each processor acts as the source to exchange message and receives a vector of values. In order to reduce the influence of disconnected or faulty transmission media, we apply the LMAJ on the messages received from each same group. After that, the values are stored into the corresponding vertices at the second level of the mg-tree. Then, we reach an agreement by MAJ in the decision making phase. The MAJ($\alpha$) is defined as follows. The val($\alpha$i) tells that the message is sent to a series of receivers, denoted as $\alpha$ and the processors in Group $G_i$ are the latest receivers.

> The majority value in the set {val($\alpha$i) | $1 \leq l \leq n$, if it exists.}
> The complement of val($\alpha$i), denoted as ¬val($\alpha$), is chosen, otherwise.

Consider the case that part of the messages may be forged by a faulty transmission medium. The decision value can be dominated due to a greater number of processors belonging to the same group and consequently communicating with other groups through the identical faulty transmission medium. As a result, taking local majority eases and reduces the influences occurred from the faulty transmission media.

**Case 3: Processor failure only**

In this case, $f_g + 2$ rounds are required to solve the BA problem. The number of faulty groups is denoted as $f_g$ ($\leq \lfloor (g-1)/3 \rfloor$). If the number of fault-free processors in a group is greater than $\lfloor p/2 \rfloor$, it is called a fault-free group; otherwise, it is a faulty group, where p is the number of processors in the group and $p_i$ denotes the number of processors in the group $G_i$ ($1 \leq l \leq n$). As a result, the system can tolerate at most $\lfloor (g-1)/3 \rfloor \lceil P_{min}/2 \rceil + \lfloor (P_{min}-1)/2 \rfloor$ faulty processors while reaching a common value, where $P_{min} = MIN \{p_1, p_2, ..., p_g\}$.

In order to prevent the influence from the faulty processors, each processor multicasts the messages received in the (r-1)th round at the beginning of the rth round and maintains its mg-tree. In each round, the LMAJ is applied to the messages from the same group to lessen the influence of false messages. For each processor, if there is no connection, replace it with λ in the mg-tree. These steps are not done by the protocol of Bar-Noy ***et al.*** (Bar-Noy ***et al.***, 1987). After $f_g$+2 rounds, the messages stored in the mg-tree are reorganized into an ic-tree. In the decision making phase, all the fault-free processors reach a common value by applying the function VOTE to the messages collected in the message exchange phase and stored in the ic-trees. The VOTE(α) function for vertex α is defined as follows.

$$\text{VOTE}(\alpha) = \begin{cases} \bullet & \text{val}(\alpha), \text{ if } \alpha \text{ is a leaf.} \\ \bullet & \text{The majority value in the set of } \{\text{VOTE}(\alpha i) \mid 1 \le l \le n \text{ and vertex } \alpha i \text{ is a child of vertex } \alpha\}, \text{ if such a majority value exists.} \\ \bullet & \text{A default value } \phi \text{ is chosen, otherwise.} \end{cases}$$

The ic-tree is similar to the mg-tree, but the ic-tree only stores the non-duplicated name of each vertex. It helps avoid the cycle influence of a faulty processor/group and thus the agreement can be reached more easily.

**Case 4: Both processor failures and transmission medium failures exist**

In this generalized case, we have to face both faulty processors and faulty transmission media at the same time. To solve the BA problem here, the influences of both the faulty processors and the faulty transmission media must be removed. An intuitive method to remove these influences is to combine both the protocol for transmission medium failures and the protocol for processor failures. For instance, the protocol for transmission medium failures solves the transmission medium failure problem only, if $f_t \le (\lfloor (c+1)/2 \rfloor - 1)$. In addition, the protocol for processor failures is valid only for the specific fallible environment. If $f_p \le f_g \lceil P_{min}/2 \rceil + \lfloor (P_{min}-1)/2 \rfloor$ (say $f_g \le \lfloor (g-1)/3 \rfloor$), each fault-free processor must reach the agreement through the collected messages. Previous protocols (Lamport ***et al.***, 1982; Meyer ***et al.***, 1991; Ramaswami ***et al.***, 1993; Wang ***et al.***, 1995; Yan ***et al.***, 1999) deal with different fallible components separately, but they are incapable of reaching a common value in a complicated environment.

To solve the BA problem in a complicated environment where both faulty processors and faulty transmission media exist, there are four possible methods to remove the influences in different order. (1) Removing the influences of faulty processors and transmission media simultaneously. (2) Removing the influences of faulty processors first and then the influences of faulty transmission media. (3) Removing the influences of faulty transmission media first and then removing the influences of faulty processors. (4) Removing the influences of faulty transmission media and faulty processors alternately. Yan ***et al.*** (1999) depicted that the third method (removing the influences of faulty transmission media first and then removing the influences of faulty processors) is capable of solving the generalized case problem. Our MAP, following their observation, solves the transmission medium failures first and then the processor failures. The

number of required rounds of message exchange, denoted as $\delta$, for the BA is as follows.

Case 1: $\delta = 1$, if all the components are fault-free in an SGCN where $c = n-1$; $\delta = 2$, if all the components are fault-free in an SGCN where $c < n-1$

Case 2: $\delta = 2$, if the fallible component is transmission medium only.

Case 3: $\delta = f_g + 2$ ($f_g \leq \lfloor (g-1)/3 \rfloor$), if the fallible component is processor only.

Case 4: $\delta = f_g + 3$, if the fallible components include both processors and transmission media.

In our system, for a given network condition, the number of required rounds is the minimum among various cases. If not, case 4 is a default case. If $f_t \leq \lfloor (c+1)/2 \rfloor - 1$, $f_g \leq \lfloor (g-1)/3 \rfloor$ (that is, $f_p \leq f_g \lceil P_{min}/2 \rceil + \lfloor (P_{min}-1)/2 \rfloor$) and the amount of the faulty units (including the faulty group and the faulty transmission medium) $f_u = f_t + f_g$ ($f_u \leq \lfloor (c+1)/2 \rfloor - 1$), then the BA problem is solved. If the minimum number of required rounds is determined, each fault-free processor works effectively for collecting enough information and uses these messages to compute a common value in the decision making phase. The BA problem in the generalized case is solved in an SGCN.

**The protocol MAP (Multicasting Agreement Protocol)**

In this subsection, the MAP will be introduced to solve the generalized BA problem in the SGCN. The concept of the MAP protocol, as shown in Fig. 4, is as follows. In each round, the messages from the same group have to undergo the LMAJ to lessen the influence of false messages. In order to remove the influence caused by the faulty transmission medium, each processor applies the MAJ. In addition, in order to prevent the influence from the faulty processors, each processor exchanges messages with the others and maintains its mg-tree. For each processor, if there is no connection, then $\lambda$ is put down in the mg-tree. After $f_g+3$ rounds, the messages stored in the mg-tree are reorganized into an ic-tree. In the decision making phase, each processor reaches a common value by applying the VOTE function to the messages in its ic-tree. The ic-tree is constructed from a corresponding mg-tree by the following reorganization rules: (1) After the message exchange phase, the leaves of the mg-tree are deleted, since the influence of a faulty transmission medium in the leaves of an mg-tree is not removed; (2) The vertices with duplicated names are deleted; the purpose is to avoid the influence of a faulty processor being repeatedly stored again in an ic-tree.

**Correctness**

A vertex $\alpha$ is called common if each fault-free processor derives the same value from $\alpha$. That is, the value stored in vertex $\alpha$ of each fault-free processor's mg-tree or ic-tree is identical if the vertex $\alpha$ is common. If each fault-free processor has the common value in the root of its ic-tree, then an agreement is reached.

To prove that a vertex is common, the term common frontier is defined as follows. If each root-to-leaf path of the tree (the mg-tree or the ic-tree) contains a common vertex, then the collection of the common vertices forms a common frontier. That is, each fault-free processor

**Protocol MAP** (For each processor $P_i$)
Preprocessing:
For the network whose health condition is given
If the transmission media are fallible, then $\beta = 1$; otherwise $\beta = 0$.
If the processors are fallible, then $f_g = \lfloor (g-1)/3 \rfloor$; otherwise, $f_g = 0$.
If the SGCN is an FCN, $\delta = f_g + 1 + \beta$; else
If the SGCN is an BCN, $\delta = 2$;
Otherwise, $\delta = f_g + 2 + \beta$.
Message Exchange Phase:
r = 1:

1) The source multicasts its initial value $v_s$ to all the other processors and itself through the transmission media.
2) Each receiver stores its received value in the root s of its mg-tree.
3) If there exists the absence of messages (there is no connectivity between a pair of groups), fill in the value $\lambda$.
4) If $r = \delta = 1$, then GOTO the Decision Making Phase.

r = 2, ...,$\delta$:

1) Each processor multicasts the value at level r-1 in its mg-tree to the others and itself.
2) Apply the LMAJ to the received values.
3) Store the value to the corresponding vertices at level r of its mg-tree.
4) Apply the MAJ to each vertex at level r-1 for each processor's mg-tree and the function value is multicast to the others and itself in the next round.

Decision Making Phase:

1) Each processor's mg-tree is reorganized into a corresponding ic-tree.
2) Function VOTE is applied to root s of each processor's ic-tree and the common value VOTE(s) is obtained.

Fig. 4: The proposed protocol MAP solves the BA problem

has the same values collected in the common frontier if a common frontier exists in a fault-free processor's tree structure. The function VOTE computes the root value of the tree structure and each fault-free processor obtains the same root value due to the same input (the same collected values in the common frontier) and computing function. This concept is similar to the protocol proposed by (Bar-Noy ***et al.***, 1987).

The vertex $\alpha i$ of a tree is a correct vertex (Yan ***et al.***, 1999) if group $G_i$ is fault-free. That is, a correct vertex is the place to store the value received from a fault-free processor. In addition, for a vertex $\alpha i$ in a tree structure of a fault-free processor in the fault-free group $G_j$, val($\alpha i$) is the true value (Yan ***et al.***, 1999) if the transmission medium $T_{ij}$ (connects to $G_i$ and $G_j$) is perfect. In the following passages, Lemma 1 proves the true values of all the correct vertices in an ms-tree can be obtained. That is, the influence of any given faulty transmission medium is removed

from each fault-free processor's mg-tree. Lemma 2 indicates that all the correct vertices in each fault-free processor's mg-tree, except the leaves, are common. Lemmas 3 and 4 state that all the correct vertices in an ic-tree (eliminating the leaves of an mg-tree) are still common. Lemma 5 shows the existence of a common frontier. Lemma 6 and Corollary 2 present that the root of a tree structure is common if the common frontier exists in an ic-tree. The computed value stored in the root of an ic-tree is a common value, which is free from the influence of a faulty processor. As a result, Theorems 1 and 2 indicate the whole correctness.

**Lemma 1**

At the (r-1)th round, the processors in a fault-free group $G_i$ multicast val($\alpha$) to the others and each fault-free processor stores the local majority of the received values from $G_i$ after using the LMAJ, denoted as val($\alpha$i) in vertex $\alpha$i of its mg-tree. At the rth round, the MAJ($\alpha$i) which is applied to the vertices at the (r-1)th level of the mg-tree of each fault-free processors in $G_j$'s mg-tree should be the true value of vertex $\alpha$i, namely val($\alpha$), for $1 \leq j \leq g$ if the number of faulty units is less than $\lfloor (c+1)/2 \rfloor - 1$ in an SGCN model, for each r in $2 \leq r \leq f_g+3$.

**Proof: Part 1. The transmission medium $T_{ij}$ is perfect**

Since $T_{ij}$ is perfect, the processors in $G_j$ will receive val($\alpha$i) from the processors in $G_i$ in the (r-1)th round after taking local majority by LMAJ and val($\alpha$i) = val($\alpha$). The processors in $G_i$ multicast val($\alpha$) to the others. There are c transmission media connected to a processor in which at most $\lfloor (c+1)/2 \rfloor - 1$ units could be failed. In the next round, each processor in $G_j$ receives at least $(c-1) - (\lfloor (c+1)/2 \rfloor - 1) = \lceil c/2 \rceil$ val($\alpha$im) = val($\alpha$) after LMAJ, stored in the children of vertex $\alpha$i, from processors in other groups such as $G_m$, where at least $\lceil c/2 \rceil + 1$ val($\alpha$)'s ($\lceil c/2 \rceil$ val($\alpha$im)'s and a val($\alpha$ij) are both equal to this val($\alpha$) stored in the children of vertex $\alpha$i of the processors in $G_j$ and MAJ($\alpha$i) is equal to val($\alpha$).

**Part 2. The transmission medium $T_{ij}$ is faulty**

If $T_{ij}$ is faulty, consider the two following cases. Note that val($\alpha$ij) = val($\alpha$i) for the processors in a fault-free group $G_j$.

**Case 1: val($\alpha$i) = val($\alpha$)**

There are at most $\lfloor (c+1)/2 \rfloor - 1$ faulty units connected to the fault-free processors in $G_j$ and at most $\lfloor (c+1)/2 \rfloor - 1$ values that may be ¬val($\alpha$)'s in the children of vertex $\alpha$i. Since val($\alpha$ij) = val($\alpha$), there are at least $(c-1)-(\lfloor (c+1)/2 \rfloor - 1)$ val($\alpha$) free from the influence of a faulty unit among the rest (c-1) messages. The number of val($\alpha$)'s is $(c-1)- (\lfloor (c+1)/2 \rfloor - 1)+1 = \lceil c/2 \rceil + 1$ in this set. As a result, the majority for the values in the set val($\alpha$), or MAJ($\alpha$i), is val($\alpha$).

**Case 2: val($\alpha$i) = ¬val($\alpha$)**

There are at most $\lfloor (c+1)/2 \rfloor - 1$ faulty units in the network. At the rth round, the number of ¬val($\alpha$)'s is not beyond $\lceil c/2 \rceil$ and the number of val($\alpha$)'s is at least $(c-1)-(\lfloor (c+1)/2 \rfloor - 1)$. If c is an even number, then $\lceil c/2 \rceil = \lfloor c/2 \rfloor$, and the majority of the set [val($\alpha$i1), val($\alpha$i2), …, val($\alpha$ij)] for

vertex $\alpha i$ is undefined. By definition, $MAJ(\alpha i) = \neg val(\alpha) = val(\alpha i)$. In addition, if c is an odd number, then $\lceil c/2 \rceil < \lfloor c/2 \rfloor$, and the majority value, $MAJ(\alpha i)$, of the set is $val(\alpha)$.

**Corollary 1**

At the end of the $(f_g+2)$nd round, the true value of each non-leaf vertex in a fault-free processor's mg-tree can be obtained.

**Lemma 2**

All the correct and non-leaf vertices in a fault-free processor's mg-tree are common.

Proof: The value stored in a correct vertex $\alpha j$ is v, which implies that most of the processors in a fault-free group $G_j$ say that the value stored in vertex $\alpha$ of its mg-tree is v. Since most of the processors in $G_j$ are fault-free, the value $val(\alpha j) = v$ is stored in each fault-free processor's mg-tree unless the transmission medium connected to $G_j$ is failed. In this case, at most $\lceil c/2 \rceil$ values are v, which is stored in the children of vertex $\alpha j$ for such a receiver. The majority value v is obtained and restored in vertex $\alpha j$ by applying MAJ to vertex $\alpha j$. As a result, all the correct vertices have the common value v.

**Lemma 3**

If a parent vertex is correct in an mg-tree, then the fault-free children of the parent should be common. The true value for these fault-free children is the same at level $\leq f_g+2$.

**Proof**

To prove this lemma is to show that if $val(\alpha) = v$ for a correct vertex $\alpha$, then the value MAJ $(\alpha i)$ of each fault-free processor should be v on the condition that group $G_m$ is fault-free, $1 \leq m \leq g$. At round $r \leq f_g+3$, by Lemma 1, the value $val(\alpha) = v$ is available for the fault-free processors in $G_j$ and $G_m$, where vertex $\alpha$ is at level r-2;

Case 1: If transmission medium $T_{im}$ is perfect, then the fault-free processors in $G_i$ and $G_m$ have the value $MAJ(\alpha m) = val(\alpha m) = v$, where vertex $\alpha m$ is at level r-1.

Case 2: If transmission medium $T_{im}$ is faulty, then $val(\alpha m)$ may not be v. Since there are at most $\lfloor (c+1)/2 \rfloor - 1$ faulty units in the network, there are at least $\lceil (c+1)/2 \rceil$ correct vertices $\alpha mi$ with a common value v among the children of vertex $\alpha m$. Thus, by Lemma 1, MAJ $(\alpha m) = v$.

According to cases 1 and 2, where the level of $\alpha r$ in r-1 is lower than or equal to $f_g+2$, the true value of each fault-free children $\alpha m$ (at the level $\leq f_g+2$, $G_m$ is a fault-free group) should be the true value of its fault-free parent $\alpha$. Regardless of whether the transmission medium between the receiver and $G_m$ is fault-free or faulty, the fault-free children are common.

**Lemma 4**

All the correct vertices of an ic-tree are common.

**Proof**

After reorganization, no repeated vertices are in an ic-tree. At level $f_g+1$ or above, the correct vertex $\alpha$ has at least $2f_g+1$ children, of which at least $f_g+1$ children are fault-free. By Corollary 1, the true values of those $f_g+1$ correct vertices are in common, and the majority value of vertex $\alpha$ is common. The correct vertex $\alpha$ is common in the ic-tree if the level of $\alpha$ is $\leq f_g+1$. At level $f_g+2$, by Corollary 1, the correct vertex is still common. As a result, all the correct vertices of the ic-tree are common.

**Lemma 5**

The common frontier exists in the ic-tree.

**Proof**

There are $f_g+2$ vertices along each root-to-leaf path of an ic-tree in which the root is labeled by the source name, and the others are labeled by sequence of group names. Since at most $f_g$ groups can be failed, there are at least one correct vertex along each root-to-leaf path of the ic-tree. By Lemma 4, the correct vertex is common, and therefore the common frontier exists in each fault-free processor's ic-tree.

**Lemma 6**

Let $\alpha$ be a vertex, $\alpha$ is common if there is a common frontier in the subtree rooted at $\alpha$.

**Proof**

If the height of $\alpha$ is 0, and the common frontier ($\alpha$ itself) exists, then $\alpha$ is common. If the height of $\alpha$ is r, plus the children of $\alpha$ are all in common by the induction hypothesis with the height of the children being r-1, then the vertex $\alpha$ is common.

**Corollary 2:** The root is common if the common frontier exists in the ic-tree.
**Theorem 1:** The root of a fault-free processor's ic-tree is common.
**Proof:** By Lemma 5 and Corollary 2, the theorem is proved.

**Theorem 2:** Algorithm MAP solves the generalized BA problem in an SGCN.
**Proof:**

To prove the theorem, we have to show that the MAP algorithm meets agreements ($BA_1'$) and ($BA_2'$).

($BA_1'$): Root s is common.

By Theorem 1, (Agreement') is satisfied.

($BA_2'$): VOTE(s) = v for all the fault-free processors if the initial value of each fault-free processor is $v_i$; namely, $v = v_i$.

Since most of the processors are fault-free, they multicast the source's initial value to all the others. The true value of the correct vertices for all the fault-free processors' mg-trees is v. When the mg-trees are reorganized into ic-trees, the correct vertices still exist. As a result, each correct vertex of the ic-tree is common (Lemma 4), and its true value is v. By Theorem 1, this root is common. The computed value VOTE(s) = v is stored in the root for all the fault-free processors. Agreement ($BA_2$') is thus satisfied.

**Complexity**

The complexity of the protocol is evaluated in terms of (1) the minimum number of rounds, and (2) the maximum number of allowable faulty components.

**Theorem 3**

It requires $f_g+3$ rounds to solve the generalized BA problem in an SGCN, and it can tolerate at most $f_u$ ($\leq\lfloor(c+1)/2\rfloor-1$) units. Here, the number of allowable faulty groups is $f_g$ ($\leq\lfloor(g-1)/3\rfloor$), and the number of allowable faulty transmission media is $f_t$ ($\leq\lfloor(c+1)/2\rfloor-1$).

**Proof**

(1) In the message exchange phase, $f_g+3$ rounds are required, and no extra round is required during the decision making phase.

(2) By Lemma 1, the MAJ in the mg-tree can take care of at most $\lfloor(c+1)/2\rfloor-1$ faulty transmission media. By Lemma 4, the ambiguity due to at most $f_g$ faulty groups can be resolved. The theorem is proved.

**Theorem 4**

The proposed protocol solves the generalized BA problem by using the minimum number of rounds.

**Proof**

For an SGCN, $f_g+2$ rounds is the minimum number of rounds to solve the BA problem with $f_g$ faulty groups. In the generalized case, at least $f_g+2$ rounds are required to reach an agreement. Suppose there is an algorithm that can solve the generalized BA problem by using $f_g+2$ rounds under the assumption of processor/transmission medium failures. Since the symptoms of faulty processors/transmission media are both malicious, a faulty transmission medium may be just as disruptive as a faulty group, and vice versa. As a result, the $f_g+2$ rounds algorithm solves the BA problem with $f_g+1$ malicious faulty groups. It contradicts the results of processor failures only, in which $f_g$ faulty groups are the maximum number of faulty groups allowed by the $f_g+2$ rounds algorithm to solve the BA problem in an SGCN. In other words, it is impossible to solve the BA problem by $f_g+2$ rounds in an SGCN with $f_g$ faulty groups and some additional faulty transmission media. As a result, $f_g+3$ rounds are the minimum number of rounds.

**Theorem 5**

The total number of allowable faulty units $f_u$ ($\leq \lfloor (c+1)/2 \rfloor -1$) is the maximum.

**Proof**

Since $f_u \leq \lfloor (c+1)/2 \rfloor -1$ is the maximum number of allowable faulty transmission media in a system with transmission medium failures only ($f_g=0$), if the number of $f_u = f_g+f_t \leq \lfloor (c+1)/2 \rfloor -1$ is not the maximum number of allowable faulty units, then $f_u = f_g + f_t = 0 + f_t \geq \lfloor (c+1)/2 \rfloor$ if $f_g = 0$. Then, $f_t$ is either equal to or larger than $\lfloor (c+1)/2 \rfloor$, which leads to a contradiction.

As a result, the MAP takes the minimum number of rounds and tolerates the maximum number of faulty components to make each fault-free processor reach the agreement. The optimality of the protocol is proved.

**Conclusions and future works**

The multicasting network is a more and more popular choice in various communication environments. The main advantage of the multicasting network is that one processor has to only send a copy of some frame across the network, and different computers under the cover of the network can receive the same frame, which is more convenient and efficient than traditional network systems. Based on the generalized network model, in this paper, the BA problem has been reviewed with malicious processor/transmission medium failures. The MAP solves the generalized BA problem at the cost of the minimum rounds of message exchange and tolerates the maximum number of faulty components.

$(f_g+2)$ rounds is the lower bound for solving the BA problem in an SGCN under the assumption of processor failures only, for the messages received at the $(f_g+2)$th round may be influenced by the faulty transmission media, and additional rounds may be required to remove the influence of faulty transmission media on the $(f_g+2)$th round. As a result, $f_g+3$ rounds is the optimal solution for the generalized BA under the assumption of both processor and transmission medium failures. The protocol MAP solves processor failures/transmission medium failures only cases as well. If the fallible component is known, then the number of required rounds is:

(1) Only two rounds are required to help each processor reach an agreement while all the components in the SGCN model are fault-free/perfect with $c < n-1$; otherwise, only one round is needed.
(2) Two rounds are required for the transmission-medium-failures-only case (the processors are fault-free).
(3) $(f_g+2)$ rounds are required to solve the BA problem when only the processors are on the fallible side.
(4) $(f_g+3)$ rounds are required when the network condition is unknown or when both the processors and transmission media may be fallible.

The MAP solves the BA problem in the SGCN model if and only if the number of faulty units is smaller than $\lfloor (c+1)/2 \rfloor - 1$. It tolerates at most $\lfloor (g-1)/3 \rfloor$ faulty groups (which implies that $\lfloor (g-1)/3 \rfloor \lceil P_{min}/2 \rceil + \lfloor (P_{min}-1)/2 \rfloor$ is the maximum number of faulty processors allowed), and at most $\lfloor (c+1)/2 \rfloor - 1$ faulty transmission media may exist. If the number of faulty components exceeds the limitation, the agreement will not be reached. The messages sent by fault-free/perfect components do not dominate the messages from faulty components, and the concept of majority, which helps to remove the influence of faulty components, is applied; as a result, we have the following facts:

(1) To solving the generalized BA, the MAP is an optimal solution that tolerates the maximum number of faulty components at the cost of the minimum number of rounds of message exchange.
(2) The protocol solves the generalized BA problem, and thus the processor-failures-only case and the transmission-medium-failures-only case are also solved.
(3) The symptom of faulty components we can deal with is the malicious type, which is the thorniest and the most generalized type. With such ability, the proposed protocol can cope with other failure types with ease, such as crash and omission with processors only or transmission media only or both.
(4) Since the SGCN incorporates the FCN, the BCN, and the GCN, the FCN, the BCN, and the GCN are in fact special cases of the SGCN. In other words, the capability of our solution to the BA problem with the SGCN covers all the other three network structures.

The multicasting network is a network system where a frame may be broadcast or sent to an individual or a set of group addresses. That is, a frame may be sent to an individual this time and sent to a set of group addresses next time. Both the receivers and the number of them can be different, and that is the case the proposed protocol is incapable of closing yet. Therefore, our future work will be focused on extending the SGCN model and designing a more generalized protocol. The SGCN model will be made more complex so that a frame can be sent to different sets of receivers in different rounds. For example, $P_1$ sends messages to $P_2$ and $P_3$ in the first round and to $P_4$, $P_5$, and $P_6$ in the second round.

**References**

Babaoglu, O. and R. Drummond, 1985. Streets of byzantium: Network architectures for fast reliable broadcasts, IEEE transactions on data and knowledge engineering, pp: 546-554.

Bar-Noy, A., D. Dolev, C. Dwork and R. Strong, 1987. Shifting Gears: Changing algorithms on the Fly to expedite byzantine agreement, in proceedings of the 6th annual ACM symposium on principles of distributed computing, pp: 42-51.

Cristian, F., 1991. Reaching Agreement on processor-group membership in synchronous distributed systems, Distributed computing, pp: 175-187.

Dolev, D., 1982, The Byzantine generals strike again, J. Algorithm, pp: 14-30.

Dolev, D. and R. Reischuk, 1985. Bounds on Information exchange for byzantine agreement, J. ACM., pp: 191-204.

Fischer, M., M. Paterson and N. Lynch, 1985. Impossibility of distributed consensus with One faulty process, J. ACM., pp: 374-382.

Gifford, G.K., 1979. Weighted Voting for replicated data, Technical report, CSL-79-14, XEROX Palo Alto research center.

Halsall, F., 1995. Data communications, computer networks and open systems, Addison-Wesley publishing company, (4th Ed.).

Lamport, L., R. Shostak and M. Pease, 1982. The byzantine generals problem, ACM transactions on programming languages and system, pp: 384-401.

Lamport, L. and P. Melliar-Smith, 1984. Byzantine clock synchronization, in proceedings of the 3rd ACM Principles of Distributed Computing Conference, pp: 10-16.

Meyer, F.J. and D.K. Pradhan, 1991. Consensus with dual failure modes, IEEE Transaction Parallel and distributed systems, pp: 214-222.

Molina, G., 1986. Application of byzantine agreement in database systems, ACM transactions on database systems, pp: 27-47.

Pease, M., R. Shostak and L. Lamport, 1980. Reaching agreement in presence of faults, J. ACM., pp: 228-234.

Ramaswami, V. and J.L. Wang, 1993. Analysis of the link error monitoring protocols in the common channel signaling network, IEEE/ACM Transactions on Networking, pp: 31-47.

Reischuk, R., 1982. A new solution for the byzantine generals problem, IBM research report, RJ-3673, computer science.

Siu, H.S., Y.H. Chin and W.P. Yang, 1996. A Note on consensus on dual failure modes, IEEE Transactions on parallel and distributed systems, pp: 225-230.

Strong, H. and D. Dolev, 1982. Byzantine agreement, IBM research report, RJ-3714, computer science.

Wang, S.C., Y.H. Chin and K.Q. Yan, 1995. Byzantine agreement in a generalized connected network, IEEE transactions on parallel and distributed systems, pp: 420-427.

Yan, K.Q., S.C. Wang and Y.H. Chin, 1999. Consensus under unreliable transmission, information processing letters, pp: 243-248.