Proceedings of International Conference on Distributed
Systems, Software Engineering and Database Systems

An Optimal Agreement Protocol for Hybrid Fault Model on Communication Links

Hin-Sing Siu†, Yeh-Hao Chin‡, and Wei-Pang Yang†

†Department of Computer and Information Science,
National Chiao Tung University,
Hsinchu, Taiwan, Republic of China

‡Department of Computer Science,
National Tsing Hua University,
Hsinchu, Taiwan, Republic of China

Abstract

The Byzantine agreement (BA) problem is one of the most important problem in designing the fault-tolerant distributed system. In practice, the components in a network (either processors or links) may be subjected to different failure types simultaneously (or called hybrid fault model or mixed faulty types). Most of the earlier work of the BA problem under a hybrid fault model assume that links are fault-free; however, a link can also fail in various faulty types. Examples of link failure include crash, omission, stuck-at, or malicious faults. In this paper, we consider the BA problem under a hybrid fault model on communication links. The proposed protocol uses the minimum number of message exchanges and can tolerate the maximum number of allowable faulty links to make each processor reach a common agreement.

Keywords: Byzantine agreement; Distributed systems; fault tolerance; hybrid fault model, link failure; mixed faulty types.

1. Introduction

The fault-free processors in most distributed systems require a facility to achieve a common agreement. To achieve such a goal is called the *Byzantine agreement* (*BA*) problem [5], of which protocols are required so that the system will run even if certain components in the system fail. The *BA* problem involves a network with n processor, and some components of the network (either processors or links) may fail. Note that each processor does not know which component is faulty. One distinguished processor, called the *Source*, starts with an initial value v_s drawn from the set of all possible values V of the *BA* problem. The goal of the *BA* protocol is to make each fault-free processor reach a common value broadcast by the Source. After the execution of the protocol, the common value obtained by the fault-free processors shall be the value satisfied the following conditions.

Agreement: Each fault-free processor agrees on the common value v.

Validity: If the Source is fault-free, the value agreed on is equal to the initial value v_s of the Source; i.e., $v_s = v$.

Most of the applications of a distributed system, such as clock synchronization, reliable communication, replicated file systems, task scheduling, and system reconfiguration, need an agreement obtained by all fault-free processors prior to for executing the cooperating tasks subsequently. Therefore, a *BA* protocol can be treated as the *building block* (primitive component) or *pre-step* for the distributed applications in a network.

In practice, more than one type of failure can exist in the components of a network. Most of the earlier work of the *BA* problem under a *hybrid fault model* (or called mixed faulty types) assumes that links are fault-free [6, 7, 13]; in practical, however, a link can also fail [9, 10, 15, 16]. For example, in the *Common Channel Signaling* (*CCS*) network based on Signaling System No. 7 (SS7) protocol has a dedicated algorithm to monitor the error of communication links [10]. Traditionally, such a link fault was treated as a processor fault [13][1]. The treatment ignores the fact that the processor connected with faulty link is fault-free (i.e., called *innocent processor* [15, 16]); hence an innocent processor does not involve an agreement under the treatment. It is a contradiction with the definition of the *BA* problem that requires all fault-free processors to reach an agreement. From this standpoint, a

[1] One of the fault-free processor adjoined a faulty link is treated as faulty.

link failure should be treated differently from the case of a processor failure.

There are four types of link failures frequently discussed with regard to the *BA* problem in a network [2, 9, 15, 16]. The first one is called a *crash fault* that reflects the case of a broken link. The second one is an *omission fault*, in which the message passed through a faulty link may be dropped. The third type is called a *stuck-at fault*, in which the message received along a faulty link is always a constant. The fourth type is called a *malicious fault* (also called a *Byzantine fault*), and the behavior of such a fault can exhibit arbitrarily behavior. According to the faulty behaviors, these types of link failures can be classified to two disjoint sets: *dormant* and *arbitrary* faults. A dormant fault is one that do not contaminate the content of a message. This faulty type can be detected by a processor via a time-out mechanism. Examples of the dormant fault are crash and omission faults. On the other hand, a stuck-at and a malicious fault can change the content of a message arbitrarily. Hence, these faults can be group as another faulty type and is called arbitrary fault.

FLINK[2], the optimal consensus protocol, proposed by Yan and Chin [15], is designed in a *synchronous fully connected network* of which at most L_a links are subjected to arbitrary faults. In such a network model, the bounds on processing and communication delays of the between fault-free components (processors and links) are finite. FLINK can solve the *BA* problem if the constraint on failures, namely $n > 2L_a + 1$, holds, where n is the number of processors. However, all faulty links are treated as arbitrary faults in [15]. This treatment ignores the fact that the faulty behavior of a dormant fault is more consistent than that of an arbitrary fault. As a result, FLINK tended to be over conservative. Based on the discussion of hybrid fault model on processors [6, 7, 10, 11, 12, 13], the maximum number of faulty links cannot be tolerated by FLINK if hybrid fault model is considered. For example, Fig. 1 shows a fully connected network with five processors. Assume that the links L_{AB} and L_{AE} are subjected to dormant and arbitrary faults, respectively. By the treatment of [15], the network will has two arbitrary links. When the FLINK protocol is applied, an agreement cannot be reached among these five processors because the number of faulty links, 2, is greater than the maximum number of allowable faulty links, namely $\lfloor n/2 \rfloor - 1 = \lfloor 5/2 \rfloor - 1 = 1$.

For completeness, Fig. 2 shows the step-by-step procedure of the FLINK protocol. The FLINK protocol consists of two phases: *message exchange phase* and *decision making phase*. The message exchange phase consists of two rounds to collect messages and the goal of the decision making phase is to compute the common value for an agreement. In the first message exchange round, the Source A first broadcasts its initial value '1' to all processors as shown in Fig. 2(a). Fig. 2(b) indicates the message received at each processor after the first round. Suppose that the processors A, C, and D receive value '1', and that the processor E receives a disturbed value '0' due to the failed link L_{AE}. Since the processor B does not receive any message, the default value, say '0', is selected by B based on the FLINK protocol. Then, each processor exchanges the message received from the Source A in the second message exchange round. As shown in Fig. 2(c), these five fault-free processors cannot agree on the value '1' that was sent by the Source A after the majority voting function is applied in the decision making phase.

In this paper, we consider the *BA* problem in a synchronous fully connected network with fault-free processors and propose a protocol that can solve the *BA* problem under the following assumptions:

(1) A fault-free processor does not require prior information of the system faulty status.
(2) Let n be the total number of processors. Each processor's identifier is unique.
(3) Let L_a be the number of links subjected to arbitrary faults.
(4) Let L_d be the number of links subjected to dormant faults.
(5) Let V be the set of all possible values of the *BA* problem. Each processor uses a predefined enumeration of the values in V: $v_1, v_2, ..., v_n$.

In such a network model, we firsts propose the protocol for the *BA* problem under a hybrid fault model on communication links, called BAHL. BAHL can solve the BA problem by using *two* message exchange rounds[3] if $n > 2L_a + L_d + 1$. Then, we will extend the proposed protocol to handle the case of hybrid fault model in a *nonfully connected network* with sufficient *connectivity*, and also extend the proposed protocol to solve the *interactive consistency problem* [8] and the *consensus problem* [1] under a hybrid fault model.

2. Concept and Approaches

2.1 General Description

In a synchronous fully connected network, processors executing BAHL will have received correct messages from all other fault-free processors at a predictable time. BAHL uses this characteristic to solve the *BA* problem under a hybrid fault model. Obviously, if the link between a sender

[2] Originally, FLINK was designed for the consensus problem, but it can also work for the *BA* problem. The difference between the consensus and the *BA* problems is in the number of the owner of the initial value; in the former it is n; while in the latter it is 1.

[3] A round denotes the interval of message exchange.

and a receiver is subjected to dormant fault, then the receiver is unable to receive any message through the link, during some rounds of the message exchange phase. In our approach, whenever this situation (no message is received) has been detected by a processor, the value ∅ ($\varnothing \notin V$) is used to replace the incoming message instead of a default value being assigned as in FLINK. When this situation occurs in the first round, the processor will relay the value ∅ to all processors (exclude the Source) in the second round. This approach can be defined in the following rule.

Absent rule: During the message exchange phase, when a processor P receives nothing from a processor Q through a link L, it uses the value ∅ to replace the incoming messages from Q and relays the value ∅ to all processors (exclude the Source) in the next round (if any).

Semantically, once a processor P discovers that L is a faulty link, the voting right of the processor Q at the other side of the faulty link L is denied by the processor P. The value ∅ from such links is counted as an *absent vote*. Since the faulty link L can no longer influence the voting process, based on majority voting, the number of allowable faulty links can be increased. In FLINK, the received message of processor B shown in Fig. 2(b) is '*0*'. The italic '*0*' indicates that the processor B does not receive the Source A's message; therefore a default value '*0*' is selected by B. Semantically, the interpretation of the FLINK's approach is that B is still treated as a *present voter*. Subsequently, this value will be relayed to all processors and counted as a *valid vote* in the decision making phase as shown in Fig. 2(c).

With the absent rule, BAHL, similar to FLINK, consists of two phases: a *message exchange phase* and a *decision making phase*, and a tree-based structure called an *information gathering tree* (IG-tree) [3] is used to collect the voting messages for each processor. Initially, each processor creates an empty IG-tree. Fig. 3 shows an example of BAHL. Initially, each processor creates an empty IG-tree. In the message exchange phase, each processor executes two rounds to collect voting messages. In the first round, the Source A sends its initial value to all other processors and then halts (select its initial value as common value). When a fault-free processor receives the message sent by A, it stores the received message, denoted by val(A), at the root of its IG-tree as shown in Fig. 3(b). By the absent rule, processor B uses the value ∅ to replace the message received from the Source. In the second round, the $n-1$ processors exchange the message received from the Source. If processor B sends a message to processor C, C stores the message, is denoted by val(B), at vertex B in the second level of its IG-tree, as shown in Fig. 3(c). During the decision making phase, each fault-free processor applies the voting function MAJ to the root of its IG-tree. MAJ can be defined as follows:

MAJ = the most common value (exclude value ∅) stored at level 2 of IG-tree

If more than one common value existed, the output of MAJ is the one that appears first in the predefined enumeration of the values in V (all processors use the same enumeration). Using MAJ, Fig. 3(c) shows that each processor obtains the common value '1' broadcast by the Source A, even though the number of faulty links, 2, exceeds FLINK's limitation, namely $\lfloor n/2 \rfloor - 1 = \lfloor 5/2 \rfloor - 1 = 1$. Moreover, more failure types can be covered by our approach. A formal description of the proposed protocol BAHL will be presented in the next section.

2.2. The Protocol

The goal of BAHL is to make each fault-free processor obtain a common value for an agreement. As mentioned in Section 2.1, BAHL consists of message exchange phase and decision making phase. During the message exchange phase, each processor applies the absent rule to remove the influence of a faulty link that do not relay messages properly. After the message exchange phase, in the worst case, each processor (except for the Source) collects at least $n\text{-}1\text{-}L_d$ valid messages after ignoring these L_d absent votes. In order to reach an agreement, by the concept of majority vote, the number of valid messages, $n\text{-}1\text{-}L_d$, must be greater than $2L_a$, namely $n > 2L_a + L_d + 1$. Hence, the maximum number of tolerable faulty links by BAHL is $L_a + L_d$. Based on the above concept, BAHL can be formal presented as follows.

Protocol: BAHL(for each processor *P*)
Input: The Source with the initial value v_s.
Output: The common value.
Method:
Initialization
Create an IG-tree with empty root.
Message Exchange Phase
Round 1:
1. If P is the Source
   then send the initial value v_s to all other n - 1 processors and halt.
2. Receive the message sent by the Source.
3. Apply the absent rule to the root of its IG-tree.

Round 2:
1. Send the value stored at the root s of its IG-tree to all other processors (except for the Source) and itself.
2. Receive the messages from all other processors (except for the Source) and itself and store the received messages to the second level of its IG-tree.

3. Apply the absent rule to the second level of its IG-tree.

Decision Making Phase

1. Apply the voting function MAJ to its IG-tree.
2. Output the value computed by MAJ as the common value and exit.

3. Correctness and Complexity Analysis

3.1. Correctness

A *BA* protocol has to make every fault-free processor reach a common value; hence the correctness of the BAHL can be proven from the fact that the common value of every fault-free processor satisfies the conditions of *Agreement* and *Validity*. We say that a root R of an IG-tree is defined to be *v-common* if the output of voting function MAJ applied to every processor's IG-tree is $\bar{v}$. Therefore, the *BA* problem can be solved if the root of the IG-tree in every processor (except for the Source) is v_s-common when the initial value of the Source is v_s.

Theorem 1. BAHL does solve the *BA* problem under a hybrid fault model on links if $n>2L_a+L_d+1$.

Proof. Because the processors are assumed to be fault-free, the Validity condition implies the Agreement condition; thus, we only have to prove that the Theorem is true when BAHL satisfies the Validity condition. By BAHL, the common value of the Source is its initial value v_s. The other $n-1$ processors exchange the message received from the Source to compute their common value. After the message exchange phase, each processor (except for the Source) collects at least $n-1-k$ valid messages in the second level of its IG-tree when k absent votes are ignored. Let m be the number of the valid messages, namely $m=n-1-k$. We can write $k=\alpha+\beta$, where α is the number of invalid votes with respect to the dormant faults, $\alpha\geq 0$, and β is the number of invalid votes with respect to the arbitrary faults (the arbitrary faulty links drop the messages), $\beta\geq 0$. Therefore we can write $L_d=L_d^{'}+\alpha$, where $L_d^{'}\geq 0$, $L_a=L_a^{'}+\beta$, where $L_a^{'}\geq 0$. Note that the set of $L_d^{'}$ elements (each element is correct) result from the dormant fault (these faulty links always send correct messages in the protocol). To compute the common value for an agreement from these valid messages, by the concept of the majority vote, the number of valid messages must be greater than $2L_a^{'}+L_d^{'}$; namely $m>2L_a^{'}+L_d^{'}$.

By hypothesis,

$$n>2L_a+L_d+1,$$

so that

$$m>2L_a+L_d-k.$$

We can write,

$$\begin{aligned} m &> 2L_a+L_d-(\alpha+\beta), \\ &> 2(L_a^{'}+\beta)+(L_d^{'}+\alpha)-(\alpha+\beta), \\ &> 2L_a^{'}+L_d^{'}+\beta. \end{aligned}$$

Since $\beta\geq 0$, we can write $m>2L_a^{'}+L_d$; therefore, the majority value of the $n-1-k$ values is Source's initial value v_s. The Validity condition is satisfied; thus, the theorem is proven. □

Corollary 1. In case of a single faulty type, BAHL can separately tolerate L_c $(<n-1)$ crashed faulty links, L_o $(<n-1)$ omission faulty links, L_s $(\leq\lfloor n/2\rfloor-1)$ stuck-at faulty links, or L_m $(\leq\lfloor n/2\rfloor-1)$ malicious faulty links, $n>1$.

3.2. Complexity

The complexity of the proposed protocol BAHL is defined in terms of the number of rounds required and the number of tolerable faulty links. Theorem 1 indicates that the number of tolerable faulty links for BAHL is L_a+L_d that is not less than the tolerable bound of the FLINK protocol, namely $(\leq\lfloor n/2\rfloor-1)$; therefore, the number of tolerable faulty links for BAHL is further increased than that for FLINK. Furthermore, we prove that the proposed protocol is optimal in terms of the minimum number of rounds required and the maximum number of faulty links allowed.

Lemma 1. BAHL is round-optimal.

Proof. In the *BA* problem with unreliable communication links, from the result of Yan and Chin [15], the minimum number of rounds required is two. It is impossible for the number of rounds required to be less than two. Thus the number of rounds required by the proposed protocol, two, is optimal. □

Theorem 2. BAHL is optimal in terms of the number of rounds and the number of tolerable faulty links.

Proof. The number of tolerable faulty links for BAHL is L_a+L_d, and Theorem 1 indicates that common agreement cannot be reached among the processors if $n\leq 2L_a+L_d+1$ for the case of hybrid fault model. Therefore, the number of tolerable faulty links for BAHL, L_a+L_d, is optimal. From Lemma 1 and Theorem 1, the theorem is proven. □

4. Conclusions and Discussion

The proposed protocol BAHL is presented to solve the *BA* problem under a hybrid fault model on communication links. Table 1 gives an example to compare the number of tolerable faulty links for BAHL and that for FLINK of [15]. As indicated, BAHL can solve the *BA* problem if the number of faulty links is not more than $L_a + L_d$ and $n > 2L_a + L_d + 1$. Note that $L_a + L_d$ is not less than the number of tolerable faulty links for the FLINK protocol, $\lfloor n/2 \rfloor - 1$, namely $L_a + L_d \geq \lfloor n/2 \rfloor - 1$. When the number of processors is 5, for example, BAHL can tolerate an arbitrary fault and a dormant fault, or three dormant faults; while FLINK tolerate one faulty link only as shown in Table 1. In this example, hence, the total number of tolerable faulty links for BAHL is greater than that for FLINK.

Although BAHL is originally designed for a fully connected network, it can be extended to the case of hybrid fault model in a *nonfully connected network* if $c > 2L_a + L_d$, where c is the system *connectivity*. By the Menger theorem [4], there exist at least c disjoint paths between every pair of processors S and R[4] if the system connectivity is c. This means that processor S can send c copies of its message through c disjoint paths to processor R. Suppose that each processor p maintains a four-tuple (*S*, *R*, *predecessor*, *successor*) information such that path <*predecessor*, *p*, *successor*> is a subpath of a path from the S to R [7, 9, 11]. The processors S and R also need the c neighbors along a prescribed set of processor-disjoint paths. We assume that the processor p only relays a message to its immediate successor if p receives a message from its immediate predecessor along the disjoint path (if any) between S and R. Using such information, the processor S can send c copies of its message of type (*R*, *S*, *message*) through c processor-disjoint paths to every immediate successor. Note that an arbitrary faulty link can affect at most one message of these c messages and that a dormant faulty link can drop at most one message of these c messages. Hence, processor R collects at least $c - k$ valid messages after ignoring the k absent votes. As in the previous discussion in Theorem 1, we can extend BAHL for the case of hybrid fault model on links in nonfully connected network if $c > 2L_a + L_d$.

Based on the discussion of Yan *et al.* [16], the proposed protocol for the *BA* problem can also be extended to solve the *interactive consistency problem* [8] and the *consensus problem* [1] for the case of hybrid fault model on links. In order to solve these two problems, each processor acts as a Source of the *BA* problem and starts BAHL simultaneously (n copies of BAHL are parallel executed). Therefore, the set of n common values, N, is obtained by each processor, and then the interactive consistency problem can be solved. Subsequently, the consensus problem can also be solved by applying a majority vote to N (the output value of the majority vote is the common value for the consensus problem).

To summarize, the preceding discussion supports the following conclusions:

1. The proposed protocol BAHL is optimal in terms of the number of rounds required and the number of tolerable faulty links.
2. BAHL can also solve the *BA* problem for a single failure type as described in section 3. Therefore, BAHL provides a general solution to the *BA* problem under a hybrid fault model on communication links.
3. BAHL can be extended to solve the *BA* problem under a hybrid fault model on communication links in nonfully connected network. It can also be further extended to solve the interaction consistency and the consensus problems.
4. BAHL, without any modification, can be used to solve the reliable broadcasting problem [9] in networks under a hybrid fault model. However, the number of tolerable faulty links is increased.

[4] We only describe the case of a pair of processors. Analyzing this case will enable us to portray the general case in which every pair of processors can exchange their messages.

References

[1] O. Babaoglu, On the reliability of consensus-based fault-tolerant distributed computing systems. *ACM Trans. Comp. Syst.* 5 (1987) 394-416.

[2] O. Babaoglu and R. Drummond, Street of Byzantium: Network architectures for fast reliable broadcasts. *IEEE Trans. Software Eng.* 11 (1985) 546-554.

[3] A. Bar-Noy *et al.*, Shifting gears: Changing algorithms on the fly to expedite Byzantine agreement. *Proc. 1987 Symp. on Principle of Distributed Computing* , (1987) 42-51.

[4] N. Deo, *GRAPH THEORY with Applications to Engineering and Computer Science*, (Prentice-Hall, 1974).

[5] L. Lamport, R. Shostak, and M. Pease, The Byzantine Generals Problem, *ACM Trans. Prog. Lang. Syst.* 4 (1982) 382-401.

[6] P. Lincoln and J. Rushby, A formally verified algorithm for interactive consistency under a hybrid fault model. *Proc. 1993 Symp. Fault-Tolerate Computing. IEEE* Computer Society (1993) 402-411.

[7] F. J. Meyer and D. K. Pradhan, Consensus with dual failure modes. *IEEE Trans. on Parallel and Distributed Systems.* 2 (1991) 214-222.

[8] M. Pease, R. Shostak, and L. Lamport, Reaching agreement in presence of faults. *J. ACM.* 27 (1980) 228-234.

[9] A. Pelc, Reliable communication in networks with Byzantine link failures. *NETWORKS.* 22 (1992) 441-459.

[10] V. Ramaswami, and J. L. Wang, Analysis of the link error monitoring protocols in the common channel signaling network. *IEEE/ACM Trans. on Networking.* 1 (1993) 31-47.

[11] H. S. Siu, Y. H. Chin, and W. P. Yang, Byzantine agreement with mixed failure types, *Proc. of IASTED Int. Conf. on Applied Informatics* (1995) 203-206.

[12] H. S. Siu, Y. H. Chin, and W. P. Yang, A note on consensus on dual failure modes, *IEEE Trans. on Parallel and Distributed Systems*, vol. 7, no. 3 (1996) 225-230.

[13] P. Thambidurai and Y.-K. Park, Interactive Consistency with Multiple failure modes. *Proc. 1988 Reliable Distributed Systems. IEEE* Computer Society (1988) 93-100.

[14] J. Turek and D. Shasha, The many faces of consensus in Distributed systems. *IEEE Computer* 6 (1992) 8-17.

[15] K. Q. Yan and Y. H. Chin, An optimal solution for consensus problem in an unreliable communication system. *Proc. ICPP* (1988) 388-391.

[16] K. Q. Yan, Y. H. Chin, and S. C. Wang, Optimal agreement protocol in malicious faulty processors and faulty links. *IEEE Trans. on Knowledge and Data Eng.* 4 (1992) 266-280.

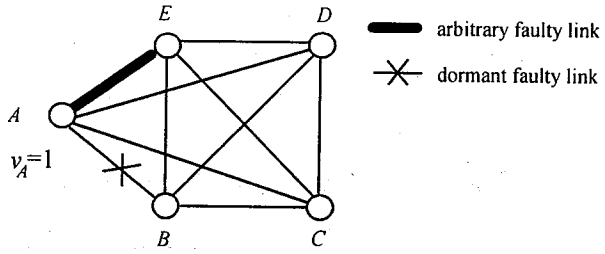
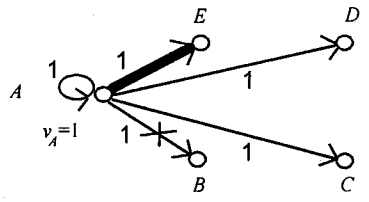


Fig. 1. A reliable-processors fully connected network.

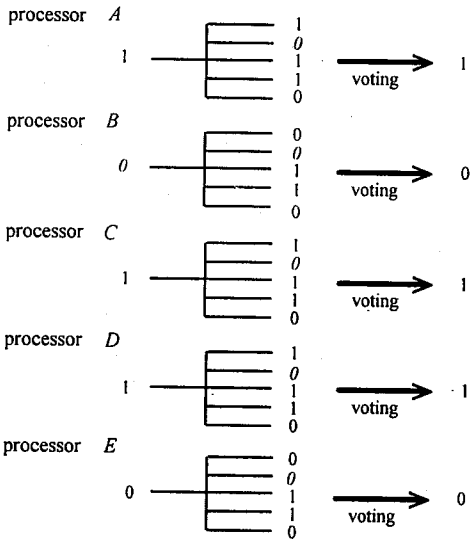


(a) The first round

The received message of each processor

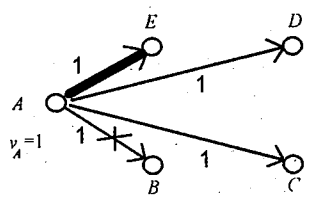
A	1	
B	*0*	(default value)
C	1	
D	1	
E	0	

(b) After the first round



(c) The decision making phase

Fig. 2. An example of FLINK for hybrid fault model.



(a) The first round

The received message of each processor

level 1 (root)

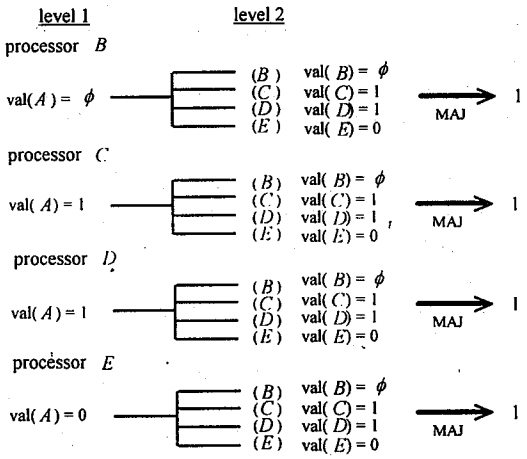
processor B : (A) val(A) = ϕ

processor C : (A) val(A) = 1

processor D : (A) val(A) = 1

processor E : (A) val(A) = 0

(b) After the first round



(c) The decision making phase

Fig. 3. The procedure of BAHL.

TABLE 1

Example of the number of tolerable faulty links for BAHL and FLINK

n	BAHL		FLINK
	L_a	L_d	$\lfloor n/2 \rfloor - 1$
5	1	1	1
	0	3	
6	2	0	2
	1	2	
	0	4	
7	2	1	2
	1	3	
	0	5	