# General Consensus Protocols *

4M−7

Takayuki Tachikawa, Chiaki Yahata, and Makoto Takizawa †
Tokyo Denki University ‡
e-mail{tachi,chii,taki}@takilab.k.dendai.ac.jp

## 1 Introduction

The distributed applications like groupware are realized by a *group* of multiple processes. The processes in the group have to make consensus in order to do the cooperation among them. There are kinds of consensus protocols [2, 3]. In the *atomic commitment* [1], each process cannot change the mind after notifying other processes of the vote. However, in the human society, individuals often change the minds even after notifying others of the votes. In addition to the atomic commitment, various kinds of decision logics have to be adopted. When considering the cooperation of multiple processes, we have to think about what process coordinates the cooperation among the processes. We have to consider the distributed control where there is no centralized controller. In this paper, we discuss a general consensus protocol [5].

In section 2, we present a general model of consensus protocol. In section 3, we discuss the ordered relation on the values taken by the processes. In section 4 and 5, the global decision and coordination schemes are discussed. In section 6, we would like to discuss extended 2PC protocol based on the general model.

## 2 General Consensus Model

A distributed system is composed of multiple processors interconnected by communication networks. A distributed application is realized by the cooperation of $n$ ($> 0$) processes $p_1, \ldots, p_n$, where each $p_i$ is computed in one processor. In the distributed applications, $p_1, \ldots, p_n$ have to make consensus among themselves. The commitment protocols [1] are used as the consensus ones where the following points are assumed:

1 no process can change the opinion after voting,
2 the decision logic is *all-or-nothing* principle,
3 there is one centralized controller,
4 process is not autonomous, i.e. it obeys the decision of the coordinator, and
5 *No* dominates *Yes*, i.e. processes voting *No* abort unilaterally without waiting for the decision from the coordinator.

The general consensus protocol has to take into account the following points :

1 each process can change the opinion,
2 each process can express the opinion *No-idea* and *Anyone − OK* in addition to *Yes* and *No*,
3 various kinds of decision logics like *all-or-nothing* and *majority-consensus* can be adopted,
4 each process may be autonomous, and
5 there are kinds of coordination among the processes, i.e, *centralized* or *distributed* scheme.

**[General consensus protocol]**

1 Each $p_i$ expresses the opinion. $p_i$ notifies all the processes of its opinion $pv_i$ which is named *pre-vote* of $p_i$. *This* step is *pre-voting.*
2 $p_i$ receives $pv_1, \ldots, pv_n$ from $p_1, \ldots, p_n$. $p_i$ makes a local decision based on $pv_1, \ldots, pv_n$. $p_i$ expresses the opinion $pv_i$ obtained by the local decision. Formally, $p_i$ obtains the *vote* $v_i = V_i(pv_1, \ldots, pv_n)$. *This* step is *voting.*
3 For the votes $v_1, \ldots, v_n$, a global decision $v = GD(v_1, \ldots, v_n)$ is obtained. *This* step is *global decision.*
4 $p_i$ obtains $v$. Based on $v$ and $v_1, \cdots, v_n$, $p_i$ makes the final local decision and obtains $d_i = LD_i(v_1, \ldots, v_n, v)$. *This* step is *final local decision.* □

Let $D$ be a set $\{d_1, \cdots, d_m, \bot, \top\}$ of values. $d_1, \cdots, d_m$ are *proper* values. $\bot$ means that it is not decided which one from $d_1, \cdots, d_m$ is taken. $\top$ means that any of $d_1, \cdots, d_m$ is allowed.

## 3 Dominant Relation on Values

A local state of each process $p_i$ is given as a tuple $\langle pv_i, v_i, d_i \rangle$ where $pv_i$ is the pre-vote, $v_i$ is the vote, and $d_i$ is the value finally decided by $p_i$. $p_i$ changes the local state on receipt of messages. For every state $\langle a, b, c \rangle$, $b = c = \bot$ if $a = \bot$, and $c = \bot$ if $b = \bot$. A state $\langle a, b, c \rangle$ is *transitable* if $b = \bot$ or $c = \bot$.

**[Definition]** For $a, b$, and $c \in D$, if $\langle a, b, \bot \rangle$ is transitable to $\langle a, b, c \rangle$, $c$ is *dominates* $b$ if $b \neq c$ (written as $b \prec c$). □

$a \prec b$ means that $p_i$ can change the pre-vote or vote from $a$ to $b$. $a \equiv b$ means that neither $a \succ b$ nor $a \prec b$. $a \succeq b$ means that $a \succ b$ or $a \equiv b$. $\langle 0, 0, 0 \rangle$ means that a process voting 0 aborts. $\langle 1, 1, \bot \rangle$ means that the process votes 1. $\langle 1, 1, \bot \rangle$ is transited to $\langle 1, 1, 1 \rangle$ if the process commits, $\langle 1, 1, 0 \rangle$ if the process aborts.

Thus, $D$ is partially ordered on $\prec$. Since $\top$ can be changed to any value in $D$, $\top$ a *bottom* of $D$, i.e. for every $d$ in $D$, $\top \prec d$. A proper value $d$ in $D$ is *minimal* in $D$ iff there is no proper value $d_k$ in $D$ such that $d_k \prec d$. If $D$ has only one minimal value $d$, *minimum.* $p_i$ can vote the minimum $d$ instead of voting $\top$.

**[Definition]** A value $d$ in $D$ is *maximal* in $D$ iff there is no value $d_k$ in $D$ such that $d \prec d_k$. □

If there is only one maximal value $d$ in $D$, $d$ is the *top* of $D$. If $p_i$ votes the maximal value $d$, $p_i$ never changes the mind because $d$ cannot be changed to any value. For every pair of $d_k$ and $d_h$ in $D$, $d_k \cup d_h$ denotes the least upper bound (*lub*) of $d_k$ and $d_h$.

Let $\langle a, b, c \rangle$ be a state. If $b$ is maximal in $D$, $c$ has to be $b$ because process voting $b$ cannot change the vote. Hence, if $b$ is maximal, $c = b$, i.e. $\langle a, b, b \rangle$.

**[Definition]** $\langle a, b, c \rangle$ with maximal $b$ is *maximal.*□

States which are not maximal are *transitable.* Here, let us consider a transition from a state $\langle a, b_1, c_1 \rangle$ into $\langle a, b_2, c_2 \rangle$ where $b_1$ is not maximal. If $b_1 \prec b_2$, or $b_1 = b_2$ and $c_1 \prec c_2$, $\langle a, b_1, c_1 \rangle$ can be transited into $\langle a, b_2, c_2 \rangle$. $\langle 1, 1, \bot \rangle$ can be transited into $\langle 1, 1, 0 \rangle$ and $\langle 1, 1, 1 \rangle$ while $\langle 0, 0, 0 \rangle$ cannot be transited. Processes which are in a transitable state after voting have to

---

*一般合意プロトコル
†立川 敬行, 矢羽田 千哲 , 滝沢 誠
‡東京電機大学

wait for the global decision. On the other hand, processes which are in a maximal state can terminate, because they made their final decisions already.

**[Example]** Let us consider that $n$ persons $p_1, \cdots, p_n$ would make agreement on where to go. Here, that there are three opinions, i.e. go skiing($K$), go swimming($W$), and go to hot springs($H$). $\alpha|\beta$ denotes $\alpha$ or $\beta$. $\alpha\beta$ denotes $\alpha$ and $\beta$. Figure 1 shows a lattice on $D$. □

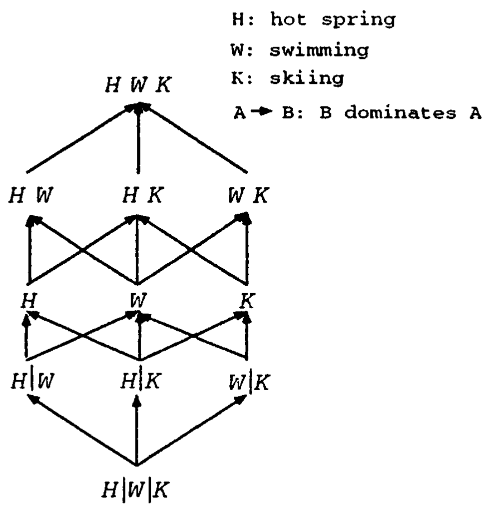


Figure 1: Lattice on $D$

## 4 Global Decision

There are the following kinds of global decisions:

1 Commitment decision : $GD(v_1, \ldots, v_n) = 1$ if every $v_i = 1$, $GD(v_1, \ldots, v_n) = 0$ if some $v_i = 0$ where $D = \{1, 0, \perp, \top\}$.

2 Majority-consensus decision on $v$: $GD(v_1, \ldots, v_n) = v$ if $|\{v_i | v_i = v\}| > \frac{n}{2}$, otherwise $GD(v_1, \ldots, v_n) = v_1 \cup \cdots \cup v_n$.

3 $\binom{n}{n}$-decision on $v$: $GD(v_1, \ldots, v_n) = v$ if every $v_i = v$ for every $i$, otherwise $GD(v_1, \ldots, v_n) = v_1 \cup \cdots \cup v_n$.

4 $\binom{n}{r}$-decision on $v$: $GD(v_1, \ldots, v_n) = v$ if $|\{v_i | v_i = v\}| \geq r$, otherwise $GD(v_1, \ldots, v_n) = v_1 \cup \cdots \cup v_n$.

5 Minimal-decision: $GD(v_1, \cdots, v_n) = v_1 \cup \cdots \cup v_n$.

6 Super-vote: $GD(v_1, \cdots, v_n) = v_i$ if $p_i$ has the highest priority.

## 5 Coordination Schemes

If one process $p_0$ named ***coordinator*** coordinates the cooperation of $p_1, \cdots, p_n$, it is ***centralized control***. If there is no centralized controller, it is ***distributed control***. Here, each $p_i$ sends the pre-vote $pv_i$ to $p_1, \ldots, p_n$. On receipt of $pv_1, \ldots, pv_n$, $p_i$ makes the local decision of $v_i = V_i(pv_1, \ldots, pv_n)$ by itself. $p_i$ sends the vote $v_i$ to $p_1, \ldots, p_n$. On receipt of $v_1, \ldots, v_n$, every $p_i$ makes the same global decision of $v = GD(v_1, \ldots, v_n)$. Then, $p_i$ makes the final local decision of $d_i = LD_i(v_i, \ldots, v_n, v)$. Each $p_i$ has the same $GD$ and makes the decision by itself on the basis of $GD$. $p_i$ can make the decision without waiting for the decision from the coordinator.

## 6 Extended Commitment Protocol

As presented before, each process can vote either 1(*Yes*) or 0(*No*) in the 2PC protocols. We would like to extend the commitment protocol so that each process can vote $\perp$ ( ***No_ idea***) and $\top$ (***Anyone-OK***). In the commitment protocol, each process $p_i$ may not be able to vote even if $p_i$ receives ***VoteReq*** from the coordinator $p_0$, e.g. $p_i$ is too heavy-loaded to vote. In such a case, $p_i$ can vote $\perp$.

**[Basic protocol]**

1 The coordinator $p_0$ sends *VoteReq* to $p_1, \ldots, p_n$.

2 On receipt of *VoteReq* from $p_0$, each $p_i$ sends 1, 0, $\perp$, or $\top$ to $p_0$.

3 If $p_0$ receives 1 from all the processes and $p_0$ would like to commit, $p_0$ sends *Commit* to $p_1, \ldots, p_n$. If $p_0$ receives 0 from at least one process or $p_0$ would not like to commit, $p_0$ sends *Abort* to all the processes voting 1, $\perp$, or $\top$. If $p_0$ receives $\top$ from all the processes, every $p_i$ obeys $p_0$'s decision.

4 Here, some $p_i$ votes $\perp$. If all the decided processes vote 1, $p_0$ sends *Commitable* to the undecided processes.

5 If $p_i$ votes $\perp$, on receipt of *Commitable*, $p_i$ sends 1 to $p_0$ if $p_i$ could commit, 0 to $p_0$ if $p_i$ could abort. $p_i$ sends $\perp$ to $p_0$ again if $p_i$ still could neither decide 1 nor 0.

6 If $p_0$ could not receive 1 or 0 from all the undecided processes after sending *Commitable* $m(\geq 1)$ times, $p_0$ sends *Abort* to all the processes.

7 After voting 1, $\perp$, or $\top$ if $p_i$ receives *Abort* from $p_0$, $p_i$ aborts. After voting $\top$ and 1, if $p_i$ receives *Commit* from $p_0$, $p_i$ commits. □

## 7 Concluding Remarks

This paper discusses general framework of various consensus protocols. The general consensus protocol is composed of four steps, i.e. pre-voting, voting, global decision, and final local decision. We have described various consensus protocols in terms of the model. By composing the procedures for pre-voting, voting, global decision, and final local decision, we can make the consensus protocols required in the applications.

## Reference

[1] Bernstein, P. A., Hadzilacos, V., and Goodman, N., "Concurrency Control and Recovery in Database Systems," ***Addison-Wesley Publishing Company***, 1987, pp.222-261.

[2] Fischer, J. M., Lynch, A. N., and Paterson, S. M., "Impossibility of Distributed Consensus with One Faulty Process," *Journal of ACM*, Vol.32, No.2, 1985, pp.374-382.

[3] Turek, J. and Shasha, D., "The Many Faces of Consensus in Distributed Systems," *Distributed Computing Systems, IEEE Computer Society Press*, 1994, pp.83-91.

[4] Yahata, C. and Takizawa, M., "Cooperation of Multiple Agents to Access Multiple Database Systems," *Proc. of the IEEE ICPADS'93*, 1993, pp.243-250.

[5] Yahata, C., Sakai, J., and Takizawa, M., "Generalization of Consensus Protocols," *Proc. of the IEEE ICOIN-9*, 1994, pp.419-424.