# Object-Based Locking scheme for Distributed Replicas *

3 A a − 3

Kyouji Hasegawa and Makoto Takizawa †
Tokyo Denki University ‡
Email {kyo, taki}@takilab.k.dendai.ac.jp

## 1 Introduction

Distributed systems are composed of multiple objects. Each object supports more abstract operations than the low-level read and write operations. The objects are replicated to increase the performance, reliability, and availability. We propose a novel object-based locking (OBL) method based on the abstract operations, where objects are first locked in weaker modes and finally locked in modes of operations. The coterie method [1] is used to lock the replicas. Here, the number of the replicas locked is decided based on the access frequency and the strength of the lock mode. If two operations do not conflict, the subsets of the replicas locked by the operation do not intersect even if the operation change the replicas. We propose a version vector to identify the newest replica in the subset of the replicas locked.

## 2 Replicated Objects

A distributed system is composed of multiple objects $o_1, \ldots, o_n$ which are cooperating by exchanging messages in the communication network. The communication network supports every pair of objects $o_i$ and $o_j$ with a reliable, bidirectional channel $\langle o_i, o_j \rangle$.

Each object $o_i$ supports a set $\tau_i$ of operations $op_{i1}, \ldots, op_{il_i}$ for manipulating $o_i$. $o_i$ is encapsulated so that $o_i$ can be manipulated only through the operations supported by $o_i$. Let $op(s)$ denote a state obtained by applying $op$ to a state $s$ of $o_i$. An operation $op_{ij}$ is *compatible* with $op_{ik}$ iff $op_{ij} \circ op_{ik}(s_i) = op_{ik} \circ op_{ij}(s_i)$ for every state $s_i$ of $o_j$. $op_{ij}$ *conflicts* with $op_{ik}$ unless $op_{ij}$ is compatible with $op_{ik}$. If $op_{ij}$ conflicts with $op_{ik}$, the state obtained by computing $op_{ij}$ and $op_{ik}$ is independent of the computation order. If some operations conflicting with $op_i$ are being computed on $o_i$, $op_i$ has to wait until the operations complete. The interleaved and parallel computation of operations has to be serializable.

On receipt of the request $op_i$, $o_i$ is locked in an *lock mode* $\mu(op_i)$ in order to make the computation serializable. Here, let $M_i$ be a set of lock modes of $o_i$. *Dlock* and *Wlock* denote the lock modes of *Deposit* and *Withdraw* of the bank object $B$, respectively. *Dlock* and *Wlock* are compatible. After computing $op_i$, the lock $\mu(op_i)$ of $o_i$ is released. Here, let $C_i(m)$ be a set of modes with which $m$ conflicts in $o_i$, i.e. $C_i(m) = \{m' \mid m' \text{ conflicts with } m\}$. In this paper, we assume that the compatibility relation among the modes are symmetric. Hence, $m$ is in $C_i(m')$ for every $m'$ in $C_i(m)$. Suppose that an object $o_i$ is locked in a mode $m$ and $op_i$ would be computed on $o_i$. If $\mu(op_i)$ is *compatible* with $m$, $op_i$ can be started to be computed on $o_i$. Otherwise, $op_i$ has to wait until the lock of the mode $m$ is released.

In order to increase the reliability, availability, and performance, an object $o_i$ is replicated in a collection $\{ o_i^1, \ldots, o_i^{r_i} \}$ $(r_i \geq 1)$ of replicas, where $o_i^j$ is a replica of $o_i$. Each $o_i^j$ supports the same data and operations as the other replicas. Let $R(o_i)$ be $\{ o_i^1, \ldots, o_i^{r_i} \}$.

## 3 Object-Based Locking

### 3.1 Weighted lock modes

The replicas of $o_i$ are locked in a mode $\mu(op_i)$ in $M_i$ before $op_i$ is computed on $o_i$. If $o_i$ is locked by modes conflicting with $\mu(op_i)$, $op_i$ has to block. The more modes conflicting with $\mu(op_i)$, the larger probability $op_i$ blocks.

**[Definition]** For every pair of $m_1$ and $m_2$ in $M_i$, $m_1$ is *more restricted* than $m_2$ iff $C_i(m_1) \supseteq C_i(m_2)$. □

Some $m_1$ may be more frequently used than $m_2$. Let $\varphi(m)$ be the usage frequency of a mode $m$. The frequencies of the modes in $o_i$ are normalized to be $\sum_{m \in M_i} \varphi(m) = 1$. The *weighted strength* $||C_i(m)||$ of $m$ is defined to be $\sum_{m' \in C_i(m)} \varphi(m')$ $(\leq 1)$.

**[Definition]** For every pair of $m_1$ and $m_2$ in $M_i$, $m_1$ is *stronger* than $m_2$ $(m_1 \succ m_2)$ iff $m_1 \in C_i(m_2)$, $m_2 \in C_i(m_1)$, and $||C_i(m_1)|| \geq ||C_i(m_2)||$. □

Let $op_1$ and $op_2$ be operations of modes $m_1$ and $m_2$ in an object $o_i$, respectively. Let us consider a *blocking* probability that $op_i$ waits for the release of the lock conflicting with $op_i$. If $||C_i(m_1)|| \geq ||C_i(m_2)||$, $op_1$ has a higher blocking probability than $op_2$. The modes in $M_i$ are partially ordered by the strength relation "$\preceq$".

### 3.2 Locking protocol

In this paper, we discuss an *object-based locking* (*OBL*) where the replicas are first locked in a mode $\nu(op_i)$ weaker than $\mu(op_i)$ and finally locked in $\mu(op_i)$ $(\nu(op_i) \preceq \mu(op_i))$. $\nu(op_i)$ may not be the weakest, i.e. not minimal and $\mu(op_i)$ may not be the strongest, i.e. not maximal. If $op_i$ is used more frequently, $\nu(op_i)$ can get stronger. In addition, the number $Q(op_{ij})$ of the replicas in $R_i$ depends on the strength of $op_i$. That is, more transactions can commit than the traditional *read* and *write* modes.

Suppose that a transaction $T$ invokes an operation $op_{it}$ on $o_i$. First, some number of the replicas in $R(o_i)$ are locked in a mode $\nu(op_{it})$ which is not stronger than $\mu(op_{it})$. Let $q(op_{it})$ $(\leq r_i)$ be the number of the replicas locked by $op_{it}$ before $op_{it}$ is computed. Unless all of $q(op_{it})$ replicas can be locked, $op_{it}$ is aborted. If all of $q(op_{it})$ replicas could be locked, $op_{it}$ is computed on the replicas as presented before. When $T$ would commit, some number $Q(op_{it})$ of the replicas are locked in $\mu(op_{it})$. $q(op_{it}) \leq Q(op_{it})$ and $\nu(op_{it}) \preceq \mu(op_{it})$.

We discuss how to decide the numbers $q(op_{it})$ and $Q(op_{it})$ of the replicas to be locked and the lock mode $\nu(op_{it})$. The more replicas are locked, the more communication and computation are required. Hence, the more frequently $op_{it}$ is invoked, the fewer replicas are

---

*分散レプリカのためのオブジェクトベースロック方式
†長谷川 享二, 滝沢 誠
‡東京電機大学

locked. $q(op_{it})$ and $Q(op_{it})$ are decided depending on the probability that $op_{it}$ conflicts with other operations of $o_i$. There are the following constraints on $q$ and $Q$:

(1) If $\mu(op_{it})$ conflicts with $\mu(op_{iu})$, $Q(op_{it}) + Q(op_{iu}) > r_i$ and $Q(op_{it}) + \nu(op_{iu}) > r_i$.

(2) $Q(op_{it}) \geq Q(op_{iu})$ iff $||C_i(op_{it})|| \geq ||C_i(op_{iu})||$.

Here, suppose that $op_{it}$ locks $Q_{it}$ replicas in $R(o_i) = \{o_i^1, \ldots, o_i^{r_i}\}$. Suppose that two operations $op_{it}$ and $op_{iu}$ are invoked and conflict in $o_i$. The probability that both $op_{it}$ and $op_{iu}$ can lock the replicas is given $1 - [Q_{it} \cdot \varphi(op_{it}) / r_i]\ [Q_{iu} \cdot \varphi(op_{iu}) / r_i]$. $C_i(op_{it})$ is a set of operations conflicting with $op_{it}$ in $o_i$. Here, the probability $P(op_{it})$ that $op_{it}$ can lock $Q_{it}$ replicas is $1 - \prod_{op_{iu} \in C_i(op_{it})} [Q_{iu} \cdot \varphi(op_{iu}) / r_i]$. $Q(op_{it})$ can be decided so that $P_2(op_{it})$ is the minimum.

### 3.3 Version vector

We introduce a *version vector* to identify the newest replica in a set of replicas locked. Each replica $o_i^h$ has a *version vector* $V_{it}^h = (V_{i1}^h, \ldots, V_{il_i}^h)$. Each $V_{it}^h$ shows the version of $o_i^h$ with respect to an operation $op_{it}$ of $o_i$. $V_{it}^h$ is a tuple $\langle v_{it}^h, b_{it}^h \rangle$. $v_{it}^h$ is a *version number* of $o_i^h$ with respect to $op_{it}$. $v_{it}^h$ is incremented by one each time $op_{it}$ is computed an $o_i^h$ and $op_{it}$ is a changing operation. $b_{it}^h$ is a bit map $\langle b_{it}^{h1}, \ldots, b_{it}^{hr_i} \rangle$ showing to which replica $op_{it}$ is issued. $b_{it}^{hk}$ is 1 if $op_{it}$ is issued to $o_i^k$, otherwise 0 ($k = 1, \ldots, r_i$) [Figure 1].

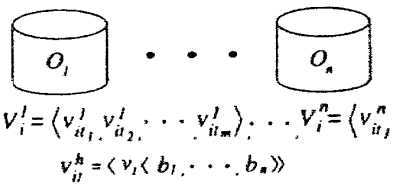


Figure 1: Version Vector.

We discuss relations among the version vectors. Let $b^h$ and $b^k$ be bit maps for $o_i^h$ and $o_i^k$, respectively. $b^h$ is included in $b^k$ ($b^h \subseteq b^k$) iff $b^{kj} = 1$ if $b^{hj} = 1$ for $j = 1, \ldots, r_i$. $b^h \cup b^k$ shows $\langle b^1, \ldots, b^{r_i} \rangle$ where $b^j = 1$ if $b^{hj} = b^{kj} = 1$, otherwise 0 for $j = 1, \ldots, r_i$.

- $V_{it}^h \leq V_{it}^k$ iff $v_{it}^h \leq v_{it}^k$ and $b_{it}^h \subseteq b_{it}^k$.
- $V_i^h \leq V_i^k$ iff $V_{it}^h \leq V_{it}^k$ for every operation $op_{it}$.

If $b_{it}^h \cap b_{it}^k \neq \phi$, there is no ordering between $V_{it}^h$ and $V_{it}^k$ even if $v_{it}^h \leq v_{it}^k$ or $v_{it}^k \leq v_{it}^h$. In a subset $N$ of replicas of $o_i$ ($N \subseteq R(o_i)$), $V_i^h$ is *maximal* iff there is no version vector $V_i^k$ of $o_i^k$ in $N$ such that $V_i^k \geq V_i^h$.

We define a following operation $\cup$ for a pair of vector elements $V_{it}^h$ and $V_{it}^k$ on $op_{it}$ :

- $V_{it}^h \cup V_{it}^k = \begin{cases} V_{it}^k & \text{if } V_{it}^h \leq V_{it}^k. \\ V_{it}^h & \text{if } V_{it}^h \geq V_{it}^k. \\ \langle v_{it}^h + v_{it}^k, b_{it}^h \cup b_{it}^k \rangle & \text{otherwise.} \end{cases}$
- $V_i^h \cup V_i^k = (V_{i1}^h \cup V_{i1}^k, \ldots, V_{il_i}^h \cup V_{il_i}^k)$.

**[Definition]** A version vector $V_i^h$ is *equivalent* with $V_i^k$ ($V_i^h \equiv V_i^k$) iff $V_{iv}^h = V_{iv}^k$ for every changing operation $op_{iv}$ conflicting with every pair of compatible operations $op_{it}$ and $op_{iu}$. □

If $V_i^h$ and $V_i^k$ are equivalent, $o_i^h$ and $o_i^k$ can get the same state by computing compatible operations which are not computed on the replicas.

We discuss a locking protocol by using the version vector. Here, suppose that $op_{it}$ is issued to an object $o_i$. Here, let $Q_{it}$ show a quorum number of $op_{it}$, and $N_{it}$ be a set of replicas to be locked by $op_{it}$. $o_i^h$ has an operation log $l_{it}^h$ for each changing operation $op_{it}$. The instances of $op_{it}$ computed on $o_i^h$ are stored in $l_{it}^h$.

**[Locking protocol]** $op_{it}$ is issued to every replica in $N_{it}$.

(1) All the replicas in $N_{it}$ are locked in a mode $\mu(op_{it})$. Unless succeeded in locking the replicas, $op_{it}$ aborts.

(2) One replica $o_i^h$ is selected in $N_{it}$ where $V_i^h = \cup \{ V_i^k \mid o_i^k \in N_{it} \}$. If such a replica is not found, one maximal replica $o_i^h$ is selected in $N_{it}$.

(3) If $V_i^h$ is not maximum in $N_{it}$, $v_{iu}$ of operations $op_{iu}$ computed on $o_i^h$ are obtained from the log $l_{iu}^h$ in $o_i^h$ if $b_{iu}^h \cap b_{iu}^h = \phi$ and $v_{iu} \neq 0$. The operation computed on $o_i^h$ for every $o_i^h$ in $N_{it}$.

(4) $op_{it}$ is computed on $o_i^h$.

(5) If $op_{it}$ is a changing operation, $o_i^k$ sends the state and $V_i^h$ to every other replica $o_i^k$ in $N_{it}$. On receipt of the state, $o_i^k$ changes the state by itself and $V_i^k := V_i^h$.

(6) For every version element $V_{it}^h = \langle v_{it}^h, b_{it}^h \rangle$, $v_{it}^{hj} := 0$ for $j = 1, \ldots, r_i$ if $b_{it}^h = \langle 1, \ldots, 1 \rangle$. All operations in $l_{it}^h$ are removed.

(7) $o_i^h$ sends a reply of $op_{it}$ back. □

When every bit $b_{it}^{hj}$ gets 1, i.e. $b_{it}^h = \langle 1, \ldots, 1 \rangle$, the version number $v_{it}^h$ and the bit map $b_{it}^h$ are initialized. Thus, $v_{it}^h$ shows how many instances of $op_{it}$ are computed on $o_i^t$.

**[Proposition]** For every changing operation $op_{it}$, $b_{it}^h = b_{it}^k$ and $v_{it}^h = v_{it}^k$ if $b_{it}^h \cap b_{it}^k \neq \phi$. □

If $b_{it}^h \cap b_{it}^k = \phi$ and $v_{it}^h > 0$ or $v_{it}^k > 0$ for a changing operation $op_{it}$, a sequence $s^h$ of instances of $op_{it}$ computed on $o_i^h$ is different from a sequence $s^k$ in $o_i^k$. $s^h$ and $s^k$ include $v_{it}^h$ and $v_{it}^k$ instances of $op_{it}$, respectively. In order for $o_i^h$ and $o_i^k$ to be consistent, the sequences $s^k$ and $s^h$ are required to be computed on $o_i^h$ and $o_i^k$, respectively.

The following properties hold.

**[Theorem]** For every operation $op_{it}$, $\cup \{ V_i^k \mid o_i^k \in N_{it} \} \geq V_i^h$ for every replica $o_i^h$. □

This theorem means that there is at least one newest replica in an intersection of $N_{it}$ and $N_{iu}$ for every pair of conflicting operations $op_{it}$ and $op_{iu}$.

## 4 Concluding Remark

This paper has discussed the *object-based locking* (*OBL*) protocol on the replicas of the objects. The objects support more abstract operations than the traditional *read* and *write* operations.

## References

[1] Garcia-Molina, H., and Barbara, D., "How to assign votes in a distributed system," *Journal of ACM, Vol 32, No.4*, pp. 841-860, 1985.