Ohkawara et al. Human-centric Computing and Information Sciences 2012, 2:11
http://www.hcis-journal.com/content/2/1/11

Human-centric Computing and Information Sciences
a SpringerOpen Journal

RESEARCH Open Access

Quorums-based Replication of Multimedia Objects in Distributed Systems

Tadateru Ohkawara[1*], Ailixier Aikebaier[3], Tomoya Enokido[2] and Makoto Takizawa[1]

*Correspondence: tadateru.ohkawara@gmail.com
[1] Department of Computer and Information Science, Seikei University, 3-3-1 Kichijoji-kitamachi, Musashino-shi, Tokyo 180-8633, Japan
Full list of author information is available at the end of the article

Abstract

Background: Multimedia objects like music and movies are distributed to peers through downloading and caching in peer-to-peer (P2P) overlay networks. In this paper, we consider multimedia objects which are characterized in terms of not only data structure but also quality of service (QoS) like frame rate and number of colours. For example, there are a pair of replicas o_i and o_j of a fully coloured movie object o. Here, a content of a replica o_i is changed by adding a subobject but another replica o_j is not changed. On the other hand, the colour of the replica o_j is changed with monochromatic one but not in the replica o_i. This means, the replica o_i is newer than the replica o_j with respect to the content but is older than o_j with respect to QoS. Thus, replicas of a multimedia object are partially ordered in terms of newness of not only content but also QoS parameters.

Methods: In traditional quorum-based (QB) protocols, replicas are totally ordered just in terms of newness of content. We discuss a multimedia quorum-based (MQB) protocol to synchronize multiple replicas to make consistent on the basis of the newness-precedent relation of replicas. Here, the replicas are ordered in vectors of version counters of content and QoS parameters. Every replica in a quorum is not updated for QoS operations to reduce the communication overhead. We evaluate the MQB protocol in terms of communication overhead and show the communication overhead can be reduced in the MQB protocol compared with the traditional QB protocol.

Conclusions: We discussed the multimedia quorum-based (MQB) protocol to keep replicas of a multimedia object mutually consistent. We evaluated the MQB protocol in terms of the total volume of data transmitted among the replicas. Then, we showed the total amount of data transmitted can be reduced in the MQB protocol compared with the traditional quorum-based (QB) protocol.

Background

In scalable distributed systems like cloud computing systems [1] and peer-to-peer (P2P) overlay networks [2] systems, resource objects like databases and files are replicated and distributed to multiple server computers in order to increase the performance, reliability, and availability. In P2P overlay networks, objects, especially multimedia objects like movies are in nature autonomously distributed through peer-to-peer communication. There are many discussions on how to maintain the mutual consistency of multiple replicas like the two-phase locking (2PL) [3], read-one-write-all (ROWA) [4], and quorum-based (QB) [5] protocols. In the 2PL protocol, all the replicas are first locked before they are read and write. On the other hand, only one replica is locked for read while every replica is locked for write in the ROWA protocol. In the QB protocol, subsets of the replicas for read and write operations are referred to as *read quorum* Q_r and *write quorum* Q_w, respectively. Every pair of read and write quorums include at least one common replica. Only if every replica in a quorum could be locked, a transaction can manipulate the replicas in the quorum. In Cassandra [6], the synchronization scheme based on the quorum concept [4] is adopted.

Various types of objects including multimedia objects are distributed in P2P overlay networks. Multimedia objects are characterized in terms of quality of service (QoS) like frame rate and number of colours in addition to the contents. Thus, not only the content but also QoS parameters of an object are manipulated. For example, suppose there are three replicas o_1, o_2, and o_3 of a fully-coloured movie object o in a quorum Q. A *scene* subobject is added to the replica o_2. On the other hand, the colour of another replica o_3 is changed with monochromatic one. The replica o_2 is newer than the replica o_3 in terms of the content while the replica o_3 is newer than the replica o_2 in terms of number of colours. Thus, replicas are partially ordered in terms of *newness* of not only content but also QoS parameters in a quorum. The partially ordering *newness-precedent* relation $\preceq$ among replicas of a multimedia object is defined in the paper [7]. On the other hand, replicas of a file object are totally ordered just in terms of newness of content in the traditional QB protocol. Here, there is no newest replica in the quorum Q. A *complete* quorum includes a newest replica. A newest replica should be a monochromatic replica with the *scene* subobjects in the quorum Q. The replicas o_2 and o_3 can be made the newest one by degrading colours and adding the *scene* subobjects, respectively. Thus, even if a quorum is not complete, some replica o_i might be made the newest by applying operations with data held in other replicas o_i in the quorum. An incomplete quorum which can be complete is referred to as *completable*. The replica Q is *completable*. We discuss how to obtain the newest replica in an incomplete but *completable* quorum. In the traditional QB protocol, every quorum is complete. However, multimedia quorums can be completable.

We propose a *multimedia quorum-based* (*MQB*) protocol in this paper. Here, each replica of a multimedia object holds the vector of counters, where there is one counter for each of the content and QoS parameters. If a transaction issues a read operation *op*, the transaction selects the newest replica o_i in a quorum Q_{op}. If not found, one replica o_i is selected and is made newest by obtaining operations and data which are not performed on the replica o_i through communicating with other replicas. Then, the transaction reads the replica o_i in the quorum Q_{op}. The content and QoS parameters of every replica are updated to be the newest. Here, computation and communication resources are consumed to update every replica in the quorum Q_{op}. In order to reduce the computation and communication overheads, every quorum is tried to be *completable*, that is, only the counter vector of every replica is updated in the quorum Q_{op} but all the replicas themselves are not updated. We evaluate the MQB protocol compared with the QB protocol and show the communication overhead in the MQB protocol can be reduced with the QB protocol.

In section “Method”, we discuss the newness-precedent relations on replicas of multimedia objects. In section “Evaluation”, we discuss the multimedia quorum-based (MQB) protocol to maintain the mutually consistency of replicas. In section “Conclusions”, we evaluate the MQB protocol compared with the QB protocol in terms of communication overheads.

Method

Partially Ordering Relations of Multimedia Replicas

Multimedia objects

A multimedia object o is characterized in terms of not only content parameter $o.c$ but also quality of service (QoS) parameters $o.Q$. A content $o.c$ shows data structure, i.e. the *part_of* structure of subobjects. QoS $o.Q$ is specified in a tuple of QoS parameters $\langle q_1, ..., q_l \rangle$ ($l \geq 0$). Frame rate and resolution are examples of QoS parameters. Thus, each replica o_i of an object o is specified in a pair of the content $o_i.c$ and QoS parameters $o_i.Q$. It is noted a traditional replica o_i like a text object is just specified in terms of a content $o_i.c$. The content $o_i.c$ in a replica o_i is manipulated in a *content* operation like delete-subobject while QoS parameters. $o_i.Q$ is manipulated in a *QoS* operation like change-colour.

In a QoS parameter *frame rate* (*fr*), 40[fps] is richer than 20[fps]. Thus, for a pair of values x and y of a QoS parameter q_k, y is *richer* than x ($x \rightarrow y$) (y is *poorer* than x) iff y includes more volume of data than x. For example, $20 \rightarrow 40$[fps]. Let c_1 and c_2 be a pair of contents of an object o. A content parameter c_2 is *richer* than a content c_1 (c_1 is *poorer* than c_2) ($c_1 \rightarrow c_2$) if c_1 is a component of c_2. A value x can be obtained by just removing some data from a value y if $x \rightarrow y$. However, if $y \rightarrow x$, the value x cannot be obtained without adding any data to the value y. For example, a fully coloured movie object can be degraded to a monochromatic one by just removing the colour data. However, we have to add colour data to a monochromatic object in order to change with a coloured one.

A scheme of an object o is written in a tuple $\langle p_0, p_1, ..., p_l \rangle$ where the first parameter p_0 stands for a content parameter $o.c$ and the kth parameter p_k indicates a QoS parameter $o.q_k$ ($k = 1, ..., l$). $o.p_i$ shows a parameter p_i of an object o ($i = 0, 1, ..., l$).

Newness-precedent relation

Let O be a set $\{o_1, ..., o_n\}$ of replicas of an object o ($n \geq 1$) in the system. Here, the content parameter $o_j.c$ of a replica o_j is *newer* than the content $o_i.c$ of another replica o_i ($o_i.c \prec o_j.c$) iff (if and only if) the content parameter $o_j.c$ is updated, e.g. some subobject is deleted but $o_i.c$ is not updated. A content $o_i.c$ *precedes* a content $o_j.c$ ($o_i.c \preceq o_j.c$) iff $o_i.c \prec o_j.c$ or $o_i.c = o_j.c$. A QoS parameter $o_j.q_k$ of a replica o_j is *newer* than $o_i.q_k$ ($o_i.q_k \prec o_j.q_k$) iff the parameter q_k is changed in the replica o_i but is not in the replica o_j. For example, a monochromatic replica o_j is obtained by changing the QoS parameter cl (colour) of a fully coloured replica o_i. Here, the QoS parameter $o_j.cl$ is newer than $o_i.cl$ ($o_i.cl \prec o_j.cl$). A QoS parameter $o_i.q_k$ *precedes* $o_j.q_k$ with respect to newness ($o_i.q_k \preceq o_j.q_k$) iff $o_i.q_k \prec o_j.q_k$ or $o_i.q_k = o_j.q_k$.

Replicas in the replica set O are partially ordered in the newness-precedent relation $\preceq$ ($\subseteq O^2$). A replica o_i *precedes* a replica o_j with respect to newness ($o_i \preceq o_j$) iff $o_i.c \preceq o_j.c$ and $o_i.q_k \preceq o_j.q_k$ for every QoS parameter q_k (k =0, 1, ..., l). A replica o_i is *equivalent* with a replica o_j ($o_i \equiv o_j$) iff $o_i.c = o_j.c$ and $o_i.q_k = o_j.q_k$ for every QoS parameter q_k. A replica o_i is newer than another replica o_j ($o_i \prec o_j$) iff $o_i \preceq o_j$ but $o_i \not\equiv o_j$. A replica o_i is *uncomparable* with a replica o_j ($o_i \mid o_j$) iff neither $o_i \preceq o_j$ nor $o_j \preceq o_i$. In the traditional quorum-based (QB) protocols [8] [3], replicas in the replica set O are totally ordered, i.e. for every pair of replicas o_i and o_j in the replica set O, either $o_i \equiv o_j$ or $o_j \prec o_i$, that is, $o_i \preceq o_j$. On the other hand, replicas of a multimedia object o are partially ordered in the newness-precedent relation $\preceq$. For example, the content parameter $o_j.c$ of a replica o_j is newer than the content $o_i.c$ ($o_i.c \prec o_j.c$) while some QoS parameter $o_i.q_k$ is newer than $o_j.q_k$ ($o_j.q_k \prec o_i.q_k$). Here, a pair of replicas o_i and o_j are uncomparable ($o_i \mid o_j$).

Newest replica in a quorum

In terms of the newness precedent relation $\preceq$, we define the least upper bound (lub) and greatest lower bound (glb) of replicas. $o_i \cup o_j$ shows a least upper bound (lub) of a pair of replicas o_i and o_j, which is a replica o_k such that $o_i \preceq o_k$ and $o_j \preceq o_k$ and there is no replica o_h such that $o_i \preceq o_h \preceq o_k$ and $o_j \preceq o_h \preceq o_k$. $o_i \cap o_j$ indicates a greatest lower bound (glb) of replicas o_i and o_j, which is a replica o_k such that $o_k \preceq o_j$ and $o_k \preceq o_j$ and there is no replica o_h such that $o_k \preceq o_h \preceq o_i$ and $o_k \preceq o_h \preceq o_j$. Let Q be a quorum of replicas o_1, ..., o_n. $\cup Q$ indicates the lub $o_1 \cup \ldots \cup o_n$ of every replica in the set Q, i.e. top replica of the set Q, which shows the newest replica in the quorum Q. A replica o_i is *maximal* iff there is no replica o_j in the quorum Q such that $o_i \preceq o_j$. Max_Q shows a subset of maximal replicas in the quorum Q. A quorum Q is referred to as *complete* iff the lub $\cup Q$ exists in the quorum Q. Here, a quorum Q is referred to as *complete* iff there is a top replica $\cup Q$ in the quorum Q.

Figure 1 shows a quorum Q of five replicas o_1, o_2, o_3, o_4, and o_5 of a multimedia object o, $Q = \{o_1, o_2, o_3, o_4, o_5\}$. Each replica o_i has a content parameter (c) and QoS parameter *colour* (cl), i.e. $o_i = \langle c, cl \rangle$. Here, a directed edge $o_i \rightarrow o_j$ shows the newness-precedent relation $o_i \preceq o_j$. A replica o_1 is composed of three fully coloured subobjects, blueberry b, orange r, and strawberry s. The content parameter c is $\langle b, r, s \rangle$, $o_1 = \langle \langle b, r, s \rangle, fully-colour\ (fc) \rangle$. Suppose initially $o_1 \equiv o_2 \equiv o_3 \equiv o_4 \equiv o_5$.

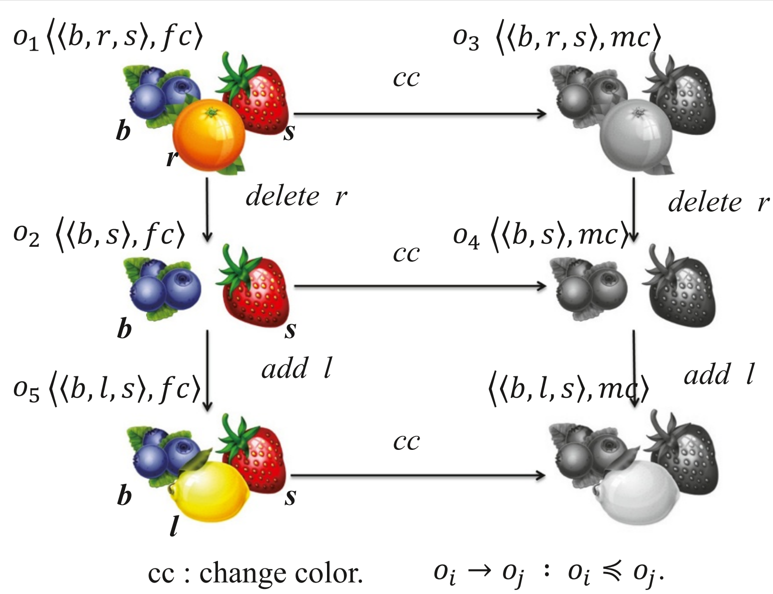

Figure 1 Replicas of a multimedia object.

1 In a content operation *delete*, a subobject r is removed in the replicas o_2 and o_5. Here, $o_2 \equiv o_5 = \langle\langle b, s\rangle, fc\rangle$. For a pair of the replicas o_2 and o_5, the content parameters $o_2.c$ and $o_5.c$ are newer than $o_1.c$ ($o_1.c \preceq o_2.c$) and $o_1.c \preceq o_5.c$ while the QoS parameters $o_2.cl$ are $o_5.cl$ are the same as $o_1.cl$, $o_1.cl = o_2.cl$. Hence, the replica o_1 precedes the replica o_2 ($o_1 \preceq o_2$) and $o_1 \preceq o_5$. Similarly, $\{o_3, o_4\} \preceq o_2$ and $\{o_3, o_4\} \preceq o_5$. $\{o_1, o_3, o_4\} \preceq o_5$.
2 Next, a pair of the replicas o_3 and o_4 are changed by degrading with monochromatic (mc) ones by a *down-colour* (dc) operation; $o_3 \equiv o_4 = \langle\langle b, r, s\rangle, mc\rangle$. Here, the QoS parameters cl of the replicas o_1 and o_2 are newer than $o_1.cl$ ($o_1.cl \preceq o_3.cl$) and $o_1.cl \preceq o_4.cl$ while $o_3.c = o_4.c = o_1.c = \langle b, r, s\rangle$. The replica o_1 precedes the replica o_3 ($o_1 \preceq o_3$) and $o_1 \preceq o_3$. Similarly, $o_1 \preceq o_4$. Here, a pair of the replicas o_2 and o_3 are uncomparable ($o_2 \mid o_3$). Similarly, $o_5 \mid o_3$, $o_2 \mid o_4$, and $o_5 \mid o_4$.
3 Then, the subobject orange r is deleted in the replica o_4; $o_4 = \langle\langle b, s\rangle, mc\rangle$. Here, $o_2 \cup o_3 = o_4$.
4 A lemon subobject l is added to the replica o_5. Here, $o_5 = \langle\langle b, l, s\rangle, fc\rangle$. Here, there is no lub $o_4 \cup o_5$ in the quorum Q. A pair of the replicas o_4 and o_5 are maximal and the replica o_1 is a bottom of the quorum Q, i.e. $o_1 = o_2 \cap o_3$ as shown in Figure 2. There is no top replica $\cup Q$ in the quorum Q. The lub $o_4 \cup o_5$ shows the newest replica for the replicas in the quorum Q. By changing the colour cl of the replica o_5 into mc or deleting the orange r from the replica o_4, a top replica $\langle\langle b, s\rangle, mc\rangle$ ($= o_4 \cup o_5$) can be obtained.

In Figure 2, the vertical axis shows the newness of the content parameter c. The content $c = \langle b, r, s\rangle$ is changed with the content $\langle b, s\rangle$, i.e. $\langle b, r, s\rangle \preceq \langle b, s\rangle$. Hence, $o_1.c$ ($= \langle b, r, s\rangle$) includes a larger volume of data than $o_2.c$ ($= \langle b, s\rangle$), i.e. $o_1.c \supseteq o_2.c$. The horizontal axis indicates the newness of the QoS parameter cl. The QoS parameter cl is changed from fc to mc, i.e. $fc \preceq mc$. The QoS parameter $o_1.cl (= fc)$ includes more volume of data than $o_2.cl$ ($= mc$). Thus, $\langle b, s\rangle \rightarrow \langle b, r, s\rangle$ and $mc \rightarrow fc$. A replica o_j is richer than a replica o_i ($o_i \rightarrow o_j$) iff $o_i.c \rightarrow o_j.c$ and $o_i.q_k \rightarrow o_j.q_k$ for every QoS parameter q_k. This means, the replica o_i can be obtained by deleting data and degrading the richer replica o_j to a less QoS one.

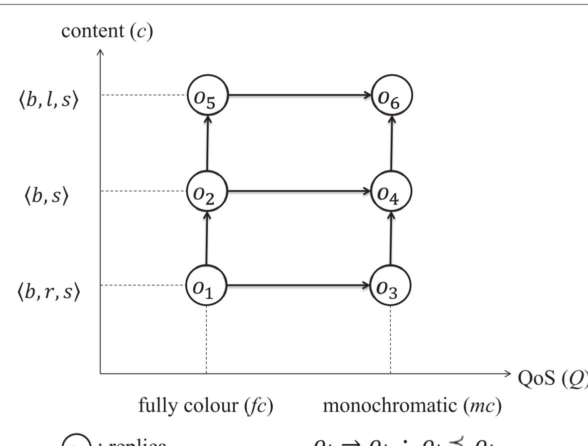

Figure 2 Newness precedence of content and QoS.

In the QB protocol [9], there is at least one newest replica o_i in a *read* quorum Q_r.A transaction reads the newest replica o_i in the quorum Q_r.A transaction writes every replica in a *write* quorum Q_w. Then, every replica in the quorum Q_w gets the newest. In the QB protocol, each replica has the version counter. The version counter of every replica in a write quorum Q_w is incremented so that the version counter of each replica in the quorum Q_w shows the maximum value in the replica set O. Hence, the write quorum Q_w includes at least one newest replica whose version counter is the maximum. A replica whose version counter is the maximum is the newest. Every replica is required to be complete in the QB protocol.

For a pair of values x and y, $max(x, y)$ is defined to be the value x if $y \preceq x$. Here, $max(x, y)$ = $max(y, x)$. $max(x, y) = \perp$ if $x \mid y$. The *upgrade* operation $o_i + o_j$ on a pair of replicas o_i and o_j a replica o_h such that $o_h.c = max(o_i.c,\ o_j.c)$ and $o_h.q_k = max(o_i.q_k,\ o_j.q_k)$ for every QoS parameter q_k. $+Q$ shows $o_1 + ... + o_n$ for a quorum Q (= $\{o_1,\ ...,\ o_n\}$). $+Q$ shows a replica o which may not be in a quorum Q but which can be the top replica $\cup Q$. Here, let m_c be max $\{o_i.c \mid o_i \in Q\}$ and m_{q_k} be max $\{o_i.q_k \mid o_i \in Q\}$ in a quorum Q. A replica o_i can be upgraded to an lub of a quorum Q if the content parameter $o_i.c$ and every QoS parameter $o_i.q_k$ could be changed to m_c and m_{q_k}, respectively. In order to reduce the overhead to upgrade a replica, one of the maximal replicas is taken. For example, a maximal replica o_i with the smallest number of parameters to be changed is taken. Then, the maximal replica o_i is upgraded. A quorum Q is referred to as *completable* iff $+Q$ is $\cup Q$. That is, some replica o_i can be upgraded to the top replica $\cup Q$. The quorum Q shown in Figure 1 is incomplete but completable since one of the maximal replicas o_5 and o_6 can be upgraded.

Types of operations

Let *op* be an operation supported by an object o, i.e. read or write operation.Let Q_{op} ($\subseteq O$) be a quorum for an operation *op*. Here, there might not be the newest, i.e. top replica in the quorum Q_{op}. That is, the lub $\cup Q_{op}$ is not in the quorum Q_{op}. Even if there is no top replica in the quorum Q_{op}, there is some maximal replica o_i in the quorum Q_{op}.

There are two types of write operations by which replicas are changed:

1 *Enriching* (E) type.
2 *Impoverishing* (I) type.

Suppose a value x is changed with another value y of a content or QoS parameter in an operation *op*. Here, the value x precedes the value y, i.e. y is newer than x ($x \preceq y$). If *op* is an enriching type of operation, y is richer than x ($x \rightarrow y$). Otherwise, $y \rightarrow x$. A richer replica o_i can be easily changed into a poorer replica because data in the replica is just removed without using additional data not in the replica o_i. On the other hand, we need additional data which is not in a replica o_i to change a poorer replica o_i in order to a richer one. Thus, in an enriching operation, some volume of data is added to a replica o_i, i.e. the replica o_i is enriched. For example, an orange subobject r is added to the replica o_4 by a content operation *insert* as shown in Figure 2. The number of colours (*cl*) is increased in a QoS operation *up-colour* (*uc*), i.e. changed with the fully coloured one as shown in Figure 3 (a). This is an enriching operation. On the other hand, some data is removed from a replica in an impoverishing operation, i.e. the replica is made poorer. For example, some subobject, say an orange r is deleted from a replica o by a content operation *delete* as shown in Figure 3 (b). On the other hand, further data which is not in the replica is

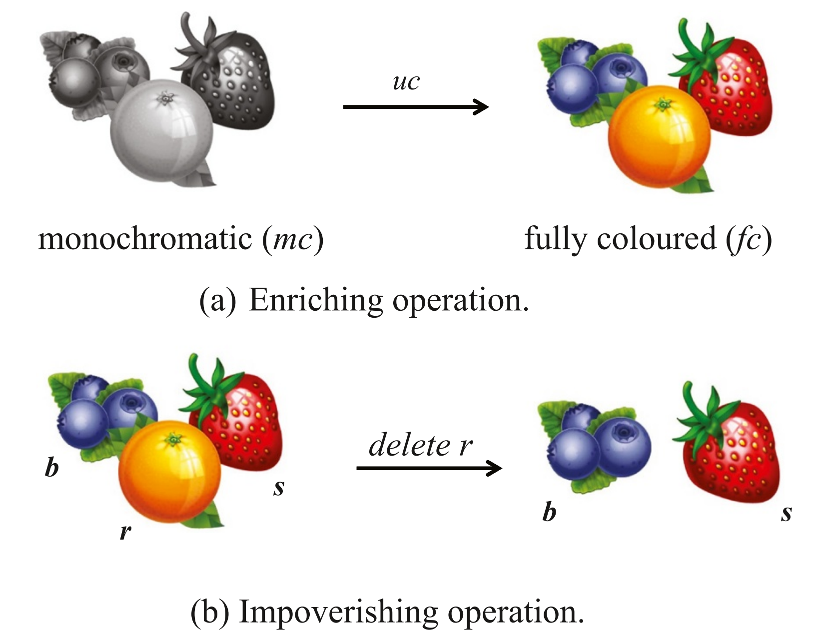

Figure 3 Enriching and impoverishing operations.

required to increase the frame rate. Thus, it is easier to perform the impoverishing type of write operation than the enriching type on a replica.

Suppose there are a pair of monochromatic replicas o_i and o_j of a multimedia object o, which are composed of a blueberry b, orange r, and strawberry s subobjects. A pair of the replicas o_i and o_j are equivalent, $o_i \equiv o_j$ where $o_i.c = o_j.c = \langle b, r, s \rangle$ and $o_i.cl = o_j.cl = mc$. Then, a transaction T_1 deletes an orange subobject r in the replica o_i. The top, i.e. newest replica is the replica o_i while o_j is obsolete. Since *delete* is an impoverishing write operation, the replica o_j can be made a newest one by just deleting the subobject r. On the other hand, suppose a transaction T_2 changes the colour (cl) parameter of the replica o_i to be fully coloured in an *up-colour* (uc) operation. The uc operation is an enriching one. In order to change the replica o_j, data to make the replica o_j fully coloured has to be sent to the replica o_j since the replica o_j does not have the data while o_i has the data. Even if the operation uc which is applied to the replica o_i is obtained to the replica o_j, the replica o_j cannot be changed without obtaining the colour data from the newest replica o_i.

In the QB protocol, $Q_{op_i} \cap Q_{op_j} \neq \phi$ for every pair of quorums Q_{op_i} and Q_{op_j} of conflicting operations op_i and op_j. The quorum-based protocol for abstract types of operations on objects is discussed in the paper [5].

Multimedia Quorum-Based (MQB) Protocol

Counter vectors

Let Q be a quorum of replicas $o_1, \ldots, o_n$ $(n \geq 1)$ of a multimedia object o. Each replica o_i is characterized in terms of the content $o_i.c$ and QoS parameters $o_i.q_1, \ldots, o_i.q_l$ $(i = 1, \ldots, n)$. Here, a tuple of parameters $\langle p_0, p_1, \ldots, p_l \rangle$ is a scheme of a replica o_i. Here, the object o_i is written as a tuple $\langle v_0, v_1, \ldots, v_l \rangle$ $(l \geq 1)$ of values, where v_0 shows the content c and v_k stands for a value of a QoS parameter q_k for $k = 1, \ldots, l$. A vector $o_i.V = \langle vc_o, vc_1, \ldots, vc_l \rangle$ of version counters is assigned to a replica $o_i = \langle o_i.v_0, o_i.v_1, \ldots, o_i.v_l \rangle$.

Initially, $o_i.V = \langle 0,\ 0,\ \ldots,\ 0\rangle$. Each time an element $o_i.q_k$ is changed ($k = 0,\ 1,\ \ldots,\ l$), the counter $o_i.vc_k$ is incremented so that the counter $o_i.vc_k$ is the maximum in the quorum Q.

Suppose a counter vc_k is incremented since a parameter $o_i.p_k$ is changed in a replica o_i by performing an operation op on the replica o_i ($k \in \{0,\ 1,\ \ldots,\ l\}$). First, the maximum counter value v is taken in a set $\{o_i.vc_k \mid o_i \in Q_{op}\}$ of the counter values of the quorum Q_{op}. In the quorum Q_{op}, the maximum value v of the counter vc_k is incremented by one, $v = v + 1$. Then, $o_i.vc_k = v$ on every replica o_i in the quorum Q_{op}. The value of the kth parameter $o_i.q_k$ of a replica o_i is the newest if the counter vc_k is maximum in the quorum Q_{op}.

Let us consider a quorum Q of five replicas $o_1,\ o_2,\ o_3,\ o_4$ and o_5 shown in Figure 1, $Q = \{o_1,\ o_2,\ o_3,\ o_4,\ o_5\}$. Initially, every replica is equivalent in the quorum Q, i.e. $o_1 \equiv o_2 \equiv o_3 \equiv o_4 \equiv o_5$. Each replica o_i has one QoS parameter colour (cl) and a vector $o_i.V = \langle o_i.vc_0,\ o_i.vc_1\rangle$ ($i = 1,\ \ldots,\ 5$). In each vector $o_i.V$, $o_i.vc_0$ is a counter for the content parameter $o_i.c$ ($= o_i.v_0$) and $o_i.vc_1$ is a counter for the QoS parameter cl (number of colours) $o_i.cl$ ($= o_i.v_1$). In every replica o_i, initially $o_i.V = \langle 0,\ 0\rangle$ where o_i is composed of three fully coloured subobjects s, g, and a, i.e. $o_i = \langle\langle s,\ g,\ a\rangle,\ fl\rangle$.

1 First, the fully coloured replicas o_2 and o_5 are updated by changing the colour parameter cl with the monochromatic one, $o_2 = o_5 = \langle\langle s,\ g,\ a\rangle,\ mc\rangle$. The counter vc_1 is incremented by one. Here, $o_2.V = o_5.V = \langle 0,\ 1\rangle$ since the second parameter cl is changed.
2 Next, a pair of replicas o_3 and o_4 are obtained by deleting an orange subobject r, i.e. $o_3.V = o_4.V = \langle 1,\ 0\rangle$ where $o_3 = o_4 = \langle\langle s,\ g\rangle,\ fc\rangle$ since the first content v_0 is changed.
3 Then, the orange subobject r is deleted in the replica o_4, $o_4 = \langle\langle s,\ g\rangle,\ mc\rangle$. Here, $o_4.V = \langle 1,\ 1\rangle$. Here, $o_1 \preceq o_2 \preceq o_4$ where $o_1.V \leq o_2.V$ and $o_2.V \leq o_4.V$.
4 Then, the fully coloured lemon subobject l is added to the replica o_5, $o_5 = \langle\langle b,\ l,\ s\rangle,\ fc\rangle$. The counter vector $o_5.V$ is changed with $\langle 2,\ 0\rangle$.

Here, a pair of the replicas o_2 and o_3 are uncomparable ($o_2 \mid o_3$) where a pair of the vectors $o_2.V = \langle 1,\ 0\rangle$ and $o_3.V = \langle 0,\ 1\rangle$ are not comparable. $o_1 \preceq o_3$ since $o_1.V < o_3.V$. $o_3 \preceq o_4$ and $o_3 \preceq o_5$ where $o_3.V \leq o_4.V$ and $o_3.V \leq o_5.V$, $o_4 \mid o_5$ since $o_4.V = \langle 1,\ 1\rangle$ and $o_5.V = \langle 2,\ 0\rangle$. Figure 4 shows the vector $o_i.V$ of each replica

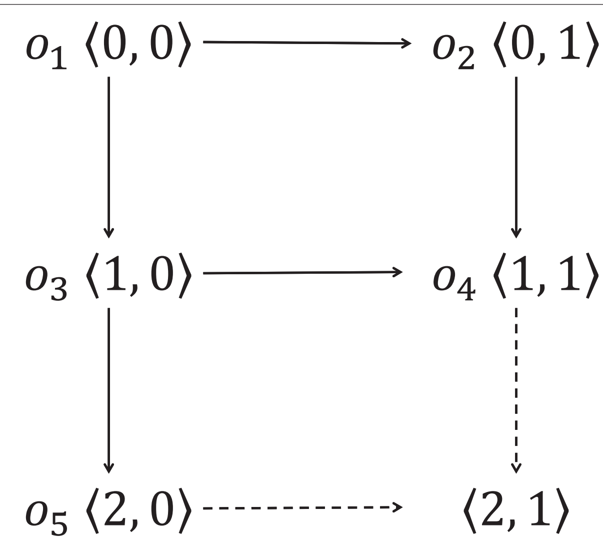

Figure 4 Counter vectors of replicas.

o_i (i = 1, ..., 5). Here, a directed edge $o_i \rightarrow o_j$ shows the newness-precedent relation $o_i \preceq o_j$. Here, there is no top replica in the quorum Q. Here, $\max(o_4.vc_0,\ o_5.vc_0) = 2$ and $\max(o_4.vc_1,\ o_5.vc_1) = 1$. Hence, if the colour parameter $o_5.cl$ of the replica o_5 is changed with monochromatic one mc, the replica o_5 gets the top replica $\langle\langle b,\ l,\ s\rangle, mc\rangle$. Here, the counter $o_5.vc_0$ is incremented by one, $o_5.V = \langle 2,\ 1\rangle$. In another way, the replica o_4 can be the newest replica $\langle\langle b,\ l,\ s\rangle,\ mc\rangle$ by adding a monochromatic lemon l to the replica o_4.

Write operations

Suppose a transaction T issues a write operation op to change the kth parameter p_k of a replica o_i ($0 \leq k \leq l$). A transaction T first locks a replica o_i in the op mode before performing an operation op on the replica o_i. Here, a lock mode op_1 is referred to as *conflict* with a lock mode op_2 if an operation op_1 conflicts with an operation op_2. If the replica o_i is already locked in a mode conflicting with the operation op, the transaction T has to wait.

[Write procedure]

1 First, the transaction T locks every replica o_i with an op mode in the quorum Q_{op}. If every replica o_i could not be locked, the transaction T waits.
2 If successfully locked, the transaction T writes the kth element $o_i.v_k$ of every replica o_i and collects the vector $o_i.V$ from every replica o_i in the quorum Q_{op}.
3 $vc = max\ (\ o_1.vc_k,\ ...,\ o_n.vc_k)$ and the transaction T changes $o_i.vc_k$ with $vc + 1$ in every replica o_i of the quorum Q_{op}.

The version counter vc_k of every replica in the quorum Q_{op} is changed with the maximum value vc. In order to reduce the computation and communication overheads, the parameter $o_i.p_k$ of every replica o_i is not always changed while the counter is updated:

1 If the operation op is an enriching type, the parameter $o_i.q_k$ of every replica o_i is updated in the quorum Q_{op}.
2 If op is an impoverishing type, the parameter $o_i.q_k$ of only the top replica o_i is updated.

We consider the replicas of the object o shown in Figure 5. Suppose a transaction T_1 adds a lemon subobject l to the replicas. The operation add is an enriching type of write operation. Suppose Q_{add} is a quorum $\{o_1,\ o_2,\ o_3\}$ of the replicas for the add operation. Here, $o_1 = \langle\langle b,\ r,\ s\rangle,\ fc\rangle$, $o_2 = \langle\langle b,\ s\rangle, fc\rangle$, and $o_3 = \langle\langle b,\ r,\ s\rangle,\ mc\rangle$. A pair of the replicas o_2 and o_3 are maximal, $Max_{Q_{add}} = \{o_2,\ o_3\} \subseteq Q_{add}$. Here, $\max(o_1.c,\ o_2.c,\ o_3.c) = \langle b,\ s\rangle$

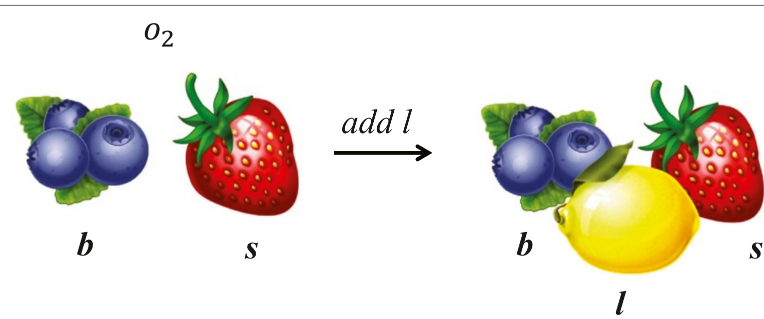

Figure 5 Upgrade of a replica.

(= $o_2.c$ = $o_3.c$) and max($o_1.cl,\ o_2.cl,\ o_3.cl$) = mc. Here, the replica o_2 can be the top replica $\langle\langle b,\ s\rangle,\ mc\rangle\rangle$ by upgrading the replica o_2, o_2 = + Q_{add}. Then, the lemon subobject l is added to the replica o_2. Here, $o_2 = \langle\langle b,\ l,\ s\rangle,\ mc\rangle$. A pair of the other replicas o_1 and o_3 are changed so that $o_1 \equiv o_2$ and $o_3 \equiv o_2$. That is, the colour parameter cl of the replica o_1 is changed with monochromatic one mc, the orange subobject r is deleted, and the lemon l is added to the replica o_1. The orange r is deleted and the lemon l is added to the replica o_3. Then, a pair of the replicas o_2 and o_3 get equivalent with the replica o_1 ($o_2 \equiv o_3 \equiv o_1$ = $\langle\langle b,\ s\rangle,\ mc\rangle$).

Next, suppose a transaction T_2 deletes a subobject b. Here, suppose there are three replicas $o_1,\ o_2$, and o_3 shown in Figure 2. One maximal replica o_2 is taken and upgraded to $\langle\langle b,\ s\rangle,\ mc\rangle$. Then, b is removed. Here, $o_2 = \langle\langle s\rangle,\ mc\rangle$. Since the *delete* operation is an impoverishing one, the other replicas o_1 and o_2 are not updated and the delete operation is logged in o_1 and o_2.

Read operations

Next, suppose a transaction T issues a operation op to replicas of an object o to read the parameter p_k (k = 0, 1, ..., l). The transaction T has to read the kth parameter $o_i.p_k$ of the newest replicas o_i in the quorum Q_{op}. In order to read the newest replica o_i, the transaction T has to read the version counter $o_j.vc_k$ from every replica o_i in the quorum Q_{op}:

[Read procedure]

1 First, the transaction T locks every replica with an op mode in the quorum Q_{op}. If every replica could not be locked, the transaction T waits.

2 If every replica could be successfully locked, the transaction T collects the vector $o_i.V$ from every replica o_i in the quorum Q_{op}.

3 If there is a replica o_i such that $o_j.V \le o_i.V$ for every replica o_j in the quorum Q_{op}, the replica o_i is the top of the replicas in the quorum Q_{op}, i.e. $o_i = \cup Q_{op}$ and is the newest in the quorum Q_{op}. The transaction T reads the replica o_i and change the counter $o_i.vc_k$ in every replica in the quorum Q_{op}.

4 If there is no top replica in the quorum Q_{op}, the transaction T has to upgrade one maximal replica o_i in Q_{op}, i.e. $o_i = +Q_{op}$ which is the top $\cup Q_{op}$ of the quorum Q_{op}. The transaction T reads the kth element $o_i.v_k$ of the replica o_i and changes the counter $o_j.vc_k$ with $o_i.vc_h$ in every replica in Q_{op}.

Upgrade of a maximal replica

If a top replica is not in the quorum Q_{op}, the transaction T has to obtain a top replica from the replicas in the quorum Q_{op} in the read procedure. We discuss how to upgrade a maximal replica o_i to the top replica, i.e. $o_i = +Q_{op}$ by using the vectors of the replicas. A replica o_i is referred to as *satisfy* a counter vector $V = \langle vc_0,\ vc_1,\ ...,\ vc_l\rangle$ iff $o_i.vc_k = vc_k$ for k = 0, 1, ..., l.

First, one maximal replica o_i is selected in the quorum Q_{op} as follows:

[Selection of a maximal replica]

1 The transaction T obtains a vector $V = \langle vc_0, vc_1, ..., vc_l\rangle$ where vc_k = $max(\{o_i.vc_k \mid o_i \in Q_{op}\})$ for $k = 0, 1, ..., l$ from the collection of the vectors collected in the quorum Q_{op}.

2 A replica o_i which satisfies the vector $V = \langle vc_0, vc_1, ..., vc_l \rangle$, i.e. $o_i.vc_k = vc_k$ for every $k = 0, 1, ..., l$, shows the top replica $\cup Q_{op}$. If the replica o_i is found, the transaction T reads the replica o_i.

3 Otherwise, the transaction T selects a replica o_i where $|\{vc_k \mid o_i.vc_k = vc_k$ for $k = 0, 1, ..., l\}|$ is the maximal in the quorum Q_{op}. That is, a maximal replica o_i is selected so that the overhead to change the replica can be reduced.

The replica o_i found at step 3 is not the top in the quorum Q_{op}. Here, a parameter p_k of a replica o_i is current if $o_i.vc_k$ is maximum. Otherwise, the parameter p_k is obsolete. Hence, the transaction T updates the parameters of the replica o_i as follows:

[Upgrade of a maximal replica]

1 For each obsolete parameter p_k, the transaction T finds a replica o_j where $o_j.p_k$ is current in the quorum Q_{op} which satisfies $o_i.vc_k < vc_k$.

2 The transaction T updates each obsolete parameter $o_i.p_k$ with the current one $o_j.p_k$ by using the replica o_j.

3 The vector $o_i.V$ is updated as $o_i.V = V$.

At step 1, the replica o_j found by the transaction T has the newest value of each obsolete parameter p_k of the replica o_i. The value of the parameter $o_i.p_k$ has to be enriched to the parameter value $o_j.p_k$. If the parameter value $o_i.p_k$ is richer than $o_j.p_k$, i.e. $o_j.p_k \rightarrow o_i.p_k$, the QoS parameter of the replica o_i can be impoverished just by deleting some data in the replica o_i without using additional data. Otherwise, the parameter p_k of the replica o_i has to be enriched, i.e. we need further data which is not in the replica o_i to enrich the value of the QoS parameter p_k. Hence, the content $o_i.v_0$ of the replica o_i is required to be the same as the replica o_j, i.e. $o_i.v_0 = o_j.v_0$.

The elements $o_i.v_0, o_i.v_1, ..., o_i.v_l$ of every replica o_i in the quorum Q_{op} ($\subseteq O$) are changed with the newest ones. In addition, the vector $o_i.V$ of every replica o_i in the quorum Q_{op} has to be changed to be larger than every replica in the replica set O. In every *read* operation *op*', every replica in the quorum $Q_{op'}$ is changed with a replica equivalent with the top replica. It is sure at least one top replica of the replica set O is included in the quorum $Q_{op'}$. However, the overhead to change every replica in the quorum $Q_{op'}$ is increased. Suppose one top replica is read by the operation op' and other replicas in the quorum $Q_{op'}$ are not changed. Hence, the quorum Q_{op} may not include the top replica. Here, the content and QoS parameters of every replica in the quorum Q_{op} can be changed since they are just overwritten. However, the maximum vector value obtained by all the replicas in the quorum Q_{op} may not be the maximum in the replica set O. Suppose $Q_{op'} = \{o_1, o_2, o_3\}$ and $Q_{op} = \{o_3, o_4\}$. Here, suppose the replica o_1 is the top replica. A transaction T_1 reads the top replica o_1 in a read operation *op*' but does not change the other replicas o_2 and o_3. Then, another transaction T_2 writes the replicas o_3 and o_4 in a *write*-type operation *op*. Here, the vectors of replicas o_3 and o_4 are not the newest while the vector of the replica o_1 is the newest.

If every replica is updated in a read operation, it implies larger communication and computation overhead to bring update data to every replica and then update every replica in the read quorum. In order to reduce the overhead, we take the following approach:

[Completable quorum]

- In a read operation *op*, only the vector $o_i.V$ of every replica o_i except the top replica is changed but the content $o_i.v_0$ and QoS parameters $o_i.Q$ of the replica o_i are not changed.

In a write operation *op*, the vector V which shows the top replica in the replica set O can be obtained in the quorum Q_{op}. In the example of the quorums $Q_{op'}$ and Q_{op}, the vectors of the replicas o_1, o_2, and o_3 are updated while the content and QoS parameters of the replicas o_2 and o_3 are not updated by the transaction T_1. Then, the transaction T_2 overwrites every replica in the quorum Q_{op}. Here, the vector $o_3.V$ of the replica o_3 is the newest since $o_3.V$ is updated by the transaction T_1. Hence, the vector of the replica o_3 is incremented and then the vector $o_4.V$ of the replica o_4 is changed with $o_3.V$.

Evaluation

We would like to evaluate the multimedia quorum-based (MQB) protocol compared with the traditional quorum-based (QB) protocol in terms of communication overhead. In the MQB protocol, if a transaction issues a read operation, every replica in a read quorum Q_r is not updated while the vector of every replica is updated. We show how much the communication overhead to update every replica in the quorum Q_r can be reduced in the MQB protocol.

Suppose there are n replicas, o_1, ..., o_n ($n \geq 1$) of an object o. Suppose there are two types of operations, read (r) and write (w). Q_r and Q_w show a pair of read and write quorums, respectively. n_r shows the number $|Q_r|$ of replicas in the quorum Q_r and $n_w = |Q_w|$. Let f_r and f_w be a pair of probability that a replica is included in the quorums Q_r and Q_w, respectively. We assume each quorum is randomly constructed. That is, $f_r = n_r/n$ and $f_w = n_w/n$. According to the quorum properties, $f_r + f_w > 1$ and $f_w > 0.5$. Let f be $f_r + f_w - 1$. Here, f shows probability that a replica is included in both the quorums Q_r and $Q_w . f > 0$.

In the QB protocol, a transaction T first issues a lock request to every replica in a quorum Q_{op} to perform an operation $op \in \{r, w\}$. If every replica is successfully locked in the quorum Q_{op}, the transaction T issues an operation *op* to replicas in Q_{op}. First, suppose the transaction T issues a write *op* to every replica in the write quorum Q_w and updates the version counter of every replica. Here, totally $4 \cdot n_w$ ($= 4 \cdot n \cdot f_r$) messages are transmitted. In order to write replicas, data is sent to every replica in the write quorum Q_w. Let d be the size of the update data, e.g. the size of a replica. The expected volume of data transmitted is $n \cdot f_w \cdot d$.

On the other hand, the transaction T issues a read operation *op* to one replica and receives a value of the replica in the read quorum Q_r. Then, the transaction T sends the newest value to every other replica and updates the version vector of every replica in the QB protocol. The totally $4 \cdot n_r$ messages are transmitted between the transaction T and the replicas. In the QB protocol, the newest value of the replicas in the quorum Q_r is read into the transaction T and is transmitted to every other replica which is in the quorum Q_r but not in the quorum Q_w. Hence, the expected volume of data transmitted is $n \cdot f_r \cdot (1 - f_w) \cdot d$.

In the MQB protocol, the transaction T reads the top replica and updates the version counter of every replica in the read quorum Q_r. However, the other replicas are not updated in the quorum Q_r. The number $4 \cdot n_r$ ($= 4 \cdot n \cdot f_w$) and $4 \cdot n_w$ ($= 4 \cdot n \cdot f_r$) of

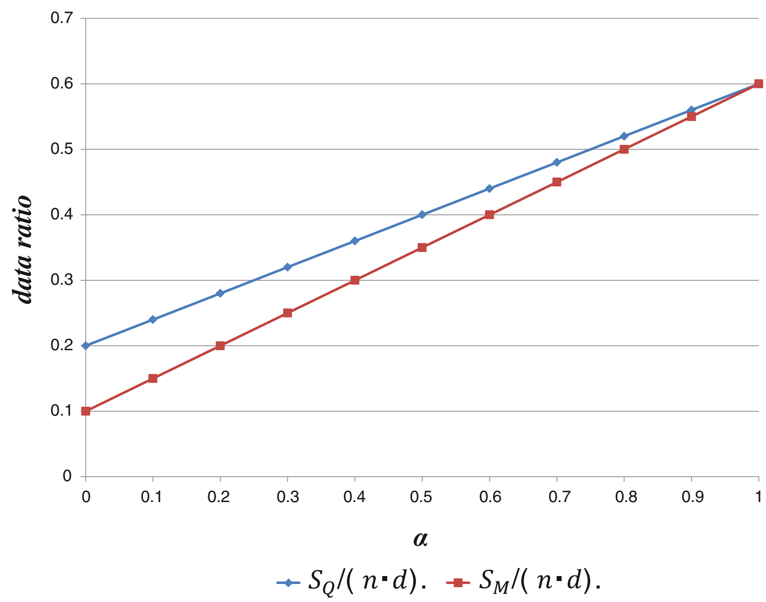

Figure 6 Transmission data volume ($f_w = 0.6$, $f = 0.1$).

messages are transmitted for read and write in the MQB protocol, respectively, as well as the QB protocol. The volume of data transmitted to write the replicas is $n \cdot f_w \cdot d$ in the traditional QB protocol and the MQB protocol. Here, let α be the ratio of the number of write operations to the total number of operations issued by transactions ($0 \leq \alpha \leq 1$). " $\alpha = 0$" means every request is *read* and "$\alpha = 1$" shows every request is write. In the QB protocol, the expected volume S_Q of data transmitted in each transaction is $\alpha \cdot n \cdot f_w \cdot d + (1-\alpha) \cdot n \cdot (1-f_w) \cdot f_r \cdot d$. The expected volume S_M of data transmitted in

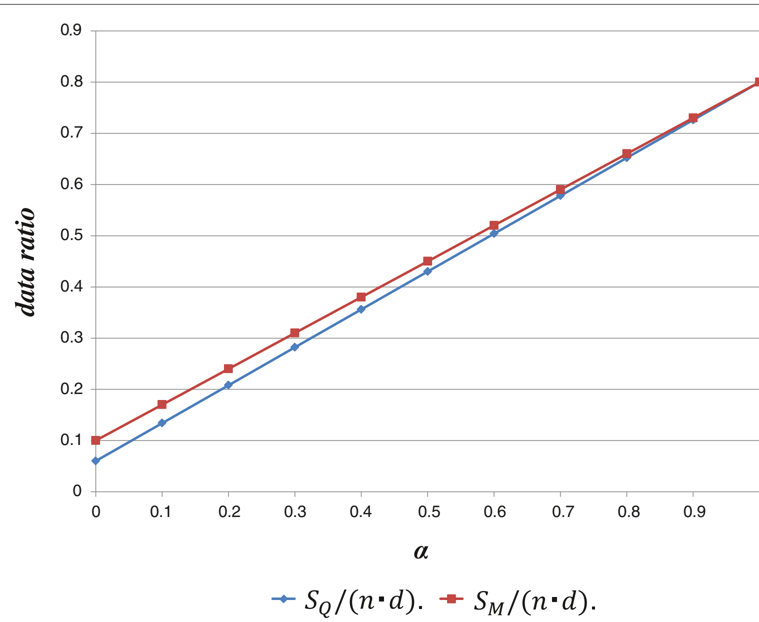

Figure 7 Transmission data volume ($f_w = 0.8$, $f = 0.1$).

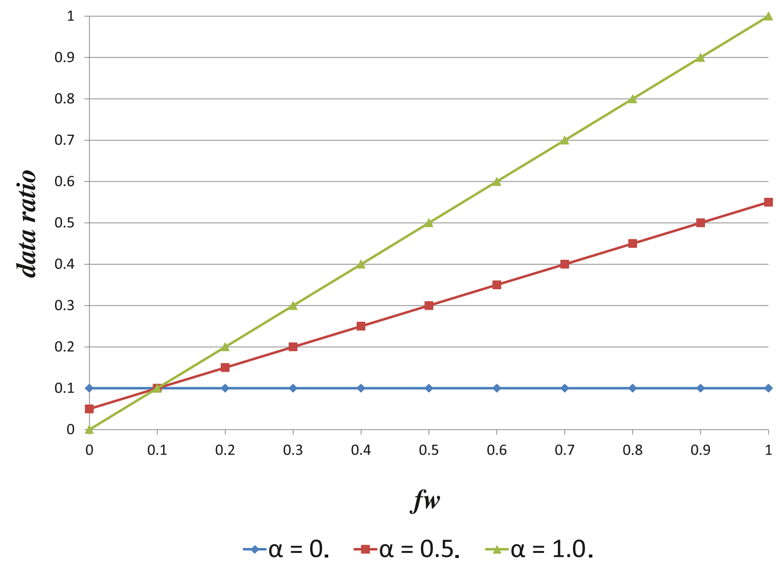

Figure 8 Transmission data volume (MQB).

the MQB protocol is $\alpha \cdot n \cdot f_w \cdot d + (1 - \alpha) \cdot d$ since no data is transmitted to every other replica in *read* than the top replica in the quorum Q_r.

Figures 6 and 7 show the ratios $S_Q/(n \cdot d)$ and $S_M/(n \cdot d)$ for the write ratio α. Here, we assume there are ten replicas, $n = 10$. In Figure 6, $f_w = 0.6$, and $f = 0.1$. In Figure 7, $f_w = 0.8$ and $f = 0.1$. $S_Q = S_M$ for $\alpha = 1$. As shown in Figures 6 and 7, the total amount of data transmitted can be reduced in the MQB protocol compared with the QB protocol. In Figure 8, the ratio S_M / $(d \cdot n)$ is shown for the write probability f_w. Here, f_w should be lager than 0.5 from the quorum constraint ($f_w > 0.5$). The lager f_w and α are, the lager amount of data is transmitted. In order to reduce the communication overhead, the write quorum

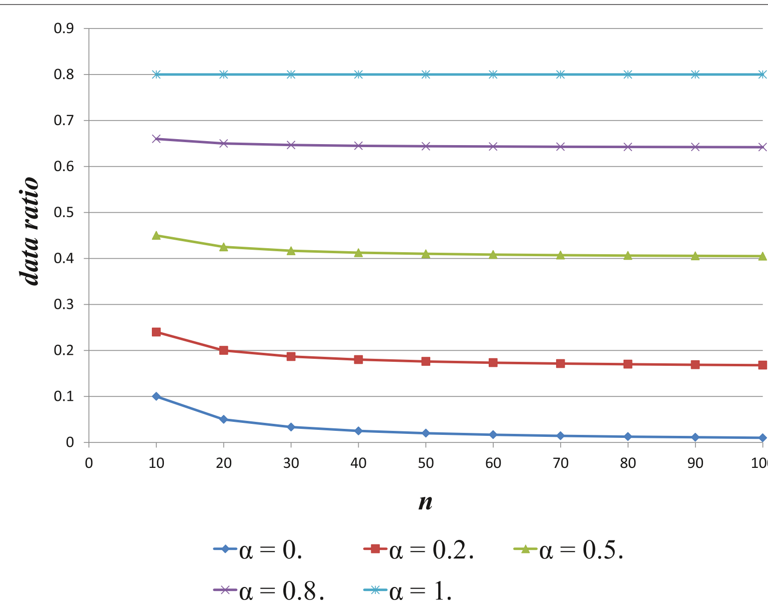

Figure 9 Transmission data volume (MQB) (f_w = 0.8, f = 0.1).

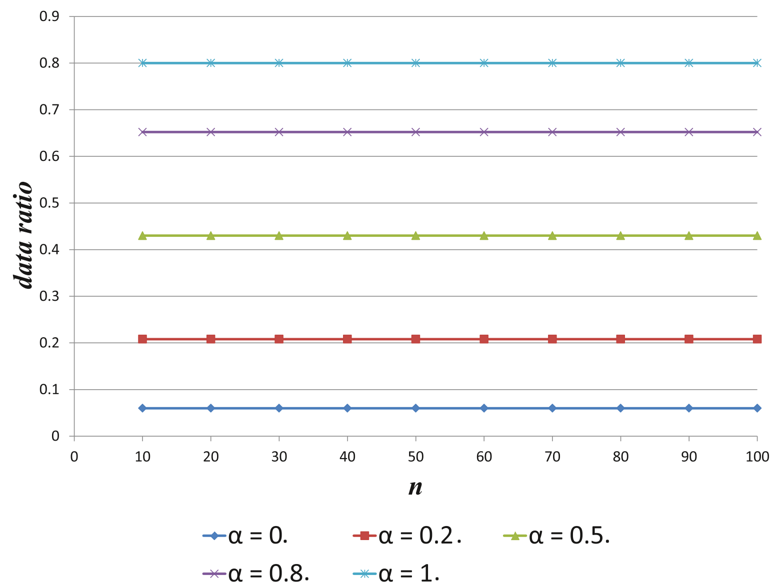

Figure 10 Transmission data volume (QB) (f_w = 0.8, f = 0.1).

should be smallest. In Figure 9, the data ratio S_M / $(d \cdot n)$ for the number n of replicas is shown where f_w = 0.8. Figure 10 shows the data ratio S_Q / $(d \cdot n)$ for the number n of the replicas. The communication overhead of the MQB protocol is increased in complexity $O(n)$ since the ratio to the number n of the replicas is almost $O(1)$.

Conclusions

In this paper, we discussed the multimedia quorum-based (MQB) protocol to keep replicas of a multimedia object mutually consistent.A multimedia object is characterized in content and QoS parameters. Replicas are partially ordered in the newness-precedent relation $\preceq$ in terms of not only content parameter but also QoS parameters. If a replica o_i has a larger vector value than another replica o_j, the replica i.e. $o_i.V > o_j.V$, o_i is newer than o_j. A replica o_i and the vector $o_i.V$ are updated each time the replica o_i is manipulated. In order to increase the performance to read replicas, only the counter vector of each replica is updated in a quorum while the content and QoS parameters of the replica are not updated. We evaluated the MQB protocol in terms of the total volume of data transmitted among the replicas. We showed the total amount of data transmitted can be reduced in the MQB protocol compared with the traditional quorum-based (QB) protocol.

Competing interests
The authors declare that they have no competing interests.

Author's contribution
Tadateru Ohkawara carried out the MQB protocol studies, participated in designing, implementing, and evaluating the MQB protocol and drafted the manuscript. Ailixer Aikebaier and Tomoya Enokido participated in the design of the algorithm used in the MQB protocol. Makoto Takizawa conceived of the study, and participated in its design and coordination and helped to draft the manuscript. All authors read and approved the final manuscript.

Acknowledgements
This research is partially supported by the strategy research of Seikei University and MEXT, Grant-in-Aid for Building Strategy Research Infrastructure.

Author details
[1] Department of Computer and Information Science, Seikei University, 3-3-1 Kichijoji-kitamachi, Musashino-shi, Tokyo 180-8633, Japan. [2] Faculty Of Business Administration, Rissho University, 4-2-16 Oosaki, Shinagawa-ku, Tokyo 141-8602, Japan. [3] New Generation Network Laboratory, NICT, 4-2-1 Nukui-Kitamachi, Koganei, Tokyo 184-8795, Japan.

Received: 17 October 2011 Accepted: 7 March 2012 Published: 10 May 2012

References

1. Hofmann P, Woods D (2010) Cloud Computing: The Limits of Public Clouds for Business Applications. Journal of IEEE Internet Computing 14: 90–93 ISBN 1089–7801
2. Schollmeier R (2001) A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. In Proc. of the First International Conference on Peer-to-Peer Computing. Linkoping, Sweden, p 101
3. Gray J (1978) Notes on Database Operating Systems. Lecture Notes in Computer Science, vol 60. Springer Verlag
4. Helal A, Bhargava B (1995) Performance Evaluation of the Quorum Consensus Replication Method. In Proc. of the Internation Computer Performance and Dependability Symposium (IPDS'95). Erlangen, Germany, pp 165-172
5. Stoica I, Morris R, Karger D, Frans Kaashoek M, Balakrishnan H (2001) Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In Proc. of ACM the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '01). San Diego, califonia, USA, pp 149–160
6. Lakshman A, Malik P (2010) Cassandra: A Decentralized Structured Storage System. ACM SIGOPS Operating Systems Review 44(26): 35–40
7. Ohkawara T, Aikebaier A, Enokido T, Takizawa M (2011) Quorums-based Replication of Multimedia Objects in Distributed Systems. In Proc. of the International Conference on Network-Based Information Systems, NBiS2011, CD-ROM. Tirana, Albania
8. Enokido T, Higaki H, Takizawa M (1998) Group Protocol for Distributed Replicated Objects. In Proc. of the 27th International Conference on Parallel Processing (ICPP-98). Minneapolis, Minnesota, USA, pp 570–577
9. Tanaka K, Takizawa M (2001) Quorum-Based Locking Protocol for Replicas in Object-Based Systems. In Proc. of IEEE the 5th International Symposium or Autonomous Decentralized Systems. Dallas, Texas, USA, pp 196–203

doi:10.1186/2192-1962-2-11
Cite this article as: Ohkawara *et al.*: **Quorums-based Replication of Multimedia Objects in Distributed Systems.** *Human-centric Computing and Information Sciences* 2012 **2**:11.