A General method to define quorums

Mitchell L. Neilsen, Masaaki Mizuno, Michel Raynal

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 1529

Programme 1
Architectures parallèles, Bases de données,
Réseaux et Systèmes distribués

A GENERAL METHOD TO DEFINE QUORUMS

Mitchell L. NEILSEN
Masaaki MIZUNO
Michel RAYNAL

Septembre 1991

A General Method to Define Quorums

Mitchell L. Neilsen Masaaki Mizuno *

Department of Computing and Information Sciences
Kansas State University
Manhattan, Kansas 66506

Michel Raynal

IRISA
Campus de Beaulieu
35042 Rennes-Cédex, FRANCE

Abstract

Quorum based protocols are widely used in distributed systems (for example, to ensure mutual consistency in distributed databases or to ensure fault tolerance in distributed operating systems). Traditional methods use a voting mechanism to implement quorums. To obtain better performance, several authors have proposed other methods to define quorums.

In this paper, we present a general method to define quorums and an efficient method to determine if a given set contains a quorum. Then, we show that this method, called **composition**, is more general than several other methods. Finally, we show that composition provides a natural method to define quorums to be used in an arbitrary network, or even in a collection of interconnected networks.

Key words: Concurrency control, coteries, distributed computing, fault tolerance, mutual exclusion, quorum consensus.

*This work was supported in part by the National Science Foundation under Grant CCR-8822378.

UNE METHODE GENERALE POUR DEFINIR DES QUORUMS DANS LES SYSTEMES REPARTIS

Résumé

Un certain nombre de protocoles utilises dans les systemes repartis sont fondes sur l utilisation de quorums (c' est par exemple le cas pour assurer la coherence mutuelle des donnees dans les bases de donnees reparties ou pour resister aux defaillances dans les systemes d exploitation repartis). Les methodes traditionnelles utilisent des mecanismes de votes pour implementer les quorums. Plusieurs auteurs ont propose d autres methodes (fondees sur des structurations logiques des sites en arbre ou en grille) afin d en ameliorer les performances.

Dans cet article une methode tres generale est presentee pour definir des quorums; elle est accompagnee d une procedure efficace permettant de decider si un ensemble donne contient un quorum. Il est montre que cette methode, appelee composition, bien qu'extrememement simple, est plus generale que les autres. Il est egalement montre que cette methode de composition permet de definir des quorums dans des reseaux arbitraires ainsi que dans des reseaux interconnectes.

1 Introduction

In distributed systems, quorum based protocols are an important class of protocols. They gracefully tolerate node and communication line failures, and may be used in a large number of applications, such as mutual exclusion, replica control, leader election, commit-abort, and name serving [3]. Garcia-Molina and Barbara formalized quorum based protocols in terms of the data structures that are used by the protocols. These data structures are a generalization of an idea which was first proposed by Lamport [10].

There are many methods available to construct these data structures. One very well known method is to use *weighted voting* (*quorum consensus*) [7, 15]. First, each node is assigned a number of votes. A quorum is defined to be a set of nodes, such that the total number of votes assigned to the nodes in the set is greater than a predefined threshold. For instance, if majority consensus is used, each node is assigned a single vote and a quorum is formed by obtaining a majority of votes.

In order to improve on the performance exhibited by quorum consensus, Kumar proposed *hierarchical quorum consensus* [9]. The nodes are organized into a multi-level hierarchy, and quorum consensus is used on each level.

Agrawal and El Abbadi proposed two methods for generating quorums: a *grid protocol* and a *tree protocol* [1]. They also proposed protocols that use a two-level hierarchy called *hybrid replica control protocols*. At the first level, quorum consensus is used; and at the second level, either the grid protocol or the tree protocol is used. If the grid protocol is used, the resulting protocol is called the *grid-set protocol*. On the other hand, if the tree protocol is used, the resulting protocol is called the *forest protocol*. They defined a *logical unit* as a single node, a grid, or a binary tree. They noted that any logical unit may be used at the second level resulting in an *integrated protocol*.

In this paper, a more general method, called **composition** (or the **join algorithm**) is presented. We define two types of structures: simple and composite. Composite structures are structures constructed by composing existing structures. Simple structures are all other nonempty structures. We also present an efficient method to determine if a given set contains a quorum, called the **quorum containment test**. If the quorum containment test is used, it is not necessary to actually compute and store all of the quorums of the composite structure in advance. Theoretical aspects of composition have been presented elsewhere [13].

Composition may be used to generate a wide range of different structures. In particular, we show that the tree protocol, hierarchical quorum consensus, and the hybrid replica control protocols may all be generalized by using composition. In general, any structures, simple or composite, may be used to generate composite structures. Other protocols impose restrictions on the types of simple structures that may be used to generate composite structures. Finally, we show that compo-

sition provides a natural method to generate structures in an arbitrary network or in a collection of interconnected networks.

The organization of this paper is as follows: Section 2 presents formal definitions of the structures and a description of composition. Section 3 describes how compostion may be used to generalize previous work and how composition may be used in an arbitrary network. Finally, Section 4 summarizes the results presented.

2 Structures and Composition

2.1 A Short Survey on Structures

Barbara and Garcia-Molina have defined structures which may be used in a wide variety of distributed protocols [3, 6]. In this section, these structures are defined.

Let U denote a nonempty set nodes. The term **nodes** may refer to computers in a network or copies of a data object in a replicated database.

A collection of sets, Q, is a **quorum set** under U iff

1. $G \in Q \Rightarrow (G \neq \emptyset$ and $G \subseteq U)$.
2. **(Minimality)**: $G, H \in Q \Rightarrow G \not\subset H$.

The sets $G \in Q$ are called **quorums**. Note that not all nodes must appear in a quorum set; that is, $\{\{a\}\}$ is a quorum set under $\{a, b, c\}$.

A quorum set, Q, is a **coterie** under U iff the **intersection property** is satisfied; that is, $G, H \in Q \Rightarrow G \cap H \neq \emptyset$.

Let Q_1 and Q_2 be coteries under U. Then, Q_1 **dominates** Q_2 iff

1. $Q_1 \neq Q_2$.
2. For each $H \in Q_2$, there is a $G \in Q_1$ such that $G \subseteq H$.

A coterie, Q, under U is **dominated** iff there is another coterie under U which dominates Q. If there is no such coterie, then Q is **nondominated**. Note that the empty coterie $Q = \emptyset$ under U is nondominated iff $U = \emptyset$. Nondominated coteries are able to resist more faults than the coteries which they dominate. This point is clearly illustrated by an example below in Section 2.2.

Let Q be a quorum set under U. Then, a **complimentary quorum set**, Q^c, is another quorum set under U such that $G \in Q$ and $H \in Q^c \Rightarrow G \cap H \neq \emptyset$.

As defined in [5, 8], the pair, $\mathbf{B} = (Q, Q^c)$, is called a **bicoterie** under U. If Q or Q^c is a coterie, then the pair $\mathbf{B}$ is called a **semicoterie**.

Let $\mathbf{B}_1 = (Q_1, Q_1^c)$ and $\mathbf{B}_2 = (Q_2, Q_2^c)$ be bicoteries under U. Then, $\mathbf{B}_1$ **dominates** $\mathbf{B}_2$ iff

1. $\mathbf{B}_1 \neq \mathbf{B}_2$; that is, $Q_1 \neq Q_2$ or $Q_1^c \neq Q_2^c$.
2. For each $H \in Q_2$, there is a $G \in Q_1$ such that $G \subseteq H$.
3. For each $H \in Q_2^c$, there is a $G \in Q_1^c$ such that $G \subseteq H$.

A bicoterie $\mathbf{B}$ under U is **dominated** iff there is another bicoterie under U which dominates $\mathbf{B}$. If there is no such bicoterie, then $\mathbf{B}$ is **nondominated**.

There are many complimentary quorum sets corresponding to a given quorum set. The **antiquorum set** of Q, denoted Q^{-1}, is the complementary quorum set with the largest number of quorums of minimal size. Thus, we say that an antiquorum set is **maximal**. More formally, let

$$I_Q = \{H \subseteq U \mid G \cap H \neq \emptyset \text{ for all } G \in Q\}, \text{ then}$$

$$Q^{-1} = \{H \in I_Q \mid H' \not\subset H \text{ for all } H' \in I_Q\}.$$

The pair, $\mathbf{QA} = (Q, Q^{-1})$, is called a **quorum agreement**. It is easy to show that quorum agreements are the same as nondominated bicoteries. To avoid confusion, we will use the term "nondominated bicoterie" in the rest of the paper.

For any nondominated bicoterie, (Q, Q^{-1}), there are only three possible cases:

1. Q and Q^{-1} are nondominated coteries, and $Q = Q^{-1}$.
2. Q is a dominated coterie, and Q^{-1} is not a coterie (or equivalently, Q^{-1} is a dominated coterie, and Q is not a coterie).
3. Neither Q nor Q^{-1} is a coterie.

2.2 Applications of Structures

There are many different distributed protocols based on the above structures. In this section, we review a few such protocols.

Let $U = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ denote the set of nodes in a distributed system. Then, $Q_1 = \{\{\mathbf{a}, \mathbf{b}\}, \{\mathbf{b}, \mathbf{c}\}, \{\mathbf{c}, \mathbf{a}\}\}$ is a nondominated coterie under U. This coterie may be used in a mutual exclusion algorithm which survives some node and communication link failures [3]. In order to enter the critical section, a node must

receive permission from all nodes in a quorum of Q_1. Because of the intersection property, the mutual exclusion property is guaranteed.

To see the advantage of using a nondominated coterie, let $Q_2 = \{\{\mathbf{a}, \mathbf{b}\}, \{\mathbf{b}, \mathbf{c}\}\}$ be another coterie under U. Note that coterie Q_2 is dominated by Q_1. If a network partition occurs between node **b** and the other nodes, or if node **b** fails, then a quorum may still be formed using Q_1, but not using Q_2. In general, a nondominated coterie is more fault tolerant than any coterie it dominates.

Semicoteries can be used by replica control protocols (based on version numbers) in distributed database management systems [1]. Writing (reading) an object requires the locking of each member of a write (read) quorum. Let Q (Q^c) denote the set of write (read) quorums. To ensure one-copy equivalence, the pair (Q, Q^c) must be a semicoterie; that is any write quorum must intersect with any read or write quorum.

There are many other applications. The interested reader may refer to numerous other papers [1, 2, 3, 12].

2.3 Composition of Structures

Composition provides a simple way of combining nonempty structures to construct new, larger structures. First, we discuss the composition of quorum sets. Then, we show that composition may be used to generate any of the above structures. Finally, we present an efficient method to determine whether a given set of nodes contains a quorum, without actually computing and storing all of the quorums in advance.

2.3.1 Composition of quorum sets

Let U_1 be a nonempty set of nodes and $x \in U_1$. Let U_2 be a nonempty set of nodes such that $U_1 \cap U_2 = \emptyset$. Let $U_3 = (U_1 - \{x\}) \cup U_2$. Given a quorum set Q_1 under U_1 and a quorum set Q_2 under U_2, a new quorum set Q_3 under U_3 may be constructed by replacing each occurrence of x in quorums of Q_1 by nodes in a quorum of Q_2. More formally, let $\mathbf{Q}_{U_i}$ denote the set of all nonempty quorum sets under U_i for $i = 1, 2, 3$, and define a function, $T_x : \mathbf{Q}_{U_1} \times \mathbf{Q}_{U_2} \rightarrow \mathbf{Q}_{U_3}$, by

$$T_x(Q_1, Q_2) = \{G_3 \mid G_1 \in Q_1, G_2 \in Q_2, G_3 = \begin{cases} (G_1 - \{x\}) \cup G_2 & \text{if } x \in G_1 \\ G_1 & \text{otherwise} \end{cases} \}$$

The function, T_x, is called a **composition function**. A quorum set constructed by using a composition function is called a **composite quorum set**; that is, $Q_3 = T_x(Q_1, Q_2)$ is a composite quorum set. All other nonempty quorum sets are called **simple quorum sets**. For instance, simple quorum sets may be constructed by using quorum consensus, the grid protocol, the tree protocol, or some other method. The **input quorum sets**, Q_1 and Q_2, may be either simple or composite.

For example, let $U_1 = \{1,2,3\}$, $x = 3$, and $U_2 = \{4,5,6\}$. Define the input quorum sets, Q_1 (under U_1) and Q_2 (under U_2), as follows:

$$Q_1 = \{\{1,2\},\{2,3\},\{3,1\}\}$$

$$Q_2 = \{\{4,5\},\{5,6\},\{6,4\}\}$$

Then, $T_3(Q_1, Q_2) = Q_3$, where Q_3 is a quorum set under $U_3 = \{1,2,4,5,6\}$, and Q_3 is constructed by replacing each occurrence of $x = 3$ in quorums of Q_1 by nodes in quorums of Q_2.

$$Q_3 = \{\{1,2\},\{2,4,5\},\{2,5,6\},\{2,6,4\},\{4,5,1\},\{5,6,1\},\{6,4,1\}\}$$

Note that the above quorum sets, Q_1, Q_2, and Q_3, are all nondominated coteries. This is no accident. In the following subsection, we state some properties that composition satisfies and show that composition may be used to generate a wide variety of structures. Complete proofs may be found elsewhere [13].

2.3.2 Properties

Suppose that U_1 is a nonempty set of nodes such that $x \in U_1$. Suppose that U_2 is a nonempty set of nodes such that $U_1 \cap U_2 = \emptyset$. Let $U_3 = (U_1 - \{x\}) \cup U_2$. Let Q_i be a nonempty coterie under U_i for $i = 1,2$. Let $Q_3 = T_x(Q_1, Q_2)$. Then, the following properties are satisfied:

1. Q_3 is a coterie under U_3.
2. If Q_1 and Q_2 are both nondominated, then Q_3 is nondominated.
3. If Q_1 is dominated, then Q_3 is dominated.
4. If Q_2 is dominated and $x \in G$ for some $G \in Q_1$, then Q_3 is dominated.

The coterie Q_3 is called a **composite coterie**. Composition may also be used to generate bicoteries. Structures generated by using composition are called **composite structures**.

1. Suppose that $\mathbf{B}_1 = (Q_1, Q_1^c)$ is a bicoterie under U_1 and $\mathbf{B}_2 = (Q_2, Q_2^c)$ is a bicoterie under U_2. Let $Q_3 = T_x(Q_1, Q_2)$ and $Q_3^c = T_x(Q_1^c, Q_2^c)$. Then, $\mathbf{B}_3 = (Q_3, Q_3^c)$ is a bicoterie under U_3.
2. Suppose that $\mathbf{QA}_1 = (Q_1, Q_1^{-1})$ is a nondominated bicoterie under U_1 and $\mathbf{QA}_2 = (Q_2, Q_2^{-1})$ is a nondominated bicoterie under U_2. Let $Q_3 = T_x(Q_1, Q_2)$ and $Q_3^{-1} = T_x(Q_1^{-1}, Q_2^{-1})$. Then, $\mathbf{QA}_3 = (Q_3, Q_3^{-1})$ is a nondominated bicoterie under U_3.

2.3.3 Quorum containment test

In practice, to determine if a given set contains a quorum, it is not necessary to actually compute and store all of the quorums in advance. Instead, we only need to store the input quorum sets used to construct the composite quorum set and information about how the composite quorum set was constructed. This yields an efficient method to determine if a given set of nodes, say S, contains a quorum of the composite quorum set Q. The following function, QC, called the **quorum containment test**, returns true if there exists a quorum $G \in Q$ such that $G \subseteq S$, and false otherwise. An example of the quorum containment test is presented in Section 3.2.1.

```
function QC(S : set of nodes, Q : quorum set) : boolean;
  begin
    if composite(Q, x, Q1, Q2, U2)
      then /* Q is a composite quorum set */
        if QC(S, Q2)
          then return QC((S − U2) ∪ {x}, Q1)
          else return QC((S − U2), Q1)
      else /* Q is a simple quorum set */
        if G ⊆ S for some G ∈ Q
          then return true
          else return false
  end
```

We assume that **composite**(Q, x, Q_1, Q_2, U_2) is a function that returns true if input parameter Q is a composite quorum set and false if Q is a simple quorum set. If Q is a composite quorum set, as a side effect, the function also returns x, Q_1, Q_2, and U_2, such that $Q = T_x(Q_1, Q_2)$ and Q_2 is a quorum set under U_2.

Since the construction of a composite quorum set is determined statically, function **composite** may be implemented by simple table indexing; therefore, it may be performed in constant time. Let $\{Q_1, Q_2, \cdots, Q_M\}$ denote the set of simple input quorum sets used to construct the composite quorum set, Q, by applying the composition function $(M - 1)$ times. The time complexity of determining if a given set of nodes, say S, contains a quorum, using the above function, is $O(M\mathbf{c}) + O(M\mathbf{d})$, where $\mathbf{c}$ is the maximum time required to determine if $G \subseteq S$ for any quorum G in any simple input quorum set, and $\mathbf{d}$ is the time required to compute the set difference and union, $(S - U_2) \cup \{x\}$. One possible implementation is to use bit vectors to denote the sets and quorums [14]. Without loss of generality, we may assume that the simple input structures are defined under disjoint sets. Then, it is not necessary to compute the set difference in the quorum containment test, so the complexity becomes $O(M\mathbf{c})$.

3 Applications of Composition

In this section, we present some protocols which may be used to construct simple structures. Then, we review the tree protocol, hierarchical quorum consensus, and the hybrid replica control protocols, and show that composition is more general than each of these protocols. In fact, composition may be applied to any structures, simple or composite. Other protocols impose restrictions on the types of simple structures used in generating composite structures.

3.1 Simple quorum sets

There are several methods available to generate simple structures, including: majority consensus, quorum consensus (weighted voting), and the grid protocols.

3.1.1 Quorum consensus

Quorum sets may be generated by using **weighted voting** [7]. Each node is assigned a specific number of votes. A quorum is formed by obtaining at least a threshold of votes. Formally, **quorum consensus** using weighted voting is defined as follows. Let U denote a nonempty set of nodes. A **vote assignment** is a function $v : U \rightarrow \mathbf{N}$, where $\mathbf{N}$ is the set of all nonnegative integers. The total number of votes is $\mathrm{TOT}(v) = \sum_{a \in U} v(a)$. The majority of votes is given by $\mathrm{MAJ}(v) = \lceil (\mathrm{TOT}(v) + 1)/2 \rceil$.

Given a threshold $q \geq 1$, the corresponding quorum set is given by

$$Q = \{G \mid G \subseteq U, \sum_{a \in G} v(a) \geq q, \, G \text{ is minimal}\}$$

To say G is **minimal** means that there is no other quorum $H \in Q$, such that $H \subset G$. Given a complementary threshold q_c, such that $q + q_c \geq \mathrm{TOT}(v) + 1$, the corresponding complementary quorum set is given by

$$Q^c = \{G \mid G \subseteq U, \sum_{a \in G} v(a) \geq q_c, \, G \text{ is minimal}\}$$

If $q \geq \mathrm{MAJ}(v)$, then Q is a coterie. Note that either q or q_c must be greater than $\mathrm{MAJ}(v)$, so either Q or Q^c must be a coterie. For example, if $q = \mathrm{TOT}(v)$ and $q_c = 1$, the resulting semicoterie (Q, Q^c) corresponds to the *write-all* approach, where Q denotes the set of write quorums and Q^c denotes the set of read quorums in a replica control protocol. However, if $q = q_c = \mathrm{MAJ}(v)$, the resulting quorum sets correspond to majority consensus [15].

3.1.2 Grid protocol

As an alternative to constructing finite projective planes, Maekawa suggested constructing coteries by using a square grid [11]. First; each node is assigned a location on a square $k \times k$ grid. Quorums are formed by choosing all elements in any one row and any one column.

Grids may also be used to construct bicoteries. Several such methods have been proposed [1, 4, 5]. However, some of these methods result in bicoteries which are dominated. In this section, we present several new methods that result in bicoteries which dominate the bicoteries produced by these methods. Recall that nondominated structures exhibit better performance than the structures which they dominate [3].

1. **Fu's rectangular bicoteries [5]**: Quorums are formed by choosing all elements in any one column. Complementary quorums are formed by choosing one element from each column. The resulting bicoteries are nondominated.

2. **Cheung's grid protocol [4]**: Quorums are formed by choosing all elements in any one column and one element from each of the remaining columns. Complementary quorums are formed by choosing one element from each column. The resulting bicoteries are dominated.

3. **Grid protocol A**: Quorums are formed by using Cheung's grid protocol. Complementary quorums are formed by choosing one element from each column. Also, complementary quorums are formed by choosing all elements in any one column. The resulting bicoteries are nondominated, and dominate the bicoteries that result from Cheung's grid protocol.

4. **Agrawal's grid protocol [1]**: Quorums are formed by choosing all elements in any row, along with all elements in any column. Complementary quorums are formed by choosing all elements is any one row or by choosing all elements in any one column. The resulting bicoteries are dominated.

5. **Grid protocol B**: Quorums are formed by using Agrawal's grid protocol. Complementary quorums are formed by choosing one element from each row or by choosing one element from each column. The resulting bicoteries are nondominated, and dominate the bicoteries that result from Agrawal's grid protocol.

For example, consider the following simple grid shown in Figure 1.

1	2	3
4	5	6
7	8	9

Figure 1. Grid

In the first case, the resulting pair, (Q_1, Q_1^c), is a nondominated bicoterie.

$$Q_1 = \{\ \{1,4,7\},\{2,5,8\},\{3,6,9\}\} \text{ and}$$

$$Q_1^c = \{\ \{1,2,3\},\{1,2,6\},\{1,2,9\},\{1,3,5\},\{1,3,8\},\{1,5,6\},\cdots,\{7,8,9\}\}.$$

In the second case, the resulting pair, (Q_2, Q_2^c), is a dominated bicoterie.

$$Q_2 = \{\ \{1,2,3,4,7\},\{1,2,4,6,7\},\{1,2,4,7,9\},\{1,3,4,5,7\},$$
$$\{1,3,4,7,8\},\{1,4,5,6,7\},\cdots,\{3,6,7,8,9\}\} \text{ and}$$

$$Q_2^c = Q_1^c.$$

In the third case, the quorum set is the same as Q_2 above, and the resulting pair is a nondominated bicoterie.

$$Q_3 = Q_2 \text{ and}$$

$$Q_3^c = Q_1 \cup Q_1^c.$$

In the fourth case, the resulting pair is a dominated bicoterie.

$$Q_4 = \{\ \{1,2,3,4,7\},\{1,4,5,6,7\},\{1,4,7,8,9\},\cdots,\{3,6,7,8,9\}\} \text{ and}$$

$$Q_4^c = \{\ \{1,2,3\},\{4,5,6\},\{7,8,9\},\{1,4,7\},\{2,5,8\},\{3,6,9\}\}.$$

Finally, in the fifth case, the quorum set is the same as above, and the resulting pair is a nondominated bicoterie.

$$Q_5 = Q_4 \text{ and}$$

$$Q_5^c = Q_4^c \cup \{\{1,2,6\},\{1,2,9\},\{1,3,5\},\{1,3,8\},\{1,4,8\},\{1,4,9\},\cdots\{6,7,8\}\}.$$

3.2 Composite structures

In this section, we review several methods used to construct composite structures: the tree protocol, hierarchical quorum consensus, and hybrid replica control protocols. Then, we show that all of these protocols may be obtained by using composition. Finally, we show that composition provides a natural method to generate structures in an arbitrary network.

3.2.1 Tree protocol

The tree protocol is a method for generating coteries [2]. The set of N nodes are logically arranged in a complete binary tree. A **path** in the tree is a sequence of nodes $a_1, a_2, \cdots, a_i, a_{i+1}, \cdots, a_j$ such that a_{i+1} is a child of a_i. A quorum is constructed by grouping all nodes on a path from the root node to a leaf node. If a node on the path is not available, paths that start at both children and terminate at the leaves may be used instead. They suggested that any k-ary tree, with $k \geq 2$, may be used.

In fact, we have shown that the algorithm may be applied to any tree in which each nonleaf node has at least two children and that coteries generated by the algorithm are always nondominated [13]. The resulting coteries are called **tree coteries**.

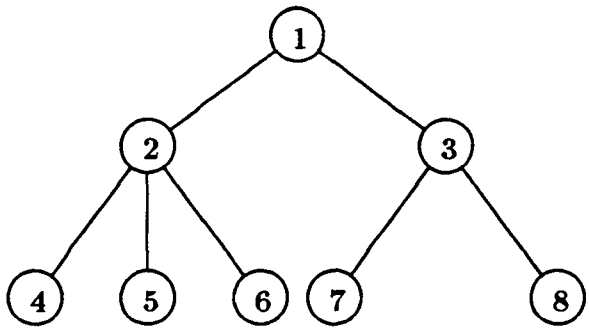

Figure 2. Tree

Consider the tree shown in Figure 2. If all nodes are available, then any of the following sets are quorums: {1,2,4}, {1,2,5}, {1,2,6}, {1,3,7}, and {1,3,8}. If node 1 is unavailable, then paths from both children, nodes 2 and 3, may be used instead. Thus, the sets {2,3,4,7}, {2,3,4,8}, {2,3,5,7}, {2,3,5,8}, {2,3,6,7}, and {2,3,6,8} are quorums. If node 2 is unavailable, the set {1,4,5,6} is a quorum. Likewise if node 3 is unavailable, the set {1,7,8} is a quorum. If both nodes 1 and 2 are unavailable, the sets {3,4,5,6,7} and {3,4,5,6,8} are quorums. Likewise, if both nodes 1 and 3 are unavailable, the sets {2,4,7,8}, {2,5,7,8}, and {2,6,7,8} are quorums. Finally, if nodes 1, 2, and 3 are unavailable, the set {4,5,6,7,8} is a quorum. The collection of all quorums is called a tree coterie.

Tree coteries may be formally described by using composition. Let $U = \{a_1, a_2, \cdots, a_n\}$ be a set of $n \geq 3$ nodes. We define a **tree coterie of depth two** over U by

$$Q = \{\{a_1, a_j\} \mid 2 \leq j \leq n\} \cup \{\{a_2, a_3, \cdots, a_n\}\}$$

Node a_1 is viewed as the root node and the remaining nodes are viewed as leaf nodes in the tree. Tree coteries are constructed by repeatedly composing tree

coteries of depth two together at one of the leaf nodes. Thus, any tree in which each nonleaf node has at least two children may be constructed.

For example, the above tree coterie may be represented by composing the following three tree coteries of depth two:

$$Q_1 = \{\{1, \mathbf{a}\}, \{1, \mathbf{b}\}, \{\mathbf{a}, \mathbf{b}\}\} \text{ under } U_1 = \{1, \mathbf{a}, \mathbf{b}\}$$
$$Q_2 = \{\{2,4\}, \{2,5\}, \{2,6\}, \{4,5,6\}\} \text{ under } U_2 = \{2,4,5,6\}$$
$$Q_3 = \{\{3,7\}, \{3,8\}, \{7,8\}\} \text{ under } U_3 = \{3,7,8\}$$

Let $Q_4 = T_{\mathbf{a}}(Q_1, Q_2)$, and $Q_5 = T_{\mathbf{b}}(Q_4, Q_3)$. Then, Q_5 under $U_5 = \{1, 2, \cdots, 8\}$ is the tree coterie corresponding to the tree shown in Figure 2.

We may use the quorum containment test on tree coteries. For example, suppose that we want to know if the set $S = \{1,3,6,7\}$ contains a quorum of Q_5.

$\mathrm{QC}(S, Q_5)$, where $S = \{1,3,6,7\}$ and $Q_5 = T_{\mathbf{b}}(Q_4, Q_3)$

$= \textbf{if } \mathrm{QC}(S, Q_3) \textbf{ then } \mathrm{QC}((S - U_3) \cup \{\mathbf{b}\}, Q_4)$
(Note that $\mathrm{QC}(S, Q_3)$ is true, because $\{3,7\} \subseteq S$ and $\{3,7\} \in Q_3$.)

$= \mathrm{QC}(S', Q_4)$, where $S' = \{1, 6, \mathbf{b}\}$ and $Q_4 = T_{\mathbf{a}}(Q_1, Q_2)$

$= \textbf{if } \mathrm{QC}(S', Q_2) \textbf{ then } \cdots \textbf{ else } \mathrm{QC}((S' - U_2), Q_1)$
(Note that $\mathrm{QC}(S', Q_2)$ is false.)

$= \mathrm{QC}(S'', Q_1)$, where $S'' = \{1, \mathbf{b}\}$ and Q_1 is a simple coterie

$=$ true, because $\{1, \mathbf{b}\} \in Q_1$.

Thus, S contains a quorum of Q_5.

3.2.2 Hierarchical quorum consensus

As stated in the introduction, in order to improve on the performance exhibited by quorum consensus, Kumar proposed *hierarchical quorum consensus* [9]. A complete tree of depth n is formed with the root at level 0. A physical node (a computer site or a copy of a replica) is assigned to each leaf node in the tree (at level n). In order to distinguish between nodes in the tree and physical nodes, we refer to nodes in the tree as **vertices**. A number of votes is assigned to each vertex, except for the root. A pair of thresholds is assigned to each level, except for level 0 (the root level). Let q_i (q_i^c) denote the quorum set (complementary quorum set) threshold assigned to level i. A quorum (complementary quorum) at level i is obtained by collecting at least q_{i+1} (q_{i+1}^c) votes from vertices at level $(i+1)$. By applying this method recursively from level 0 (the root) down to level $(n-1)$, a quorum (complementary quorum) of the system is constructed.

For example, consider the 9 nodes organized into a tree of depth 2 shown in Figure 3.

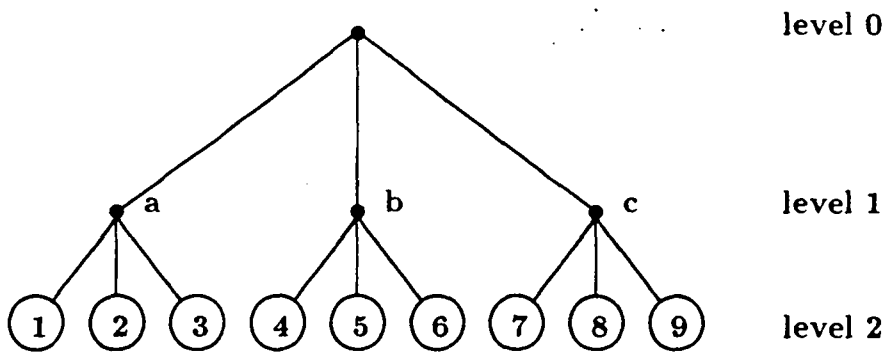

Figure 3. Tree

Suppose that each vertex is assigned a single vote. Table 1 is a list of possible threshold values and the resulting quorum sizes.

Table 1. Threshold Values

No.	q_1	q_1^c	q_2	q_2^c	$\|q\|$	$\|q^c\|$
1	3	1	3	1	9	1
2	3	1	2	2	6	2
3	2	2	3	1	6	2
4	2	2	2	2	4	4

Suppose that $q_1 = 3$, $q_1^c = 1$, $q_2 = 2$, and $q_2^c = 2$. Then, the corresponding quorum set and complementary quorum set are given by

$$Q = \{\ \{1,2,4,5,7,8\},\{1,2,4,5,7,9\},\{1,2,4,5,8,9\},\{1,2,4,6,7,8\},\\ \{1,2,4,6,7,9\},\{1,2,4,6,8,9\},\cdots,\{2,3,5,6,8,9\}\} \text{ and}$$

$$Q^c = \{\ \{1,2\},\{1,3\},\{2,3\},\{4,5\},\{4,6\},\{5,6\},\{7,8\},\{7,9\},\{8,9\}\}.$$

For example, {1,2,4,5,7,8} in Q is formed by collecting {**a**,**b**,**c**} at level 1 ($q_1 = 3$), and then by collecting {1,2}, {4,5}, and {7,8} for **a**, **b**, and **c**, respectively, at level 2 ($q_2 = 2$).

Since each node is assigned a single vote, the size of each quorum in the quorum set, $|q|$, is equal to the product of the thresholds.

Hierarchical quorum consensus may be generalized by using composition; the quorum sets are formed by repeatedly applying composition to quorum sets defined by quorum consensus. For instance, consider the example given above. Since $q_1 = 3$ and $q_1^c = 1$, let

$U_1 = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$, $Q_1 = \{\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}\}$, and $Q_1^c = \{\{\mathbf{a}\}, \{\mathbf{b}\}, \{\mathbf{c}\}\}$.

Next, suppose that $U_\mathbf{a} = \{1, 2, 3\}$, $U_\mathbf{b} = \{4, 5, 6\}$, and $U_\mathbf{c} = \{7, 8, 9\}$. Let $Q_\mathbf{a} = Q_\mathbf{a}^c = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$ because $q_2 = q_2^c = 2$. Similarly, define $Q_\mathbf{b}$, $Q_\mathbf{b}^c$, $Q_\mathbf{c}$, and $Q_\mathbf{c}^c$. Then, apply composition to generate the quorum sets, Q and Q^c, shown above; that is,

$$Q = T_\mathbf{c}(T_\mathbf{b}(T_\mathbf{a}(Q_1, Q_\mathbf{a}), Q_\mathbf{b}), Q_\mathbf{c}) \text{ and } Q^c = T_\mathbf{c}(T_\mathbf{b}(T_\mathbf{a}(Q_1^c, Q_\mathbf{a}^c), Q_\mathbf{b}^c), Q_\mathbf{c}^c).$$

3.2.3 Hybrid replica control protocols

Hybrid replica control protocols (or integrated protocols) are essentially methods to construct quorum sets by combining quorum consensus with a structured quorum protocol, such as the grid protocol or the tree protocol [1, 2]. If the grid protocol is used, the resulting protocol is called the *grid-set protocol.* On the other hand, if the tree protocol is used, the resulting protocol is called the *forest protocol.*

In the grid-set protocol, the nodes are organized as a set of square grids. Suppose there are n grids and m nodes on each grid. Then, there are a total of $n * m$ nodes. The quorum sets, in terms of grids, are defined by using quorums consensus. Suppose, q is the quorum set threshold and q^c is the complementary quorum set threshold such that

$$q + q^c \geq (n + 1) \text{ and } q \geq \lceil((1/2)(n + 1))\rceil.$$

In order to form a quorum of Q (Q^c), quorums of nodes must be obtained from at least q (q^c) grids.

For instance, consider the example shown in Figure 4. Suppose that we use the same quorum sets based on quorum consensus, as in Section 3.2.2; that is,

$$U_1 = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}, Q_1 = \{\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}\}, \text{ and } Q_1^c = \{\{\mathbf{a}\}, \{\mathbf{b}\}, \{\mathbf{c}\}\}.$$

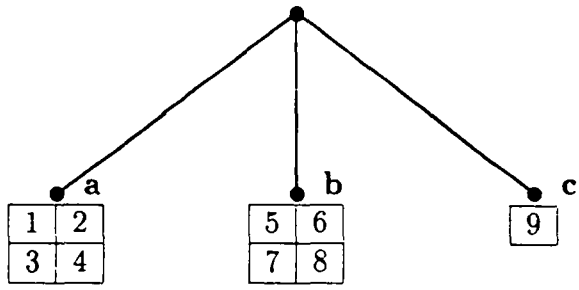

Figure 4. Grid-set protocol

Next, the grid protocol, described by Agrawal and El Abbadi, is applied to obtain:

$U_{\mathbf{a}} = \{1,2,3,4\}$, $U_{\mathbf{b}} = \{5,6,7,8\}$, $U_{\mathbf{c}} = \{9\}$,
$Q_{\mathbf{a}} = \{\{1,2,3\},\{1,2,4\},\{1,3,4\},\{2,3,4\}\}$, $Q_{\mathbf{a}}^c = \{\{1,2\},\{3,4\},\{1,3\},\{2,4\}\}$,
$Q_{\mathbf{b}} = \{\{5,6,7\},\{5,6,8\},\{5,7,8\},\{6,7,8\}\}$, $Q_{\mathbf{b}}^c = \{\{5,6\},\{7,8\},\{5,7\},\{6,8\}\}$,
$Q_{\mathbf{c}} = \{\{9\}\}$, and $Q_{\mathbf{c}}^c = \{\{9\}\}$.

The resulting quorum set and complementary quorum set are:

$$Q = \{\ \{1,2,3,5,6,7,9\},\{1,2,3,5,6,8,9\},\{1,2,3,5,7,8,9\},$$
$$\{1,2,3,6,7,8,9\},\cdots,\{2,3,4,6,7,8,9\}\} \text{ and}$$
$$Q^c = \{\ \{1,2\},\{3,4\},\{1,3\},\{2,4\},\{5,6\},\{7,8\},\{5,7\},\{6,8\},\{9\}\}.$$

For example, {1,2,3,5,6,7,9} in Q is formed by collecting {**a**,**b**,**c**} using quorum consensus, and then by collecting {1,2,3}, {5,6,7}, and {9} for **a**, **b**, and **c**, respectively, by using the grid protocol.

Using composition, Q and Q^c are obtained as follows:

$$Q = T_{\mathbf{c}}(T_{\mathbf{b}}(T_{\mathbf{a}}(Q_1, Q_{\mathbf{a}}), Q_{\mathbf{b}}), Q_{\mathbf{c}}) \text{ and } Q^c = T_{\mathbf{c}}(T_{\mathbf{b}}(T_{\mathbf{a}}(Q_1^c, Q_{\mathbf{a}}^c), Q_{\mathbf{b}}^c), Q_{\mathbf{c}}^c).$$

Note that Q^c is not maximal because $Q_{\mathbf{a}}^c$ and $Q_{\mathbf{b}}^c$ are not maximal; for instance, $\{1,4\} \cap G \neq \emptyset$ for all $G \in Q$. Thus, (Q, Q^c) is a dominated bicoterie.

3.2.4 Arbitrary network protocol

Composition provides a natural method for combining structures in an arbitrary network or collection of interconnected networks. For example, consider the graph depicting interconnected networks shown in Figure 5.

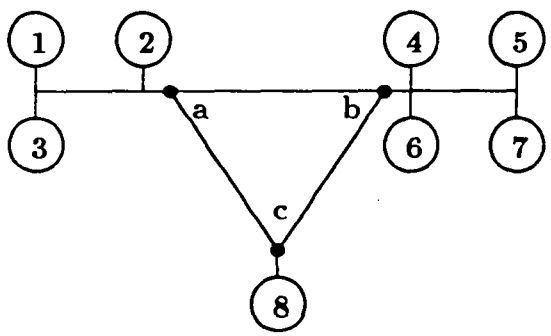

Figure 5. Example

There are three interconnected networks: **a**, **b**, and **c**. Suppose that locally, each network administrator has decided on a coterie to be used in a mutual exclusion algorithm. In order to provide a mutual exclusion algorithm over the entire collection of networks, composition may be used. For example, suppose that the coteries on each of the networks are defined as follows:

$Q_{\mathbf{a}} = \{\{1,2\},\{2,3\},\{3,1\}\}$,
$Q_{\mathbf{b}} = \{\{4,5\},\{4,6\},\{4,7\},\{5,6,7\}\}$, and
$Q_{\mathbf{c}} = \{\{8\}\}$.

Further, suppose that the coterie for the networks is given by:

$$Q_{\mathbf{net}} = \{\{\mathbf{a},\mathbf{b}\},\{\mathbf{b},\mathbf{c}\},\{\mathbf{c},\mathbf{a}\}\}.$$

That is, if a process requires mutually exclusive access to an object over the collection of networks, then permission must be obtained from any two of the networks. Thus, the coterie for the entire collection of networks (in terms of nodes) is given by:

$$Q = T_{\mathbf{c}}(T_{\mathbf{b}}(T_{\mathbf{a}}(Q_{\mathbf{net}}, Q_{\mathbf{a}}), Q_{\mathbf{b}}), Q_{\mathbf{c}}).$$

4 Conclusion

Composition not only provides a unifying framework for many existing protocols, but also allows us to define very general, application oriented quorums which may be used in any distributed system. The unifying aspect of composition is summarized in Table 2, where "$\oplus$" is used to denote the application of compostion.

Table 2. Summary

Protocol	Structures Formed By:
Hierarchical Quorum Consensus	Quourm Consensus $\oplus$ Quorum Consensus
Grid-set Protocol	Quorum Consensus $\oplus$ Grid Protocol
Forest Protocol	Quorum Consensus $\oplus$ Tree Protocol
Integrated Protocol	Quorum Consensus $\oplus$ Logical Unit
Composition	Any Protocol $\oplus$ Any Protocol

As we have seen, each of these protocols may be generalized by using composition, and the quorum containment test may be used to efficiently determine whether a given set of nodes contains a quorum. In addition, composition provides a natural method to generate quorum sets in an arbitrary network or in a collection of interconnected networks.

This paper has shown the generality offered by composition to define quorums well-suited for any distributed systems. The reader interested in more theoretical aspects of composition can consult the companion paper [13], in which theoretical results concerning the composition of coteries are presented.

References

[1] D. Agrawal and A. El Abbadi. Exploiting logical structures in replicated databases. *Information Processing Letters*, 33:255–260, 1990.

[2] D. Agrawal and A. El Abbadi. An efficient and fault tolerant solution for mutual exclusion. *ACM Transactions on Computer Systems*, 9(1):1–20, 1991.

[3] D. Barbara and H. Garcia-Molina. Mutual exclusion in partitioned distributed systems. *Distributed Computing*, 1:119–132, 1986.

[4] S.Y. Cheung, M. Ammar, and M. Ahamad. The grid protocol: a high performance scheme for maintaining replicated data. In *IEEE 6th International Conference on Data Engineering*, 1990.

[5] W.C.A. Fu. *Enhancing Concurrency and Availability for Database Systems.* PhD thesis, Simon Fraser University, Burnaby, B.C., Canada, April 1990.

[6] H. Garcia-Molina and D. Barbara. How to assign votes in a distributed system. *Journal of the ACM*, 32(4):841–860, 1985.

[7] D.K. Gifford. Weighted voting for replicated data. In *Proc. 7th ACM Symposium on Operating Systems Principles*, pages 150–162, 1979.

[8] T. Ibaraki and T. Kameda. Theory of coteries. Technical Report CSS/LCCR TR90-09, Kyoto University, Kyoto, JAPAN, 1990.

[9] A. Kumar. Performance analysis of a hierarchical quorum consensus algorithm for replicated objects. In *IEEE 10th International Conference on Distributed Computing Systems*, pages 378–385, 1990.

[10] L. Lamport. The implementation of reliable distributed multiprocess systems. *Computer Networks*, 2:95–114, 1978.

[11] M. Maekawa. A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems. *ACM Transactions on Computer Systems*, 3(2):145–159, 1985.

[12] M. Mizuno, M.L. Neilsen, and R. Rao. A token based distributed mutual exclusion algorithm based on quorum agreements. In *IEEE 11th International Conference on Distributed Computing Systems*, pages 361–368, 1991.

[13] M.L. Neilsen and M. Mizuno. Coterie join algorithm. *IEEE Transactions on Parallel and Distributed Systems*, in press.

[14] J. Tang and N. Natarajan. A static pessimistic scheme for managing replicated databases. Technical Report CS-90-07, Pennsylvania State University, University Park, Pennsylvania, 1990.

[15] R.W. Thomas. A majority consensus approach to concurrency control for multiple copy databases. *ACM Transactions on Database Systems*, 4(2):180–209, 1979.

ISSN 0249 - 6399