Quorum Structures for Fault-Tolerant Distributed Mutual Exclusion

Student: Jehn-Ruey Jiang

Advisor: Professor Shing-Tsaan Huang

July, 1995

Department of Computer Science
National Tsing Hua University
HsinChu, Taiwan, 30043
Republic of China

Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Computer Science

Abstract

Quorum-based algorithms are an important class of algorithms to achieve distributed mutual exclusion. They are resilient to network partitioning caused by site and/or network link failures and usually evoke low communication cost. The basic idea of them is simple—a site should collect permissions (votes) from all sites of a quorum to enter the critical section. If we can assure that any pair of quorums have a non-empty intersection and each site gives its permission to only one site at a time, mutual exclusion is then guaranteed.

The collection of quorums used by a quorum-based algorithm is called a *quorum structure*. According to different mutual exclusion scenarios, several types of quorum structures have been proposed: coterie, *wr*-coterie and *k*-coteries, which are related to distributed mutual exclusion, replicated data consistency and distributed *k*-mutual exclusion, respectively.

In this dissertation, we propose novel methods for constructing coteries, *wr*-coteries and *k*-coteries that are *nondominated* and/or of constant expected quorum size. The proposed methods can easily be extended to solve the problems of mutual exclusion, replicated data consistency or *k*-mutual exclusion in a distributed system. Nondominated quorum structures are favorable because they are candidates to achieve the optimal availability, the probability that a quorum can be form in an error-prone environment. On the other hand, quorum structures of constant expected quorum size are preferable because when the proposed methods are applied to solve the problems mentioned, the message cost is directly proportional to the quorum size.

Keywords: Distributed systems, fault-tolerance, *k*-mutual exclusion, mutual exclusion, replica control, quorum structures.

Contents

Chapter 1 Introduction to quorum structures 1

1.1 Coteries..3

1.2 *WR*-Coteries..4

1.3 *K*-Coteries...5

1.4 Nondominance of quorum structures 6

1.5 Dissertation organization...7

Chapter 2 Constructing *ND* coteries of constant expected quorum size 9

2.1 Introduction 9

2.2 Preliminaries of coteries...11

2.3 Construction of quorums..13

2.4 Correctness 14

2.5 Analysis and comparison..17

2.5.1 Availability..17

2.5.2 Quorum size 19

2.5.3 Comparison 21

2.6 Summary...23

Chapter 3 Constructing *ND wr*-coteries of constant expected quorum size 29

3.1 Introduction 29

3.2 Preliminaries of *wr*-coteries 31

3.3 Construction of quorums 33

3.4 Correctness 35

3.5 Analysis and comparison 39

3.5.1 Availability 39

3.5.2 Quorum size 42

3.5.3 Comparison 44

3.6 Summary 48

Chapter 4 Constructing *k*-coteries of constant expected quorum size 55

4.1 Introduction 55

4.2 Preliminaries of *k*-coteries 56

4.3 Construction of quorums 58

4.4 Correctness 59

4.5 Analysis and comparison 62

4.5.1 Availability 63

4.5.2 Quorum size 64

4.5.3 Comparison 67

4.6 Summary 68

Chapter 5 Constructing *ND k*-coteries from known *ND k*-coteries..................71

5.1 Introduction 71

5.2 Related work...73

5.3 *ND k*-coteries 78

5.4 The join and union operations 85

5.4.1 Coterie union operation...86

5.4.2 Coterie join operation 89

5.5 Concluding remarks...95

Chapter 6 Conclusion and future work 97

6.1 Conclusion...97

6.2 Future work 99

Bibliography 101

List of Tables

Table 2.1 Bounds on quorum sizes for various coteries 25

Table 3.1 Bounds on quorum sizes for various *wr*-coteries...................................49

Table 4.1 Bounds on quorum sizes for the cohort coterie and the *k*-majority coterie 69

List of Figures

Figure 2.1 A function that can generate minimal quorums under *Coh*(*l*)..................25

Figure 2.2 The availability of quorums under various Cohorts structures 26

Figure 2.3 The availability comparison of various coteries for the 15-site system ...27

Figure 2.4 The availability comparison of various coteries for the 31-site system ...28

Figure 3.1 Functions that can generate read and write quorums under *Coh*(*l*)..........50

Figure 3.2 The availability of read quorums under various Cohorts structures.........51

Figure 3.3 The availability of write quorums under various Cohorts structures 52

Figure 3.4 The availability comparison of various *wr*-coteries for the 15-replica system 53

Figure 3.5 The availability comparison of various *wr*-coteries for the 31-replica system 54

Figure 4.1 A function that can generate quorums under *Coh*(*k*,*l*) 69

Figure 4.2 The (*k*,*h*)-availability comparision of the cohort coterie (CC) and the *k*-majority coterie (*k*-MC) for the 53-site system 70

Chapter 1

Introduction to quorum structures

A distributed system consists of autonomous, interconnected computers (called sites) that can communicate with each other by exchanging messages. The problem of mutual exclusion is essential in the design of distributed systems. It is concerned with how to control sites to have mutually exclusive access of a designated section of code called *critical section*. Quorum-based algorithms are an important class of algorithms for solving distributed mutual exclusion. They are resilient to network partitioning [DGS85] caused by site and/or network link failures and usually evoke low communication cost. The basic idea of them is simple—a site should collect permissions (votes) from all sites of a quorum to enter the critical section. If we can assure that any pair of quorums have a non-empty intersection and each site gives its permission to only one site at a time, mutual exclusion is then guaranteed.

The collection of quorums used by a quorum-based algorithm is called a *quorum structure*. According to different mutual exclusion scenarios, several types of quorum structures have been proposed: coterie [GB85], *wr*-coterie [IK93], and *k*-coteries [FYA91], which correspond to the problems of mutual exclusion, replicated data consistency and *k*-mutual exclusion in distributed systems, respectively. There are many researches concentrating on quorum structures: some for developing new methods to construct quorum structures with good characteristics (say, high availability or small quorum size) [AE90, AE91, AE92a, AE92b, AJ92, CAA92, Chu94, Gif79, HJK93, Jia95, JH94, KC91, Kum91, Mae85, Nei92, Nei93, RST92, SW93a, Tho79, WB92, Wu93a, Wu93b], some for developing new measures for quorum structures [AA89, BG86, BG87, CY94a, CY94b, KFYA93, KRS92, KRS93, Nei93, RJT93], some for transforming quorum structures into new ones [GB85, JH94, NM92, SW93b], and still some for developing theories about a special class of quorum structures—*nondominated* (*ND*) quorum structures [GB85, IK93].

In this dissertation, we propose novel methods for constructing coteries, *wr*-coteries and *k*-coterie that are *nondominated* and/or of constant expected quorum size. The proposed methods can easily be extended to solve the problems of mutual exclusion, replicated data consistency and *k*-mutual exclusion in distributed systems, respectively. Note that nondominated quorum structures are favorable because they are candidates for achieving the optimal (highest) availability, the probability that a quorum can be formed in an error-prone environment. And constant quorum size is preferable because when the proposed methods are applied to solve the problems just mentioned, the message cost is directly proportional to the quorum size.

Below, we introduce the concept of coteries, *wr*-coteries and *k*-coteries. Note that we assume *U* is the underlying set containing all system sites $u_1,...,u_n$, and we may not specify *U* wherever there is no ambiguity.

1.1 Coteries

A *coterie* [GB85] ***C*** is a family of non-empty subsets of *U*. Each member in ***C*** is called a *quorum*; the following properties should hold for the quorums:

Intersection Property:

$\forall G, \forall H$: $G, H \in \mathrm{C}$: $G \cap H \neq \emptyset$;

Minimality Property:

$\forall G, \forall H$: $G, H \in \mathrm{C}$: $G \not\subset H$.

For example, $C = \{\{u_1, u_2\}, \{u_1, u_3\}, \{u_2, u_3\}\}$ is a coterie because every pair of quorums have a non-empty intersection, and no quorum is a super set of another quorum.

By the intersection property, a coterie can be used to develop algorithms for mutual exclusion in a distributed system. To access the critical section, a site is required to receive permissions (votes) from all the sites of some quorum. Since any pair of quorums have at least one site in common and every site grants its permission to only one site at a time, mutual exclusion is then guaranteed. Note that the minimality property is not necessary for the correctness of mutual exclusion but is used to enhance efficiency. Mutual exclusion algorithms using coteries are fault-tolerant in the sense that even when network partitioning [DGS85] occurs and makes some sites inaccessible, quorums not including inaccessible sites may still be constructed.

1.2 *WR*-Coteries

A *wr*-coterie [IK93] (W, R) is a pair of two families of non-empty quorums (subsets of U) satisfying

Write-Write Intersection Property:

$\forall G, \forall H$: $G,H \in W$: $G \cap H \neq \emptyset$;

Write-Read Intersection Property:

$\forall G, \forall H$: $G \in W$, $H \in R$: $G \cap H \neq \emptyset$;

Write Quorum Minimality Property:

$\forall G, \forall H$: $G,H \in W$: $G \not\subset H$;

Read Quorum Minimality Property:

$\forall G, \forall H$: $G,H \in R$: $G \not\subset H$.

For example, let $W=\{\{u_1,u_2,u_3\}, \{u_1,u_2,u_4\}, \{u_3,u_4\}\}$ and $R=\{\{u_1,u_3\}, \{u_1,u_4\}, \{u_2,u_3\}, \{u_2,u_4\}, \{u_3,u_4\}\}$, then the pair ($W$, R) is a *wr*-coterie since it satisfies all the above-mentioned properties.

In a distributed system, data can be replicated at different sites to tolerate site and/or network link failures. However, complex replica control algorithms are required to make multiple replicas of a data object behave as a single one, i.e., to ensure *one-copy equivalence* [BHG87]. Below, we describe how one-copy equivalence is achieved by *wr*-coterie (W, R). Each replica is associated with a *version number*. A read operation should *read-lock* and access replicas of a read quorum (a quorum in R) and return the replica owning the largest version number. On the other hand, a write operation should *write-lock* and access replicas of a write quorum (a quorum in W), and then updates them and assigns them with the new version number which is one more than the largest version number just encountered. Since any pair of a read and a write quorum and any two write quorums have a non-empty intersection, the read operation will always return the most up-to-date replica. Again, the minimality properties of *wr*-coteries are used to enhance efficiency. Replica control algorithms using *wr*-coteries are fault-tolerant in the sense that even when network partitioning [DGS85] occurs and makes some replicas inaccessible, quorums not including inaccessible replicas may still be found.

1.3 *K*-Coteries

A *k-coterie* [FYA91] ***C*** is a family of non-empty quorums (subsets of *U*) satisfying

Non-intersection Property:

For any h ($< k$) pairwise disjoint quorums $Q_1,...,Q_h$ in ***C***, there exists one quorum Q_{h+1} in ***C*** such that $Q_1,...,Q_{h+1}$ are pairwise disjoint;

Intersection Property:

There are no m, $m > k$, pairwise disjoint quorums in ***C*** (i.e., there are at most k pairwise disjoint quorums in ***C***);

Minimality Property:

There are no two quorums Q_1 and Q_2 in ***C*** such that Q_1 is a super set of $Q_{2.}$

For example, $\{\{u_1,u_2\}, \{u_3,u_4\}, \{u_1,u_3\}, \{u_2,u_4\}\}$ is a 2-coterie because it satisfies all the properties of a 2-coterie—given one quorum Q_1, we can always find another quorum Q_2 such that Q_1 and Q_2 are disjoint; there are at most two pairwise disjoint quorums; and every quorum is not a super set of another quorum. The reader should note that an 1-coterie (the value of *k* is taken as 1) is exactly a coterie introduced in Section 1.1.

By the intersection and the non-intersection properties, *k*-coteries can be used to develop distributed *k*-mutual exclusion algorithms which allow at most *k* sites in the critical section simultaneously. To access the critical section, a site is required to obtain permissions from all the sites of some quorum. By the intersection property, no more than *k* sites can form quorums simultaneously, so no more than *k* sites can access the critical section at the same time. The non-intersection property assures that if there exists one unoccupied critical section entry, then some site that waits for accessing the critical section can proceed. Again, the minimality property of the *k*-coterie is not for the correctness but for the efficiency. The *k*-mutual exclusion algorithm using *k*-coteries is fault-tolerant in the sense that even when network partitioning [DGS85] occurs and makes some sites inaccessible, quorums including only available sites may still be formed.

1.4 Nondominance of quorum structures

A quorum structure is said to *dominate* [GB85] another quorum structure if and only if every quorum in the dominated one is a super set of some quorum in the dominating one. Obviously, the dominating one has more chance than the dominated one for a quorum to be formed in an error-prone environment. Thus, if optimizing the availability is the main concern, we should always concentrate on *nondominated* quorum structures that no one else can dominate. However, it is very difficult to verify that a quorum structure is nondominated. The verification is usually done on the basis of Garcia-Molina and Barbara's theorem [GB85] and Ibaraki and Kameda's theorem [IK93].

Below, we illustrate the nondominance concept of quorum structures with an example of coteries.

Let C and D be two coteries. *C dominates D* iff ($C \neq D$) and ($\forall G, \exists H$: $G \in D, H \in C$: $H \subseteq G$).

A coterie is said to be *nondominated* if and only if no coterie can dominate it.

For example, consider the following two coteries under $U=\{u_1,u_2,u_3,u_4\}$:

$C = \{\{u_1, u_2\}, \{u_1, u_3\}, \{u_1, u_4\}, \{u_2, u_3, u_4\}\}$

$D = \{\{u_1, u_2, u_3\}, \{u_1, u_2, u_4\}, \{u_1, u_3, u_4\}, \{u_2, u_3, u_4\}\}$.

It is easy to see that C dominates D because for every quorum in C, we can find its super set in D. Thus, if we can form a quorum in D, then we can also form a quorum in C. Particularly, C is a coterie constructed by the method proposed in Chapter 2, it will be shown to be nondominated later.

1.5 Dissertation Organization

The organization of the dissertation is as follows:

In Chapter 2, we propose a method for constructing a class of nondominated coteries of constant expected quorum size and prove the correctness of the method with Ibaraki and Kameda's theorem [IK93]. We analyze the constructed coteries in terms of quorum availability and quorum size; the analyzed results are also compared with those of related coteries.

In Chapter 3, we propose a method for constructing a class of nondominated *wr*-coteries of constant expected quorum size and prove the correctness of the method with Ibaraki and Kameda's theorem [IK93]. We analyze the constructed *wr*-coteries in terms of quorum availability and quorum size; the analyzed results are also compared with those of related *wr*-coteries.

In Chapter 4, we propose a method for constructing a class of *k*-coteries of constant expected quorum size. We prove the correctness of the method and analyze the constructed *k*-coteries in terms of quorum availability and quorum size; the analyzed results are also compared with those of related *k*-coteries.

In Chapter 5, we develop a theorem that can be used to check the nondominance of a *k*-coterie. On the basis of the theorem, we prove that some *k*-coteries are nondominated. Moreover, we propose two methods (operations) which can generate nondominated *k*-coteries from known nondominated *k*-coteries.

In Chapter 6, we give a summary of this dissertation and address further research directions.

Chapter 2
Constructing *ND* coteries of constant expected quorum size

2.1 Introduction

Quorum-based algorithms are an important class of algorithms to achieve mutual exclusion in distributed systems. Such algorithms usually incur low message cost and can tolerate site and/or network link failures, even when these failures lead to network partitioning [DGS85]. The basic idea of this type of algorithms is simple—a site should collect permissions (votes) from all sites of a quorum to enter the critical section. Mutual exclusion is guaranteed if we can assure that any pair of quorums have at least one common site and that a site gives its permission to only one site at a time. The majority quorum consensus algorithm [Tho79], the tree quorum algorithm [AE91] and the hierarchical quorum consensus algorithm [Kum91] are typical quorum-based algorithms.

The coterie concept [GB85] is usually used to formalize quorum-based mutual exclusion algorithms. A coterie [GB85] is a family of quorums (sets) with the property that any pair of quorums have a non-empty intersection. Among all the coteries, nondominated (*ND*) coteries [GB85] are preferable because they are candidates to achieve the highest availability, the probability that a quorum can be formed. Some classes of coteries, such as the majority coterie (MC), the tree coterie (TC), the hierarchical coterie (HC) and the Lovasz coterie (LC) [Nei93] have been shown to be *ND*. Note that the first three classes of coteries correspond to the majority quorum consensus algorithm [Tho79], the tree quorum algorithm [AE91] and the hierarchical quorum consensus algorithm [Kum91], respectively.

In this chapter, we propose a method to construct quorums of an *ND* coterie; the method can easily be extended to be a solution to distributed mutual exclusion. The method utilizes a logical structure named *Cohorts* to construct quorums of O(1) (constant) size in the best case. When some sites are inaccessible, the quorum size increases gradually and may be as large as O(n), where n is the number of sites. However, the expected quorum size is shown to remain constant as n grows. This is a desirable property since the message cost for accessing the critical section is directly proportional to the quorum size. In addition, the availability of the constructed quorum is shown to be asymptotically high. With the two properties—constant expected quorum size and asymptotically high availability, the proposed method is thus applicable to systems possessing an increasing number of sites. We also analyze and compare the constructed quorums with others in terms of availability and quorum size.

The rest of this chapter is organized as follows. In Section 2.2, we elaborate some preliminaries of coteries. Then, in Section 2.3, we present the Cohorts structure and show how to construct quorums with its aid. In Section 2.4, we show that the collection of the constructed quorums is a nondominated coterie. In Section 2.5, we analyze and compare the constructed quorums with others in terms of availability and quorum size. At last, we conclude this chapter with Section 2.6

2.2 Preliminaries of coteries

In this section, we show some preliminaries of coteries. In the following discussion, we assume $u_1,...,u_n$ are all system sites and let $U = \{u_1,...,u_n\}$ be the underlying set that contains all system sites.

A *coterie* ***C*** is a family of subsets of *U*. Each member in ***C*** is called a *quorum* and should observe the following two properties:

Intersection Property:

$\forall G, \forall H$: $G,H \in C$: $G \cap H \neq \emptyset$;

Minimality Property:

$\forall G, \forall H$: $G,H \in C$: $G \not\subset H$.

For example, $C = \{\{u_1, u_2\}, \{u_1, u_3\}, \{u_2, u_3\}\}$ is a coterie under $U=\{u_1, u_2, u_3\}$ because every pair of quorums have a non-empty intersection, and no quorum is a super set of another quorum.

By the intersection property, the coterie can be used to develop algorithms for mutual exclusion in distributed systems. To enter the critical section, a site is required to receive the permissions (votes) from all sites of some quorum. Since any pair of quorums have at least one site in common and every site grants its permission to only one site at a time, mutual exclusion is then guaranteed. The reader should note that the minimality property is not necessary for the correctness of mutual exclusion but is used to enhance efficiency.

Let *C* and *D* be two distinct coteries. *C* is said to *dominate D* iff $\forall G, \exists H$: $G \in D$, $H \in C$: $H \subseteq G$. For example, coterie $C = \{\{u_1, u_2\}, \{u_1, u_3\}, \{u_1, u_4\}, \{u_2, u_3, u_4\}\}$

dominates coterie $D = \{\{u_1, u_2, u_3\}, \{u_1, u_2, u_4\}, \{u_1, u_3, u_4\}, \{u_2, u_3, u_4\}\}$ because for every quorum G in D we can find a quorum H in C such that G is a super set of H. A dominating coterie, such as C, is more resilient to site and/or network link failures than a dominated coterie, such as D since if a quorum can be formed in the dominated one then a quorum can be formed in the dominating one. A coterie is *nondominated* (*ND*) if no other coterie can dominate it. We should always concentrate on *ND* coteries because they are candidates to achieve the highest availability.

In Ibaraki and Kameda's work [IK93], any subset of U is represented by an n-tuple vector X, $X=(x_1,...,x_n) \in \{0,1\}^n$ where x_i is 1(resp., 0) if u_i is in (resp., not in) the subset. Let C be a family of subsets of U. Then, a Boolean function $f_C : \{0,1\}^n \rightarrow \{0,1\}$ associated with C is defined as $f_C(X) \equiv \bigvee_{Q \in C} \{ \bigwedge_{u_i \in Q} u_i \}$. Note that we follow the convention in [IK93] and use u_i (which is an element of U) as the ith component of vector X. The function f_C so defined has the property: $f_C(X)=1$ if vector X represents a super set of some quorum in C; otherwise $f_C(X)=0$. The dual f^d of a Boolean function f is defined as $f^d = f'(X')$, where X' and f' are complements of X and f, respectively. For example, under $U=\{u_1, u_2, u_3\}$, the set $\{u_1, u_2\}$ is represented as (1,1,0); and $\{u_2, u_3\}$, as (0,1,1). Let $C=\{\{u_1, u_2\}, \{u_2, u_3\}, \{u_1, u_3\}\}$, then $f_C(X) = (u_1u_2 \vee u_2u_3 \vee u_1u_3)$. $f_C^d(X)= f'(X')=(u_1' u_2' \vee u_2' u_3' \vee u_1' u_3')' = (u_1' u_2')' (u_2' u_3')' (u_1' u_3')' =(u_1 \vee u_2) (u_2 \vee u_3) (u_1 \vee u_3) = (u_1u_2 \vee u_2u_3 \vee u_1u_3)$.

The association of a Boolean function with a family of sets provides a facile way for checking some properties of the family. For example, the following Theorem 2.1 is actually Theorem 2.2 in [IK93] which can be used to check whether a family of sets is an *ND* coterie. For example, with Theorem 2.1, we can show that C, $C=\{\{u_1,$

u_2}, {u_2, u_3}, {u_1, u_3}}, is an *ND* coterie since $f_C(X) = f_C^d(X)$, as shown in the last paragraph.

Theorem 2.1. Let *C* be a family of non-empty subsets of *U* satisfying the minimality property. Then, *C* is an *ND* coterie if and only if $f_C = f_C^d$.

2.3 Construction of quorums

In this section, we present the Cohorts structure and show a method (function *Get_Quorum* in Figure 2.1) that can generate quorums by organizing system sites into a Cohorts structure.

A ***Cohorts structure*** $Coh(l)=(C_1,...,C_l)$ is a list of subsets of *U*. Each member C_i is called a *cohort* and should observe the following properties:

(P1) $|C_1| = 1$.

(P2) $\forall i$: $2 \le i \le l$: $|C_i| \ge 2$.

(P3) $\forall i$: $1 \le i \le l$:$C_i \not\subset \bigcup_{j \ne i} C_j$.

To sum up, the first cohort in a Cohorts structure should have only one member with other cohorts having at least two members and each cohort should have at least one unique member that does not appear in any other cohort. For example, ({u_1}) is a *Coh*(1), ({u_1},{u_2,u_3,u_4}) is a *Coh*(2) and ({u_1},{u_2, u_3},{u_3, u_4}) is a *Coh*(3).

For a Cohorts structure $Coh(l)=(C_1,...,C_l)$, a set *Q* is said to be a **quorum under *Coh*(*l*)** if *Q* satisfies both (D1) and (D2).

(D1) *Q* contains all the members of some cohort C_i, $1 \le i \le l$ (we say that *Q* ***fully covers*** C_i or that C_i is *Q*'s ***primary cohort***).

(D2) Q contains at least one member of each cohort C_j, $i<j\leq k$ (we say that Q ***covers*** C_j or that C_j is Q's ***supporting cohort***).

For example, under *Coh*(2)=($\{u_1\}$,$\{u_2,u_3,u_4\}$), the possible quorums are $\{u_1, u_2\}$, $\{u_1, u_3\}$, $\{u_1, u_4\}$ and $\{u_2, u_3, u_4\}$. For a quorum under *Coh*(l)=(C_1,...,C_l), the less is the index of the primary cohort, the smaller is the quorum size. In an extreme case, if C_l is the primary cohort, then no supporting cohort is necessary. In such a case, the quorum size is a constant $|C_l|$. In another extreme case, if C_1 is the primary cohort with the other cohorts being supporting cohorts, then the quorum may be of size O(n). To sum up, a quorum under *Coh*(l) is of constant size in the best case, and of O(n) size in the worst case.

A function named *Get_Quorum*, which can produce quorums under *Coh*(l), is shown in Figure 2.1. Function *Get_Quorum* can easily be modified and extended to solve the distributed mutual exclusion problem. In such a case, as in other quorum-based algorithms, a site is allowed to access the critical section after obtaining permissions from all sites of a quorum; a site is to return all its obtained permissions on leaving the critical section. Since a site may hold some permissions while waiting for other permissions, deadlock may thus occur. Mechanism proposed in [Mae85] or [San87] can be incorporated for avoiding deadlock (and starvation); however, the details are not our focus and are thus omitted.

2.4 Correctness

In this section, we show that the collection of quorums returned by function *Get_Quorum* is an *ND* coterie. We start by showing that *Get_Quorum* returns minimal quorums. Note that a quorum Q is said to be *minimal* if and only if any proper subset of Q is not a quorum.

Lemma 2.1. (Minimality property) The quorums returned by *Get_Quorum* are minimal.

Proof:

Let Q_1 and Q_2 be two quorums returned by *Get_Quorum* such that Q_1=*Min*(R_1, C_i,...,C_l) and Q_2=*Min*(R_2, C_i,...,C_l), where R_1 and R_2 are sets of sites that grant permissions. We have $C_i \subseteq Q_1$, $Q_1 \subseteq (C_i \cup ... \cup C_l)$, $C_j \subseteq Q_2$ and $Q_2 \subseteq (C_j \cup ... \cup C_l)$. Below, we want to show that neither $Q_2 \subset Q_1$ nor $Q_1 \subset Q_2$. There are three cases to consider: (1) i=j, (2) i<j and (3) i >j.

Case (1). $i = j$.

It is trivial that neither $Q_2 \subset Q_1$ nor $Q_1 \subset Q_2$ since function *Min* removes all the sites from R_1 and R_2 that are not essential for coverage of C_{i+1}(or C_{j+1}),...,C_l and full coverage of C_i (or C_j).

Case (2). $i < j$. The proof is by contradiction.

Assume $Q_1 \subset Q_2$, we have $C_i \subset (C_j \cup ... \cup C_l)$ because $C_i \subseteq Q_1$, $Q_1 \subset Q_2$ and $Q_2 \subseteq (C_j \cup ... \cup C_l)$. Contradiction occurs since $C_i \subset (C_j \cup ... \cup C_l)$ violates (P3). On the other hand, assume $Q_2 \subset Q_1$, we have $C_j \subset Q_1$ because $C_j \subseteq Q_2$ and $Q_2 \subset Q_1$. By (P3), there exists one member u in C_j such that u does not belong to any other cohorts. Since $C_j \subset Q_1$, all the sites, including u, in C_j belong to Q_1; i.e., function *Min* returns Q_1 with u involved. Because u only belongs to C_j and function *Min* does not remove u from Q_1, we have that $Q_1-\{u\}$ does not cover C_j. By $C_j \subset Q_1$ and $(Q_1-\{u\}) \cap C_j = \emptyset$ ($Q_1-\{u\}$ does not cover C_j), we have $C_j=\{u\}$. Contradiction occurs since $C_j=\{u\}$ violates (P2).

Case (3). $i > j$. The proof of this case is similar to that of case (2) and is omitted. □

By now, we have proved that *Get_Quorum* generates minimal quorums under *Coh*(l)=(C_1,...,C_l). Let C(l) be the collection of all minimal quorums under *Coh*(l). Below, we further prove that C(l) is an *ND* coterie by showing $f_{C(l)} = f^d_{C(l)}$ with some Boolean algebra laws [Lip79]. Note that later we call C(l) *cohort coterie*.

Lemma 2.2. $f_{C(l)} = f^d_{C(l)}$, for $l \geq 1$.

Proof: The proof is by induction on the value of l.

Basis (l=1):

By (P1), *Coh*(1)=({u_1}), from which the only derived quorum is {u_1}. So, we have $f_{\mathrm{C}(1)} = u_1$. The theorem holds for the basis case because $f_{C(1)} = f^d_{C(1)}$.

Induction Hypothesis:

We assume that $f_{C(l)} = f^d_{C(l)}$, for some l, $l \geq 1$.

Induction Step:

Consider *Coh*(l+1)=(C_1,...,C_{l+1}). Let C_{l+1}={u_1,...,u_m}, where m>1. By (D1) and (D2), a quorum under *Coh*(l+1) is composed of either (form-1) all sites in C_{l+1} or (form-2) one of the sites in C_{l+1} and a quorum under *Coh*(l). Thus, we have

$$f_{\mathrm{C}(l+1)} = (\bigwedge_{i=1,\dots,m} u_i) \vee (\bigvee_{i=1,\dots,m} u_i\, f_{\mathrm{C}(l)})$$

$$= (\bigwedge_{i=1,\dots,m} u_i) \vee (f_{\mathrm{C}(l)} \wedge (\bigvee_{i=1,\dots,m} u_i)) \qquad \text{(by commutative law and distributive law)}$$

Therefore, we have

$$f^d_{C(l+1)} = ((\bigwedge_{i=1,\dots,m} u_i') \vee (f_{\mathrm{C}(l)}(X') \wedge (\bigvee_{i=1,\dots,m} u_i')))' \text{(by the definition of Dual of a function)}$$

$$= (\bigwedge_{i=1,\dots,m} u_i')' \wedge (f_{\mathrm{C}(l)}(X') \wedge (\bigvee_{i=1,\dots,m} u_i'))' \qquad \text{(by De Morgan's law)}$$

$$= (\bigwedge_{i=1,\dots,m} u_i')' \wedge (\ f'_{C(l)}\ (X') \vee (\bigvee_{i=1,\dots,m} u_i')') \qquad \text{(by De Morgan's law)}$$

$$= (\bigvee_{i=1,\dots,m} u_i'') \wedge (\ f'_{C(l)}\ (X') \vee (\bigwedge_{i=1,\dots,m} u_i'') \qquad \text{(by De Morgan's law)}$$

$$= (\bigvee_{i=1,\dots,m} u_i) \wedge (\ f'_{C(l)}\ (X') \vee (\bigwedge_{i=1,\dots,m} u_i) \qquad \text{(by involution law, i.e., } u_i'' = u_i \text{)}$$

$= (\bigvee_{i=1,\ldots,m} u_i) \wedge (f_{C(l)}(X) \vee (\bigwedge_{i=1,\ldots,m} u_i))$ (since by hypothesis, $f_{C(l)} = f^d_{C(l)}$)

$= ((\bigvee_{i=1,\ldots,m} u_i) \wedge f_{C(l)}(X)) \vee ((\bigvee_{i=1,\ldots,m} u_i) \wedge (\bigwedge_{i=1,\ldots,m} u_i))$ (by distributive law)

$= ((\bigvee_{i=1,\ldots,m} u_i) \wedge f_{C(l)}(X)) \vee (\bigwedge_{i=1,\ldots,m} u_i)$ (since $(\bigvee_{i=1,\ldots,m} u_i) \wedge (\bigwedge_{i=1,\ldots,m} u_i) = (\bigwedge_{i=1,\ldots,m} u_i)$)

$= (\bigwedge_{i=1,\ldots,m} u_i) \vee (f_{C(l)} \wedge (\bigvee_{i=1,\ldots,m} u_i))$ (by commutative law)

$= f_{C(l+1)}$

Therefore, by the induction principle, we have $f_{C(l)} = f^d_{C(l)}$ for any l, $l \geq 1$. □

Theorem 2.2. $C(l)$ is an *ND* coterie for $l \geq 1$.
Proof: This is a direct consequence of Theorem 2.1, Lemma 2.1 and Lemma 2.2. □

2.5 Analysis and comparison

In this section, we analyze the availability and the size of quorums under *Coh*(l). We also compare the analyzed results with those of the quorums of the majority, the tree, the hierarchical and the Lovasz coteries. To simplify the analysis, we just discuss the Cohorts structures that have disjoint cohorts, i.e., we assume $C_i \cap C_j = \emptyset$, $i \neq j$, for *Coh*(l)=(C_1,...,C_l).

2.5.1 Availability

The availability of a coterie is defined as the probability that a quorum can be successfully formed in an error-prone environment. In homogeneous systems, every site has the same *up-probability p,* which stands for the probability that a single site is *up* (i.e., available). Let *AV*(l) be the function evaluating the availability of the quorum under *Coh*(l)=(C_1,...,C_l). Below, we show how to evaluate *AV*(l).

For l>1, if all the sites in C_l are up, then a quorum under *Coh*(l) can be formed. On the other hand, if at least one site but not all the sites in C_l are up, then one of the up sites together with a quorum under *Coh*(l−1) can form a quorum under *Coh*(l). For l>1, we have

$$\begin{aligned} AV(l) &= Prob.(\text{all sites in } C_l \text{ are up}) + \\ &\quad Prob.(\text{at least one site but not all sites in } C_l \text{ are up}) \times AV(l-1) \\ &= p^{S_l} + (1- p^{S_l} - (1-p)^{S_l})AV(l-1) \qquad (2.1) \end{aligned}$$

For l=1, the only sites in C_1 (note that by (P1) $|C_1|$=1) being up is necessary to form a quorum under *Coh*(1). Thus, we have $AV(1)=p$.

Below, we restrict each of cohorts $C_2,...,C_l$ to have the same size s to further simplify the analysis. That is, we assume $S_2=...=S_l=s$ for $Coh(l)=(C_1,...,C_l)$ (by (P2) $s \geq 2$). We denote such a Cohorts structure as *Coh*(l,s) and the value of s is called the *cohort size*. When *Coh*(l,s) is considered, the recursive equation (2.1) can be regarded as a first-order linear difference equation [DOSE86][*], which can be solved analytically. We have

$$AV(l)=(1-p^s-(1-p)^s)^{l-1}(p-p^s/(p+(1-p)^s)) \;+\; (p^s/(p^s+(1-p)^s)) \qquad (2.2)$$

We first apply equation (2.2) to investigate the influence of cohort sizes on quorum availability under a fixed number of sites. We assume the following Cohorts structures for a 31-site system: *Coh*(16,2), *Coh*(11,3), *Coh*(7,5), *Coh*(6,6), *Coh*(4,10) and *Coh*(3,15). The quorum availabilities corresponding to those structures are depicted in Figure 2.2, which reveals that smaller cohort sizes usually render the availability higher. Thus, we suggest adopting small cohort sizes, say 3 or 5. We do not suggest adopting the cohort size of 2, which leads to lower availability than those resulting from sizes of 3 and 5 for large up-probability p (e.g., for p>0.5). Note that most practical systems have large up-probability p, under which the cohort size of 2

[*] A first-order linear difference equation of the form $X_k=aX_{k-1}+b$ for $k \geq 2$ with X_1 being the first term has as its kth term $X_k=a^{k-1}(X_1+b/(a-1))-(b/(a-1))$ if $a \neq 1$.

causes a relatively large probability of no site in a cohort being up, which prohibits the construction of any quorum.

We now apply equation (2.2) to investigate the asymptotic value of quorum availability. When l goes to infinity, the term $(1-p^s-(1-p)^s)^{l-1}$ goes to 0, and $AV(l)$ goes to $p^s/(p^s+(1-p)^s)=1/(1+((1-p)/p)^s)$. In other words, the asymptotic availability of quorums under $Coh(l,s)$ is $1/(1+((1-p)/p)^s)$. For p=0.5, the asymptotic availability is 0.5 whatever the column size is. For p<0.5, $(1-p)/p$ is larger than 1 and thus $((1-p)/p)^s$ increases as s grows. It is easy to see that the smaller s is, the larger the asymptotic availability is. For p>0.5, $(1-p)/p$ is less than 1 and thus $((1-p)/p)^s$ decreases as s grows. It is easy to see that the larger s is, the larger the asymptotic availability is. To sum up, smaller column sizes are preferable when p<0.5 and larger column sizes are preferable when p>0.5. However, we still suggest adopting small column sizes because the asymptotic availability is high even for small column sizes when p>0.5. For example, when s=3, the asymptotic availability is 0.998630, 0.984615 and 0.927027 for p=0.9, 0.8 and 0.7, respectively. When s=4, the asymptotic availability is 0.999847, 0.996108 and 0.967365 for p=0.9, 0.8 and 0.7, respectively.

2.5.2 Quorum size

In this section, we analyze the size of quorums under $Coh(l,s)$. The smallest quorums under $Coh(l,s)$, l>>s, are of size s; such quorums are formed by including only all sites in the last cohort. However, under $Coh(l,s)$, l>>s, the largest quorums, which are composed of one site from each of $C_1,...,C_l$ (note that C_1 has only one site), is of size $l=(n-1)/s$, which is of $O(n)$.

Using the lower and the upper quorum size bounds to estimate critical section access cost may be too optimistic and too pessimistic respectively. Below, we analyze the expected quorum size as cost estimation of accessing the critical section. Let $ES(l)$ denote the expected size of the quorum under $Coh(l)$. We apply parameter f, which is also adopted in the tree quorum algorithm [AE91], to indicate the fraction of the quorums composed of only all sites in C_l (note that f is used in the tree quorum algorithm [AE91] to indicate the fraction of quorums including the root node). For $l>1$, we have

$$ES(l)=fS_l+(1-f)(1+ES(l-1))=(fS_l+1-f)+(1-f)ES(l-1) \qquad (2.3)$$

The term fS_l arises because there are f quorums of size S_l that are composed of only all sites in C_l. And the term $(1-f)(1+ES(l-1)$ arises because there are $(1-f)$ quorums of size $ES(l-1)+1$ that are composed of not all sites of C_l, but one site of C_l and one quorum under $Coh(l-1)$. Since C_1 has only one site, a quorum under $Coh(1)$ has size 1. We have $ES(1)=1$.

When $Coh(l,s)$, $l>>s$, is considered, the case of $f=1$ corresponds to the lower bound of the quorum size, which occurs when all the sites in C_l are always included in the quorum. On the other hand, the case of $f=0$ corresponds to the upper bound of the quorum size, which occurs when a larger quorum is always chosen instead of a smaller one. Note that the probability that all sites in C_l are up (i.e., p^S) can reflect the value of f. For example, the value of f can be reflected by $0.65^3=0.274625$ when $p=0.65$ and $s=3$.

Under $Coh(l,s)$ where $S_2=...=S_l=s$, the recursive equation (2.3) can be regarded as a first-order linear difference equation and can be solved analytically. For $f>0$, we have

$$ES(l)=(1-f)^{l-1}(1-(fs+1-f)/f)\ +\ (fs+1-f)/f \qquad (2.4)$$

When l goes to infinity (and so does n), the term $(1-f)^{l-1}$ goes to 0, and hence $ES(l)$ goes to $(fs+1-f)/f=s+(1/f)-1$, which is a constant. In other words, the expected size of the quorum under $Coh(l,s)$ remains constant when n grows. It is easy to see that smaller s or larger f produces smaller asymptotic expected quorum size. Take the following four cases for example: (case 1) f=0.5, s=3 (case 2) f=0.5, s=5 (case 3) f=0.25, s=3 and (case 4) f=0.25, s=5. The asymptotic expected quorum sizes for these four cases are 4, 6, 6 and 8, respectively.

2.5.3 Comparison

In this subsection, we first compare the cohort coterie (i.e., $C(l)$, the collection of all minimal quorums under $Coh(l)$) with the majority coterie [Tho79], the tree coterie [AE91], the hierarchical coterie [Kum91] and the Lovasz coterie [Nei93] in terms of quorum size and the nondominance property. Then, we further compare the availability of the cohorts coterie with those of the tree coterie and the majority coterie.

Every quorum in the majority coterie is composed of over half of the system sites; therefore, its quorum size is $\lceil (n+1)/2 \rceil$. The majority coterie is shown to be *ND* in [GB85] if the system has odd number of sites.

The tree coterie is constructed by organizing system sites into a binary tree of $\lceil \log n \rceil$ levels. Its quorum is formed by obtaining all the sites along a root-to-leaf path, and if the root fails, the obtaining should then follow two paths: one root-to-leaf path of the left subtree plus one root-to-leaf path of the right subtree. The smallest quorum comprises all the sites along a root-to-leaf path, which is of size

$\lceil \log n \rceil$, while the largest quorum comprises all leaf nodes, which is of size $\lceil (n+1)/2 \rceil$. The tree coterie is shown to be *ND* in [NM92].

By organizing sites in leaves of a mutilevel tree with non-leaf nodes being logical, quorums of $O(n^{0.63})$ size in a hierarchical coterie are formed. The quorum forming is hierarchical: a quorum corresponding to a node at level *i* is formed by collecting enough (over half) quorums corresponding to its child nodes at level *i*+1. Thus, any two quorums corresponding to the root have a non-empty intersection. The nondominance property of the hierarchical coterie, although not explicitly stated, can be inferred from some remarks (about coterie composition for hierarchical coteries) in [NM92].

The Lovasz coterie [Nei93] is based on the partition of the underlying set *U*. Let $\{P_1,...,P_k\}$ be a partition of *U* (i.e., $P_i \cap P_j = \varnothing$ for $i \neq j$ and $P_1 \cup ... \cup P_k = U$) such that $|P_i| = i$. Then a quorum in a Lovasz coterie is formed by obtaining all the sites in P_i and one site from each P_j for all $j > i$. A similar quorum forming algorithm was proposed in [SW93a]. All quorums in a Lovasz coterie are of the same $O(n^{0.5})$ size, and the Lovasz coterie has been shown to be *ND* in [Nei93] on the basis of a classical theorem, Theorem 2.1 in [GB85]. It is obvious that the list of $(P_1,...,P_k)$ is a special type of Cohorts structure; therefore, Lovasz coteries are a special type of cohort coteries.

A summary of quorum sizes of the above-mentioned coteries and the cohort coterie appears in Table 2.1. Below, we further compare the quorum availability of the majority coterie (MC) [Tho79], the tree coterie (TC) [AE91] and the cohort coterie (CC) for 15- and 31-site systems. For cohort coteries, we assume that sites are arranged as $Coh(5)=(C_1,...,C_5)$, where $|C_1|=1$, $|C_2|=...=|C_4|=3$ and $|C_5|=5$ (recall that we suggest adopting small column sizes except 2) for the 15-site system, and as *Coh*(11,3) for the 31-site system. The formulas for calculating the availabilities of MC and TC are shown below.

The availability of MC is given in [AE91] as

Prob.(h sites are up) + Prob.(h+1 sites are up) + ... + Prob.(n sites are up)

$=\sum_{i=h}^{n}\left[\mathrm{C}(n,i)\times[p^{i}\times(1-p)^{(n-i)}]\right]$, where $h=\lceil(n+1)/2\rceil$.

Assuming system sites are organized as a binary tree **T**, the availability of TC is given in [AE91] as

Availability(**T**) =
Prob.(**T**'s root is up)×Availability(**T**'s left subtree)×Unavailability(**T**'s right subtree) +
Prob.(**T**'s root is up)×Unavailability(**T**'s left subtree)×Availability(**T**'s right subtree) +
Prob.(**T**'s root is up)×Availability(**T**'s left subtree)×Availability(**T**'s right subtree) +
Prob.(**T**'s root is not up) × Availability(**T**'s left subtree) × Availability(**T**'s right subtree).

Figures 2.3 and 2.4 depicts the availability comparisons of MC, TC and CC. From these figures, we can observe that TC's availability is better (resp., worse) than MC's when up-probability is smaller (resp., larger) than 0.5. We also observe that CC's availability is very close to TC's but CC's is larger (resp., smaller) when up-probability is smaller (resp., larger) than 0.5.

2.6 Summary

In this chapter, we have devised a method to construct quorums of an *ND* coterie; the method survives network partitioning and can easily be extended to be a solution to distributed mutual exclusion. With the aid of a logical structure named *Cohorts*, the method constructs quorums of constant size in the best case. When some sites are inaccessible, the quorum size increases gradually and may be as large as O(n), where n is the number of sites. However, the expected quorum size has been shown to remain constant as n grows. This is a desirable property since the message cost to access the critical section is directly proportional to the quorum size. In addition, the availability of the constructed quorum has been shown to be asymptotically high. With the two properties—constant expected quorum size and asymptotically high availability, the proposed method is thus applicable to systems possessing an increasing number of sites. We have also analyzed and compared the constructed quorums with others in terms of availability and quorum size.

	MC	TC	HC	LC	CC
Quorum size (Lower Bound)	$\lceil (n+1)/2 \rceil$	$\lceil \log n \rceil$	$n^{0.63}$	$n^{0.5}$	Constant
Quorum size (Upper Bound)	$\lceil (n+1)/2 \rceil$	$\lceil (n+1)/2 \rceil$	$n^{0.63}$	$n^{0.5}$	O(n)

MC: The majority coterie. TC: The tree coterie. HC: The hierarchical coterie.
LC: The Lovasz coterie. CC: The cohort coterie.

Table 2.1 Bounds on quorum sizes for various coteries.

```
Function $Get\_Quorum$( $Coh(l)=(C_1,...,C_l)$: Cohorts Structure): Set;
Var $R,S,T$: Set;
If $l < 1$ Then $Exit$(failure);        // Illegal function call, claim failure //
$R = \emptyset$;        // $R$: The set of available sites that have granted permissions. //
  For ($i = l,...,1$ ) Do
    $S = C_i - R$;        // $S$: The set of the sites whose permissions are necessary to make $C_i$ the primary cohort. //
    $T = Obtain(S)$;   // $Obtain(S)$ will try to get permissions form sites of $S$ and return a set of sites that can grant permissions. //
    If $T = S$ Then $Return( Min( R \cup T, C_i,...,C_l ) )$;// $C_i$ can be the primary cohort, and a minimal quorum is returned. //
  If $(R \cup T) \cap C_i = \emptyset$ Then $Exit$(failure);        // No member of $C_i$ grants its permission. Claim failure. //
  If $(C_i \cap R) = \emptyset$ Then $R=R\cup\{t\}$, where $t \in T$.   // $R$ does not cover $C_i$. So, add $t$ of $T$ into $R$ to make $R$ covers $C_i$. //
EndFor
  $Exit$(failure);        // No quorum can be formed. Claim failure. //
End $Get\_Quorum$

Function $Min(R, C_i, ... ,C_l$ : Set): Set;
For ( $r \in R$ ) Do If $Cover(R-\{r\}, C_i,...,C_l)$ Then $R= R - \{r\}$; EndFor
// If $r$ is not essential in the coverage of $C_i,...,C_l$, remove $r$ from $R$. Note that we assume $Cover(R, C_i,...,C_l)$ is a predicate
  that returns true if $R$ covers $C_{i+1},...,C_l$ and fully covers $C_i$ for some $i$, and returns false, otherwise. //
$Return(R)$;
End $Min$
```

Figure 2.1 A function that can generate minimal quorums under $Coh(l)$.

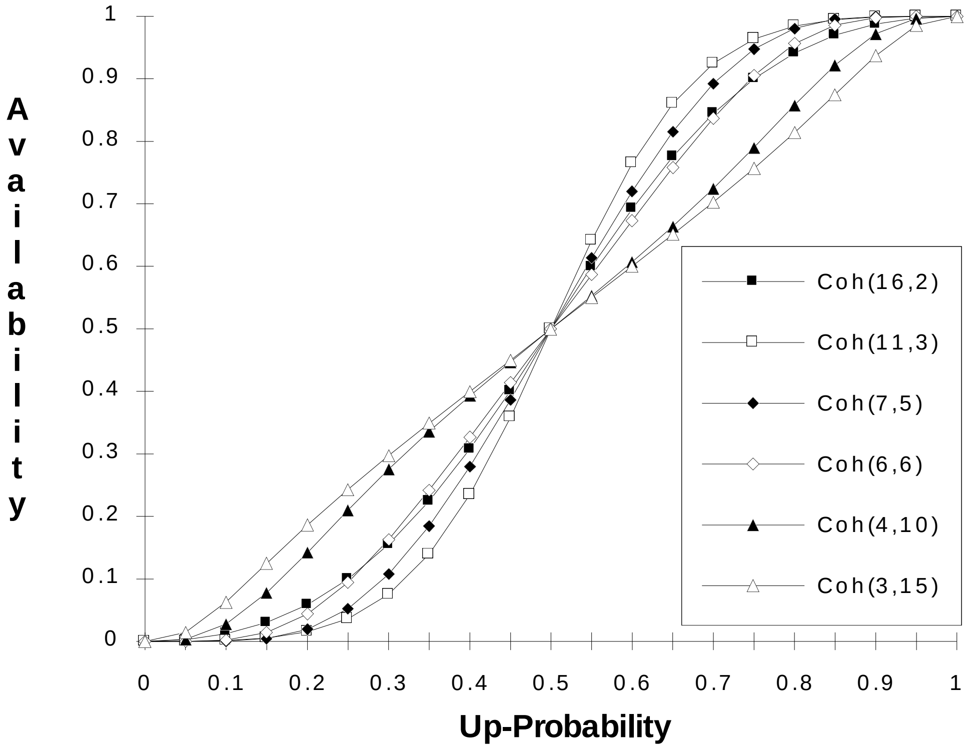

Figure 2.2 The availability of quorums under various Cohorts structures.

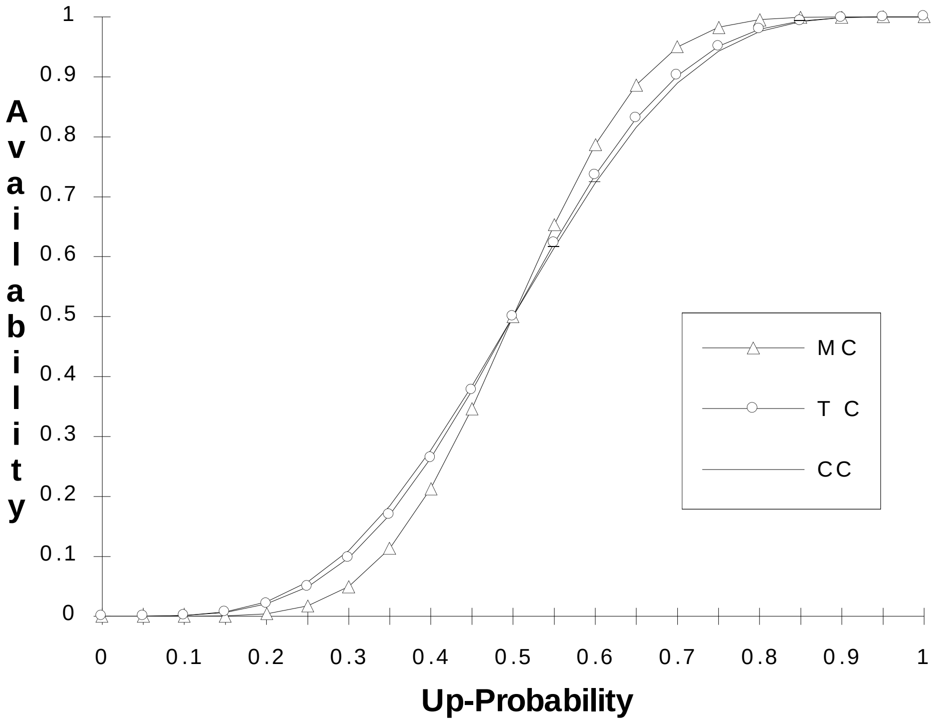

Figure 2.3 The availability comparison of various coteries for the 15-site system.

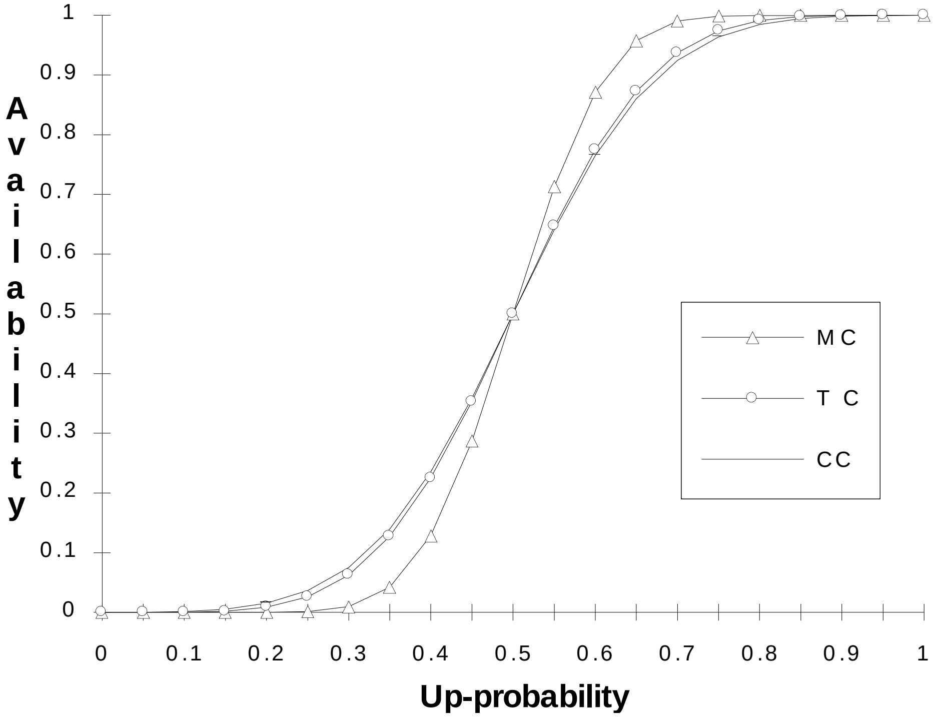

Figure 2.4 The availability comparison of various coteries for the 31-site system.

Chapter 3

Constructing *ND wr*-coteries of constant expected quorum size

1. Introduction

In a distributed system, data can be replicated at different sites to tolerate site and/or network link failures. However, complex replica control schemes are required to make multiple replicas of a data object behave as a single one, i.e., to ensure *one-copy equivalence* [BHG87]. Several replica control algorithms [AE90, AE91, BG84, CAA92, KC91, KRS93, Kum91, Nei92, Tho79] have been developed on the basis of quorum consensus concept, which is described below. Each replica is associated with a *version number*. A read operation should *read-lock* and access a read quorum of replicas and return the replica owning the largest version number. On the other hand, a write operation should *write-lock* and access a write quorum of replicas and then updates them with the new version number which is one more than the largest version number just encountered. To ensure that a read operation can always return the most up-to-date replica, any pair of a read and a write quorum and any two write quorums are required to have a non-empty intersection. The quorum-based replica control algorithms are fault-tolerant in the sense that even when network partitioning [DGS85] occurs and makes some replicas inaccessible, quorums containing only available replicas may still be found.

The *wr*-coterie concept [IK93] is usually used to formalize quorum-based replica control algorithms. A *wr-coterie* [IK93] is a pair (*W*, *R*), where *W* and *R* are families of quorums (sets) satisfying that each member of *W* or *R* has a non-empty intersection with any member of *W*. Among all the *wr*-coteries, *nondominated* (*ND*) *wr*-coteries [IK93] are preferable because they are candidates to achieve the highest availability, the probability that a quorum can be formed in an error-prone environment. Thus, we should concentrate on *ND wr*-coteries if availability is the main concern.

In this chapter, we propose a method for constructing *ND wr*-coteries; the proposed method can easily be extended for maintaining replicated data consistency. The method utilizes a logical structure named *Cohorts* to construct quorums of O(1) (constant) size in the best case. When some replicas are inaccessible, the quorum size increases gradually and may be as large as O(*n*), where *n* is the number of replicas. However, the expected quorum size is shown to remain constant as *n* grows. This is a desirable property since the message cost for accessing replicated data is directly proportional to the quorum size. In addition, the availability of the constructed quorums is shown to be asymptotically high. With the two properties—constant expected quorum size and asymptotically high availability, the proposed solution is thus applicable to systems possessing an increasing number of replicas. We also analyze and compare the constructed quorums with others in terms of availability and quorum size.

The remainder of this chapter is organized as follows. In Section 3.2, we elaborate some preliminaries of *wr*-coteries. Then, in Section 3.3, we introduce the Cohorts structure and show how to construct read quorums and write quorums with its aid. In Section 3.4, we show that the pair of collections of the constructed read quorums and write quorums is an *ND wr*-coterie. In Section 3.5, we analyze and compare the constructed quorums with others in terms of availability and quorum size. At last, we conclude this chapter with Section 3.6

3.2 Preliminaries of *wr*-coteries

In this section, we show some preliminaries of *wr*-coteries. In the following discussion, we assume $u_1,...,u_n$ are all replicas and let $U = \{u_1,...,u_n\}$ be the underlying set that contains all replicas.

A *wr*-coterie [IK93] (W, R) is a pair of two families of subsets of U satisfying

(P1) *Write-Write Intersection Property*
$\forall G,H$: $G,H\in W$: $G\cap H\neq\emptyset$.
(P2) *Write-Read Intersection Property*
$\forall G,H$: $G\in W$, $H\in R$: $G\cap H\neq\emptyset$.
(P3) *Write Quorum Minimality Property*
$\forall G,H$: $G,H\in W$: $G\not\subset H$.
(P4) *Read Quorum Minimality Property*
$\forall G,H$: $G,H\in R$: $G\not\subset H$.

For example, let $W=\{\{u_1,u_2,u_3\}, \{u_1,u_2,u_4\}, \{u_3,u_4\}\}$, $R=\{\{u_1,u_3\}, \{u_1,u_4\}, \{u_2,u_3\}, \{u_2,u_4\}, \{u_3,u_4\}\}$, then the pair (W, R) is a *wr*-coterie since it satisfies all the properties (P1), (P2), (P3), and (P4). By the write-write and write-read intersection properties, *wr*-coteries can be used to formalize replica control algorithms. Note that the minimality properties are not necessary for the correctness of replica control but can be used to enhance efficiency.

The *domination* concept for *wr*-coteries [IK93] can be used to compare two *wr*-coteries in terms of the possibility of successful quorum forming. Let (W_1,R_1) and (W_2,R_2) be two *wr*-coteries. (W_1,R_1) is said to be *dominated* by (W_2,R_2) if and only if the following three statements are all satisfied

(1) $W_1 \neq W_2$ or $R_1 \neq R_2$

(2) $\forall G$: $G \in W_1$: [$\exists H$: $H \in W_2$: $H \subseteq G$]

(3) $\forall G$: $G \in R_1$: [$\exists H$: $H \in R_2$: $H \subseteq G$].

For example, let $W_1=\{\{u_1,u_2,u_3\}, \{u_1,u_2,u_4\}, \{u_1,u_3,u_4\}, \{u_2,u_3,u_4\}\}$, $R_1=\{\{u_1,u_3\}, \{u_1,u_4\}, \{u_2,u_3\}, \{u_2,u_4\}\}$, $W_2=\{\{u_1,u_2,u_3\}, \{u_1,u_2,u_4\}, \{u_3,u_4\}\}$, and $R_2=\{\{u_1,u_3\}, \{u_1,u_4\}, \{u_2,u_3\}, \{u_2,u_4\}, \{u_3,u_4\}\}$. Then, by definition, (W_1,R_1) and (W_2,R_2) are *wr*-coteries, and (W_1,R_1) is dominated by (W_2,R_2). The dominating *wr*-coterie (W_2,R_2) has more chance than the dominated *wr*-coterie (W_1,R_1) for a quorum to be formed in an error-prone environment because if a quorum can be formed in the dominated one, then a quorum can be formed in the dominating one. Thus, we should always concentrate on *nondominated* (*ND*) *wr*-coteries [IK93] that no other *wr*-coterie can dominate, and we can claim that nondominated *wr*-coteries bias toward the highest availability.

In Ibaraki and Kameda's work [IK93], any subset of U is represented by n-tuple vector X, $X=(x_1,...,x_n) \in \{0,1\}^n$ where x_i, $1 \leq i \leq n$, is 1 (resp., 0) if u_i is in (resp., not in) the subset. Let C be a family of subsets of U. Then, Boolean function $f_C : \{0,1\}^n \rightarrow \{0,1\}$ associated with C is defined as $f_C(X) \equiv \bigvee_{Q \in C} \{ \bigwedge_{u_i \in Q} u_i \}$. Note that we follow the convention of [IK93] and also use u_i (which is an element of U) to represent the ith component of vector X. Function f_C so defined has the property: $f_C(X)=1$ if vector X represents a super set of some quorum in C; otherwise $f_C(X)=0$. The dual f^d of Boolean function f is defined as $f^d=f\,'(X'\,)$, where X' and $f\,'$ are complements of X and f, respectively. For example, under $U=\{u_1, u_2, u_3\}$, the set $\{u_1, u_2\}$ is represented as (1,1,0); $\{u_2, u_3\}$, as (0,1,1); and $\{u_1, u_2, u_3\}$, as (1,1,1). Let $C=\{\{u_1, u_2\}, \{u_2, u_3\},$

$\{u_1, u_3\}\}$, then $f_C(X) = (u_1u_2 \vee u_2u_3 \vee u_1u_3)$. $f_C^d(X) = f'(X') = (u_1' u_2' \vee u_2' u_3' \vee u_1' u_3')' = (u_1' u_2')' (u_2' u_3')' (u_1' u_3')' = (u_1 \vee u_2)(u_2 \vee u_3)(u_1 \vee u_3) = (u_1u_2 \vee u_2u_3 \vee u_1u_3)$.

The association of a Boolean function with a family of sets provides a facile way for checking some properties for the family. For example, the following Theorem 3.1 and Theorem 3.2 are actually Theorem 2.3 and Theorem 2.4 in [IK93], respectively. They are related to properties of a pair of families of subsets of *U*.

Theorem 3.1. Let *W* and *R* be families of non-empty subsets of *U* satisfying the minimality properties (P3) and (P4). Then, the pair (*W*,*R*) is a *wr*-coterie if and only if (1) $f_W \le f_W^d$ and (2) $f_W \le f_R^d$.

Theorem 3.2. Let *W* and *R* be as defined in Theorem 3.1. Then, the pair (*W*,*R*) is a nondominated *wr*-coterie if and only if (1) $f_W \le f_W^d$ and (2) $f_W = f_R^d$.

By Theorem 3.1 and Theorem 3.2, we can easily derive the following corollary that can be used to verify the nondominance of *wr*-coteries.

Corollary 3.1. Let (*W*,*R*) be a *wr*-coterie. It is nondominated if and only if $f_W = f_R^d$. □

3.3 Construction of quorums

In this section, we propose the Cohorts structure and two methods (functions *Get_Write_Quorum* and *Get_Read_Quorum* in Figure 3.1) that can generate read and write quorums with Cohort structure's help.

A ***Cohorts structure*** $Coh(l)=(C_1,...,C_l)$ is a list of pairwise disjoint sets of replicas. Each set C_i is called a *cohort* and must satisfy $|C_i|>1$ for $1 \le i \le l$ (the necessity for this restriction will be explained later). For example, $(\{u_1,u_2\}, \{u_3,u_4,u_5\}, \{u_6,u_7,u_8,u_9\})$

and $(\{u_1,u_2,u_3,u_4,u_5\}, \{u_6,u_7\}, \{u_8,u_9\})$ (with $u_1,...,u_9$ being replicas) are Cohorts structures.

By organizing data replicas as Cohorts structure $Coh(l)\equiv(C_1,...,C_l)$, we define the write and read quorums as follows:

A ***write quorum*** under $Coh(l)$ is a set that contains all replicas of some cohort C_i, $\mathbf{1\leq i\leq l}$ (note that i=1 is included), and one replica of each of the cohorts $C_{i+1},...,C_l$.

A ***read quorum*** under $Coh(l)$ is either
Type-1: a set that contains one replica of each of the cohorts $C_1,...,C_l$.
or
Type-2: a set that contains all replicas of some cohort C_i, $\mathbf{1<i\leq l}$ (note that i=1 is excluded), and one replica of each of the cohorts $C_{i+1},...,C_l$.

For example, under $Coh(2)=(\{u_1,u_2,u_3\}, \{u_4,u_5\})$, the possible write quorums are $\{u_4,u_5\}$, $\{u_1,u_2,u_3,u_4\}$, $\{u_1,u_2,u_3,u_5\}$, and the possible read quorums are $\{u_1,u_4\}$, $\{u_1,u_5\}$, $\{u_2,u_4\}$, $\{u_2,u_5\}$, $\{u_3,u_4\}$, $\{u_3,u_5\}$ (of type-1) and $\{u_4,u_5\}$ (of type-2). Note that the write quorum definition and the type-2 read quorum definition are identical except that the latter does not include the sets composed of all replicas in C_1 and one replica from each of $C_2,...,C_l$. For the sake of efficiency, the sets just mentioned are not regarded as read quorums because each of them is a super set of a type-1 read quorum.

In an extreme case, only all replicas in C_l can constitute a quorum that is of a constant size $|C_l|$. And in another extreme case, one replica from each of $C_1,...,C_l$ (for a type-1 read quorum) or all replicas in C_1 together with one replica from each of $C_2,...,C_l$ (for a write quorum) can constitute a quorum. If the size of each C_i, $1\leq i\leq l$, is constant (or bounded above and below by a constant), then the quorum mentioned is of size $O(n)$.

Two functions, *Get_Write_Quorum* and *Get_Read_Quorum*, which can respectively produce read quorums and write quorums under $Coh(l)$ are shown in Figure 3.1. Note that we assume $wlock(C_i)$ is a function that tries to write-lock and return replicas of C_i. It locks and returns (case 1) the set of all replicas of C_i if they are all lockable, or (case 2) a singleton set of one arbitrary lockable replica if more than one replica is lockable, or (case 3) an empty set, otherwise. Note that when $wlock(C_1)$ (i=1) is performed, (case 2) is ruled out, i.e., either the set of all replicas of C_1 or an empty set is returned. Function $rlock(C_i)$ is identical to $wlock(C_i)$ except that $rlock(C_i)$ uses read-lock instead of write-lock and that when $rlock(C_1)$ is performed, (case 1) is ruled out, i.e., either a singleton set of one lockable replica of C_1 or an empty set is returned.

3.4 Correctness

Let $W(l)$ denote the collection of all write quorums under $Coh(l)$, and $R(l)$, the collection of all read quorums. Below, we prove that the pair $(W(l), R(l))$, $l \geq 1$, is an *ND wr*-coterie. Note that later we call $(W(l), R(l))$ *cohort coterie*.

Theorem 3.3. The pair $(W(l), R(l))$ is a *wr*-coterie for $l \geq 1$.

Proof: The proof is by induction on the value of l.

Basis (l=1):

Consider $Coh(1) \equiv (\{u_1,...,u_m\})$, where $m>1$. Then, under $Coh(1)$, the only write quorum is $\{u_1,...,u_m\}$ and the read quorums are $\{u_1\},...,\{u_m\}$. It is obvious that these quorums satisfy all of (P1), (P2), (P3) and (P4). Therefore, $(W(1), R(1))$ is a *wr*-coterie, and the theorem holds for the basis case.

Induction Hypothesis:

We assume that ($W(l)$, $R(l)$) is a *wr*-coterie satisfying (P1), (P2), (P3) and (P4) for some l, $l \geq 1$.

Induction Step:

Consider $Coh(l+1) \equiv (C_1,...,C_{l+1})$. Let $C_{l+1}=\{u_1,...,u_m\}$, where $m>1$ (note that m should be larger than one according to the Cohorts structure definition in Section 2). Then, a write quorum of $W(l+1)$ may be of the form: either (form-1) $\{u_1,...,u_m\}$ or (form-2) $\{u_i\} \cup$any quorum of $W(l)$ for $1 \leq i \leq m$. A read quorum of $R(l+1)$ may be of the form: either (form-1) $\{u_1,...,u_m\}$ or (form-2) $\{u_i\} \cup$any quorum of $R(l)$ for $1 \leq i \leq m$. Below, we show that ($W(l+1)$, $R(l+1)$) satisfies (P1), (P2), (P3) and (P4) on the basis of induction hypothesis.

Satisfaction of (P1): The form-1 write quorum overlaps any form-2 write quorum since $\{u_1,...,u_m\} \cap \{u_i\} = \{u_i\} \neq \emptyset$. Two form-2 write quorums overlap each other since by hypothesis any two quorums of W(l) overlap each other. (And trivially, the form-1 write quorum overlaps itself.)

Satisfaction of (P2): The form-1 write quorum overlaps any form-2 read quorum since $\{u_1,...,u_m\} \cap \{u_i\} = \{u_i\} \neq \emptyset$. In the same way, the form-1 read quorum overlaps any form-2 write quorum. Any form-2 write quorum overlaps any form-2 read quorum since by hypothesis any quorum of W(l) overlaps any quorum of R(l). And obviously, the form-1 write quorum overlaps the form-1 read quorum.

Satisfaction of (P3): The form-1 write quorum is not a proper subset of any form-2 write quorum since $\{u_1,...,u_m\}$ is not a proper subset of any quorum in W(l) (by $C_1,...,C_{l+1}$ being pairwise disjoint) and $\{u_1,...,u_m\} \not\subset \{u_i\}$ (by $m>1$). Any form-2 write quorum is not a proper subset of the form-1 write quorum since any quorum in W(l) is not a proper subset of $\{u_1,...,u_m\}$ (by $C_1,...,C_{l+1}$ being pairwise disjoint). Any form-2 write quorum is not a proper subset of any form-2 write quorum since by hypothesis any quorum in W(l) is not a proper subset of any quorum in W(l). (And trivially, the form-1 write quorum is not a proper subset of itself.)

Satisfaction of (P4): The proof is similar to that provided in **Satisfaction of (P3)** and is omitted.

By now, on the basis of induction hypothesis, we have shown that ($W(l+1)$, $R(l+1)$) satisfies (P1), (P2), (P3) and (P4), which means ($W(l+1)$, $R(l+1)$) is a *wr*-coterie.

Thus, by the induction principle, ($W(l)$, $R(l)$) is a *wr*-coterie for arbitrary l, $l \geq 1$. □

Now, the reason why we restrict each cohort of a Cohorts structure to contain more than one replica is clear—it is for the correctness of the minimality properties (P3) and (P4). Note that the first cohort (C_1) having only one replica will not violate the minimality properties but will make the set of all write quorums and the set of all read quorums identical. We prefer to have different sets of read and write quorums so that we can treat read and write operations differently when facing practical systems where the numbers of read and write operations are usually quite different. Therefore, we still limit C_1 to contain more than one replica.

By now, we have proved that ($W(l)$,$R(l)$) is a *wr*-coterie. Below, we further prove that ($W(l)$,$R(l)$) is nondominated. We start by showing the relation between $f_{W(l)}$ and $f^d_{R(l)}$ with the aid of Boolean algebra laws [Lip79].

<u>Lemma 3.1</u>. $f_{W(l)} = f^d_{R(l)}$ for $l \geq 1$.

Proof: The proof is by induction on the value of l.

Basis (l=1):

Consider $Coh(1)\equiv(\{u_1,...,u_m\})$, where $m>1$. Under $Coh(1)$, the only write quorum is $\{u_1,...,u_m\}$ and the read quorums are $\{u_1\},...,\{u_m\}$. We have $f_{W(1)}=(u_1\wedge...\wedge u_m)$ and $f_{R(1)}=(u_1\vee...\vee u_m)$. Since $f^d_{R(1)}=(u_1'\vee...\vee u_m')'=(u_1')'\wedge...\wedge(u_m')'=(u_1\wedge...\wedge u_m)=f_{W(1)}$ (by De Morgan's law and $(u_i')'=u_i$), the lemma holds for the basis case.

Induction Hypothesis:

We assume that $f_{W(l)}=f^d_{R(l)}$ for some l, $l\geq 1$.

Induction Step:

Consider $Coh(l+1)\equiv(C_1,...,C_{l+1})$. Let $C_{l+1}=\{u_1,...,u_m\}$, where $m>1$. Then, a quorum of $W(l+1)$ may be of the form: either (1) $\{u_1,...,u_m\}$ or (2) $\{u_i\}\cup$ any quorum of $W(l)$, for $1\leq i\leq m$. A quorum of $R(l+1)$ may be of the form: either (1) $\{u_1,...,u_m\}$ or (2) $\{u_i\}$ $\cup$ any quorum of $R(l)$, for $1\leq i\leq m$. Thus, we have $f_{\mathrm{W}(l+1)}=(\bigwedge_{i=1,...,m}u_i)\vee(\bigvee_{i=1,...,m}u_i\, f_{\mathrm{W}(l)}(X))$ and $f_{\mathrm{R}(l+1)}=(\bigwedge_{i=1,...,m}u_i)\vee(\bigvee_{i=1,...,m}u_i\, f_{\mathrm{R}(l)}(X))$, where X is a vector that can represent subsets of $C_1\cup...\cup C_l$. Note that below $f_{\mathrm{W}(l)}(X)$ and $f_{\mathrm{R}(l)}(X)$ are occasionally abbreviated as $f_{\mathrm{W}(l)}$ and $f_{\mathrm{R}(l)}$, respectively.

Below, we show that $f^d_{R(l+1)}=f_{\mathrm{W}(l+1)}$ on the basis of induction hypothesis. We have

$f_{\mathrm{R}(l+1)}=(\bigwedge_{i=1,...,m}u_i)\vee((\bigvee_{i=1,...,m}u_i)\wedge f_{\mathrm{R}(l)})$ (by distributive law)

$=(\bigwedge_{i=1,...,m}u_i)\vee(f_{\mathrm{R}(l)}\wedge(\bigvee_{i=1,...,m}u_i))$ (by commutative law).

Therefore,

$f^d_{R(l+1)}=((\bigwedge_{i=1,...,m}u_i')\vee(f_{\mathrm{R}(l)}(X')\wedge(\bigvee_{i=1,...,m}u_i')))'$ (by definition of $f^d_{R(l+1)}$)

$=(\bigwedge_{i=1,...,m}u_i')'\wedge(f_{\mathrm{R}(l)}(X')\wedge(\bigvee_{i=1,...,m}u_i'))'$ (by De Morgan's law)

$=(\bigwedge_{i=1,...,m}u_i')'\wedge(f'_{R(l)}(X')\vee(\bigvee_{i=1,...,m}u_i')')$ (by De Morgan's law)

$=(\bigvee_{i=1,...,m}(u_i')')\wedge(f'_{R(l)}(X')\vee(\bigwedge_{i=1,...,m}(u_i')'))$ (by De Morgan's law)

$=(\bigvee_{i=1,...,m}u_i)\wedge(f'_{R(l)}(X')\vee(\bigwedge_{i=1,...,m}u_i))$ (by involution law, i.e., $(u_i')'=u_i$)

$$\begin{aligned}
&= (\bigvee_{i=1,\ldots,m} u_i) \wedge (f^d_{R(l)} \vee (\bigwedge_{i=1,\ldots,m} u_i)) && \text{(since } f^d_{R(l)} = f'_{R(l)}(X') \text{ by definition)}\\
&= (\bigvee_{i=1,\ldots,m} u_i) \wedge (f_{W(l)} \vee (\bigwedge_{i=1,\ldots,m} u_i)) && \text{(since } f_{W(l)} = f^d_{R(l)} \text{ by hypothesis)}\\
&= ((\bigvee_{i=1,\ldots,m} u_i) \wedge f_{W(l)}) \vee ((\bigvee_{i=1,\ldots,m} u_i) \wedge (\bigwedge_{i=1,\ldots,m} u_i)) && \text{(by distributive law)}\\
&= ((\bigvee_{i=1,\ldots,m} u_i) \wedge f_{W(l)}) \vee (\bigwedge_{i=1,\ldots,m} u_i) && \text{(since } (\bigvee_{i=1,\ldots,m} u_i) \wedge (\bigwedge_{i=1,\ldots,m} u_i) = (\bigwedge_{i=1,\ldots,m} u_i) \text{)}\\
&= (\bigwedge_{i=1,\ldots,m} u_i) \vee ((\bigvee_{i=1,\ldots,m} u_i) \wedge f_{W(l)}) && \text{(by commutative law)}\\
&= (\bigwedge_{i=1,\ldots,m} u_i) \vee (\bigvee_{i=1,\ldots,m} u_i\, f_{W(l)}) && \text{(by distributive law)}
\end{aligned}$$

$=f_{W(l+1)}$

Therefore, by the induction principle, we have $f_{W(k)} = f^d_{R(k)}$ for any l, $l \geq 1$. □

Theorem 3.4. ($W(l)$,$R(l)$) is a nondominated *wr*-coterie for any l, $l \geq 1$.
Proof: This is a direct consequence of Theorem 3.3, Lemma 3.1, and Corollary 3.1. □

3.5 Analysis and comparison

In this section we analyze and compare the quorums under *Coh*(l) with some other types of quorums in terms of availability and quorum size. Below, we assume that all data replicas have the same *up-probability p*, the probability that a single replica is up (i.e., accessible). We also use S_i to denote $|C_i|$ for $1 \leq i \leq l$, where C_i is the ith cohort of *Coh*(l)=(C_1,...,C_l).

3.5.1 Availability

The read (resp., write) availability is defined to be the probability of a read (resp., write) quorum being successfully formed in an error-prone environment. For $l>1$, if all replicas in C_l are up, then a read (or write) quorum under *Coh*(l) can be formed. On the other hand, if at least one replica but not all replicas in C_l are up, then one of the up replicas together with a read (resp., write) quorum under *Coh*(l–1) can form a read (resp., write) quorum under *Coh*(l). Let $AV_R(l)$ denote the availability of read quorums under $Coh(l)$, and $AV_W(l)$, the availability of write quorums under $Coh(l)$. For $l>1$, we have

$$\begin{aligned}
AV_R(l) \;&=\; Prob.(\text{all replicas in } C_l \text{ are up}) + \\
&\quad\; Prob.(\text{at least one replica but not all replicas in } C_l \text{ are up}) \times AV_R(l-1) \\
&=\; p^{S_l} + (1- p^{S_l} - (1-p)^{S_l})AV_R(l-1) \qquad (3.1)
\end{aligned}$$

$$\begin{aligned}
AV_W(l) \;&=\; Prob.(\text{all replicas in } C_l \text{ are up}) + \\
&\quad\; Prob.(\text{at least one replica but not all replicas in } C_l \text{ are up}) \times AV_W(l-1) \\
&=\; p^{S_l} + (1- p^{S_l} - (1-p)^{S_l})AV_W(l-1) \qquad (3.2)
\end{aligned}$$

For $l=1$, if at least one replica in C_1 is up, then a read quorum under $Coh(1)$ can be formed. And all replicas in C_1 are required to be up to form a write quorum under $Coh(1)$. Thus, we have $AV_R(1)=(1-(1-p)^{S_1})$ and $AV_W(1)=p^{S_1}$.

A fixed number of replicas can be arranged as a variety of Cohorts structures. To reduce the number of analysis cases, we limit all cohorts to have the same size s; that is, we assume $|C_1|=...=|C_l|=s$ (i.e., $S_1=...=S_l=s$) for $Coh(l)=(C_1,...,C_l)$. Below, we use $Coh(l,s)$ to denote such a structure.

When $Coh(l,s)$ is considered, the recursive equations (3.1) and (3.2) can be regarded as first-order linear difference equations [DOSE86]*, which can be solved analytically. We have

$$AV_R(l)=(1-p^s-(1-p)^s)^{l-1}(1-(1-p)^s-p^s/(p^s+(1-p)^s)) \;+\; (p^s/(p^s+(1-p)^s)) \qquad (3.3)$$

$$AV_W(l)=(1-p^s-(1-p)^s)^{l-1}(p^s-p^s/(p^s+(1-p)^s)) \;+\; (p^s/(p^s+(1-p)^s)) \qquad (3.4)$$

We first apply equations (3.3) and (3.4) to investigate the influence of cohort sizes under a fixed number of replicas. We assume the following Cohorts structures for a 30-replica system: $Coh(15,2)$, $Coh(10,3)$, $Coh(6,5)$, $Coh(5,6)$, $Coh(3,10)$ and $Coh(2,15)$. The read availabilities corresponding to those structures are depicted in

* A first-order linear difference equation of the form $X_k=aX_{k-1}+b$ for $k\geq 2$ with X_1 being the first term has as its kth term $X_k=a^{k-1}(X_1+b/(a-1))-(b/(a-1))$ if $a\neq 1$.

Figure 3.2, which reveals that larger cohort sizes usually render the read availability higher (because they make the construction of type-1 read quorums easier). The write availabilities corresponding to those structures are depicted in Figure 3.3, which reveals that smaller cohort sizes usually render the write availability higher (because they make the construction of write quorums easier).

There are trade-offs between the read availability and the write availability. However, one can choose a proper cohort size according to practical situations, such as the fractions of read and write operations, and the constraints on the lowest read or write availabilities, etc. Since the read availabilities are on the upper side and the write availabilities are on the lower side, we suggest adopting small cohort sizes, say 3 or 5, so that both the read and write availabilities are comparably high. We do not suggest adopting the cohort size of 2, which leads to lower write availabilities than those resulting from sizes of 3 and 5 for large up-probability p (e.g., for p>0.75). Note that most practical systems have large up-probability p, under which the cohort size of 2 causes a relatively large probability of no replica in a cohort being up, which prohibits the construction of any quorum.

We now apply equations (3.3) and (3.4) to investigate the asymptotic value of quorum availability. When l goes to infinity, the term $(1-p^s-(1-p)^s)^{l-1}$ goes to 0, and both $AV_R(l)$ and $AV_W(l)$ go to $p^s/(p^s+(1-p)^s)=1/(1+((1-p)/p)^s)$. In other words, the asymptotic availability of the quorums under $Coh(l)$ is $1/(1+((1-p)/p)^s)$. For p=0.5, the asymptotic availability is 0.5 whatever the cohort size is. For p<0.5, $(1-p)/p$ is larger than 1 and thus $((1-p)/p)^s$ increases as s grows. It is easy to see that the smaller s is, the larger the asymptotic availability is. For p>0.5, $(1-p)/p$ is less than 1 and thus $((1-p)/p)^s$ decreases as s grows. It is easy to see that the larger s is, the larger the asymptotic availability is. To sum up, smaller cohort sizes are preferable when p<0.5 and larger cohort sizes are preferable when p>0.5. However, we still suggest adopting small cohort sizes because the asymptotic availability is high even for small cohort sizes when p>0.5. For example, when s=3, the asymptotic availability is 0.998630, 0.984615 and 0.927027 for p=0.9, 0.8 and 0.7, respectively. When s=4, the asymptotic availability is 0.999847, 0.996108 and 0.967365 for p=0.9, 0.8 and 0.7, respectively.

3.5.2 Quorum size

In this section, we analyze the size of quorums under *Coh*(l,s). Both the smallest read and write quorums under *Coh*(l,s), l>>s, are of size s; such quorums are formed by including only all replicas in the last cohort. This is a desirable property since the message cost for accessing replicated data is directly proportional to the quorum size. However, the size of the largest quorums under *Coh*(l,s), l>>s, are of size O(n). The largest read quorum, which is composed of one replica from each of C_1,...,C_l, is of size l=n/s. And the largest write quorum, which is composed of all replicas of C_1 and one replica from each of C_2,...,C_l, is of size s+l–1=s+n/s–1.

Using the lower and the upper quorum size bounds to estimate data access cost may be too optimistic and too pessimistic respectively. Below, we analyze the expected quorum size as estimation of average cost for accessing replicated data. Let $ES_R(l)$ and $ES_W(l)$ denote respectively the expected sizes of read and write quorums under *Coh*(l). We apply parameter f, which is also adopted in the tree quorum algorithm [AE91], to indicate the fraction of the quorums composed of only all replicas in C_l (note that f is used in the tree quorum algorithm [AE91] to indicate the fraction of quorums including the root node). For l>1, we have

$ES_R(l)=fS_l+(1-f)(1+ES_R(l-1))=(fS_l+1-f)+(1-f)ES_R(l-1)$ (3.5)

$ES_W(l)=fS_l+(1-f)(1+ES_W(l-1))=(fS_l+1-f)+(1-f)ES_W(l-1)$ (3.6)

The term fS_l arises because there are f quorums of size S_l that are composed of only all replicas in C_l. And the term $(1-f)(1+ES_R(l-1)$ (resp., $(1-f)(1+ES_W(l-1)$) arises because there are $(1-f)$ quorums of size $ES_R(l-1)+1$ (resp., $ES_W(l-1)+1$) that are composed of not all replicas of C_l, but one replica of C_l and one quorum under $Coh(l-1)$. Since one arbitrary replica of C_1 can form a read quorum under $Coh(1)$, and all replicas in C_1 can form a write quorum under $Coh(1)$, we have $ES_R(1)=1$ and $ES_W(1)=S_1$.

When $Coh(l,s)$, $l>>s$, is considered, the case of $f=1$ corresponds to the lower bound of the quorum size, which occurs when all the replicas in C_l are always included in the quorum. On the other hand, the case of $f=0$ corresponds to the upper bound of the quorum size, which occurs when a larger quorum is always chosen instead of a smaller one. Note that the probability that all replicas in C_l are up (i.e., p^S) can reflect the value of f. For example, the value of f can be reflected by $0.65^3=0.274625$ when $p=0.65$ and $s=3$.

Under $Coh(l,s)$ where $S_1=...=S_l=s$, the recursive equations (3.5) and (3.6) can be regarded as first-order linear difference equations and can be solved analytically. For $f>0$, we have

$ES_R(l)=(1-f)^{l-1}(1-(fs+1-f)/f) \ + \ (fs+1-f)/f$ (3.7)

$ES_W(l)=(1-f)^{l-1}(s-(fs+1-f)/f) \ + \ (fs+1-f)/f$ (3.8)

When l goes to infinity (and so does n), the term $(1-f)^{l-1}$ goes to 0, and hence both $ES_R(l)$ and $ES_W(l)$ go to $(fs+1-f)/f=s+(1/f)-1$, which is a constant. In other words, the expected size of the quorum under $Coh(l,s)$ remains constant when n grows. It is easy to see that smaller s or larger f produces smaller asymptotic expected quorum size. Take the following four cases for example: (case 1) f=0.5, s=3 (case 2) f=0.5, s=5 (case 3) f=0.25, s=3 and (case 4) f=0.25, s=5. The asymptotic expected quorum sizes for these four cases are 4, 6, 6 and 8, respectively.

3.5.3 Comparison

In this section we first describe some related algorithms [AE91, BG84, CAA92, Kum91, KRS93, KC91, Nei92, Tho79] that generate quorums of a *wr*-coterie. We then compare the cohort coterie (CC) with the *wr*-coteries corresponding to these algorithms in terms of quorum size, quorum availability and the nondominance property.

The simplest replica control scheme is the read-one-write-all algorithm (ROWA) [BG84], in which any replica can form a read quorum and all the replicas can form a write quorum. ROWA can be regarded as a special case of our proposed method—when the Cohorts structure with only one cohort containing all the replicas is applied. The *wr*-coterie corresponding to ROWA (referred to as ROWAC later) is thus *ND*. The majority quorum algorithm [Tho79] requires both the read and the write quorums to have over half (i.e., at least $\lceil (n+1)/2 \rceil$) replicas; thus, its quorum size is O(n). The *wr*-coterie corresponding to the majority quorum algorithm (denoted as MC) has been shown to be *ND* if n is odd [GB85]

Some algorithms [AE91, Kum91] form quorums with the aid of tree structures. By placing replicas in leaves of a mutilevel tree with non-leaf nodes being logical, the hierarchical quorum algorithm [Kum91] achieves O($n^{0.63}$) quorum size. Its quorum forming is hierarchical: a quorum of a node at level i is formed if enough (over half) quorums of its child nodes at level i+1 are formed. Thus, any two quorums formed at the root have a non-empty intersection and can be used as a write (or read) quorum. Although not explicitly stated, the *wr*-coterie corresponding to the hierarchical quorum algorithm (referred to as HC later) can be proved to be *ND* with some remarks in [NM92]. It is *ND* if each non-leaf node has odd number of child nodes in the multilevel tree.

Assuming replicas are logically organized as a binary tree, the tree quorum algorithm [AE91] has $\lceil \log n \rceil$ quorum size in the best case. Its quorum forming (for both read and write quorums) is recursive and can be regarded as attempting to obtain replicas from nodes along a root-to-leaf path. If the root fails, then the obtaining should follow two paths: one root-to-leaf path on the left subtree and one root-to-leaf path on the right subtree. The largest quorum is composed of all leaf nodes and is of size $\lceil (n+1)/2 \rceil$; however, it has been shown in [AE91] that the tree quorum algorithm has O(log n) quorum size for most practical environments. The *wr*-coterie corresponding to the tree quorum algorithm (referred to as TC later) has been shown to be *ND* in [MN92].

In the grid algorithm [CAA92], replicas are organized as a rectangular grid of l rows and m columns, where $l \times m = n$ (the number of replicas). A *column-cover,* which contains one replica of each column, can form a read quorum, and a column-cover along with all replicas of some column can form a write quorum. Thus, a read quorum contains m replicas and a write quorum contains $l+m-1$ replicas. If a square grid is assumed, i.e., $l=m=\sqrt{n}$, then both the read and write quorums have O($\sqrt{n}$) size.

The *wr*-coterie corresponding to the grid algorithm (referred to as GC later) is dominated by CC (the cohort coterie), which means that if a quorum can be formed in GC then a quorum can be formed in CC. Below, we verify the last statement. Consider the Cohorts structure *Coh*(*l*,*s*), which is exactly a *l*-column, *s*-row grid structure. Under such a structure, a write quorum of GC is a super set of some write quorum under *Coh*(*l*,*s*) (by the definitions of the two quorums discussed), and a read quorum of GC is actually a type-1 read quorum under *Coh*(*l*,*s*) (and CC still has type-2 read quorums). Therefore, we can conclude that GC is dominated by CC.

In the hierarchical grid algorithm [KC91], a hierarchical grid structure is used in which nodes at the lowest level 0 are physical replicas and nodes at level i (i>0) are defined as a square grid of level i–1 nodes. The quorum forming is recursive and is identical to that of the grid algorithm if viewed at a single level. The read (resp., write) quorum formed at the top level allows a read (resp., write) operation to proceed. If a square grid structure is assumed in each level, the hierarchical grid algorithm also owns O($\sqrt{n}$) quorum size for both write and read quorums. The hierarchical grid algorithm has the property that its quorum availability increases asymptotically when more replicas are used, a property not owned by grid algorithm. The *wr*-coterie corresponding to the hierarchical grid algorithm (referred to as HGC later) is dominated since GC is dominated.

The general grid algorithm [KRS93] improves the grid algorithm by regarding either a column-cover or a full column of replicas as a read quorum (this improvement was also suggested independently in [Nei92]) and by allowing the existence of empty (hollow) grid positions that correspond to no data replica. It has been shown in [KRS93] that empty grid positions usually make quorum availability higher. The *wr*-coterie corresponding to the general grid algorithm (referred to as GGC later) has been shown to be *ND* in [KRS92]. GGC has the same write quorum size as GC and any GGC's write quorum is a super set of some CC's write quorum. However, any CC's read quorum is a super set of some GGC's read quorum.

A summary of quorum sizes of some of the discussed *wr*-coteries appears in Table 3.1. Availability comparisons of CC, ROWAC, MC and TC for 15- and 31-replica systems appear in Figures 3.4 and 3.5. When CC is concerned, we assume that replicas are arranged as *Coh*(5,3) in the 15-replica system, and as *Coh*(10)=(C_1,...,C_{10}), where $|C_1|$=...=$|C_9|$=3 and $|C_{10}|$=4, in the 31-replica system (recall that we suggest adopting small cohort sizes except 2). The formulas for calculating the availabilities of ROWAC, MC and TC are discussed below.

It is easy to see that ROWAC's read and write availabilities are $1-(1-p)^n$ and p^n, respectively. MC does not differentiate read quorums from write quorums. Its availability is given in [AE91] as

Prob.(h replicas are up) + Prob.(h+1 replicas are up) + ... + Prob.(n replicas are up) $=\sum_{i=h}^{n}\left[\mathrm{C}(n,i)\times[p^i\times(1-p)^{(n-i)}]\right]$, where $h=\lceil(n+1)/2\rceil$.

Assuming data replicas are organized as a binary tree **T**, TC's availability is given in [AE91] as

Availability(**T**) =
Prob.(**T**'s root is up)×Availability(**T**'s left subtree)×Unavailability(**T**'s right subtree)
+
Prob.(**T**'s root is up)×Unavailability(**T**'s left subtree)×Availability(**T**'s right subtree)
+
Prob.(**T**'s root is up) × Availability(**T**'s left subtree) × Availability(**T**'s right subtree)
+
Prob.(**T**'s root is not up) × Availability(**T**'s left subtree) × Availability(**T**'s right subtree).

Figures 3.4 and 3.5 reveal that the read availability and the write availability of ROWAC are almost bounds of those of other *wr*-coteries. The availability of TC is better (resp., worse) than that of MC when up-probability is smaller (resp., larger) than 0.5. For a wide range of up-probabilities, the read (resp., write) availability of CC is a little better (resp. worse) than the availability of TC.

3.6 Summary

In this chapter, we have devised a method to construct quorums of an *ND wr*-coterie; the method survives network partitioning and can easily be extended to maintain replicated data consistency. With the aid of a logical structure named *Cohorts*, the method constructs quorums of constant size in the best case. When some replicas are inaccessible, the quorum size increases gradually and may be as large as O(*n*), where *n* is the number of replicas. However, the expected quorum size has been shown to remain constant as *n* grows. This is a desirable property since the message cost for accessing the replicated data is directly proportional to the quorum size. In addition, the availability of the constructed quorum has been shown to be asymptotically high. With the two properties—constant expected quorum size and asymptotically high availability, the proposed method is thus applicable to systems possessing an increasing number of replicas. We have also analyzed and compared the constructed quorums with others in terms of availability and quorum size.

	MC	HC	TC	GC	HGC	CC
Lower Bound	$\lceil (n+1)/2 \rceil$	$O(n^{0.63})$	$\lceil \log n \rceil$	$O(n^{0.5})$	$O(n^{0.5})$	s
Upper Bound	$\lceil (n+1)/2 \rceil$	$O(n^{0.63})$	$\lceil (n+1)/2 \rceil$	$O(n^{0.5})$	$O(n^{0.5})$	$O(n)$

MC: The *wr*-coterie corresponding to the majority quorum algorithm [Tho79].
HC: The *wr*-coterie corresponding to the hierarchical quorum algorithm [Kum91].
TC: The *wr*-coterie corresponding to the tree quorum algorithm [AE91].
GC: The *wr*-coterie corresponding to the grid algorithm [CAA92].
HGC: The *wr*-coterie corresponding to the hierarchical grid algorithm [KC91].
CC: The cohort coterie (under *Coh*(l,s), where l>>s).

Table 3.1 Bounds on quorum sizes for various *wr*-coteries.

```
Function Get_Write_Quorum( $Coh(l)=(C_1,...,C_l)$: Cohorts Structure): Set;
Var $S$: Set;
  If $l < 1$ Then Exit(failure);              // Illegal function call, claim failure //
  $S = wlock(C_l)$;
  If $S = C_l$ Then Return($S$);
  If $|S| = 1$ Then Return($S \cup$ Get_Write_Quorum($Coh(l-1)=(C_1,...,C_{l-1})$));
  If $S = \emptyset$ Then Exit(failure);           // Unable to form a write quorum, claim failure //
End Get_Write_Quorum

Function Get_Read_Quorum( $Coh(l)=(C_1,...,C_l)$: Cohorts): Set;
Var $S$: Set;
  If $l < 1$ Then Exit(failure);              // Illegal function call, claim failure //
  $S = rlock(C_l)$;
  If $S = C_l$ Then Return($S$);
  If $|S| = 1$ and $l > 1$ Then Return($S \cup$ Get_Read_Quorum($Coh(l-1)=(C_1,...,C_{l-1})$));
  If $|S| = 1$ and $l = 1$ Then Return($S$);
  If $S = \emptyset$ Then Exit(failure);           // Unable to form a read quorum, claim failure //
End Get_Read_Quorum
```

Figure 3.1 Functions that can generate read and write quorums under *Coh*(*l*).

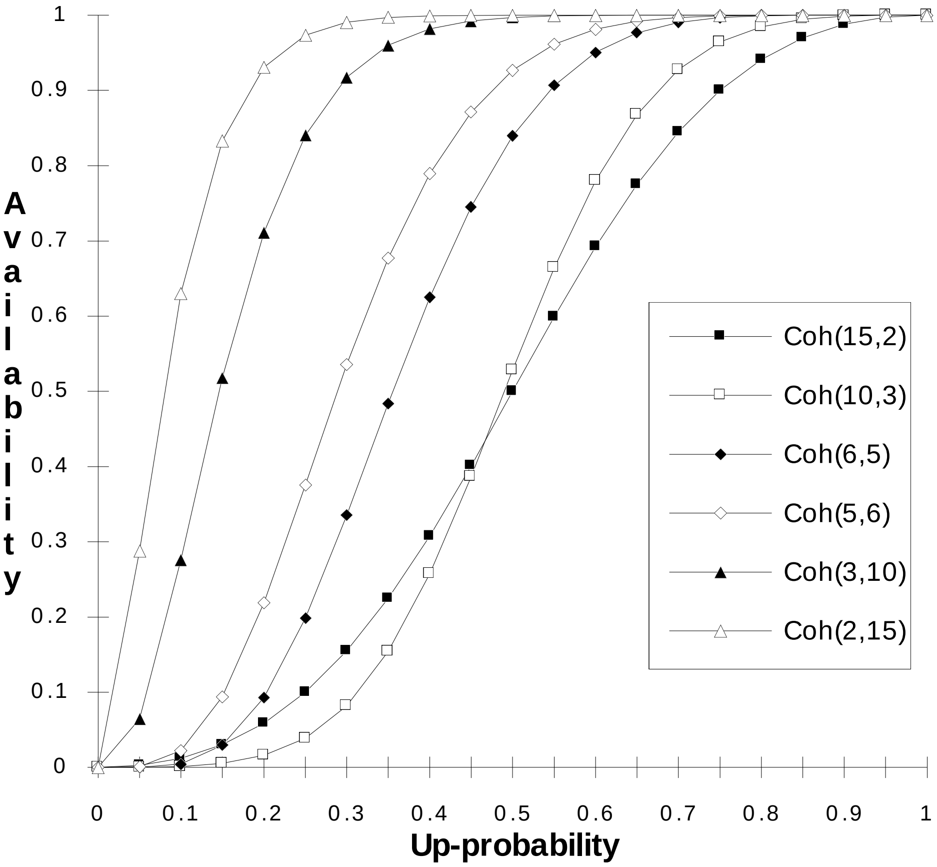

Figure 3.2 The availability of read quorums under various Cohorts structures.

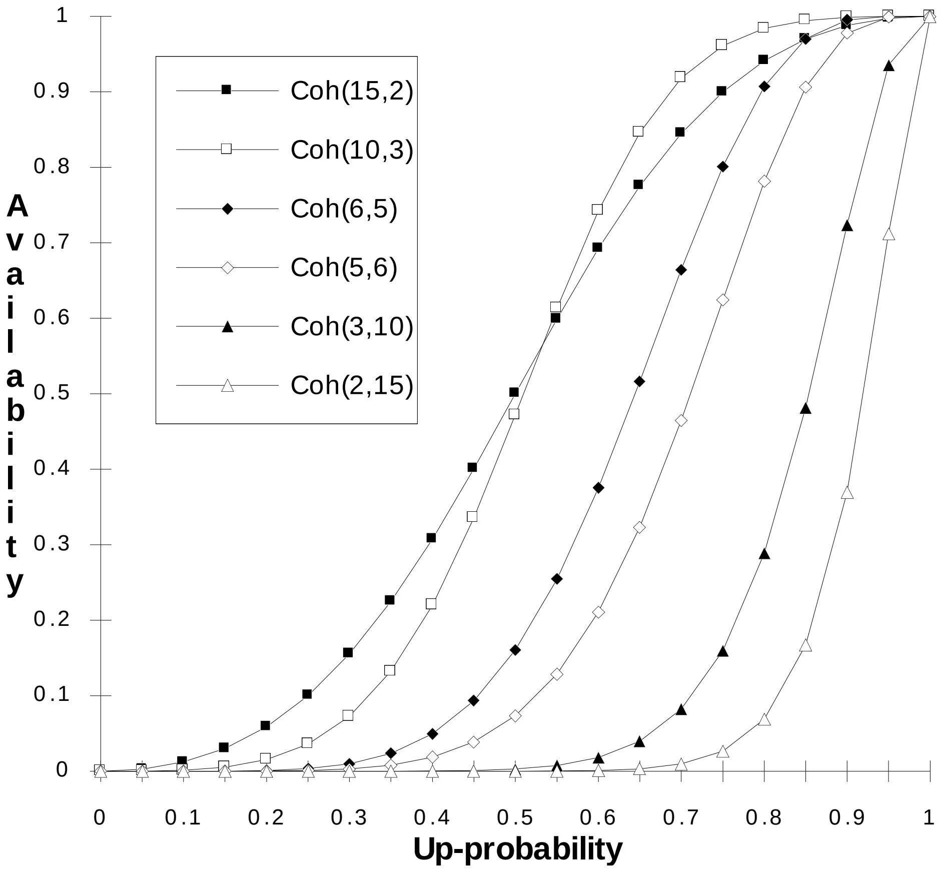

Figure 3.3 The availability of write quorums under various Cohorts structures.

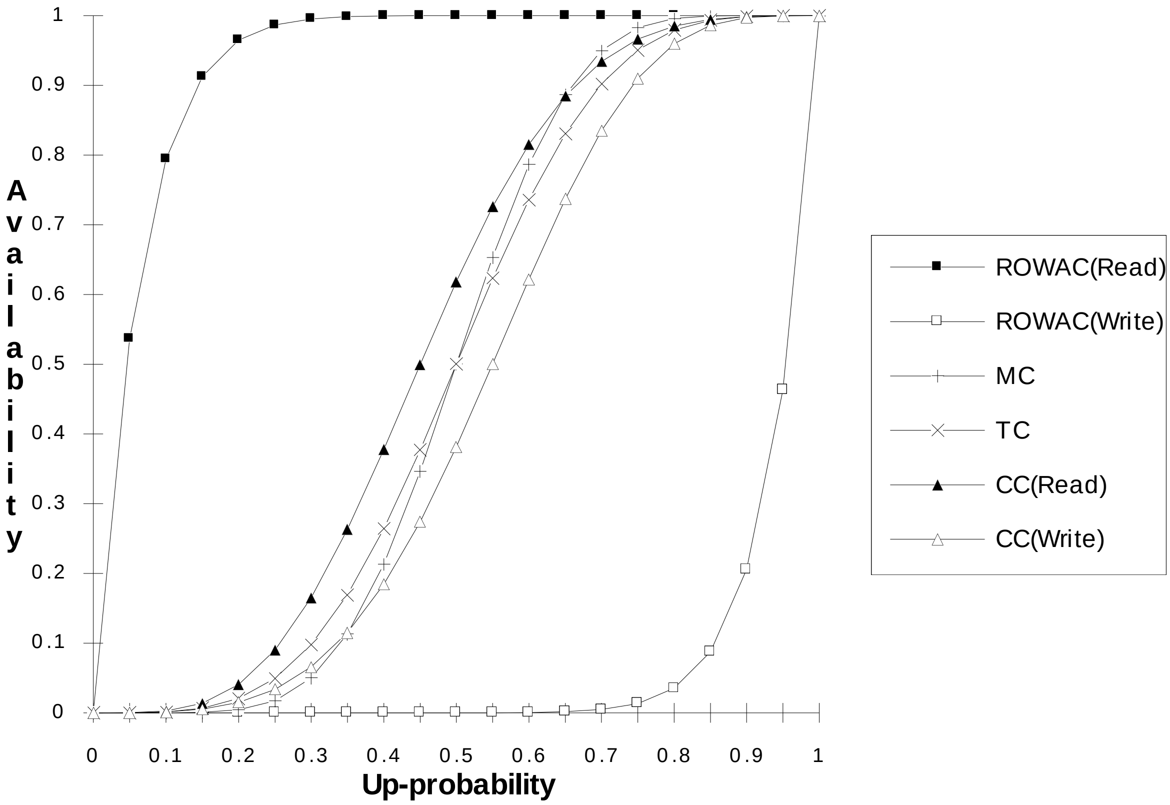

Figure 3.4 The availability comparison of various *wr*-coteries for the 15-replica system.

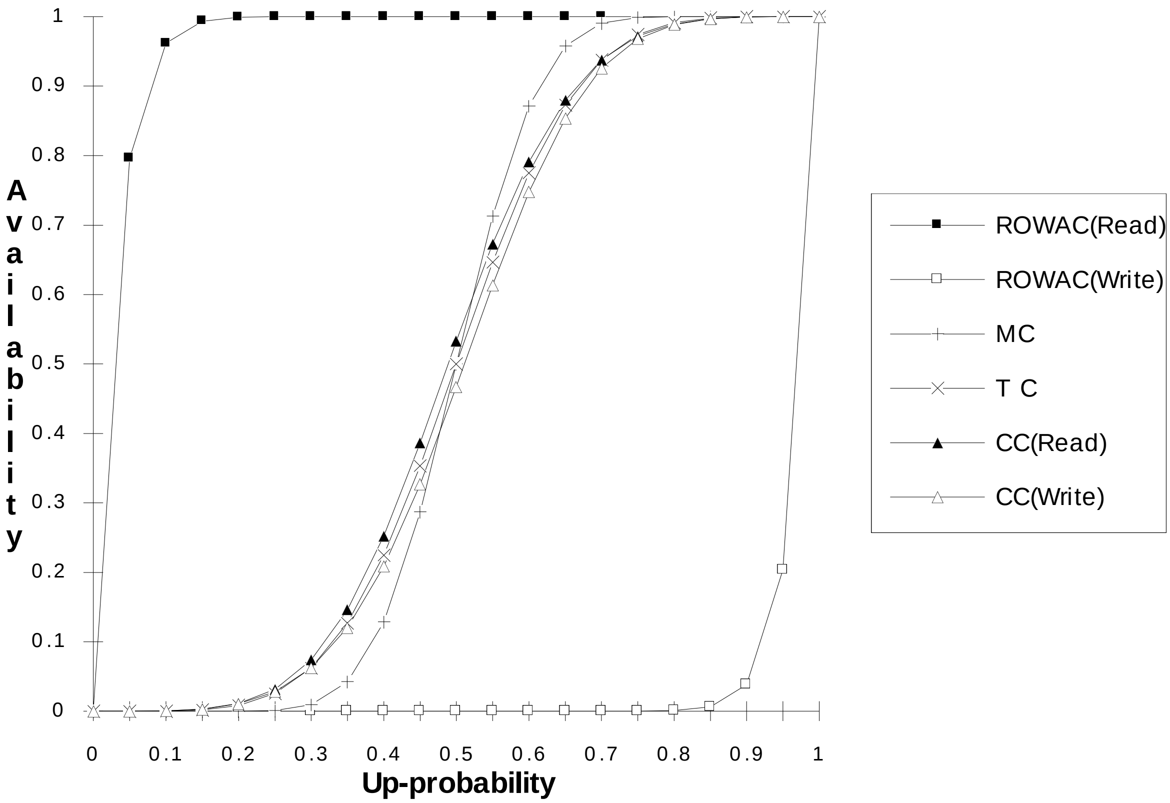

Figure 3.5 The availability comparison of various *wr*-coteries for the 31-replica system.

Chapter 4
Constructing *k*-coteries of constant expected quorum size

1. Introduction

A distributed system is a collection of sites that may communicate with each other by exchanging messages. *K*-mutual exclusion algorithms concern themselves with controlling the sites such that at most *k* sites can simultaneously access their critical sections. Such algorithms can be used to coordinate the sharing of a resource that can be allocated to no more than *k* sites at a time. Several distributed *k*-mutual exclusion algorithms [FYA91, HJK93, KFYA94, Nai93, Ray89, SR92] are proposed in the literature; some of them [FYA91, HJK93, KFYA94] rely on the concept of *k*-coteries. A *k*-coterie [FYA91, HJK93] is a family of sets (called quorums) in which any (*k*+1) quorums contain at least a pair of quorums intersecting each other. The concept of *k*-coteries is an extension of that of coteries [GB85]; that is, an 1-coterie (the value of *k* is taken as 1) is exactly a coterie. *K*-mutual exclusion algorithms using *k*-coteries require a site to collect enough permissions (votes) to form a quorum before accessing the critical section; they are fault-tolerant in the sense that a quorum may still be formed even when network partitioning [DGS85] occurs and makes some sites unavailable.

In this chapter, we propose a method for constructing k-coteries; the method can easily be extended to be a solution to distributed k-mutual exclusion. The solution utilizes a logical structure named *Cohorts* to construct quorums of $O(k)$ (k is a constant independent of n) size in the best case. When some sites are inaccessible, the quorum size increases gradually and may be as large as $O(n)$, where n is the number of sites. However, the expected quorum size is shown to remain constant as n grows. This is a desirable property since the message cost for accessing the critical section is directly proportional to the quorum size. We have also analyzed the availability of the constructed quorums and find that the availability of the constructed quorums is comparably high in comparison with those of relevant ones.

The remainder of this chapter is organized as follows. In section 4.2, we elaborate the concept of k-coteries. Then, in Section 4.3, we introduce the Cohorts structure and show how to construct quorums with its aid. In Section 4.4, we show that the collection of the constructed quorums is an k-coterie. In Section 4.5, we analyze and compare the constructed quorums with others in terms of availability and quorum size. At last, we conclude this chapter with Section 4.6

4.2 Preliminaries of *k*-coteries

A *k-coterie* [FYA91] $\boldsymbol{C}$ is a family of non-empty subsets of an underlying set U, which is a set containing all system sites $u_1,...,u_n$. Each member Q in $\boldsymbol{C}$ is called a quorum, and the following properties should hold for the quorums. The reader should note that an 1-coterie (the value of k is taken as 1) is exactly a coterie [GB85] introduced in Chapter 2.

Non-intersection Property:

For any h ($< k$) pairwise disjoint quorums Q_1, ... , Q_h in ***C***, there exists one quorum Q_{h+1} in ***C*** such that Q_1, ..., Q_{h+1} are pairwise disjoint.

Intersection Property:

There are no m, $m > k$, pairwise disjoint quorums in ***C*** (i.e., there are at most k pairwise disjoint quorums in ***C***).

Minimality Property:

There are no two quorums Q_1 and Q_2 in ***C*** such that Q_1 is a super set of $Q_{2.}$

For example, $\{\{u_1, u_3\}, \{u_1, u_4\}, \{u_2, u_3\}, \{u_2, u_4\}\}$ is a 2-coterie under $U=\{u_1,...,u_4\}$ because it satisfies all the properties of a 2-coterie—given one quorum Q_1, we can always find another quorum Q_2 such that Q_1 and Q_2 are disjoint; there are at most two pairwise disjoint quorums; and every quorum is not a super set of another quorum.

By the intersection and the non-intersection properties, the k-coterie can be used to develop algorithms to achieve k-entry critical sections. To enter the critical section, a site is required to receive permissions from all the members of some quorum in the system. By the intersection property, no more than k sites can form quorums simultaneously, so no more than k sites can access the critical section at the same time. The non-intersection property assures that if there exists one unoccupied entry of the critical section, then some site that waits for entering the critical section can proceed. Again, the minimality property for the k-coterie is for the enhancement of efficiency.

4.3 Construction of quorums

In this section, we present the Cohorts structure and propose an algorithm (function *Get_Quorum* in Figure 4.1) that can generate quorums with its help.

A ***Cohorts structure*** $Coh(k,l)=(C_1,...,C_l)$ is a list of pairwise disjoint sets; each set C_i is called a *cohort*. The Cohorts structure should observe the following two properties:

(P1) $|C_1|=k$.

(P2) $\forall i : 1<i\leq l$: $|C_i|>max(2k-2,k)$, where $max(a,b)=a$, if $a\geq b$; otherwise, $max(a,b)=b$. (Note that $max(2k-2, k)=2k-2$ when $k>1$; $max(2k-2, k)=k$ when $k=1$.)

To sum up, a Cohorts structure $Coh(k,l)$ has l pairwise disjoint cohorts with the first cohort having k members and the other cohorts having more than $2k-2$ members (or more than one member when $k=1$). For example, $(\{u_1, u_2\}, \{u_3, u_4, u_5\}, \{u_6, u_7, u_8, u_9, u_{10}\})$ is $Coh(2,3)$ since it has three pairwise disjoint cohorts with the first cohort and the other cohorts having 2 ($=k$) and more than 2 ($=2k-2$) members, respectively.

In this chapter, a member of a cohort is assumed as a physical site in the system, and henceforth, the words "site" and "member" are used exchangeably.

A set Q is said to be a **quorum under *Coh*(*k,l*)** if some cohort C_i in $Coh(k,l)$ is Q's *primary cohort*, and each cohort C_j, $j > i$, is Q's *supporting cohort*, where

(D1) a cohort C is Q's primary cohort if $|Q\cap C|=|C|-(k-1)$ (i.e., Q contains all except $k-1$ members of C), and

(D2) a cohort C is Q's supporting cohort if $|Q\cap C|=1$ (i.e., Q contains exactly one member of C).

For example, the following sets are quorums under $Coh(2,2)=(\{u_1,u_2\}, \{u_3,u_4,u_5\})$:

$Q_1=\{u_3, u_4\}$, $Q_2=\{u_3, u_5\}$, $Q_3=\{u_4, u_5\}$,

$Q_4=\{u_1, u_3\}$, $Q_5=\{u_1, u_4\}$, $Q_6=\{u_1, u_5\}$,

$Q_7=\{u_2, u_3\}$, $Q_8=\{u_2, u_4\}$, $Q_9=\{u_2, u_5\}$.

Quorums Q_1,...,Q_3 take $\{u_3, u_4, u_5\}$ as their primary cohort and no supporting cohort is needed, and quorums Q_4,...,Q_9 take $\{u_1, u_2\}$ as their primary cohort and $\{u_3, u_4, u_5\}$ as their supporting cohort. It is easy to check that these nine sets constitute a 2-coterie.

Note that for a quorum Q under *Coh*(k,l), the larger Q's primary cohort's index (subscript) is, the fewer the number of Q's supporting cohorts is. No supporting cohort is necessary when C_l is selected as Q's primary cohort.

A function called *Get_Quorum*, which can produce quorums under *Coh*(k,l), is shown in Figure 4.1. Function *Get_Quorum* can be modified and extended to solve the distributed k-mutual exclusion problem. In such a case, as in other quorum-based algorithms, a site is allowed to access the critical section after obtaining permissions from all sites of a quorum; a site is to return all its obtained permissions on leaving the critical section. Since a site may hold some permissions while waiting for other permissions, deadlock may thus occur. Mechanism proposed in [Mae85] or [San87] or [KFYA94] may be incorporated to avoid deadlock (and starvation); however, the details are not our focus and are thus omitted.

4.4 Correctness

In this subsection, we prove that the collection of quorums under *Coh*(k,l) is a k-coterie. Below, we will refer to such a k-coterie as *cohort coterie*.

Theorem 4.1. The collection of quorums under *Coh*(k,l) is a k-coterie for any l, $l\geq 1$.

Proof: (by induction on the value of l)

Basis: l=1.

Consider $Coh(k,1)=(C_1)$. Let C_1 be $\{u_1,...,u_k\}$ (note that by (P1) $|C_1|=k$). Then, all the quorums under $Coh(k,1)$ are $\{u_1\},...,\{u_k\}$. Those quorums obviously satisfy the non-intersection, the intersection, and the minimality properties of a k-coterie; hence, the theorem holds for the basis case.

Induction Hypothesis:

Assume the collection of quorums under $Coh(k,l-1)$ is a k-coterie, i.e., quorums under $Coh(k,l-1)$ satisfy the non-intersection, the intersection, and the minimality properties.

Induction Step:

On the basis of the induction hypothesis, we show below that quorums under $Coh(k,l)$ satisfy the non-intersection, the intersection, and the minimality properties of a k-coterie.

Let $C_l=\{u_1,...,u_s\}$, where $s=|C_l|>max(2k-2, k)$ (by (P2)). Then, a quorum under $Coh(k,l)$ may be of the form: either (**form-1**) a set of $s-(k-1)$ members of C_l, or (**form-2**) $\{u_i\} \cup$ a quorum under $Coh(k,l-1)$, $1 \le i \le s$. Note that C_l serves as the primary cohort for a form-1 quorum, and serves as a supporting cohort for a form-2 quorum.

● **Satisfaction of the non-intersection property**:

Suppose there are h, $h<k$, pairwise disjoint quorums $Q_1,...,Q_h$ under $Coh(k,l)$. We show that there still exists one quorum Q_{h+1} under $Coh(k,l)$ such that $Q_1,...,Q_{h+1}$ are pairwise disjoint. There are two cases to consider: (1) all h quorums are of form-2, and (2) one quorum is of form-1 and $h-1$ quorums are of form-2. Note that at most one of the quorums $Q_1,...,Q_h$ can be of form-1, for any two quorums of form-1 are not disjoint because $s-(k-1)+s-(k-1)>s$ (by $s>max(2k-2, k)$).

(1) All h quorums $Q_1,...,Q_h$ are of form-2:

It follows that $Q_1,...,Q_h$ take totally h ($h<k$) sites from C_l with $s–h$ sites left. Note that $s–h>s–k\geq s–(k–1)$. Let Q_{h+1} be a set that involves $s–k+1$ sites left in C_l. It is obvious that Q_{h+1} is a quorum under $Coh(k,l)$ and $Q_1,...,Q_{h+1}$ are pairwise disjoint.

(2) One quorum (say Q_h) is of form-1, and $h-1$ quorums (say $Q_1,...,Q_{h-1}$) are of form-2:

It follows that Q_h takes $s–(k–1)$ sites from C_l and each of $Q_1,...,Q_{h-1}$ takes one site from C_l. So, there are $s–(s–(k–1)+(h–1))=k–h$ (> 0, by $h<k$) sites left in C_l. Suppose that each form-2 quorum Q_i, $1\leq i\leq h–1$, contains a quorum R_i under $Coh(k,l–1)$, where $R_1,...,R_{h-1}$ are pairwise disjoint. Then, by hypothesis, we can find a quorum R under $Coh(k,l–1)$ such that $R_1,...,R_{h-1}$ and R are pairwise disjoint. Let $Q_{h+1} = R \cup$ the set of one arbitrary site left in C_l. It is obvious that Q_{h+1} is a quorum under $Coh(k,l)$ and $Q_1,...,Q_{h+1}$ are pairwise disjoint.

- **Satisfaction of the intersection property**:

Assume that there are m, $m>k$, pairwise disjoint quorums under $Coh(k,l)$. There are three cases to consider: (1) all m quorums are of form-2, (2) one quorum is of form-1 and $m–1$ quorums are of form-2, and (3) at least tow quorums are of form-1. For each case, we show that a contradiction occurs to conclude that there are at most k pairwise disjoint quorums under $Coh(k,l)$.

(1) All m quorums are of form-2:

This means that there are m, $m>k$, pairwise disjoint quorums under $Coh(k,l–1)$, which is a contradiction because, by hypothesis, there are at most k pairwise disjoint quorums under $Coh(k,l–1)$.

(2) One quorum (say Q_m) is of form-1, and m–1 quorums (say Q_1,...,Q_{m-1}) are of form-2:

This means that Q_m obtains s–(k–1) sites from C_l, and Q_1,...,Q_{m-1} obtain totally m–1 sites from C_l. This is a contradiction since s–(k–1)+m–1=s+(m–k)>s (by m>k).

(3) At least two quorums are of form-1:

Let Q_1 and Q_2 be two of the quorums of form-1. Then either of Q_1 and Q_2 takes s−(k−1) sites of C_l. This is a contradiction because s−(k−1)+s−(k–1)>s (by s> max(2k–2, k)).

● **Satisfaction of the minimality property**:

Any form-1 quorum is not a super set of any form-2 quorum because a quorum under Coh(k,l–1) is not contained in any set with s–k+1 sites of C_l. Also, any form-2 quorum is not a super set of any form-1 quorum because s–k+1>1 (by s>max(2k–2, k)). And it is obvious that any form-1 quorum is not a super set of another form-1 quorum, and any form-2 quorum is not a super set of another form-2 quorum (note that by hypothesis any quorum under Coh(k,l–1) is not a super set of another quorum under Coh(k,l–1)).

By now, on the basis of induction hypothesis, we have shown that the collection of quorums under Coh(k,l) is a k-coterie. Therefore, by the induction principle, the theorem holds for any l, l≥1. □

4.5 Analysis and comparison

In this section we analyze and compare quorums under Coh(k,l) with some other types of quorums in terms of availability and quorum size. Below, we assume that all sites have the same *up-probability p*, the probability that a single site is up (i.e., accessible). We also use S_i to denote $|C_i|$ for $1 \le i \le l$, where C_i is the ith item of $Coh(k,l)=(C_1,...,C_l)$. And we use $PR(s, a, b)$ to denote $\sum_{i=a}^{b}$ $[\mathrm{C}(s, i) \times p^i \times (1-p)^{s-i}]$, the probability that there exist a or a+1 or ... or b up members in a cohort with s members.

4.5.1 Availability

The availability of a coterie is defined as the probability that a quorum can be successfully formed. Since up to k pairwise disjoint quorums can be simultaneously formed in a k-coterie, we should discuss up to k cases for the availability of a k-coterie: the probability of a quorum being formed successfully, the probability of two pairwise quorums being formed successfully,..., and the probability of k pairwise disjoint quorums being formed successfully. The (k,h)-*availability*, $1 \le h \le k$, [KFYA93] is defined to be the probability that h pairwise disjoint quorums of a k-coterie can be formed successfully; it is used as a measure for the fault-tolerant ability of a solution using k-coterie.

Let $AV(h,l)$ be the function evaluating the probability that h pairwise disjoint quorums under $Coh(k,l)$ can be formed simultaneously. Function $AV(h,l)$ has the following two boundary conditions:

(1) $AV(0,l) = 1$.

(2) $AV(h,1) = PR(S_1, h, S_1)$. (Note that a quorum takes only one member from the first cohort to make it the primary cohort because $S_1-k+1=k-k+1=1$).

There are two possibilities for h quorums under $Coh(k,l)$ to be (recursively) constructed:

(1) One quorum is constructed with S_l-k+1 up sites of C_l (C_l thus serves as the primary cohort), and each of the other $h-1$ quorums is constructed with a quorum

under *Coh*(k,l–1) and an up site in C_l (C_l thus serves as a supporting cohort). Note that no two pairwise disjoint quorums can take C_l as their primary cohort, for (P2) $S_l > max(2k-2, k)$ implies $2(S_l-k+1)>S_l$.

(2) Each of the h quorums is constructed with a quorum under *Coh*(k,l–1) and an up site in C_l (C_l thus serves as a supporting cohort).

For the first case, C_l should have at least $(S_l-k+1)+(h-1)=S_l-k+h$ up members to be the primary cohort for one quorum and supporting cohorts for the remaining h–1 quorums. And for the second case, C_l should have at least h up sites to be supporting cohorts for the h quorums. However, the possibility of C_l having at least S_l-k+h up members should be ruled out from the second case since it has already been considered in the first case. Hence, we have

$$\begin{aligned} AV(h, l) = \; & AV(h-1, l-1) \times PR(S_l, S_l-k+h, S_l) + \\ & AV(h, l-1) \times PR(S_l, h, S_l-k+h-1) \qquad (4.1) \end{aligned}$$

4.5.1 Quorum size

In this section we analyze the size of the quorums under *Coh*(k,l). As mentioned earlier, for a quorum Q under *Coh*(k,l), the larger Q's primary cohort's index (subscript) is, the fewer the number of Q's supporting cohorts is. No supporting cohort is necessary when C_l is selected as Q's primary cohort. In such a case, Q has size S, $S=S_l-(k-1)$. For l=1, we have $S=C_1-k+1=1$ since by (P1) $C_1=k$. For l>1, we have $S>max(2k-2, k)-(k-1)$ since by (P2) $S_l>max(2k-2, k)$. If k=1, $max(2k-2, k)=k$; thus, we have $S>max(2k-2, k)-(k-1)=k-(k-1)=1$ (i.e., $S\geq 2$). If k>1, then $max(2k-2, k)=(2k-2)$; thus, we have $S>max(2k-2, k)-(k-1)=2k-2-(k-1)=k-1$ (i.e., $S\geq k$). To sum up, the lower bound of the sizes of quorums under *Coh*(k,l) is k if l>1 and k>1, is 2 if

l>1 and k=1. As for the upper bound of the size of quorums under *Coh*(k,l), it depends on the structure of *Coh*(k,l); it may be of O(n), however. For example, under Cohorts structure *Coh*(2, (n–2)/3)=({u_1, u_2}, {u_3, u_4, u_5}, {u_6, u_7, u_8},...,{u_{n-2}, u_{n-1}, u_n}), the largest quorum is of size O(n). Such a case occurs when C_1 is chosen as the primary cohort with others being supporting cohorts.

The lower bounds and upper bounds of the quorum sizes of the cohort coterie and the k-majority coterie are shown in Table 4.1.

The lower and upper bounds of the sizes of quorums under *Coh*(k,l) may be too optimistic and too pessimistic, respectively. Below, we analyze the expected size of quorums under *Coh*(k,l).

We apply the parameter f, as also used in [AE91], to indicate the fraction of quorums that take the last cohort as the primary cohort. Thus, 1–f is the fraction of quorums that take the last cohort as a supporting cohort rather than the primary cohort.

Let *ES*(l) denote the expected size of quorums under *Coh*(k,l). When l > 1, we have

$$ES(l)= f(S_l-k+1)+ (1-f)(1+ES(l-1)) \qquad (4.2)$$

The term $f(S_l–k+1)$ arises because there are f quorums of size (S_l–k+1); such quorums take C_l as the primary cohort and are composed of (S_l–k+1) sites of C_l. And the term (1–f)(1+*ES*(l–1) arises because there are (1–f) quorums of size *ES*(l–1)+1 that are composed of one site of C_l and one quorum under *Coh*(k,l–1). Since C_1 contains k site, a quorum under *Coh*(k,1) has size |C_1|–k+1=k–k+1=1. That is, *ES*(1)=1.

If we further restrict cohorts $C_2,...,C_l$ to have an equal size s (i.e., $S_2=...=S_l=s$), equation (4.2) can be regarded as a first-order linear equation [DOSE86]* and be solved analytically. Note that below we use $Coh(k,l,s)$ to denote such Cohorts structure. For $l>1$ and $f>0$, we have

$$ES(l)=(1-f)^{l-1}(1-s+k-(1/f)) + (s-k+(1/f)) \quad (4.3)$$

When l goes to infinity (and so does n), the term $(1-f)^{l-1}$ goes to 0, and hence $ES(l)$ goes to $s-k+(1/f)$, which is a constant. In other words, the expected size of the quorum under $Coh(k,l,s)$ remains constant when n grows. It is easy to see that smaller s or larger f produces smaller asymptotic expected quorum size. Take the following four cases for example: (case 1) f=0.5, s=3 (case 2) f=0.5, s=5 (case 3) f=0.25, s=3 and (case 4) f=0.25, s=5. When k=2, the asymptotic expected quorum sizes for these four cases are 3, 5, 5 and 7, respectively.

When $Coh(k,l,s)$, $l>>s$, is considered, the case of f=1 corresponds to the lower bound of the quorum size, which occurs when C_l is always chosen as the primary cohort. On the other hand, the case of f=0 corresponds to the upper bound of the quorum size, which occurs when a larger quorum is always chosen instead of a smaller one. Note that the probability that at least C_l-k+1 sites in C_l are up (i.e., $PR(s,s-k+1,s)$) can reflect the value of f. For example, the value of f can be reflected by $PR(3,2,3)$=0.71825 when s=3, k=2 and p=0.65.

4.5.3 Comparison

* A first-order linear difference equation of the form $X_k=aX_{k-1}+b$ for $k\geq2$ with X_1 being the first term has as its kth term $X_k=a^{k-1}(X_1+b/(a-1))-(b/(a-1))$ if $a\neq1$.

In this subsection we compare the cohort coterie with the *k*-majority coterie [KFYA93] and the *k*-singleton coterie [KFYA93] in terms of quorum size and availability.

A *k*-singleton coterie is a family $\{\{u_1\},...,\{u_k\}\}$, where $u_i \in U$, $1 \le i \le k$, and u_i's are distinct. It can be regarded as a special type of cohort coteries if we assume Cohorts structure $Coh(k,1)=(\{u_1\},...,\{u_k\})$ when generating quorums. Any set of $\lceil (n+1)/(k+1) \rceil$ sites can constitute a quorum of the *k*-majority coterie. Therefore, the quorum size of the *k*-majority coterie is $\lceil (n+1)/(k+1) \rceil$, which is of O($n$). If there are at least $h \times \lceil (n+1)/(k+1) \rceil$, $1 \le h \le k$, up sites, then *h* quorums of the *k*-majority coterie can be formed simultaneously. Let $H=h \times \lceil (n+1)/(k+1) \rceil$. The (k,h)-availability of *k*-majority quorums is then

Probability(*H* sites are available) +
Probability(*H*+1 sites are available) + ... +
Probability(*n* sites are available)=

$$\sum_{i=H}^{n} [\mathrm{C}(n,i) \times [p^i \times (1-p)^{(n-i)}]]$$

Figure 4.2 illustrates the (k,h)-availability, k=1,...,4 and h=1,..,k, of cohort coterie for 53-site system. Note that we choose the 53-site system so that the Cohorts structure $Coh(k, l, 2k-1)$, for k>1, or $Coh(1, \text{l}, 2)$, for k=1, may fit for the system size. The curves for the *k*-majority coterie are also depicted for comparison. When k=1, the availability (i.e., (1,1)-availability) of cohort coterie is better (resp., worse) than that of the *k*-majority coterie when up-probability *p* is smaller (resp., larger) than 0.5. And when k>1, cohort coterie are better than *k*-majority coterie for almost every up-probability in (3,3)-, (3,4)-, and (4,4)-availability (i.e., when both *k* and *h* are large). The cohort coterie are better (resp., worse) than the *k*-majority coterie in (2,1)-, (2,2)-, (3-1), and (3,2)-availability (i.e., when either *k* or *h* is small) if *p* is smaller (resp., larger) than a specific value (e.g., for *k*=3 and *h*=2, the specific value is about 0.5).

4.6 Summary

In this chapter, we have devised a method to construct quorums of a *k*-coterie; the method survives network partitioning and can easily be extended to be a solution to distributed *k*-mutual exclusion. With the aid of a logical structure named *Cohorts*, the method constructs quorums of constant size in the best case. When some sites are inaccessible, the quorum size increases gradually and may be as large as O(*n*), where *n* is the number of sites. However, the expected quorum size has been shown to remain constant as *n* grows. This is a desirable property since the message cost to access the critical section is directly proportional to the quorum size. We have also analyzed the availability of the constructed quorums and found that the availability of the constructed quorums is comparably high.

	k-majority coterie	cohort coterie (under $Coh(k,l)$, $l>1$)
Quorum size (Lower Bound)	$\lceil (n+1)/(k+1) \rceil$	2 (if $k=1$) k (if $k>1$)
Quorum size (Upper Bound)	$\lceil (n+1)/(k+1) \rceil$	$O(n)$

Table 4.1 Bounds on quorum sizes for the cohort coterie and the *k*-majority coterie.

```
Function Get_Quorum( $Coh(k,l)=(C_1,...,C_l)$: Cohorts Structure): Set;
VAR S: Set;
  If $l < 1$ Then Exit(failure);                // Illegal function call, claim failure //
  $S = Obtain(C_l)$;
  If $|S| = |C_l|-(k-1)$ Then Return(S);        // $C_l$ can be the primary cohort //
  If $|S| = 1$ Then Return($S \cup$ Get_Quorum($Coh(k,l-1)=(C_1,...,C_{l-1})$));
                                                // $C_l$ can be a supporting cohort but not the primary cohort //
  If $S = \emptyset$ Then Exit(failure);        // Unable to form a quorum, claim failure //
End Get_Quorum
```

Figure 4.1 A function that can generate quorums under $Coh(k,l)$.

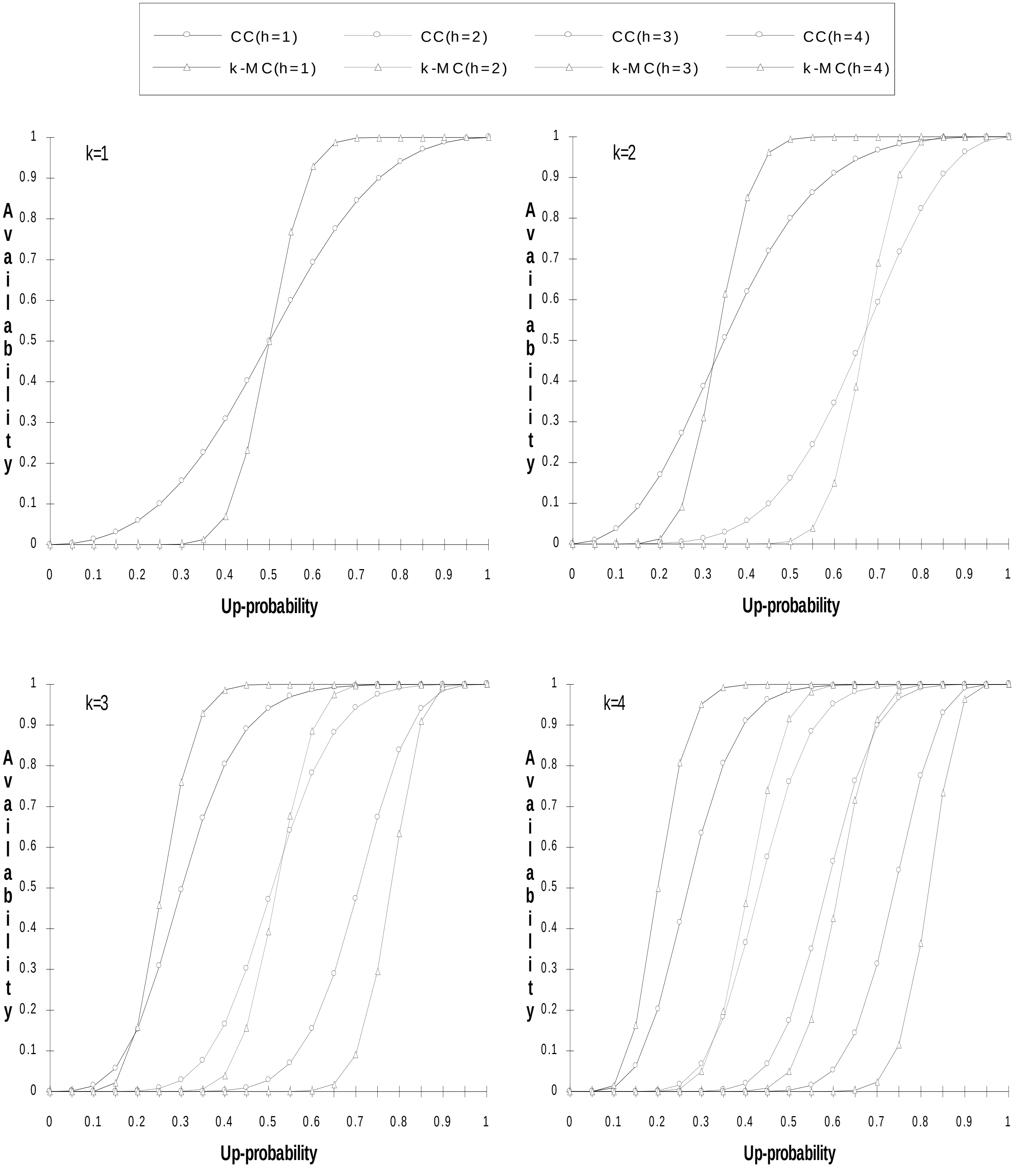

Figure 4.2 The (*k*,*h*)-availability comparison of the the cohort coterie (CC) and the *k*-majority coterie (*k*-MC) for the 53-site system.

Chapter 5
Constructing *ND k*-coteries from known *ND k*-coteries

1. Introduction

A distributed system is a collection of sites that may communicate with each other by exchanging messages. *K*-mutual exclusion algorithms concern themselves with controlling the sites such that at most *k* sites can simultaneously access their critical sections. Such algorithms can be used to coordinate the sharing of a resource that can be allocated to no more than *k* sites at a time. Several distributed *k*-mutual exclusion algorithms [FYA91, HJK93, KFYA94, Nai93, Ray89, SR92] are proposed in the literature; some of them [FYA91, HJK93, KFYA94] rely on the concept of *k*-coteries. A *k*-coterie [FYA91, HJK93] is a family of sets (called quorums) in which any (*k*+1) quorums contain at least a pair of quorums intersecting each other. The concept of *k*-coteries is an extension of that of coteries [GB85]; that is, an 1-coterie (the value of *k* is taken as 1) is exactly a coterie. *K*-mutual exclusion algorithms using *k*-coteries require a site to collect enough permissions (votes) to form a quorum before accessing the critical section; they are fault-tolerant in the sense that a quorum may still be formed even when network partitioning [DGS85] occurs and makes some sites unavailable.

A *k*-coterie is said to *dominate* another *k*-coterie if and only if every quorum in the dominated one is a super set of some quorum in the dominating one. The dominating one obviously has more chance than the dominated one for a quorum to be formed successfully in an error-prone environment. Thus, we should always concentrate on *nondominated* (*ND*) *k*-coteries that no *k*-coterie can dominate. Theorem 2.1 in [GB85] can be used to check the nondominance of coteries (1-coteries). On the basis of this theorem, many coteries proposed in the literature have been shown to be *ND*, such as the majority coterie (proposed in [Tho79] and shown to be *ND* for some special cases in [GB85]), the tree coterie (proposed in [AE91] and shown to be *ND* in [NM92]), the composite coterie (proposed and shown to be *ND* in [NM92]), the level coterie (proposed and shown to be *ND* in [SW93a]), the Lovasz coterie (proposed and shown to be *ND* in [Nei93]), and so on. Several *k*-coteries have been proposed in the literature, such as the cohorts coterie [HJK93], the *k*-majority coterie [KFYA93], and the *k*-singleton coterie [KFYA93]. The cohorts coterie is dominated (as shown in [NM94]), the *k*-majority coterie is *ND* for some special cases, and the *k*-singleton coterie is *ND*. The nondominance of the last two *k*-coteries will be addressed later.

In this chapter, we first introduce a theorem for checking the nondominance of *k*-coteries. Then, we define a special type of *ND* *k*-coteries—*strongly nondominated* (*SND*) *k*-coteries, and propose two operations—*union* and *join*—for generating new *SND* *k*-coteries from known *SND* *k*-coteries. An *SND* *k*-coterie is also an *ND* one, but not vice versa. We further show that every *ND* 1-coterie and every *ND* 2-coterie are *SND*. Thus, known *ND* 1-coteries and *ND* 2-coteries can be directly applied to the *union* or *join* operation to generate new *SND* *k*-coteries. We also show that the *k*-

singleton coterie is *SND* and that under some special conditions, the *k*-majority coterie is *SND* as well. An independently developed paper [NM94] also discussed properties of *ND k*-coteries; it introduced a theorem about *ND k*-coteries and two methods to generate *ND k*-coteries—the weighted voting (similar to the construction method of the *k*-majority coterie) and the composition (the same as the *union* operation). However, only part of the theorem introduced in [NM94] is proved correctly, thus, only part of the theorem can be assumed to be tenable. Later, we will point out the mistakes of [NM94] at proper places.

The remainder of this chapter is organized as follows. In Section 5.2, we introduce some related work. Then, in Section 5.3, we discuss *ND k*-coteries: we present a theorem for checking the nondominance of *k*-coteries, give the definition of *SND k*-coteries, and investigate some properties of *SND k*-coteries. Next, in Section 5.4, we introduce the two operations, *union* and *join*. The correctness of the two operations is also verified in this section. And finally, we conclude this chapter with Section 5.5.

5.2 Related Work

In this section, we review some related work about *ND k*-coteries. Since *k*-coteries are extended from coteries, below we first introduce the concept of coteries. In the following context we let *U* be the underlying set of all system sites. Note that we may not specify *U* wherever there is no ambiguity.

The concept of coteries was first proposed by Garcia-Molina and Barbara [GB85]. A *coterie* [GB85] ***C*** under *U* is a family of non-empty subsets of *U*; each member of

C is called a *quorum*. The following properties should hold for the quorums in a coterie:

Intersection Property:

There are no two quorums Q_1 and Q_2 in C such that $Q_1 \cap Q_2 = \emptyset$

Minimality Property:

There are no two quorums Q_1 and Q_2 in C such that Q_1 is a proper subset of $Q_{2.}$

For example, C ={{1, 2}, {2, 3}, {2, 3}} is a coterie under U={1, 2, 3} because every pair of quorums have a non-empty intersection, and no quorum is a proper subset set of another quorum.

By the intersection property, the coterie can be used to develop mutual exclusion (1-mutual exclusion) algorithms in distributed systems. To enter the critical section, a site is required to receive permissions from all sites of some quorum. Since any pair of quorums have at least one member in common, mutual exclusion is then guaranteed. The reader should note that the minimality property is not necessary for the correctness of mutual exclusion algorithm but is used to enhance efficiency. Mutual exclusion algorithms using coteries are fault-tolerant because even in the presence of inaccessible sites, quorums including no inaccessible sites may still be found.

Let C and D be two coteries. D is said to *dominate* [GB85] C if and only if ($C \neq D$) and ($\forall R \in C\ \exists S \in D, S \subseteq R$). For example, coterie D = {{1, 2}, {1, 3}, {1, 4}, {2, 3, 4}} dominates coterie C = {{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {2, 3, 4}} because for every quorum R in C, we can find a quorum S in D such that S is a subset of R.

Coterie *D* is more resilient to site and/or communication link failures than coterie *C*. Assuming that a site or a communication link failure occurs to make both site 2 and site 3 unavailable, then no quorum can be formed in *C*, but one quorum {1, 4} can still be formed in *D*.

A coterie is said to be *nondominated* (*ND*) if no coterie can dominate it. A dominating coterie, such as D in the last paragraph, is superior to a dominated coterie, such as C in the last paragraph because if a quorum can be formed in the dominated one, then a quorum can be formed in the dominating one. Thus, we should always focus on the *ND* coteries. However, checking the nondominance of a coterie seems to be a hard problem, as mentioned in [GB85].

The following Theorem 5.1 is actually Theorem 2.1 developed by Garcia-Molina and Barbara in [GB85]. This theorem is useful in examining the nondominance of coteries.

Theorem 5.1. Let ***C*** be a coterie under *U*. Then, ***C*** is dominated if and only if there exists a set $S \in U$ such that

L1. For any quorum $R \in C$, $R \not\subseteq S$.
L2. For any quorum $R \in C$, $R \cap S = \emptyset$.

By Theorem 5.1, if there does not exist a set satisfying (L1) and (L2) for a coterie *C*, then *C* is *ND*; otherwise, *C* is not *ND* (dominated).

There are many *ND* coteries proposed in the literature, such as the majority coterie [Tho79], the tree coterie [AE91], the composite coterie [NM92], the level coterie [SW93], the Lovasz coterie[Nei93], and so on. The majority coterie corresponds to the majority quorum consensus algorithm [Tho79], in which each quorum is required to have the majority (over half) of sites. This coterie is shown to be *ND* when *n* is odd [GB85], where *n* is the cardinality of the underlying set *U*. The quorum of the tree coterie is formed by the tree quorum algorithm [AE91]. By organizing system sites into a binary tree, the tree-quorum algorithm forms a quorum recursively; it attempts to obtain permissions from nodes along a root-to-leaf path. If the root node fails, then the obtaining should follow two paths: one root-to-leaf path on the left subtree and one root-to-leaf path on the right subtree. The tree coterie is shown to be *ND* in [NM92]. The composite coterie [NM92] is generated by joining two coteries. As shown in [NM92], if the coteries used for joining are both *ND*, then the composite coterie is also *ND* (see Section 5.4.2 for more details of joining two coteries). By logically organizing sites into different levels (except the last one, every level should have more than one sites), a quorum of the level coterie [SW93a] is formed by obtaining permissions from all sites in some level (say *i*) and one site in each of levels $i-1$, $i-2$,...,1. The level coterie, as shown in [SW93a], is *ND* if the last level has exactly one site. If the last level has more than one sites, then the following steps should be taken to make the level coterie *ND*: (1) construct an *ND* coterie C under the set of the last-level sites, and (2) when the last level is considered, permissions from sites in any quorum of C (instead of all sites in the last level) and one site in every level (except the last level) are enough to form a quorum of the level coterie. The Lovasz coterie [Nei93] is based on a partition of the underlying set *U*. Let $\mathbf{P}=\{P_1, P_2, \ldots, P_m\}$ be a partition of *U* (i.e., P_i's are pairwise disjoint and $\bigcup_{i=1}^{m} P_i = U$) such that $|P_i|=i$. A quorum in the Lovasz coterie is formed by obtaining permissions from all the sites in P_i and one site from each P_j, where $i<j\leq m$. The Lovasz coterie has been shown to be *ND* in [Nei93]. Note that the Lovasz coterie can

be regarded as a special case of the level coterie (by reversing the indices of the levels).

Below, we introduce the concept of *k*-coteries. Two different definitions of *k*-coteries are given in the literature: the one by Fujita, Yamashita and Ae [FYA91], and the one by Huang, Jiang and Kuo [HJK93]. The former is more restrictive than the latter, and we adopt the more restrictive one (i.e., the one proposed by Fujita, Yamashita and Ae [FYA91]), however.

A *k-coterie* [FYA91] ***C*** under *U* is a family of non-empty subsets of *U*; each member *Q* in ***C*** is called a quorum. The following properties should hold for the quorums in a *k*-coterie ***C***.

Non-intersection Property:

For any h ($< k$) pairwise disjoint quorums $Q_1,...,Q_h$ in ***C***, there exists one quorum Q_{h+1} in ***C*** such that $Q_1,...,Q_{h+1}$ are pairwise disjoint.

Intersection Property:

There are no m, $m > k$, pairwise disjoint quorums in ***C*** (i.e., there are at most k pairwise disjoint quorums in ***C***).

Minimality Property:

There are no two quorums Q_1 and Q_2 in ***C*** such that Q_1 is a proper subset of $Q_{2.}$

For example, {{1,2}, {3,4}, {1,3}, {2,4}} is a 2-coterie because it satisfies all the properties of a 2-coterie—given one quorum Q_1, we can always find another quorum

Q_2 such that Q_1 and Q_2 are disjoint; there are at most two pairwise disjoint quorums; and every quorum is not a proper subset of another quorum.

K-coteries can be used to develop *k*-mutual exclusion algorithms [FYA91, HJK93, KFYA94]. To enter the critical section, a site is required to obtain permissions from all sites of some quorum. By the intersection property, no more than *k* sites can form quorums simultaneously, so no more than *k* sites can access the critical section at the same time. The non-intersection property assures that if there exists one unoccupied critical section entry, then some site that is not in the critical section can enter the critical section. Again, the minimality property has nothing to do with the correctness of *k*-mutual exclusion algorithms; it is only for the enhancement of efficiency. *K*-mutual exclusion algorithms using *k*-coteries are fault-tolerant in the sense that even though there are inaccessible sites in the system, quorums not including inaccessible sites may still be found.

According to the definition of coterie nondominance [GB85], the nondominance of *k*-coteries can also be defined identically. We will leave all of problems of *ND k*-coteries to be discussed in the next section.

5.3 *ND k*-coteries

In this section, we address some properties about nondominated *k*-coteries. We start by giving, according to the definition of coterie domination, the definition of domination of *k*-coteries:

***Definition* 5.1.**

Let C and D be two k-coteries. D *dominates* C if and only if ($C \neq D$) and ($\forall R \in C\ \exists S \in D, S \subseteq R$).
(We say that S is the quorum that dominates R.)

For example, consider the following 2-coteries:

A = { {1, 2}, {3, 4}, {1, 3}, {2, 4} }

B = { {1, 2}, {3, 4}, {1, 3}, {2, 4}, {1, 4}, {2, 3} }

C = { {1, 2}, {1, 3}, {2, 3}, {4} }

It is easy to see that A is dominated by both B and C, and B is dominated by C.

The dominating k-coterie (such as C) is superior to the dominated k-coterie (such as A or B) since if a quorum can be formed in the latter then a quorum can be formed in the former. Thus, we should always concentrate on the *nondominated* (*ND*) k-coteries that no k-coteries can dominate. In the light of Theorem 5.1, we introduce Theorem 5.2 for the examination of k-coterie nondominance. In comparison with Theorem 5.1, Theorem 5.2 merely has "only if" part, and (L1) is the same as (L1), and when k is taken as 1, (L2) is the same as (L2).

A theorem (Theorem 2.1 in [NM94]) similar to Theorem 5.2 has been independently developed. Theorem 5.2 and Theorem 2.1 in [NM94] are identical except that the latter has "if" and "only if" parts and the former just asserts the "only if" part. The proof of the "if" part of Theorem 2.1 in [NM94] is not correct because it depends on the following incorrect assertion that if there exists a set satisfying (L1) and (L2) for a k-coterie C and there is no super set of S in C, then $C \cup \{S\}$ is a k-coterie that dominates C. Note that $C \cup \{S\}$ indeed satisfies the intersection and the minimality properties but it may not fulfill the non-intersection property and hence $C \cup \{S\}$ may not be a k-coterie. For example, let C={{1, 2}, {3, 4}} be a 2-coterie.

Then, S={1, 3} is a set satisfying (L1) and (L2) for C, and there is no super set of S in C. It is easy to see that $C\cup\{S\}$={{1, 2}, {3, 4}, {1, 3}} is not a 2-coterie since it violates the non-intersection property.

Theorem 5.2. Let $\boldsymbol{C}$ be a k-coterie under U. Then, $\boldsymbol{C}$ is dominated *only if* there exists a set $S\in U$ such that

L1. For any quorum $R\in C$, $R \not\subseteq S$.

L2. For any k pairwise disjoint quorums $R_1,...,R_k\in C$, $R_1,...,R_k$ and S are not pairwise disjoint.

Proof:

Assume that C is dominated by D. We show that (L1) and (L2) hold by considering two cases: $C \subset D$ or $C \not\subset D$.

For the first case, $C \subset D$. Let S be one of the quorums in $D–C$. We have $S\in$D and $S\notin$C. On one hand, since each quorum R in C is also a quorum in D, and $S\neq R$ (by $S\in$ D and $S\notin$C), (L1) must hold or else D would violate the minimality property. On the other hand, since quorums $R_1,...,R_k$ in C are also quorums in D, (L2) must hold or else D would violate the intersection property.

For the second case, $C \not\subset$ D. Let R be one of the quorums in C–D. We have $R\in$C and $R\notin$D. Further, let S be the member in D that dominates R; i.e., $S\in$D and $S\subseteq R$. Hence, we have $S\neq R$ (by $S\in$D and $R\notin$D) and therefore $S\subset R$. On one hand, we assume that (L1) is false for S; i.e., there exists an R' such that $R'\in$C and $R'\subseteq S$. We have $R'\subseteq S\subset R$, which concludes that C violates the minimality property. This is a contradiction, and thus (L1) must hold for S. On the other hand, we assume that (L2) does not hold for S; i.e., we can find pairwise disjoint quorums $R_1,...,R_k$ in C such that $R_1,...,R_k$ and S are pairwise disjoint. Let S_i, $1\leq i\leq k$, be the quorum in D that dominates R_i (i.e., $S_i \in D$ and $S_i \subseteq R_i$). Then, we have that $S_1,...,S_k$ and S are pairwise disjoint, which concludes that D violates the intersection property. This is a contradiction, and thus (L2) must hold for S. □

The contrapositve of Theorem 5.2—if we can not find any subset of U that satisfies both (L1) and (L2) for a k-coterie C, then C is not dominated—can be used to examine the nondominance of k-coteries. However, the existence of a set satisfying (L1) and (L2) for a k-coterie C does not mean that C is dominated (i.e., C may still be nondominated). Below, we define a more strict type of *ND* k-coteries—*strongly nondominated* (*SND*) such that if, and only if, we can not find any subset of U that satisfies both (L1) and (L2) for a k-coterie C, then can C be called an *SND* k-coterie.

***Definition* 5.2.**

Let C be a k-coterie. C is *strongly nondominated* (*SND*) if and only if we cannot find a set satisfying (L1) and (L2) for C.

Note that by Theorem 5.2 and Definition 5.2, an *SND* k-coterie is also an *ND* k-coterie, but not vice versa. In Section 4, we will introduce two operations that can generate new *SND* k-coteries from known *SND* k-coteries. Below, we discuss some properties about *SND* k-coteries. We first show the relation between *SND* and *ND* k-coteries for k=1 and 2, and then show that for some special cases the k-majority coterie [KFYA93] is *SND* and that the k-singleton coterie [KFYA93] is *SND*, too.

Theorem 5.3. Every *ND* 1-coterie is *SND*.

Proof:

Let C be an 1-coterie. By Theorem 5.1, we have that if C is not dominated (i.e., nondominated), then we can not find a set satisfying (L1) and (L2) for C, which means that C is *SND*. □

As we have shown earlier, there are many *ND* 1-coteries proposed: the majority coterie [Tho79], the tree coterie [AE91], the composite coterie [NM92], the level coterie [SW93a], the Lovasz coterie[Nei93], and so on. As Theorem 5.3 states, these *ND* coteries are all *SND*; they can be used to generate new *SND* k-coteries with the operations developed in Section 4.

Now, we discuss the relation between *ND* and *SND* 2-coteries. Consider a 2-coterie C for which we can find a set S satisfying (L1) and (L2). The following function *Reduce* can reduce S to S' (S'=*Reduce*(C, S)) such that S' still satisfies (L1) and (L2) for C.

```
Function Reduce(C: 2-coterie, S: Set): Set;
For (every member $s$ in $S$) Do
  For (every two disjoint quorums $Q_1$ and $Q_2$ in C) Do
    If $(S\cap(Q_1\cup Q_2))=\{s\}$ Then goto Skip;
  EndFor
  $S=S-\{s\}$;
  Skip:
EndFor
Return($S$);
End Reduce
```

Function *Reduce* checks each element s in S one by one: if there exists a pair of disjoint quorums Q_1 and Q_2 in C such that $(S\cap(Q_1\cup Q_2))=\{s\}$ then s is retained in S;

otherwise *s* is removed from *S* (i.e., *s* is removed from *S* if for all pairs of disjoint quorums Q_1 and Q_2 in *C*, either $s \notin (S \cap (Q_1 \cup Q_2))$ or ($s \in (S \cap (Q_1 \cup Q_2))$ and $|S \cap (Q_1 \cup Q_2)| > 1$)). It is obvious that S', $S'=Reduce(C,S)$, still satisfies (L1) and (L2) for *C* and there exists a pair of disjoint quorums Q_1 and Q_2 in *C* such that $|S' \cap (Q_1 \cup Q_2)|=1$.

With the aid of function *Reduce*, we can show the following Lemma 5.1, by which we can show Theorem 5.4 — every *ND* 2-coterie is *SND*.

Lemma 5.1. Let *C* be a 2-coterie. If we can find a set *S* satisfying (L1) and (L2) then C is dominated.

Proof:

Let $S'=Reduce(S)$. Then, S' satisfies (L1) and (L2) and there exist a pair of disjoint quorums Q_1 and Q_2 such that $|S \cap (Q_1 \cup Q_2)|=1$. Since $Q_1 \cap Q_2 = \emptyset$, we have $S' \cap Q_1 = \emptyset$ or $S' \cap Q_2 = \emptyset$; i.e., we can find a set *Q* ($Q=Q_1$ or $Q=Q_2$) such that $Q \cap S' = \emptyset$.

Below, we consider two cases: either (1) there are no super set of *S* in *C* or (2) there are quorums $Q_1,...,Q_h$ in *C* such that $Q_1,...,Q_h \supset S'$.

(1). There is no super set of *S* in *C*. Let $D=C \cup \{S'\}$. *D* is a 2-coterie because *C* is a 2-coterie, S' satisfies (L1) and (L2), and we can find a set *Q* in C such that $Q \cap S' = \emptyset$. It is obvious that *D* dominates *C*.

(2). There are quorums $Q_1,...,Q_m$ in *C* such that $Q_1,...,Q_m \supset S'$. Let $D=(C-\{Q_1,...,Q_m\}) \cup \{S'\}$. *D* is a 2-coterie because *C* is a 2-coterie, S' satisfies (L1) and (L2), $Q_1,...,Q_m \supset S'$ (hence, for any quorum *R* in $C-\{Q_1,...,Q_m\}$, if $R \cap Q_i = \emptyset$, $1 \le i \le m$, then $R \cap S' = \emptyset$), and we can find a set *Q* in C such that $Q \cap S' = \emptyset$. It is obvious that *D* dominates *C*.

□

For example, consider a 2-coterie $C=\{\{1, 3\}, \{1, 4\}, \{2, 5\}\}$. We can find a set $S=\{3, 4\}$ satisfying (L1) and (L2) for C. Let $S'=Reduce(C, S)=\{3, 4\}$ and $D=C\cup\{S'\}=\{\{1, 3\}, \{1, 4\}, \{2, 5\}, \{3, 4\}\}$. It is obvious that D is a 2-coterie and D dominates C. For another example, consider a 2-coterie $C=\{\{1, 2\}, \{3, 4\}\}$. We can find a set $S=\{1, 3\}$ satisfying (L1) and (L2) for C. Let $S'=Reduce(C, S)=\{3\}$ and $D=(C-\{3, 4\})\cup\{S'\}=\{\{1, 2\}, \{3\}\}$. It is obvious that D is a 2-coterie and D dominates C.

Theorem 5.4. Every *ND* 2-coterie is *SND*.

Proof:

Let C be a 2-coterie. By Lemma 5.1, we have that if C is not dominated (i.e., nondominated), then we can not find a set satisfying (L1) and (L2) for C, which means that C is *SND*. □

By now, we have shown that every *ND* 1-coterie and every *ND* 2-coterie are *SND*. Thus, we can use the operations provided in Section 5.4 to generate new *SND* k-coteries from known *ND* 1-coteries and *ND* 2-coteries. However, the problem of whether any *ND* k-coterie, $k>2$, is *SND* remains open.

Below, we show that the k-majority coterie is *SND* if $(n+1)$ is a multiple of $(k+1)$, where n is the cardinality of U. Note that a k-majority coterie [KFYA93] is a k-coterie that consists of quorums with $\lceil (n+1)/(k+1) \rceil$ sites.

Theorem 5.5. Let C be a k-majority coterie. If $(n+1)$ is a multiple of $(k+1)$, then C is *SND*.

Proof: (The proof is by contradiction)

Suppose C is not *SND*, then we can find a set S that satisfies (L1) and (L2). Let $R_1,...,R_k$ be any pairwise disjoint quorums in C. We have

(1) $\lceil (n+1)/(k+1) \rceil = (n+1)/(k+1)$ (since $(n+1)$ is a multiple of $(k+1)$)

(2) $|R_i| = (n+1)/(k+1)$ for $1 \leq i \leq k$ (by (1) and the k-majority coterie definition)

(3) $|S| < (n+1)/(k+1)$ (by (L1))

(4) $|S| > n-(|R_1|+...+|R_k|)$ (by (L2))

(5) $|S| > n-k(n+1)/(k+1)=(n+1)/(k+1)-1$ (by (2) and (4))

By (3) and (5), we have a contradiction. Therefore, C is *SND*. □

Below, we show that the k-singleton coterie [KFYA93] is also *SND*. Note that a k-singleton coterie is a family $\{\{u_1\},...,\{u_k\}\}$, where $u_i \in U$, for $1 \leq i \leq k$, and u_i's are distinct.

Theorem 5.6. Let C be a k-singleton coterie, then C is *SND*.
Proof:

Because we can not find a set satisfying (L1) and (L2) for a k-singleton coterie, it is *SND* by definition. □

By now, we have shown that both the k-majority coterie (for the case of $(n+1)$ being a multiple of $(k+1)$) and the k-singleton coterie are *SND*. Thus, they can both be used to generate new *SND* k-coteries with the operations provided in Section 4.

5.4 The *Join* and *Union* Operations

In this section, we introduce two operations, $\oplus$ (*union*) and $\otimes$ (*join*), which can generate new *SND* *k*-coteries from known *SND* *k*-coteries. We first introduce $\oplus$ (*union*), and then $\otimes$ (*join*).

5.4.1 Coterie *Union* Operation

Let U_1 and U_2 be two non-empty sets of sites, where $U_1 \cap U_2 = \varnothing$. Also, let X be a k_1-coterie under U_1, and Y be a k_2-coterie under U_2. The coterie *union* operation $\oplus$ is defined as $X \oplus Y = \{Q | Q \in \mathrm{X} \text{ or } Q \in \mathrm{Y}\}$.

Paper [NM94] has also proposed the union operation (called composite operation in [NM94]) to produce new *k*-coteries from known *k*-coteries. However, part of its correctness prove is based on Theorem 2.1 in [NM94], which is incorrect as mentioned earlier.

Let $U = U_1 \cup U_2$ and $\mathrm{Z} = X \oplus Y$. The following Theorem 5.7 and Theorem 5.8 are about properties of Z.

Theorem 5.7. Z is a (k_1+k_2)-coterie under U.

Proof:

There are at most k_1+k_2 pairwise disjoint quorums in Z because there are at most k_1 pairwise disjoint quorums in X and there are at most k_2 pairwise disjoint quorums in Y. Further, every quorum in Z is not a proper subset of any quorum in Z because every quorum in X is not a proper subset of any quorum in X, every quorum in Y is not a proper subset of any quorum in Y, and by $U_1 \cap U_2 = \varnothing$, every quorum in X (resp., Y) is not a proper subset of any quorum in Y (resp., X).

Below, we show that for any h, $h < k_1+k_2$, pairwise disjoint quorums $Z_1, \ldots, Z_h$ in Z, we can find a quorum Z_{h+1} in Z such that $Z_1, \ldots, Z_{h+1}$ are pairwise disjoint. Since Z=X

$\cup$Y, we may assume that among $Z_1,...,Z_h$, there are h_1 quorums (say $X_1,...,X_{h_1}$) coming from X and h_2 quorums (say $Y_1,...,Y_{h_2}$) coming from Y, where $h=h_1+h_2$. Since $h<k_1+k_2$, we have (1) $h_1<k_1$ or (2) $h_2<k_2$ because if not so (i.e., $h_1\geq k_1$ and $h_2\geq k_2$), we have $h=h_1+h_2\geq k_1+k_2$, which contradicts to $h<k_1+k_2$.

Without loss of generality, let $h_1<k_1$. Then, we can find a quorum X in X such that X and $X_1,...,X_{h_1}$ are pairwise disjoint since X is a k_1-coterie. Moreover, X and $Y_1,...,Y_{h_2}$ are pairwise disjoint since $U_1\cap U_2=\varnothing$. Hence, X and $Z_1,...,Z_h$ are pairwise disjoint. Let $Z_{h+1}=X$; we then have that $Z_{h+1}\in Z$ and $Z_1,...,Z_{h+1}$ are pairwise disjoint.

Z satisfies all the properties of a (k_1+k_2)-coterie and it is obvious that any quorum in Z is non-empty and is contained in U. Hence, Z is a (k_1+k_2)-coterie under U. □

Theorem 5.7 states that if X is a k_1-coterie and Y is a k_2-coterie, then Z=$X \oplus Y$ is a (k_1+k_2)-coterie. For example, let X be a 2-coterie $\{\{a, b\},\{c, d\}, \{a, c\}, \{b, d\}\}$ under $\{a, b, c, d\}$, and Y be a coterie $\{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$ under $\{1, 2, 3\}$, then $Z=X \oplus Y=\{\{a, b\},\{c, d\}, \{a, c\}, \{b, d\}, \{1, 2\}, \{1, 3\}, \{2, 3\}\}$ is a 3-coterie under $\{a, b, c, d, 1, 2, 3\}$.

Below, we discuss the nondominance property of Z in Theorem 5.8.

Theorem 5.8. If X and Y are *SND*, then Z is *SND*.
Proof: (The proof is by contradiction)

Suppose Z is not *SND*, then we can find a set S, $S\subseteq U$, satisfying (L1) and (L2) for Z. Let $S_1=S\cap U_1$ and $S_2=S\cap U_2$. Then, $S_1\subseteq S$ and $S_2\subseteq S$. Further, $S=S_1\cup S_2$ since $U=U_1\cup U_2$ and $S\subseteq U$. By (L1), we have $\forall R\in Z, R\not\subseteq S$. Thus, we have $\forall R\in Z, R\not\subseteq S_1$ and $\forall R\in Z$,

$R\not\subseteq S_2$ because $S_1\subseteq S$ and $S_2\subseteq S$. Since $X\subseteq Z$ and $Y\subseteq Z$ (by $Z=X\cup Y$), we have $\forall R\in X$, $R\not\subseteq S_1$ and $\forall R\in Y$, $R\not\subseteq S_2$; i.e., S_1 satisfies (L1) for X and S_2 satisfies (L1) for Y.

Let $X_1,\ldots,X_{k_1}$ be any k_1 pairwise disjoint quorums in X, and $Y_1,\ldots,Y_{k_2}$ be any k_2 pairwise disjoint quorums in Y. Then, $X_1,\ldots,X_{k_1}\in$Z and $Y_1,\ldots,Y_{k_2}\in$Z. Since $U_1\cap U_2=\emptyset$, we have $X_1,\ldots,X_{k_1}$ and $Y_1,\ldots,Y_{k_2}$ are k_1+k_2 pairwise disjoint quorums in Z. By (L2), $X_1,\ldots,X_{k_1}$, $Y_1,\ldots,Y_{k_2}$ and S are not pairwise disjoint, or equivalently, $(S_1\cup S_2)\cap((X_1\cup\ldots\cup X_{k_1})\cup(Y_1\cup\ldots\cup Y_{k_2}))\neq\emptyset$ (note that $S=S_1\cup S_2$ and $X_1,\ldots,X_{k_1}$, $Y_1,\ldots,Y_{k_2}$ are pairwise disjoint). We have $(S_1\cap(X_1\cup\ldots\cup X_{k_1}))\cup(S_2\cap(Y_1\cup\ldots\cup Y_{k_2}))\neq\emptyset$ by $S_1\subseteq U_1$, $S_2\subseteq U_2$, $X_1,\ldots,X_{k_1}\subseteq U_1$, $Y_1,\ldots,Y_{k_2}\subseteq U_2$, and $U_1\cap U_2=\emptyset$. Hence, we have (1) $S_1\cap(X_1\cup\ldots\cup X_{k_1})\neq\emptyset$ or (2) $S_2\cap(Y_1\cup\ldots\cup Y_{k_2})\neq\emptyset$.

Without loss of generality, let $S_1\cap(X_1\cup\ldots\cup X_{k_1})\neq\emptyset$. Then, S_1 satisfies (L2) for X since we assume that $X_1,\ldots,X_{k_1}$ are any k_1 pairwise disjoint quorums in X. Thus, X is not *SND* since S_1 also satisfies (L1) for X. A contradiction occurs; therefore, Z is *SND*. □

On the basis of Theorem 5.7 and Theorem 5.8, the following two corollaries exhibit the extension of the coterie *union* operation combining more than two known *SND* *k*-coteries to generate new *SND* *k*-coteries.

Corollary 5.1. Let $Z=Z_1\oplus\ldots\oplus Z_i$, where Z_1 is an *SND* k_1-coterie under $U_1,\ldots,Z_i$ is an *SND* k_i-coterie under U_i, and $U_1\cap\ldots\cap U_i=\emptyset$. Then, Z is an *SND* $(k_1+\ldots+k_i)$-coterie under U, where $U=U_1\cup\ldots\cup U_i$. □

Corollary 5.2. Let $Z=Z_1\oplus\ ...\ \oplus Z_i$, where Z_1 is an *SND* 1-coterie under U_1,...,Z_i is an *SND* 1-coterie under U_i, and $U_1\cap ...\cap U_i=\varnothing$. Then, Z is an *SND* i-coterie under U, where $U=U_1\cup ...\cup U_i$. □

5.4.2 Coterie *Join* Operation

The coterie *join* operation, which was first proposed by Neilsen and Mizuno [NM92], provides a way of combining known 1-coteries to construct new, larger 1-coteries. In this subsection, we will show how to derive new k-coteries from known k-coteries and 1-coteries by the coterie *join* operation.

Let U_1 and U_2 be two non-empty sets of sites, $x\in U_1$ and $U_1\cap U_2=\varnothing$. Also, let $U=(U_1-\{x\})\cup U_2$. The coterie *join* operation $\otimes_x$ is defined by

$X\otimes_x Y=\{CT_x(X,Y)|\ X\in X,\ Y\in Y\}$

where X is a family of subsets of U_1, Y is a family of subsets of U_2, and

$$CT_x(X,Y)=\begin{cases}(X-\{x\})\cup Y & \text{if } x\in X \quad \text{(Type1)}\\ X & \text{otherwise} \quad \text{(Type2)}\end{cases}$$

Let X be an 1-coterie under U_1, Y be an 1-coteries under U_2, and $Z=X\otimes_x Y$. Neilsen and Mizuno [NM92] have shown that Z is an 1-coterie under U and also that Z inherits some properties (e.g., nondominance and dominance properties) from X and Y. Below, we discuss the properties of the *join* operation when its first operand and second operand are a k-coterie and an 1-coterie, respectively.

Let X be a k-coterie under U_1, Y be an 1-coterie under U_2, and Z=$X\otimes_x Y$. On the basis of Theorem 3.1 and Theorem 3.3 in [NM92] by Neilsen, we introduce the following Theorem 5.9 and Theorem 5.10 about properties of Z.

Theorem 5.9. Z is a k-coterie under U.

Proof:

First, it is obvious that $Z\neq\emptyset$ and $Z\subseteq U$ for any quorum $Z\in Z$.

Next, we will show that Z satisfies the intersection property; i.e., there exist at most k mutually disjoint quorums in Z. For any $Z_1,...,Z_{k+1}\in Z$, we show that $Z_1,...,Z_{k+1}$ are not pairwise disjoint by considering the following three cases:

(1). $Z_1,...,Z_{k+1}$ are all of type 2; i.e., $Z_1=X_1,...,Z_{k+1}=X_{k+1}$ for certain quorums $X_1,...,X_{k+1}$ $\in$X.

Since X is a k-coterie, there are at most k pairwise disjoint quorums in X. Thus, $X_1,...,X_{k+1}$ are not pairwise disjoint. Therefore, $Z_1,...,Z_{k+1}$ are not pairwise disjoint.

(2). One of $Z_1,...,Z_{k+1}$ is of type 1 and the others are of type 2.

Without loss of generality, we let $Z_i=X_i$, where $1\leq i\leq k$, $X_i\in$X and $x\notin X_i$, and let $Z_{k+1}=(X_{k+1}-\{x\})\cup Y$, where $X_{k+1}\in$X, $x\in X_{k+1}$ and $Y\in$Y. Since X is a k-coterie, there are at most k pairwise disjoint quorums in X. Thus, $X_1,...,X_{k+1}$ are not pairwise disjoint. Since $x\notin X_i$, for $1\leq i\leq k$, and $x\in X_{k+1}$, we have that x will not be in the intersection of any pair of quorums among $X_1,...,X_{k+1}$. Thus, $X_1,...,X_k$ and $(X_{k+1}-\{x\})$ are not pairwise disjoint. So, $X_1,...,X_k$ and $(X_{k+1}-\{x\})\cup Y$ are not pairwise disjoint. Hence, $Z_1,...,Z_{k+1}$ are not pairwise disjoint.

(3). More than one quorum of $Z_1,...,Z_{k+1}$ is of type 1 and the others are of type 2.

Without lost of generality, we let $Z_1=(X_1-\{x\})\cup Y_1$, where $X_1\in$X and $Y_1\in$Y, and let $Z_2=(X_2-\{x\})\cup Y_2$, where $X_2\in$ X and $Y_2\in$ Y (note that we leave $Z_3,...,Z_{k+1}$ unspecified). Since Y is a coterie, Y_1 and Y_2 are not disjoint. So, Z_1 and Z_2 are not disjoint. Hence, $Z_1,...,Z_{k+1}$ are not pairwise disjoint.

Next, we will show that Z satisfies the non-intersection property. Let $Z_1,...,Z_h$, $h<k$, be any pairwise disjoint quorums in Z. We show that we can still find a quorum Z_{h+1} in Z such that $Z_1,...,Z_{h+1}$ are pairwise disjoint. Note that any pair of type 1 quorums are not disjoint because every type 1 quorum contains a quorum of Y, and no two quorums of Y are disjoint. Thus, for pairwise disjoint quorums $Z_1,...,Z_{h+1}$, we only have to consider the following two cases:

(1). All of $Z_1,...,Z_h$ are of type 2; i.e., $Z_i=X_i$, $1\leq i\leq h$, for some quorum $X_i\in X$.

Since X is a k-coterie, we can find a quorum X_{h+1} such that $X_1,...,X_{h+1}$ are pairwise disjoint. If $x\in X_{h+1}$, then we let $Z_{h+1}=(X_{h+1}-\{x\})\cup Y$ for some quorum Y in Y. Then $Z_{h+1}\in Z$. Since $X_1,...,X_{h+1}\subseteq U_1$, $Y\subseteq U_2$, $U_1\cap U_2=\emptyset$, and $X_1,...,X_{h+1}$ are pairwise disjoint, $Z_1,...,Z_{h+1}\in Z$ are pairwise disjoint. On the other hand, if $x\notin X_{h+1}$, we let $Z_{h+1}=X_{h+1}$. Then $Z_{h+1}\in Z$. Since $X_1,...,X_{h+1}$ are pairwise disjoint, $Z_1,...,Z_{h+1}$ ($Z_1,...,Z_{h+1}\in Z$) are pairwise disjoint.

(2). One of $Z_1,...,Z_h$ is of type 1, and the others are of type 2.

Without loss of generality, we let $Z_i=X_i$, where $1\leq i\leq h-1$, $X_i\in X$ and $x\notin X_i$, and let $Z_h=(X_h-\{x\})\cup Y$, where $X_h\in X$, $x\in X_h$ and $Y\in Y$. Since $Z_1,...,Z_h$ are pairwise disjoint, $X_1,...,X_{h-1}$ and $((X_h-\{x\})\cup Y)$ are pairwise disjoint, hence $X_1,...,X_{h-1}$ and $(X_h-\{x\})$ are pairwise disjoint. Thus, $X_1,...,X_h$ are pairwise disjoint since $x\notin X_1,...,\mathrm{x}\notin X_{h-1}$. Since X is a k-coterie, we can find a quorum X_{h+1} in X such that $X_1,...,X_{h+1}$ are pairwise disjoint. Since $x\in X_h$, we have that $x\notin X_{h+1}$ or else $X_1,....,X_{h+1}$ would not be pairwise disjoint. Let $Z_{h+1}=X_{h+1}$. Then $Z_{h+1}\in Z$. Thus, we have that $Z_1,...Z_{h+1}\in Z$ and $Z_1,...,Z_{h+1}$ are pairwise disjoint because $X_1,...,X_{h+1}\subseteq U_1$, $Y\subseteq U_2$, $U_1\cap U_2=\emptyset$ and $X_1,...,X_{h+1}$ are pairwise disjoint.

Finally, we will show that Z satisfies the minimality property. Let Z_1, $Z_2 \in \mathrm{Z}$. We will show that $Z_1 \not\subset Z_2$. There are four cases to consider:

(1). $Z_1=X_1$ and $Z_2=X_2$, where $X_1 \in \mathrm{X}$, $X_2 \in \mathrm{X}$, $x \notin X_1$ and $x \notin X_2$.

Since X is a *k*-coterie, $X_1 \not\subset X_2$, and hence $Z_1 \not\subset Z_2$.

(2). $Z_1=X_1$ and $Z_2=(X_2-\{x\}) \cup Y$, where $X_1 \in \mathrm{X}$, $x \notin X_1$, $X_2 \in \mathrm{X}$, $x \in X_2$ and $Y \in \mathrm{Y}$.

Since X is a *k*-coterie, we have $X_1 \not\subset X_2$. So, there must exists $x' \in U_1$ such that $x' \in X_1$, and $x' \notin X_2$. By $U_1 \cap U_2=\varnothing$, we have $x' \notin Y$. Thus, $x' \notin Z_2$ because $x' \notin X_2$ and $x' \notin Y$. So, $Z_1 \not\subset Z_2$ because $x' \in Z_1 (=X_1)$, but $x' \notin Z_2$.

(3). $Z_1=(X_1-\{x\}) \cup Y$ and $Z_2=X_2$, where $X_1 \in \mathrm{X}$, $x \in X_1$, $X_2 \in \mathrm{X}$, $x \notin X_2$ and $Y \in \mathrm{Y}$.

Assume $Z_1 \subset Z_2$, i.e., $(X_1-\{x\}) \cup Y \subset X_2$. Since $(X_1-\{x\}) \subseteq U_1$, $X_2 \subseteq U_1$, $Y \subseteq U_2$ and $U_1 \cap U_2=\varnothing$, we have $Y=\varnothing$. This is a contradiction because Y is a coterie having non-empty quorums. Therefore, we have $Z_1 \not\subset Z_2$.

(4). $Z_1=(X_1-\{x\}) \cup Y_1$ and $Z_2=(X_2-\{x\}) \cup Y_2$, where $X_1 \in \mathrm{X}$, $Y_1 \in \mathrm{Y}$, $x \in X_1$, $X_2 \in \mathrm{X}$, $Y_2 \in \mathrm{Y}$, $x \in X_2$.

Assume $Z_1 \subset Z_2$; i.e., $((X_1-\{x\}) \cup Y_1) \subset ((X_2-\{x\}) \cup Y_2)$. Since $X_1-\{x\} \subseteq U_1$, $X_2-\{x\} \subseteq U_1$, $Y_1 \subseteq U_2$, $Y_2 \subseteq U_2$, and $U_1 \cap U_2=\varnothing$, we have either (a) $X_1-\{x\} \subset X_2-\{x\}$ or (b) $Y_1 \subset Y_2$. For both cases, we show a contradiction to conclude that $Z_1 \not\subset Z_2$.

(a). $X_1-\{x\} \subset X_2-\{x\}$ means $X_1 \subset X_2$, which contradicts to the minimality property of *k*-coterie *X*.

(b). $Y_1 \subset Y_2$ contradicts to the minimality property of coterie *Y*. □

Theorem 5.9 states that if X is a *k*-coterie and Y is an 1-coterie, then $\mathrm{Z}=X \otimes_x Y$ is a *k*-coterie. For example, let X be a 2-coterie $\{\{a, b\},\{c, d\}, \{a, c\}, \{b, d\}\}$ under $\{a, b, c, d\}$, and Y be an 1-coterie $\{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$ under $\{1, 2, 3\}$, then $Z=X \otimes_a$

Y={{1, 2, b}, {1, 3, b}, {2, 3, b}, {c, d}, {1, 2, c}, {1, 3, c}, {2, 3, c},{b, d}} is a 2-coterie under {b, c, d, 1, 2, 3}. However, if X is an 1-coterie and Y is a k-coterie, then Z may or may not be k-coterie. For example, let X be an 1-coterie { {1, 2}, {1, 3}, {2, 3}} under {1, 2, 3}, and Y be a 2-coterie {{a, b},{c, d}, {a, c}, {b, d}} under {a, b, c, d}, then $Z=X \otimes_3 Y$={{1, 2}, {1, a, b}, {1, c, d}, {1, a, c}, {1, b, d} {1, 3}, {2, a, b}, {2, c, d}, {2, a, c}, {2, b, d}} is not a 2-coterie.

Below, let us discuss the nondominance property of Z in Theorem 5.10.

Theorem 5.10. If X and Y are *SND*, then Z is *SND*.
Proof: (The proof is by contradiction)

Assume that Z is not *SND*; i.e., there exists a set $S\subseteq U$ such that $Z\not\subseteq S$ for any quorum Z in Z, and $Z_1,...,Z_k$ and S are not pairwise disjoint for any k pairwise disjoint quorums $Z_1,...,Z_k$ in Z.

We will consider the relation between S and the quorums in Y. There are two cases to consider: either (1) $\forall Y\in \mathrm{Y}$, $Y\cap S\neq\emptyset$ or (2) $\exists Y\in Y$, $Y\cap S=\emptyset$.

In either case, we show that we can obtain a contradiction.

(1). $\forall Y\in \mathrm{Y}$, $Y\cap S\neq\emptyset$.

Let $S_1=(S\cup\{x\})\cap U_1$ and $X_1,...,X_k$ be any k pairwise disjoint quorums in X. Below, we want to show that $X_1,...,X_k$ and S_1 are not pairwise disjoint. There are two cases to consider: either (a) none of $X_1,...,X_k$ involves x or (b) only one quorum of $X_1,...,X_k$ involves x (note that if more than one quorums of $X_1,...,X_k$ involves x, then $X_1,...,X_k$ would not be pairwise disjoint).

(a). None of $X_1,...,X_k$ involves x.

Since $x \notin X_1, \ldots, x \notin X_k$, we have $X_1, \ldots, X_k \in Z$. So, $X_1, \ldots, X_k$ and S are not pairwise disjoint. Hence, $X_1, \ldots, X_k$ and S_1 are not pairwise disjoint.

(b). Only one quorum of $X_1, \ldots, X_k$ involves x.

Without loss of generality, we suppose only X_1 involves x. It is obvious that $X_1, \ldots, X_k$ and S_1 are not pairwise disjoint, for $S_1 \cap X_1 \supseteq \{x\}$.

So, we have shown that $X_1, \ldots, X_k$ and S_1 are not pairwise disjoint for any pairwise disjoint quorums $X_1, \ldots, X_k \in \mathrm{X}$. We conclude that there must exist a quorum $X^* \in X$ such that $X^* \subseteq S_1$ or else S_1 would satisfy both (L1) and (L2), and X would not be *SND*.

Let $S_2 = S \cap U_2$. Then, we have $\forall Y \in Y$, $Y \cap S_2 \neq \varnothing$; hence (L2) holds. Therefore, there must exist a quorum $Y^* \in Y$ such that $Y^* \subseteq S_2$ or else S_2 would satisfy both (L1) and (L2), and Y would not be *SND*.

By now, we have shown that ($\exists X^* \in X$, $X^* \subseteq S_1$) and ($\exists Y^* \in Y$, $Y^* \subseteq S_2$). We further consider the following two cases: (a) $x \in X^*$ or (b) $x \notin X^*$. For case (a), let $Z^* = (X^* - \{x\}) \cup Y^*$ and for case (b), let $Z^* = X^*$. It is obvious that $Z^* \in Z$ and $Z^* \subseteq S$. A contradiction occurs since we assume that $Z \not\subseteq S$ for any quorum Z in Z.

(2). $\exists Y \in Y$, $Y \cap S = \varnothing$.

Let $S_1 = S \cap U_1$ and $X_1, \ldots, X_k$ be any pairwise disjoint quorums in X. We want to show that $X_1, \ldots, X_k$ and S_1 are not pairwise disjoint. There are two cases to consider: either (a) none of $X_1, \ldots, X_k$ involves x or (b) only one quorum of $X_1, \ldots, X_k$ involves x (note that if more than one quorums of $X_1, \ldots, X_k$ involves x, then $X_1, \ldots, X_k$ would not be pairwise disjoint).

(a). None of $X_1, \ldots, X_k$ involves x.

Since $x \notin X_1, \ldots, x \notin X_k$, we have $X_1, \ldots, X_k \in Z$. Therefore, $X_1, \ldots, X_k$ and S are not pairwise disjoint. Hence $X_1, \ldots, X_k$ and S_1 are not pairwise disjoint.

(b). Only one quorum of $X_1,...,X_k$ involves x.

Without loss of generality, suppose $x\in X_1,x\notin X_2,...,x\notin X_k$. Let $Z_1=(X_1-\{x\})\cup Y$ where $Y\in Y$ and $Y\cap S=\emptyset$ (we can find such a Y because we have assumed $\exists Y\in Y$, $Y\cap S=\emptyset$), and let $Z_2=X_2,...,Z_k=X_k$. Then $Z_1,...,Z_k\in Z$. Since $Z_1,...,Z_k$ and S are not pairwise disjoint, $(X_1-\{x\})\cup Y$ and $X_2,...,X_k$ are not pairwise disjoint. Since $Y\cap S=\emptyset$, it follows that $X_1,...,X_k$ and S_1 are not pairwise disjoint.

By now, we have shown that $X_1,...,X_k$ and S_1 are not pairwise disjoint for any pairwise disjoint quorums $X_1,...,X_k\in$X. We conclude that ($\exists X^*\in X$, $X^*\subseteq S_1$) or else S_1 would satisfy (L1) and (L2) for X and X would not be *SND*. Since $S\subseteq U$, $U=(U_1-\{x\})\cup U_2$ and $S_1=S\cap U_1$, we have $x\notin S_1$. It follows that $x\notin X^*$ because $x\notin S_1$ and $X^*\subseteq S_1$. Let $Z^*=X^*$. Then $Z^*\in$Z. Since $Z^*=X^*$, $X^*\subseteq S_1$ and $S_1\subseteq S$ (by $S_1=S\cap U_1$), it follows that $Z^*\subseteq S$. A contradiction occurs since we assume that $Z\not\subseteq S$ for any quorum Z in Z.

Therefore, we have shown that a contradiction occurs for both cases of (1) $\forall Y\in$Y, $Y\cap S\neq\emptyset$ and (2) $\exists Y\in Y$, $Y\cap S=\emptyset$. Hence, Z is *SND*. □

5.5 Concluding remarks

K-coteries can be used to develop *k*-mutual exclusion algorithms that are resilient to site and/or communication link failures. A *k*-coterie is superior to any *k*-coterie it dominates; thus, we should always concentrate on the *ND* *k*-coteries that no *k*-coterie can dominate. In this paper, we have introduced a theorem for examining the nondominance of *k*-coteries, and define a special type of *ND* *k*-coteries—*SND* *k*-coteries. We have also shown that the *k*-singleton coterie is *SND* and that the *k*-majority coterie is *SND* for some special cases. Further, we have shown that every

ND 1-coterie and every *ND* 2-coterie are *SND*. However, the problem of whether there every *ND* *k*-coterie, *k*>2, is *SND* remains open.

We have also proposed two operations, *union* and *join*, by which we can generate new *SND* *k*-coteries form known *SND* *k*-coteries, such as the *k*-singleton coterie [KFYA93], the *k*-majority coterie [KFYA93], the tree coterie [AE91], the composite coterie [NM92], the level coterie [SW93a], the Lovasz coterie [Nei93], and so forth. It is obvious that by mixing and repeating *union* and *join* operations, we can generate a large number of *SND* *k*-coteries.

Chapter 6
Conclusion and future work

6.1 Conclusion

This chapter concludes our research on constructing novel quorum structures—coteries, *wr*-coteries and *k*-coteries—that are nondominated (*ND*) and/or of constant quorum size. The constructing methods survive network partitioning and can easily be extended to solve the problems of distributed mutual exclusion, replicated data consistency or distributed *k*-mutual exclusion. The nondominance property of the quorum structures is favorable since nondominated quorum structures are candidates to achieve optimal availability, the probability that a quorum can be formed in an error prone environment. On the other hand, constant quorum size of the quorum structures is preferable because when those quorum-constructing methods are applied to solve the problems mentioned, the message cost are directly proportional to the quorum size.

In Chapter 2, we have devised a method to construct quorums of an *ND* coterie; the method can easily be extended to be a solution to distributed mutual exclusion. The method utilizes a logical structure named *Cohorts* to construct quorums of constant size in the best case. When some sites are inaccessible, the quorum size increases gradually and may be as large as O(*n*), where *n* is the number of sites.

However, the expected quorum size has been shown to remain constant as *n* grows. In addition, the availability of the constructed quorum has been shown to be asymptotically high. With the two properties—constant expected quorum size and asymptotically high availability, the proposed method is thus applicable to systems possessing an increasing number of sites. We have also analyzed and compared the constructed quorums with others in terms of availability and quorum size.

In Chapter 3, we have devised a method to construct *ND wr*-coteries; the method can easily be extended for maintaining replicated data consistency. The method utilizes a logical structure named *Cohorts* to construct quorums of constant size in the best case. When some replicas are inaccessible, the quorum size increases gradually and may be as large as O(*n*), where *n* is the number of replicas. However, the expected quorum size has been shown to remain constant as *n* grows. In addition, the availability of the constructed quorums has been shown to be asymptotically high. With the two properties—constant expected quorum size and asymptotically high availability, the proposed solution is thus applicable to systems possessing an increasing number of replicas. We have also analyzed and compared the constructed quorums with others in terms of availability and quorum size.

In Chapter 4, we have devised a method to construct *k*-coteries; the method can easily be extended to be a solution to distributed *k*-mutual exclusion. The solution utilizes a logical structure named *Cohorts* to construct quorums of constant size in the best case. When some sites are inaccessible, the quorum size increases gradually and may be as large as O(*n*), where *n* is the number of sites. However, the expected quorum size has been shown to remain constant as *n* grows. We have also analyzed the availability of the constructed quorums and found that the availability of the constructed quorums is comparably high in comparison with those of relevant ones.

In Chapter 5, we have proposed a theorem for checking the nondominance of *k*-coteries. We have also defined a special type of *ND k*-coteries—*strongly nondominated* (*SND*) *k*-coteries, and proposed two operations (methods)—*union* and *join*—for generating new *SND k*-coteries from known *SND k*-coteries. We have further shown that every *ND* 1-coterie (i.e., coterie) and every *ND* 2-coterie are *SND*. Thus, known *ND* 1-coteries and *ND* 2-coteries can be directly applied to the *union* operation or the *join* operation to generate new *SND k*-coteries. We have also shown that the *k*-singleton coterie is *SND* and that under some special conditions, the *k*-majority coterie is *SND* as well.

6.2 Future work

Since the problem of whether any *ND k*-coterie, *k*>2, is *SND* remains open. We would like to contribute ourselves to this problem in the future. This may ends up as two cases: either we show that any *ND k*-coterie is *SND* or we show that there is an *ND k*-coterie that is not *SND*.

The quorums generated by the methods proposed in Chapters 2 and 3 have been shown to have asymptotically high availability. However, more work is needed to accomplish the asymptotic availability analysis for the quorums generated by the method proposed in Chapter 4. Moreover, we would like to analyze the asymptotic availability for other related methods, such as the tree quorum algorithm and the majority quorum algorithm, etc. Thus, we may have a comparison of our constructing methods and those quorum-based algorithms on the aspect of asymptotic availability.

In addition to the applications of quorum structures on solving distributed mutual exclusion, replicated data control and distributed *k*-mutual exclusion, quorum structures can also be applied to solve many other problems, such as those of distributed atomic commitment [AE91, Ske82], replicated data security [MN91] and distributed consensus [NM91], etc. We would also like to concentrate ourselves on finding new applications of quorum structures in the future.

Bibliography

[AA89]

M. Ahamad and M. H. Ammar, "Performance characterization of quorum-consensus algorithms for replicated data," *IEEE Trans. on Software Engineering*, vol. 15, no. 4, pp. 492-496, April 1989.

[AE90]

D. Agrawala and A. El Abbadi, "Exploiting logical structures in replicated databases," *Inf. Process. Lett.*, vol. 33, no. 5, pp. 255-260, Jan. 1990.

[AE91]

D. Agrawala and A. El Abbadi, "An efficient and fault-tolerant solution for distributed mutual exclusion," *ACM Trans. Comp. Syst*., vol. 9, no. 1, pp. 1-20, Feb. 1991.

[AE92a]

D. Agrawala and A. El Abbadi, "The generalized tree protocol: an efficient approach for managing replicated data," *ACM Trans. Database. Syst.*, vol. 17, no. 4, pp. 689-717, Dec. 1992.

[AE92b]

D. Agrawala and A. El Abbadi, "Resilient logical structures for efficient management of replicated data," in *Proc. of the 18th VLDB Conf.*, Canada, pp. 151-162. 1992.

[AJ92]

G. Agrawal and P. Jalote, "An efficient protocol for voting in distributed systems," in *Proc. 12th IEEE Internat. Conf. on Dist. Comput. Systems*, pp. 640-647, May 1992.

[BG84]

P. A. Bernstein and N. Goodman, "An algorithm for concurrency control and recovery in replicated distributed databases," *ACM Trans. on Database Syst.*, vol. 9, no. 4, pp. 596-615, 1984.

[BG86]

D. Barbara and H. Garcia-Molina, "The vulnerability of vote assignments," *ACM Trans. Comp. Syst*., vol. 4, no. 3, pp. 187-213, 1986.

[BG87]

D. Barbara and H. Garcia-Molina, "The reliability of voting mechanisms," *IEEE Trans. on Computers*, vol. C-36, no. 10, pp. 1197-1208, Oct. 1987.

[BHG87]

P. A. Bernstein, V. Hadzilacos and N. Goodman, *Concurrency control and recovery in database Systems*, Addison-Wesley, 1987.

[CAA92]

S. Y. Cheung, M. H. Ammar and M. Ahamad, "The grid protocol: a high performance scheme for maintaining replicated data," *IEEE Trans. Knowledge and Data Engineering*, vol. 4, no. 6, pp. 582-592, Dec. 1992.

[Chu94]

S. M. Chung, "Enhanced tree quorum algorithm for replica control in distributed database systems", *Data and Knowledge Engineering*, vol. 12, pp. 63-81, 1994.

[CY94a]

H.-K. Chang and S.-M. Yuan, "Message complexity of hierarchical quorum consensus algorithm," *Inf. Process. Lett.*, vol. 50, pp. 69-73, 1994.

[CY94b]

H.-K. Chang and S.-M. Yuan, "Message complexity of the tree quorum algorithm," in *Proc. 14th IEEE Internat. Conf. on Dist. Comput. Systems*, 1994.

[DGS85]

S. B. Davidson, H. Garcia-Molina, and D. Skeen, "Consistency in partitioned networks," *ACM Comput. Surv*., vol. 17, no. 3, pp. 341-370, Sept. 1985.

[DOSE86]

J. A. Dossey, A. D. Otto, L. E. Spence and C. V. Eynden, *Discrete Mathematics*, Scott, Foresman and Company, 1986.

[FYA91]

S. Fujita, M. Yamashita and T. Ae, "Distributed *k*-mutual exclusion problem and *k*-coteries," in *Proc. 2nd Internat. Symp. on Algorithms*, Lecture Notes in Computer Science 557, Springer, Berlin, pp. 22-31, 1991.

[GB85]

H. Garcia-Molina and D. Barbara, "How to assign votes in a distributed system," *JACM*., vol. 32, no. 4, pp. 841-860, Oct. 1985.

[Gif79]

D. K. Gifford, "Weighted voting for replicated data," in *Proc. 7th ACM*

SIGOPS Symp. Oper. Syst. Principles, Pacific Grove, CA, pp. 150-159, Dec. 1979.

[HJK93]
S. T. Huang, J. R. Jiang and Y. C. Kuo, "*K*-coteries for fault-tolerant *k* entries to a critical section," in *Proc. 13th IEEE Internat. Conf. on Dist. Comput. Systems*, pp. 74-81, May 1993.

[IK93]
T. Ibaraki and T. Kameda, "A theory of coteries: mutual exclusion in distributed systems," *IEEE Trans. Parall. and Distrib. Syst.*, vol. 4, no. 7, pp. 779-794, July 1993.

[Jia95]
J.-R. Jiang, "Fault-tolerant distributed mutual exclusion with O(1) message overhead," in *Proceedings of the 12th IASTED International Conference on Applied Informatics*, pp. 228-231, Feb. 1995.

[JH94]
J.-R. Jiang and S.-T. Huang, "Obtaining nondominated *k*-coteries for fault-tolerant distributed *k*-mutual exclusion," in *Proceedings of 1994 International Conference on Parallel and Distributed Systems*, HsinChu, Taiwan, pp. 582-587, Dec. 1994.

[KC91]
A. Kumar and S. Y. Cheung, "A high availability $\sqrt{n}$ hierarchical grid protocol for replicated data," *Inf. Process. Lett.*, vol. 40, no. 6, pp. 311-316, Dec. 1991.

[KFYA93]
H. Kakugawa, S. Fujita, M. Yamashita, and T. Ae, "Availability of *k*-coterie," *IEEE Trans. Comput.*, vol. 42, no. 5, pp. 553-558, May 1993.

[KFYA94]
H. Kakugawa, S. Fujita, M. Yamashita, and T. Ae, "A distributed *k*-mutual exclusion algorithm using *k*-coterie," *Inf. Process. Lett.*, vol. 49, pp. 213-218, Mar. 1994.

[KRS92]
A. Kumar, M. Rabinovich and R. K. Sinha, "A performance study of a new grid protocol and general grid structures for replicated data," Technical Report, Cornell University, 1992.

[KRS93]
A. Kumar, M. Rabinovich and R. K. Sinha, "A performance study of general

grid structures for replicated data," in *Proc. 13th IEEE Internat. Conf. on Dist. Comput. Systems*, pp. 178-185, May 1993.

[Kum91]
A. Kumar, "Hierarchical quorum consensus: A new algorithm for managing replicated data," *IEEE Trans. Comp.*, vol. 40, no. 9, pp. 996-1004, Sep. 1991.

[LG94]
T. V. Lakshman and D. Ghosal, "Performance evaluation of an efficient multiple copy update algorithm," *IEEE Trans. on Parall. and Distrib. Syst.*, vol. 5, no. 2, pp. 217-224, Feb. 1994.

[Lip79]
S. Lipschutz, *Theory and problems of set theory and related topics,* Schaum Publishing Company, 1979.

[Mae85]
M. Maekawa, "A $\sqrt{n}$ algorithm for mutual exclusion in decentralized systems," *ACM Trans. Comput. Syst.*, vol. 3, no. 2, pp. 145-159, May 1985.

[MN91]
M. Mizuno and M. L. Neilsen, "A secure quorum protocol," in *National Computer Security Conference,* pp. 514-523, 1991.

[Nai93]
M. Naimi, "Distributed algorithm for *k*-entries to critical section based on the directed graphs," *Operating Systems Review*, vol. 27, no. 4, pp. 67-75, Oct. 1993.

[Nei92]
M. L. Neilsen, "Quorum structures in distributed systems," Ph. D. Thesis, Kansas State University, May 1992.

[Nei93]
M. L. Neilsen, "Measures of importance and symmetry in distributed systems," in *5th IEEE Symposium on Parallel and Distributed Computing,* Dallas, TX, Dec. 1993.

[NM91]
M. L. Neilsen and M. Mizuno, "Decentralized consensus protocol," in in *Proc. of 10th International Phoenix Conf. on Computers and Communications*, pp. 257-262, 1991.

[NM92]

M. L. Neilsen and M. Mizuno, "Coterie join algorithm," *IEEE Trans. on Parall. and Distrib. Syst.*, vol. 3, no. 5, pp. 759-765, Sept. 1992.

[NM94]

M. L. Neilsen and M. Mizuno, "Nondominated *k*-coteries for multiple mutual exclusion," *Inf. Process. Lett*., vol. 50, no. 5, pp. 247-252, June 1994.

[Ray89]

K. Raymond, "A distributed algorithm for multiple entries to a critical section," *Inf. Process. Lett*., vol. 30, no. 4, pp. 189-193, Feb. 1989.

[RJT93]

S. Rangarajan, P. Jalote and S. K. Tripathi, "Capacity of Voting Systems," in *IEEE Trans. on Soft. Engineering*, vol. 19, no. 7, pp. 698-706, July 1993.

[RST92]

S. Rangarajan, S. Setia and S. K. Tripathi, "A Fault-Tolerant Algorithm for Replicated Data Management," in *Proc. 1992 Internat. Conf. on Data Engineering*, pp. 230-237, 1992.

[San87]

Sanders, B. A. "The information structure of distributed mutual exclusion algorithms," *ACM Trans. Comput. Syst.*, pp. 284-299, Apr. 1987.

[Ske82]

D. Skeen, "A quorum based commit protocol," in *Proc. 6th Berkeley Workshop on Distributed Data Management and Computer Networks*, pp. 69-80. Feb, 1982.

[SR92]

P. K. Srimani and R. L. N. Reddy, "Another distributed algorithm for multiple entries to a critical section", *Inf. Process. Lett*., vol. 41, no. 1, pp. 51-57, Jan. 1992.

[SW93a]

D. Shou and S. D. Wang, "An efficient quorum generating approach for distributed mutual exclusion," *Journal of Information Science and Engineering*, vol. 9, pp. 201-227, 1993.

[SW93b]

D. Shou and S. D. Wang, "A new transformation method for nondominated coterie design," *Information Sciences*, pp. 223-246, 1993.

[Tho79]

R. H. Thomas, "A majority consensus approach to concurrency control," *ACM Trans. Database Syst*., vol. 4, no. 2, pp. 180-209, June 1979.

[WB92]

G. Wu and G. G. Belford, "The Triangular Lattice Protocol: A Highly Fault Tolerant and Highly Efficient Protocol for Replicated Data," in *11th Symp. on Reliable Distributed Systems*, 1992.

[Wu93a]

C. Wu, "A fault tolerant O($\sqrt{n}$) algorithm for distributed mutual exclusion," in *Proc. of 1993 International Phoenix Conf. on Computers and Communications*, pp. 175-180, 1993.

[Wu93b]

C. Wu, "Achieving high performance and fault tolerant for distributed mutual exclusion," Technical Report, University of Illinois at Urbana-Champaign, 1993.