Coterie Join Operation and Tree Structured *k*-Coteries

Takashi Harada, *Member*, *IEEE Computer Society*, and
Masafumi Yamashita, *Member*, *IEEE Computer Society*

Abstract—The coterie join operation proposed by Neilsen and Mizuno produces, from a *k*-coterie and a coterie, a new *k*-coterie. For the coterie join operation, this paper first shows 1) a necessary and sufficient condition to produce a nondominated *k*-coterie (more accurately, a nondominated *k*-semicoterie satisfying Nonintersection Property) and 2) a sufficient condition to produce a *k*-coterie with higher availability. By recursively applying the coterie join operation in such a way that the above conditions hold, we define nondominated *k*-coteries, called tree structured *k*-coteries, the availabilities of which are thus expected to be very high. This paper then proposes a new *k*-mutual exclusion algorithm that effectively uses a tree structured *k*-coterie, by extending Agrawal and El Abbadi's tree algorithm. The number of messages necessary for k processes obeying the algorithm to simultaneously enter the critical section is approximately bounded by $k\log(n/k)$ in the best case, where n is the number of processes in the system.

Index Terms—Availability, distributed systems, *k*-coteries, *k*-semicoteries, *k*-mutual exclusion problem, message complexity, nondominatedness, quorums.

1 Introduction

The distributed *k*-mutual exclusion problem is the problem of controlling a distributed system in such a way that at most k processes in the system are granted to be simultaneously in the critical section. The 1-mutual exclusion problem is known as the distributed mutual exclusion problem. By definition, a distributed k'-mutual exclusion algorithm also works as a distributed *k*-mutual exclusion algorithm for all $k \geq k'$ and, hence, any mutual exclusion algorithm can be used as a *k*-mutual exclusion algorithm for all $k \geq 1$ at the risk of decreasing of the level of concurrency and consequently system performance. A main concern in the design of a *k*-mutual exclusion algorithm is to allow k processes to be in the critical section without blocking processes that are not requesting the critical section.

Several *k*-mutual exclusion algorithms have been proposed from this viewpoint (e.g., [11], [14], [21], [22]). In particular, algorithm *k*-MUTEX proposed by Kakugawa et al. [14], which uses a *k*-coterie under the set of processes in the system, is superior to others in its strong descriptive power: A variety of different algorithms, ranging from centralized to fully distributed, are describable using this algorithm by choosing an appropriate *k*-coterie [14].

A *k-coterie* $\mathcal{C}$ under a finite set U is a set of nonempty subsets (called *quorums*) $Q \subseteq U$ of U such that all of the following three conditions hold [8], [14].

1. **Minimality**. For all $P, Q \in \mathcal{C}, P \not\subset Q$.
2. **Intersection Property**. There are k pairwise disjoint quorums in $\mathcal{C}$, but no more than k.
3. **Nonintersection Property**. For any set $\mathcal{D}$ of $h(<k)$ pairwise disjoint quorums in $\mathcal{C}$, there is a set $\mathcal{D}'$ of k pairwise disjoint quorums in $\mathcal{C}$ such that $\mathcal{D}' \supseteq \mathcal{D}$.

A set $\mathcal{C}$ of quorums that holds Minimality and Intersection Properties is called *k-semicoterie* [11].[1] By definition, any 1-semicoterie is a 1-coterie and a 1-coterie (and, hence, a 1-semicoterie) is known as a coterie [10].

A *k*-coterie (respectively, *k*-semicoterie) $\mathcal{C}$ is said to be *nondominated* (ND, for short) if $\mathcal{C}$ is not dominated by any *k*-coterie (respectively, *k*-semicoterie) $\mathcal{D}$, where $\mathcal{D}$ *dominates* $\mathcal{C}$, if $\mathcal{C} \neq \mathcal{D}$ and, for any quorum $P \in \mathcal{C}$, there exists a quorum $Q \in \mathcal{D}$ such that $Q \subseteq P$. It is worth noting the following: Since a *k*-coterie is a *k*-semicoterie, any ND *k*-semicoterie that satisfies Nonintersection Property is an ND *k*-coterie. However, an ND *k*-coterie $\mathcal{C}$ may not be an ND *k*-semicoterie since there may be a *k*-semicoterie $\mathcal{D}$ dominating $\mathcal{C}$ but not satisfying Nonintersection Property.

Algorithm *k*-MUTEX uses a *k*-coterie under the set of processes. A quorum is then a set of processes. Since a process wishing to enter the critical section can actually enter it only when the process has locked a quorum, i.e., locked all processes in a quorum of the *k*-coterie, Intersection Property guarantees *k*-mutual exclusion, i.e., at most k processes can simultaneously be in the critical section. However, it does not imply that a process can always find an unlocked quorum when less than k quorums

- *T. Harada is with the Graduate School of Management, Hiroshima University, 1-1-89 Higashi Senda, Naka-ku, Hiroshima 730-0053 Japan. E-mail: haradat@mgt.hiroshima-u.ac.jp.*
- *M. Yamshita is with the Department of Computer Science and Communication Engineering, Kyushu University, 6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581 Japan. E-mail: mak@csce.kyushu-u.ac.jp.*

Manuscript received 11 Aug. 1999; revised 1 Jan. 2001; accepted 11 Apr. 2001.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number 110417.

1. The term *k*-coterie was defined in several different ways. Fujita et al. first defined the *k*-coterie [8]. We adopt this definition. The *k*-coterie in [11] corresponds to the *k*-semicoterie in this paper. In [13], Jiang and Huang adopt our definition of *k*-coterie. In [19], Neilson and Mizuno do not prepare different terms, but, in [20], they call a *k*-coterie (in this paper) a proper *k*-coterie and a *k*-semicoterie a *k*-coterie.

have been locked; whether or not there is such an unlocked quorum depends on which quorums have been locked. The Nonintersection Property guarantees its existence. Finally, an ND k-coterie $\mathcal{C}$ is definitely superior to any one it dominates, in terms of availability, i.e., the survivability from process and/or link fail-stop failures. Hence, an efficient method to construct a variety of ND k-coteries is sought.

In spite of the demand, relatively little is known about constructing k-coteries (and k-semicoteries) [2], [8], [13], [16], [19], although there are many methods for constructing coteries (see, e.g., [1], [4], [5], [9], [10], [12], [15], [17], [18]). Fujita et al. gave some primitive methods Div and Maj. They also proposed a recursive method based on the grid coterie, but it may create dominated k-coteries [8]. Agrawal et al. recently discussed generalizations of Div and Maj [2].

Neilsen and Mizuno [18] proposed an operation, called the *coterie join operation*, that produces a coterie $\mathcal{D}$ by joining two coteries, $\mathcal{C}_1$ and $\mathcal{C}_2$, and showed that $\mathcal{D}$ is ND if and only if both of $\mathcal{C}_1$ and $\mathcal{C}_2$ are ND. Jiang and Huang [13] then observed that, given a k-coterie $\mathcal{C}_1$ and a coterie $\mathcal{C}_2$, the operation produces a new k-coterie $\mathcal{D}$ and showed a sufficient condition for product $\mathcal{D}$ to be ND. This paper constructs, by using the coterie join operation as a primitive tool, a method for producing a variety of ND k-coteries.

We first show a necessary and sufficient condition for the coterie join operation to produce an ND k-semicoterie with Nonintersection Property. This condition is also sufficient to produce an ND k-coterie since every ND k-semicoterie with Nonintersection Property is an ND k-coterie.

We next show a sufficient condition for the coterie join operation to produce an ND k-coterie whose availability is higher than input. By repeatedly applying the coterie join operation in such a way that the sufficiency holds, we define ND k-coteries, called *tree structured k-coteries*, whose availabilities are expected to be very high.

We finally propose a new k-mutual exclusion algorithm that effectively uses a tree structured k-coterie. A tree structured k-coterie is regarded as an extension of a tree coterie [1]. Agrawal and El Abbadi's mutual exclusion algorithm that uses a tree coterie achieves a low message complexity [1], [6], [23]. The number of messages necessary for a process to enter the critical section is bounded by $\log n$ in the *best* case, where n is the number of processes in the system. Our algorithm is an extension of theirs and the number of messages necessary for k processes obeying the algorithm to simultaneously enter the critical section is approximately bounded by $k \log(n/k)$ in the *best* case.

The rest of this paper is organized as follows: Section 2 gives a necessary and sufficient condition for the coterie join operation to produce an ND k-semicoterie with Nonintersection Property and discusses the availability. We introduce tree structured k-coteries and show their properties in Section 3. The new k-mutual exclusion algorithm using a tree structured k-coterie is described in Section 4. Section 5 concludes the paper by giving some remarks.

2 The Coterie Join Operation

Following the definition of the coterie join operation, we first characterize when it produces an ND k-semicoterie with Nonintersection Property and then investigate conditions for the operation to produce a k-coterie with high availability. The coterie join operation defined below was first introduced by Neilsen and Mizuno to construct a coterie [18]. Then, Jiang and Huang observed that it generally produces a k-coterie, given a k-coterie and a coterie [13]. For a k-semicoterie $\mathcal{C}$, let $\cup\mathcal{C}$ denote $\cup_{Q\in\mathcal{C}} Q$.

Definition 1. *Let U be a finite set, $\mathcal{C}$ be a k-semicoterie under U, $\mathcal{D}$ be a coterie under U, and u be an element in $\cup\mathcal{C}$. Assume that $\cup\mathcal{C} \cap \cup\mathcal{D} \subseteq \{u\}$ holds. Then,* the coterie join operation *for inputs $\mathcal{C}$ and $\mathcal{D}$ produces a quorum set $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ defined by*

$$\mathcal{J}_u(\mathcal{C},\mathcal{D}) = \{R \mid R = (P - \{u\}) \cup Q, P \in \mathcal{C}, Q \in \mathcal{D} \text{ and } u \in P\} \cup \{R \mid R = P, P \in \mathcal{C} \text{ and } u \notin P\}.$$

Example 1. Let $U = \{1,2,3,4,5,6\}$. Consider a 2-semicoterie

$$\mathcal{C} = \{\{1,2\},\{3,4\},\{1,3\},\{2,4\}\}$$

and a coterie $\mathcal{D} = \{\{4,5\},\{4,6\}\}$ under U. Observe that $\cup\mathcal{C} \cap \cup\mathcal{D} \subseteq \{4\}$ holds. Then, $\mathcal{J}_4(\mathcal{C},\mathcal{D})$ is defined and

$$\mathcal{J}_4(\mathcal{C},\mathcal{D}) = \{\{1,2\},\{1,3\},\{2,4,5\},\{2,4,6\},\{3,4,5\},\{3,4,6\}\}.$$

Observe that $\mathcal{J}_4(\mathcal{C},\mathcal{D})$ is a 2-semicoterie.

As mentioned, Jiang and Huang [13, Theorem 9] showed that if $\mathcal{C}$ is a k-coterie and $\mathcal{D}$ is a coterie, then $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is a k-coterie, the proof of which implies the following theorem.

Theorem 1 [13]. *Let U be a finite set and assume that, for a k-semicoterie $\mathcal{C}$ under U, a coterie $\mathcal{D}$ under U, and an element $u \in U$, the coterie join operation $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is defined. Then, $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is a k-semicoterie.*

2.1 Constructing ND *k*-Semicoteries with Nonintersection Property

Let us start with the following theorem:

Theorem 2 [19], [20]. *Let $\mathcal{C}$ be a k-semicoterie under a finite set U. $\mathcal{C}$ is dominated if and only if there exists a set $S \subseteq U$ such that*

1. *$Q \not\subseteq S$ for any $Q \in \mathcal{C}$ and*
2. *For any k pairwise disjoint quorums*

$$Q_1, Q_2, \ldots, Q_k \in \mathcal{C},$$

there exists an i such that $Q_i \cap S \neq \emptyset$.

Let U, $\mathcal{C}$, $\mathcal{D}$, and u be a finite set, a k-semicoterie under U, a coterie under U, and an element in $\cup\mathcal{C}$, respectively. In the rest of this section, we assume that $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is defined, i.e., $\cup\mathcal{C} \cap \cup\mathcal{D} \subseteq \{u\}$. Then, $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is a k-semicoterie by Theorem 1.

This section shows the following theorem, the only if part of which is due to Jiang and Huang [13, Theorem 10]. Thus, we only prove the "if" part.

Theorem 3. *Both of $\mathcal{C}$ and $\mathcal{D}$ are ND if and only if $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is ND.*

Proof of the If part. We show that if either $\mathcal{C}$ or $\mathcal{D}$ is dominated, so is $\mathcal{J}_u(\mathcal{C},\mathcal{D})$.

1. Assume first that $\mathcal{C}$ is dominated. By Theorem 2, there exists an $S_\mathcal{C} \subseteq \cup\mathcal{C}$ such that:

   1.1. $P \not\subseteq S_\mathcal{C}$ for any $P \in \mathcal{C}$ and
   1.2. For any k pairwise disjoint quorums $P_1, P_2, \ldots, P_k \in \mathcal{C}$, there exists an i such that $P_i \cap S_\mathcal{C} \neq \emptyset$.

   There are two cases to consider. Suppose first that $u \notin S_\mathcal{C}$. For any $R \in \mathcal{J}_u(\mathcal{C},\mathcal{D})$, we first show that $R \not\subseteq S_\mathcal{C}$. Without loss of generality, we may assume that $R \notin \mathcal{C}$ by the definition of $S_\mathcal{C}$. Then, $R = (P - \{u\}) \cup Q$ for some $P \in \mathcal{C}$ and $Q \in \mathcal{D}$. Since $u \notin S_\mathcal{C}$, $Q \cap S_\mathcal{C} = \emptyset$ and hence $R \not\subseteq S_\mathcal{C}$.

   Arbitrarily choose k pairwise disjoint quorums $R_1, R_2, \ldots, R_k \in \mathcal{J}_u(\mathcal{C},\mathcal{D})$. We next show that there is an i such that $R_i \cap S_\mathcal{C} \neq \emptyset$. For any $1 \leq i \leq k$, let $P_i \in \mathcal{C}$ be the quorum from which R_i is constructed, i.e., either $u \notin P_i$ and $R_i = P_i$ or $u \in P_i$ and $R_i = (P_i - \{u\}) \cup Q_i$ for some $Q_i \in \mathcal{D}$.

   If $P_i = R_i$ for all $1 \leq i \leq k$, then $R_i \cap S_\mathcal{C} \neq \emptyset$ for some $1 \leq i \leq k$. Observe that $P_i \neq R_i$ implies $Q_i \subseteq R_i$. Hence, there is at most one i in $1 \leq i \leq k$ such that $P_i \neq R_i$ since $\mathcal{D}$ is a coterie. Suppose that there is exactly one i such that $P_i \neq R_i$. For any $j \neq i$, $R_i \cap R_j = \emptyset$ implies $P_i \cap P_j = \emptyset$, since $u \notin P_j$ and $\cup\mathcal{C} \cap \cup\mathcal{D} \subseteq \{u\}$. That is, P_is are pairwisely disjoint. Hence, by the definition of $S_\mathcal{C}$, it follows that $R_i \cap S_\mathcal{C} \neq \emptyset$ for some $1 \leq i \leq k$. Thus, by Theorem 2, $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is dominated.

   Suppose next that $u \in S_\mathcal{C}$. Let

   $$S^* = (S_\mathcal{C} - \{u\}) \cup Q$$

   for some $Q \in \mathcal{D}$. For any $R \in \mathcal{J}_u(\mathcal{C},\mathcal{D})$, we first show that $R \not\subseteq S^*$. Without loss of generality, we may assume that $R \notin \mathcal{C}$. Then, $R = (P - \{u\}) \cup Q$ for some $P \in \mathcal{C}$ and $Q \in \mathcal{D}$. Since $P \not\subseteq S_\mathcal{C}$, there exists $v \in P$ such that $v \neq u$ and $v \notin S_\mathcal{C}$ and, hence, $R \not\subseteq S^*$ follows.

   Arbitrarily choose k pairwise disjoint quorums $R_1, R_2, \ldots, R_k \in \mathcal{J}_u(\mathcal{C},\mathcal{D})$. We next show that there is an i such that $R_i \cap S^* \neq \emptyset$. For any $1 \leq i \leq k$, let $P_i \in \mathcal{C}$ be the quorum from which R_i is constructed, i.e., either $u \notin P_i$ and $R_i = P_i$ or $u \in P_i$ and $R_i = (P_i - \{u\}) \cup Q_i$ for some $Q_i \in \mathcal{D}$.

   If $P_i = R_i$ for all $1 \leq i \leq k$, then $R_i \cap S_\mathcal{C} \neq \emptyset$ for some $1 \leq i \leq k$. Since $u \notin R_i$ for any $1 \leq i \leq k$, it follows that $R_i \cap S^* \neq \emptyset$ for some $1 \leq i \leq k$. Otherwise, if there is an i in $1 \leq i \leq k$ such that $P_i \neq R_i$, then it follows that $R_i \cap S^* \neq \emptyset$ since $\mathcal{D}$ is a coterie. Thus, $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is dominated.
2. Assume that $\mathcal{D}$ is dominated. By Theorem 2, there exists $S_\mathcal{D} \subseteq \cup\mathcal{D}$ such that $Q \not\subseteq S_\mathcal{D}$ and $Q \cap S_\mathcal{D} \neq \emptyset$ hold for all $Q \in \mathcal{D}$.

   Let $S^* = (P^* - \{u\}) \cup S_\mathcal{D}$ for some $P^* \in \mathcal{C}$ such that $u \in P^*$. For any $R \in \mathcal{J}_u(\mathcal{C},\mathcal{D})$, we first show that $R \not\subseteq S^*$. Without loss of generality, we can assume that $R \cap \cup\mathcal{D} \neq \emptyset$, i.e., there exist $P \in \mathcal{C}$ and $Q \in \mathcal{D}$ such that $R = (P - \{u\}) \cup Q$. Since $Q \not\subseteq S_\mathcal{D}$, $R \not\subseteq S^*$.

   Arbitrarily choose k pairwise disjoint quorums $R_1, R_2, \ldots, R_k \in \mathcal{J}_u(\mathcal{C},\mathcal{D})$. We next show that there is an i such that $R_i \cap S^* \neq \emptyset$. For any $1 \leq i \leq k$, let $P_i \in \mathcal{C}$ be the quorum from which R_i is constructed, i.e., either $u \notin P_i$ and $R_i = P_i$ or $u \in P_i$ and $R_i = (P_i - \{u\}) \cup Q_i$ for some $Q_i \in \mathcal{D}$. If $P_i = R_i$ for all $1 \leq i \leq k$, by the Intersection Property of $\mathcal{C}$, $P^* \cap P_i \neq \emptyset$ for some $1 \leq i \leq k$. Otherwise, if there is an i in $1 \leq i \leq k$ such that $P_i \neq R_i$, then $Q_i \cap S_\mathcal{D} \neq \emptyset$. Thus, by Theorem 2, $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is dominated. □

The following theorem characterizes when *k*-coteries are produced by the coterie join operation. Again the only if part is due to Jiang and Huang [13, Theorem 9]. We therefore concentrate on the "if" part.

Theorem 4. *$\mathcal{C}$ has Nonintersection Property if and only if $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ has Nonintersection Property.*

Proof of the If part. Assume that Nonintersection Property does not hold for $\mathcal{C}$ and show that Nonintersection Property does not hold for $\mathcal{J}_u(\mathcal{C},\mathcal{D})$, either. By definition, there are h $(1 \leq h < k)$ pairwise disjoint quorums $P_1, P_2, \ldots, P_h \in \mathcal{C}$ such that, for any

$$P \in \mathcal{C} - \{P_1, P_2, \ldots, P_h\},$$

$P \cap P_i \neq \emptyset$ holds for some $1 \leq i \leq h$.

If $u \notin P_i$ for all $1 \leq i \leq h$, all P_is are quorums in $\mathcal{J}_u(\mathcal{C},\mathcal{D})$. We show that there is no quorum

$$R \in \mathcal{J}_u(\mathcal{C},\mathcal{D}) - \{P_1, P_2, \ldots, P_h\}$$

such that $R \cap P_i = \emptyset$ for all $1 \leq i \leq h$. Arbitrarily select $R \in \mathcal{J}_u(\mathcal{C},\mathcal{D}) - \{P_1, P_2, \ldots, P_h\}$. If $R \in \mathcal{C}$, there is an i such that $R \cap P_i \neq \emptyset$ by the assumption. If $R \notin \mathcal{C}$, then $R = (P - \{u\}) \cup Q$ for some $P \in \mathcal{C}$ and $Q \in \mathcal{D}$. Since $u \notin P_i$ for all $1 \leq i \leq h$, there is an i such that

$$((P - \{u\}) \cup Q) \cap P_i \neq \emptyset.$$

We may assume that $u \in P_i$ for some $1 \leq i \leq h$. Since P_is are pairwise disjoint, no two P_is contain u. Without loss of generality, we assume that $u \notin P_i$ for all

$$1 \leq i \leq h - 1$$

and that $u \in P_h$. Let $R_i = P_i$ for $1 \leq i \leq h-1$ and

$$R_h = (P_h - \{u\}) \cup Q_h$$

for some $Q_h \in \mathcal{D}$. Then, R_is $(1 \leq i \leq h)$ are pairwise disjoint and are all in $\mathcal{J}_u(\mathcal{C},\mathcal{D})$. We show that there is no quorum

$$R \in \mathcal{J}_u(\mathcal{C},\mathcal{D}) - \{R_1, R_2, \ldots, R_h\}$$

such that $R \cap R_i = \emptyset$ for all $1 \leq i \leq h$. Arbitrarily select

$$R \in \mathcal{J}_u(\mathcal{C},\mathcal{D}) - \{R_1, R_2, \ldots, R_h\}.$$

If $R = (P - \{u\}) \cup Q$ for some $P \in \mathcal{C}$ and $Q \in \mathcal{D}$, then $R \cap R_h \neq \emptyset$ because $\mathcal{D}$ is a coterie. If $R = P$ for some $P \in \mathcal{C}$ such that $u \notin P$, then $R \cap R_i \neq \emptyset$ for some $1 \leq i \leq h$ since $P \cap (P_i - \{u\}) \neq \emptyset$ for some $1 \leq i \leq h$. □

Corollary 1. *1) k-semicoterie $\mathcal{C}$ is ND and satisfies Nonintersection Property and 2) (1-semi)coterie $\mathcal{D}$ is ND if and only if k-semicoterie $\mathcal{J}_u(\mathcal{C},\mathcal{D})$ is ND and satisfies Nonintersection Property.*

We would like to make a remark. As mentioned, Jiang and Huang showed that if both *k*-coterie $\mathcal{C}$ and coterie $\mathcal{D}$ are ND, then so is *k*-coterie $\mathcal{J}_u(\mathcal{C},\mathcal{D})$. However, the correctness of the other direction is open. Note that we cannot apply Corollary 1 to this end because of the differece between ND *k*-semicoteries and ND *k*-coteries mentioned in Section 1.

2.2 Availability

Let U be the set of processes in a distributed system and assume that every pair of processes has a distinct bidirectional communication link between them. Given a function $g: U \to [0,1]$ for specifying the probability $g(v)$ that a process $v \in U$ is operational, the *availability* $A_g(\mathcal{C})$ of a *k*-coterie $\mathcal{C}$ under U is defined by

$$A_g(\mathcal{C}) = \sum_{S \in Max(U,\mathcal{C})} p_g(U,S),$$

where

$$Max(U,\mathcal{C}) = \{S \subseteq U \mid P \subseteq S \text{ for some } P \in \mathcal{C}\},$$

and $p_g(U,S)$ is the probability that exactly the processes in S are operational, i.e.,

$$p_g(U,S) = \Pi_{v\in S} g(v) \Pi_{v \in U-S}(1-g(v)).$$

The availability of a *k*-coterie $\mathcal{C}$ is the probability that there is a quorum in $\mathcal{C}$ such that all processes in the quorum are operational. Thus, it is the probability that there exists a process that can enter the critical section when $\mathcal{C}$ is used in algorithm *k*-MUTEX, provided that the process operating probability is given by g and the communication links never fail.

Let g be any operating probability function of U. Define an operating probability function g' of U from g by $g'(u) = A_g(\mathcal{D})$ and $g'(v) = g(v)$ for all $v \in U - \{u\}$. We first introduce the following lemma whose proof, which is straightforward but lengthy, is given in the Appendix.

Lemma 1.

$$A_g(\mathcal{J}_u(\mathcal{C},\mathcal{D})) = A_{g'}(\mathcal{C}).$$

Proof. See Appendix. □

The following theorem states a sufficient condition for the coterie join operation to produce a *k*-coterie without decreasing the availability. An intuitive idea behind the proof is that if we increase the reliability of a process, then the availability of the *k*-coterie will not decrease.

Theorem 5. *If $A_g(\mathcal{D}) \geq g(u)$, then $A_g(\mathcal{J}_u(\mathcal{C},\mathcal{D})) \geq A_g(\mathcal{C})$.*

Proof. Assume that $A_g(\mathcal{D}) \geq g(u)$. By definition,

$$\begin{aligned} A_g(\mathcal{C}) &= \sum_{S\in Max(\cup\mathcal{C},\mathcal{C})} p_g(\cup\mathcal{C},S) \\ &= \sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\in S} p_g(\cup\mathcal{C},S) + \sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\notin S} p_g(\cup\mathcal{C},S) \\ &= g(u) \sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\in S} p_g(\cup\mathcal{C}-\{u\},S-\{u\}) \\ &\quad + (1-g(u)) \sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\notin S} p_g(\cup\mathcal{C}-\{u\},S). \end{aligned}$$

On the other hand, by Lemma 1,

$$\begin{aligned} A_g\mathcal{J}_u(\mathcal{C},\mathcal{D}) =& A_{g'}(\mathcal{C}) \\ =& A_g(\mathcal{D}) \sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\in S} p_g(\cup\mathcal{C}-\{u\},S-\{u\}) \\ &+ (1-A_g(\mathcal{D})) \\ &\sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\notin S} p_g(\cup\mathcal{C}-\{u\},S). \end{aligned}$$

Since $g(u) \leq A_g(\mathcal{D})$, clearly

$$\sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\in S} p_g(\cup\mathcal{C}-\{u\},S-\{u\}) \geq \sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\notin S} p_g(\cup\mathcal{C}-\{u\},S)$$

implies $A_g(\mathcal{C}) \leq A_g\mathcal{J}_u(\mathcal{C},\mathcal{D})$.

To show this inequality, it suffices to observe the following: Let $\mathcal{F} = \{S \in Max(\cup\mathcal{C},\mathcal{C}) \mid u \in S\}$ and $\mathcal{F}' = \{S \in Max(\cup\mathcal{C},\mathcal{C}) \mid u \notin S\}$.

1. If $S \in \mathcal{F}'$, then $S \cup \{u\} \in \mathcal{F}$.
2. If $S \in \mathcal{F}'$, then $(S \cup \{u\}) - \{u\} = S$. □

Suppose that $0 < g(v) < 1$ for any $v \in U$. Then,

$$\sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\in S} p_g(\cup\mathcal{C}-\{u\},S-\{u\}) > 0$$

and

$$\sum_{S\in Max(\cup\mathcal{C},\mathcal{C}),u\notin S} p_g(\cup\mathcal{C}-\{u\},S) > 0.$$

Hence, we have the following corollary:

Corollary 2. *Suppose that $0 < g(v) < 1$ for any $v \in U$. Then, $A_g(\mathcal{J}_u(\mathcal{C},\mathcal{D})) > A_g(\mathcal{C})$, if $A_g(\mathcal{D}) > g(u)$.*

The problem of constructing a *k*-coterie with higher availability is now reduced to the problem of searching for a coterie $\mathcal{D}$ such that $A_g(\mathcal{D}) > g(u)$ holds. Although this search looks to be difficult in general, it is tractable if we restrict g to be a constant function.

A coterie $\mathcal{C} = \{\{u\}\}$ for some $u \in U$ is called a *singleton coterie* under U. For an odd $n = |U|$, the *majority coterie* is defined by $\mathcal{C} = \{Q \subseteq U \mid |Q| = \lceil n/2 \rceil\}$. It is well known that the ND coteries that have the highest availability are 1) the majority coterie for $g(v) = g > 0.5$ [3] or 2) a singleton coterie for $g(v) = g < 0.5$ [7], which implies that the ND coteries that have the lowest availability are 1) a singleton coterie for $g(v) = g > 0.5$ or 2) the majority coterie for $g(v) = g < 0.5$. All other ND coteries are placed between them.

Since the availability of a singleton coterie $\{\{u\}\}$ is $g(u) = g$, we have the following corollary:

Corollary 3. *Suppose that $g(v) = g$ is a constant function such that $1 > g > 0.5$. Then, $A_g(\mathcal{J}_u(\mathcal{C}, \mathcal{D})) > A_g(\mathcal{C})$ if $\mathcal{D}$ is an ND coterie and $\mathcal{D}$ is not a singleton coterie.*

3 Tree Structured *k*-Coteries

Given a sequence of ND coteries $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{\ell-1}$, starting from an ND *k*-coterie $\mathcal{C}_0$, we can construct a sequence of ND *k*-coteries $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_\ell$ by applying the coterie join operation to $\mathcal{C}_i$ and a coterie $\mathcal{D}_i$ to construct $\mathcal{C}_{i+1}$. Corollary 3 guarantees that for any $0 \le i \le \ell - 1$, $A_g(\mathcal{C}_i) < A_g(\mathcal{C}_{i+1})$ holds if g is a constant function greater than 0.5 and $\mathcal{D}_i$s are not singleton coteries. This section discusses *k*-coteries constructed in this way.

3.1 Vote Assignable *k*-Semicoteries

Let $\mathcal{D}$ be a set of nonempty subsets of U. By $Min(\mathcal{D})$ we denote a subset of $\mathcal{D}$ constructed from $\mathcal{D}$ by removing each element if a proper subset of the element is in $\mathcal{D}$.

Definition 2. *To each element $u \in U$, we assign a nonnegative integer $w(u)$ and call it the* weight *of u. A threshold θ is an integer satisfying $1 \le \theta \le W$, where $W = \sum_{u \in U} w(u)$. Given a weight function w and a threshold θ, the* voting system *$\mathcal{V}_{w,\theta}(U)$ under U is defined by*

$$\mathcal{V}_{w,\theta}(U) = Min\left(\left\{Q \subseteq U \mid \sum_{u \in Q} w(u) \ge \theta\right\}\right).$$

A k-semicoterie $\mathcal{C}$ under U is said to be vote assignable *if there exists a weight function w and a threshold θ such that $\mathcal{C} = \mathcal{V}_{w,\theta}(U)$.*

The next theorem states a sufficient condition for a voting system to be an ND *k*-semicoterie and is used to prove the ND-ness of tree *k*-coteries.

Theorem 6. *Let $\mathcal{V}_{w,\theta}(U)$ be a voting system under U. For any integer $1 \le k \le |U|$, $\mathcal{V}_{w,\theta}(U)$ is an ND k-semicoterie if $\mathcal{V}_{w,\theta}(U)$ satisfies both of the following two conditions:*

1. *$(k+1)\theta = W + 1$ and*
2. *For any $S \subseteq U$, if $\sum_{u \in S} w(u) \ge k\theta$, then there exist k pairwise disjoint quorums*

$$Q_1, Q_2, \ldots, Q_k \in \mathcal{V}_{w,\theta}(U)$$

such that $Q_1 \cup Q_2 \cup \ldots \cup Q_k \subseteq S$.

Proof. We first show that $\mathcal{V}_{w,\theta}(U)$ is a *k*-semicoterie. Clearly, Minimality holds by Definition 2.

As for Intersection Property, there are k pairwise disjoint quorums in $\mathcal{V}_{w,\theta}(U)$ by Condition 2 since

$$\sum_{u \in U} w(u) = W = (k+1)\theta - 1 \ge k\theta$$

by Condition 1. Assume that there are $k+1$ pairwise disjoint quorums $Q_1, Q_2, \ldots, Q_{k+1} \in \mathcal{V}_{w,\theta}(U)$. Since

$$\sum_{u \in Q_i} w(u) \ge (W+1)/(k+1)$$

for $1 \le i \le k+1$ by Condition 1,

$$\begin{aligned} W = \sum_{u \in U} w(u) &\ge \sum_{1 \le i \le k+1} \sum_{u \in Q_i} w(u) \\ &\ge (k+1)((W+1)/(k+1)) = W + 1, \end{aligned}$$

a contradiction.

Next, we show that $\mathcal{V}_{w,\theta}(U)$ is ND. Suppose, otherwise, that $\mathcal{V}_{w,\theta}(U)$ is dominated. Then, by Theorem 2, there exists an $S \subseteq U$ such that 1) $Q \not\subseteq S$ for any $Q \in \mathcal{V}_{w,\theta}(U)$ and, 2) for any k pairwise disjoint quorums

$$Q_1, Q_2, \ldots, Q_k \in \mathcal{V}_{w,\theta}(U),$$

there exists an i such that $Q_i \cap S \ne \emptyset$. If $\sum_{u \in S} w(u) \ge \theta$, then there is a quorum $Q \in \mathcal{V}_{w,\theta}(U)$ such that $Q \subseteq S$, a contradiction. Hence, $\sum_{u \in S} w(u) < \theta$. Consider the complement $\overline{S}$ of S (i.e., $\overline{S} = U - S$). Since it follows that

$$\begin{aligned} &\sum_{u \in S} w(u) < \theta, \\ &\sum_{u \in \overline{S}} w(u) > W - \theta, \end{aligned}$$

and, hence, $\sum_{u \in \overline{S}} w(u) \ge k\theta$. Then, by Condition 2 of Theorem 6, there exist k pairwise disjoint quorums

$$Q_1, Q_2, \ldots, Q_k \in \mathcal{V}_{w,\theta}(U)$$

such that $Q_i \subseteq \overline{S}$ or, equivalently, $Q_i \cap S = \emptyset$ for $1 \le i \le k$, a contradiction. □

Note that Condition 2 of Theorem 6 always holds for $k = 1$. A sufficient condition for a vote assignable coterie $\mathcal{V}_{w,\theta}(U)$ to be ND is thus $\theta = (W+1)/2$, which was obtained in [10].

Example 2. Let $U = \{1, 2, 3, 4, 5\}$ and $k = 2$. Consider the voting system $\mathcal{V}_{w,\theta}(U)$ under U, where $w(i) = 1$ for $1 \le i \le 5$ and $\theta = 2$. Then,

$$\begin{aligned} \mathcal{V}_{w,\theta}(U) = &\{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\}, \\ &\{2,4\},\{2,5\},\{3,4\},\{3,5\},\{4,5\}\}. \end{aligned}$$

Since $\mathcal{V}_{w,\theta}(U)$ satisfies both conditions of Theorem 6, it is an ND 2-semicoterie.

Next, consider another voting system $\mathcal{V}_{w',\theta'}(U)$, where $w'(1) = w'(2) = w'(3) = 2$, $w'(4) = w'(5) = 1$, and $\theta' = 3$. Then,

$$\begin{aligned} \mathcal{V}_{w',\theta'}(U) = &\{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\}, \\ &\{2,4\},\{2,5\},\{3,4\},\{3,5\}\}. \end{aligned}$$

Although $\mathcal{V}_{w',\theta'}(U)$ satisfies Condition 1 of Theorem 6, it is not ND since Condition 2 does not hold for $S = \{1, 2, 3\}$. In fact, $\mathcal{V}_{w,\theta}(U)$ dominates $\mathcal{V}_{w',\theta'}(U)$.

3.2 Basic Tree *k*-Coteries

We now define what we called a *basic tree k-coterie* and associate a rooted tree of depth 2 with it. This rooted tree is used to define general tree *k*-coteries in the next section and is effectively used in the tree *k*-coterie based *k*-mutual exclusion algorithm we will propose in the Section 4.

Definition 3. *Given a positive integer k $(1 \le k \le |U|)$, let H and r, respectively, be a subset of U such that $|H| = km + 1$ for some integer m $(m \ge 2)$ and an element in H. A* basic tree k-coterie *$\mathcal{C}$ (with respect to H and r) is defined by*

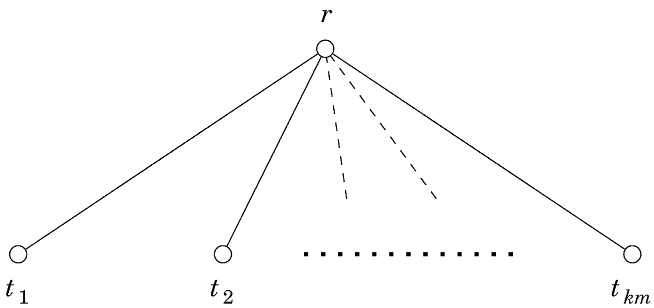


Fig. 1. An illustration of the rooted tree T_c associated with a basic tree k-coterie $\mathcal{C}$.

$$\mathcal{C} = \{Q \subseteq H \mid \{r\} \cap Q \neq \emptyset \text{ and } |Q| = 2\} \cup \{Q \subseteq H \mid \{r\} \cap Q = \emptyset \text{ and } |Q| = m\}.$$

The rooted tree, $T_\mathcal{C}$, associated with $\mathcal{C}$ has root r. The other elements t_i in H are children of r and form leaves of $T_\mathcal{C}$. The depth of $T_\mathcal{C}$ is, hence, 2 (see Fig. 1 for illustration).

Example 3. Let $U = \{1,2,3,4,5,6,7\}$. First, consider the case $k = 1$ and $m = 2$. Hence, $|H| = 3$. Let us select an $H = \{1,2,3\}$ and an $r = 1$. Then, we have the basic tree coterie (with respect to H and r)

$$\mathcal{C} = \{\{1,2\},\{1,3\},\{2,3\}\},$$

that is, in essence, a majority coterie.

Next, consider the case $k = 2$ and $m = 3$. Hence, $|H| = 7$. Let us select this time an $H = \{1,2,3,4,5,6,7\}$ and an $r = 1$. Then, we have the basic tree 2-coterie (with respect to H and r)

$$\begin{aligned}\mathcal{D} = \{&\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{1,6\},\{1,7\},\\ &\{2,3,4\},\{2,3,5\},\{2,3,6\},\{2,3,7\},\{2,4,5\},\\ &\{2,4,6\},\{2,4,7\},\{2,5,6\},\{2,5,7\},\{2,6,7\},\\ &\{3,4,5\},\{3,4,6\},\{3,4,7\},\{3,5,6\},\{3,5,7\},\\ &\{3,6,7\},\{4,5,6\},\{4,5,7\},\{4,6,7\},\{5,6,7\}\}.\end{aligned}$$

Theorem 7. *Any basic tree k-coterie defined above is indeed an ND k-coterie.*

Proof. Let $\mathcal{C}$ be a basic tree k-coterie with respect to H and r. We first show that $\mathcal{C}$ is a vote assignable k-semicoterie satisfying both conditions of Theorem 6.

Let $|H| = km + 1$. Define a weight function w by $w(r) = m - 1$, $w(u) = 1$ for $u \in H - \{r\}$, and $w(u) = 0$ for $u \in U - H$, where $|H| = km + 1$. Then, for threshold $\theta = m$, it is obvious to observe that $\mathcal{C} = \mathcal{V}_{w,m}(U)$, i.e., $\mathcal{C}$ is vote assignable.

Then, we show that $\mathcal{C}$ satisfies Conditions 1 and 2 of Theorem 6. As for Condition 1,

$$W = (m-1) + km = (k+1)m - 1$$

since the number of leaves is km. To verify Condition 2, consider any $S \subseteq U$ such that $\sum_{u \in S} w(u) = km$. Suppose first that $r \notin S$. Then, $w(u) = 1$ for all $u \in S$. Since $|S| = km$, there are k pairwise disjoint quorums in S, each of which consists of m leaves. Suppose next that $r \in S$. Then, there are $km - (m-1) = (k-1)m + 1$ leaves in S. Again there are k pairwise disjoint quorums in S; one consists of the root and a leaf and $k-1$ others each consists of m leaves. Hence, $\mathcal{C}$ is an ND k-semicoterie.

Finally, we show that Nonintersection Property holds for $\mathcal{C}$. Fix any h pairwise disjoint quorums $Q_1, Q_2, \ldots, Q_h \in \mathcal{C}$, where $1 \leq h < k$. There are two cases to consider. Suppose first that $r \notin Q_i$ for all $1 \leq i \leq h$. Since $w(u) = 1$ for any $u \in \cup_{i=1}^h Q_i$, $|\cup_{i=1}^h Q_i| = hm$. Since the number of leaves is km and $h < k$, there is a leaf t such that $t \in H - \cup_{i=1}^h Q_i$. Then, $\{r,t\} \in \mathcal{C}$ and $\{r,t\} \cap Q_i = \emptyset$ for all $1 \leq i \leq h$.

Suppose otherwise that $r \in Q_i$ for some $1 \leq i \leq h$. Then, $\cup_{i=1}^h Q_i$ consists of r and $hm - m + 1$ leaves, so there are $(k-h+1)m - 1$ leaves in $H - \cup_{i=1}^h Q_i$. Observe that $(k-h+1)m - 1 > m$ since $k > h$ and $m > 2$. Then, a set $Q = \{t_1, t_2, \ldots, t_m\}$ of m leaves in $H - \cup_{i=1}^h Q_i$ satisfies $Q \in \mathcal{C}$ and $Q \cap Q_i = \emptyset$ for all $1 \leq i \leq h$.

Finally, recall that an ND k-semicoterie with Nonintersection Property is an ND k-coterie. □

3.3 Tree *k*-Coteries

In the spirit we described at the beginning of Section 3, this section constructs tree structured k-coteries from a basic tree k-coterie $\mathcal{C}_0$ and a sequence of basic tree (1-)coterie $\mathcal{D}_i$. Since k-semicoterie $\mathcal{C}_0$ and coteries $\mathcal{D}_i$s are ND and satisfy Nonintersection Property, k-semicoterie $\mathcal{C}_\ell$ is ND and the satisfies Nonintersection Property, by Corollary 1. Furthermore, by Corollary 3, the availability of $\mathcal{C}_\ell$ is higher than that of $\mathcal{C}_{\ell-1}$, provided that the operating probability function g is a constant function greater than 0.5, since a basic tree coterie is not a singleton coterie by definition.

In order for $\mathcal{J}_u(\mathcal{C}, \mathcal{D})$ to be defined, $\cup\mathcal{C} \cap \cup\mathcal{D} \subseteq \{u\}$ must be required. In the following construction, we further restrict the selection of u. Our intention is to construct a new message-efficient k-mutual algorithm that effectively makes use of the structure of tree k-coteries at the expense of the variety of tree k-coteries.

A *tree k-coterie* is recursively defined by using the coterie join operation as follows: In the definition, we associate a rooted tree T for each tree k-coterie $\mathcal{J}_u(\mathcal{C}, \mathcal{D})$. This tree T plays an important role in the tree k-coterie based k-mutual exclusion algorithm.

1. Any basic tree k-coterie $\mathcal{C}$ is a tree k-coterie. The rooted tree $T_\mathcal{C}$ associated with $\mathcal{C}$ was already defined in Section 3.2.
2. Let $\mathcal{C}$ and $\mathcal{D}$, respectively, be a tree k-coterie and a basic tree (1-)coterie and assume that $T_\mathcal{C}$ and $T_\mathcal{D}$ are the rooted trees associated with them.

   If $\cup\mathcal{C} \cap \cup\mathcal{D} = \{u\}$ and u is a leaf of $T_\mathcal{C}$, then $\mathcal{J}_u(\mathcal{C}, \mathcal{D})$ is a tree k-coterie. If $\cup\mathcal{C} \cap \cup\mathcal{D} = \emptyset$, then $\mathcal{J}_u(\mathcal{C}, \mathcal{D})$ is a tree k-coterie for any leaf u of $T_\mathcal{C}$. The associated rooted tree T is constructed from $T_\mathcal{C}$ by replacing leaf u with tree $T_\mathcal{D}$, i.e., we remove leaf u and place the root of $T_\mathcal{D}$ instead of u.[2] All leaves of $\mathcal{C}$, except u, and all leaves of $\mathcal{D}$ are now leaves of $\mathcal{J}_u(\mathcal{C}, \mathcal{D})$.
3. No other quorum sets are tree k-coteries.

2. The root of $T_\mathcal{D}$ can be u when $\cup\mathcal{C} \cap \cup\mathcal{D} = \{u\}$.

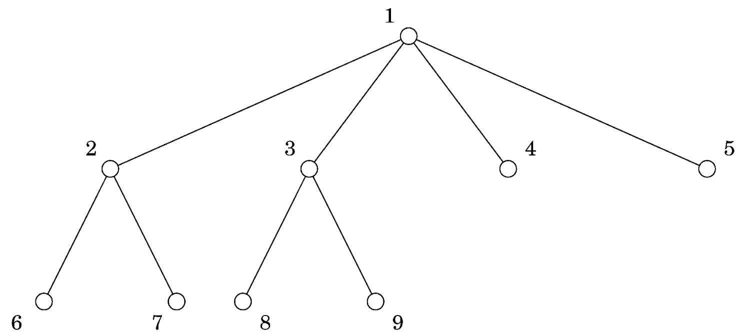


Fig. 2. An illustration of the rooted tree associated with $\mathcal{C}_2$ in Example 4.

Example 4. Consider the following three coteries:

$$\mathcal{C}_0 = \{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,4\},\{2,5\},\{3,4\},\{3,5\},\{4,5\}\},$$
$$\mathcal{D}_0 = \{\{2,6\},\{2,7\},\{6,7\}\},$$
$$\mathcal{D}_1 = \{\{3,8\},\{3,9\},\{8,9\}\}.$$

$\mathcal{C}_0$ is a basic tree 2-coterie with root $r = 1$ and $m = 2$. $\mathcal{D}_0$ and $\mathcal{D}_1$ are basic tree coteries with roots 2 and 3, respectively. Since $\cup\mathcal{C}_0 \cap \cup\mathcal{D}_0 \subseteq \{2\}$ and 2 is a leaf of $\mathcal{C}_0$, $\mathcal{C}_1$ defined by

$$\mathcal{C}_1 = \mathcal{J}_2(\mathcal{C}_0, \mathcal{D}_0) = \{\{1,3\},\{1,4\},\{1,5\},\{3,4\},\{3,5\},\{4,5\},\{1,2,6\},\{1,2,7\},\{1,6,7\},\{2,3,6\},\{2,3,7\},\{2,4,6\},\{2,4,7\},\{2,5,6\},\{2,5,7\},\{3,6,7\},\{4,6,7\},\{5,6,7\}\},$$

is a tree 2-coterie. Since $\cup\mathcal{C}_1 \cap \cup\mathcal{D}_1 \subseteq \{3\}$ and 3 is a leaf of $\mathcal{C}_1$, $\mathcal{C}_2$ defined by

$$\mathcal{C}_2 = \mathcal{J}_3(\mathcal{C}_1, \mathcal{D}_1) = \{\{1,4\},\{1,5\},\{4,5\},\{1,2,6\},\{1,2,7\},\{1,3,8\},\{1,3,9\},\{1,6,7\},\{1,8,9\},\{2,4,6\},\{2,4,7\},\{2,5,6\},\{2,5,7\},\{3,4,8\},\{3,4,9\},\{3,5,8\},\{3,5,9\},\{4,6,7\},\{4,8,9\},\{5,6,7\},\{5,8,9\},\{2,3,6,8\},\{2,3,6,9\},\{2,3,7,8\},\{2,3,7,9\},\{2,6,8,9\},\{2,7,8,9\},\{3,6,7,8\},\{3,6,7,9\},\{6,7,8,9\}\},$$

is also a tree 2-coterie.

Fig. 2 illustrates the rooted tree associated with $\mathcal{C}_2$.

As observed, we have the following theorem:

Theorem 8. *Every tree k-coterie is an ND k-coterie.*

4 Tree Algorithm for *k*-Mutual Exclusion

Agrawal and El Abbadi proposed a mutual exclusion algorithm called the *tree algorithm* [1], which is one of the most well-known coterie-based algorithms. In this section, we extend their mutual exclusion algorithm and propose a new *k*-mutual exclusion algorithm that effectively makes use of the rooted tree associated with a tree *k*-coterie. We call the algorithm *k*-TREE.

4.1 Algorithm *k*-TREE

Let U be the set of processes forming the distributed system under consideration. Suppose that a tree *k*-coterie $\mathcal{C}$ under U is used in *k*-TREE, where $\mathcal{C}(= \mathcal{C}_\ell)$ is constructed from a basic tree *k*-coterie $\mathcal{C}_0$ and a sequence of basic tree 1-coterie $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{\ell-1}$ and $m = m_0$ is used to construct $\mathcal{C}_0$. Let T and r be the rooted tree associated with $\mathcal{C}$ and its root, respectively. Note that, in real applications, T, not $\mathcal{C}$, is usually given since the description length of $\mathcal{C}$ can be exponential in that of T.

Algorithm *k*-TREE works as follows: When a process u wishes to enter the critical section, u calls the following recursive function $GetQuorum(r)$, which is evaluated among a set of processes. If $GetQuorum(r)$ returns a set Q of processes, then $Q \in \mathcal{C}$ and every process in Q has been locked for u; i.e., u can enter the critical section. When u leaves the critical section, it unlocks all processes in Q. If $GetQuorum(r)$ returns "fail," then, currently there is no quorum in $\mathcal{C}$ that is available for u.

Function $GetQuorum(p$: process): $Quorum$

1. **Case *p* is root *r*.** If r is unlocked, then lock itself and return $\{r\} \cup Q(t)$ as $GetQuorum(r)$ if a child t returns a set $Q(t)$ as $GetQuorum(t)$. If another child $x(\neq t)$ also returns a set $Q(x)$ as $GetQuorum(x)$, then unlock all processes in $Q(x)$. If all children return "fail," then return "fail"as $GetQuorum(r)$.
   If r is locked, then return $\cup_{i=1}^{m_0} Q(t_i)$ as
   $$GetQuorum(r)$$
   if m_0 children $t_i, (1 \le i \le m_0)$ return a set $Q(t_i)$ as $GetQuorum(t_i)$. If another child $x(\neq t_i, (1 \le i \le m_0))$ also returns a set $Q(x)$ as $GetQuorum(x)$, then unlock all processes in $Q(x)$. If less than m_0 children t_i return a set $Q(t_i)$ as $GetQuorum(t_i)$, then return "fail" as $GetQuorum(r)$ and unlock all processes in $Q(t_i)$s.
2. **Case *p* is a leaf.** If p is unlocked, then lock itself and return $\{p\}$ as $GetQuorum(p)$; otherwise, return "fail" as $GetQuorum(p)$.
3. **Case *p* is an intermediate vertex.** If p is unlocked, then lock itself and return $\{p\} \cup Q(t)$ as $GetQuorum(p)$ if a child t returns a set $Q(t)$ as $GetQuorum(t)$. If another child $x(\neq t)$ also returns a set $Q(x)$ as
   $$GetQuorum(x),$$

then unlock all processes in $Q(x)$. If all children return "fail," then return "fail" as $GetQuorum(p)$.

If p is locked, then return $\cup_{i=1}^{d} Q(t_i)$ as $GetQuorum(p)$ if every child $t_i, (1 \le i \le d)$ of p returns a set $Q(t_i)$ as $GetQuorum(t_i)$, where d is the number of the children. If not all children t_i return a set $Q(t_i)$ as $GetQuorum(t_i)$, then return "fail" as $GetQuorum(p)$ and unlock all processes in $Q(t_i)$s.

The procedures for the root and an intermediate vertex are quite similar; when p is locked, the former needs only m_0 (out of km_0) successful children, while the latter needs all children to be successful.

Depending on which processes are now being locked, $GetQuorum(r)$ may return a different quorum. Let $\mathcal{Q}(T)$ be the set of quorums that $GetQuorum(r)$ function can produce for T.

Theorem 9. $\mathcal{Q}(T) = \mathcal{C}$.

Proof. The proof is by induction on the order that $\mathcal{C} = \mathcal{C}_\ell$ is constructed. Since the base case, i.e., the case $\mathcal{C} = \mathcal{C}_0$, is obvious, by the definitions of basic tree *k*-coterie and function $GetQuorum$, we concentrate on the induction step.

Let T_i be the rooted graph associated with $\mathcal{C}_i$ for any $0 \le i \le \ell$. The induction hypothesis guarantees $\mathcal{Q}(T_{\ell-1}) = \mathcal{C}_{\ell-1}$. By assumption, there is a leaf u of $T_{\ell-1}$ such that 1) $\cup\, \mathcal{C}_{\ell-1} \cap \cup \mathcal{D}_{\ell-1} \subseteq \{u\}$ and 2) $\mathcal{C}_\ell = \mathcal{J}_u(\mathcal{C}_{\ell-1}, \mathcal{D}_{\ell-1})$. Let x be the root of the tree associated with $\mathcal{D}_{\ell-1}$. Then, T_ℓ is constructed from $T_{\ell-1}$ by replacing u with x (and the whole rooted tree). Note that x can be u.

By the definition of *k*-TREE, $P \in \mathcal{Q}(T_{\ell-1})$ and $u \in P$ if and only if $(P - \{u\}) \cup Q \in \mathcal{Q}(T_\ell)$ for any $Q \in \mathcal{D}_{\ell-1}$ since $\cup \mathcal{C}_{\ell-1} \cap \cup\, \mathcal{D}_{\ell-1} \subseteq \{u\}$ and, for any tree *k*-coterie, each element in U appears at most once as a vertex in the associated rooted tree. On the other hand, $P \in \mathcal{Q}(T_{\ell-1})$ and $u \notin P$ if and only if $P \in \mathcal{Q}(T_\ell) \cap \mathcal{Q}(T_{\ell-1})$.

Since $\mathcal{Q}(T_{\ell-1}) = \mathcal{C}_{\ell-1}$,

$$\mathcal{C}_\ell = \mathcal{J}_u(\mathcal{C}_{\ell-1}, \mathcal{D}_{\ell-1}) = \mathcal{Q}(T_\ell).$$

□

4.2 Message Complexity

In order to demonstrate the effectiveness of Algorithm *k*-TREE, let us estimate its message complexity, i.e., the number of messages necessary to exchange for a process to enter the critical section. Observe that messages are consumed when 1) a process p calls $GetQuorum(t)$ for some of its children t, 2) t returns its value to p, and 3) p unlocks some of locked processes. A basic assumption we make regarding *k*-TREE is that p calls $GetQuorum(t)$ one by one, i.e., p always calls $GetQuorum(t')$ for another child t' after receiving the value of $GetQuorum(t)$ from a child t. Note that this prohibition against concurrent search for unlocked processes is a well-known practical strategy for avoiding deadlocks and is called the ordered resource policy. For eliminating meaningless message exchanges, we can further assume that if a child t of r receives $GetQuorum(t)$ when it is locked, it immediately returns fail.

Consider an execution on $GetQuorum(r)$ on a rooted tree $T = (U, E)$ (associated with a tree *k*-coterie $\mathcal{C}$ under U) and let $S = (V, A)$ be the subgraph of T consisting of vertices and edges on which messages are flowed. Suppose that $GetQuorum(r)$ returns a quorum $Q \in \mathcal{C}$. Clearly, $Q \subseteq V$ and the message complexity is bounded by $3|A|$. If $GetQuorum(r)$ returns fail, the number of messages that are exchanged in vain is bounded by $2|A|$. For our purpose, it suffices to estimate $|A|$.

The size of S depends both of T and the set of currently locked processes. As for T, an extremal case is a tree of depth 2, i.e., $\mathcal{C}$ is a basic tree *k*-coterie. Then, $|A|$ is terribly large and is $\Omega(n/k)$ even if root r alone is locked, although $|A| = 1$ if no processes are locked. Another extremal case is a balanced tree such that root r has $k+1$ children and every internal vertex, except r, has two children. In this case, $|A|$ is bounded by $O(\log(n/k))$ for the case in which r alone is locked, whereas $|A| = \Omega(\log(n/k))$, even if no processes are locked. For making the message complexity in the worst case better, we suggest the balanced tree as T and assume it in the following analysis.

We now estimate the total number N_k of messages necessary to exchange for k processes to enter the critical section. Let $n_i (0 \le i \le k-1)$ be the size of S, provided that i processes are already in the critical section. Obviously, $n_0 = n_1 = O(\log(n/k))$. Observe that $n_i = O(i + \log(n/k))$ since the first i children of r are locked and the search for the $i + 1\text{th}$ child succeeds in $O(\log(n/k))$ messages. Then, we have $N_k = O(k(k + \log(n/k))$.

However, N_k is actually reducible to $O(k \log(n/k))$, since r knows which of its children are currently unlocked and, hence, we may be able to assume that r can instruct currently unlocked child. Such a modification makes *k*-TREE resemble a centralized algorithm. We would like to emphasize the fact that *k*-TREE works even if r is down, which is the point completely different from a centralized algorithm, although more messages would be required to enter the critical section than a centralized algorithm.

5 Conclusion

In this paper, we first considered the coterie join operation that produces a new *k*-semicoterie from a given *k*-semicoterie and a (1-semi)coterie. We characterized when ND *k*-semicoteries and/or *k*-semicoteries with Nonintersection Property are produced by the operation and discussed conditions when the operation increases the availability. Based on those results, we next proposed a method to produce a sequence of ND *k*-coteries called tree *k*-coteries. Furthermore, we can guarantee that the sequence is sorted in increasing order of the availability, assuming a certain natural condition on the operating probability. Finally, we proposed a new *k*-mutual exclusion algorithm that effectively makes use of a tree *k*-coterie and briefly discussed its message complexity, assuming that the distributed system is reliable. However, we leave the analysis for the unreliable case as an important future work.

APPENDIX

PROOF OF LEMMA 1

By $\mathcal{J}$ we denote $\mathcal{J}_u(\mathcal{C}, \mathcal{D})$. The availability of $\mathcal{J}$ with respect to g is, by definition,

$$A_g(\mathcal{J}) = \sum_{S \in Max(U,\mathcal{J})} p_g(U, S) = \sum_{S \in Max(U,\mathcal{J}) \cap Max(U,\mathcal{D})} p_g(U, S) + \sum_{S \in Max(U,\mathcal{J}) - Max(U,\mathcal{D})} p_g(U, S). \quad (1)$$

For any $S \subseteq U$, let $S_D = S \cap \cup\mathcal{D}$ and $S_C = S - S_D$. Clearly,

$$p_g(U, S) = p_g(\cup\mathcal{D}, S_D) \cdot p_g(U - \cup\mathcal{D}, S_C).$$

We first evaluate the first sum of the righthand side of (1). Let $\mathcal{C}_1 = \{P - \{u\} | P \in \mathcal{C}\}$. Then,

$$S \in Max(U, \mathcal{J}) \cap Max(U, \mathcal{D})$$

if and only if $S_D \in Max(\cup\mathcal{D}, \mathcal{D})$ and $S_C \in Max(U - \cup\mathcal{D}, \mathcal{C}_1)$. Hence,

$$\sum_{S \in Max(U,\mathcal{J}) \cap Max(U,\mathcal{D})} p_g(U, S) = \left(\sum_{S_D \in Max(\cup\mathcal{D},\mathcal{D})} p_g(\cup\mathcal{D}, S_D) \right) \left(\sum_{S_C \in Max(U - \cup\mathcal{D},\mathcal{C}_1)} p_g(U - \cup\mathcal{D}, S_C) \right).$$

Since

$$\sum_{S_D \in Max(\cup\mathcal{D},\mathcal{D})} p_g(\cup\mathcal{D}, S_D) = \sum_{S_D \in Max(U,\mathcal{D})} p_g(U, S_D),$$

$$\sum_{S_D \in Max(\cup\mathcal{D},\mathcal{D})} p_g(\cup\mathcal{D}, S_D) = A_g(\mathcal{D}) = g'(u)$$

holds. On the other hand, since $\cup\mathcal{C}_1 = \cup\mathcal{C} - \{u\}$,

$$\sum_{S_C \in Max(U - \cup\mathcal{D},\mathcal{C}_1)} p_g(U - \cup\mathcal{D}, S_C) = \sum_{S_C \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_1)} p_g(\cup\mathcal{C} - \{u\}, S_C) = \sum_{S_C \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_1)} p_{g'}(\cup\mathcal{C} - \{u\}, S_C).$$

Hence,

$$\sum_{S \in Max(U,\mathcal{J}) \cap Max(U,\mathcal{D})} p_g(U, S) = g'(u) \sum_{S_C \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_1)} p_{g'}(\cup\mathcal{C} - \{u\}, S_C).$$

Next, we evaluate the second sum of the righthand side of (1). By definition, $S \in Max(U, \mathcal{J}) - Max(U, \mathcal{D})$ if and only if $S_D \notin Max(\cup\mathcal{D}, \mathcal{D})$ and $S_C \in Max(U - \cup\mathcal{D}, \mathcal{C}_2)$, where $\mathcal{C}_2 = \{P \in \mathcal{C} | u \notin P\}$. Then,

$$\sum_{S \in Max(U,\mathcal{J}) - Max(U,\mathcal{D})} p_g(U, S) = \left(\sum_{S_D \notin Max(\cup\mathcal{D},\mathcal{D})} p_g(\cup\mathcal{D}, S_D) \right) \left(\sum_{S_C \in Max(U - \cup\mathcal{D},\mathcal{C}_2)} p_g(U - \cup\mathcal{D}, S_C) \right).$$

By definition,

$$\sum_{S_D \notin Max(\cup\mathcal{D},\mathcal{D})} p_g(\cup\mathcal{D}, S_D) = 1 - A_g(\mathcal{D}) = 1 - g'(u).$$

Since $\cup\mathcal{C}_2 = \cup\mathcal{C} - \{u\}$,

$$\sum_{S_C \in Max(U - \cup\mathcal{D},\mathcal{C}_2)} p_g(U - \cup\mathcal{D}, S_C) = \sum_{S_C \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_2)} p_{g'}(\cup\mathcal{C} - \{u\}, S_C).$$

Hence,

$$\sum_{S \in Max(U,\mathcal{J}) - Max(U,\mathcal{D})} p_g(U, S) = (1 - g'(u)) \sum_{S_C \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_2)} p_{g'}(\cup\mathcal{C} - \{u\}, S_C).$$

Finally, we evaluate $A_{g'}(\mathcal{C})$.

$$\begin{aligned} A_{g'}(\mathcal{C}) &= \sum_{S \in Max(U,\mathcal{C})} p_{g'}(U, S) \\ &= \sum_{S \in Max(U,\mathcal{C}), u \in S} p_{g'}(U, S) + \sum_{S \in Max(U,\mathcal{C}), u \notin S} p_{g'}(U, S) \\ &= g'(u) \sum_{S \in Max(U,\mathcal{C}), u \in S} p_{g'}(U - \{u\}, S - \{u\}) \\ &\quad + (1 - g'(u)) \sum_{S \in Max(U,\mathcal{C}), u \notin S} p_{g'}(U - \{u\}, S) \\ &= g'(u) \sum_{S \in Max(\cup\mathcal{C},\mathcal{C}), u \in S} p_{g'}(\cup\mathcal{C} - \{u\}, S - \{u\}) \\ &\quad + (1 - g'(u)) \sum_{S \in Max(\cup\mathcal{C},\mathcal{C}), u \notin S} p_{g'}(\cup\mathcal{C} - \{u\}, S). \end{aligned}$$

The proof completes if both

$$\sum_{S \in Max(\cup\mathcal{C},\mathcal{C}), u \in S} p_{g'}(\cup\mathcal{C} - \{u\}, S - \{u\}) = \sum_{S \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_1)} p_{g'}(\cup\mathcal{C} - \{u\}, S)$$

and

$$\sum_{S \in Max(\cup\mathcal{C},\mathcal{C}), u \notin S} p_{g'}(\cup\mathcal{C} - \{u\}, S) = \sum_{S \in Max(\cup\mathcal{C} - \{u\},\mathcal{C}_2)} p_{g'}(\cup\mathcal{C} - \{u\}, S)$$

hold. Clearly, $S \in \{X (\in Max(\cup\mathcal{C}, \mathcal{C})) | u \in X\}$ if and only if $S = T \cup \{u\}$ for some $T \in Max(\cup\mathcal{C} - \{u\}, \mathcal{C}_1)$ and $S = T$ for some $T \in \{X (\in Max(\cup\mathcal{C}, \mathcal{C})) | u \notin X\}$ if and only if $S \in Max(\cup\mathcal{C} - \{u\}, \mathcal{C}_2)$. Thus,

$$\sum_{S \in Max(\cup\mathcal{C},\mathcal{C}), u \in S} p_{g'}(\cup\mathcal{C} - \{u\}, S - \{u\}) = \sum_{S \in Max(\cup\mathcal{C} - \{u\}, \mathcal{C}_1)} p_{g'}(\cup\mathcal{C} - \{u\}, S)$$

and

$$\sum_{S \in Max(\cup\mathcal{C},\mathcal{C}), u \notin S} p_{g'}(\cup\mathcal{C} - \{u\}, S) = \sum_{S \in Max(\cup\mathcal{C} - \{u\}, \mathcal{C}_2)} p_{g'}(\cup\mathcal{C} - \{u\}, S)$$

hold. □

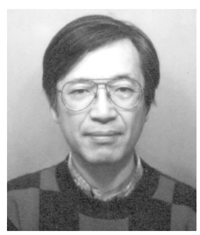
Acknowledgments

The authors wish to thank the anonymous referees for their valuable comments and suggestions.

References

[1] D. Agrawal and A. El Abbadi, "An Efficient and Fault-Tolerant Solution for Distributed Mutual Exclusion," *ACM Trans. Computer Systems,* vol. 9, no. 1, pp. 1-20, Feb. 1991.
[2] D. Agrawal, O. Egecioglu, and A. El Abbadi, "Analysis of Quorum-Based Protocols for Distributed (k+1)- Exclusion," *IEEE Trans. Parallel and Distributed Systems,* vol. 8, no. 5, pp. 533-537, May 1997.
[3] D. Barbara and H. Garcia-Molina, "The Reliability of Voting Mechanisms," *IEEE Trans. Computers,* vol. 36, no. 10, pp. 1197-1208, Oct. 1987.
[4] J.C. Bioch and T. Ibaraki, "Generating and Approximating Nondominated Coteries," *IEEE Trans. Parallel and Distributed Systems,* vol. 6, no. 9, pp. 905-914, Sept. 1995.
[5] 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. 582-59, Dec. 1992.
[6] H.K. Chang, S.M. Yuan, "Performance Characterization of the Tree Quorum Algorithm," *IEEE Trans. Parallel and Distributed Systems,* vol. 6, no. 6, pp. 658-662, June 1995.
[7] K. Diks, E. Kranakis, D. Krizanc, B. Mans, and A. Pelc, "Optimal Coteries and Voting Schemes," *Information Processing Letters,* vol. 51, no. 1, pp. 1-6, July 1994.
[8] S. Fujita, M. Yamashita, and T. Ae, "Distributed *k*-Mutual Exclusion Problem and *k*-coteries," *Lecture Notes in Computer Science* vol. 557, pp. 22-31, 1991.
[9] D.K. Gifford, "Weighted Voting for Replicated Data," *Proc. Seventh Symp. Operating Systems Principles,* Dec. 1979.
[10] H. Garcia-Molina and D. Barbara, "How to Assign Votes in a Distributed Systems," *J. ACM,* vol. 32, no. 4, pp. 841-860, Oct. 1985.
[11] S.T. Huang, J.R. Jiang, and Y.C. Kuo, "K-Coteries for Fault-Tolerant k Entries to a Critical Section," *Proc. 13th Int'l Conf. Distributed Computing Systems,* May 1993.
[12] T. Ibaraki and T. Kameda, "A Theory of Coteries: Mutual Exclusion in Distributed Systems," *IEEE Trans. Parallel and Distributed Systems,* vol. 4, no. 7, pp. 779-794, July 1993.
[13] J.R. Jiang and S.T. Huang, "Obtaining Nondominated K-Coteries for Fault-Tolerant Distributed K-Mutual Exclusion," *Proc. Int'l Conf. Parallel and Distributed Systems,* 1994.
[14] H. Kakugawa, S. Fujita, M. Yamashita, and T. Ae, "A Distributed *k*-Mutual Exclusion Algorithm Using *k*-Coterie," *Information Processing Letters,* vol. 49, no. 4, pp. 213-218, Feb. 1994.
[15] A. Kumar, "Hierarchical Quorum Consensus: A New Algorithm for Managing Replicated Data," *IEEE Trans. Computers,* vol. 40, no. 9, pp. 996-1004, Sept. 1991.
[16] Y.-C. Kuo and S.-T. Huang, "A Geometric Approach for Constructing Coteries and *k*-Coteries," *IEEE Trans. Parallel and Distributed Systems,* vol. 8, no. 4, pp. 402-411, Apr. 1997.
[17] M. Maekawa, "A $\sqrt{N}$ Algorithm for Mutual Exclusion in Decentralized Systems," *ACM Trans. Computer Systems,* vol. 3, no. 2, pp. 145-159, May 1985.
[18] M.L. Neilsen and M. Mizuno, "Coterie Join Algorithm," *IEEE Trans. Parallel and Distributed Systems,* vol. 3, no. 5, pp. 582-590, Sept. 1992.
[19] M.L. Neilsen and M. Mizuno, "Nondominated *k*-Coteries for Multiple Mutual Exclusion," *Information Processing Letters,* vol. 50, no. 5, pp. 247-252, June 1994.
[20] M.L. Neilsen and M. Mizuno, "Erratum to Nondominated *k*-Coteries for Multiple Mutual Exclusion," *Information Processing Letters,* vol. 60, no. 6, pp. 319-23, Dec. 1996.
[21] K. Raymond, "A Distributed Algorithm for Multiple Entries to a Critical Section," *Information Processing Letters,* vol. 30, no. 4, pp. 189-193, Feb. 1989.
[22] P.K. Srimani and R.L.N. Reddy, "Another Distributed Algorithm for Multiple Entries to a Critical Section," *Information Processing Letters,* vol. 41, no. 1, pp. 51-57, Jan. 1992.
[23] S.M. Yuan and H.K. Chang, "Message Complexity of the Tree Quorum Algorithm," *IEEE Trans. Parallel and Distributed Systems,* vol. 6, no. 8, pp. 887-890, Aug. 1995.

Takashi Harada received the BE and ME degrees from Hiroshima University, Japan, in 1983 and 1985, respectively, and the DE degree from Kyushu University, Japan, in 2000. From 1986 to 2000, he was a research associate in the Information Processing Center at Hiroshima University. He is currently an associate professor in the Graduate School of Management at Hiroshima University. His research interests include distributed computing, fault-tolerant computing, and communication networks. He is a member of the IEEE Computer Society.



Masafumi Yamashita received the BE and ME degrees from Kyoto University, Kyoto, Japan, in 1974 and 1977, respectively, and the DE degree from Nagoya University, Nagoya, Japan, in 1981. He worked from 1980 to 1985, as a research associate of Toyohashi University of Technology, Toyohashi, Japan, and then, from 1985 to 1998, as an associate professor and a full professor with Hiroshima University, Higashi-Hiroshima, Japan. Since 1998, he has been a full professor with Kyushu University, Fukuoka, Japan. His research interests include distributed/parallel algorithms/systems and computational geometry. He is a member of the ACM, the IEICE of Japan, the IEEE Computer Society, the IPS of Japan, and the SIAM of Japan.