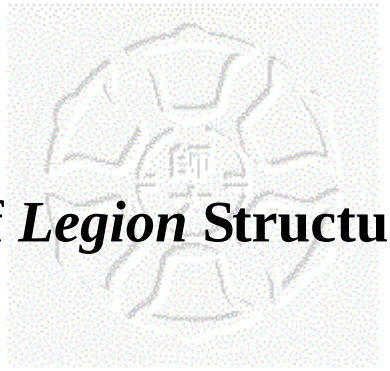
Chapter 2



The Theory of *Legion* Structure

2.1 Quorum, Set System, and *K*-Coterie

In this section, we introduce the concepts of set system [31], quorum [32,33], and *k*-coterie [34]. A set system is composed of some quorums and can be a *k*-coterie if it satisfies some properties (Definition 2). A *k*-coterie can be applied to develop distributed algorithms to achieve *k*-mutual exclusion. Fujita et al. [34] and Huang et al. [35] defined *k*-coteries in 1991 and 1993.

DEFINITION 1. *A* ***set system*** [31] $C=\{Q_1, Q_2, ..., Q_n\}$ *is a collection of nonempty subsets* $Q_i \subseteq U$ *of a finite universe U.*

DEFINITION 2. *A* ***k-coterie*** [34] *is a nonempty set system that has the following properties:*

[I] Nonintersection property: *Given any h* ($h < k$) *elements* $Q_1, Q_2, ..., Q_h \in C$ *such that* $Q_i \cap Q_j = \emptyset$ ($i \neq j$, $1 \leq i, j \leq h$), *there exists another element* $Q \in C$ *such that* $Q \cap Q_t = \emptyset$, ($1 \leq t \leq h$).

[II] Intersection Property: *Among any k*+1 *elements* $Q_1, Q_2, ..., Q_{k+1} \in C$,

there exist at least two elements Q_i *and* Q_j ($i \neq j$, $1 \leq i, j \leq k+1$), *such that* $Q_i \cap Q_j \neq \emptyset$.

[III] Minimality property: *For any pair of distinct elements* $Q_i, Q_j \in C$, *there doesn't exist* $Q_i \subset Q_j$.

Each element Q of C in Definitions 1 and 2 is called a **quorum**.

Take the set C = {{0,1}, {1,2}, {2,3}, {3,4}, {4,5}, {5,6}, {6,0}} as an example. Set C is a 3-coterie since it satisfies all three properties of k = 3. Given one quorum Q_1 = {1, 2}, we can always find another quorum Q_2 = {3, 4} such that Q_1 and Q_2 are disjoint. On the other hand, given two quorums Q_a = {1, 2}, Q_b = {4, 5}, we can also find another quorum Q_c = {6, 0} such that Q_a, Q_b, and Q_c are disjoint. However, among any four quorums, for example, {0, 1}, {2, 3}, {4, 5}, and {6, 0}, there exist two of them, for example, {0, 1}, and {6, 0}, that intersect.

2.2 *Legion* Structure

In this section, we propose the *Legion* structure. A *Legion* is constructed from set systems or k-coteries by giving them parameters. The theory behind the *Legion* structure can be applied to develop many distributed schemes or approaches to enable various applications, for example, mutual-exclusion, data replication, or location management in mobile systems.

DEFINITION 3. *A* ***Legion*** $Leg(k_1,k_2,...,k_m) \equiv \{C_1,C_2,...,C_m\}$, where $1 \leq m$, $k_i \in \{null,1,2,\ldots,n\}$, *is a collection of set systems that has the following properties:*

[I] $C_i = \{Q_1, Q_2, ..., Q_n\}$ *is a* k_i*-coterie, if* $1 \le k_i \le n$.

[II] $C_i = \{Q_1, Q_2, ..., Q_n\}$ *is a set system and can be a k-coterie* ($1 \le k \le n$) *or may not be* (*i.e., don't care*)*, if* k_i*=null.*

[III] *For any pair of quorums* $Q_S \in C_i$ *and* $Q_t \in C_j$ *, there is* $Q_S \cap Q_t \neq \emptyset$*, where* C_i *and* C_j *are different set systems* (*i.e.,* $i \neq j$, $1 \le i, j \le m$).

Take the m=2, *Leg*(1, *null*) ≡{C_1, C_2} as an example, where k_1=1, k_2=*null*. Consider the universal set U={1,2,3,4}. The *Leg*(1, *null*) ≡ {C_1, C_2} could be {{{1,3,4},{2,4,1},{3,1,2},{4,2,3}},{{1,2},{2,3},{3,4},{4,1}}}. Since C_1 is a 1-coterie (k_1=1), any two quorums of C_1, for example {1,3,4} and {3,1,2}, should be joint. On the other hand, C_2 is a don't case (k_2=*null*) that any two quorums of C_2 may be joint or disjoint, for example {1,2} and {3,4}. But, two quorums belonging to C_1 and C_2 respectively, for example {3,1,2} and {4,1}, should have an intersection.

According to Definition 3, a *Legion* has $m \ge 1$ parameters. These parameters can be *null*, 1, 2, ..., or n, depending on the application, for example, *Leg*(1), *Leg*(5), *Leg*(1, *null*), etc. We can apply the *Legion* structure to various applications of distributed computing. New schemes can also be developed according to the definition. We will discuss some applications of the *Legion* structure in the following section and, in section 2.4, propose a new scheme for mobile location management based on the definition of *Legion*.

2.3 Applications of the *Legion* Structure

Depending on the different parameters and the numbers of parameters employed, we can derive various applications of the *Legion* structure. In the

following, we discuss some applications of *Legion*:

1) m=1, *Leg*(1) ≡{C_1}: There is one parameter k_1=1. The only element C_1 of *Leg*(1) is a 1-coterie that can be applied to develop distributed algorithms to achieve mutual exclusion. Take the universal set U={1,2,3,4,5,6,7} as an example. The *Leg*(1) could be {{{1,2,4},{2,3,5},{3,4,6},{4,5,7},{5,6,1},{6,7,2},{7,1,3}}}. Any two quorums, for example {2,3,5} and {6,7,2}, have an intersection. By the intersection, mutual exclusion algorithm in distributed system can be implemented. In order to enter the critical section, a node needs to receive permissions from all nodes of an appropriate quorum. Since any two quorums have at least one common member and each node has only one permission, mutual exclusion is then guaranteed.

2) m=1, *Leg*(k) ≡{C_1}: This case is similar to *Leg*(1). There is still one parameter k_1=k, where 2≤k ≤n. The only element C_1 of *Leg*(k) is a k-coterie that can be applied to develop distributed algorithms to achieve k-mutual exclusion. By the property [I] of Definition 2, if less than k nodes are in the critical section, any other nodes can enter the critical section by gaining permissions from all nodes of an appropriate quorum. Moreover, by the property [II] of Definition 2, it is always guaranteed that no more than k nodes can enter the critical section at a time.

3) m=2, *Leg*(1, *null*) ≡{C_1, C_2}: There are two parameters: k_1=1 and k_2=*null*. The two elements, C_1, C_2 of *Leg*(1, *null*), are a 1-coterie and a set system, respectively. *Leg*(1, *null*) can be used to develop schemes for replica control in distributed database systems. Consider the universal set U={1,2,3,4,5,6,7,8,9}. The *Leg*(1, null) ≡ {C_1, C_2} could be {{{1,4,7,8,9},{2,5,8,9,1},{3,6,9,1,2},

{4,7,1,2,3},{5,8,2,3,4},{6,9,3,4,5},{7,1,4,5,6},{8,2,5,6,7},{9,3,6,7,8}},{{1,2,3},{2, 3,4},{3,4,5},{4,5,6},{5,6,7},{6,7,8},{7,8,9},{8,9,1},{9,1,2}}}. Any two quorums of C_1, for example {2,5,8,9,1} and {6,9,3,4,5}, have an intersection. On the other hand, any two quorums of C_2 may be disjoint, for example {3,4,5} and {7,8,9}. But, two quorums belonging to C_1 and C_2 respectively, for example {3,6,9,1,2} and {5,6,7}, should have an intersection. The properties of the pair (C_1, C_2) are similar to those of the bicoterie proposed in [36]. Each quorum in C_1 can be assigned as a write quorum. Meanwhile, each quorum in C_2 can be assigned as a read quorum. By the definition, we ensure that any two write quorums or any pair of read and write quorums have at least one common member.

4) m=2, *Leg*(*null*, *null*) ≡{C_1, C_2}: This type of structure has two identical parameters k_1=k_2=*null*. Two elements, C_1, C_2, of *Leg*(*null*, *null*) are both set systems. Hence, we do not care about the relation between the quorums in each set system, but we should note that any two quorums of the pair (Q_i, Q_j) are joint set, where Q_i is a quorum in C_1 and Q_j is a quorum in C_2. This structure can be applied to develop a location management scheme for mobile systems. In mobile systems, if one of the servers requires information from the other, it suffices to query one server from an appropriate quorum. While using this quorum-based location scheme, we can assign quorums in C_1 as update-quorums, and quorums in C_2 as query-quorums. According to the definition of *Legion*, the set of queried servers is bound to contain at least one server that belonged to the quorum that received the latest update (i.e., the update-quorum and query-quorum have at least one common server). In the next section, based on the *Leg*(*null*, *null*) structure, we will propose a $\sqrt{n}$ quorum-based

location management scheme for mobile network systems.

Those applications discussed above can be mapped on the dimension and set system type (Fig. 2-1). In Fig. 2-1, the horizontal axe is the set system type that could be *null*, 1,2,…,*n*, as the parameter k_i defined in Definition 3; the vertical axe is the dimension that is the number of parameters, *m*, also defined in Definition 3. We will continue to study and find more applications that do not appear on the mapping.

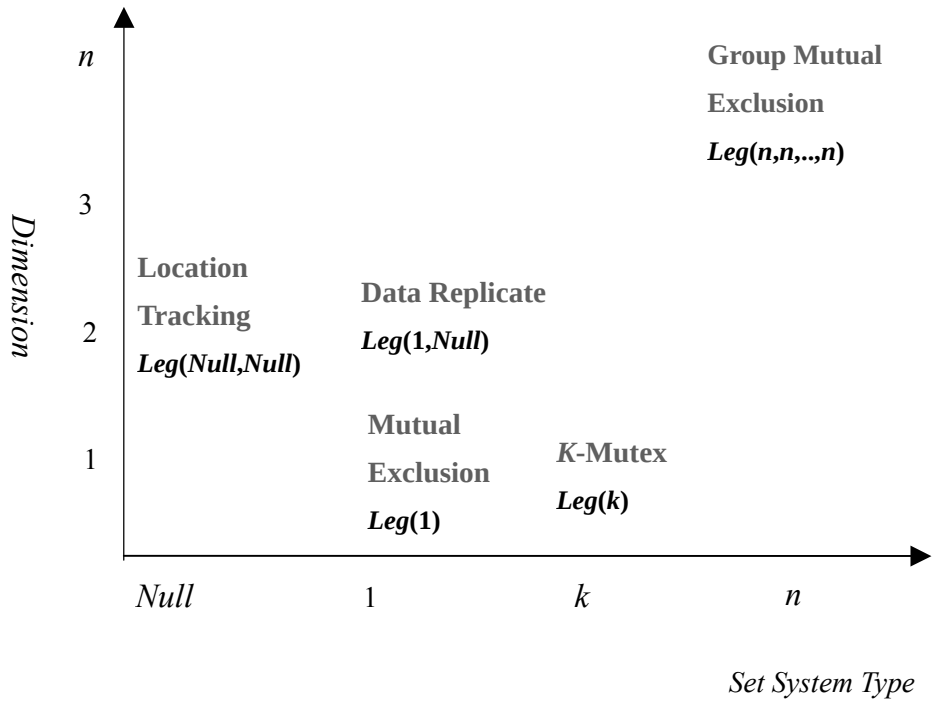


Fig. 2-1. The applications of *Legion* mapped on the dimension and set system type.

2.4. A Location Management Scheme Based on *Leg*(*null*, *null*)

In this section, we propose a simple quorum-based location management scheme constructed using *Leg*(*null*, *null*), which was discussed in section 2.3. We also make a claim in the following.

CLAIM 1. The *Legion* structure *Leg*(*null*, *null*) defined in Definition 3 can be used as a mathematic model for quorum-based location management in PCS networks.

Assume *Leg*(*null*, *null*) $\equiv\{C_i, C_j\}$. According to the Definition 3, any two quorums of a pair (Q_s, Q_t) have at least one common element, where Q_s is a quorum in C_i and Q_t is a quorum in C_j. We can assign quorums in C_i as update-quorums, and quorums in C_j as query-quorums. Sine the update-quorum and the query-quorum should be joint, the newest location information can be extracted from the common server(s).

2.4.1 A Simple *LegRing* Scheme

Based on the properties of *Leg*(*null*, *null*), we use ring-based approach to construct a quorum scheme called *LegRing* in order to manage location information. First, *N* location servers (LSs) are arranged as a logical ring, denoted as *N*-*LegRing*. Every LS in the mobile systems is assigned a distinct number from 0 to *N*-1 and arranged according to its number sequentially. In the following, some sequences of patterns in the *N*-*LegRing* are employed as Update-quorum (U-quorum) and

Query-quorum (Q-quorum).

DEFINITION 4. In an *N-LegRing* system, the Update-set system *(*U-set*)* and Query-set system *(*Q-set*)* are defined as follow*:*

U-set = {{n, (n+1) mod N, (n+2) mod N, ..., (n+d-*1*) mod N} | 0≤ n ≤ N-1},

Q-set = {{n, (n+d) mod N, (n+2d) mod N, ..., (n+kd) mod N}| 0≤ n ≤ N-1, $k=\lfloor (N-1)/d \rfloor$ },

where $d = \lceil \sqrt{N} \rceil$; n, k, and N are all integers.

Each element of U-set and Q-set is called an U-quorum and a Q-quorum, respectively.

THEOREM 1. *The U-set and Q-set of an N-LegRing system defined in Definition 4 satisfies the properties of Leg*(*null, null*).

PROOF. According to Definition 3, the properties of *Leg*(*null*, *null*) are: (1) U-set and Q-set are set systems. (2) Any pair of U-quorum in U-set and Q-quorum in Q-set are joint. We need to prove these properties. First, according to Definition 4 and Definition 1, it is easy to see that U-set and Q-set are set systems. Second, we define the distance between an ordered pair of servers v_1, v_2 in the *N-LegRing* system as Dist(v_1, v_2)= v_2-v_1, if $v_1 \le v_2$; or v_2+N-v_1, if $v_1 > v_2$. We choose an arbitrary U-quorum $\{u_1, u_2, ..., u_d\}$={n, (n+1) mod N, (n+2) mod N, ..., (n+d-1) mod N}. It is obvious that this U-quorum consists of d consecutive servers in the *N-LegRing* system since any two adjacent servers of U-quorum have Dist(u_i, u_{i+1})=1, 1≤i≤ d-1. Now we choose another arbitrary

Q-quorum $\{v_1, v_2,\ldots, v_{k+1}\}=\{n, (n+d) \bmod N, (n+2d) \bmod N, ..., (n+kd) \bmod N\}$. The distance between any two adjacent servers of Q-quorum is $\text{Dist}(v_i, v_{i+1})=d$ $(1\le i\le k)$ and $\text{Dist}(v_{k+1}, v_1)\le d$. Since $\text{Dist}(v_i, v_{i+1})=d$ for any two adjacent servers v_i, v_{i+1} and $\text{Dist}(v_{k+1}, v_1)\le d$ in Q-quorum, and $u_1, u_2,\ldots, u_d$ are d consecutive servers, we can conclude that at least one server v_i in Q-quorum intersects with one server u_j in U-quorum, i.e., $v_i = u_j$, for some j $(1\le j\le d)$. This satisfies property (2). Hence, the U-set and Q-set of the *N-LegRing* system satisfy the properties of *Leg*(*null*, *null*). □

The purpose of the *LegRing* system is to construct update and query quorums of a service region of the mobile system before the system begins to run the update or query processes. By using this quorum-based approach, which is the construction of the *LegRing* system, the newest location information of the MH is stored in some location registers of the U-quorum. When a MH wishes to communicate with another MH, some location registers of the Q-quorum are queried. According to the definition of *Legion*, the Q-quorum is bound to contain at least one register that belongs to the U-quorum that received the latest update. Hence, the newest location information is extracted from the common register(s).

2.4.2 The Scheme with $\sqrt{N}$ Quorum Size

THEOREM 2. *The size of U-quorum defined in Definition 4 is* $\lceil\sqrt{N}\rceil$, *and the size of Q-quorum is* $\lceil\sqrt{N}\rceil$ or $\lfloor\sqrt{N}\rfloor$.

PROOF. According to Definition 4, it is not difficult to see that U-quorum = $\{n, (n+1) \bmod N, (n+2) \bmod N, ..., (n+d-1) \bmod N\}$ has the number of elements

(d-1)+1=d=$\lceil\sqrt{N}\rceil$. Similarly, Q-quorum = {n, (n+d) mod N, (n+2d) mod N, ..., (n+kd) mod N} has the number of elements k+1= $\lfloor(N-1)/d\rfloor$ +1, where k=$\lfloor(N-1)/d\rfloor$, d =$\lceil\sqrt{N}\rceil$ is defined in Definition 4. Hence, the size of Q-quorum is$\lceil\sqrt{N}\rceil$ or$\lfloor\sqrt{N}\rfloor$. □

According to the discussions of the fault-tolerance for quorum-related algorithms in [35], an N nodes system with S quorum size can tolerate up to N-S nodes failures. But S nodes failures may prohibit the forming of any quorum. Therefore, our *LegRing* construction can tolerate up to $N-\sqrt{N}$ nodes failures.

2.5 Symmetric Property and Load Balance

Quorum-based protocols introduce the concept of symmetry that makes it possible to distribute the overheads of the algorithm equally across the entire system. Ng and Ravishankar [37] identified the Symmetric Coterie Construction (*SCC*) problem. Using their concept, we define the Symmetric Set System (*SSS*).

DEFINITION 5. (*SSS*) *Given a finite set* $S=\{0,1,...,N-1\}$ *representing the nodes of a network, find* $N=|S|$ *subsets* $Q_i \subseteq S$, $Q_i \neq \varnothing$ *such that:*

[I] (*Covering*) $\mathrm{U}_{i=0}^{N-1} Q_i = S$.

[II] (*Minimality*) $Q_i \not\subset Q_j$, $0 \le i, j \le N-1$, $i \neq j$.

[III] (*Equal size*) $|Q_i|=k$, $0 \le i \le N-1$.

[IV] (*Equal responsibility*) $|\{Q_j | i \in Q_j\}|=k$, $0 \le i \le N-1$.

Properties [III] and [IV] enforce that all nodes perform an equal amount of work.

THEOREM 3. *The U-set and Q-set of an N-LegRing system as defined in Definition 4 satisfy the properties of the Symmetric Set System* (*SSS*).

PROOF. First, we prove that the U-set of an *N-LegRing* system satisfies the properties of *SSS*. [I] (Covering): From Definition 4, U-set = {{n, (n+1) mod N, (n+2) mod N, ..., (n+d-1) mod N} | 0≤ n ≤ N-1}, we let Q_i ={i, (i+1) mod N, (i+2) mod N, ..., (i+d-1) mod N}(i=0,1, ...,N-1). Since i=0,1, ...,N-1 and each element in Q_i is mode N, the numbers from 0,1,…,N-1 are covered in some Q_i, and no element in Q_i is larger than N-1. Hence, the union of all Q_i covers all the numbers (elements) {0,1, ...,N-1} in an *N-LegRing* system. [II] (Minimality): Since each element Q_i (0≤ i ≤ N-1) in the U-set has d consecutive elements from i, it has at least one different element for any pair of Q_i and Q_j ($i \neq j$). Hence, $Q_i \not\subset Q_j$, 0≤ $i, j \leq N$-1 , $i \neq j$. [III] (Equal size): According to Definition 4, we know that each Q_i (0≤ i ≤ N-1) in the U-set has d elements. Therefore, all the sets Q_i have the same size. [IV] (Equal responsibility): If we look at all the first elements of all Q_i (i=0,1, ...,N-1), we can find that the composition of all the first elements corresponds to the numbers 0,1, ...,N-1. Similarly, the composition of all the second, third, ..., or d^{th} also corresponds to the numbers 0,1, ...,N-1. Hence, among Q_i (0≤ i ≤ N-1), each server i (0≤ i ≤ N-1) is included d times. The proof of the Q-set is similar to that of the U-set. □

According to Theorem 3, we can say that the size of the quorums in the U-set or Q-set is the same, and that each server in this system is included in the same number of quorums in the U-set or Q-set.

We draw a layer relation among the quorum system, coterie, set system, and

Legion (Fig. 2-2). Based on quorum system, our *Legion* structure integrates all the system and can be used to describe and develop many schemes for distributed applications, such as update-query, read-write, mutual exclusion, *k*-mutex, and Group mutual exclusion, etc.

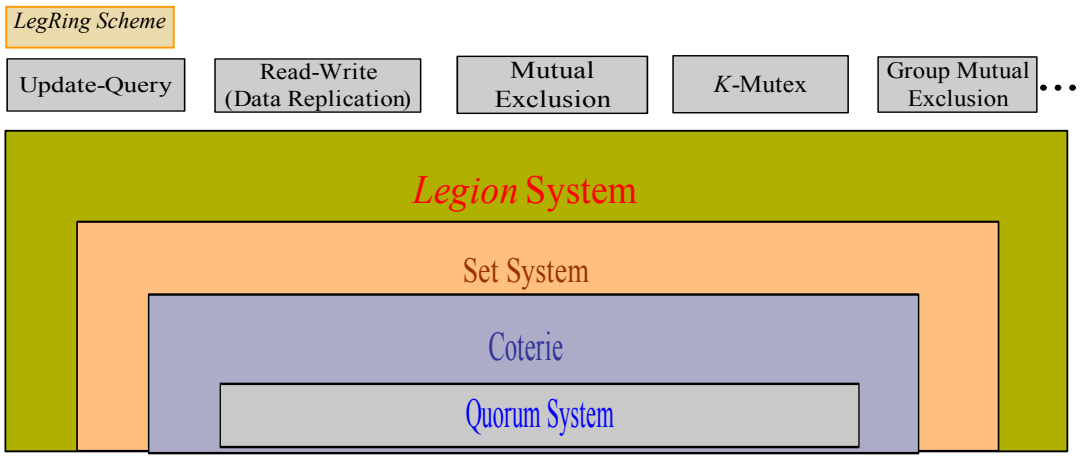


Fig. 2-2. System structure based on the quorum system.