2002 International Computer Symposium (ICS2002)

Workshop on Computer Networks

Legion Structure for Quorum-Based Location Management in Mobile Computings

Ming-Jeng Yang, Yao-Ming Yeh, and Yao-Ming Chang
Dept. of Information & Computer Education
National Taiwan Normal University
Taipei, Taiwan
E-mail: {mjyang, ymyeh, lming}@ice.ntnu.edu.tw

Abstract-- An important issue in the design of mobile computing systems is the efficient management of location information. In this paper, we propose a theory of *Legion* structure that can be used to construct some schemes of distributed applications, such as location management, information dissemination, mutual exclusion, etc. We also present a new and simple distributed quorum-based location management scheme – *LegRing*, which is developed from the theory of *Legion* structure. With small quorum size $\sqrt{N}$ and symmetric property, the *LegRing* scheme can be extended to a fault tolerant and load balanced location management algorithm. Also, it is applicable to distributed mobile platforms with any arbitrary numbers of nodes.

Index Terms-- Mobile computing, location management, legion, quorum, coterie, fault tolerance.

All correspondence could be addressed to Ming-Jeng Yang,
Parallel and Distributed Processing Laboratory,
Department of Information and Computer Education,
National Taiwan Normal University,
162 Hoping E. Rd., Sec. 1, Taipei,
Taiwan, R.O.C.

E-mail: mjyang@ice.ntnu.edu.tw or mjyang@ieee.org
Phone: 886-2-23944288 Ext. 60 Cel l Phone: 0968035278

1 Introduction

Advances in network technology have provided extensive use of portable computers and enabled the on-line services through wireless communication channels. This kind of computing paradigm is called mobile computing. In mobile computing systems, the mobile host (MH) means that its movement could cause changes in the physical topology of the network with time. The mobility of some hosts in the network raises an important issue in the management of location information. The location of a mobile host must be identified before the call to the mobile host can be established.

Location management includes location update and location query. When a mobile host changes its location, it should inform one or some location server(s) of its position. On the other hand, when a mobile host wishes to communication with the other host whose location is unknown, the query sequence is invoked.

Several schemes for mobile location management can be found in the literature. These include both centralized [1,2] and distributed schemes [3,4]. Each scheme has its advantages and shortages. A centralized scheme is simpler to implement and manage, but it is neither robust, nor scalable. The distributed scheme provides fault tolerance, load balance, scalability and modularity at the expense of increased control traffic and connection delay.

The two most commonly used standards, IS-41 [1] and GSM MAP [2], use centralized location management schemes. They utilize *home location registers* (HLR) and *visitor location registers* (VLR) to keep track of the location information of the MHs. The newest location information of the MHs is recorded in the HLR/VLR databases through the location update procedure. When a call to a mobile host is made, the information is retrieved from the HLR/VLR databases through the query procedure.

In Prakash's paper [5], a dynamic load-balanced location management scheme with an iterative and grid-based quorum construction is proposed. According to Prakash's theory, N location servers are divided into quorums of cardinality of $0.97N^{0.63}$ and $2\sqrt{N}$ -1 respectively. Any two quorums of servers have at least one common server. Upon receiving a location update request, the update

procedure uses hashing function, which takes the mobile's ID and its current location into account to select the quorum for update. When receiving a call request, the search procedure uses the hashing function with the caller's location and the called mobile's ID to select the quorum for query. From the common server, the finding of location information is guaranteed.

Ihn-Han Bae in [6] proposed a distributed location management scheme using the quorum, which is based on the triangle configuration of location servers. Without using virtual identity, the Bae's algorithm is not only simpler than Prakash's [5] algorithm but also cost-reduced. Since the quorum structure of triangle configuration is not symmetric, the Bae's scheme is neither load-based nor fully tolerant.

In this paper, we propose a new structure called *Legion* and then develop a simple distributed location management scheme from the new structure. This simple scheme, which is named *LegRing*, can achieve the efficient location updates and queries.

In the next section, the structure of system model in mobile computing is described. In section 3, we propose a new *Legion* structure and describe some of its mobile applications. In section 4, a new location management scheme and its algorithm are presented. Section 5 is the comparison between our scheme and other location management schemes. Finally, in section 6, we draw a conclusion to our research.

2 The System Model

Several existing systems of mobile computing network assume a cellular system of mobile nodes and stationary nodes. In this paper, the cellular system is modeled as a geographical area, which consists of a lot of hexagonal cells. A fixed base station, called *mobile support station* (MSS), supports each cell. The mobile support station is static and connected through dedicated wire-line link to an existing wired backbone. The mobile node, referred to as *mobile host* (MH), is a part of one and the only one cell at a time. Each mobile host can only communicate through the MSS of a cell with any particular node, whose position is located. The communication between MH and MSS is through radio waves or

infrared waves which are wireless. A *registration area* (RA) consists of several cells. Each registration area has one *location server* (LS) that maintains the location information of the mobile hosts in the RA [6]. All the location servers form a distributed location database in the mobile computing network (Fig. 1).

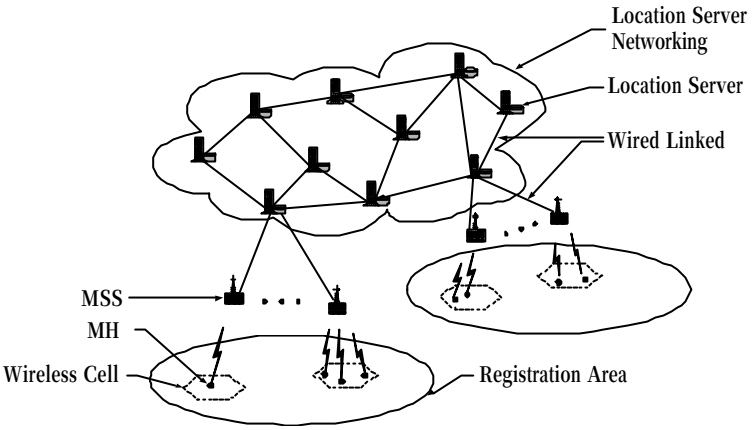

Fig.1. Logical view of a mobile network with distributed location database.

3 Legion Structure and Applications

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

In this section, we introduce the concepts of set system [7], quorum, and *k*-coterie [8]. A set system is composed of some quorums and could 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. [8] and Huang et al. [9] have defined *k*-coteries in 1991 and 1993.

DEFINITION 1. *A* ***set system*** [7] $C=\{Q_1, Q_2, \ldots, Q_n\}$ *is a collection of nonempty subsets* $Q_i \subseteq U$ *of a finite universe U.*

DEFINITION 2. *A* ***k-coterie*** [8] *is a nonempty set system that has the following properties:*

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

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

3.2 Legion Structure

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

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

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

[II] *$C_i = \{Q_1, Q_2, \ldots, Q_n\}$ is a set system and could be a k-coterie ($1 \leq k \leq n$)or not (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., $i \neq j$, $1 \leq i, j \leq m$).*

According to definition 3, a *Legion* has $m \geq 1$ parameters. These parameters could be *null*, 1, 2, ..., or n depending on the application that is applied, for example, *Leg*(1), *Leg*(5), *Leg*(1, *null*), etc. From the theory of *Legion* Structure, we could apply it to some applications of distributed computings. Someone

might develop new schemes according to the definition. We will discuss some applications of *Legion* Structure in the following section and, in section 4, propose a new scheme of mobile location management from the definition of *Legion*.

3.3 The Applications of Legion Structure

According to the different parameters and the number of parameters, we find some applications of *Legion* Structure. The following discussions are some applications represented by *Legion*:

1) m=1, *Leg*(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): This case is similar to *Leg*(1). There is still one parameter k_1=k, where $2 \le k \le n$. 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*): There are two parameters: k_1=1 and k_2=*null*. The two elements, C_1, C_2 of *Leg*(1, *null*), are an 1-coterie and a set system respectively. According to the definition 3, all quorums in C_1 are pairwise joint, while any two quorums in C_2 may have intersection or not. Moreover, any pair (Q_i, Q_j) of quorums is joint, where Q_i is a quorum in C_1 and Q_j is a quorum in C_2. 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. *Leg*(1, *null*) can be used to develop schemes for replica control in distributed database systems. In order to maintain replicated data in consistency, any two write operations or any pairs of read and write operations must access at least one common member. This control guarantees that every read operation always retrieves the latest update data and all write operations are free from conflicts. Since the first element C_1 of *Leg*(1, *null*) is a 1-coterie, each quorum in it could be assigned as a write quorum. Meanwhile, each quorum in the second element C_2 of *Leg*(1, *null*) could be assigned as a read quorum. By the definition, we ensure that any pair of read and write quorums have at least one common member.

4) m=2, *Leg*(*null*, *null*): This type of structure has two same parameters $k_1 = k_2 = null$. Two elements C_1, C_2 of *Leg*(*null*, *null*) are all set systems. Hence, we don't care the relation of quorums in each set system, but we should note that any two quorums of pair (Q_i, Q_j) are joint, 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 next section, according to the *Leg*(*null*, *null*) structure, we will propose a $\sqrt{n}$ quorum-based location management scheme for mobile network systems.

4 A Location Management Scheme from *Leg*(*null, null*)

In this section, we propose a simple quorum-based location management scheme constructed from *Leg*(*null*, *null*), which is discussed in section 3.3.

4.1 A Simple Scheme with $\sqrt{n}$ Quorum Size

From 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 by *N-LegRing*. Every LS in the mobile system is assigned a distinct number from 0 to *N*-1 and arranged by 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 = $\{\{n, (n+1) \bmod N, (n+2) \bmod N, ..., (n+d-1) \bmod N\} \mid 0 \le n \le N-1\}$

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

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 (Fig. 2).

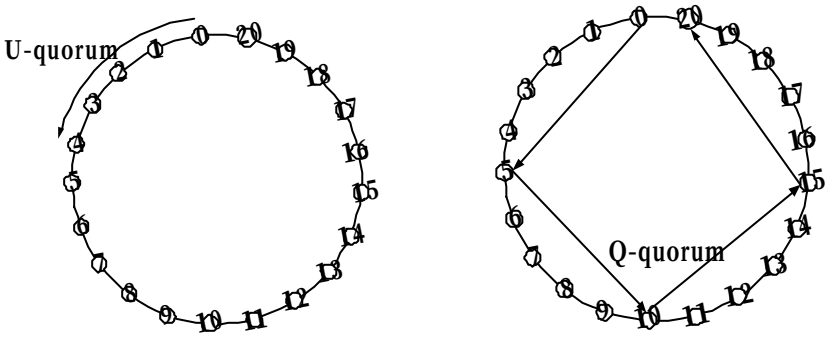

Figure 2. The construction of U-quorum and Q-quorum.

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

PROOF. According to the 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 of 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 $\{u_1, u_2, ..., u_d\}=\{n, (n+1) \bmod N, (n+2) \bmod N, ..., (n+d-1) \bmod N\}$. It is obvious that this U-quorum are d consecutive servers in the *N-LegRing* system, since any tow adjacent servers of U-quorum have Dist(u_i, u_{i+1})=1, $1 \le i \le d-1$. 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 of any two adjacent servers of Q-quorum is Dist(v_i, v_{i+1})=d ($1 \le i \le k$) and Dist(v_{k+1}, v_1)$\ge d$. Since Dist(v_i, v_{i+1}) d for any two adjacent servers v_i, v_{i+1} and Dist(v_{k+1}, v_1)$\ge d$ in Q-quorum, and u_1, u_2,…, 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 *N-LegRing* system satisfy the properties of *Leg*(*null*, *null*).

In cellular mobile systems, location management is achieved by query and update. A query occurs when a host needs to communicate with another mobile host whose location is unexpected and an update occurs when a mobile host changes its location. In a quorum-based scheme, the location information of a mobile host in an RA is stored in the local LS and replicated at all the LSs in the selected quorum. To exact the information from other LSs, the quorum of query server and the quorum of update server must be joint. Since the theorem 1 indicates the intersection property of U-quorum and Q-quorum, the U-Set and Q-Set in definition 4, a quorum-based scheme, could be used in location management.

In the following, we use the method of random selection [10] of quorums and timestamps with our *LegRing* location management scheme defined in 4 to implement the location update and query algorithms.

Location Update: When a mobile host h moves from one cell to the others, its location information

has to be updated. Therefore, the following steps of update procedure are performed:

Step 1. The UPDATE message associated with timestamp is stored to the cache of local LS and sent to all the LSs in the randomly selected quorum from U-set.

Step 2. Upon receiving the UPDATE message, the LSs add or overwrite the new location information received to their caches.

Location Query: When a mobile host *h* wishes to communicate with another host *h'* whose location is unknown, the query procedure is invoked with the following steps:

Step 1. First, host *h* queries the location information of *h'* from its local LS. If such information is found in LS's cache, the corresponding MSS is probed to determine if *h'* is still in the cell. If so, the call is connected and then the query procedure is stopped. Otherwise, the local LS clears the location information of host *h'* and then goes to step 2.

Step 2. The procedure randomly selects a quorum from Q-set and sends a QUERY message to all LSs in the quorum for the location information of *h'*.

Step 3. When receiving a QUERY for information, the LS, which has a copy of the queried information, sends a REPLY containing the timestamp associated with the location information. Otherwise, the LS sends a NULL reply.

Step 4. When receiving all the REPLY messages from all the LSs in the quorum, the procedure selects the location information with the latest timestamp, stores it in the local LS's cache, and returns the information to the MSS of mobile host *h*. According to the location information, the call to host *h'* can be connected.

Consider the *N-LegRing* system with 21 location servers. According to definition 4, the U-set and Q-set are constructed as following (Fig.3):

U-set={{0,1,2,3,4},{1,2,3,4,5},{2,3,4,5,6},{3,4,5,6,7},{4,5,6,7,8,},{5,6,7,8,9},{6,7,8,9,10},{7,8,9,10,11},{8,9,10,11,12},{9,10,11,12,13},{10,11,12,13,14},{11,12,13,14,15},{12,13,14,15,16},{13,14,15,16,

17},{14,15,16,17,18},{15,16,17,18,19},{16,17,18,19,20},{17,18,19,20,0},{18,19,20,0,1},{19,20,0,1,2},{20,0,1,2,3}}

Q-set={{0,5,10,15,20},{1,6,11,16,0},{2,7,12,17,1},{3,8,13,18,2},{4,9,14,19,3},{5,10,15,20,4},{6,11,16,0,5},{7,12,17,1,6},{8,13,18,2,7},{9,14,19,3,8},{10,15,20,4,9},{11,16,0,5,10},{12,17,1,6,11},{13,18,2,7,12},{14,19,3,8,13},{15,20,4,9,14},{16,0,5,10,15},{17,1,6,11,16},{18,2,7,12,17},{19,3,8,13,18},{20,4,9,14,19}}

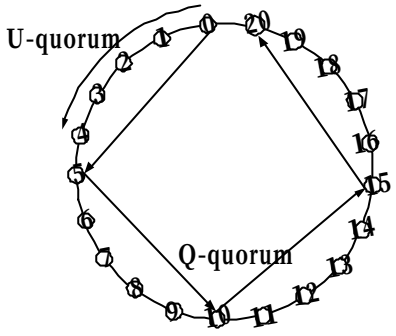

Figure 3. The construction of U-quorum and Q-quorum in a 21- *LegRing* system

If a mobile host *h* moves from one cell to another, new location information should be stored in the (new) local LS, for example server 4, and all the LSs, for example server 19,20,0,1,and 2, in the randomly selected U-quorum. When a mobile host wants to communicate with host *h* whose location information is not in the cache of local LS, for example server 17, it can require the information from the other LSs, for example server 1,6,11,16 and 0, in the randomly selected Q-quorum. Since quorum {19,20,0,1,2} and {1,6,11,16,0} have the common servers, 0 and 1, the newest location information of host *h* can be extracted through these servers.

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 the definition 4, it is not difficult to know that the U-quorum = {n, (n+1) mod N, (n+2) mod N, ..., (n+d-1) mod N} has the number of elements (d-1)+1=d=$\lceil\sqrt{N}\rceil$. Similarly, the

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 $\lfloor \sqrt{N} \rfloor$.

4.2 Symmetric Property, Load Balance, and Fault Tolerance

Quorum-based protocols introduce a concept of symmetry that serves to distribute the overheads of algorithm equally across the entire system. Ng and Ravishankar [11] 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*) $Q_i \not\subset Q_j$, $0 \leq i, j \leq N-1$, $i \neq j$.

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

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

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 defined in definition 4 satisfy the properties of 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≤ $n \leq N$-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 any element in Q_i is not 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≤ $i \leq N$-1) in 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, $Q_i \not\subset Q_j$, $0 \leq i, j \leq N-1$, $i \neq j$. [III] (Equal size): According to

definition 4, we know each Q_i ($0 \le i \le N-1$) in U-set has d elements. Therefore all the sets Q_i have same size. [IV] (Equal responsibility): If we look at all the first elements of all Q_i ($i=0,1, ...,N-1$), we could find that the composition of all the first elements is the numbers 0,1, ...,N-1. Similarly, the composition of all the second, third, ..., or d^{th} is also the numbers 0,1, ...,N-1. Hence, among Q_i ($0 \le i \le N-1$), each server i ($0 \le i \le N-1$) is included d times. The proof of Q-set is similar to U-set.

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

The responsibility of location tracking by using a deterministic quorum selection approach that a quorum is initially selected by a procedure is not guaranteed to be shared equally by all location servers. Hence, load unbalance arises when using the deterministic quorum-based approach that selects the quorum by geographical location of the mobile host. This is due to the following two major reasons mentioned in [12]. First, a significant fraction of mobile hosts may quite often concentrate in a very small area, while there are few mobile hosts in the rest area of the system. Second, even if all the mobile hosts are evenly distributed across the system, some hot mobile hosts may be queried more often and increase the loads of location servers of some quorum. By using random selection method in our quorum-based *LegRing* scheme, the implementation is given a better chance of load balancing among the location servers, since the selection of quorum is dynamic and each server bears even probabilities.

The quorum-based method is naturally fault tolerant if we use dynamic assignment of quorums. In the algorithm of section 4.1, the procedure can select another quorum if it does not receive all the REPLY information from all the servers of selected quorum for a given period of time, since some servers of queried quorum may crash. In order to tolerate the failures of servers, we could rewrite some steps in the algorithms of update and query in section 4.1 as following:

In **Location Update,** step 2 is rewritten into two steps:

Step 2. Upon receiving the UPDATE message, the LSs add or overwrite the new location information received to their caches and send back the ACK message.

Step 3. If the procedure does not receive all the ACK messages from all LSs in the quorum for a given period of time, then it randomly selects another quorum from U-set, sends the UPDATE message to all LSs in the new quorum again, and goes to step 2, else it stops.

In **Location Query**, step 4 is rewritten for fault tolerance as following:

Step 4. When receiving all the REPLY messages from all the LSs in the quorum, the procedure selects the location information with the latest timestamp, stores it in the local LS's cache, and returns the information to the MSS of the mobile host *h*. According to the location information, the call to host *h'* can be connected. If the procedure does not receive all the REPLY messages after a given period of time, then it goes back to step 2 (another loop to randomly reselect a new query quorum).

5 Comparison

A comparison of our scheme with some other quorum-based schemes is shown in table 1. In triangle configuration, the number of nodes $n(n+1)/2$ is equal to N, where n is an integer. Similarly, in grid based scheme, the number of nodes $m*n$ is equal to N, where m and n are integers. In the other two schemes, iterative and *LegRing*, the number of nodes n is equal to N. Obviously, in this property, only the iterative scheme and our *LegRing* scheme are applicable to the system with any arbitrary numbers of nodes.

	Quorum Size	Symmetric	Load Balance	Fault Tolerance	Number of Nodes
Triangle configuration [6]	$\sqrt{2N}$	No	No	Yes/Partial	$n(n+1)/2$
Dynamic hashing + Grid based scheme [5]	$2\sqrt{N}-1$	Yes	Yes	Yes	m*n
Dynamic hashing + Iterative approach [5]	$0.97N^{0.63}$	Yes	Yes	Yes	n
Random + *LegRing* scheme	$\sqrt{N}$	Yes	Yes	Yes	n

Table 1. Quorum-based location management schemes

Since the quorum structure of Bae's triangle configuration is not symmetric, the loads are heavier in the corner nodes of the triangle topology. Therefore, triangle configuration does not achieve load balance. Only grid based scheme, iterative approach, and our *LegRing* scheme are symmetric. With dynamic hashing, the grid based scheme and iterative approach can achieve load balance. With random selection of quorum, our *LegRing* scheme can achieve load balance.

In quorum-based location management scheme, when one node of selected quorum is fault, the update or query procedure will cause the information retrieval failure. The dynamic hashing or random selection of quorum could avoid the information retrieval failure, since they can reselect another quorum by re-executing hashing or random function. But, in triangle configuration, the selection of quorum according to the row or column cannot be achieved to a fully tolerant scheme.

Obviously, our *LegRing* scheme has the smallest quorum size among all of the schemes in table 1. Hence, the control traffic and connection delay can be reduced.

6 Conclusion

In this paper, we have proposed a structure named *Legion* for constructing some schemes of distributed applications, which include location management, information dissemination, mutual exclusion, etc.. We will continue to develop new distributed application schemes from the *Legio*n theory in the future.

In addition, we present a new simple quorum-based mobile location management scheme named

LegRing that is developed from the theory of *Legion* structure. As we have shown in the above discussion, our *LegRing* scheme has the smallest quorum size, which can reduce the message cost of communication. The other advantages of our scheme are: (1) It is applicable to distributed mobile platforms with any arbitrary numbers of nodes. (2) The quorum sets of a *LegRing* system satisfy the properties of symmetry. Therefore, every node in this system is included in the same number of quorums and bears the same responsibility if the quorums are selected evenly. (3) By using the method of random selection of quorums with our scheme, the algorithms can achieve fully distributed and fault tolerant location management.

Reference

[1] EIA/TIA, "Cellular Radio Telecommunications Intersystem Operation, PN-2991", Nov. 1995.

[2] M. Mouly, M.-B. Pautet, "The GSM System for Mobile Communications", Palaiseau, France, 1992.

[3] B. Awerbauch, D. Peleg, "Online Tracking of Mobile Users", Journal of the Association for Computing Machinery, 42(5), pp. 1021-1058, Sept. 1995.

[4] J. Z. Wang, "A Fully Distributed Location Register Strategy for Universal Personal Communication Systems", IEEE Journal on Selected Areas of Communication, vol. 11, n.6, pp. 850-860, Aug. 1993.

[5] R. Prakash, M. Singhal, "Dynamic Hashing + Quorum = Efficient Location Management for Mobile Computing Systems," Proc. ACM Symp. Principals Distributed Computing, p.291, Aug. 1997.

[6] Ihn-Han Bae, "A Quorum-Based Dynamic Location Management Method for Mobile Computings," RTCSA '99. Sixth Internat. Conf. on Real-Time Computing Systems and Applications, pp.398 – 401,1999.

[7] Moni Naor and Avishai Wool, "Access Control and Signatures via Quorum Secret Sharing," IEEE Trans. Parallel and Distributed Systems, vol. 9, no. 9, pp. 909-922, 1998.

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

[9] S.T. Huang, J.R. Jiang, Y.C.Kuo, "*K*-coteries for fault-tolerant *k* entries to a critical section," Proc. 13th IEEE Internat. Conf. on Distributed Computing systems, pp. 74-81, 1993.

[10] G. Karumanchi, S. Muralidharan, R. Prakash, "Information Dissemination in Partitionable Mobile Ad Hoc Networks," Proc. IEEE Symp. On Reliable Distributed Systems, Lausanne, Otc. 1999.

[11] Ng, W.K.; Ravishankar, C.V., "Coterie Templates: A New Quorum Construction Method," Proc. 15th IEEE Internat. Conf. on Distributed Computing Systems, pp. 92-99, 1995.

[12] R. Prakash, M. Singhal, "A Dynamic Approach to Location Management in Mobile Computing Systems," Proc. 8th Int. Conf. on Software Eng. & Knowledge Eng. SEKE '96, 488-495, Lake

Tahoe, Nevada, June 1996.