

Short Paper

Legion Structure for Quorum-Based Location Management in Mobile Computing*

MING-JENG YANG, YAO-MING YEH AND YAO-MING CHANG

Department of Information & Computer Education

National Taiwan Normal University

Taipei, 106 Taiwan

E-mail: {mjyang, ymyeh, lming}@ice.ntnu.edu.tw

An important issue in the design of mobile computing systems is the efficient management of location information. In this paper, we propose the Legion structure that can be used to construct schemes for 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 based on the Legion structure. With a small quorum size \sqrt{N} and the 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.

Keywords: mobile computing, location management, legion, quorum, coterie, fault tolerance

1. INTRODUCTION

Advances in network technology have led to extensive use of portable computers and enabled on-line services through wireless communication channels. This kind of computing paradigm is called mobile computing. In mobile computing systems, movement of the mobile host (MH) can cause changes in the physical topology of the network over 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 a 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 communicate with another host whose location is unknown, a query sequence is invoked.

Received January 31, 2003; accepted July 4, 2003.

Communicated by Shih-Pyng Shieh.

* A preliminary version of this paper has been presented at the 2002 International Computer Symposium, 2002.

Several schemes for mobile location management have been proposed in the literature. These include both centralized [1, 2] and distributed schemes [3, 4]. Each scheme has its advantages and disadvantages. 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 was proposed. In Prakash's scheme, 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 a hashing function, which takes the mobile's ID and its current location into account to select the quorum for updating. When it receives a call request, the search procedure uses the hashing function along with the caller's location and the called mobile's ID to select the quorum for querying. From the common server, finding location information is guaranteed.

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

In this paper, we propose a new structure called *Legion* and develop a simple distributed location management scheme based on this new structure. This simple scheme, called *LegRing*, can achieve efficient location updating and querying.

In the next section, the structure of system model for mobile computing is described. In section 3, we propose our 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 compares our scheme with other location management schemes. Finally, in section 6, we draw conclusions based on our research.

2. THE SYSTEM MODEL

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

MH and MSS occurs 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 networks (Fig. 1).

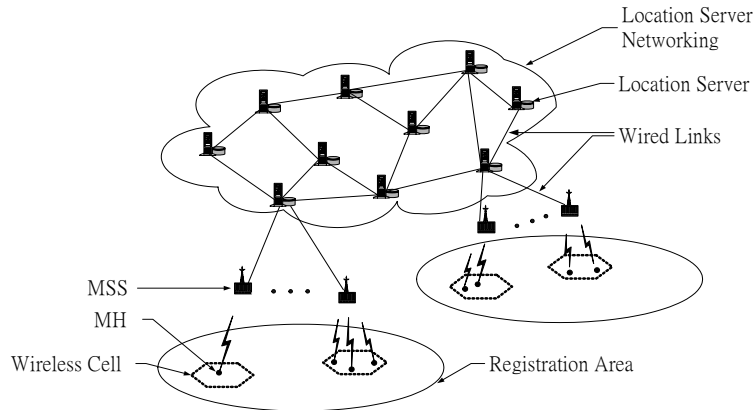


Fig. 1. Logical view of a mobile network with distributed location databases.

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 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. [8] and Huang et al. [9] defined k -coteries in 1991 and 1993.

Definition 1 A set system [7] $C = \{Q_1, Q_2, \dots, Q_n\}$ 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 $Q_1, Q_2, \dots, Q_h \in C$ 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 $Q_1, Q_2, \dots, Q_{k+1} \in C$, 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.

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

- [I] $C_i = \{Q_1, Q_2, \dots, Q_n\}$ is a k_i -coterie, if $1 \leq k_i \leq n$.
- [II] $C_i = \{Q_1, Q_2, \dots, Q_n\}$ is a set system and can be a k -coterie ($1 \leq k \leq n$) 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., $i \neq j$, $1 \leq i, j \leq m$).

According to Definition 3, a *Legion* has $m \geq 1$ 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 4, propose a new scheme for mobile location management based on the definition of *Legion*.

3.3 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) $m = 1$, $Leg(1) \equiv \{C_1\}$: 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.
- 2) $m = 1$, $Leg(k) \equiv \{C_1\}$: This case is similar to $Leg(1)$. There is still one parameter $k_1 = k$, where $2 \leq k \leq n$. 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.
- 3) $m = 2$, $Leg(1, null) \equiv \{C_1, C_2\}$: 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 sys-

tems. The properties of the pair (C_1, C_2) are similar to those of the bicoterie proposed in [10]. 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) $m = 2$, $Leg(null, null) \equiv \{C_1, C_2\}$: 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.

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 3.3.

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

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:

$$U\text{-set} = \{ \{n, (n + 1) \bmod N, (n + 2) \bmod N, \dots, (n + d - 1) \bmod N\} \mid 0 \leq n \leq N - 1 \},$$

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

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).

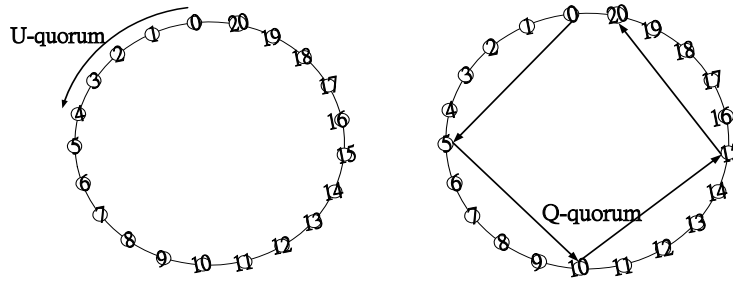


Fig. 2. The construction of U-quorum and Q-quorum.

Theorem 1 The U-set and Q-set of an N -LegRing system defined in Definition 4 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 $v_1 \leq v_2$; or $v_2 + N - v_1$, if $v_1 > v_2$. We choose an arbitrary U-quorum $\{u_1, u_2, \dots, u_d\} = \{n, (n + 1) \bmod N, (n + 2) \bmod N, \dots, (n + d - 1) \bmod N\}$. 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 $Dist(u_i, u_{i+1}) = 1$, $1 \leq i \leq d - 1$. Now we choose another arbitrary Q-quorum $\{v_1, v_2, \dots, v_{k+1}\} = \{n, (n + d) \bmod N, (n + 2d) \bmod N, \dots, (n + kd) \bmod N\}$. The distance between any two adjacent servers of Q-quorum is $Dist(v_i, v_{i+1}) = d$ ($1 \leq i \leq k$) and $Dist(v_{k+1}, v_1) \leq d$. Since $Dist(v_i, v_{i+1}) = d$ for any two adjacent servers v_i, v_{i+1} and $Dist(v_{k+1}, v_1) \leq d$ in Q-quorum, and u_1, u_2, \dots, 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 ($1 \leq j \leq d$). This satisfies property (2). Hence, the U-set and Q-set of the N -LegRing system satisfy the properties of $Leg(null, null)$. \square

In cellular mobile systems, location management is achieved by querying and updating. A query occurs when a host wants 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 extract the information from other LSs, the quorum of the query server and the quorum of the update server must be joint. Since Theorem 1 presents the intersection property of U-quorum and Q-quorum, the U-Set and Q-Set defined in Definition 4, a quorum-based scheme, can be used in location management.

In the following, we use the method of random selection [11] of quorums and time-stamps along with our *LegRing* location management scheme defined in Definition 4 to implement the location update and query algorithms.

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

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

- Step 1:** The UPDATE message associated with the timestamp is stored into the cache of local LS and sent to all the LSs in the randomly selected quorum from the 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 the 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 the LSs in the quorum for the location information of h' .
- Step 3:** When it receives 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 all the REPLY messages from all the LSs in the quorum are received, 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 an N -LegRing system with 21 location servers. According to Definition 4, the U-set and Q-set are constructed as follows (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 = $\{\{0, 5, 10, 15, 20\}, \{1, 6, 11, 16, 0\}, \{2, 7, 12, 17, 1\}, \{3, 8, 13, 18, 2\}, \{4, 9, 14, 19, 3\}, \{5, 10, 15, 20, 4\}, \{6, 11, 16, 0, 5\}, \{7, 12, 17, 1, 6\}, \{8, 13, 18, 2, 7\}, \{9, 14, 19, 3, 8\}, \{10, 15, 20, 4, 9\}, \{11, 16, 0, 5, 10\}, \{12, 17, 1, 6, 11\}, \{13, 18, 2, 7, 12\}, \{14, 19, 3, 8, 13\}, \{15, 20, 4, 9, 14\}, \{16, 0, 5, 10, 15\}, \{17, 1, 6, 11, 16\}, \{18, 2, 7, 12, 17\}, \{19, 3, 8, 13, 18\}, \{20, 4, 9, 14, 19\}\}.$

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, servers 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 the local LS, for example, server 17, it can acquire the information from the other LSs, for example,

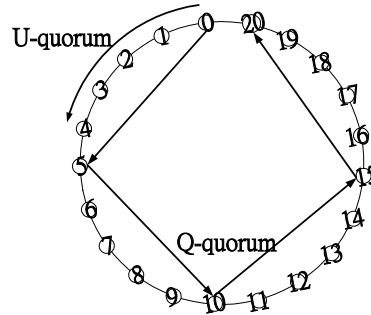


Fig. 3. The construction of U-quorum and Q-quorum in a 21-LegRing system.

servers 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 common servers, 0 and 1, the newest location information of host h can be extracted from 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 Definition 4, it is not difficult to see that $U\text{-quorum} = \{n, (n+1) \bmod N, (n+2) \bmod N, \dots, (n+d-1) \bmod N\}$ has the number of elements $(d-1)+1 = d = \lceil \sqrt{N} \rceil$. Similarly, $Q\text{-quorum} = \{n, (n+d) \bmod N, (n+2d) \bmod N, \dots, (n+kd) \bmod N\}$ 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 $\lfloor \sqrt{N} \rfloor$. \square

4.2 Symmetric Property, Load Balance, and Fault Tolerance

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 [12] identified the Symmetric Coterie Construction (SCC) problem. Using their concept, we define the Symmetric Set System (SSS).

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

- [I] (Covering) $\bigcup_{i=0}^{N-1} Q_i = S$.
- [II] (Minimality) $Q_i \not\subseteq Q_j$, $0 \leq i, j \leq N-1$, $i \neq j$.
- [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 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 = $\{\{n, (n + 1) \bmod N, (n + 2) \bmod N, \dots, (n + d - 1) \bmod N\} \mid 0 \leq n \leq N - 1\}$, we let $Q_i = \{i, (i + 1) \bmod N, (i + 2) \bmod N, \dots, (i + d - 1) \bmod N\}$ ($i = 0, 1, \dots, N - 1$). Since $i = 0, 1, \dots, N - 1$ and each element in Q_i is mode N , the numbers from $0, 1, \dots, N - 1$ 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) $\{0, 1, \dots, N - 1\}$ in an N -LegRing system. [II] (Minimality): Since each element Q_i ($0 \leq i \leq N - 1$) 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, $Q_i \not\subset Q_j$, $0 \leq i, j \leq N - 1$, $i \neq j$. [III] (Equal size): According to Definition 4, we know that each Q_i ($0 \leq i \leq N - 1$) 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 ($i = 0, 1, \dots, N - 1$), we can find that the composition of all the first elements corresponds to the numbers $0, 1, \dots, N - 1$. Similarly, the composition of all the second, third, ..., or d^{th} also corresponds to the numbers $0, 1, \dots, N - 1$. Hence, among Q_i ($0 \leq i \leq N - 1$), each server i ($0 \leq i \leq N - 1$) is included d times. The proof of the Q-set is similar to that of the U-set. \square

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.

The responsibility for location tracking using a deterministic quorum selection approach in which a quorum is initially selected by a procedure is not guaranteed to be shared equally by all the location servers. Hence, load unbalance arises when the deterministic quorum-based approach is used, which selects the quorum based on the geographical location of the mobile host. This occurs for the following two major reasons mentioned in [5]. First, a significant fraction of the mobile hosts may quite often be concentrated in a very small area, while there may be few mobile hosts in the remaining 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, thus increasing the loads of the location servers of some quorums. Because a random selection method is used in our quorum-based *LegRing* scheme, the implementation has a better chance of achieving load balance 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 presented in section 4.1, the procedure can select another quorum if it does not receive all the REPLY information from all the servers of the selected quorum during a given period of time since some servers of the queried quorum may crash. In order to tolerate the server failures, we can rewrite some steps in the update and query algorithms presented in section 4.1 as follows:

In **Location Update**, step 2 is rewritten as 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 the LSs in the quorum during 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; otherwise, it stops.

In **Location Query**, step 4 is rewritten so as to achieve fault tolerance as follows:

Step 4: When all the REPLY messages from all the LSs in the quorum are received, 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 has not received 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 the triangular 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, with this property, only the iterative scheme and our *LegRing* scheme are applicable to a system with any arbitrary numbers of nodes.

Table 1. Quorum-based location management schemes.

	Quorum Size	Symmetric	Load Balance	Fault Tolerance	Number of Nodes
Triangular 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 + <i>LegRing</i> scheme	\sqrt{N}	Yes	Yes	Yes	n

Since the quorum structure of Bae's triangular configuration is not symmetric, the loads are heavier in the corner nodes of the triangle topology. Therefore, the triangular configuration does not achieve load balance. Only the 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 quorums, our *LegRing* scheme can achieve load balance.

In the quorum-based location management scheme, when one node of the selected quorum is faulty, the update or query procedure will cause information retrieval failure.

Dynamic hashing or random selection of the quorum can avoid information retrieval failure since they can reselect another quorum by re-executing the hashing or random function. However, in the triangular configuration, the selection of quorums according to rows or columns cannot be achieved to a fully tolerant scheme.

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

6. CONCLUSIONS

In this paper, we have proposed a structure called *Legion* for constructing various schemes for distributed applications, which include location management, information dissemination, mutual exclusion, etc., we will continue to develop new distributed application schemes based on the *Legion* in the future.

In addition, we have presented a simple, new quorum-based mobile location management scheme called *LegRing* that was developed based on the *Legion* structure. As we have shown in the above discussion, our *LegRing* scheme has a small 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.

REFERENCES

1. EIA/TIA, "Cellular radio telecommunications intersystem operation," PN-2991, 1995.
2. M. Mouly and M. B. Pautet, *The GSM System for Mobile Communications*, France, Palaiseau, 1992.
3. J. Xie and I. F. Akyildiz, "A novel distributed dynamic location management scheme for minimizing signaling costs in mobile IP," *IEEE Transactions on Mobile Computing*, Vol. 1, 2002, pp. 163-175.
4. R. Shankaran, V. Varadharajan, and M. Hitchens, "A distributed location management scheme for mobile hosts," in *Proceedings of the 8th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, 2001, pp. 297-304.
5. R. Prakash and M. Singhal, "A dynamic approach to location management in mobile computing systems," in *Proceedings of the 8th International Conference on Software Engineering & Knowledge Engineering*, 1996, pp. 488-495.
6. I. H. Bae, "A quorum-based dynamic location management method for mobile computations," in *Proceedings of the 6th International Conference on Real-Time Computing Systems and Applications (RTCSA)*, 1999, pp. 398-401.
7. M. Naor and A. Wool, "Access control and signatures via quorum secret sharing," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 9, 1998, pp. 909-922.
8. S. Fujita, M. Yamashita, and T. Ae, "Distributed k -mutual exclusion problem and

- k -coterie,” in *Proceedings of the 2nd International Symposium on Algorithms, LNCS, Vol. 557, 1991, pp. 22-31.*
9. S. T. Huang, J. R. Jiang, and Y. C. Kuo, “ K -coterie for fault-tolerant k entries to a critical section,” in *Proceedings of the 13th IEEE International Conference on Distributed Computing Systems, 1993, pp. 74-81.*
 10. C. M. Lin, G. M. Chiu, and C. H. Cho, “A new quorum-based scheme for managing replicated data in distributed systems,” *IEEE Transactions on Computers, Vol. 51, 2002, pp. 1442-1447.*
 11. G. Karumanchi, S. Muralidharan, and R. Prakash, “Information dissemination in partitionable mobile ad hoc networks,” in *Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems, 1999, pp. 4-13.*
 12. W. K. Ng and C. V. Ravishankar, “Coterie templates: a new quorum construction method,” in *Proceedings of the 15th IEEE International Conference on Distributed Computing Systems, 1995, pp. 92-99.*

Ming-Jeng Yang (楊明正) received the M.S. degree in computer science from the Syracuse University, New York, in 1991. He is currently working toward the Ph.D. degree in the Department of Information and Computer Education, National Taiwan Normal University, Taiwan. His research interests include wireless networks, mobile computing, fault-tolerant computing, and distributed computing. He is a student member of the IEEE Computer Society and the ACM.

Yao-Ming Yeh (葉耀明) received the B.S. degree in computer engineering from National Chiao-Tung University, Taiwan, in 1981, and the M.S. degree in computer science and information engineering from National Taiwan University, Taiwan, in 1983. In August 1991, he received the Ph.D. degree in the Department of Electrical and Computer Engineering, The Pennsylvania State University, Pa., U.S.A. He is an associate professor in the Department of Information and Computer Education, National Taiwan Normal University, Taiwan. His research interests include fault-tolerant computing, web and XML computing, and distributed computing.

Yao-Ming Chang (張耀明) was born on June 15, 1974. He received the B.S. degree in 1996 from the Department of Computer Science and Information Engineering at TamKang University. He received his M.S. degree in 1999 from the Department of Information and Computer Education at National Taiwan Normal University. Now, he is pursuing Ph.D. degree in Information and Computer Education of National Taiwan Normal University. His research interests include parallel and distributed processing, fault-tolerant systems.