Journal of the Korean Data & Information Science Society
2012, **23**(6), 1259–1269

http://dx.doi.org/10.7465/jkdi.2012.23.6.1259
한국데이터정보과학회지

A GGQS-based hybrid algorithm for inter-cloud time-critical event dissemination[†]

Ihn-Han Bae[1]

[1]School of Information Technology Engineering, Catholic University of Daegu

Received 5 October 2012, revised 7 November 2012, accepted 13 November 2012

Abstract

Cloud computing has rapidly become a new infrastructure for organizations to reduce their capital cost in IT investment and to develop planetary-scale distributed applications. One of the fundamental challenges in geographically distributed clouds is to provide efficient algorithms for supporting inter-cloud data management and dissemination. In this paper, we propose a geographic group quorum system (GGQS)-based hybrid algorithm for improving the interoperability of inter-cloud in time-critical event dissemination service, such as computing policy updating, message sharing, event notification and so forth. The proposed algorithm first organizes these distributed clouds into a geographic group quorum overlay to support a constant event dissemination latency. Then it uses a hybrid protocol that combines geographic group-based broadcast with quorum-based multicast. Our numerical results show that the GGQS-based hybrid algorithm improves the efficiency as compared with *Chord*-based, *Plume* and GQS-based algorithms.

Keywords: Cloud federation, distributed clouds, inter-cloud, quorum system, time-critical event dissemination.

1. Introduction

Cloud computing, the new computation paradigm, is pursuing new levels of efficiency in delivering services, representing a tempting business opportunity for IT operators of increasing their revenues. Services range from software to platform or infrastructure (SaaS, PaaS, IaaS), while clients might include other clouds, organizations, enterprises and single users (Celesti *et al.*, 2010). Figure 1.1 shows cloud computing layers. There are three types of cloud providers that you can subscribe to: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) (Alvarez, 2011).

- IaaS - rents processing, storage, network capacity, and other fundamental computing resources, e.g. rent of space on the Internet,
- PaaS - deploys customer-created applications to a cloud, e.g. provision of developer tools on the Internet and

[†] This work was supported by research grants from the Catholic University of Daegu in 2012.
[1] Professor, School of Information Technology Engineering, Catholic University of Daegu, Gyeongbuk 712-702, Korea. E-mail: ihbae@cu.ac.kr

- SaaS -uses provider's applications over a network, e.g. use of an application via the Internet.

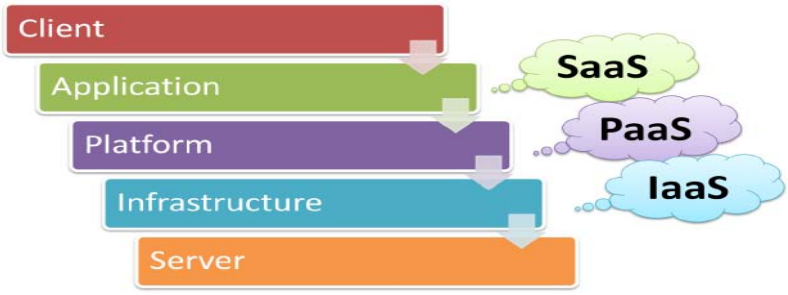

Figure 1.1 Cloud computing layers

Cloud service providers have to scatter geographically as many data-center containers to support modular data-centers that can provide predictable, repeatable components for the desired organizations. Moreover, IT organizations/enterprises, government/departments, universities may deploy their own clouds in the future (Decandia *et al.*, 2007). In the future, a lot of distributed clouds, including public and private clouds, will soon be built around the glove.

The inter-cloud scenario is based on the key concept that each single cloud does not have infinite physical resources. If a cloud saturates the computational and storage resources of its virtualization infrastructure, it could not be able to satisfy further requests for service allocations sent from its clients. The inter-cloud scenario aims to address such situation, in fact, each cloud can use the computational and storage resources of the virtualization infrastructures of other clouds. Such form of pay-for-use introduces new business opportunities among cloud providers. Nevertheless, the inter-cloud raises many challenges concerning cloud federation, security, interoperability, QoS, monitoring and billing (Majumdar and Garimella, 2012; Endo *et al.*, 2011).

One of the fundamental challenges in large-scale geographically distributed clouds is to provide efficient algorithms for supporting inter-cloud data management and dissemination. In other words, it is that inter-cloud applications/systems have to interact with a set of clouds via the Internet. Especially for time-critical services across a large geographic area, these service providers prefer to delivery the events as early as possible. Therefore, as the number of clouds continues to grow, the scale of distributed clouds may achieve a planetary-scale, with even thousand or hundred thousands of clouds. One of major challenges in these planetary-scale distributed clouds is to promptly disseminate the time-critical event to other clouds (Wu *et al.*, 2011).

A typical time-critical service example is the disaster warning system which has a disaster alert cloud for hosting the disaster early warning system. The alert notification for the system should be quickly disseminated to clouds to minimize the damage.

In this paper, we present a GGQS-based hybrid dissemination algorithm for improving the interoperability of inter-cloud in time-critical event dissemination service, such as computing policy updating, message sharing, event notification and so forth. The proposed GGQS-based

hybrid algorithm first organizes these distributed clouds into a group quorum overlay to support a constant event dissemination latency, then it uses a hybrid protocol that combines geographic group-based broadcast with quorum-based multicast. Our numerical results show that the GGQS-based hybrid algorithm improves the efficiency as compared with *Chord*-based, *Plume* and group quorum system (GQS)-based algorithms.

The remainder of this paper is organized as follows. Section 2 describes the background and the related works on disseminating time-critical message to distributed clouds. Section 3 designs the GGQS-based hybrid time-critical event dissemination algorithm. Section 4 presents performance results on our proposed GGQS-based hybrid, GQS-based, *Chord*-based and *Plume* methods. Section 5 concludes the paper and discusses future work.

2. Related works

To manage and deliver contents effectively, distributed clouds have to organize themselves into a federated inter-cloud overlay network. Currently, each large-scale cloud service provider uses its own approach to manage or deliver contents. For instance, Amazon's Dynamo leverages a fully-meshed topology for managing data across multiple data centers (Dikaiakos *et al.*, 2009). To meet the demand of time-critical services, this overlay requires a fundamental property, which is that scalability, constant dissemination delay and robustness in the inter-cloud overlay should be considered carefully. The fully-meshed design is the simplest way to disseminate time-critical message to each other nodes. This scheme takes constant transmission delay easily. However, each node in the fully-meshed design is required to maintain $O(n)$ connections with other nodes in a distributed system of n nodes. Thus this scheme does not scale well in large-scale distributed systems.

Chord (Stoica *et al.*, 2001) is a scalable system for node lookup in a dynamic peer-to-peer system with frequent node arrivals and departures. In this way it is similar to other distributed hash tables (DHTs). *Chord* its supports just one operation: given a key as part of a large, circular key space $(0.2^{160}-1)$, it maps the key to a node. It achieves this in a scalable manner by limiting the routing information each node needs to only a few other nodes. Because the routing table is distributed, a *Chord* node communicates with its neighbours in order to perform a lookup. In the steady state with n-node system, each node maintains information about only $O(logn)$ other nodes, and resolves all look-ups via $O(logn)$ messages to other nodes. *Chord* maintains its routing information as nodes join and leave the system.

Plume (Wu *et al.*, 2011) is a generic distributed inter-cloud overlay for time-critical event dissemination services. *Plume* organizes these distributed clouds into a novel quorum ring overlay to support a constant event dissemination latency. The *Plume* nodes can construct the quorum ring overlay (QRO) topology according to their ring edges and q-tables respectively. The size of ring edges plus q-table in some *Plume* nodes is $2\sqrt{n}-1$, where n represents the size of an inter-cloud network.

GQS-based algorithm (Bae, 2011) is based on the inter-cloud which organizes the nodes of distributed clouds into the inter-cloud overly based on GQS ring overlay (GQO) topology. In GQS-based algorithm, a sender cloud disseminates a time-critical event to all cloud nodes within the quorum of the sender cloud and the row and/or column clouds within two squares of the quorum. The size of the quorum is $4\sqrt{n/3}-1$.

3. Geographic GQS-based hybrid algorithm

3.1. System model

We consider a inter-clouds network with a set of geographically distributed clouds connected via the Internet, and each cloud has a cloud manager (resource management system), applications and a cloud node (provider) as shown earlier in Figure 1.1.

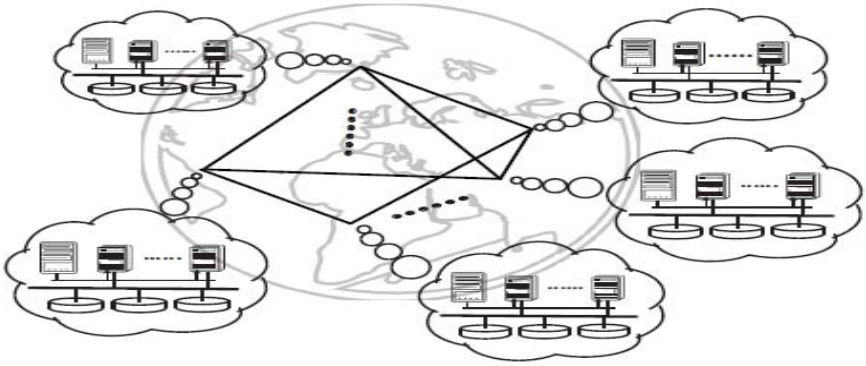

Figure 3.1 The geo-distributed cloud model

Federation of geo-distributed cloud services is a recent development of cloud computing technologies. The open data center alliance, for instance, aims to provide solutions to unify cloud resources from different providers to produce a global scale cloud platform. We consider a geo-distributed cloud infrastructure which consists of multiple disparate cloud sites distributed in different geographical locations, and owned by one or multiple cloud service providers. Each cloud site resides in one data center, and contains a collection of interconnected and virtualized servers.The cloud architecture is illustrated in Figure 3.1. If a time-critical event is occurred in a cloud, the event must be disseminated to all clouds via the inter-cloud overlay (Wu *et al.*, 2012).

Formally, an inter-cloud network can be represented an undirected graph $G = (V, E)$, where vertices V corresponding to cloud nodes and edges E corresponding to unidirectional overlay links. It is assumed that distributed clouds are uniquely identified. An edge $e(u, v) \in E$ indicates that u and v know each other and can communicate directly through the inter-cloud overlay.

3.2. GQS Construction

In a proposed algorithm, the dissemination operation for the time-critical event of a cloud node is performed based on GQS. GQS can be constructed from the unfolded surface quorum system with cloud nodes in a distributed clouds as shown Figure 3.1. The definition of GQS is as follows (Joung, 2003).

Definition Let P=$\{1, 2, \cdots, n\}$ be a set of nodes. An m-GQS $C = (C_1, C_2, \cdots, C_m)$ over P consists of m sets, where each $C_i \in 2^P$ is a set of subset of P satisfying the following properties:

intersection: $\forall 1 \leq i,\ j \leq m,\ i \neq j,\ \forall Q_1 \in C_i,\ \forall Q2 \in C_j : Q_1 \cap Q_2 \neq \phi$
minimality: $\forall 1 \leq i \leq m,\ \forall Q_1,\ Q_2 \in C_1,\ Q_1 \neq Q_2 : Q_1 \nsubseteq Q_2$
Each C_i is called a *cartel* and each Q is called a *quorum*.

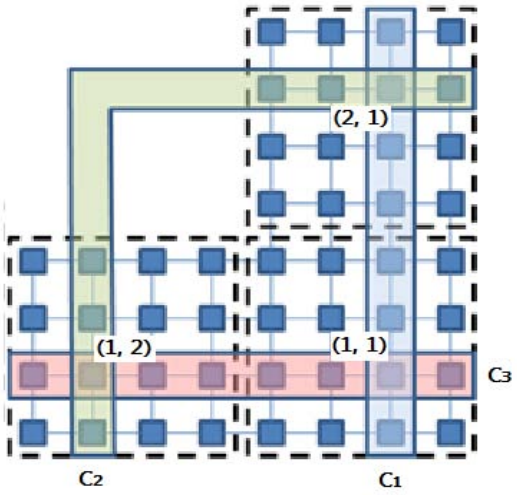

Figure 3.2 Unfolded surface quorum system S_3

If we spread three sides of the cubic on a plane, the 3-group quorum system, $S_3 = (C_1, C_2, C_3)$ can be constructed as shown in Figure 3.2. Each quorum in C_1 corresponds to the vertical line crossing the right row of the square. Moreover, each quorum in C_2 corresponds to the horizontal line across the top square, and a vertical line across the left square on the bottom. Finally, each quorum in C_3 corresponds to the horizontal line cross the two squares on the bottom.

Each C_i in S_3 needs 2 squares, each of which is to be shared with one of the other 2 cartels so that the corresponding lines of the two cartels intersect at exactly one node on the square. Overall, there are 3 squares. Let k be the width of each square, Then, each square consists of k^2 nodes, the total number of nodes on the 3 squares is $3k^2$. Let $3k^2 = n$, where n represents the total number of cloud nodes composing a distributed clouds. Therefore, $k = \sqrt{n/3}$. The size of quorum is $q = 2k = 2\sqrt{n/3}$.

3.3. Event dissemination algorithm

The proposed event dissemination algorithm is based on the inter-cloud which organizes the nodes of distributed clouds into the inter-cloud overlay based on GGQS overlay topology. In the GGQS, we can construct 3-GQS, $S_3 = (C_1, C_2, C_3)$ by the staircase construction of the surficial quorum system as shown in Figure 3.2.

Figure 3.3 shows a GGQS overlay which consists of 12 cloud nodes and 3 squares, where each square that is a geographic group consists of 4 cloud nodes and the dimension of squares is 2. In the all squares of the GGQS, the cloud nodes are arranged in the elements of the squares by row-wise.

In Figure 3.3, the set of cloud nodes within the geographic group of cloud node 3, G_3={1, 2, 3, 4}, and the quorum of cloud node 3, Q_3 is consisted of C_1 and C_3, where C_1={1, 3, 9, 11} and C_3={3, 4, 7, 8}. Cloud node 3 disseminates a time-critical event (e) to all cloud nodes within $G_3 \cup Q_3$ with calling $send(e, 3, p)$, $\forall p \in G_3 \cap Q_3$. Then, the cloud nodes 1, 2 and 4 that are located in the same geographic group with cloud 3 receive the event from the cloud 3. Also, the cloud nodes 9 and 11 receive the event from the cloud node 3 and relay the event to the cloud nodes $10 \in Row_9$ and $12 \in Row_{11}$. The cloud nodes 7 and 8 receive from the cloud node 3 and relay the event to the cloud nodes $5 \in Col_7$ and $6 \in Col_8$, where Row_i and Col_i represent the row entries and the column entries in the square that includes cloud node i, respectively. Therefore, the proposed GGQS-based hybrid event dissemination algorithm ensures that any cloud node only needs at most 2 hops to reach any other cloud nodes in the distributed clouds. From this instance, we know that every event relays of the received nodes are bounded on their local square. Thus, the dissemination latency of the event relay will be shorten.

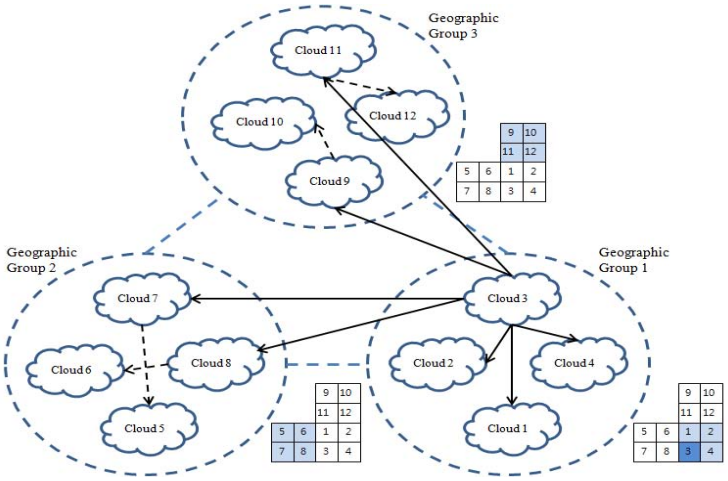

Figure 3.3 An Example of GGQS overlay of 12 cloud nodes

We design a GGQS-based hybrid event dissemination algorithm, in which time-critical events are disseminated to all cloud nodes via the GGQS overlay topology. The inter-cloud which is based on GGQS overlay topology can typically provides quick event dissemination in large-scale distributed clouds.

The event dissemination algorithm can be invoked by any cloud node to initiate a dissemination operation. The event dissemination algorithm is represented in Algorithm 3.1, where a cloud node s requires to disseminate an event e to every other nodes in a distributed clouds, and G_s represents the set of cloud nodes within the geographic group of the sender s. Algorithm 3.2 represents the $relay(e)$ operation in a cloud node r that receives the event e from s, where $SQ(i, j)$represents the square (i, j) of GGQS.

Algorithm 3.1 *Dissemination*(e) algorithm

```
/* (i, j) represents the index of the square that sender s is located */
if (i == j) then
    for all cloud nodes p ∈ G_s ∪ C_1 ∪ C_3 do
        send(e, s, p);
    end for
else if (i > j) then
    for all cloud nodes p ∈ G_s ∪ C_1 ∪ C_2 do
        send(e, s, p);
    end for
else
    for all cloud nodes p ∈ G_s ∪ C_2 ∪ C_3 do
        send(e, s, p);
    end for
end if
```

Algorithm 3.2 *Relay*(e) algorithm

```
/* Cloud node r receives an event e from s */
if (s ∈ SQ_(1,1)) then
  for r ∈ C_1 ∩ SQ_(2,1) do
    for all cloud nodes q ∈ Row_r do
        send(e, r, q);
    end for
  end for
  for r ∈ C_3 ∩ SQ_(1,2) do
    for all cloud nodes q ∈ Col_r do
        send(e, r, q);
    end for
  end for
else if (s ∈ SQ_(2,1)) then
  for r ∈ C_1 ∩ SQ_(1,1) do
    for all cloud nodes q ∈ Row_r do
        send(e, r, q);
    end for
  end for
  for r ∈ C_2 ∩ SQ_(1,2) do
    for all cloud nodes q ∈ Row_r do
        send(e, r, q);
    end for
  end for
else
  for r ∈ C_2 ∩ SQ_(2,1) do
    for all cloud nodes q ∈ Col_r do
        send(e, r, q);
    end for
  end for
  for r ∈ C_3 ∩ SQ_(1,1) do
    for all cloud nodes q ∈ Col_r do
        send(e, r, q);
    end for
  end for
end if
```

4. Performance evaluation

In this section, we present the performance of the proposed GGQS-based event dissemination algorithm through analytical model. The performance is evaluated in terms of an average dissemination delay. In a stable inter-cloud networks of n nodes, the sender cloud node disseminates the events to the set of cloud nodes within its geographic group and the two quorums of its GGQS with one hop, The size of geographic group is $n/3$, and the size of two quorums which exclude from the nodes within geographic group is $2\sqrt{n/3}$. Accordingly, the number of cloud nodes that receives the event with one hop is $n/3 + 2\sqrt{n/3} - 1$. Therefore, the cloud nodes which exclude from the nodes of two quorums receive the event with two hops, the total number of the cloud nodes is $2n/3 - 2\sqrt{n/3} + 1$. Thus, the total dissemination delay in the GGQS-based hybrid algorithm is approximately equal to $1 \times \left(n/3 + 2\sqrt{n/3}\right) + 2 \times \left(2n/3 - 2\sqrt{n/3} + 1\right) = 5n/3 - 2(\sqrt{n/3} - 1)$ and the average dissemination delay of the GGQS-based hybrid algorithm is $\left[5n/3 - 2\left(\sqrt{n/3} - 1\right)\right] / (n-1)$.

Figure 4.1 shows the results of the performance evaluation of *Chord*-based method and our GGQS-based hybrid method in terms of average dissemination delay over number of nodes in distributed clouds. As shown in Figure 4.1, the average dissemination delay of GGQS-based hybrid method does not grow with the number of nodes in distributed clouds. However, the average dissemination operation of *Chord*-based method requires $log\ n/2$ number hops to delivery event to other clouds (Stoica *et al.*, 2001; Terpstra *et al.*, 2003). Therefore, the average dissemination delay of GGQS-based hybrid method is much lower than that of *Chord*-based method regardless of the number of nodes in distributed clouds.

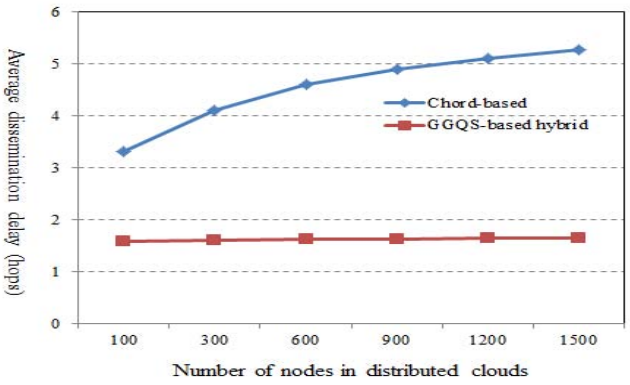

Figure 4.1 A comparison of average dissemination delays in *Chord*-based and GQS-based methods

Figure 4.2 plots the results of the performance evaluation of *Plume*, GQS-based and our GGQS-based hybrid methods in terms of average dissemination delay over number of nodes in distributed clouds. Where the average dissemination delay of *Plume* and GQS-based methods in a stable inter-cloud network of n nodes is equal to $(2n - 2\sqrt{n} - 1)/(n - 1)$ and $(2n - 4\sqrt{n/3} - 1)/(n - 1)$, respectively (Wu *et al.*, 2011; Bae, 2011). Therefore, the average dissemination delay of GGQS-based hybrid method is little lower than that of *Plume* and GQS-based methods regardless of the number of nodes in distributed clouds.

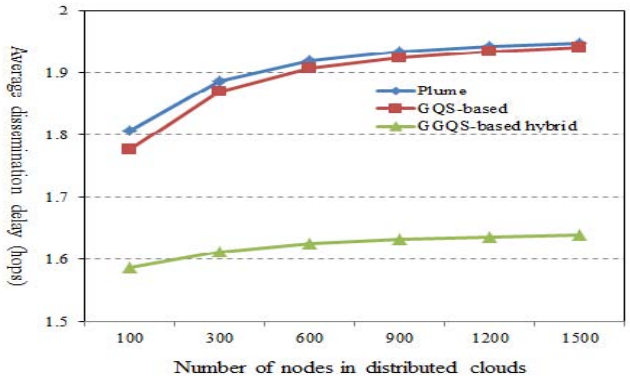

Figure 4.2 A comparison of average dissemination delays in *Plume*, GQS-based and GGQS-based hybrid methods

Table 4.1 shows the number of event receiving nodes for hops of quorum-based event dissemination methods which are similar with GGQS-based hybrid method, where the number of distributed cloud nodes is set to 1,024. The average event dissemination hops of proposed GGQS-based hybrid method are smaller than *Plume* and GQS-based methods because the number of event receiving nodes through one hop by the GGQS-based hybrid method is larger than *Plume* and GQS-based methods.

Table 4.1 The number of event receiving nodes for number of hops (n: 1,024)

Algorithms	Number of hops	The number of event receiving nodes	Average event dissemination hops
Plume	1	63	1.938
	2	960	
GQS-based	1	73	1.929
	2	950	
GGQS-based hybrid	1	378	1.631
	2	645	

When analysis of categorical data is concerned with more than one variable, two-way tables are employed. These tables provide a foundation for statistical inference, where statistical tests question the relationship between the variables the basis of the data observed. We can present data on two categorical variables in two-way tables of counts. Table 4.2 is the two-way table classifying the nodes by their algorithm and by number of hops.

Table 4.2 The number of event receiving nodes by algorithm and hop (n: 1,024)

Algorithms	Number of hops		Total
	1	2	
Plume	63 (6.2%)	960 (93.8%)	1,023 (33.3%)
GQS-based	73 (7.1%)	950 (92.9%)	1,023 (33.3%)
GGQS-based hybrid	378 (37%)	645 (63%)	1,023 (33.3%)
Total	514 (16.7%)	2,555 (83.3%)	3,069 (100%)

The chi-square test provides a method for testing the association between the row and column variables in the two-way table. The null hypothesis H_0 assumes that there is no association between the variables, while the alternative hypothesis H_a claims that some association does exist. The chi-square test is a measure of how far the observed values in two-way table are from the expected values. The formula for chi-square test is as follows (Moore, 2009):

$$\chi^2 = \sum \frac{(observed\ value - expected\ value)^2}{expected\ value} \tag{4.1}$$

This requires calculation of the expected values based on the data, where the expected values of one hop and two hops in Table 4.2 are 171.3 and 851.7, respectively. The chi-square test statistic for Table 4.2 is computed by equation (4.1).

$$\chi^2 = \frac{(63 - 171.3)^2}{171.3} + \frac{(960 - 815.7)^2}{851.7} + \cdots + \frac{(645 - 815.7)^2}{851.7} = 449.5$$

The degrees of freedom are equal to 2, so we are interested in the probability $P(\chi^2 \geq 449.5) = 0.0$. This indicates that there is strong association between event dissemination algorithm and number of hops.

From these results, we know that the proposed GGQS-based hybrid method reduces the average dissemination delay of time-critical events is reduced because number of hops which the events pass through is small, and provides a suitable method for time-critical event dissemination in large-scale distributed clouds.

5. Conclusions

Development of fundamental techniques and software systems that integrate distributed clouds in a federated fashion is critical to enabling composition and deployment of elastic application services. We believe that outcomes of this research vision will make significant scientific advancement in understanding the theoretical and practical problems of engineering services for federated environments. The resulting framework facilitates the federated management of system components and protects customers with guaranteed quality of services in large, federated and highly dynamic environments. The different components of the proposed framework offer powerful capabilities to address both services and resources management, but their end-to-end combination aims to dramatically improve the effective usage, management, and administration of cloud systems. This will provide enhanced degrees of scalability, flexibility, and simplicity for management and delivery of services in federation of clouds.

One of the fundamental challenges in large-scale geographically distributed clouds is to provide efficient algorithms for supporting inter-cloud data management and spread. Accordingly, we presented the GGQS-based hybrid event dissemination algorithm for improving the interoperability of inter-cloud in time-critical event dissemination service. The proposed GGQS-based hybrid algorithm first organizes these distributed clouds into a group quorum overlay to support a constant event dissemination latency, then it uses a hybrid protocol that combines geographic group-based broadcast with quorum-based multicast. The performance of the proposed GGQS-based hybrid event dissemination method has been evaluated through an analytical model. From the results of the analytical evaluation, we knew that the average dissemination delay of GGQS-based hybrid method was lower than those of *Chord*-based, *Plume* and GQS-based methods regardless of the number of nodes in distributed clouds. Therefore, the performance of the GGQS-based hybrid method was superior to those of *Chord*-based, *Plume* and GQS-based methods. Also, the proposed GGQS-based hybrid method provided a suitable technique for time-critical event dissemination in large-scale distributed clouds.

Our future works include a cloud-based event system for managing context information in ubiquitous services and mobile applications. This approach will use an event service to manage the events representing local and remote changes in the environment, allowing relevant information to be shared amongst interested services and mobile users.

References

Alvarez, C. A. M. (2011). *Cloud computing: Concerns and challenges for its adoption in SMEs and large companies in Japan*, Master Thesis, Graduate School of Management, Ritsumeikan Asia-Pacific University, Japan.

Bae, I-H. (2011). Design and evaluation of a GQS-based time-critical event dissemination for distributed clouds. *Journal of the Korean Data & Information Science Society*, **22**, 989-998.

Celesti, A., Tusa, F., Villari M. and Puliafito, A. (2010). How to enhance cloud architectures to enable cross-federation. *IEEE 3rd International Conference on Cloud Computing*, 337-345.

DeCandia, G., Hastorun, D., Jamd, M. and Vogels, W. (2007). Dynamo: Amazon's highly available key-value store. *Proceeding of 21st ACM Symposium on Operating Systems Principles*, 205-220.

Dikaiakos, M. D., Katsaros, D., Mehra, P., Pallis, G. and Vakali, A. (2009). Cloud computing: Distributed internet computing for IT and scientific research. *IEEE Internet Computing*, **13**, 10-13.

Endo, P. T., de Almeida Palhares, A. V., Pereira, N. N., Goncalves, G. E., Sadok, D., Kelner, J., Melander, B. and Mangs, J.-E. (2011). Resource allocation for distributed cloud: concepts and research challenges. *IEEE Network Magazine*, **25**, 42-46.

Joung, Y-J. (2003). Quorum-based algorithms for group mutual exclusion. *IEEE Transaction on Parallel and Distributed Systems*, **14**, 463-476.

Majumdar, R. and Garimella, S. (2012). Features, benefits, futuristic projections of cloud and inter-cloud extensions to the internet. *Special Issue of International Journal of Computer Applications, 3rd International IT Summit Confluence 2012 - The Next Generation Information Technology Summit*, 18-22.

Moore, D. S. (2009). *The basic practice of statistics*, 5th Ed., W. H. Freeman, New York.

Stoica, I., Morris, R., Karger, D., Kaashoek, M. F. and Balakrishnan, H. (2001) *Chord*: A scalable peer-to-peer lookup service for internet applications. *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, 149-160.

Terpstra, W. W., Behnel, S., Fiege, L., Zeidler, A. and Buchmann, A. P. (2003). A peer-to-peer approach to content-based publish/subscribe. *Proceedings of the 2nd International Workshop on Distributed Event-Based Systems*, 1-8.

Wu, C-J., Ho, J-M. and Chen, M-S. (2011). Time-critical event dissemination in geographically distributed clouds. *IEEE Conference on Computer Communications Workshops*, 654-659.

Wu, Y., Wu, C., Li, B., Zhang, L., Li, Zongpeng and Lau, F.C.M. (2012). Scaling social media applications into geo-distributed clouds. *Proceedings IEEE INFOCOM*, 684-692.