Available online at www.sciencedirect.com
ScienceDirect
Future Generation Computer Systems 24 (2008) 1–9
FGCS FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

An efficient replicated data access approach for large-scale distributed systems

Mustafa Mat Deris[a,*], Jemal H. Abawajy[b], Ali Mamat[c]

[a] *Faculty of Information Technology and Multimedia, College University Technology Tun Hissein Onn, 86400 Batu Pahat, Johor, Malaysia*
[b] *Deakin University, School of Engineering and Information Technology, Geelong, VIC, Australia*
[c] *Faculty of Computer Science and Information Technology, University Putra Malaysia, Serdang, Selangor, Malaysia*

Received 19 January 2007; received in revised form 29 March 2007; accepted 5 April 2007
Available online 29 April 2007

Abstract

In data-intensive distributed systems, replication is the most widely used approach to offer high data availability, low bandwidth consumption, increased fault-tolerance and improved scalability of the overall system. Replication-based systems implement replica control protocols that enforce a specified semantics of accessing the data. Also, the performance depends on a number of factors, the chief of which is the protocol used to maintain consistency among object replica. In this paper, we propose a new low-cost and high data availability protocol called the box-shaped grid structure for maintaining consistency of replicated data on networked distributed computing systems. We show that the proposed protocol provides high data availability, low communication costs, and increased fault-tolerance as compared to the baseline replica control protocols.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Data replication; Box-shaped grid structure; Fault–tolerance; Communication cost; Data availability

1. Introduction

Distributed database systems are useful in many large-scale applications such as in finance, sciences, engineering, medicine and other areas which are data-intensive [1,2] where many remotely located users should share the data to produce useful results. For example, financial instruments' prices will be read and updated from all over the world [3]. Also, data is being produced at a tremendous rate and volume especially from scientific experiments in the fields of high-energy physics, molecular docking, computer micro-tomography and many others. For example, experiments in the Large Hadron Collider (LHC) are forecasted to generate massive quantities of data. ATLAS [4], currently the largest of the experiments to be conducted on the LHC, is projected to generate several petabytes of data per year alone. Another example is the Laser Interferometer Gravitational Wave Observatory (LIGO), a multi-site national research facility whose objective is the detection of gravitational waves. The data management challenge faced by LIGO [5] is therefore to replicate approximately 1 TB/day of data to multiple sites securely, efficiently, robustly, and automatically; to keep track of where replicas have been made for each piece of the data; and to use the data in a multitude of independent analysis runs. This necessitates the organizations to provide current data to users who may be geographically remote and to handle a volume of requests of data distributed around multiple sites in a distributed database environment.

Thus, there is an urgent requirement to obtain solutions to manage, distribute and access large sets of raw and processed data efficiently and effectively in the emerging distributed systems such as the data grid. Data replication is one of the key components in data grid architecture as it enhances data access and reliability. Replication is a widely used method for providing high availability [6,7], fault-tolerance and good performance in distributed systems [8–11]. Changes applied at one site are captured and stored locally before being forwarded and applied at each of the remote locations. Thus, replication provides users with fast, local access to shared data,

* Corresponding address: Information Systems, College University Technology Tun Hussein Onn, Kampus Bandar Parit Raja, 86400 Batu Pahat, Johor, Malaysia. Fax: +60 7 4532199.
E-mail addresses: mmustafa@kuittho.edu.my (M. Mat Deris), jemal@deakin.edu.au (J.H. Abawajy), ali@fsktm.upm.edu.my (A. Mamat).

0167-739X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2007.04.010

and protects availability of applications because alternate data access options exist. Even if one site becomes unavailable, the user can continue to query or even update the remaining locations.

In this paper, we address the problem of data management on large-scale distributed systems with the objectives of allowing distributed users efficient access to consistent data from many different sites simultaneously. Typical replicated data management parameters are data availability and communication costs or data access times: the higher the data availability with lower communication costs the better the system is. However, providing high availability without compromising the performance of the system and correctness of the data is quite a daunting task for distributed systems designers. Also, while data replication can improve availability and reliability of the data, it complicates the task of maintaining data integrity. Therefore, a mechanism that maintains the consistency of the data in the face of site and communication link failures is paramount in distributed systems such as cluster and grid computing.

We present a new quorum-based protocol for maintaining replicated data that can provide both high data availability and low response time. The proposed protocol imposes a logical three-dimensional grid structure on data objects based on a box shape organization. We show that the proposed protocol presents better average quorum size, high data availability, low bandwidth consumption, increased fault-tolerance and improved scalability of the overall system as compared to standard replica control protocols.

The remainder of the paper is organized as follows. Section 2 presents the related work. Section 3 presents the model and method of the proposed replica management protocol. In order to show the merits of the proposed protocol, we present comparative analysis of the proposed protocol against an existing protocol in Section 4. Concluding remarks and future directions is reported in Section 5.

2. Related work

Many vendors have adopted asynchronous solutions for managing replicated data such as Lotus Notes. Lotus Notes works reasonably well for single object updates, but it fails when multiple objects are involved in a single update. To overcome such problems, an epidemic approach has been proposed recently by Holliday et al. [1]. The approach has imposed the happened-before relation as a partially ordered sequence to ensure serializability. However, more formal mechanisms are needed to avoid conflicting transactions in this approach. Such mechanisms are the vector clock timestamps and log records containing the read set and the write set used to identify the conflicting transactions, thus, causing delay in the overall operations.

Another solution is based on synchronous replication [13]. It is based on quorum to execute the operations with a high degree of consistency and also to ensure serializability. Synchronous replication can be categorized into several schemes, i.e., all data to all sites (full replication) and all data to some sites. The all data to some sites scheme minimizes the storage capacity [12,14]. However, the data availability is still an issue because data is not replicated to all sites. Thus, the all data to all sites scheme is the focus of this paper in order to increase the data availability. The most straightforward protocol in this scheme is read-once write-all (ROWA) protocol. In ROWA, a logical read operation on a replicated data item is converted to one physical read operation on any one of its copies, but a logical write operation is translated to physical writes on all of the copies. The ROWA protocol is good for environments where the data is mostly read-only. This is because it provides read operation with a high degree of availability at low communication overhead. However, the write operation has very high overhead as all replicas must be updated simultaneously. Also, it has very low availability as an update cannot be performed in the presence of a single replica failure or network partitions. Nevertheless, the ROWA protocol is popular and has been used in mobile and peer-to-peer environments [2,15], database systems [6] and Grid computing [13,16–18]. To address some of the problems associated with the ROWA protocol, a replication protocol analogous to the ROWA protocol is discussed in [6]. If one site is not accessible, the processing of an object is noted in the partial commit state, and resolved after some time delay. The problem with this approach is that it increases the response time, which is one of the major performance parameters in replicated systems, and therefore decreases the performance of the system.

An alternative approach is the quorum-based protocols which are characterized by a read (write) quorum and an intersection requirement between read and write operations. A read (write) operation quorum is defined as a set of copies whose number is sufficient to execute the read (write) operation. Since each site stores a copy of a data object, thus a quorum of copies is equivalent to the number of copies in the quorum. The selection of a quorum is restricted by the quorum intersection property to ensure one-copy equivalence. That is, for any two operations $o[x]$ and $o'[x]$ on an object x, where at least one of them is a write, the quorum must have a non-empty intersection. Since read operations do not change the value of the accessed data object, a read quorum does not have to meet the intersection property. The write quorum needs to satisfy read–write and write–write intersections. This ensures that a read operation will access the up-to-date copy of the data. Intuitively, quorum-based protocols will increase the availability of the data while decreasing the communication overhead as compared to the ROWA protocol. This is because quorum-based protocols do not have to access all replicas. Many quorum-based protocols have been proposed in the literature [18–21]. Nevertheless, these protocols do not address the issue of low-cost read operations. In [13,22], a quorum-based protocol referred to as a Tree Quorum (TQ) protocol in which a logical tree is imposed on the replicas. One advantage of this protocol is that a read operation may access only one copy and the number of copies to be accessed by a write operation is always less than a majority of quorum, i.e., the size of quorum may increase to a maximum of $n/2+1$ servers when

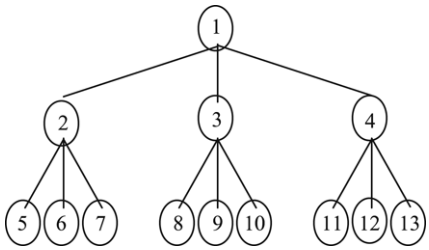

Fig. 1. A tree quorum protocol.

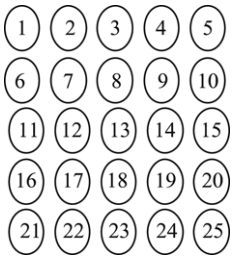

Fig. 2. Grid configuration protocol.

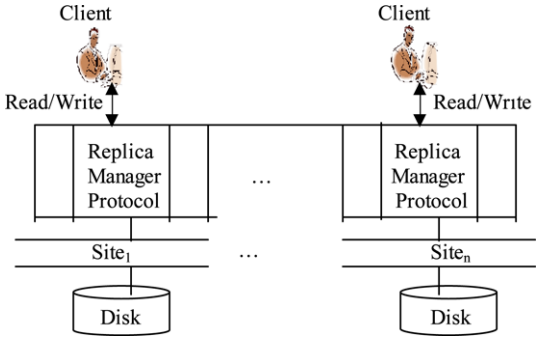

Fig. 3. An overview of the replicated distributed system.

failures occur. Write operations in the tree protocol must access a write quorum which is formed from the root, a majority of its children, and a majority of their children, and so forth until the leaves of the tree are reached. In Fig. 1, an example of a ternary tree with thirteen copies is illustrated. For read operations, a read quorum can be formed by the root or the majority of the children of the root. If any server in the majority of the children of the root fails, it can be replaced by the majority of its children, and so on recursively can replace it. In the best case, a read quorum consists of only root, {1}. As the root fails, a quorum is formed by the majority of the copies at level 1, e.g., {2, 3}, {3, 4}, or {2, 4}. The majority of their children replace servers 2 and 3, respectively, if no majority at level 1 is accessible and only server 4 is accessible. Such quorums are {4, 5, 6} or {4, 9, 10}. If copies at level 0 and level 1 fail, a quorum is formed by the majorities of the children of the selected majority at level 1, e.g., {5, 7, 8, 10} or {9, 10, 11, 12}. For write operations, the size of a write quorum for a given tree is fixed, but the members can be different. Such quorums are {1, 2, 3, 5, 6, 8, 10}, {1, 3, 4, 9, 10, 11, 13}, etc.

The desirable aspects of this protocol are that the cost of executing read operations is comparable to the ROWA protocol while the availability of write operations are significantly better. There are some drawbacks from this protocol. As an example, if more than a majority of the copies at any level of the tree become unavailable, write operations cannot be executed.

Grid Configuration (GC) protocol is proposed by Cheung et al. [19], to further increase data availability and fault-tolerance. In this protocol, n copies of data items are logically organized in the form of a $\sqrt{n} \times \sqrt{n}$ grid as shown in Fig. 2. Read operations on the data item are executed by acquiring a read quorum that consists of a copy from each column in the grid. Write operations, on the other hand, are executed by acquiring a write quorum that consists of all copies in one column and a copy from each of the remaining columns. In Fig. 2, copies {1, 2, 3, 4, 5} are sufficient to execute a read operation whereas copies {1, 6, 11, 16, 21, 7, 13, 19, 25} will be required to execute a write operation. However, it still has a larger number of copies for read and write quorums, thereby degrading the communication cost and data availability. It is also vulnerable to the failure of entire column or row in the grid [23]. The notion of quorums has been generalized by Agrawal and El Abbadi [23] in the context of the grid structure. In general, a grid quorum has a length l and a width w, and is represented as a pair $\langle l, w\rangle$. A read operation is executed by accessing a read grid quorum $\langle l, w\rangle$, which is formed from l copies in each of w different columns. A write operation is executed by writing a write grid quorum, $\langle l, w\rangle$, which is formed from: (a) l copies in each of w columns; as well as (b) any $\sqrt{n} - l + 1$ copies in each of $\sqrt{n} - w + 1$ columns. (The second component of the write quorum is to ensure the intersection between two write quorums.) In order to ensure the quorum intersection property between read and write quorums, if the read grid quorum is of size $\langle l, w\rangle$ then the write grid quorum must be $\langle \sqrt{n} - l + 1, \sqrt{n} - w + 1\rangle$. However, the problem with the GC is that communication overhead is high as the required number of reads and writes quorums are quite large. Also, it is susceptible to the failures of entire column or a row in the two-dimensional grid.

3. Model and method

3.1. Model

A distributed system of interest consists of clients, a replica control protocol, and a set of data objects stored on a set of sites, $S_1, \ldots, S_n$, interconnected by an interconnection network as shown in Fig. 3. In this paper, the terms a number of copies and a number of sites will be used interchangeably. Clients interact with the system by sending requests for data, which are partially ordered sequences of atomic read and write operations. In a replicated system, multiple copies of an object must appear as a single logical object to the transactions. This is termed as one-copy equivalence and is enforced by the replica control protocol. A site may become inaccessible due to site or network failure. No assumptions are made regarding the speed or reliability of the network. We assume that sites are fail stop and communication links may fail to deliver messages. Combinations of such failures may lead to partitioning failures where sites in a partition may communicate with each other,

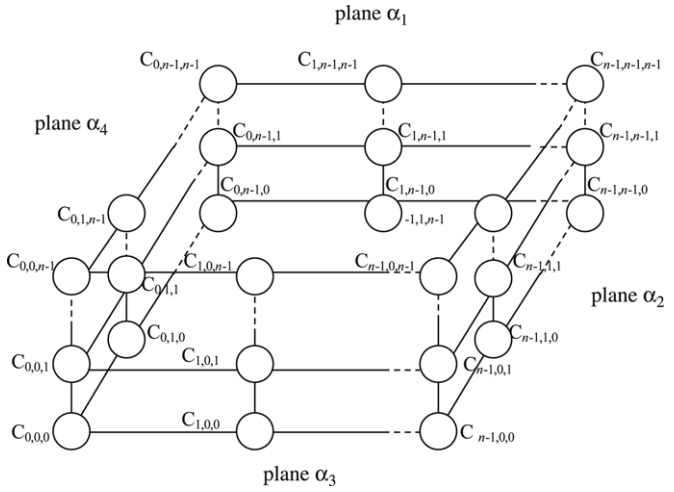

Fig. 4. A BSG organization with N copies of a data object. α_i denotes planes, circles represent a copy of an object replica at location $C_{x,y,z}$.

but no communication can occur between sites in different partitions [3].

3.2. Method

In this section, we present the proposed replica control protocol, which we refer to as a box-shaped grid (BSG) replica control protocol. We discuss the read and write operations in BSG protocol and prove the correctness of the proposed protocol. We also show that BSG protocol tolerates the failure of more than three quarters of the copies and illustrate transaction management under the proposed protocol.

Given N copies of a data object, BSG logically organizes the N copies into a box-shaped structure with four planes (i.e., α_1, α_2, α_3, and α_4) as shown in Fig. 4. Each copy of the object in Fig. 4 is represented by a circle located at x, y, z coordinates (i.e., $C_{x,y,z}$) in a given plane (e.g., $C_{0,0,0}, C_{0,0,1}, \ldots, C_{n-1,n-1,n-1} C_{0,0,0}$, $C_{0,0,1}, \ldots, C_{n-1,n-1,n-1}$). We define a pair of copies that can be constructed from a hypotenuse edge in a BSG (Fig. 4) as hypotenuse copies.

BSG could be easily extended to more than four planes to accommodate more copies. Also, BSG allows new replica(s) to be added as well as exiting replicas to be deleted with little or no impact on the availability of the data. For example, let (x, y, z) be the coordinates of the C_{xyz}. The following step is used to add a new replica:

- If the BSG is in a complete logical structure of copies and the new replica(s) will be added in the system, then the new BSG logical structure will transform the coordinates with $x \leftarrow x + 1$, $y \leftarrow y + 1$ and $z \leftarrow z + 1$, $\forall x, y, z \neq 0$, of the present C_{xyz}. Thus the free coordinates where the new copies can be located are the coordinates with the value of either x or y or $z = 1$ for $\forall x, y, z \leq n$.
- If the BSG is not in a complete logical structure (i.e., there exists free coordinates for the new copies) then the new copies will be added accordingly.

Similarly, the BSG allows current replica(s) which is (are) non-hypotenuse copy (copies) to be deleted from the system. It is the same as if some copies are down or inaccessible, where the process will run smoothly so long as the BSG quorum can be constructed. In other words, the logical structure under BSG does not have to be reconfigured in order to obtain the BSG quorum. If that replica(s) would like to re-unite afterwards, the recovery process will take over to update the information about that replica.

3.2.1. Managing read/write operations

We now describe how the read and write operations are handled by the BSG protocol in the following subsections.

Read operations

Read operations on an object are executed by acquiring a read quorum that consists of any hypotenuse copies. This shows that two copies are sufficient in the read quorum to execute a read operation. In Fig. 4, copies $\{C_{0,0,0}, C_{n-1,n-1,n-1}\}$, $\{C_{0,0,n-1}, C_{n-1,n-1,0}\}$, $\{C_{0,n-1,n-1}, C_{n-1,0,0}\}$, or $\{C_{n-1,0,n-1}, C_{0,n-1,0}\}$ are hypotenuse copies any one pair of which is sufficient to execute a read operation. Since each pair of them consists of hypotenuse copies, it is clear that the read operation can be executed if one of them is accessible, thus increasing the fault-tolerance of the BSG protocol.

Write operations

In order to obtain the minimum number of copies in the write quorum without violating the quorum intersection property, write operations are executed by acquiring a write quorum from any plane that consists of:

(1) hypotenuse copies; and

(2) all vertices copies.

For example, if the hypotenuse copies, say $\{C_{0,0,0}, C_{n-1,n-1,n-1}\}$ are required to execute a read operation, then copies $\{C_{0,0,0}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}, C_{0,n-1,n-1}, C_{0,n-1,0}\}$ are sufficient to execute a write operation, since one possible set of copies of vertices that correspond to $\{C_{0,0,0}, C_{n-1,n-1,n-1}\}$ is $\{C_{n-1,n-1,n-1}, C_{n-1,n-1,0}, C_{0,n-1,n-1}, C_{0,n-1,0}\}$. Other possible write quorums are $\{C_{0,0,0}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}, C_{n-1,0,n-1}, C_{n-1,0,0}\}$, $\{C_{n-1,n-1,n-1}, C_{0,0,0}, C_{0,0,n-1}, C_{n-1,0,n-1}, C_{n-1,0,0}\}$, $\{C_{n-1,n-1,n-1}, C_{0,0,0}, C_{0,0,n-1}, C_{0,n-1,n-1}, C_{0,n-1,0}\}$, etc. It can be easily shown that a write quorum intersects with both read and write quorums in this protocol.

3.2.2. Fault-tolerance

The BSG protocol tolerates the failure of more than three quarters of the copies. This is because the BSG protocol allows us to construct a write quorum even if three out of four planes are unavailable as long as the hypotenuse copies are accessible. To show this, consider the case when only one plane which consists of four copies of vertices and hypotenuse copies are available, e.g., the set $\{C_{l-1,l-1,l-1}, C_{0,0,0}, C_{0,0,l-1}, C_{l-1,0,l-1}, C_{l-1,0,0}\}$ is available as shown in Fig. 4. A BSG transaction can be executed successfully by accessing those copies in a BSG quorum. Hence the write quorum in the BSG protocol is formed by accessing those available copies. Read operations, on the other hand, need to access the available hypotenuse

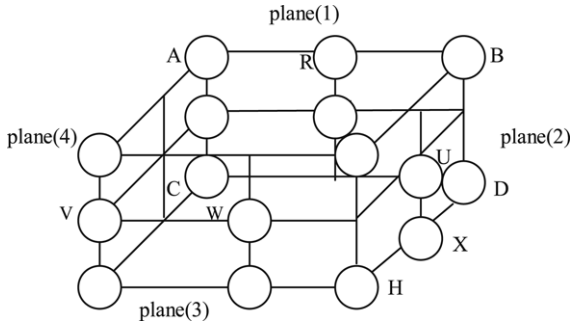

Fig. 5. A BSG organization with 16 copies.

copies. Thus, the BSG protocol enhances the fault-tolerance in write operations compared to the grid configuration protocol. Moreover, BSG protocol ensures that read operations have a significantly lower cost, i.e., two copies, and have a high degree of availability, since they are not vulnerable to the failure of more than three quarters of the copies. Write operations, on the other hand, are more available than the grid configuration protocol since only five copies are needed to execute write operations.

In addition, BSG allows us to construct a write quorum even though three out of four planes are unavailable as long as the hypotenuse copies are accessible. In other words, this protocol tolerates the failure of more than three quarters of the copies in the BSG protocol. Consider the case when only one plane which consists of four copies of vertices and hypotenuse copies are available, e.g., the set $\{C_{l-1,l-1,l-1}, C_{0,0,0}, C_{0,0,l-1}, C_{l-1,0,l-1}, C_{l-1,0,0}\}$ is available as shown in Fig. 4. A BSG transaction can be executed successfully by accessing those copies in a BSG quorum. Hence the write quorum is formed by accessing those available copies. Read operations, on the other hand, need to access the available hypotenuse copies. Thus the proposed protocol enhances the fault-tolerance in write operations compared to the grid configuration protocol.

Therefore, this protocol ensures that read operations have a significantly lower cost, i.e., two copies, and have a high degree of availability, since they are not vulnerable to the failure of more than three quarters of the copies. Write operations, on the other hand, are more available than the grid configuration protocol since only five copies are needed to execute write operations.

Example of BSG protocol

We define a primary copy for a data object as a copy that initiates a transaction. As an example, we assume that there are 16 copies $A, B, C, D, E, F, G, H, R, S, T, U, V, W, X$, and Y in the system (with full replication of the data object) as in Fig. 5. A set of read quorum that is sufficient to execute read operation is

$$\{\{A, H\}, \{C, F\}, \{G, B\}, \{E, D\}\},$$

and a set of write quorum that is sufficient to execute write operation is

$$\{\{A, H, B, C, D\}, \{A, H, B, D, F\}, \{A, H, E, F, G\}, \{A, H, C, E, G\}, \{C, F, A, B, D\}, \{C, F, B, D, H\}, \{C, F, A, E, G\}, \{C, F, E, G, H\}, \{G, B, A, C, D\}, \{G, B, D, F, H\}, \{B, G, E, F, H\}, \{B, G, A, C, E\}, \{E, D, A, B, C\}, \{E, D, B, F, H\}, \{D, E, F, G, H\}, \{D, E, A, C, G\}\}.$$

Thus, any set of copies that consists of any write quorum is able to execute write operation. For example, if $\{A, H\}$ is required to execute read operation, then $\{A, H, B, C, D\}$, $\{A, H, B, D, F\}$, $\{A, H, E, F, G\}$, $\{A, H, C, E, G\}$, $\{A, H, B, C, D, R\}$,$\{A, H, B, C, S\}$, $\{H, B, C, D, T\}$, $\{A, H, B, C, D, R, S\}$, $\ldots$, $\{A, H, B, C, D, \ldots, Y\}$ are able to execute a write operation.

Without loss of generality, we assume that R is a primary copy. In this example, write operations will only be considered since it will involve all copies to update a data object. Two system components are involved in processing a transaction submitted by R.

- The primary copy, R, sends the transaction to all copies (including copies of BSG write quorum) of a data object.
- Each copy of BSG write quorum of the data object accepts, from R, and the request to check the executability of the transaction. It then cooperates with R by returning the executability check. Checking the executability of a transaction is essentially a request for a lock on the data object.

The primary copy uses a two-phase commit (2PC) protocol to ensure the replication consistency. In the first phase, R asks the BSG write quorum whether it can be constructed or not. If the BSG write quorum can be constructed (copies under BSG write quorum return an 'OK' messages for such execution), then R returns 'OK' to the transaction manager. If the transaction manager requests a commit, then in the second phase, R asks all copies to commit the execution. If the BSG write quorum cannot be constructed, then R returns a 'FAIL' to the transaction manager and asks all copies to abort the operation in the second phase.

3.2.3. The correctness

In this section, we will show that the BSG protocol is one-copy serializable. We start by defining sets of groups (coterie) [24] and to avoid confusion we are referring to sets of copies as groups. Thus, sets of groups are sets of sets of copies.

Definition 3.2.1. Coterie. Let U be a set of groups that compose the system. A set of groups T is a coterie under U iff

(i) $G \subset T$ implies that $G \neq \emptyset$ and $G \subseteq U$.
(ii) If $G, H \subset T$ then $G \cap H \neq \emptyset$ (intersection property).
(iii) There are no $G, H \subset T$ such that $G \subset H$ (minimality).

Definition 3.2.2. Let R be a set of read quorums which consists of groups of hypotenuse copies, that is sufficient to execute read operations, and W be a set of write quorums which consists of groups that are sufficient to execute write operations under the BSG protocol. Then, from Fig. 4,

$$R = \{\{C_{0,0,0}, C_{n-1,n-1,n-1}\}, \{C_{0,0,n-1}, C_{n-1,n-1,0}\}, \{C_{0,n-1,n-1}, C_{n-1,0,0}\}, \{C_{n-1,0,n-1}, C_{0,n-1,0}\}\},$$

and

$$
\begin{aligned}
W = \{&\{C_{0,0,0}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}, C_{0,n-1,n-1}, C_{0,n-1,0}\},\\
&\{C_{0,0,0}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}, C_{n-1,0,n-1}, C_{n-1,0,0}\},\\
&\{C_{n-1,n-1,n-1}, C_{0,0,0}, C_{0,0,n-1}, C_{n-1,0,n-1}, C_{n-1,0,0}\},\\
&\{C_{n-1,n-1,n-1}, C_{0,0,0}, C_{0,0,n-1}, C_{0,n-1,n-1}, C_{0,n-1,0}\},\\
&\{C_{n-1,n-1,0}, C_{0,0,n-1}, C_{0,0,0}, C_{n-1,0,n-1}, C_{n-1,0,0}\},\\
&\{C_{n-1,n-1,0}, C_{0,0,n-1}, C_{0,0,0}, C_{0,n-1,n-1}, C_{0,n-1,0}\},\\
&\{C_{n-1,n-1,n-1}, C_{0,0,0}, C_{0,0,n-1}, C_{n-1,0,n-1}, C_{n-1,0,0}\},\\
&\{C_{n-1,n-1,n-1}, C_{0,0,0}, C_{0,0,n-1}, C_{0,n-1,n-1}, C_{0,n-1,0}\},\\
&\{C_{0,n-1,n-1}, C_{n-1,0,0}, C_{n-1,0,n-1}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}\},\\
&\{C_{0,n-1,n-1}, C_{n-1,0,0}, C_{n-1,0,n-1}, C_{0,0,n-1}, C_{0,0,0}\},\\
&\{C_{n-1,0,0}, C_{0,n-1,n-1}, C_{0,n-1,0}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}\},\\
&\{C_{n-1,0,0}, C_{0,n-1,n-1}, C_{0,n-1,0}, C_{0,0,n-1}, C_{0,0,0}\},\\
&\{C_{0,n-1,0}, C_{n-1,0,n-1}, C_{n-1,0,0}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}\},\\
&\{C_{0,nn-1,0}, C_{n-1,0,n-1}, C_{n-1,0,0}, C_{0,0,n-1}, C_{0,0,0}\},\\
&\{C_{n-1,0,n-1}, C_{0,n-1,0}, C_{0,n-1,n-1}, C_{n-1,n-1,n-1}, C_{n-1,n-1,0}\},\\
&\{C_{n-1,0,n-1}, C_{0,n-1,0}, C_{0,n-1,n-1}, C_{0,0,n-1}, C_{0,0,0}\}\}.
\end{aligned}
$$

By the definition of coterie, then W is a coterie, because it satisfies all a coterie's properties. The next definition will formally define the read and the write quorums.

Definition 3.2.3. Let $\boldsymbol{\eta}$ be a group of hypotenuse copies and $\boldsymbol{\omega}$ be a group of copies from any plane that consists of hypotenuse copies and all copies which are vertices as shown in Fig. 4. A set of read quorum, **R**, can be defined as

$$\mathbf{R} = \{\boldsymbol{\eta}_i | \boldsymbol{\eta}_i \cap \boldsymbol{\eta}_j = \emptyset, i \neq j\},$$

and a set of write quorum, **W**, can be defined as

$$\mathbf{W} = \{\boldsymbol{\omega}_i | \boldsymbol{\omega}_i \cap \boldsymbol{\omega}_j \neq \emptyset, i \neq j, \text{ and } \boldsymbol{\omega}_i \cap \boldsymbol{\eta}_j \neq \emptyset \text{ for } \boldsymbol{\eta}_j \in \mathbf{R}\}.$$

Since read operations do not change the value of the accessed data object, a read quorum does not need to satisfy the intersection property. In contrast, a write quorum needs to satisfy read–write and write–write intersection properties.

The correct criterion for a replicated database is one-copy serializable. The next theorem shows that the proposed protocol is one-copy serializable.

Theorem 3.2.1. *The BSG protocol is one-copy serializable.*

Proof. The theorem holds on condition that the BSG protocol satisfies the quorum intersection properties, i.e., write–write and read–write intersections. Since W is a coterie and by Definition 3.2.3, then it satisfies read–write and write–write intersection properties. □

4. Comparative analysis

In this paper, we compare the performance of BSG protocol with the grid configuration (GC) and tree quorum (TQ) protocols. The comparisons are made based on the communication cost and the availability of each protocol for update operations. The communication cost of an operation is directly proportional to the size of the quorum required to execute the operation. Therefore, we represent the communication cost in terms of the quorum size.

4.1. Communication analysis

The communication cost of an operation is directly proportional to the size of the quorum required to execute the operation. Therefore, we represent the communication cost in terms of the quorum size. We use $C_{X,Y}$ to denote the communication cost of the X protocol for Y operation (i.e., R (read) or W (write) operation).

4.1.1. Grid configuration (GC) protocol

Let n be the number of copies which are organized as a grid of dimension $\sqrt{n} \times \sqrt{n}$. Read operations on the data item are executed by acquiring a read quorum that consists of a copy from each column in the grid. Write operations, on the other hand, are executed by acquiring a write quorum that consists of all copies in one column and a copy from each of the remaining columns. The read and write operations of this protocol is of the size $O(\sqrt{n})$; this protocol is normally referred to as the sqrt(R/W) protocol. In Fig. 2, copies {1, 2, 3, 4, 5} are sufficient to execute a read operation whereas copies {1, 6, 11, 16, 21, 7, 13, 19, 25} will be required to execute a write operation. The communication cost, $C_{\mathrm{GC},R}$, as given in [23] can be represented as:

$$C_{\mathrm{GC},R} = \sqrt{n},$$

and the communication cost, $C_{\mathrm{GR},W}$, can be represented as:

$$C_{\mathrm{GC},W} = \sqrt{n} + (\sqrt{n} - 1) = 2\sqrt{n} - 1.$$

4.1.2. Tree quorum (TQ) protocol

Let h denote the height of the tree, D the degree of the copies in the tree, and $M = \lfloor (D+1)/2 \rfloor$ the majority of the degree of copies. When the root is accessible, the read quorum size is 1. As the root fails, the majority of its children replace it, thus the quorum size increases to M. Therefore, for a tree of height h, the maximum quorum size is M^h. Hence, the cost of read operation, C_{TQR}, ranges from 1 to M^h [13,23], i.e., $1 \leq C_{TQ,R} \leq M^h$. The cost of a write operation, $C_{TQ,W}$, can be represented as: $C_{TQ,W} = \sum M^i, i = 1 \ldots h$.

4.1.3. BSG protocol

The size of a read quorum in BSG is hypotenuse copies, i.e., 2. Thus, the cost of a read operation, $C_{\mathrm{BSG},R}$, can be represented as:

$$C_{\mathrm{BSG},R} = 2,$$

and the cost of a write operation, $C_{\mathrm{BSG}W}$, can be represented as:

$$
\begin{aligned}
&\text{hypotenuse copies} + (\text{all copies of vertices in a plane}\\
&\qquad - \text{hypotenuse copy in the same plane})\\
&= 2 + (4 - 1) = 5.
\end{aligned}
$$

For example, if hypotenuse copies is $\{C_{0,0,0}, C_{l-1,l-1,l-1}\}$, then all copies of vertices in plane α_1 that correspond to $\{C_{0,0,0}, C_{l-1,l-1,l-1}\}$ is $\{C_{l-1,l-1,l-1}, C_{l-1,l-1,0}, C_{0,l-1,l-1}, C_{0,l-1,0}\}$.

Table 1
Comparison of the read and write cost of the three protocols

	Number of copies in the system		
	13	40	121
TQ(R)	4	8	16
TQ(W)	7	15	31
GC(R)	4	7	11
GC(W)	7	13	21
BSG(R)	2	2	2
BSG(W)	5	5	5

Therefore,

$$\begin{aligned} C_{\mathrm{BSG}W} &= |\{C_{0,0,0}, C_{l-1,l-1,l-1}\}| \\ &\quad + |\{C_{l-1,l-1,l-1}, C_{l-1,l-1,0}, C_{0,l-1,l-1}, C_{0,l-1,0}\}| \\ &\quad - |\{C_{l-1,l-1,l-1}\}| \\ &= 2 + (4-1) = 5. \end{aligned}$$

4.2. Comparison of costs

The communication cost of an operation is directly proportional to the size of the quorum required to execute the operation. Therefore, we represent the communication cost in terms of the quorum size. Table 1 shows the read and write costs of the three techniques between TQ, GC, and BSG for different total number of copies, $n = 13, 40$, and 121.

From Table 1, it is apparent that BSG has the lowest cost for both read and write operations in spite of having a larger number of copies when compared with TQ and GC quorums. It can be seen that BSG needs only 2 copies for the read quorum for all instances. In contrast, for TQ with a tree of height 3 on 40 copies, the maximum cost is 8, meanwhile for GC with 40 copies, the cost is 7. So as for write operations, BSG needs only 5 copies for the write quorum for all instances. Conversely, the cost is 13 for GC with 40 copies, and 15 for TQ with a tree of height 3 on 40 copies.

4.3. Availability analysis

In this section, the three replica control protocols are analyzed and compared in terms of the operation availability. In estimating the availability of operations, all copies are assumed to have the same availability p. We also use "$A_{x,y}$" to represent the availability of Y operation with X protocol.

4.3.1. Grid configuration (GC) protocol

As Fig. 4 shows, in the case of the quorum protocol, read quorums can be constructed as long as a copy from each column is available. If n is the number of copies which are organized as a grid of dimension $\sqrt{n} \times \sqrt{n}$, then the read availability in the GC protocol as given in [23], $A_{\mathrm{GC},R}$, is:

$$\begin{aligned} &= \left(\sum_{i=1}^{\sqrt{n}} \binom{\sqrt{n}}{i} p^i \, (1-p)^{\sqrt{n}-i} \right)^{\sqrt{n}} \\ &= [1-(1-p)^{\sqrt{n}}]^{\sqrt{n}}. \end{aligned} \tag{1}$$

On the other hand, write quorums can be constructed as all copies from a column and one copy from each of the remaining columns are available. Then the write availability in the GC protocol as given in [23], $A_{\mathrm{GC},W}$, is:

$$\begin{aligned} &= \sqrt{n} \left(1-(1-p)^{\sqrt{n}}\right)^{\sqrt{n}-1} p^{\sqrt{n}} \\ &\quad + \binom{\sqrt{n}}{2} \left(1-(1-p)^{\sqrt{n}}\right)^{\sqrt{n}-2} \left(p^{\sqrt{n}}\right)^2 + \cdots \left(p^{\sqrt{n}}\right)^n \\ &= [1-(1-p)^{\sqrt{n}}]^{\sqrt{n}} - [1-(1-p)^{\sqrt{n}} - p^{\sqrt{n}}]^{\sqrt{n}}. \end{aligned} \tag{2}$$

4.3.2. Tree quorum (TQ) protocol

As Fig. 5 shows, the availability of the read and write operations in the TQ protocol can be estimated by using recurrence equations based on the tree height h. Let AR_{h+1} and AW_{h+1} be the availability of the read and the write operations with a tree of height h, respectively. D denotes the degree of sites in the tree and M is the majority of D. Then the availability of a read operation for a tree of height $h+1$ can be represented as:

$$\mathrm{AR}_{h+1} = p + (1-p) \sum_{i=M}^{D} \binom{D}{i} \mathrm{AR}_h^i \, (1-\mathrm{AR}_h)^{D-i} \tag{3}$$

and the availability of a write operation for a tree of height $h+1$ is given as:

$$\mathrm{AW}_{h+1} = p \sum_{i=M}^{D} \binom{D}{i} \mathrm{AW}_h^i \, (1-AW_h)^{D-i} \tag{4}$$

where p is the probability that a copy is available, and $R_0 = W_0 = p$.

4.3.3. BSG protocol

In the BSG protocol, a read quorum can be constructed from any hypotenuse copies. The read availability, $A_{\mathrm{BSG},R}$, is:

= 1-probability {all the hypotenuse copies are not available}.

Since it has 4 hypotenuse copies, then

$$A_{\mathrm{BSG},R} = 1-(1-p^2)^4. \tag{5}$$

In contrast, a write quorum can be constructed as follows: Let $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ be a set of planes in the BSG protocol as shown in Fig. 4. Let $i = \{C_{0,0,0}, C_{l-1,l-1,l-1}\}$, be the hypotenuse copies, then write availability that consists of hypotenuse copies i, W_i, can be represented as:

$$\begin{aligned} &\text{Probability}\{C_{0,0,0} \text{ is available}\} * [\varphi \text{ available}] \\ &\quad + \text{Probability}\{C_{l-1,l-1,l-1} \text{ is available}\} * [\phi \text{ available}] \\ &\quad - \text{Probability}\{C_{l-1,l-1,l-1} \text{ and } C_{0,0,0} \text{ are available}\} \\ &\quad * [(\varphi \text{ and } \phi) \text{ are available}] \end{aligned} \tag{6}$$

where

$$\varphi = \Omega(\alpha_1) + \Omega(\alpha_2) - \Omega(\alpha_1 \cap \alpha_2),$$
$$\phi = \Omega(\alpha_3) + \Omega(\alpha_4) - \Omega(\alpha_3 \cap \alpha_4),$$

and $\Omega(\alpha_i) =$ Probability that plane α_i is available.

Table 2
The read and write availabilities when $n = 40$ and $0.1 \leq p \leq 0.9$

Protocol type	Read (R) and write (W) availability								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
TQ(R)	0.147	0.432	0.753	0.937	0.991	0.999	1	1	1
TQ(W)	2E−12	4.0E−8	1.0E−5	0.001	0.009	0.063	0.248	0.568	0.853
GC(R)	0.011	0.193	0.548	0.820	0.947	0.989	0.999	1	1
GC(W)	1.0E−8	2.0E−5	0.001	0.010	0.051	0.179	0.452	0.807	0.989
BSG(R)	0.039	0.151	0.314	0.502	0.684	0.832	0.932	0.986	1
BSG(W)	1.0E−4	0.004	0.030	0.112	0.319	0.614	0.875	0.983	0.999

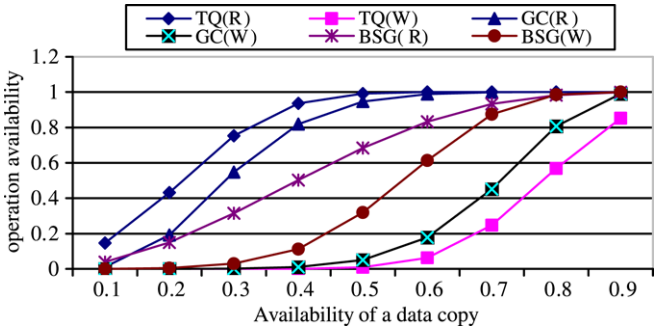

Fig. 6. Comparison of the read and write availability between TQ, GC, and BSG.

Without loss of generality, we assume that a non-vertex copy, say $C_{1,l-1,l-1} \in \alpha_1$ is a primary copy. The probability that α_1 is available, $\Omega(\alpha_1)$, can be represented as:

Probability{all copies of vertices from α_1 and primary copy are available} + Probability{(all copies of vertices and primary copy + 1 copy) from α_1 are available} $+ \cdots +$ Probability{all copies from α_1 are available}

$$= p^5 \sum_{j=0}^{m-5} \binom{m-5}{j} p^j (1-p)^{m-5-j} = p5. \tag{7}$$

However, the probability that α_i, $i = 2, 3, 4$, is available, $\Omega(\alpha_i)$, can be represented as:

Probability{ all copies of vertices from α_i are available} + Probability{ (all copies of vertices $i + 1$ replica) from α_i are available} $+ \cdots +$ Probability {all copies from α_i are available}

$$= p^4 \sum_{j=0}^{m-4} \binom{m-4}{j} p^j (1-p)^{m-4-j} = p^4 \tag{8}$$

where m is a number of copies in each plane. Thus $\Omega(\alpha_1) = p^5$, and $\Omega(\alpha_i) = p^4$, for $i = 2, 3, 4$.

The $\Omega(\alpha_1 \cap \alpha_2)$, can easily be calculated using a Venn diagram,

$$\Omega(\alpha_1 \cap \alpha_2) = p^6. \tag{9}$$

Substitute (7)–(9) into φ and ϕ, then

$$\varphi = p^4(1 + p - p^2) \tag{10}$$

$$\phi = p^4(2 - p^2). \tag{11}$$

Since, the probability {V is available } = probability {C is available} = p, then, by substituting Eqs. (10) and (11) into Eq. (6), W_i is:

$$= p\varphi + p\phi - p^2(\varphi * \phi). \tag{12}$$

Similarly, with the write availability that consists of other hypotenuse copies, thus $W_i = W_j = W_k = W_l$, where $i \in \mathbf{R}$. To compute the write availability, $A_{\mathrm{BSG},W}$, it is analogous to the read availability. Let $W_i = \beta$, $i \in \mathbf{R}$, then $A_{\mathrm{BSG},W}$ is:

$$= \binom{4}{1} x W_i - \binom{4}{2} x (W_i \cap W_j) + \binom{4}{3} x (W_i \cap W_j \cap W_k) - W_i \cap W_j \cap W_k \cap W_l, \quad \text{for } i, j, k, l \in \mathbf{R}$$

$$= 1 - (1-\beta)^4. \tag{13}$$

The read and the write availabilities of the three protocols, TQ, GC, and BSG are compared. Table 2 and Fig. 6, show the results obtained from the analysis between those three protocols for read and write availabilities when $n = 40$ and $0.1 \leq p \leq 0.9$. It shows the imbalance between read and write availability when the TQ protocol is used. For example, when an individual copy has availability 70%, read availability is 100% whereas write availability is approximately 25%. However, the GC and BSG protocols show better performance by reducing the imbalance between read and write availability. Nevertheless, BSG has a higher write availability than GC because the number of data copies to be written in BSG is smaller than that of GC. For example, when an individual copy has availability 70%, the write availability in the BSG is more than 87%, whereas write availability in the GC is approximately 45%.

5. Conclusions and future directions

The problem of protocols for maintaining replicated data has been widely studied. However, existing protocols are designed primarily to achieve high availability by updating a large fraction of the copies which provides some (although not significant) load sharing. We presented a new quorum-based protocol for maintaining replicated data across distributed systems. The proposed approach is constructed on the organization of data in a box-shaped structure (BSG). The analysis of the BSG protocol was presented in terms of data availability and communication costs. It showed that the BSG protocol provides a convenient approach to high availability for read and write operations. This is due to the minimum number of quorum size required, i.e., a read operation is allowed as

long as when one of the hypotenuse copies is available, and a write operation is allowed when the hypotenuse copies and all copies which are vertices where one of which is a hypotenuse copy are available. Thus BSG tolerates the failure of more than three quarters of the copies. In comparison to the tree quorum and grid structure protocols, BSG requires significantly lower communication cost for an operation, while providing higher availability, and increases fault-tolerance which is preferred for large systems. We are planning to implement the proposed protocol on data grid and cluster–server systems.

References

[1] J. Holliday, R. Steinke, D. Agrawal, A. El Abbadi, Epidemic algorithms for replicated databases, IEEE Transactions on Knowledge and Data Engineering 15 (5) (2003) 1218–1238.

[2] S.K. Madria, M. Mohania, S.S. Bhowmick, B. Bhargava, Mobile data and transaction management, Journal of Information Sciences, Elsevier 141 (2002) 279–309.

[3] O. Wolfson, S. Jajodia, Y. Huang, An adaptive data replication algorithm, ACM Transactions on Database Systems 22 (2) (1997) 255–314.

[4] ATLAS at the University of Chicago, http://hep.uchicago.edu/atlas/.

[5] Large-scale Data Replication for LIGO, http://www.globus.org/solutions/data_replication/.

[6] W. Zhou, A. Goscinski, Managing replicated remote procedure call transactions, The Computer Journal 42 (7) (1999) 592–608.

[7] S. Vazhkudai, S. Tuecke, I. Foster, Replica selection in the globus data grid, in: International Workshop on Data Models and Databases on Clusters and the Grid, DataGrid 2001, 2001, pp. 106–113.

[8] L. Gao, M. Dahlin, A. Nayate, J. Zheng, A. Lyengar, Improving availability and performance with application-specific data replication, IEEE Transactions on Knowledge and Data Engineering 17 (1) (2005) 106–120.

[9] J. Huang, Qingfeng Fan, Qiongli Wu, Yang Xiang He, Improved Grid Information Service Using The Idea of File-Parted Replication, in: Lecture Notes in Computer Science, vol. 3584, Springer, 2005, pp. 704–711.

[10] J.H. Abawajy, File replacement algorithm for storage resource managers in data grids, in: The Proceedings of the International Conference on Computational Science 2004, ICCS 2004, Kraków, Poland, June 7–9, 2004.

[11] M. Tang, B.S. Lee, X. Tang, C.K. Yeo, The impact of data replication on job scheduling performance in the data grids, International Journal of Future Generation of Computer Systems 22 (2006) 254–268.

[12] M. Mat Deris, D.J. Evans, M.Y. Saman, A. Noraziah, Binary vote assignment on grid for efficient access of replicated data, International Journal of Computer Mathematics 80 (2003) 1489–1498.

[13] D. Agrawal, A. El Abbadi, The generalized tree quorum protocol: An efficient approach for managing replicated data, ACM Transactions on Database Systems 17 (4) (1992) 689–717.

[14] R.S. Chang, P.H. Chen, Complete and fragmented replica selection and retrieval in data grids, International Journal of Future Generation of Computer Systems 23 (2007) 536–546.

[15] B.S. Noshio, M. Tsukamoto, Data management issues in mobile and peer-to-peer environment, Data and Knowledge Engineering 41 (2002) 183–204.

[16] P. Kunszt, E. Laure, H. Stockinger, K. Stockinger, Advanced replica management with raptor, in: Proceedings of 5th International Conference on Parallel Processing and Applied Mathemetics, 2003.

[17] P. Kunszt, Erwin Laure, Heinz Stockinger, Kurt Stockinger, File-based replica management, International Journal of Future Generation of Computer Systems 21 (2005) 115–123.

[18] H. Stockinger, F. Donno, E. Laure, S. Muzaffar, P. Kunszt, Grid data management in action: Experience in running and supporting data management services in the EU data grid project, in: Int'l. Conf. On Computing in High Energy and Nuclear Physics, La Jolla, California. http://gridpp.ac.uk/papers/chap3-TUAT007.pdf, March 2003.

[19] S.Y. Cheung, M.H. Ammar, M. Ahmad, The grid protocol: A high performance schema for maintaining replicated data, IEEE Transactions on Knowledge and Data Engineering 4 (6) (1992) 582–592.

[20] P.A. Bernstein, N. Goodman, An algorithm for concurrency control and recovery in replicated distributed databases, ACM Transactions on Database Systems 9 (4) (1994) 596–615.

[21] W. Hoschek, F.J. Jaén-Martínez, A. Samar, H. Stockinger, K. Stockinger, Data management in an international data grid project, in: Proceedings of the First IEEE/ACM International Workshop on Grid Computing, 2000, pp. 77–90.

[22] H. Lamehamedi, B.K. Szymanski, Decentralized data management framework for data grids, International Journal of Future Generation Computer Systems 23 (2007) 109–115.

[23] D. Agrawal, A. El Abbadi, Using reconfiguration for efficient management of replicated data, IEEE Transactions on Knowledge and Data Engineering 8 (5) (1996) 786–801.

[24] H. Garcia-Molina, D. Barbara, How to assign votes in a distributed system, Journal of the ACM 32 (4) (1985) 841–860.

Mustafa Mat Deris received the B.Sc. from University Putra Malaysia, M.Sc. from University of Bradford, England and Ph.D. from University Putra Malaysia. He is a professor of computer science in the Faculty of Information Technology and Multimedia, KUiTTHO, Malaysia. His research interests include distributed databases, data grid, database performance issues and data mining. He has published more than 80 papers in journals and conference proceedings. He was appointed as one of editorial board members for International Journal of Information Technology, World Enformatika Society, a reviewer of a special issue on International Journal of Parallel and Distributed Databases, Elsevier, 2004, a special issue on International Journal of Cluster Computing, Kluwer, 2004, IEEE conference on Cluster and Grid Computing, held in Chicago, April, 2004, and Malaysian Journal of Computer Science. He has served as a program committee member for numerous international conferences/workshops including Grid and Peer-to-Peer Computing, (GP2P 2005, 2006), Autonomic Distributed Data and Storage Systems Management (ADSM 2005, 2006), WSEAS, International Association of Science and Technology, IASTED on Database, etc. Currently he is the Dean of the Faculty of Information Technology and Multimedia, KUiTTHO, Batu Pahat.

Dr. Jemal H. Abawajy a faculty member in the School of Engineering and Information Technology at Deakin University where he is the "deputy leader" of the IT Security stream and directing the "Pervasive Computing & Networks" as well as the "Wireless Sensor Networks" research groups at Deakin University. He is actively involved in funded research in robust, secure and reliable resource management for pervasive computing (mobile, clusters, enterprise/data grids, web services) and networks (wireless and sensors) and has published more than 100 research articles in refereed international conferences and journals as well as a number of technical reports. He is currently principal supervisor of 5 Ph.D. and co-supervising 3 Ph.D. students. Dr. Abawajy has guest-edited several international journals and served as an associate editor of international conference proceedings. In addition, he is on the editorial board of several international journals. Dr. Abawajy has been a member of the organizing committee for over 60 international conferences serving in various capacity including chair, general co-chair, vice-chair, best paper award chair, publication chair, session chair and program committee. He is also a frequent reviewer for international research journals (e.g., FGCS, TPDS and JPDC), Ph.D. examinations and research grants (e.g. ARC).

Dr. Ali Mamat is an associate professor in computer science at University Putra Malaysia Serdang. He obtained his Ph.D. in Computer Science from University of Bradford, U.K. in 1992. He has published more than 50 papers in international journals and proceedings. His research interests include databases, XML storage and web semantics.