A Framework for Fault-Tolerant Distributed Mutual Exclusion and Replica Control Using Grid Structures

Jehn-Ruey Jiang

Department of Computer Science

National Tsing Hua University

HsinChu, Taiwan, 30043

R. O. C.

8th International Conference on Parallel and Distributed Systems,
pp.311-315, Chicago IL, Oct. 1996.

A Framework for Fault-Tolerant Distributed Mutual Exclusion and Replica Control Using Grid Structures

Abstract

This paper presents a framework for fault-tolerant distributed mutual exclusion and replica control algorithms utilizing overlapping grid quorums, which are sets constructed with the aid of grid structures. There are two components of the framework: the first one concerns generic representation of grid quorums, and the second one, generic verification of the overlapping properties of grid quorums. This framework not only allows us to view and verify a variety of grid quorum based algorithms clearly and consistently, but also provides us with opportunities to exploit grid quorums for mutual exclusion and replica control. With the effectiveness of the framework, we have devised five novel methods of grid quorum construction for distributed mutual exclusion and replica control in this paper.

Keywords: Distributed system, fault-tolerance, mutual exclusion, quorum, replica control

1. Introduction

Overlapping quorums are widely used for synchronizing nodes over a distributed system. For example, algorithms in [2, 8-13] apply overlapping quorums to achieve mutual exclusion. The concept of these algorithms is simple: a node should collect permissions (votes) from all nodes of a quorum to enter the critical section. If we can assure that any pair of quorums have a non-empty intersection (i.e., the overlapping property holds) and that a node grants its permission to only one node at a time, mutual exclusion is then guaranteed. Such algorithms are fault-tolerant in the sense that a quorum may still be formed even when some nodes are unavailable due to node and/or communication link failures. Many replica control algorithms [1, 3-7] also apply a similar concept to ensure consistency of replicated data. The difference between the mutual exclusion and the replica control algorithms is that (1) the latter has two types of quorums—read and write quorums, each for the execution of read and write operations, and (2) the overlapping property should hold for any pair of write quorums and any pair of a read quorum and a write quorum.

Many of the above-mentioned algorithms [1, 3-4, 7-10, 12-13] take advantage of grid quorums, which are sets constructed with the aid of grid structures. This motivates us to generalize these grid quorum based algorithms. Thus, the goal of this paper is to develop a framework for fault-tolerant distributed mutual exclusion and replica control algorithms using grid quorums. There are two components of the framework: the first one concerns generic representation of grid quorums; and the second one, generic verification for the overlapping properties of grid quorums. This framework not only allows us to view and verify a variety of grid quorum based algorithms clearly and consistently, but also provides us with opportunities to exploit grid quorums for mutual exclusion and replica control. With the effectiveness of the framework, we will propose five novel methods of grid quorum construction for distributed mutual exclusion and replica control in this paper.

The rest of this paper is organized as follows. In Section 2, we propose generic notations for representing grid quorums; we also introduce some lemmas that can facilitate the verification of grid quorums' overlapping properties. In Section 3, we represent the quorums of the algorithms [1, 3-4, 7-10, 12-13] with our generic notations, and show how to prove the quorums' overlapping properties with the lemmas provided in Section 2. We then introduce five novel methods of grid quorum construction for distributed mutual exclusion and replica control in Section 4. The correctness proofs of these methods are related to the lemmas provided in Section 2. And finally, we conclude this paper with Section 5.

2. The framework

The framework consists of two components: the first one concerns generic representation of grid quorums, and the second one deals with generic verification of grid quorums' overlapping properties. Below, we describe the first component in Section 2.1 and the second component in Section 2.2.

2.1 Generic notations for grid quorums

Assume that there are N nodes in the system and they are logically organized as a grid structure of R rows and C columns. In Figure 1, for example, 12 nodes are organized as a 3-row by 4-column grid structure (N=12, R=3, C=4). Below, we introduce notations to represent sets (quorums) that partially overlap rows and columns of the grid structure.

Notation 1. The pair $(\#r, *c)$ denotes a set that overlaps c columns by r nodes each.

Notation 2. The pair $(*r, \#c)$ denotes a set that overlaps r rows by c nodes each.

Notation 3. The pair $(\#r, \leftarrow)$ denotes a set that overlaps column i by all its nodes, $1 \leq i \leq C$, and meanwhile overlaps column 1,...,column $i-1$ by r nodes each.

Notation 4. The pair $(\uparrow, \#c)$ denotes a set that overlaps row i by all its nodes, $1 \leq i \leq R$, and meanwhile overlaps row 1,...,row $i-1$ by c nodes each.

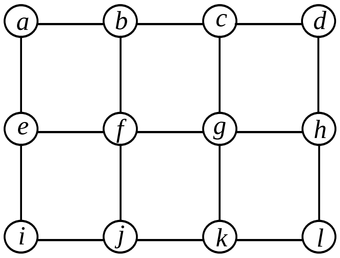

Figure 1. A 3-row by 4-column grid structure composed of 12 nodes.

Take the grid structure in Figure 1 for example again, $\{a, j, g\}$ and $\{b, k, h\}$, etc. are denoted as $(\#1,*3)$; $\{a, e, b, j, h, l\}$ and $\{b, f, g, k, d, h\}$, etc. are denoted as $(\#2,*3)$; $\{a, c, d, i, j, k\}$ and $\{e, f, h, j, k, l\}$, etc. are denoted as $(*2,\#3)$; $\{a, b, c, d\}$, $\{e, f, g, h\}$ and $\{i, j, k, l\}$ are denoted as $(*1,\#4)$; $\{a, e, i\}$, $\{b, f, j\}$, $\{c, g, k\}$ and $\{d, h, l\}$ are denoted as $(\#3,*1)$; $\{a, e, i\}$, $\{a, b, f, j\}$ and $\{e, j, c, g, k\}$, etc. are denoted as $(\#1,\leftarrow)$; $\{a, i, b, f, j\}$ and $\{a, e, f, j, c, g, k\}$, etc. are denoted as $(\#2,\leftarrow)$; $\{a, b, c, d\}$ and $\{b, e, f, g, h\}$, etc. are denoted as $(\uparrow,\#1)$; $\{a, b, e, f, g, h\}$, and $\{b, c, e, h, i, j, k, l\}$, etc. are denoted as $(\uparrow,\#2)$.

2.2. Generic proofs for overlapping properties of grid quorums

In this subsection, we show five lemmas concerning overlapping properties of grid quorums. These lemmas are useful in proving the correctness of grid quorum based mutual exclusion and replica control algorithms in terms of overlapping properties.

<u>Lemma 1</u>. Let Q_1 and Q_2 be sets denoted respectively by $(\#r_1,*c_1)$ and $(\#r_2,*c_2)$, where $1 \le r_1$, $r_2 \le R$ and $1 \le c_1$, $c_2 \le C$. Q_1 overlaps Q_2 if $r_1+r_2>R$ and $c_1+c_2>C$.

Proof:

Since $c_1+c_2 > C$, there must be at least one column that yields r_1 nodes to Q_1 and r_2 nodes to Q_2. Since $r_1+r_2 > R$, there is at least one node belonging to both Q_1 and Q_2. This concludes that Q_1 overlaps Q_2. □

<u>Lemma 2</u>. Let Q_1 and Q_2 be sets denoted respectively by $(*r_1,\#c_1)$ and $(*r_2,\#c_2)$, where $1 \le r_1$, $r_2 \le R$ and $1 \le c_1$, $c_2 \le C$. Q_1 overlaps Q_2 if $r_1+r_2>R$ and $c_1+c_2>C$.

Proof:

Since $r_1+r_2 > R$, there must be at least one row that yields c_1 nodes to Q_1 and c_2 nodes to Q_2. Since $c_1+c_2 > C$, there is at least one node belonging to both Q_1 and Q_2, This concludes that Q_1 overlaps Q_2. □

Lemma 3. Let Q_1 and Q_2 be sets denoted respectively by $(\#R,*c)$ and $(*r,\#C)$, where $1 \le r \le R$ and $1 \le c \le C$. Then, Q_1 overlaps Q_2.

Proof:

Because Q_1 contains c full columns of nodes, Q_1 must take c nodes from each row. And since Q_2 contains r full rows of nodes, there are $c \times r$ (≥ 1) nodes belonging to both Q_1 and Q_2. This concludes that Q_1 overlaps Q_2. □

Lemma 4. Let Q_1 and Q_2 be sets denoted respectively by $(\#r_1,\leftarrow)$ and $(\#r_2,\leftarrow)$, where $1 \le r_1, r_2 \le R$. Then, Q_1 overlaps Q_2.

Proof:

Without loss of generality, we may assume that Q_1 and Q_2 contain all nodes of columns i and j, $i<j$, respectively. Q_1 must overlap Q_2 because Q_1 should contain all the nodes of column i while Q_2 should contain r_2 nodes of column i. □

Lemma 5. Let Q_1 and Q_2 be sets denoted respectively by $(\uparrow,\#c_1)$ and $(\uparrow,\#c_2)$, where $1 \le c_1, c_2 \le C$. Then, Q_1 overlaps Q_2.

Proof:

Without loss of generality, we may assume that Q_1 and Q_2 contain all nodes of rows i and j, $i<j$, respectively. Q_1 must overlap Q_2 because Q_1 should contain all the nodes of row i while Q_2 should contain c_2 nodes of row i. □

3. Representing quorums of grid quorum based algorithms

Below, we represent the quorums of the grid quorum based algorithms [1, 3-4, 7-10, 12-13] with the notations proposed in Section 2.1, and relate their correctness proofs concerning overlapping properties to the lemmas provided in Section 2.2. The quorums are listed in chronological order. Note that we assume there are N system nodes and they are organized as a grid structure with R rows and C columns.

3.1. Maekawa's mutual exclusion algorithm [8]

- Quorums: $(*1, \#C) \cup (\#R,*1)$.

In this algorithm, a quorum is required to contain both a full row of nodes and a full column of nodes. If a square grid is assumed (i.e., R=C=$\sqrt{N}$), then the algorithm is fully distributed, i.e., all quorums are of the same O($\sqrt{N}$) size, and each node appears in the same number of quorums. The overlapping property of quorums can be inferred from Lemma 3.

3.2. Agrawal and El Abbadi's first replica control algorithm [1]

- Write quorums: $(*1, \#C) \cup (\#R,*1)$.
- Read quorums: either $(*1, \#C)$ or $(\#R,*1)$.

This algorithm extends Maekawa's algorithm [8] for controlling replicated data. In this algorithm, a write quorum is defined as that of Maekawa's algorithm, and a read quorum should contain either a full row of nodes or a full column of nodes. The write-write and the read-write overlapping properties of the quorums can be shown on the basis of Lemma 3.

3.3. Neilsen's replica control algorithm [9]

- Write quorums: $(*1, \#C) \cup (\#R,*1)$.
- Read quorums: either $(\#1,*C)$ or $(*R,\#1)$.

This algorithm further improves Agrawal and El Abbadi's algorithm [1] by using higher-availability read quorums, where the *availability* means the probability that a quorum can be formed in an error-prone environment. Note that $(\#1,*C)$ covers $(*1, \#C)$ and $(*R,\#1)$ covers $(\#R,*1)$, i.e., there are more sets that can be represented as $(\#1,*C)$ than as $(*1, \#C)$ and more sets that can be represented as $(*R,\#1)$ than as $(\#R,*1)$. This accounts for the higher read availability of Neilsen's algorithm. The write-write overlapping property can be inferred from Lemma 3, and the read-write overlapping property can be inferred from Lemmas 1 and 2.

3.4. Cheung, Ammar and Ahamad's replica control algorithm [4]

- Write quorums: $(\#1, *C) \cup (\#R, *1)$.
- Read quorums: $(\#1, *C)$.

In this algorithm, a read quorum should contain one node for each column, which is called a *column cover* in [4], whereas a column cover plus a full column of nodes can constitute a write quorum. The write-write and read-write overlapping properties can be inferred from Lemma 1.

3.5. Kumar, Rabinovich and Sinha's replica control algorithm [7]

- Write quorums: $(\#1, *C) \cup (\#R, *1)$.
- Read quorums: either $(\#1, *C)$ or $(\#R, *1)$.

This algorithm improves Cheung et al's algorithm [4] by allowing a read quorum to contain either a column cover or a full column of nodes (the same improvement also appeared in [9]). The write-write and read-write overlapping properties can be inferred from Lemma 1.

3.6. Agrawal and El Abbadi's second replica control algorithm [3]

- Write quorums: $(\#R, *\lceil (C+1)/2 \rceil)$
- Read quorum: $(\#1, *\lceil (C+1)/2 \rceil)$

In this algorithm, a write quorum is required to contain all nodes from a majority of columns (i.e., $\lceil (C+1)/2 \rceil$ columns), while a read quorum is required to contain only one node from a majority of columns. The write-write and read-write overlapping properties can be inferred from Lemma 1.

3.7. Agrawal and El Abbadi's third replica control algorithm [3]

- Write quorum: $(\#\lceil (R+1)/2 \rceil, *\lceil (C+1)/2 \rceil)$
- Read quorum: $(\#\lceil (R+1)/2 \rceil, *\lceil (C+1)/2 \rceil)$

In this algorithm, both the read and the write quorums are required to contain a majority of nodes (i.e., $\lceil(R+1)/2\rceil$ nodes) from a majority of columns (i.e., $\lceil(C+1)/2\rceil$ columns). The write-write and read-write overlapping properties can be inferred from Lemma 1.

3.8. Wu's first mutual exclusion algorithm [12]

- Quorums: either $(\#1, *(C-k+1)) \cup (\#R, *k)$, for some k, $1 \le k \le C$
  or $(*l, \#C) \cup (*(R-l+1), \#1)$, for some l, $1 \le l \le R$.

In this algorithm, there are two types of quorums: a type-1 quorum contains k columns of nodes and one node from $(C-k+1)$ columns, where $1 \le k \le C$, and a type-2 quorum contains l rows of nodes and one node form $R-l+1$ rows, where $1 \le l \le R$. The overlapping properties for any pair of two type-1 quorums, any pair of two type-2 quorums and any pair of a type-1 quorum and a type-2 quorum can be inferred from Lemmas 1, 2 and 3, respectively. It is worth mentioning that Wu's algorithm outperforms Maekawa's algorithm and Cheung et al's algorithm; i.e., if a quorum can be formed in Maekawa's or Cheung et al's algorithms then a quorum can be formed in Wu's algorithm, but not vice versa.

3.9. Wu's second mutual exclusion algorithm [13]

- Quorums: either $(\#1, *(C-k+1)) \cup (\#R, *k)$ for some k, $1 \le k \le C$
  or $(\#l, *C) \cup (\#(R-l+1), *1)$, for some l, $1 \le l \le R$.

In this algorithm, there are two types of quorums. A type-1 quorum contains k columns of nodes and one node from $(C-k+1)$ columns, where $1 \le k \le C$. And a type-2 quorum contains an l-node column cover and a column of $(C-l+1)$ nodes, where $1 \le l \le C$ and an *l-node column cover* stands for a set that contains l nodes for each column. The overlapping properties (including a type-1 quorum overlapping a type-1 quorum, a type-2 quorum overlapping a type-2 quorum, and a type-1 quorum overlapping a type-2 quorum) can all be inferred from Lemma 1.

3.10. Shou and Wang's mutual exclusion algorithm [10]

- Quorums: $(\#1, \leftarrow)$.

In this algorithm, a quorum should contain all nodes of some column i and one node from each of column 1,...,column i–1, where $1 \le i \le C$. The overlapping property of quorums can be inferred from Lemma 4. Note that in the best case, all nodes in column 1 alone can constitute a quorum. That is to say, the smallest quorum size is of constant R when $R << C$. This is a desirable property because the message overhead of a quorum based algorithm is directly proportional to the quorum size.

4. New methods for quorum construction

In this section, with the effectiveness of the framework, we devise five novel methods of quorum construction for distributed mutual exclusion and replica control. As we will show, the correctness of these methods in terms of overlapping properties can easily be inferred from lemmas provided in Section 2.2.

4.1. The first method of quorum construction for mutual exclusion

- Quorums: either $(\#1, \leftarrow)$ or $(\uparrow, \#1)$

In this method, a quorum is required to contain either (1) all nodes of some column i and one node from each of column 1,...,column i–1, where $1 \le i \le C$ or (2) all nodes of some row j and one node from each of row 1,...,row j–1, where $1 \le j \le R$. Note that in the best case, only the nodes in column 1 or row 1 are sufficient to form a quorum. Thus, the smallest quorum size is the smaller one between R and C. The overlapping property of this method can be inferred from Lemmas 3, 4, and 5 along with the fact that $(\#1, \leftarrow)$ contains a full column of nodes (i.e., $(\#R, *1)$) and $(\uparrow, \#1)$ contains a full row of nodes (i.e., $(*1, \#C)$).

4.2. The second method of quorum construction for replica control

- Write quorums: either $(\#1, \leftarrow)$ or $(\uparrow, \#1)$
- Read quorums: $(\#1, *C) \cup (*R, \#1)$

This method is an extension of the first method. The write quorum construction is the same as that of the first method, and the read quorum should contain one node from each column and one node from each row. The write-write overlapping property of this method can be shown in a similar way taken by the first method. And the read-write overlapping property can be inferred from Lemmas 1 and 2 along with the fact that (#1,←) contains a full column of nodes (i.e., (#*R*,∗1)) and (↑,#1) contains a full row of nodes (i.e., (∗1,#*C*)).

4.3. The third method of quorum construction for replica control

- Write quorums: (#1, ←).
- Read quorums: either (#1, ←) or (#1,∗*C*).

In this method, the write quorum construction is the same as that of Shou and Wang's algorithm [10]. And a read quorum is either the same as a write quorum or is a column-cover. The write-write overlapping property can be inferred from Lemma 4, and the read-write overlapping property can be inferred from Lemmas 1 and 4 and the fact that (#1,←) contains a full column of nodes (i.e., (#*R*,∗1)).

4.4. The fourth method of quorum construction for replica control

- Write quorums: either (#1, ∗(*C*–*k*+1)) ∪ (#*R*, ∗*k*), for some *k*, 1≤*k*≤*C*
  or (∗*l*, #*C*) ∪ (∗(*R*–*l*+1),#1), for some *l*, 1≤*l*≤*R*.
- Read quorums: (#1,∗*C*) ∪ (∗*R*,#1)

The write quorums of this method are the same as those of Wu's first mutual exclusion algorithm [12], and a read quorum should contain one column cover and one row cover (note that a *row cover* is defined to be a set that contains one node from each row). The write-write overlapping property can be inferred from Lemmas 1, 2 and 3 (in the same way taken by Wu's first mutual exclusion algorithm [12]), and the read-write overlapping property can be inferred from Lemmas 1 and 2.

4.5. The fifth method of quorum construction for replica control

- Write quorums: either (#1,∗*C*) ∪ (#*R*, ∗1)
  or (∗1,#*C*) ∪ (∗*R*,#1)
- Read quorums: (#1,∗*C*) ∪ (∗*R*,#1).

In this method, there are two types of write quorums. A type-1 write quorum contains a column of nodes and a column cover, while a type-2 write quorum contains a row of nodes and a row cover. And a read quorum should contain both a column cover and a row cover. The overlapping properties for any pair of two type-1 write quorums, any pair of two type-2 write quorums and any pair of a type-1 write quorum and a type-2 write quorum can be inferred from Lemmas 1, 2 and 3, respectively. The read-write overlapping property can be inferred from Lemmas 1 and 2.

5. Conclusion

In this paper, we have proposed a framework for fault-tolerant distributed mutual exclusion and replica control algorithms that utilize overlapping grid quorums. The contribution of this paper is three-fold. First, we have provided generic notations to represent quorums for a variety of grid quorum based algorithms, allowing us to view grid quorum based algorithms clearly and consistently. Second, we have provided generic proofs for the overlapping properties of grid quorums. These proofs can facilitate the correctness proofs for grid quorum based algorithms in terms of overlapping properties. And third, with the effectiveness of the framework, we have proposed five novel methods of quorum construction for distributed mutual exclusion and replica control.

In our future work, we will concentrate on making the framework more complete. We are planning to develop generic procedures to produce grid quorums and generic analysis tools to measure performance of grid quorums on the aspects of quorum size, quorum availability, and so on.

References

[1] D. Agrawala and A. El Abbadi, "Exploiting logical structures in replicated databases," *Inf. Process. Lett.*, vol. 33, no. 5, pp. 255-260, Jan. 1990.

[2] D. Agrawala and A. El Abbadi, "An efficient and fault-tolerant solution for distributed mutual exclusion," *ACM Trans. Comp. Syst.*, vol. 9, no. 1, pp. 1-20, Feb. 1991.

[3] D. Agrawal and A. El Abbadi, "Resilient logical structures for efficient management of replicated data," in *Proc. of the 18th VLDB Conf.*, Canada, pp. 151-162. 1992.

[4] S. Y. Cheung, M. H. Ammar and M. Ahamad, "The grid protocol: a high performance scheme for maintaining replicated data," in *Proc. Int'l Conf. on Data Engineering,* pp. 438-445. 1990.

[5] D. K. Gifford, "Weight voting for replicated data," in *Proc. 7th ACM SIGOPS Symp. Oper. Syst. Principles*, Pacific Grove, CA, pp. 150-159, Dec. 1979.

[6] A. Kumar, "Hierarchical quorum consensus: a new algorithm for managing replicated data," *IEEE Trans. Comp*., vol. 40, no. 9, pp. 996-1004, Sept. 1991.

[7] A. Kumar, M. Rabinovich and R. K. Sinha, "A performance study of general grid structures for replicated data," in *Proc. 13th IEEE Int'l Conf. on Distrib. Comput. Syst.*, pp. 178-185, May 1993.

[8] M. Maekawa, "A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems," *ACM Trans. Comput. Syst*., vol. 3, no. 2, pp. 145-159, May 1985.

[9] M. L. Neilsen, "Quorum structures in distributed systems," Ph. D. Thesis, Kansas State University, May 1992.

[10] D. Shou and S. D. Wang, "An efficient quorum generating approach for distributed mutual exclusion," *Journal of Information Science and Engineering,* vol. 9, pp. 201-227, June 1993.

[11] R. H. Thomas, "A majority consensus approach to concurrency control," *ACM Trans. Database Syst*., vol. 4, no. 2, pp. 180-209, June 1979.

[12] C. Wu, "A fault tolerant O($\sqrt{N}$) algorithm for distributed mutual exclusion," in *Proc. of 1993 Int'l Phoenix Conf. on Computers and Communications,* pp. 175-180, 1993.

[13] C. Wu, "Achieving high performance and fault tolerant for distributed mutual exclusion," Technical Report, University of Illinois at Urbana-Champaign, 1993.