http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (16): 3781-3784, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.3781.3784

Symmetric Distributed Quorum Generation Algorithm Based on Backup Nodes

LI Meian, Wang Buyu, Chao Lumeng and Liu Jiangping
College of Computer and Information Engineering, Inner Mongolia Agriculture University, Huhehaote, Inner Mongolia, 010018, China

Abstract: An advanced symmetric distributed quorum generation algorithm based on limited recursion and backup nodes has been proposed in this paper to reduce the time complexity of a symmetric distributed quorum generation algorithm efficiently and ensure the quorum length not increased significantly. It reduced the time complexity through reducing the optional nodes at every recursion level significantly without increasing the quorum length and the space complexity significantly. Theoretical analysis shows that it's quorum length is $N^{1/2}$, it's space complexity is O(N) and it's time complexity is $O(N^{1/2}!)$. Experimental results show that this algorithm reduced the real time complexity 79% at least to the limited recursion quorum generation algorithm when $37 \le N \le 51$.

Key words: Symmetric, distributed, quorum generation algorithm, backup nodes

INTRODUCTION

For satisfying the computing demands, many advanced computing models such as distributed computing, parallel computing and cloud computing had been proposed. At the same time the distributed mutual exclusion as a basic method to maintain the data consistence and resource mutual access for these advanced computing models becomes more and more important.

Earlier years the distributed mutual exclusion mainly used the master-slave mode. Lamport (1978) and Ricart and Agrawala (1981) proposed some new distributed mutual exclusion algorithms based on message communication. These algorithms are called as global mutual algorithms because of all nodes involved in the mutual exclusion process.

Global mutual exclusion has some properties that the nodes in the distributed system had equipotent accountability, shorter synchronization time but higher message complexity. Maekawa (1985) proposed a local or limited distributed mutual exclusion algorithm based on quorum. It reduced the nodes number involved in the mutual exclusion process from N to $\sqrt{N}$ but increased the synchronization time to 2T. Maekawa (1985) algorithm can generate symmetric quorum when n*(n-1)-1 = N but can not generate a symmetric quorum when n* (n-1)-#N.

So, Maekawa (1985) proposed four conditions to determine if it is a symmetric quorum. According these conditions, Li *et al*. (2005) proved four conclusions as follow:

- Cyclic quorum is symmetric quorum
- In cyclic quorum array, if the first quorum interacts with every other quorum, any two quorums must interact with each other
- The quorum can be described by 0-1 vector
- Cyclic quorum has the rotational symmetry property. This property describes that if the quorum of node 0(Call it quorum 0) interacts with the quorum of node i(Call it quorum i), quorum 0 must interact with quorum N-i

For searching the symmetric quorum with the shortest length (Li *et al*., 2012a, b) proposed some new algorithms based on recursion. These quorum generation algorithms have still higher time complexity.

In this study, a new quorum generation algorithm will be proposed. It reduce the optional node number from N-n or N- n*(n-1) to n at the nth recursion calculating and the time complexity is very lower than the global or limited recursion algorithms when the quorum length is same to the quorum length of the global or limited recursion algorithms.

ALGORITHM IDEA

Proposed distance theorem to simplify the procedure of quorum generation

Definition 1 Distance of two nodes in quorum: Among any numbered distributed system, the distance of two nodes means the numbers difference of the two nodes. In

Corresponding Author: LI Meian, College of Computer and Information Engineering, Inner Mongolia Agriculture University, Huhehaote, Inner Mongolia, 010018, China

a quorum denoted by 0-1 vector it means the indexes difference of two 1.

Distance theorem: Among the cyclic quorum array, if the distance of two nodes in quorum 0 is k, quorum 0 must interact with quorum k.

Proof: Assume the indexes of these two nodes with distance k in quorum 0 are i and i+k, because it's a cyclic quorum, quorum k can be generated by quorum 0 through cyclic shifting towards to right k bits. So node i+k and (i+2k)%N should be included in quorum k. Because there is a common node i+k in quorum 0 and quorum k, quorum 0 and quorum k must interact with each other at node i+k. End.

Definition 2 Distance denoted vector: Use a 0-1vector with N elements to denote if the distance has been covered by the difference set of quorum 0. Call this 0-1 vector as distance denoted vector. The indexes of elements in distance denoted vector denote the distances should be covered by the difference set of quorum 0.

All elements in the distance denoted vector are 1 denotes all distances should be covered had been covered. This means quorum 0 had interacted with all other quorums in the cyclic quorum array. Because the distance of any two nodes in quorum 0 is larger than 0, element 0 of the distance denoted vector can not be set to 1 by the quorum generation algorithm. So set element 0 of the distance denoted vector to 1 before the beginning of the quorum generation algorithm running.

Quorum length theorem: The cyclic quorum length is same as Maekawa (1985) algorithm's quorum length, is $\sqrt{N}$ too.

Proof: Because the difference of two nodes in quorum 0 denotes the index of an element in the distance denoted vector, the length of difference set should be larger than or equal to the length of distance denoted vector. And cyclic quorum has the rotational symmetry property. So the length n of the difference set generated by the quorum and the system scale N should satisfy the equation (2) as follow:

$$C_n^2 \geq (N-1)/2 \quad (2)$$

Equation (2) can be transformed to equation (3) as follow:

$$n*(n-1)/2 \geq (N-1)/2 \quad (3)$$

Equation 3 can be transformed to equation (4) as follow too:

$$n*(n-1)+1 \geq N \quad (4)$$

Equation (4) is same as equation (1). So the cyclic quorum length is same as the Maekawa (1985) algorithm's quorum length, are all $\sqrt{N}$ End.

Proposed backup nodes to reduce the optional nodes at every recursion level: The basic reason that these recursion quorum generation algorithms have higher time complexity is that there are so many optional nodes and the start number almost equal to N-1. So it would be best that change the number of optional nodes at the first recursion to 1, the number of optional nodes at the second recursion is 2 and so on.

When there is only one node in quorum 0, if quorum 0 must interact with quorum k, according to the distance theorem, there must be two nodes in quorum 0 with distance k. Because there is only one node in quorum 0 before, if the index of the first node adds k or subtracts k and set the new location to 1, these two nodes must have the distance k. So the index of the new node should be (i+k)%N or(i-k)%N if the first node's index is i. (i+k)%N will be selected in this paper. Assume the distance should be covered is k, add k to the indexes of all the nodes in quorum 0, n optional nodes can be gotten and select any of these nodes to insert in quorum 0 can ensure distance k can be covered. When the number of nodes in quorum 0 increases from 1 to $\sqrt{N}$ the number of optional nodes becomes from 1 to $\sqrt{N}$. That's to say, the number of optional nodes reduced from N-n*(n-1)-1 to n at the nth recursion level. The time complexity of the quorum generation algorithm should decrease to a lower level.
Definition 3 Backup nodes In the process of optional nodes generated as above, the number of optional nodes at the nth recursion level is limited and is same as the number of nodes in quorum 0. Call the set of those optional nodes as backup nodes.

Searching those quorums not interacting with quorum 0 from low to high and not care the additional distance when a new node inserted in quorum and assume $n*(n-1)+1 \geq N$ the recursion depth should be $\sqrt{N}$ the number of backup nodes should be 1, 2, 3..... $\sqrt{N}$. And the time complexity of quorum generation algorithm should be $\sqrt{N}$.

Design the core process of quorum generation algorithm through limit the quorum length: The quorum generation algorithm will be proposed in this paper will use the idea of cyclic quorum.

According to the distance theorem, if there are two nodes in quorum 0 with distance k, quorum 0 can interact with quorum k in cyclic quorum array. If the elements of distance denoted vector are not all 1, select any element

of distance denoted vector with a value of 0 to generate the backup nodes. Then select one node in the backup nodes to attempt to inserted it in quorum, generate the distance denoted vector and repeat the process above until all the elements of distance denoted vector are 1.

The process described above can not ensure the quorum length is $\sqrt{N}$. For ensuring this condition, one lowest quorum length should be given to limit the quorum length. After a node has been inserted in quorum, the lowest quorum length should subtract 1. If a quorum with $\sqrt{N}$ nodes is not existed, add 1 to the lowest quorum length to generate quorum again until a quorum with the lowest length has been gotten. So the core process of the quorum generation algorithm proposed in this paper should be described as follow:

- Initiate the quorum and the distance denoted vector
- Generate backup nodes based on the quorum and the distance denoted vector
- Attempt to select one node in backup nodes to insert in the quorum
- Judge the algorithm running over or not

ALGORITHM DESCRIPTION

Data structures and processe:

- **Q[N]:** Quorum vector. Use it to record those nodes included in quorum 0
- **S[N]:** Distance denoted vector. Use it to record the state that quorum 0 interacts with other quorums
- **B[]:** Backup nodes. Use it to store those nodes that can be included in quorum 0 at the next step
- **Z[]:** Zero state vector. Use it to store those locations with 0 values in the distance denoted vector
- **int[] backup(int[] quorum, int[]zero):** Use it to search and return those backup nodes
- **int[] generateState(int[] quorum):** Use it to generate the distance denoted vector
- **int[] testInsert(int[] quo, int back[], int k):** It's a core process. Use it to test weather a node in the backup nodes can be inserted in the quorum successfully This process is a recursion process.

Algorithm description: According to the data structures and the processes described above, the running process of the quorum generation algorithm proposed in this paper described as Fig. 1.

PERFORMANCE ANALYSIS

Quorum length: The lowest quorum length has been limited in this algorithm. So the quorum length should be same as the quorum length of the limited recursion algorithm or the global recursion algorithm, are all $\sqrt{N}$. Show the comparison of the real quorum length with several algorithms as Table 1 when N∈[37, 51]. In Tab.1, CA represents the cyclic quorum generation algorithm, LRA represents the limited recursion algorithm, LLA d represents the lowest limit quorum length algorithm and ARA represents the advance recursion algorithm proposed in this paper. And N represents the distributed system scale.

When N∈[37, 51], the quorum length of ARA proposed in this paper is very close to the quorum length of LLA or LRA.

```
Algorithm Descripition

InPUT: N
OUTPUT Q
PROCESS DESCRIPITION

Input (N);
Init(Q);
While (Q.testInsertQ.quorumQ.backup(Q.quorum.
Q.searchStateZer(Q.generatateState(Q. quorum) )), k-2-nill)}
            Q =new Quorum(n);
            k = k+1
}
Output  (Q) ;
Output   (Q.lemgth) ;
```

Fig. 1: Formalized description of algorithm

Time complexity: The basic aim that ARA proposed in this paper is to reduce the time complexity of quorum generation algorithm.

The global recursion quorum generation algorithm has N-n optional nodes when there were n nodes in the quorum. The atte mpt times of this algorithm should be $(N-1)*...*(N-\sqrt{N}) = (N-1)!\sqrt{(\sqrt{N}-1)}$. LRA has N-(n-1)*n-1 optional nodes when there were n nodes in the quorum. So the attempt times of this algorithm should be:

$$(N-1)...*(N-(\sqrt{N}-1)*\sqrt{N}-1)=\prod_{n=1}^{\sqrt{N}}(N-n*(n-1)-1)$$

Use the average value of N-n*(n-1) to get the approximate value of this outcome:

$$\prod_{n=1}^{\sqrt{N}}(N-n*(n-1)-1)\approx N^{\sqrt{N}}/2^{\sqrt{N}}$$

can be gotten. ARA only has n optional nodes when there are n nodes in the quorum. When the quorum length is $\sqrt{N}$ the attempt times should be:

Table 1: Comparison with several quorum generation algorithms

N	37	38	39	40	41	42	43	44
LLA	7	7	7	7	7	7	7	8
CA	9	10	9	10	9	10	10	11
LRA	7	8	7	8	8	8	8	8
ARA	8	8	8	8	8	8	8	8
N		45	46	47	48	49	50	51
LLA		8	8	8	8	8	8	8
CA		11	12	10	12	11	11	11
LRA		8	8	8	8	8	8	8
ARA		8	8	8	9	9	9	9

Table 2: Comparison the real time of several quorum generation algorithms

N	WKA	LRA	ARA	Ratio (%)
37	1.03E+07	21348	921	4.3142
38	6.15E+07	7725089	930	0.0120
39	1.54E+07	17516	930	5.3094
40	9.55E+07	11446634	939	0.0082
41	1.18E+08	13608195	939	0.0069
42	2.70E+07	17516	940	5.3665
43	1.77E+08	19401328	1121	0.0058
44	1.77E+08	41898	4389	10.4754
45	2.16E+08	21671	2218	10.2349
46	2.61E+08	42481	5997	14.1169
47	3.14E+08	28647	5904	20.6095
48	3.77E+08	1194468	5998	0.5021
49	4.51E+08	1774481	3229	0.1820
50	5.37E+08	28142109	6310	0.0224
51	6.37E+08	5677414	6310	0.1111

$$\prod_{n=1}^{\sqrt{N}} n = (\sqrt{N})!$$

So the time complexity of ARA should be $O((\sqrt{N})!)$. Show the real attempt times comparison of LLA, LRA and ARA when $N\epsilon[37, 51]$ as Ta.2 when the initiation nodes all are two. Among it Ratio is the real times ratio of ARA and LRA.

From Table 2 it's sure that the time complexity of ARA is far less than the time complexity of LRA. The maximum Ratio is 20.6% when $N \epsilon[37, 5]$. It means ARA reduced the time complexity at least 79% to LRA and the quorum length is ensured $\sqrt{N}$ when $N\epsilon[37, 51]$.

Space complexity: ARA needs one quorum vector, one distance denoted vector and one backup nodes vector, so the space complexity is $O(N)$. It's same as the space complexity of LLA or LRA.

ACKNOWLEDGMENTS

This study is subsidized by the National Natural Science Foundation of China (61063004) and Natural Science Foundation of Inner Mongolia(2010BS0902.)

CONCLUSION

One advanced symmetric distributed mutual exclusion quorum generation algorithm ARA has been proposed in this paper. It reduced the time complexity from $O(N^{\sqrt{N}}/2^{\sqrt{N}})$ of LLA and of LRA to $O((\sqrt{N})!)$.on the condition that the quorum length has been ensured to be $\sqrt{N}$, the space complexity has been ensured to be $O(N)$. They are all same as the quorum length and space complexity of LLA and LRA. When $N\epsilon[37, 51]$. ARA can reduce the time complexity 79% at least to LRA.

REFERENCES

Lamport, L., 1978. Time, clocks and the ordering of events in a distributed system. Commun. ACM., 21: 558-565.

Li, M.A., L. Lin and Z.D. Chen, 2012a. Distributed cyclic quorum generation algorithm based on binary plus one. Comput. Eng., 38: 59-61.

Li, M.A., L. Lin and Z.D. Chen, 2012b. Quorum generation algorithm of dynamic and multi-node initiation based on local recursion. J. Comput. Appl., 32: 606-608.

Li, M.A., X.S. Liu and Z. Wang, 2005. A high performance distributed mutual exclusion algorithm based on cyclic coding. Acta Electronica Sinica, 33: 1397-1402.

Maekawa, M., 1985. An algorithm for mutual exclusion in decentralized systems. ACM Trans. Comput. Syst., 3: 145-159.

Ricart, G. and A.K. Agrawala, 1981. An optimal algorithm for mutual exclusion in computer networks. Commun. ACM, 24: 9-17.