Comparative Study of Mutual Exclusion in Distributed Environment

Varsha Bhatia
Research Scholar
Dept of Computer Sc & Engineering
Amity University, Haryana 122413 India

Parveen Kumar, PhD
Professor & Head
Dept of Computer Sc & Engineering
Amity University, Haryana 122413 India

ABSTRACT
The traditional mutual exclusion problem in distributed systems occurs when only one process should access a shared resource. The mutual exclusion algorithm performance is calculated by the number of messages exchange per critical section execution called Message complexity and the delay between successive executions of the critical section, known as Synchronization delay. For designing mutual exclusion algorithm, one has to compromise either for the message complexity or for the synchronization delay. Hence a comparative study based on these two metrics is performed. An organized approach is essential to solve Mutual exclusion problem. This study will provide a suitable context for technical and clear assessment of existing algorithms.

General Terms
Comparison of Distributed mutual exclusion algorithms.

Keywords
Distributed Mutual Exclusion, Mutual Exclusion, Critical Section (CS), Synchronization Delay (SD), Message Complexity.

1. INTRODUCTION
A distributed system is a set of independent computers, which appears to its user as a single consistent system, and which are capable of collaborating on a task. The problem of mutual exclusion is one of the basic problems in distributed systems. The computers or sites neither have shared memory nor the global clock hence these sites communicate with each other by passing messages. A Distributed Mutual Exclusion Algorithm frames the laws for managing entry into the CS and mediates conflicts when more than one site needs to execute a CS simultaneously. The algorithms depend on message exchanges between sites to coordinate entry into the CS. In event of sharing of a resource between several processes, only one process is allowed to use it at a time. Message ordering in the distributed system is done, by either using timestamps or sequence number. Time stamps are assigned to messages according to **Lamport's logical clocks** [1].

1. System Model
A common model is used in general, for the majority of mutual exclusion algorithms. A distributed system is a group of physically isolated autonomous computers connected via a communication network. The system comprises of N sites, $Site_1$, $Site_2$, --, $Site_N$. The term site is used for referring both process and computer executing the process. All the processes communicate asynchronously over a fundamental communication network through message passing.

1.2 Types ofMutual Exclusion

Algorithms
Many algorithms exist to solve mutual exclusion in distributed systems. They are classified into two groups. First group is token-based, where a unique token exist in the system which ensures mutual exclusion. Only the token possessing node can enter the critical section. Second one is Non Token based where to enter the critical section, a requesting node has to seek permissions from all other nodes, In this paper, for comparison algorithms are classified in four groups as Token-based, Non-token, Hybrid and K-mutual exclusion. Hybrid is based on combination of both token and non token approach and K mutual algorithm uses either of them for k resources. In hybrid approach sites are grouped in Local groups and Global group, and different algorithms are used to resolve Local (inter group) and Global (Intra group) conflicts. By framing rules for interaction between the local and the global algorithms, one can minimize both message traffic and synchronization delay simultaneously. The K-mutual exclusion problem is elementary distributed problems that guarantee the assigning of the k units of a shared resource by restraining the number of processes that can access them at the same time. These algorithms can again follow either token-based or permission-based approach.

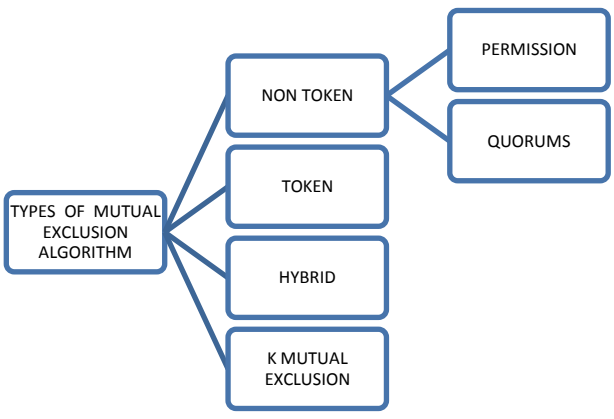

Figure1 Types of Message Passing Mutual Exclusion Algorithm

1.3 Performance Metrics

1.3.1 Message Complexity - It is the number of messages sent or received by a process for each CS execution.

1.3.2 Synchronization Delay - It is the time interval between when a site leaves the CS, and the next site enters the CS. Average message delay is represented by 'T' and is measure of delay.

1.3.3 Response Time - It is the time gap which a request waits, starting from the time its request message was sent out, to the time of completion of its CS execution.

1.3.4 System Throughput - It is the speed at which CS requests are executed per unit time in a system. If synchronization delay is SD and average CS execution time is E, then the system throughput will be $(\frac{1}{SD+E})$.

1.4 Requirements for Mutual exclusion

Mutual exclusion is essential property for designing distributed systems. It is complex to design mutual exclusion algorithms for distributed system, as these algorithms have to deal with irregular message delays and incomplete information of the system state. Every mutual exclusion algorithm should fulfill following prerequisites'

1.4.1 Mutual Exclusion – No two sites can simultaneously access the critical section.

1.4.2 Freedom From Starvation – It is ensured that there is no process which waits indefinitely to execute CS whereas other processes are repeatedly executing CS.

1.4.3 Deadlock Freedom - Two or more process are blocked indefinitely, waiting for the messages from one another in a circular manner**.**

1.4.4 Fairness - If the order in which requests are made and order of execution of requests are identical in the system, then fairness is ensured.

2 REVIEW OF DIFFERENT MUTUAL EXCLUSION ALGORITHMS

2.1 Token Based Algorithms

In the token-based approach, the privilege to enter a critical section is facilitated by a unique object, named token. In token-based algorithms in place of timestamps, sequence numbers are used. One of the initial token based mutual exclusion algorithms is by **LeLann [2]**.In this algorithm a token circulates on a ring of sites. Algorithms differ in the way how the nodes are arranged, the way requests are stored, and the token is distributed. **Suzuki and Kasami [3]** presented a broadcast based algorithm, where a site who wants to enter CS, increases it sequence number and broadcast request message along with sequence number to all sites. When a site receives the request message, it records the sequence number. Site holding the token sends token, after completing execution of its CS, else token is granted immediately. The main function of the sequence numbers is to resolve the ambiguity between current and old requests. Before releasing the token, the token holder checks to see which processes have pending requests and add those processes to the request queue in the token (if it is not there already) and sends the token to the top request in the queue. **Raymond[2]**[4] gave a algorithm based on a static logical **tree structure**, which remains unchanged but direction of its edges changes dynamically as token propagates. Raymond's algorithm works by maintaining a pointer at each node, which is represented by the "holder" variable. The holder variable points to a node, that either holds the token, or to a node which is a link between the current process and the token. In this way, a tree is maintained in which the root of the tree holds the token. Each process also contains a request queue. In case a process wants to enter a critical section, the process initially adds itself to its request queue, and forwards a request message to its "holder". When the request is received, holder process adds the request to its queue and forwards the request to the next holder until it reaches the token holding node. Once the request reaches to the token holder, and the holder wants to release the token, it sends the token to the first request on the queue. If there are further requests on the queue, it also sends a request to the new holder. Requesting and releasing the token tends to make message sequences traverse the branches of a tree. Traversing a tree can be difficult in the worst case when the tree is large. **Naimi and Trehel[2][5]** algorithm does not use sequence numbers. Instead, it organizes the sites into a dynamic, logical, rooted tree and the last site present in the request queue is the root of the tree. This algorithm uses two data structures, a queue to store requests and a logical rooted dynamic tree for assigning token. A queue of pending request is implicitly maintained by each node; hence the nodes and the token are not required to maintain the queue of pending requests. **Singhal [2][6]** developed a heuristically-aided algorithm to achieve mutual exclusion which uses state information to reduce message traffic. **Mishra and Srimani [2][6]** extended the algorithm of Suzuki and Kazami [3] and incorporated a fault tolerance mechanism, to recover the system from a single node failure.

Table1. Comparison of Token-Based Algorithms

Algorithm	Sync Delay	Messages Complexity (HL)	Messages Complexity (LL)	Description
Le Lann [2]	–	1	O(N)	Uses Ring structure
Suzuki kazami [3]	T	N	0	Token as a privilege message.
Naimi Trehel [2][5]]	T	O (log N)	O(log N)	Uses two structures: Queue & logical tree.
Raymond [2] [4]	$T\left(\frac{\log N}{2}\right)$	4	O(log N)	Static logical Tree.
Singhal Heuristic based [2][6]	T	$\frac{N}{2}$	N	Uses state information and heuristics.
Mishra Srimani [2]	2T	$(L*N)+(N-1)$ L>2	0	Regeneration of new token and the elimination of duplicated tokens.
Nishio [2]	T	N	0	Fault tolerant. Regeneration of new and Elimination of duplicate tokens are possible.

Using Queue Migration [7]	–	2	O($\sqrt{N}$)	Local and Global groups are formed each of $\sqrt{N}$ nodes.

In this algorithm, it is possible to regenerate the lost token and the state of each site can be reconstructed. **Nishio algorithm [2]** is based on Suzuki and Kazami algorithm [3], but the sites are arranged in a logical circular list. The token is issued to the nearest node, whose last request has not been served. Fault tolerance and token regeneration is possible. In **Distributed Mutual Exclusion Using Queue Migration [7]** algorithm system is divides the n nodes of the system into$\sqrt{n}$ sets of $\sqrt{n}$ nodes each, Nodes are grouped in local and global group consisting of $\sqrt{n}$ nodes**.**

2.1.1 Evaluation of Token Based Algorithms

The advantage of token based approach is its simplicity. Scalability is a problem as waiting time of each node is directly proportional to number of nodes. Message traffic is less in token based algorithms as compared to non-token based algorithms. But their resiliency to failure is poor as token loss and its regeneration are major issues. Fair scheduling of token among competing sites is also a concern. Efficiency is more in case of heavy load situation.

2.2 Non Token Based Algorithms

Lamport [1][2] provided the first truly distributed mutual exclusion algorithm. Each process maintains its own request queue. Assuming N process system model, a requesting process sends a time stamped request to all other $(N-1)$ processes. A process can enter critical section when permission from all other processes is received, and its request is next in its ordered request queue. This ensures the mutual exclusion condition. A process that receives a request message sends back a time stamped reply message to the requesting process. When a process releases its critical section, it sends a $(N-1)$release message to notify that its request has been granted. **Lamport [1][2]** algorithm is fair as the request for CS are executed in the order of their time-stamps. The algorithm necessitates$(N-1)$ requests, $(N-1)$ replies, and $(N-1)$ releases, or $3(N-1)$ messages for a critical section execution. **Ricart and Agrawala [8]** inferred that if all sites must grant permission by sending replies, then the release messages are not required, since a reply involves an implicit release. It reduced message complexity to $2(N-1)$. **Carvalho and Roucairol [2][6]** algorithm has further improved the number of messages by avoiding some unnecessary requests and reply messages and reduces message complexity to $[0,2(N-1)]$. In **Makewa's [9]** algorithm, a set of sites called a quorum is associated with each site. Permission has to be taken only from a subset of the sites, these subsets are called quorums. Any pair of set (quorum) has a nonempty intersection among them. To enter the CS, a site locks all sites in its quorum. Although the message complexity is dramatically reduced, but this can lead to deadlock. The number of messages required is $c\sqrt{n}$ messages per mutual exclusion where c lies between 3 and 5. The **Agarwal–El Abbadi [6]** quorum-based algorithm, constructs quorums from trees. Such quorums are called "tree-structured quorums. The sites are logically arranged into a tree with a well-defined root. A quorum is constructed by selecting any path starting from the root and ending with any of the leaves. If a path is not available due to the failure of a node i, then i is substituted with two paths, both (all) starting with the children of node i and terminating with leaves. This algorithm requires O (logN) messages per critical section execution. S**inghal [10]** proposed a dynamic information structure algorithm which reduces message traffic by cleverly initializing an information structure and updating it as the algorithm evolves. **Lodha and Kshemkalyani's Fair [11]** is best known algorithm that guarantees fairness, it assigns priority based on FIFO property. Message complexity is $(N-1)$ under low load conditions and $2(N-1)$ in high load condition.

2.2.1 Evaluation of Non Token Based Algorithms

Requests for access to the critical section are satisfied, based on priority determined by their timestamps, therefore fairness is guaranteed. Permission-based algorithms have high message overhead in their communications as compared with the token-based algorithms. Due to this permission based algorithm are generally slower than token based algorithm. Message overhead was optimized to logarithm factor of N by quorum based strategy. Quorum based algorithm have low message complexity and high resiliency. However, the major disadvantage with quorum based algorithm are recreating and managing quorums.

Table2. Comparison of Non Token Based Approach

Algorithm	Sync Delay	Messages Complexity (HL)	Messages Complexity (LL)	Description
Lamport [1][2]	T	$3(N-1)$	$3(N-1)$	Give priority with Time-stamp.
Ricart Agarwala [8]	T	$2(N-1)$	$2(N-1)$	Uses Implicit Release Message strategy.
Carvalho Roucairol [2]	T	$2(N-1)$	0	Uses Implicit No Reply Message strategy..
Maekawa [9]	2T	$5\sqrt{N}$	$3\sqrt{N}$	Uses Quorum. Prone to deadlock.
Agarwal Abbadi [2]	2T	O(log N)	O(log N)	Uses Tree structured quorum.
Singhal Dynamic [10]	T	$3(N-1)/2$	$(N-1)$	Uses Dynamic Information Structure.
Fair Mutual exclusion algorithm [11]	T	$2(N-1)$	$(N-1)$	Give priority with FIFO.

2.3 Hybrid Algorithms

Paydar et al [12] algorithm It uses a two-dimensional array based logical topology with quorum-based concept. In which it arranges n nodes in two-dimensional array. This array is composed of $\sqrt{n}$ rows and $\sqrt{n}$ columns. **Kakugawa [6] [13]** quorums share minimum one process. A **Hybrid token based algorithm [14]** consists of two dimensional wrap around network composed of N nodes, which communicate by message exchanges. It applies the concept of token asking in rows and continuous mobility of the token in columns. This algorithm was further enhanced by using information based technique in **Info-based approach in distributed mutual exclusion algorithms [15].** It is also based on wrap around two dimensional logical topology, and token based approach. By info-based it is meant, that requesting node forwards CS requests directly to the token holding node, as some nodes are informed nodes and they know the present location of the token.

Table3. Comparisons of Hybrid-Based algorithms

Algorithm	Sync Delay	Messages Complexity (HL)	Messages Complexity (LL)	Description
Paydar [12]	-	$4\sqrt{N}$	$4\sqrt{N}$	Quorum contains of same column and row nodes
Kakugawa [13]	3T	5|Q| + 1	0	With coterie and main token & sub token
Hybrid Token Based[14]	-	$\sqrt{N}$	O($\sqrt{N}$)	WTDA topology Uses perpetual mobility and token asking.
Info-Based Approach [15]	-	2	4*$\sqrt{N}$ +1	*WTDA topology but some nodes are informed about the token holding node..

* **WTDA**: Wrap around two dimensional array

2.3.1 Evaluation of Hybrid Algorithms

The algorithms discussed above are logical structured based algorithm which reduces the number of messages considerably. The hybrid approach combines the best of both token and non token based mutual exclusion techniques. **Hybrid algorithm [14]** uses continuous mobility the token. Under very light situation sometimes token circulates uselessly. **Info- based approach [15]** has drawback in heavy load situation, when all the nodes try to enter CS, token will not be forwarded downwards in the two dimensional topology. Only a portion of topology will have frequent access to token, because only token holding node and its neighboring nodes will consecutively use the token.

presented an algorithm that used the Coterie concept. A coterie is set of quorums, which are characterized by non-empty intersection between them. In other words any two

2.3 K Mutual Exclusion Algorithms

A generalization of the mutual exclusion problem is the K-mutual exclusion problem, where no more than K processes are allowed to enter the critical section simultaneously**.** These algorithms can be categorized as token-based or permission based algorithms**. Raymond [16]** was the first to provide a solution to the K-mutual exclusion. **Raymond [16]** algorithm is permission based which extends the algorithm of Ricart and Agrawala [4] to allow up to K nodes in the system to execute the critical section (CS) simultaneously. A node is allowed to enter the CS when at least N - (K - 1) nodes are not executing within the CS. **Srimani and Reddy [17]** developed an algorithm based on Suzuki and Kazami's algorithm [4]. *K* tokens exits in the system to allow *K* nodes to execute simultaneously in their CS. In this algorithm *K* is fixed. **Kakugawa et al** [18] adopted Maekawa's algorithm [2], but a K-coterie is constructed**. K-queue migration [19]** is extension of distributed mutual exclusion using queue migration [21].

2.4.1 Evaluation of K Mutual Exclusion

Ever**y** algorithm designed to solve the K-mutual exclusion problem must ensure that, a node seeking entry to the critical section is permitted to do so, if less than K nodes are in CS. It is denied access, if K processes are already executing CS. Performance of K mutual exclusion is restricted depending on whether algorithm is token based or non token based. In **Raymond's [16]** and **Srimani-Reddy's [17]** algorithms, a major concern is to maintain the information about the state of the system. In **Srimani-Reddy [17]** algorithm *K* tokens are generally updated at different nodes. Hence, each token will have different state information about the system. The information must be updated to avoid sending tokens to requests which are already served A major task in **Kakugawa's algorithm [18]** is to construct the quorums for the K-coterie, which is a complex task.

Table4. Comparison of K Mutual exclusion

Algorithm	Sync Delay	Messages Complexity (HL)	Messages Complexity (LL)	Description
Raymond with multi entries [16]	–	$(2N) - 1$	(2N) – K – 1	Permission based K resources & N-k replies
Srimani and Reddy [17]	–	$N + (K - 1)$	0	Token based approach, K tokens
Kakugawa et al [18]	–	–	$(3 \times S)$ to $(5 \times S)$	Quorum Based. S is the quorum size

K-queue migration [19]	4T	O($\sqrt{N}$)	0	Use parent token & generate k-1 tokens

3 CONCLUSION

The distributed mutual exclusion has been seeking research attention since its beginning. In this paper, only the basic concepts of algorithms and performance metrics are discussed, each kind of algorithm has its own varying characteristic and performance. A comparative study of their performance based on message complexity, and synchronization delay is done. These two attributes used can effectively compare the algorithms.

Token-based algorithms are highly susceptible to the loss of the token. Complex mechanisms, based on time-outs, must be executed in order to regenerate a lost token and to discard duplicate tokens. Permission based algorithm have high message overhead, which can be reduced by reducing the permission set. A certain type of structure called coterie reduces message overhead and cost significantly. In order to reduce message complexity, few algorithms use logical structures and information based technique. Few algorithms are dynamic in nature; nodes are made aware of the current state of the system, and algorithm incorporates various methods to take decision depending on the current state. The K Mutual algorithms presented are designed to allow multiple processes to execute the CS simultaneously, so that a higher level of concurrency could be achieved.

4. REFERENCES

[1] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," Communications of the ACM, vol. 21, no. 7, pp. 558-565, 1978.

[2] Velaquez, M. G, 1993, A Survey of Distributed Mutual Exclusion Algorithms. Technical Report. Colorado State University.

[3] I. Suzuki and T. Kasami., "A distributed mutual exclusion algorithm," ACM Transactions on Computer Systems, vol. 3, no. 4, pp. 344-349, 1985.

[4] K. Raymond, " A tree based algorithm for distributed mutual exclusion," ACM transactions on computer system, pp. 61-77, February 1989

[5] Trehel and M. Naimi, " A distributed algorithm for mutual exclusion based on data structures and fault tolerance.," in IEEE 6th International Conference on Computers and Communications, 1987

[6] Leila Omrani, Z. R. "A new function-based framework for classification and evaluation of mutual exclusion algorithms in distributed system," International Journal of Computer Science and Security (IJCSS), Vol - 5 , no. 2, pp. 168-297, 2011.

[7] P. Chaudhuri and T. Edward, "An O($\sqrt{n}$) Distributed Mutual Exclusion Algorithm Using Queue Migration," Journal of Universal Computer Science, vol. 12, no. 2, pp. 140-159, 2006

[8] G. Ricart and A. K. Agrawala, "An optimal algorithm for mutual exclusion in computer networks," Communications of the ACM, vol. 24, no. 1, pp. 9-17, 1981

[9] M. Maekawa, "A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems," ACM Transactions on Computer Systems, vol. 3, no. 2, pp. 145-159, 1985.

[10] M. Singhal, "A dynamic information-structure mutual exclusion algorithm for distributed systems," IEEE Transactions on Parallel and Distributed systems, vol. 3, no. 1, January 1992.

[11] S.Lodha, and A.Kshemkalyani, " A fair distributed mutual exclusion algorithm," IEEE Transactions on Parallel and Distributed Systems, vol. 11, no. 6, June 2000. .

[12] S. Paydar, M. Naghibzadeh and A. Yavari,"A hybrid distributed mutual exclusion algorithm", in IEEE - International Conference on Engineering and Technology, 2006.

[13] H.Kakugawa, S. Kamei, and T. Masuzawa, "A token-based distributed group mutual exclusion algorithm with quorums," Parallel and Distributed Systems, IEEE Transactions, vol. 19, no. 9, pp. 1153- 1166, 2008.

[14] H.Taheri, P.Neamatollahi, and M. Naghibzadeh, "A hybrid token-based distributed mutual exclusion algorithm using wraparound two-dimensional array logical topology," Information Processing Letters, pp. 841-847, 2011

[15] H.Taheri, P.Neamatollahi, and M. Naghibzadeh, "Info-based approach in distributed mutual exclusion algorithms," Journal of Parallel and Distributed Computing, vol. 72, no. 5, pp. 650-655, May 2012.

[16] K. Raymond, "A distributed algorithm for multiple entries to a critical section," Information Processing Letters, February 1989.

[17] P. Srimani and R. Reddy, "Another distributed algorithm for multiple entries to a critical section," Information Processing Letters, vol. 41, no. 1, pp. 51-57, Januaury 1992.

[18] H.Kakugawa; S. Fujita; M.Yamashita and AE, T., "Availability of k-Coterie," IEEE Transactions on Computers, vol. 42, no. 5, pp. 553-558, May 1993.

[19] P. Chaudhuri and T. Edward, "An algorithm for k-mutual exclusion in decentralized systems," Computer Communications, vol. 31, no. 14, 2008.

[20] Kshemkalyani,A. and Singhal,M. 2008. Distributed Computing: Principles, Algorithms, and Systems. pp 305-348

[21] M.Bouillaguet, L.Arantes, and P.Sens, "Fault tolerant k-mutual exclusion algorithm using failure detector," in International Symposium on Parallel and Distributed Computing, 2007. .

[22] P. Saxena and J. Rai, " A survey of permission-based distributed mutual exclusion algorithms," Computer Standards & Interfaces, vol. 25, no. 2, pp. 159-181, May 2003.

[23] Abhishek Swaroop and Awadhesh Kumar Singh, " A Study of Token Based Algorithms for Distributed Mutual Exclusion," International Review on Computers & Software;Jul2007, Vol. 2 Issue 4, pp 302,July 2007