© 2017 IJSRST | Volume 3 | Issue 8 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology

Single Queue Based Algorithm for Mutual Exclusion iIn Distributed Systems

Ashish Chauhan[1], Chandrabhushan Prasad[2], Kanupriya[3]

[1]Department of Computer Science & Engineering, IIMT College of Engineering, Greater Noida, Uttar Pradesh, India
[2]Department of Computer Science & Engineering, IIMT College of Engineering, Greater Noida, Uttar Pradesh, India
[3]Department of Electronics & Communication, Shri Ram Group of colleges, Muzaffarnagar, Uttar Pradesh, India

ABSTRACT

Distributed System is a class of computing systems in the field of computing where the hardware or software components of the system are located at networked locations. Computers that are the part of this system can communicate and coordinate their action only by exchange of messages in the system. Mutual exclusion is a mechanism in which multi-process can make access to the single sharable resource without affecting the integrity of the resource. The number of messages among the sites of the distributed system is one of the very prime concerned issue in analysing the performance of any algorithm. Also the amount of data structures needed in the one of prime consideration in the performance analysis of algorithm. The algorithm proposed in this paper reduces the number of messages to a large extend and also there is need of a single queue as a data structure.

Keywords : Distributed System, Critical Section, Single Queue Algorithm, Non-Token, Mutual Exclusion.

I. INTRODUCTION

A system is said to be distributed if in the system the hardware and software components are located at network computers and the only way by which these components can communicate is through the exchange of messages among themselves. The prime limitation of Distributed system is the Absence of shared memory as well as the absence of global clock. One of the very interesting problem with which the distributed system has to deal with is the problem of mutual exclusion. Mutual exclusion is a mechanism in which multi-process can make access to the single sharable resource without affecting the integrity of the resource. The requests are needed to be serialized in such a way that the requests of all the sites must be fulfilled along with the non-affecting the integrity of the shared resource. Thus it is quite clear that the action which is being performed on the shared object by the process must need to be atomic. The problem of Mutual Exclusion frequently arises in Distributed System whenever concurrent access to shared resources by several sites is involved. To maintain the consistency of a system, it is necessary that the shared resources will be accessed by single user at a time. Distributed Mutual Exclusion algorithm can be classified into two types –

1.1 Token based algorithm: A unique type of algorithm in the field of implementation of mutual exclusion in the distributed systems is Token based Algorithm. In this approach a unique token is shared among all the sites and a site is allowed to enter its Critical Section only if it possesses the valid token. Also one thing is to be considered that the token based algorithm uses Sequences no, which distinguish the current request from previous request for the token, every time a site makes a request for a token, it increments its sequences no counter and merge it with request messages. Example of Token based is Raymond's algorithm (log (N) messages) and Suzuki Kasami Algorithm (N messages).

1.2 Non-Token based algorithm: Another class of mutual exclusion algorithm is the Non-Token based algorithm. In this approach a site communicate with set of others site of its concern to decide who should execute critical section next. This algorithm uses the unique concept of the Timestamp to order the request for Critical section in order to avoid the conflict between sites. Example of Non-Token based are Lamport`s algorithm 3(N-1) messages and Ricart-Agrawala`s algorithm 2(N-1) messages

II. OBJECTIVES OF MUTUAL EXCLUSION IN DISTRIBUTEDSYSTEMS

The primary objective of a Mutual Exclusion algorithm is to maintain Mutual Exclusion; that is, to guarantee that only one request accesses the Critical section at a time. In addition, the following characteristics are considered important in a Mutual exclusion algorithm:

2.1 Freedom from Deadlocks

Two or more sites should not endlessly wait for messages that will never arrive.

2.2 Freedom from Starvation

A site should not be forced to wait indefinitely to execute Critical section while other sites are repeatedly executing Critical section. That is, every requesting site should get an opportunity to execute a Critical Section in a finite time.

2.3 Fairness

Fairness dictates that request must be executed in the order they are made (or the order in which they arrive in the system). Since a Physical Global clock does not exist, time is determined by Logical clocks. Note that Fairness implies freedom from starvation, but not vice-versa.

2.4 Fault Tolerance

A Mutual Exclusion algorithm is fault tolerant if in the wake of failure, it can re-organise itself so that it continues to function without any disruptions.

III. PERFORMANCE METRICS OF MUTUAL EXCLUSION ALGORITHM

3.1 Response Time

It is the time interval a request waits for its Critical Section Execution to be over after its Requested messages have been sending out. Smaller the Response Time betters the performance.

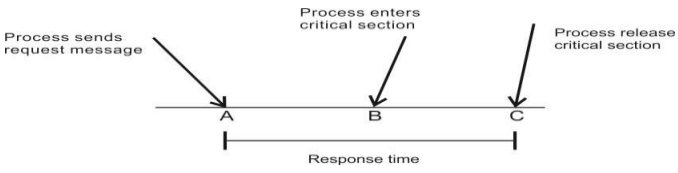

Figure 1. Response Time

3.2 Synchronization Delay

The time interval after the site exits the critical section and before the next site enters the critical section is called synchronization delay. Smaller the synchronization delay better the performance is.

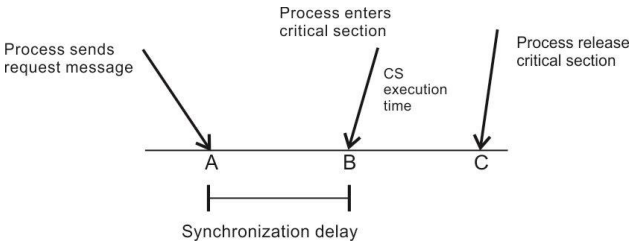

Figure 2. Synchronization Delay

3.3 Number of Messages necessary per critical section Invocation

Before entering in the Critical section, site sends a request messages to all the sites and after the execution of Critical section site send a reply messages to all other sites. Lesser the No. of Messages necessary or Critical section invocation betters the Performances.

3.4 System Throughput The rate at which system executes the request for Critical Section isknown as System Throughput. Maximum the System Through put better the Performances.

System Throughput= 1/ (SD + E)

Where SD is Synchronization Delay and E is Average Critical Section Execution Time.

IV. PROPOSED WORK

System Model: The system is consisting of N sites, Out of those N sites some or all may want to execute in the critical section. The system consists of only a single data structure i.e. denoted as Request_Queue of the System which is responsible for holding the request of all the sites that are willing to execute in the critical section. The requests are arranged in the increasing order of their time-stamp.

At any time site may have several request for critical section. A Request_Queue queues up these request and serves them one at a time.

A site can be in one of the three states-

1. Requesting Critical Section:-the site is blocked and cannot make further requests for Critical section
2. Idle:-In idle state site is executing outside its critical section (no conflict).
3. Executing:-the site is executing in its critical section.

V. ALGORITHM

5.1 Requesting the critical section.

1. When a site Si wants to enter the CS, it sends REQUEST (tsi, i) message to all the sites in its request set Ri as well as to the designated Request_Queue of the system (tsi is the timestamp of the request).

2. When a site Sj receives the REQUEST (tsi, i) message from site Si, it sends a REPLY message to site Si if site Sj is neither requesting nor executing the CS or if site Sj is requesting and Si`s request's timestamp is smaller than Sj's own request's timestamp. The request is deferred otherwise it returns a REPLY message to Si.

5.2 Executing the critical section.

1. Site Si enters the CS when the two following conditions hold:

a) [L1:] Si has received a REPLY message from all other sites.
b) [L2:] Si`s request is at the top Request_Queue.

5.3 Releasing the critical section

1. When site Sj exits the CS, it sends REPLY messages to all the deferred requests and removes its entry from the top of Request_Queue.

When a site exiting the CS removes its request from its Request_Queue, The next entry of the queue come at the top of the queue, enabling it to enter CS. The algorithm executes CS requests in the increasing order of timestamps.

VI. RESULTS AND ANALYSIS

Suppose The system is composed of N Sites, and the space complexity for storing one queue in the system is of order 1. Thus the below is the comparison of the proposed algorithm is with the major algorithms studied in the field of Mutual exclusion in the distributed systems.

Table 1.

Algorithm	Number of Messages Exchanged	Data Structure Used	Space Complexity
Lamport's Algorithm	3(N-1)	N Queues	N
Ricart Agrawala Algorithm	2(N-1)	N Queues	N
Proposed Algorithm	2(N-1)	1 Queue	1

VII. CONCLUSION

This algorithm implements the Mutual Exclusion in a system by arranging the request of the multiple sites of a system in a single data structure known as the Request_Queue. The modification is made by the advantage arrangement of the single data structure ie The Request_Queue. Thus for the system of N sites the space complexity is reduced to 1/N for the system as compared to the previous generated algorithms. All The sites executes in an increasing order of their timestamp. Sites only communicate by passing messages as well as to the Request_Queue. Proposed algorithm uses Single Data Structure, operates faster and has a distributed control.

VIII. REFERENCES

[1]. G. Ricart and A. K. Agrawala, &ldquo, "An Optimal Algorithm for Mu tual Exclusion in Computer Netw orks &rdquo, Co mm".ACM, vol. 24, no. 1, pp. 9-17, Jan. 1981.

[2]. J.-H . Yang and J. An derson, &ldquo,Time Bounds for Mutual Exclusion and Related Problem s &rdquo, Proc. 26th Ann. ACM Symp. "Theory of Co mputing, pp. 224-2 33," May 1994.

[3]. D.A grawal, A.El. Ab badi, "An efficie nt and fault toler ant solution for distributed mutual exclusion"ACM Transaction on Co mputer Systems 9 (1) (1991) 1-20.

[4]. Ashish Chauhan, Kanupriya, "Centralized Approach to mutual Exclusion in Distributed Systems" International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 12, December 2015