PBCBT: AN IMPROVEMENT OF NTBCBT ALGORITHM

Kamla Vivient Corneille[1], Souleymanou[2] and Damakoa Irépran [3]

[1]Department of Mathematics and Computer Science, National School of Agro-Industrial Sciences, University of Ngaoundéré, Cameroon
[2]Department of Mathematics and Computer Science, Faculty of Sciences, University of Ngaoundéré, Cameroon
[3]Department of Applied Mathematics and Computer Science, School of Geology and Mining Engineering, University of Ngaoundéré, Cameroon

ABSTRACT

The classic mutual exclusion problem in distributed systems occurs when only one process should access a shared resource. Various algorithms are proposed in order to solve this problem. When using a permission based approach which consist in exchanging permission messages to grant access to the critical resource, less messages should be sent over the network because bandwidth consumption and synchronization delay should be reduced. Richa, shikha and Pooja proposed an algorithm using nodes logically organized in a complete binary tree. This algorithm called NTBCBT requires 4log2(N) messages per access to critical section and a synchronization delay of 3log2(N) for a set of N nodes competing for the critical ressource. In this paper, we study NTBCBT and we show that this algorithm has problems related with safety, liveness and scheduling. We improve this algorithm by correcting these weaknesses. Moreover, our algorithm requires 3log(N) messages per access to critical section and a synchronization delay of 2log(N). This improvement is due to the removal of useless messages, a reorganization of instructions on each node and an insertion of access requests using their timestamp.

KEYWORDS

Distributed algorithm, mutual exclusion, complete binary tree, permission, critical resource.

1. INTRODUCTION

The growing demand of material resources in computing forced computer manufacturers to combine many processors into a single or a distributed system. This technique known as parallel computing is used to provide simultaneous data processing for the purpose of increasing the computational speed of a computer system.

Parallel computers can be classified according to several criteria. M.J. Flynn [2] proposes a classification of architectures according to the flow of instruction and data manipulated simultaneously. Parallel processing may occur in the instruction stream, in the data stream or in both. Flynn' classification divides computers into four major groups: SISD architectures (Single Instruction Single Data), SIMD architectures (Single Instruction Multiple Data) and MIMD architectures (Multiple Instruction Multiple Data).

The last category (MIMD) refers to a computer capable of processing several programs at the same time. The main characteristic of this architecture is the way how information is shared

DOI : 10.5121/ijp2p.2014.5101

(Distributed or shared memory). Computers with distributed memory are characterized by the absence of global memory or and global clock. Among problems related to global memory, there is the problem of simultaneous access to the same resource. In fact, processes are either in cooperation by exchanging messages or in competition for a shared resource. The simultaneous access to the shared resource may result into an incoherent value for this resource: This problem is known as the problem of mutual exclusion.

Several algorithms have been proposed in order to solve the problem of mutual exclusion. According to Richa, Shikha and Pooja [3], these algorithms can be classified into two main groups: Token based and non-token based algorithms. Examples of token-based mutual exclusion algorithms are Suzuki- Kasami's algorithm [11], (N or 0 messages for each CS), Singhal's heuristic algorithm [10], ([N/2,N] messages), Raymond's tree-based algorithm [8], (log (N) messages), and Naimi et-al.'s algorithm [7], (O(log(N)) messages). Non token-based mutual exclusion algorithms exchange messages to determine which process can access the CS next. Examples of nontoken-based mutual exclusion algorithms are Lamport's algorithm [5], (3(N-1) messages), Ricart- Agrawala's algorithm [9], (2(N-1) messages), Carvalho-Roucairol's [1], modification on Ricart- Agrawala's algorithm ([0,2(N-1)] messages),Maekawa's algorithm [6], ([3rootN, 5rootN] messages), and Singhal's dynamic data structure algorithm [10], ([N-1, 3(N-1)/2] messages). On july 2011, Richa, Shikha and Pooja proposed a non-token based algorithm named NTBCBT (Non Token Based using Complete Binary Tree) which grants access to the critical resource with 4log (N) messages with a synchronization delay of 3log (N).

In this paper, we study NTBCBT and we show that this algorithm has problems related with safety, liveness and scheduling. We improve this algorithm by correcting these weaknesses. Moreover, our algorithm requires 3log(N) messages per access to critical section and a synchronization delay of 2log(N). This improvement is due to the removal of useless messages, a reorganization of instructions on each node and an insertion of access requests using their timestamp.

The rest of this paper is organized as follows: Section 2 presents the NTBCBT algorithm analyses and its weaknesses. Section 3 presents an improved algorithm and analyses its performances. We finally round up with a conclusion summarizing results and further works we intend to continue.

2. NTBCBT ANALYSIS

2.1 General considerations for the algorithm.

NTBCBT run on a distributed system of N physically dispersed autonomous computers sites that logically form a complete Binary Tree and communicate with one another only by sending messages [3]. The sites do not share a common memory or global clock. It assumed that the sites numbered from 1 to N form a tree topology. It is also assumed that the underlying network is reliable and sites do not crash. The delay of a message delivery is unpredictable, but it is finite. The messages between any pair of nodes are transferred in the order they were sent. There are four types of messages used in the algorithm. REQUEST, INFORM, REPLY, and REMOVE [3]. All of them have the same simple format. Every type of messages includes its Sender-ID, a Recipient- ID and a Time-stamp [4]. The four types of messages will be assigned the following semantics. A REQUEST indicates a requesting node request for using the CS. Inform Message is used to inform to children of a node that set their flag variable. Reply message is used to give permission to the requesting node for entering into critical section and REMOVE message is used to exit from critical section [3].

2.2 The NTBCBTalgorithm [3].

Requesting CS

If the s_i site want to enter into CS then

1 Make an entry in the request queue of ith site.
2 It send a request message to its parent located at i/2
3 Inform its children to set their flag to 1and enter id of its children to the RF Array.
4 On getting a request message site j will perform the following
   a. It will make an entry in its request queue
   b. Repeat the following while j!=root
   c. If(flag[j]=1)
      Then
         wait
      Else
         Set child=j
         Set j=j/2
      End of if structure
   End of loop
   d. if (flag[root] =1)
      Then
         wait
      Else
         set flag=1
         send the inform message to another child (other than child) so that it will set its flag variable to 1 and make its entry to RF array.
         send the reply message to the child pointed by child
   [End of if structure]
5 On getting reply message the site will set its flag to 1
6 Remove the topmost entry from the request queue and send the reply message to the site i
7 Send an inform message to other child to set its flag variable and make its entry in the Rf array.

Executing CS

8 On getting reply message from its parent site the site will enter into the CS

Release CS

9 Set its flag variable to 0.
10 Send a REMOVE message to all the sites in the RF Array and delete them from the array.
11 On getting remove message sites will clear their flag variable.
12 Send a release message to its parent if it's not ROOT.
13 On getting release message go to step 9
14 If its request queue is not empty then go to step 4B.

2.3 NTBCBT Weaknesses

2.3.1 The ROOT access problem

In the NTBCBT algorithm, there is an abnormal behavior when the ROOT node requests the critical resource. First, if we assume that the ROOT does not have any parent (we round i/2 to 0 and there is no node numbered 0), then its request will never reach a node and he will never get access to the critical resource. Secondly if we assume that the ROOT is its own parent(we round i/2 to 1 and ROOT will behave as its own parent), it will make an entry in its request queue, Inform its children to set their flag to 1and enter id of its children to the RF Array and send a request message to itself. On getting its own request message site j will make an entry in its request queue again, making a double entry. If flag [root] is not 1, the variable child will not hold any value. Finally, the whole system will be in starvation.

2.3.2 The Message communication problem

On getting a request message a site j should perform the while j!=root loop. In a distributed environment, sites do not share a common memory and cannot access the variable flag of another node. It is impossible for a node to loop over other nodes. It can only read its own flag and send a message to another node. It is noticeable that no request message is sent in the while loop. It is clear that the algorithm cannot work in a fully distributed system.

2.3.3 The scheduling problem

When a node asks the access to the critical resource, it automatically blocks the flags of its children, disabling the transmission of request's messages coming from the children which requests could be earlier for any reason. Meantime, other independent nodes are transmitting their requests. This situation is not favorable to the blocked nodes while their parent is not yet in critical section.

3. AN IMPROVEMENT OF NTBCBT: PBCBT

Regarding different problems mentioned in our last section, we propose here an improvement of the NTBCBT algorithm. We call this algorithm Permission Based using Complete Binary Tree (PBCBT).

3.1 General considerations for the algorithm

We keep the same network topology used in NTBCBT, a distributed system of N physically autonomous computers that logically form a complete Binary Tree and communicate with one another only by sending messages. The sites do not share a common memory or global clock. Sites numbered from 1 to N form a tree topology. Children of a node numbered i are nodes numbered 2i and 2i+1. Parent of a node numbered i is the node numbered i/2. It is also assumed that the underlying network is reliable and sites do not crash. The delay of a message delivery is unpredictable, but it is finite. The messages between any pair of nodes are transferred in the order they were sent.

There are three types of messages: INFORM to request access to critical section, REPLY to grant access to critical section, EXIT to release critical section. Each message includes its sender, its receiver, the initial Lamport's timestamp [4] of the message and the message type.

3.2 Improved algorithm

We present here our improved algorithm. The variable node holds the number of the computer node executing the algorithm. When a node transmits a message REQUEST coming from a child node, it becomes the sender of the message, but the TIMESTAMP of the message remains unchanged.

INITIALISATION

```
procedure init ( )
flag=0
granted=0
msg  = NIL
REQUEST_Q=empty
Process_messages()
```

REQUESTING CRITICAL RESSOURCE

```
procedure request_section ( )
Insert a REQUEST message in REQUEST_Q
If node !=ROOT
        Send the REQUEST message to its parent node.
End if
```

EXTITING CRITICAL SECTION

```
Procedure exit_section( )
flag=0
granted=0
if node== ROOT
        send EXIT message to itself
else
        send EXIT message to the parent node
end if
process_messages( )
```

MESSAGE PROCESSING

```
procedure process_messages( )
while ( flag==0)
        receive (msg)
        if msg==REQUEST
                if node==ROOT
                        if granted=1
                                Insert msg in REQUEST_Q
                        else
                                granted=1
                                send REPLY message to the sender of msg
                        end if
                else
                        Insert msg in REQUEST_Q;
                        Send the REQUEST message to its parent node;
```

```
            End if
        if msg==REPLY
            if node== sender of Head(REQUEST_Q);
                flag=1;
                granted=1;
                remove Head(REQUEST_Q);
                enter critical section;
            else
                granted=1;
                send a REPLY message to the sender of Head(REQUEST_Q);
                remove Head(REQUEST_Q);
            end if
        if msg==EXIT
            granted=0
            if node == ROOT
                if  REQUEST_Q is not empty
                    granted=1
                    send a REPLY to itself
                end if
            else
                send EXIT message to the parent
            end if
        end if
End while
```

3.2 Analysis of the improved algorithm.

We analyse the performance improvement of our algorithm by evaluating the bandwidth and the synchronization delay. Bandwidth is the total number of messages sent per access to the critical section. These messages include among others those requesting, granting and releasing critical resource. Synchronization delay is the number of messages sent from the exit of the critical section by one process to the entry of the next process.

3.2 .1 Bandwidth analysis

PBCBT requires at worst $\log_2$ (N) REQUEST messages, $\log_2$ (N) REPLY messages and $\log_2$ (N) EXIT messages per access to critical section. This is a total of $3\log_2$ (N) messages per access.
We compare in the table below bandwidth of permission based mutual exclusion algorithms for N nodes competing for the critical resource.

Table 1: Bandwidth comparison of permission based mutual exclusion algorithms.

Algorithm	Lamport	Ricart-Agrawala	Carvalho-Roucairol	Maekawa	NTBCBT	PBCBT
Bandwidth	3(N-1)	2(N-1)	0 to 2(N-1)	$2\sqrt{N}$ to $5\sqrt{N}$	$4\log_2$ (N)	$3\log_2$ (N)

From this comparison, it clear that PBCBT is the algorithm with the smallest bandwidth.

3.2 .1 Synchronization delay analysis

PBCBT needs $\log_2$ (N) EXIT messages and $\log_2$ (N) REPLY messages for synchronization. This is a total of $2\log_2$ (N) as synchronization delay.

We compare in the table below the Synchronization delay of permission based mutual exclusion algorithms for N nodes competing for the critical resource.

Table 2: Synchronisation delay comparison of permission based mutual exclusion algorithms

Algorithm	Lamport	Ricart-Agrawala	Carvalho-Roucairol	Maekawa	NTBCBT	PBCBT
Bandwidth	1	1	1	1	$3\log_2$ (N)	$2\log_2$ (N)

PBCBT is not the best at synchronization delay, but improves upon NTBCBT algorithm.

4 CONCLUSION AN FUTURE WORKS

We analysed NTBCBT and found problems related with liveness of the node ROOT resulting into a starvation of the whole system. We also presented an improved algorithm which needs only $3\log_2$ (N) messages per access to the critical section and $2\log_2$ (N) messages as synchronization delay. This algorithm improves upon other permission based algorithms. In future works, we will take fault tolerance into account and improve scheduling of access to the critical resource.

REFERENCES

[1] O.S.F. Carvalho & G. Roucairol (1983). On mutual exclusion in computer networks.Communications of the ACM, Vol.21 :PP.146 – 147.

[2] M.J. Flynn (1996). Very high-speed computing systems. Proceedings of the IEEE, Vol.54.

[3] G. Richa, G. Shikha, & D. Pooja (July 2011)NTBCBT: a distributed mutual exclusion algorithm. International Journal of Peer to Peer Networks (IJP2P), Vol.2, No.3, pp25-31.

[4] L. Lamport (1978). Time, clocks, and the ordering of events in a distributed system.Communications of the ACM, Vol.26 :PP.558 – 565.

[5] L. Lamport (Nov. 1990). Concurrent reading and writing of clocks. ACM Transactions on Computer Systems, Vol.8 No.4 :PP.305 – 310.

[6] M. Maekawa (May 1985). A square-root(n) algorithm for mutual exclusion in decentralized systems. ACM Transactions on Computer Systems, Vol.3 No.4 :PP.145 – 159.

[7] M. Naimi, M. Trehel, & A. Arnold (1996). Distributed mutual exclusion algorithm based on path reversal. Journal of Parallel and Distributed Computing, Vol.34 :PP.1 – 13.

[8] K. Raymond (February 1989). A tree-based algorithm for distributed mutual exclusion. ACM Transactions on Computer Systems, Vol.7 :PP.61 – 77.

[9] G. Ricart & A. Agrawala (January 1981). An optimal algorithm for mutual exclusion in computer networks. Communications of the ACM, Vol.24 :PP.9 – 17.

[10] M. Singhals(2001) “Advanced Concepts IN Operating Systems”Tata McGraw-Hill Edition,Vol- 1, PP 122-132

[11] I. Suzuki & T. Kasami (November 1985). A distributed mutual exclusion algorithm. ACM Transactions on Computer Systems, Vol.3 No.4 :PP.344 – 349.

Authors

KAMLA Vivient Corneille born in Cameroun, on January 31, 1976. After completing, Master's Degree (with thesis) Success Testimonial in Computer Science, Yaounde I University (Cameroun) , 2003, he has done a Joint Guardianship Ph.D. thesis in computer science /Applied Mathematics at the University of Yaoundé I / University of Pau (France), 2008. Presently, he is working as Assistant Lecturer in the Department of Mathematics and Computer Science, National School of Agro-Industrial Sciences, University of Ngaoundéré, Cameroon.

SOULEYMANOU, born in Cameroon on Sept. 25, 1978. After working as survey controller in the National Institute of Statistics of Cameroon for seven years, he completed a Bachelor degree in mathematics and Computer Science with specialization in architecture of systems and networks from University of Ngaoundéré in Cameroon. In 2014, he completed a master in Systems and Software in Distributed environments from the same university. Presently not working permanently, he plans to continue soon with a Ph.D. thesis.

DAMAKOA Irépran, born in Cameroon on 1962, B.Sc. in Math. from the University of Yaoundé I (Cameroun), M.Sc in Engineering Math. from the University of Limoge (France), Ph.D. in Applied Math. from the University of Toulouse III (France). Currently Deputy Director of the School of Geology and Mining Engineering of the University of Ngaoundéré.