Research Journal of Applied Sciences, Engineering and Technology 5(19): 4691-4694, 2013
DOI:10.19026/rjaset.5.4303
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: September 26, 2012 Accepted: December 11, 2012 Published: May 10, 2013

Research Article

The PSRS Algorithm based on Synchronous Barrier

Yuqiang Sun, Huanhuan Cai, Xian Chang, Xin Gao and Yuwan Gu
International Institute of Ubiquitous Computing, Chang Zhou University, Chang Zhou 213164, China

Abstract: The biggest characteristic of LogGP model based on LogP mode is sending long messages, if all the elements to be sent are seem as a long message and sent in a single processor, a sorting algorithm should be introduced to merge those elements, but the algorithm designed in the LogP mode is heavily dependent on the accuracy of parameters such as l, o, g, p. However, parameters are often inaccurate in reality. This may lead to message traffic congestion in the transfer process and the degradation of Communication performance of system. Therefore, this study proposes a new algorithm, that is, synchronization barrier is introduced into PSRS algorithm, which can improve LogGP Model further. Network congestion will be avoided when sending a long message and system performance will be improved .The barrier synchronization method can be applied to other algorithms of LogGP model, so it has a certain practicality.

Keywords: LogGP model, PSRS, sorting algorithm, synchronization barriers

INTRODUCTION

Now many parallel computers provide special support for processing long messages, so a new model called LogGP was formed by introducing processing of long messages into LogP model. Compared with LogP model, the LogGP introduced an extra parameter G, which represents spacing of every word of long messages, that is, the time needed for sending long messages byte by byte. Its count backwards was corresponding to the bandwidth of long messages sent by the processors. In order to send all elements as a long message in a single processor, we should introduce a sorting algorithm to combine the elements. But the algorithm designed under the LogGP model relied heavily on accuracy of parameters such as l o, g, p. However the actual parameters is often not accurate, which could cause communication congestion in the process of transmitting messages, reduce communication performance of the system and make the actual operation results of the algorithm very different from original designed expectations. Therefore, this study proposes a new algorithm, that is, synchronization barrier is introduced into PSRS algorithm, which can improve LogGP Model further. Network congestion will be avoided when sending a long message and system performance will be improved, the barrier synchronization method can be applied to other algorithms of LogGP model, so it has a certain practicality. Hoefler *et al*. (2009) analyse the LogGP in theory and practice-An in-depth analysis of modern interconnection networks and benchmarking methods for collective operations. Zheng and Wang (2003) have a analysis of Parallel computing model LogGP. Shuwei *et al*. (2008) have a research of the barrier synchronization for cell multi-processor architecture. Yang *et al*. (2008) have a research of the parallel 0-1 programming algorithm based on improved PSRS. Wang and Qiu (2005) study a new Algorithm for Parallel mergesorting. Ding *et al*. (1999) study the parallel merging algorithm based on MPP.

THE INTRODUCTION AND ROLE OF BARRICADES SYNCHRONIZATION

The idea of barricades synchronization: There is a common phenomenon-synchronous phenomenon in the parallel processing system. As is shown in Fig. 1, N processes P1, P2…Pn run on the different processors respectively, when one of these processes arrives at the point of interaction in Fig. 1, it should wait for all other processes until they get to this interactive point and then they can continue to go down. How to effectively coordinate the parallel computing of each process (subtask) is an important factor, which directly affect the implementation performance of the entire system. A synchronous technology based on the sharing storage model is often used in many parallel processing systems; it coordinates parallel computing by setting synchronous barricades. An application process can set many synchronous barricades points to coordinate parallel computing steps. The main idea of the barricade synchronization method is that all processes can continue to execute until they arrive at synchronous

Corresponding Author: Yuqiang Sun, International Institute of Ubiquitous Computing, Chang Zhou University, Chang Zhou 213164, China

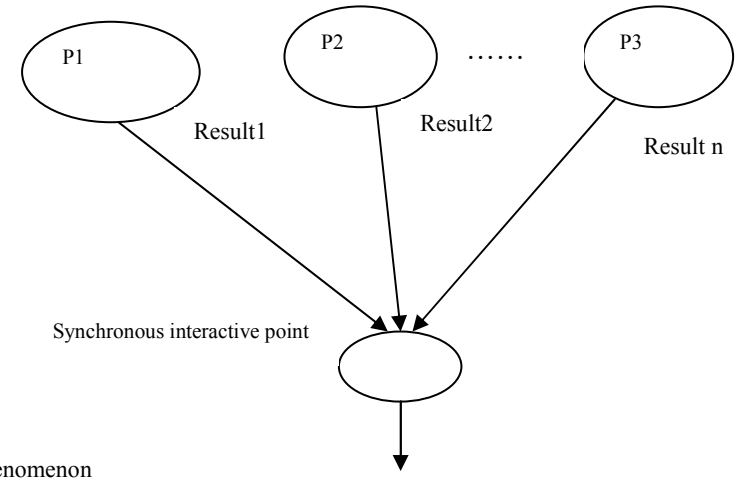

Fig. 1: Synchronous phenomenon

interaction point. If some of these processes can't arrive at the point for some reasons, we should set up barricades to prevent reached processes from continue execution.

The realization way of the barricade: When implementing barricades algorithm specifically, we need to consider each process how to know the synchronization barricades information of other processes. For example, how many processes have arrived at synchronization barricade point at one time, one of the simplest methods is to use centralized way, which uses a global shared variable to record this information and then each process executing the algorithm can test and set this shared variable. The advantage of this method is simple, but when the number of the processors is large, it must cause communication block of the system BUS (MIMD-BUS model) or processor/memory Internet network (MIMD-MIN model). Another improved method is using the "combination" trees, its basic idea is to divide the processor into groups and assign a leaves (equivalent to a shared variables) for each group, so each processor only to test and set the leaves corresponding to its group. If we find that it is already the last processor that changes the state of leaves, then we can deliver final results of state of leaves to their fathers. In turn on, the last processor arriving at the synchronous point delivers the results to the node of the root; this can reduce system overhead caused by continuously system competition for a shared variable in "centralized" method. In addition, it is a technology based on "binary tree competition" that can realize the barricades. In order to discuss simply, this study uses "centralized" method to access to a shared state variable.

The benefits of introducing barricade synchronization mechanism: There are at least two benefits of introducing barricade synchronization mechanism:

- When the capacity of receiving messages of a node arrived at the network capacity constraints, we can separate the messages sent to the same node continually to avoid network congestion.

We can use global flow control, which can make sending and receiving speed of the whole network match. This can improve the performance of the system and to further improve the efficiency of the algorithm.

ALGORITHM DESIGN

PSRS algorithm: PSRS (Parallel Sorting Regular Sample) algorithm is parallel sorted in standard sampling. It is a ranking method based on comparing exchange of key words, it is suitable for distributed computing environment and it has very good characteristics of load balance and good local for memory accessing.

PSRS algorithm by introducing of synchronous barricades: The basic thoughts of the PSRS algorithm based on synchronization barricade are as follows:As is known from the original PSRS algorithm, not until step 3 sends p period of orderly data to step 4, can step 4 merge do p road merge, so here is a problem, the p period of orderly data sequence certainly won't be arrived at the same time, if a data series comes and do a merge sorting at the same time, it will have low time efficiency and will also waste space resources. If a lot of data sequences come suddenly, they will cause information communication congestion in the process of delving messages. So, this study considers inserting a barricade synchronous algorithm between step 3 and step 4, namely p period of orderly data sequence must gain synchronous interaction Point (Interactive Point), which is the processor p0 in this place, then we can continue to execute step 4. If some of these processes can't arrive at the point for some reasons, we should set up barricades to prevent reached processes from continue execution. The improved algorithm is as follows:

Suppose the number of data is n, the number of processor is p and n data evenly distributed on p

Table 1: Comparison of incorporated times of improved and non-improved algorithm

The number of arrived processes	Incorporated times in non-improved algorithm	Incorporated times in improved algorithm
3~4	5	2
5~8	17	3
9~16	44	4
…	…	…
…	…	…
…	…	…

processors, the PSRS algorithm can be divided into the following several steps:

Step1: Each processor gets a good sort of sequence by doing quick sort for own n/p data
Step 2: Each processor selects (p-1) data (w, 2w, 3w… (p-1) w) as delegate elements from sorted series, among them w = n/p2
Step 3: Each processor will choose good delegate element and send it to the processor p0
Step 4: Setting up synchronous barricades, not until P period of orderly sequence arrive at processor p0, can we carry the next operation
Step 5: Processor p0 will merge into p period of orderly data sequence sent from step 3, then choose (p-1) main elements from sorted data, the main elements are p-1, 2(p-1), (p-1)(p-1)
Step 6: Processor p0 will broadcast the (p-1) main elements to all processors
Step 7: Each processor will divide the n/p data into p segments according to the p-1 main elements sent from step 6, remember Wij as the (j + 1) segment of processor p, among them, i = 0, 1…p-1
Step 8: Each processor will delivered the (i+1) segment to processor pi, then make the i processor has the i segment data of all processors, (i=0,1…p-1)
Step 9: Each processor sort data sent from step 8 through p road merge sort, thus making n data more orderly.

The false code of Barricade synchronization algorithm is as follows:

Shared integer: count = 0/*count is an integer shared variables used to record processes arrived at this barricade, the initial value is 0 */
Shared Boolean: g = true/* g is a logical shared variable */
Private Boolean: l = true/* logical variable local to each process */ Adaptive Barrier () l = not l;/ * Each process changes the state of local variables, which is also an initial value of the next barricade ,waking up blocking process */if fetch-and-increment(count) = numb-process-1 then initia-next();wake up();/*Setting an initial value for the next synchronous algorithm and waking up blocking process */
else for test-time = 1 to TestTime do /* testing g in the period of Test Time */ if g =l then exit barrier ();/*entering to the next step until all processes arrive at the synchronous points */

Algorithm performance analysis between the improved and non-improved: The time spent on step 3 and step 4 of non-improved algorithm mainly consists of two parts:

Interval time between two arrived adjacent processes in the p processes is t1, t2,…t (p-1), remember T1=t1+t2+…t（p-1; (2) the merging time of non-improved algorithm is calculated as follows: every time a new process comes, the whole process will do a merge sort, this will waste a lot of time, Comparison of incorporated times of improved and non-improved algorithm is shown in Table 1:

We can see from Table 1, the execution of merging time of improved algorithm are much less than that of non-improved algorithm, so the time complexity of improved merging algorithm is O nlog n), while the time complexity of non-improved merging algorithm is O (nlogn+nlog2). So the total time complexity of non-improved algorithm is O (nlogn+nlog2+T1), that is O (nlogn+nlog2+ t1+t2+…t (p-1)):

- The time spent on step 3 and step 4 of improved algorithm also consists of two parts:
- we can use an average time to compute cycle testing time, which used for setting barricades to wait for all processors' arrival, namely T1' = 1/(p-1)*(t1+t2+…t（p-1))
- Incorporated time needed by the improved algorithm. So total time complexity is O ($n\log n$ + T1'), namely O ($n\log n$+1/(p-1)*(t1+t2+…t (p-1)))
- The time complexity of improved algorithm is less than that of non-improved algorithm.

CONCLUSION

As the algorithm designed in the LogP mode is heavily dependent on the accuracy of parameters such as l, o, g, p, while parameters are often inaccurate in reality. This may lead to message traffic congestion in the transfer process and the degradation of Communication performance of system. Therefore, this study takes a specific algorithm called PSRS ($) algorithm on the model of LogGP for example and propose an improved algorithm, which introduces barricade synchronization mechanism based on PSRS algorithm to further improve the LogGP model, to avoid network congestion when sending long messages, to improve the performance of the system and improve the efficiency of the algorithm. The barrier synchronization method can be applied to other

algorithms of LogGP model, so it has a certain practicality.

ACKNOWLEDGMENT

Supported by The project of general office of Broadcasting and Television (GD10101) and Natural Science Fund in JiangSu (BK2009535) and Natural Science Fund in ZheJiang (Y1100314)

REFERENCES

Ding, W.Q., Y.C. Ji and G.L. Chen, 1999. A parallel merging algorithm based on MPP. J. Comp. Res. Dev., 36(1): 52-56.

Hoefler, T., T. Schneider and A. Lumsdaine, 2009. LogGP in theory and practice-An in-depth analysis of modern interconnection networks and benchmarking methods for collective operations. Simul. Mod. Pract. Theory, 17: 1511-1521.

Shuwei, B., Z. Qingguo, Z. Rui and L. Lian, 2008. Barrier Synchronization for CELL Multi-Processor Architecture. Distributed and Embedded System Lab, SISE, Lanzhou University, China.

Wang, W.Y. and C. Qiu, 2005. A new Algorithm for Parallel mergesorting.

Yang, L., J. Jain and T. Li, 2008. Parallel algorithm of MCQoS routing based on improved PSRS and binary search. Proceedings of the International Conference on Advanced Infocomm Technology, New York, ISBN: 978-1-60558-088-3.

Zheng, W.K. and X.D. Wang, 2003. Analysis of Parallel Computing Model Log GP.