ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING

National Conference on Advanced Innovation in Engineering and Technology (NCAIET-2015)

Alva's Institute of Engineering and Technology, Moodbidri

Vol. 3, Special Issue 1, April 2015

An Approach for Adaptive Load Balancing using Centralized Load Scheduling in Distributed Systems

Merlyn Melita Mathias [1], Manjunath Kotari [2]

Department of Computer Science and Engineering, Alva's Institute of Engineering and Technology, Moodbidri[1,2]

Abstract: With the advances in technology, faster growth in the use of computer systems has led to increase in the applications that share the resources there by increasing the amount of load .As the demand for resource sharing applications is grown, load balancing approach has become a necessity factor. In a distributed network, the overall performance of the system depends on the effective division of the workload among all the available set of processing nodes.

Load balancing is an approach in which the redistribution of load takes place equally among the set of processing nodes in the system. The main objective is to improve the overall performance there by obtaining the best job response time. In general, classification of load balancing algorithms is done into two major types called static and dynamic. In dynamic, the decentralized approach suffers from the communication overhead because of the exchange of information frequently among the processors. In the existing centralized node based load balancing technique, the workload is assigned randomly to the processing nodes. Once allocated, load balancing conditions is checked at every processing node and the job migration is involved.

A centralized node based load scheduling and balancing approach is proposed to overcome the limitations of existing centralized load balancing technique by allocating the random arrival of load among the processing nodes based on FCFS (First Come First Served) scheduling algorithm taking into consideration the current status of every processing node. Here, the load balancing conditions are checked only by the central node to overcome the overhead on the processing nodes. In this approach the transfer of job from one node to another is not done in order to overcome the migration overhead and concentrate much on executing jobs rather than transferring. This job scheduling policy is adaptive that considers the dynamic changes in the system and accordingly schedules and balances the load.

Keywords: Load Balancing, Distributed systems, FCFS, Migration, Communication Overhead, Response Time

I. INTRODUCTION

As the technology is been developing, the demand for the resource sharing application is increased. Load balancing has become an essential factor to address these growing application demands. In a distributed network, the overall performance of the system depends on the effective division of the workload among all the available processing nodes [1]. Load balancing is an approach in which the redistribution of load takes place equally among the set of processing nodes in the system. It will ensure that each and every processor will do the equal amount of work at any instance of time.

In a distributed network, the nodes in the system will be heterogeneous where each processor will be having its own processing power. Estimation of load is done based upon the processing rate as well as the processing speed including overall configuration of the processor. Load will arrive from various users in random interval of time. In the context of assigning the load to the available processing nodes, a proper load scheduling policy is required so as to finish the processing of the load in short period of time [3]. The major benefits of load balancing algorithm includes

- Improving the performance of each processing server and thereby improving overall performance.
- Load balancing reduces the response time of the jobs.
- Achieves greater improvement in the system due to higher system throughput.
- Optimal utilization of resources.

Several load balancing algorithms are being developed. For the dynamic load balancing algorithms to be effective, several important issues like estimation of load, job resource requirement, processing rate of processing nodes, communication overhead need to be considered while developing. Whenever the load balancing is carried out, certain type of information can be exchanged among the processors in order to improve the performance. The parameters taken into consideration include the number of jobs waiting in queue, CPU processing rate at each processor, memory utilization, arrival rate of jobs and availability of nodes.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING

National Conference on Advanced Innovation in Engineering and Technology (NCAIET-2015)

Alva's Institute of Engineering and Technology, Moodbidri

Vol. 3, Special Issue 1, April 2015

Taking into account the classification, load balancing algorithms are broadly divided into two major categories: static and dynamic. In static, the current state of the system is

not considered to make load balancing decisions. In this approach, fixed number of jobs is assigned to a particular processor and prior knowledge is required about the type of the job, the communication network being used and the status of the system. Static approach does not react to the dynamic change that takes place in the system. Whereas in dynamic, current system state is being considered to make more informative load balancing decisions. Dynamic algorithm performs better when compared to static since it is responsive to the dynamic changes in the system. Adaptive load balancing algorithms are a type of dynamic algorithm in which parameters of the algorithm keep changing according to the state of the system.

According to the degree of centralization, load scheduling algorithms are categorized into centralized or decentralized [4]. In centralized, scheduling of the load for processing is done by single processor. This single processor is itself responsible to collect all the parameters which are necessary to make the load balancing decisions. Whereas in decentralized, scheduling of the load is done by many processors, if not all in the distributed system. Centralized algorithms are less reliable than decentralized, whereas the problem in decentralized algorithms is the communication overhead that occurs from the exchange of information frequently among the processors.

There are various methodologies to perform the load balancing as well as scheduling. Here importance is given to centralized algorithm since decentralized algorithms has the problem of communication overhead. In existing centralized node based load balancing, whenever the process arrives it is randomly assigned to processing nodes. The jobs are migrated from heavily loaded node to the lightly loaded. The central node is responsible to handle the overload from the primary nodes and the overload conditions are checked at individual processing nodes.

The rest of this paper is organized as follows. Section II includes the related work which provides an overview of different approaches used for balancing the load. Section III describes the system model and the proposed work. Section IV includes the implementation and results. Conclusion and future work is included in Section V.

II. RELATED WORK

There are various methodologies to perform the load balancing as well as scheduling. For the dynamic load balancing algorithms to be effective, several important issues need to be considered while developing the algorithm. Several approaches for load balancing have been introduced.

A. Static Load Scheduling and Balancing

X. Tang and S.T. Chanson [5] introduced optimization techniques for job dispatching and allocation of the workload. Workload allocation is done in such a way that high percentage of jobs is allocated to the more power processors to improve the overall system performance. The main objective of this approach is optimizing the static job scheduling algorithms by focusing on the heterogeneous machines. Jobs are assumed to be independent of each other and they do not communicate with the users.

Static load scheduling and balancing approach involves two components: workload allocation and the job dispatching policy. In workload allocation scheme, the fraction of total amount of load that has to be assigned to the individual processor is being determined. Based on the fractions computed, the task of the job dispatching policy is to decide which jobs have to be assigned to the particular computer in the system. An optimized round robin based job dispatching strategy is proposed. Since higher number of jobs is assigned to the more powerful processors, it may lead to load imbalance condition. Though the static approach is simple and involves little overhead, it is not responsive to the dynamic changes.

B. Dynamic Load Balancing Approach

Ali M. Alakeel [6] highlighted a framework for dynamic load balancing by focusing on the three major components called information, transfer and location policy. In a distributed system, all the nodes are connected via the local area network. Dynamic load balancing takes into consideration the current state of system and allocates the jobs to the node for processing. Various type of information is collected in order to do the balancing and the calculation of load is done based upon the processing speed and the configuration of the node.

In this approach, the arriving jobs are placed in a queue initially or they are assigned to the nodes as they arrive for processing. The random arrival of the jobs is assigned to the primary nodes. Load balancing conditions are checked at each and every primary node to know which node is overloaded or under loaded. Migration of job takes from heavily loaded to lightly loaded node. The major problem in dynamic approach is the communication and transfer overhead since processing nodes in turn communicate with each other to balance the load.

C. Centralised Node Based Load Balancing

Sandip Kumar Goyal, R.B. Patel, Manpreet Sing [7] have briefed how the Dynamic load balancing algorithms can be further improved by having a central node placed in the system. In the simple dynamic approach whenever the processing nodes will become overloaded, it will try to find other lightly loaded nodes in the cluster. If such a lightly loaded processing node is found then only the load is migrated to that. In case such processing node is not available then the load is further migrated to a centralized

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING

National Conference on Advanced Innovation in Engineering and Technology (NCAIET-2015)

Alva's Institute of Engineering and Technology, Moodbidri

Vol. 3, Special Issue 1, April 2015

node. This central node is responsible to take the overload from other primary nodes.

Following Fig.1 shows the existing centralized approach with a central node. Jobs are allocated randomly to the processing nodes and migration takes place. When the processing nodes are overloaded the job will move to the central node for execution.

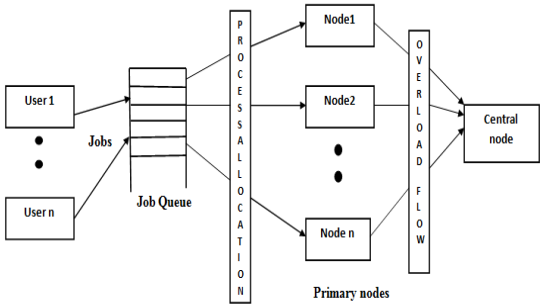


Fig.1. Existing centralized approach for dynamic load balancing

There are various drawbacks involved in this approach. Firstly, the continuous reallocation of workload to the nodes takes place randomly. Load balancing conditions will be checked at each and every processing node which leads to more tasks in balancing. Migration of jobs from one node to other leads to the transfer overhead and the primary node has to maintain the information about every other primary node. Much of time will be spent in transferring the job rather than executing the jobs.

III. SYSTEM MODEL

The proposed model takes into account the problems of migration overhead, checking load balancing conditions at every primary node that occur in existing approach and rectifies it by using a central node that will handle the random load arriving from various users, checks the conditions and other information and then allocates the load to the processing nodes and thereby balancing it.

A. Problem Statement

In the existing centralized load balancing technique, the incoming jobs are randomly assigned to the processing nodes. Load balancing conditions are not checked in prior to allocation of jobs. Once the jobs are allocated, the processing nodes will check whether it can process the job execution request. In fact, if the processing nodes are unable to handle the job then it will migrate jobs from one node to other resulting in the migration overhead. In this way, much overhead is incurred on the processing nodes due to migration thus making load balancing difficult. The proposed model is designed to overcome these problems.

B. Objectives

- To redistribute the workload equally among the set of processing servers.
- Ensure that system spends more time in executing the jobs rather than transferring it.
- Improving the performance of each processing server and thereby improving overall performance.
- To reduce the overhead on the processing servers and response time of the jobs.
- Have the ability to modify itself in accordance with any changes.

.

C. Proposed Work

In this model, the central system is considered as load balancer and scheduler since it performs both scheduling and balancing task. The following Fig.2 shows the modified approach.

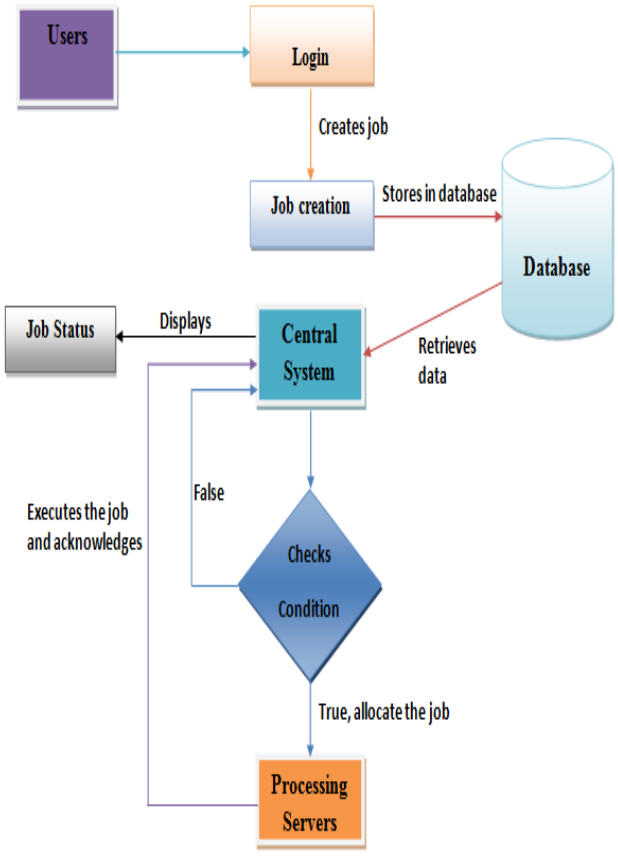


Fig.2. Centralized node based scheduling and balancing approach that adapts itself to the dynamic changes in the system.

In this approach, the processing nodes or servers will have certain capacity. The capacity of every processing server including the processing speed as well as the number of processes running is collected by the central system which is also called as load scheduler and balancer. Various users will login to the system with their respective

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING

National Conference on Advanced Innovation in Engineering and Technology (NCAIET-2015)

Alva's Institute of Engineering and Technology, Moodbidri

Vol. 3, Special Issue 1, April 2015

IP Address to create the jobs having different job length. All the jobs created will be stored in the database for later retrieval by the central system. For every predefined interval, the central system will check for the incoming jobs from users which is placed in the queue and scheduled based on First Come First Served scheduling process.

The load of each incoming job is calculated, compared with the capacity of every processing server and assigned to the smallest possible processing server by the central system. Execution time for each job is randomly assigned based on the length of each job. Once the job is allocated for processing, capacity status of the processing server is updated and the job starts executing. After executing the job, the status of the job is updated at the central system and hence acknowledged to the user.

Here the load is not migrated from one processing server to the other sever, instead the load is assigned by the central system to all processing nodes taking into account the current status of every processing node. The servers considered for processing are heterogeneous in nature having different capacity and processing speed.

D. Proposed Load Balancing Algorithm

Input: Jobs from the users
Output: Executing the jobs and balancing the load
Algorithm: Load Balancing
1. Get the load threshold from each processing server
2. Sort the load threshold in ascending order
3. Return the sorted list
4. for every predefined interval
5. Wait for incoming jobs
6. if any job arrives
8. Calculate the load
9. Compare the load with every processing server threshold capacity
10. if any processing server available
11. Assign job to that processing server
12. update/set the load status at each processing server
13. else
14. Wait until the processing server becomes available
15. Repeat

E. Modules

There are three modules included in this project. They are:

1. User Module

User will register to the system to create the jobs for processing. Each user will have its own username and password to login along with the IP Address of the user. Security credentials are given for user password and the IP Address. User will create the job of certain job length and also gives the job name.

2. Central System

Central system plays a major role in this approach where it has to do both the scheduling and balancing. The jobs that are created by the user will be retrieved for every interval of one second by the central system. A grid view of the incoming jobs from the users is maintained here to know the status of every job. First come first served scheduling policy is used where the jobs are allocated for processing based on its arrival. Job queue is maintained at central system.

Main aim of this module is to check the load balancing conditions by comparing the length of each job with the capacity of every available processing server. Once the conditions are checked the job is assigned to the possible processing server for execution. After execution the status of the job is updated and acknowledged to user. If none of the server is able to do the processing then the jobs will be in queue until the server becomes available.

3. Processing Servers

Processing servers are responsible for executing the jobs that are being allocated by the central system. Every processing will have a server name, server capacity, processing speed and the current capacity. Heterogeneous servers are being considered here. In this project a minimum of three processing servers are used. Execution time of the jobs is not known in prior. Once the job is allocated to the server for execution the capacity at server will be updated based on the job length. Job will be executed for certain random execution time which is calculated at the time of execution based on the job length. After the job execution it will be acknowledged to the central system for updating the status.

IV. IMPLEMENTATION AND RESULTS

Proposed approach is implemented in windows platform with .NET framework using Visual Studio. C Sharp language is used for the front end design and the implementation. Database is maintained to keep the user records. Snapshots are as follows:

A. Registration Page for Users

User registers with the IP address and the password. Get IP function is used to get the system IP address.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING

National Conference on Advanced Innovation in Engineering and Technology (NCAIET-2015)

Alva's Institute of Engineering and Technology, Moodbidri

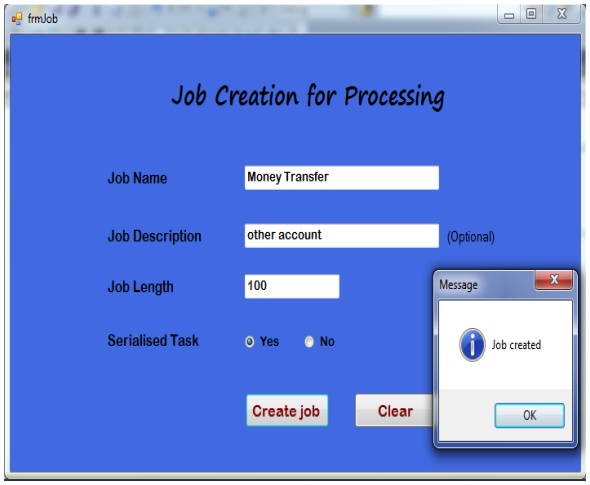
Vol. 3, Special Issue 1, April 2015

B. User Login

User will login with the particular username and password. Once logged in it will be directed to the job creation page.

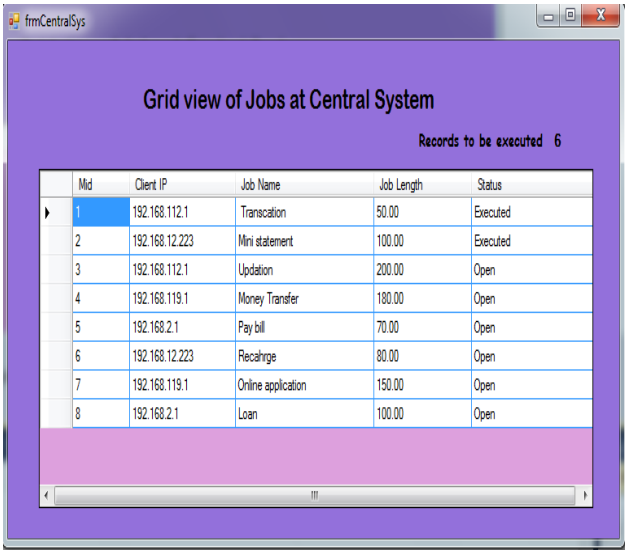
C. Job Creation

User will create the job with particular job length specifying the job name. Usually the jobs are executed in serial fashion. Independent jobs are executed in parallel by splitting the jobs. Execution time for the job will be calculated once the job is allocated for processing.



D. Central System

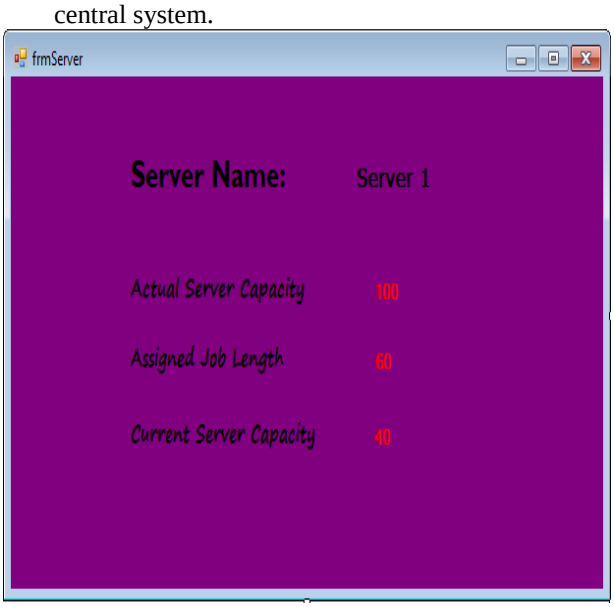
Jobs created are retrieved by central system for allocation. Two timers called pending timer and the job timer are kept for job retrieval and the job execution. The jobs are maintained in queue for execution. Pending jobs are indicated with open status. Based on the Master id the jobs are assigned for execution. Before assigning, the capacity of the server is compared with the length of the job. Following snapshot shows the grid view of the jobs in the central system with the status.



Mid	Client IP	Job Name	Job Length	Status
1	192.168.112.1	Transcation	50.00	Executed
2	192.168.12.223	Mini statement	100.00	Executed
3	192.168.112.1	Updation	200.00	Open
4	192.168.119.1	Money Transfer	180.00	Open
5	192.168.2.1	Pay bill	70.00	Open
6	192.168.12.223	Recahrge	80.00	Open
7	192.168.119.1	Online application	150.00	Open
8	192.168.2.1	Loan	100.00	Open

E. Processing Servers

Processing server includes the actual capacity of the server, job length allocated, current capacity and the server name. Various servers are used but the image of one server is given below. Once a job is allocated the capacity of the server will be updated. After execution of each job it will acknowledge the central system.



V. CONCLUSION

Load balancing has become an area of interest among many researchers. Several load balancing schemes are highlighted here. Implementation of dynamic load balancing algorithms is complex, but the benefits obtained are more compared to static. A centralized node based scheduling and balancing approach is proposed that considers the problems of existing centralized approach and distributes the workload as equally as possible.

A central system of this model is responsible for both the scheduling as well as balancing of load. Migration of

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING

National Conference on Advanced Innovation in Engineering and Technology (NCAIET-2015)

Alva's Institute of Engineering and Technology, Moodbidri

Vol. 3, Special Issue 1, April 2015

load from one node to other is not performed instead the central node will assign the load to all the processing nodes considering the current status and thereby balances it. The major objective is to overcome the job migration overhead and ensure that the system spends more time in executing jobs rather than transferring it. This model considers the dynamic changes in the system, adapts itself in accordance with the changes and accordingly takes the load balancing decisions.

Implementation of the model is done for varied number of users and the processing servers. The analysis of the results and the performance of this model compared to the previous approaches will be done along with the graphs.

REFERENCES

[1] Parveen Jain, Daya Gupta, "An Algorithm for Dynamic Load Balancing in Distributed Systems with Multiple Supporting Nodes by Exploiting the Interrupt Service", International Journal of Recent Trends in Engineering, Vol 1, No. 1, May 2009.

[2] UrjashreePatil, RajashreeShedge, "Improved Hybrid Dynamic Load Balancing Algorithm for Distributed Environment", International Journal of Scientific and Research Publications, Volume 3, Issue 3, March 2013.

[3] Priyanka Gonnade1, Sonali Bodkhe, "An Efficient load balancing using Genetic algorithm in Hierarchical structured distributed system" International Journal of Advanced Computer Research Volume-2 Number-4 Issue-6 December-2012.

[4] Misra, M. et al. (2007), "On the Design of Adaptive and Decentralized Load-Balancing Algorithms with Load Estimation for Computational Grid Environments", Institute of Electrical and Electronics Engineer Transaction On Parallel And Distributed System,18(12), pp. 1675-1686.

[5] X. Tang and S.T. Chanson, "Optimizing Static Job Scheduling in a Network of Heterogeneous Computers," Proc. of the Intl. Conf. on Parallel Processing, pp. 373-382, August 2000.

[6] Ali M Alakeel(2010), "A guide to Dynamic Load Balancing in Distributed Computer system", IJCSNS (International Journal of Computer Science and Network Security), VOL.10 No.6, June 2010.

[7] Sandip Kumar Goyal, R.B. Patel, Manpreet Sing,"Adaptive and Dynamic Load Balancing Methodologies For Distributed Environment: A Review", International Journal of Engineering Science and Technology , Vol.3 No. 3,2011.

[8] Bahi J.M., Vivier C., and Couturier R.,"DynamicLoadBalancing and Efficient LoadEstimators for Asynchronous IterativeAlgorithms", IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 4, Apr. 2005.

[9] Sunita Bansal, Divya Gupta, and Chittaranjan Hota,"Adaptive Decentralized Load Sharing Algorithms with Multiple Job Transfers In Distributed Computing Environments", International Journal of Recent Trends in Engineering, Vol 2, No. 2, November 2009.

[10] Arora, M. et al. (2002),"A De-Centralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environments", Proceedings of International Conference on Parallel Processing Workshops (ICPPW '02), *pp. 499-505*.