



A Sender-Initiated Load Balancing Approach with Genetic Algorithm

Kulvinder Singh¹, Chetna Kukreja²

University Institute of Engineering and Technology
Kurukshetra University Kurukshetra, Haryana, INDIA.

E-mail (s): kshanda@rediffmail.com and chetna_kukreja2000@yahoo.com

Abstract: Load balancing is a technique in which workload is distributed across multiple computers or other resources to get optimal resource utilization, minimum time delay, maximize throughput and avoid overload. Mainly in operation of parallel and distributed computing systems, load balancing problem arises. Load balancing scheme can be characterized as: static and dynamic load balancing. This paper represents a sender-initiated load balancing scheme, which is a part of dynamic load balancing scheme, with the help of genetic algorithm (GA), as due to easily adaptable search technique and robust, immense popularity have gained by genetic algorithms.

Keywords: sender-initiated, load balancing, genetic algorithm

I. Introduction

Load balancing can be defined as distributing the available workload among the available machines as equally as possible in such a way that the workload imbalance performance in the system can be minimized [1]. Load balancing is a technique in which workload is distributed across multiple computers or other resources to get optimal resource utilization, minimum time delay, maximize throughput and avoid overload. Objective of load balancing is to improve systems performance by keeping processor's load even.

S. Petrovic and C. Fayad [2] deals with the load balancing of machine in a real world job shop scheduling, with identical machines with the help of genetic algorithm. Fogarty *et al.* [3] described a model to balance the load of presses in the station, to maximize the percentage of dry substances in the pressed pulp they produce, with the help of genetic algorithm which optimize an on-line computer model of a sugar beet pressing station. Choi *et al.* [4] proposed a new method to solve the distribution system loss minimization reconfiguration problem which applies a genetic algorithm for determining which sectionalizing switch to operate. A. Y. Zomaya and Y. H. Teh [5] presented work on how a genetic algorithm can be employed to solve the dynamic load balancing problem.

Generally characterization of load balancing algorithms occurs as: static and dynamic.

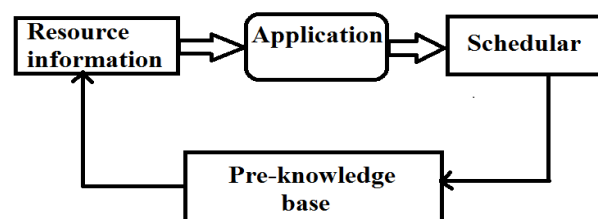


Fig. 1.1 static load balancing [6]

In static load balancing scheme, allocation of all tasks to processors occurs using priori information of the task before their execution. The main advantage of adopting static load balancing scheme is its simplicity in terms of both implementation as well as overhead. Task of parallel program to workstations is allocated either based on the load at the time nodes are allocated to some task, or on the basis of average load of our workstation cluster, in static load balancing algorithm. At compile time, decisions are made when resource requirements are estimated about the load balancing. Static load balancing does not require constant monitoring of workstation for performance statistics.

There are several flows in static load balancing algorithms:

- Number of steps needed to get the result is not known in advance for some problems.
- Communication delays also exists sometimes which vary in an uncountable way.
- In an accurate way estimation of a-priori the execution time of various parts of a program is very difficult.

Dynamic load balancing algorithm

Dynamic load balancing algorithms offer the possibility of improving load distribution at the expense of additional communication and computation overheads [7].

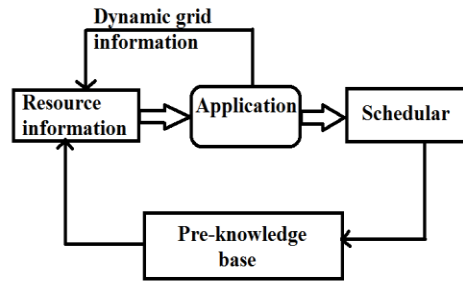


Fig. 1.2 dynamic load balancing algorithm [6]

In dynamic load balancing scheme, tasks are transferred during execution from the heavily loaded processor to lightly loaded one. In dynamic load balancing, at run time the changes are made to the distribution of work which uses the information about load that is current or recent for making distribution decisions.

There are different kinds of strategies used in dynamic load balancing such as: sender-initiated, receiver-initiated, global, local, centralized, decentralized, co-operative, non-co-operative, adaptive, non-adaptive, one time assignment and dynamic reassignment strategy. In this paper, for determining destination processor, sender-initiated strategy is used.

II. Algorithm Design

In sender-initiated algorithms, sender having high load tries to transfer task to another processor (receiver) which have low load i.e. the node which is congested tries to move some workload to the node which is not congested. In this scheme, mainly five types of messages are used: request, accept, reject, migrate, and terminate. In this paper genetic algorithm is used to find the destination processor. The heavily loaded processor (sender) issues a request message for task migration to another processor selected by genetic algorithm. The selected processor by viewing its waiting queue sends a accept message if it is lightly loaded otherwise returns a reject message. If a reject message is returned then heavily loaded processor send request message to another processors until an accept message is returned. If no accept message is returned, no task migration takes place. Task migration decision is made in each processor independently, in sender-initiated algorithm. If accept message is returned then task is migrated to that processor which sent accept message and at the end process of migrating task is ended with the terminate message.

Task distributor, which takes task from waiting queue, is the one which decide whether the task should be migrated or not and which is the destination processor to which the task should be sent off. Before execution, there is a chance for every task to migrate. Depending upon the length of a task queue in a processor, three-level load system (Light, Normal, Heavy) is employed. For example,

Load-state	Criteria
Light load	$CQL \leq 2$
Normal load	$2 < CQL \leq 4$
Heavy load	$CQL > 4$

Task distributor tries to migrate the task to another processor if a processor's load is heavy. In proposed algorithm, demand driven scheme is employed for migrated a task. When on a processor migration procedure starts, request messages are sends by task distributor of the processor. And load information is send back to that processor by the receivers.

Each processor contains its own population onto which selection, crossover and mutation, genetic operations are applied. Binary coded vector is used to define a string which represents a set of processors to which the requests are sent off. If migration request is sent to a processor then its vector value equals to 1 otherwise 0. Payoff values and fitness value is also associated with each string.

III. Fitness Evaluation

Fitness value can be evaluated of a string by averaging last T pay off values, each of which is given at every migration. A string is provided with positive payoff it at least one of migration request is accepted otherwise value of payoff is zero [8].

$$\text{Payoff} = \begin{cases} 0 & \text{if all of the requests are rejected} \\ g(m) & \text{otherwise,} \end{cases}$$

Here $g(m)$ represents the monotonic decreasing function which is greater than zero for all $x > 0$ and $m (\leq n-1)$ is the total number of requests issued according to a string. That means request dispatched is inversely proportional to the payoff value. If the fitness values of all are 0, then a string is randomly selected. In every population, duplication of string is not permitted because it leads to inconsistent fitness values as it may happen that same string may have different fitness value. Probability at which genetic operations such as crossover and mutation are executed, are defined in advance. From parent string, the initial fitness value of an offspring is inherited.

S. Lee and C. Hwang [9] given the formula for evaluating Fitness value:

$$F = \frac{1}{\alpha \times TMP + \beta \times TMT + \lambda \times TTP}$$

Where TMP is the Total Message Processing Time, which is the summation of processing times for request messages to be transferred; TMT is Total Message Transfer Time, which is the summation of each message transfer times (EMTT) from the sender to processors corresponding to the bits set '1' in selected string; and TTP is Total Task Processing Time which is the summation of the times needed to perform a task at each processor corresponding to bits set '1' in selected string; and α , β and λ are weights of parameters TMP, TMT and TTP respectively.

IV. Conclusion

A dynamic load balancing algorithm is presented which tries to find a receiver which is willing to receive an additional task, with the help of genetic algorithm. Genetic operations are used to decide to which processors the request messages to be sent off.

V. References

- [1] S. Rajakumar, V. P. Arunachalam, and V. Selladurai, "Workflow balancing strategies in parallel machine scheduling", International journal of advance manufacturing technology, vol. 23, 2004, , pp. 366-374.
- [2] S. Petrovic and C. Fayad, "A Genetic Algorithm for Job Shop Scheduling with Load Balancing", Springer, 2005.
- [3] T. C. Fogarty, F. Vavak and P. Cheng, "System Identification for Load Balancing with the Genetic Algorithm", IEE, March 1994, pp. 750-753.
- [4] D. S. Choi, C. S. Kim and J. Hasegawa, "An Application of Genetic Algorithms to the Network Reconfiguration for Loss Minimization and Load Balancing Problem(Part 2)", IEEE, vol. 1, 1995, pp. 376-381, doi:10.1109/EMPD.1995.500756.
- [5] A. Y. Zomaya and Y. H. teh, "Observations on Using Geneic Algorithms for Dynamic Load Balancing", IEEE, vol. 12, September 2001, pp. 899-911.
- [6] Hood R. and Jimenez J. B., "An Active objects load balancing mechanism for intranet", CiteSeerx, doi:10.1.1.89.6324.
- [7] Z. Zeng and B. veeravalli, "Rate based and queue based dynamic load balancing algorithm in distributed systems", IEEE, 2004, pp. 349-356.
- [8] M. Munetomo, Y. Takai and Y. Sato, "A Genetic Approach to Dynamic Load Balancing in a Distributed Computing System", IEEE, vol. 1, 1994, pp. 418-421, doi:10.1109/ICEC.1994.349914.
- [9] S. Lee and C. Hwang, "A dynamic load balancing approach using genetic algorithm in distributed systems", IEEE, 1998, pp. 639-644.