A Note on "Optimal Static Load Balancing in Distributed Computer Systems"

Sakib A. Mondal[?]

Infosys Technologies Limited

Abstract: The problem of minimizing mean response time of generic jobs submitted to a heterogenous distributed computer systems is considered in this paper. A static load balancing strategy, in which decision of redistribution of loads does not depend on the state of the system, is used for this purpose. The article is closely related to a previous article on the same topic. The present article points out number of inconsistencies in the previous article, provides a new formulation, and discusses the impact of new findings, based on the improved formulation, on the results of the previous article.

[?] Address for Correspondence: Infosys Technologies Limited,

27, Bannerghatta Road, JP Nagar 3rd Phase,

Bangalore 560 076, India Fax 91-080-6587967

email: abdulsakib@inf.com, sakib@hotmail.com

1 Introduction

Recent years have been witness to an increasing use of distributed computing system. This may be attributed to two main factors: growth of the Internet, and low cost solution of end-user computing devices. Many business processes such as supply chain management are distributed due to the inherent nature of tasks involved with them. Besides, scale of economy is often possible due to the use of clusters of less powerful computers instead of a central computer of significantly high power. However, a distributed solution can yield the true advantage only if it is possible to distribute works evenly among the processors (terms processor, host and computer are used inter-changeably in this article). In other words, when load on the computers in a distributed environment has significant variance of workloads, high performance can be achieved by redistributing loads. The task of redistributing the loads on the computers is called load balancing.

Load can be characterized as jobs or tasks. A job indicates a complete and independent entity of work to be completed. However, a job may consist of number of sub-units called tasks, and for the job to be completed the comprising tasks need to often communicate among themselves. For simplification, in this article, we consider mutually independent tasks units as jobs. Jobs can be classified into two broad categories – dedicated and generic. Dedicated jobs can be processed only on specified hosts, while generic jobs can be processed on any host in the system. Once a generic job is submitted to the system, it can be processed at the place of its origin or can be transferred it to another processor. Irrespective of the processor chosen, once the job is started at a processor, it remains there till completion. Some of the most commonly used performance indicators of a load balancing algorithm are mean response time, throughput, variance of response time. In this article, we confine our analysis based on mean response time only.

Algorithms or heuristics for load balancing fall into two categories: static and dynamic. Static load balancing techniques do not depend on the state (characterized by workload etc.) of the processors. Simple static load balancing techniques such as join-the-shortest queue (SQ), minimum expected delay (MED) have been in use for over a decade. The SQ policy allocates an arriving task to the processor having the minimum load and, the MED policy allocates an arriving task to the processor having the minimum expected value of waiting time for currently scheduled tasks. Note that though these two policies may work similarly for homogenous processors, their behaviors differ for heterogeneous processors. Tantawi and Towsley [5] have proposed an optimization model and load balancing algorithms to determine static optimal allocation of loads among the hosts connected in a broadcast network. Ross et al. [4] considered a more general problem consisting of dedicated and generic jobs, and also dealt with scheduling decision at each host. The authors have noted that the problem is separable over local scheduling decisions, and suggested a solution procedure based on this finding.

Dynamic load balancing techniques, on the other hand, rely on present and past states of the processors. In dynamic load balancing, though an initial assignment of work to the processors is done through static load balancing, subsequent adjustment of work depends on the workload profile of the processors. Dynamic load balancing is expected to have higher overhead, but better performance than static load balancing. Most of the dynamic load balancing algorithms have two important components: Information exchange through which processors exchange their load information, and subsequent load exchange to redistribute loads. Depending on the characteristics of these components, there can be different types of dynamic load balancing ([1] [2], [3], [6]).

However, in this article we concentrate on static load balancing. We express some of the points of our disagreement with [5] and provide alternatives. The rest of the paper is organized as follows. Section 2 suggests an improved problem formulation. The solution procedure is illustrated in Section 3. Concluding remarks are provided in Section 4. Points of disagreement are marked in italics.

2 Improved Problem Formulation

We adopt the model of distributed computing as proposed in [5]. In this model host computers are treated as node. Each node $i \in \{1, 2, \dots, n\}$ is capable of executing any incoming job k. Arrival rate of job at node i is ${}_i$, and total arrival rate of jobs on the system is $= \sum_{i=1}^{n} {}_i$. When a job k arrive at a node i, it may be executed at node i itself or transferred to another node j, depending on the distribution of loads at that time. If job is transferred to node j, it is executed at node j without being transferred to any other host, and results of execution is transferred back to node i so that the user submitting the job gets the illusion of the job being executed at the host i. Processing rate of job at node i is ${}_i$ and transfer rate of jobs from node i to node j is x_{ij}. Therefore, $= \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij}$ is the total traffic on the network.

It is also assumed that hosts may be heterogeneous (different CPU speed, memory and swap availability etc.) and speed of execution depends on the instantaneous load of the host. Therefore the instantaneous marginal delay due to local processing of a job k at host i depends on host i as well as load on the host i, and is denoted by $F_i(\ {}_i)$. So the mean response time of all the jobs on the network due to local processing is given by $\sum_{i=1}^{n} \frac{{}_i}{\sum_{i=1}^{n} {}_i} F_i(\ {}_i) = \sum_{i=1}^{n} \frac{{}_i}{} F_i(\ {}_i)$.

In the distributed set up, another major component of mean response time is the network delay. The incremental communication delay for transfer of a packet can be modeled as $G_{ij}(x)$, where x is the vector of load transfer. Note that we are assuming that the network delay due to sending the response back is negligible. In case it is not so, the argument of G should be suitably replaced. [?] So the contribution of network delay to mean response time is given by $\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{x_{ij}}{\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}}G_{ij}(x)$.

[It is to be noted that in [5] as well as in [4], denominator used was which is not equal to $\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}$. Hence, their formulations did not use average communication delay, contrary to their claims. Also the claim in [5] that x in G(x) can include response packets is wrong as in that case total ow no longer remains .]

Hence the problem of load balancing can be formulated as:

Minimize

$$\sum_{i=1}^{n}\frac{\ _i}{\sum_{i=1}^{n}\ _i}F_i(\ _i)+\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{x_{ij}}{\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}}G_{ij}(x)$$

subject to

$$\sum_{i=1}^{n}\ _i=\sum_{i=1}^{n}\ _i;$$

$$\ _i+\sum_{j=1}^{n}x_{ij}=\ _i+\sum_{j=1}^{n}x_{ji};\ i=1;2;\ \ ;n$$

$$=\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij};$$

$$\ _i\ \ 0;\ i=1;2;\ \ ;n;$$

[?] If we assume that y_{ij} is the amount of ow of results from node i to node j, then a more accurate model would have network delay given by: $\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{x_{ij}+y_{ji}}{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ij}+y_{ji})}G_{ij}(x+y)$, where y is the vector of additional tra c due to response packets.

$$x_{ij} \geq 0; \; i,j = 1,2,\cdots,n;$$

$$x_{ii} = 0; \; i = 1,2,\cdots,n;$$

$$\tag{1}$$

Note that first two constraints ensure that all incoming load is processed, and flow balance holds true. As in [5], we assume that $F_i(\beta_i)$ are increasing convex functions and $G_{ij}(x)$ is a non-decreasing convex function. Ethernet and satellite are quite common on network infrastructure. Due to the type of data transmission means on these, the network delay for transfer of a packet from node i to node j depends on the total network load. Henceforth, we also assume that $G_{ij}(x)$ depends on the total traffic λ on the network but not on any component of it. So the second term in the objective function can be written as $\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{x_{ij}}{\Phi}G_{ij}(\lambda)$.

3 Solution of the Problem

Solution to the above problem can be expressed in terms of disjoint set of nodes. We classify the nodes as source (idle or active), neutral and sinks as in [5]. A node i is said to be source if $x_{ji} = 0 \; \forall j$ and $\exists j \; x_{ij} > 0$. A source i is idle if $\beta_i = 0$, else it is active. A node i is said to be sink if $x_{ij} = 0 \; \forall j$ and $\exists j \; x_{ji} > 0$. For a neutral node i, $x_{ij} = 0 \; \forall j$ and $x_{ji} = 0 \; \forall j$. We denote the set of sinks, idle sources, active sources and neutrals by $S; R_d; R_a$ and N

respectively.

Tantawi and Towsley [5] claimed that triangular inequality of network delays (i.e., $G_{ij}(\ ) \quad G_{ik}(\ ) + G_{kj}(\ )$) was sufficient condition for solution of the problem as formulated by them to yield only source, sink and neutral nodes. We illustrate in the following theorem that the claim is wrong.

Theorem 1 The triangular inequality of network delay and non-decreasing property of $\frac{G(\ )}{}$ are sufficient conditions for Problem 1 to yield nodes which are either sources, sinks or neutrals.

Proof: For a given incoming load is constant, and hence we just need to show that the stated conditions are sufficient for a flow rate assignment x with nodes either sinks, sources or neutral nodes to minimize the communication term $C = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{x_{ij}}{} G_{ij}(\ )$. Let us suppose that $i(x)$ denotes the number of nodes that are neither sinks, sources nor neutrals for the flow rate assignment x. Therefore we need to show that $i(x) = 0$. Of all the flow rate assignment minimizing communication delay, choose the flow rate assignment x which has maximum $i(x)$ and it is strictly positive. [In Proof of Theorem 1 in [5], x was chosen to have minimum $i(x)$ from candidate solutions minimizing communication delay. And it was proved that $i(x) = 0$. But this does not rule out the possibility that there can be other optimal solutions with $i(x) > 0$. So it only proves that there is at least one flow rate assignment minimizing the objective function for which nodes are either sources, sinks or neutrals.] Since $i(x) > 0$, there is at least one node k and a pair l and m such that $x_{lk} > 0$ and $x_{km} > 0$. Consider a new assignment x' which di ers from x only in the following.

$$x'_{lk} = x_{lk} \quad \min\{x_{lk}; x_{km}\};$$

$$x'_{km} = x_{km} \quad \min\{x_{lk}; x_{km}\};$$

$$x'_{lm} = x_{lm} + \min\{x_{lk}; x_{km}\}:$$

Let the new Communication delay be C'. So,

$$C' = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{x'_{ij}}{'} G_{ij}(') =$$

$$x'_{lk}\frac{G_{lk}(')}{'} + x'_{km}\frac{G_{km}(')}{'} + x'_{lm}\frac{G_{lm}(')}{'} + (\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}\frac{G_{ij}(')}{'} \mid (i;j) \in \{(l;k);(k;m);(l;m)\}):$$

Since $'$ and $\frac{G()}{}$ is non-decreasing,

$$C' \quad C \quad \{x'_{lk}\frac{G_{lk}(')}{'} \quad x_{lk}\frac{G_{lk}()}{}\} + \{x'_{km}\frac{G_{km}(')}{'} \quad x_{km}\frac{G_{km}()}{}\}$$

$$+ \{x'_{lm}\frac{G_{lm}(')}{'} \quad x_{lm}\frac{G_{lm}()}{}\} \quad (x'_{lk} \quad x_{lk})\frac{G_{lk}()}{}$$

$$+ (x'_{km} \quad x_{km})\frac{G_{km}()}{} + (x'_{lm} \quad x_{lm})\frac{G_{lm}()}{}$$

$$= \frac{1}{}[\min\{x_{lk}; x_{km}\}(G_{lk}() + G_{km}() \quad G_{lm}())] \quad 0:$$

Therefore $i(x') < i(x)$, a contradiction.

[in Proof of Theorem 1 in [5], with new x' total network load no longer remains . Also it is possible to have $i(x') = i(x) \quad 2$, and not always $i(x') = i(x) \quad 1$).] □

As for broadcast network, delay can be assumed to be independent of source-destination pair, $G_{ij}(\ )$ is independent of i and j. Moreover, since $\sum_{i=1}^{n} {}_i = \sum_{i=1}^{n} {}_i$ and total incoming load is constant, the objective function in Problem 1 is equivalent to

$$\sum_{i=1}^{n} {}_i F_i({}_i) + \sum_{i=1}^{n} {}_i G(\ ):$$

For simplifying the solution procedure, let $u_i = \sum_{j=1}^{n} x_{ji}; v_i = \sum_{j=1}^{n} x_{ij}$. So Problem 1 can be written as:

Minimize

$$\sum_{i=1}^{n} {}_i F_i({}_i) + \sum_{i=1}^{n} {}_i G(\ \sum_{i=1}^{n} \ v_i)$$

subject to

$$
\begin{aligned}
{}_i + v_i = {}_i + u_i; \ i = 1;2; \quad ;n; \\
\sum_{i=1}^{n} v_i \quad \sum_{i=1}^{n} u_i = 0; \ i = 1;2; \quad ;n \\
{}_i \quad 0; \ i = 1;2; \quad ;n; \\
u_i \quad 0; \ i = 1;2; \quad ;n; \\
v_i \quad 0; \ i = 1;2; \quad ;n:
\end{aligned}
\qquad (2)
$$

Note that $\sum_{i=1}^{n} {}_i = \sum_{i=1}^{n} {}_i = \ $ is constant. After eliminating the rst constraint, problem 2 has the following form.

Minimize

$$\sum_{i=1}^{n}(u_i \quad v_i + \quad _i)F_i(u_i \quad v_i + \quad _i) + \quad G(\sum_{i=1}^{n} \quad v_i)$$

subject to

$$\sum_{i=1}^{n} v_i \quad \sum_{i=1}^{n} u_i = 0; i = 1;2; \qquad ;n$$

$$u_i \quad v_i + \quad _i \quad 0; i = 1;2; \qquad ;n;$$

$$u_i \quad 0; i = 1;2; \qquad ;n;$$

$$v_i \quad 0; i = 1;2; \qquad ;n:$$

(3)

Theorem 2 The optimal solution to the above problem can be expressed as:

$$f_i(\quad _i) \qquad + \quad G^0(\quad); \quad _i = 0 \ (i \in R_d);$$

$$f_i(\quad _i) = \quad + \quad G^0(\quad); 0 < \quad _i < \quad _i \ (i \in R_a);$$

$$f_i(\quad _i) \qquad + \quad G^0(\quad); \quad _i = \quad _i \ (i \in N);$$

$$f_i(\quad _i) = \quad ; \quad _i > \quad _i \ (i \in S);$$

subject to total flow constraint,

$$\sum_{i \in R_a} f_i^{\ 1}(\quad + \quad G^0(\quad)) + \sum_{i \in N} \quad _i + \sum_{i \in R_a} f_i^{\ 1}(\quad) = \quad ;$$

where

$$\alpha = \sum_{i \in S} (f_i^{-1}(\lambda) - \theta_i).$$

Proof: The Lagrangean function for this problem is

$$L(u, v; \lambda, \gamma, \mu, \nu) = \sum_{i=1}^{n} (u_i - v_i + \theta_i) F_i(u_i - v_i + \theta_i) + G(\sum_{i=1}^{n} v_i)$$
$$+ \lambda(\sum_{i=1}^{n} v_i - \sum_{i=1}^{n} u_i) + \sum_{i=1}^{n} \gamma_i (u_i - v_i + \theta_i) + \sum_{i=1}^{n} \mu_i u_i + \sum_{i=1}^{n} \nu_i v_i.$$

It can be shown easily that objective function and the constraints satisfy conditions for applying Karush-Kuhn-Tucker (KKT) conditions. Let $f_i(\theta_i) = F_i(\theta_i) + \theta_i \frac{\partial F_i(\theta_i)}{\partial \theta_i}$. Applying KKT conditions, we get the following set of equations.

$$\frac{\partial L}{\partial u_i} = f_i(u_i - v_i + \theta_i) - \lambda + \gamma_i + \mu_i = 0, \; i = 1, 2, \ldots, n, \tag{4}$$

$$\frac{\partial L}{\partial v_i} = -f_i(u_i - v_i + \theta_i) + G'(\sum_{i=1}^{n} v_i) + \lambda - \gamma_i + \nu_i = 0, \; i = 1, 2, \ldots, n, \tag{5}$$

$$\frac{\partial L}{\partial \lambda} = -\sum_{i=1}^{n} u_i + \sum_{i=1}^{n} v_i = 0, \tag{6}$$

$$u_i - v_i + \theta_i \geq 0, \; \gamma_i(u_i - v_i + \theta_i) = 0, \; \gamma_i \geq 0, \; i = 1, 2, \ldots, n, \tag{7}$$

$$u_i \geq 0, \; \mu_i u_i = 0, \; \mu_i \geq 0, \; i = 1, 2, \ldots, n, \tag{8}$$

$$v_i \geq 0, \; \nu_i v_i = 0, \; \nu_i \geq 0, \; i = 1, 2, \ldots, n. \tag{9}$$

Let us consider two cases separately, $u_i - v_i + \theta_i = 0$ and $u_i - v_i + \theta_i > 0$.

Case 1: $u_i - v_i + \theta_i = 0$, i.e., $\xi_i = 0$.

Again we consider two sub cases:

1A : $_i > 0$.

In this case $v_i > 0$, and consequently from Equation 9 $_i = 0$. So from Equation 5

$$f_i(\ _i) \quad + \ G^0(\ ):$$

Also for this case, $_i = 0;\ _i > 0$. It can easily be noticed that these are idle source nodes.

1B : $_i = 0$.

It is obvious that in this case $u_i = v_i$. From Equation 4 and Equation 5,

$$G^0(\ ) + \ _i + \ _i = 0:$$

But by assumption $G^0(\ ) > 0$ and from Equation 7 and Equation 9 $_i$ $0;\ _i \quad 0$. This implies atleast one of $_i$ and $_i$ is strictly negative, and correspondingly from Equation 7 or Equation 9 either u_i or v_i is zero. Therefore $u_i = v_i = 0$. Hence, from Equation 5 $f_i(\ _i) \quad$: Also $_i = 0;\ _i = 0$.

These nodes correspond to neutral nodes without external load.

Case 2: $u_i \quad v_i + \ _i > 0$.

From Equation 7, $_i = 0$. Now consider three subcases:

$2A: v_i > 0; u_i = 0.$

Since $v_i > 0$, from Equation 9 $\ _i = 0$. Hence from Equation 5, we get

$$f_i(\ _i) = \ \ + \ \ G^0(\ ): \tag{10}$$

For this case, $0 < \ _i < \ _i$. Therefore these nodes correspond to active sources.

$2B: v_i = 0; u_i > 0.$

Since $u_i > 0$, from Equation 8 $\ _i = 0$. Hence from Equation 4,

$$f_i(\ _i) = \ :$$

In this case $\ _i > \ _i$. These are sink nodes.

$2C.\ u_i = v_i = 0.$

Since $\ _i \quad 0$, from Equation 4 we get $f_i(\ _i) \quad$. Similarly, since $\ _i \quad 0$, from Equation 5 we get $f_i(\ _i) \quad\ + \ \ G^0(\ )$. Here $\ _i = \ _i$, and hence nodes are neutrals.

To solve the system completely, we need to nd out value of . From Equation 6, $\sum_{i=1}^{n} \ _i = \ $. Substituting values of $\ _i$ from the solution nodes in the expression we get $\sum_{i2R_a} f_i^{\ 1}(\ + \ \ G^0(\ )) + \sum_{i2N} \ _i + \sum_{i2R_a} f_i^{\ 1}(\ ) = \ $. We still need value of to solve the above equation to nd out for the incumbent solution. Fortunately $\ = \sum_{i=1}^{n} u_i = \sum_{i2S} u_i = \sum_{i2S}(\ _i + v_i \quad _i) = \sum_{i2S}(\ _i \quad _i) = \sum_{i2S}(f_i^{\ 1}(\ ) \quad _i)$. Now we are at a position to nd out from the above equation.

[There are some mistakes in the proof of corresponding theorem in [5] which we presume to be typographical error.] 2

The result of the above theorem can be intuitively explained as illustrated below. All the sink nodes have the lowest and equal incremental node delay( ). Loads are shared among the sink nodes such that all have equal incremental node delay. For all the active source nodes the incremental node delay equals sum of incremental node delay of sink nodes and total incremental communication delay at the present load ( ). If the incremental node delay at a source is greater than this value then it is pro table, from the perspective of the objective to reduce mean response time, to transfer all the load of the source to the sink nodes, and these nodes correspond to idle source nodes. Note that load distribution among the active source nodes are such that all of them have equal incremental node delay(+ G^0( )), and these nodes process portion of their incoming load ($_i$) transferring the remaining loads to sinks. However, if at a node the incremental node delay is less than the above value, then it is not pro table to send any portion of the load to sink node and thereby incur communication delay. All the incoming load at such a node is processed by itself, and it is a neutral node. The above solution is soothing to practitioners' eyes also. The optimal solution suggests a distribution such that nodes with lower incremental node delay are utilized more extensively than others.

4 Conclusion

This article points out a number of inconsistencies in [5], and improves on them. One of the nice aspects of the results in the present article is that the parametric analysis as well as single-point load balancing algorithm as suggested in [5] can be applied with suitable modification of expressions from the solution given in the previous theorem. It is also perceivable that due to the simplicity of the expression, the present procedures can not be more difficult than those in [?].

Fortunately, the present finding does not affect the results of [4]. As mentioned before, Ross et al. had considered a general problem consisting of generic as well as dedicated jobs, and included the task of scheduling jobs at each host. The authors showed that problem was separable over scheduling decisions. They also showed that given an allocation of the jobs on the hosts, the task of scheduling can be solved as a polymatroid optimization problem. The present results in this paper only changes the allocation of the jobs on the hosts and does not violate any of the assumptions made in [4].

As noted in [5], dynamic load balancing can yield better response time. Most of the dynamic load balancing tool uses a decision making unit (called load balancer) which periodically monitors load on hosts. Also hosts can decide to inform of any exigency when load on them exceed a high level threshold or goes below a low level threshold (both these thresholds can be dynamic).

Load balancer then decides on new load distribution and executes the transfer. The present result can be used for dynamic load balancing. The expected values of and + $G^{0}(\ )$ can serve as lower and higher thresholds. The load balancer can keep track of incremental node delay (a function of allocated load on the node. See [3] for some illustrative measures of load) of hosts as well as incremental communication delay. When it receives any receiver-initiated request for a load, generated by load on the host going below the lower threshold), it can invoke single-point algorithm to reallocate the load. Similarly, it can redistribute load when it receives any sender-initiated request, generated by load on the host going above the upper threshold.

References

[1] T. Decker, M. Fischer, R. Luuling and Stefan Tschoke, "A Distributed Load Balancing Algorithm for Heterogeneous Parallel Computing Systems", Proceeding of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications, 933-940 (1998).

[2] G. Lee, "Using System State Information for Adaptive State Polling Policy in Distributed Load Balancing", Proceeding of the 2nd AIZU International Symposium on Parallel Algorithms/Architecture Synthesis, 156-162 (1997).

[3] S. A. Mondal, "Load balancing in Unix and Win32", Dr. Dobb's Journal, forthcoming, 25(7)(2000).

[4] K.W. Ross and D.D. Yao, "Optimal Load Balancing and Scheduling in a Distributed Computer System", Journal of the ACM, 38(3), 676-690 (1991).

[5] A. A. Tantawi and D. Towsley, "Optimal Static Load Balancing in distributed Computer Systems," Journal of the ACM, 32(2), 445-465(1985).

[6] M. J. Zaki, W. Li and S. Parthasarathy, "Customized Dynamic Load Balancing for a Network of Workstations", Journal of Parallel and Distributed Computing, special issue on workstation, clusters and network-based computing (performance, scheduling, and fault-tolerance), 43(2), 156-162 (1997).