REFERENCES

[1] L. M. Ni, K. Y. Wong, D. T. Lee, and R. K. Poon, "A microprocessor-based office image processing system," *IEEE Trans. Comput.*, vol. C-31, pp. 1017–22, Oct. 1982.

An Algorithm for Optimal Static Load Balancing in Distributed Computer Systems

Chonggun Kim and Hisao Kameda

***Abstract*—This paper proposes a load balancing algorithm that determines the optimal load for each host so as to minimize the overall mean job response time in a distributed computer system that consists of heterogeneous hosts. The algorithm is a simplified and easily understandable version of the single-point algorithm originally presented by Tantawi and Towsley.**

***Index Terms*—Distributed computer systems, local area networks, optimal load, optimal static load balancing, single-point algorithm, star network configurations.**

I. INTRODUCTION

Tantawi and Towsley studied a model of a distributed computer system that consists of a set of heterogeneous host computers connected by a communications network [5]. They considered an optimal static load balancing strategy which determines the optimal load at each host so as to minimize the mean job response time. A key assumption of theirs was that the communication delay does not depend on the source–destination pair. This assumption may apply to single channel networks such as satellite networks and some LAN's. Given this assumption, they determined the requirement that the optimal load at each host satisfies, and derived an algorithm that determines the optimal load at each host for given system parameters. It is this algorithm that they call a single-point algorithm.

The Tantawi and Towsley single-point algorithm [5] is surprising in the sense that it does not calculate the load at each node iteratively. Note that previous algorithms on related models such as flow-deviation type algorithms (see, e.g., Fratta, Gerla, and Kleinrock [2]) and Gauss–Seidel type algorithms (see, e.g., Dafermos and Sparrow [1] and Magnanti [3]) require iterative calculation of loads. However, the algorithm appears to be complicated and rather difficult to understand.

In this paper, we consider the same model as Tantawi and Towsley [5] under the same assumptions concerning the communication delay. Additionally, we derive some properties that the optimal solution satisfies. On the basis of these properties, we offer another single-point algorithm that is more easily understandable and more straightforward than that of Tantawi and Towsley [5]. Furthermore, we identify properties relating to the convergence of our algorithm and demonstrate its performance.

Manuscript received May 9, 1989; revised August 26, 1989.

C. Kim is with the Department of Computer Engineering, College of Engineering, Yeungnam University, Gyongsan, 712-749 Korea.

H. Kameda is with the Department of Computer Science, and Information Mathematics, The University of Electro-Communications, Chofu-shi, Tokyo 182, Japan.

IEEE Log Number 9102593.

II. NOTATION AND ASSUMPTIONS

We assume the same model as that of Tantawi and Towsley [5]. That is, the system consists of n nodes (hosts) connected by a communications network. For reference, we repeat a portion of the notation and assumptions contained in [5] here (see Appendix C of [5]).

- β_i Job processing rate (load) at node i.
- $\boldsymbol{\beta}$ $[\beta_1, \beta_2, \cdots, \beta_n]$.
- ϕ_i External job arrival rate to node i.
- Φ Total external arrival rate ($\Phi = \sum_{i=1}^{n} \phi_i$).
- λ Network traffic.
- F_i Mean node delay of a job processed at node i—an increasing positive function.
- G Source-destination-independent mean communication delay—a nondecreasing positive function.

III. PROPERTIES OF THE OPTIMAL SOLUTION AND AN OPTIMAL LOAD BALANCING ALGORITHM

The problem of minimizing the mean response time of a job is expressed in the following formulations, as stated by Tantawi and Towsley [5].

$$\text{minimize} \quad D(\boldsymbol{\beta}) = \frac{1}{\Phi}\left[\sum_{i=1}^{n} \beta_i F_i(\beta_i) + \lambda G(\lambda)\right], \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \beta_i = \Phi,$$
$$\beta_i \geq 0, \qquad i = 1, 2, \cdots, n$$

where the network traffic λ may be expressed in terms of the variable β_i as

$$\lambda = \frac{1}{2}\sum_{i=1}^{n} |\phi_i - \beta_i|. \tag{2}$$

Define the following two functions.

$$f_i(\beta_i) = \frac{\partial}{\partial \beta_i}(\beta_i F_i(\beta_i)),$$
$$g(\lambda) = \frac{\partial}{\partial \lambda}(\lambda G(\lambda)).$$

Tantawi and Towsley [5] derived the following theorem by using the Kuhn–Tucker theorem (see Theorem 2 of [5]):

[Tantawi-Towsley Theorem]: The optimal solution to problem (1) satisfies the relations

$$\begin{aligned}
&f_i(\beta_i) \geq \alpha + g(\lambda), && \beta_i = 0 && (i \in R_d),\\
&f_i(\beta_i) = \alpha + g(\lambda), && 0 < \beta_i < \phi_i && (i \in R_a),\\
&\alpha \leq f_i(\beta_i) \leq \alpha + g(\lambda), && \beta_i = \phi_i && (i \in N),\\
&\alpha = f_i(\beta_i), && \beta_i > \phi_i && (i \in S),
\end{aligned} \tag{3}$$

subject to the total flow constraint

$$\sum_{i \in R_a} f_i^{-1}(\alpha + g(\lambda)) + \sum_{i \in N} \phi_i + \sum_{i \in S} f_i^{-1}(\alpha) = \Phi \tag{4}$$

where α is the Lagrange multiplier.

Tantawi and Towsley's single-point algorithm [5] first determines the node partitions (see steps 2–5 [5]). Then it solves (4) for α, and

obtains the values of β_i determined by that value of α (see step 6 [5]). In practical situations, we can rarely obtain the closed form giving α from (4). Therefore, some kind of iterative calculation of α is necessary. Since load balancing does not usually need very high accuracy, a simple method such as a binary search may be practical in solving (4) iteratively for α.

For the purpose of developing our algorithm, we derive the following properties directly from the Tantawi–Towsley theorem. Let β be an optimal solution to problem (1) and let

$$\alpha = \min_i f_i(\beta_i). \tag{5}$$

$$\lambda = \frac{1}{2}\sum_{i=1}^{n} |\phi_i - \beta_i|. \tag{6}$$

We now show from the above theorem that the following three properties hold true in the optimal solution.

Property 1:

$$\begin{array}{ll} f_i(0) \geq \alpha + g(\lambda). & \text{iff } \beta_i = 0. \\ f_i(\phi_i) > \alpha + g(\lambda) > f_i(0). & \text{iff } 0 < \beta_i < \phi_i. \\ \alpha \leq f_i(\phi_i) \leq \alpha + g(\lambda). & \text{iff } \beta_i = \phi_i. \\ \alpha > f_i(\phi_i). & \text{iff } \beta_i > \phi_i. \end{array} \tag{7}$$

Proof: From the original assumption, note that $f_i(\beta_i)$ and $g(\lambda)$ are increasing and nondecreasing, respectively, and both are positive. It is clear from the relations noted at (3) that $\alpha = \min_i f_i(\beta_i)$, which is the same as (5). Thus, both α's are the same. Note that λ defined by (6) is the network traffic in the optimal solution.

The necessity in (7): We derive from the stated set of relations (3),

$$\begin{array}{ll} f_i(0) \geq \alpha + g(\lambda). & \text{if } \beta_i = 0. \\ f_i(\phi_i) > \alpha + g(\lambda) > f_i(0). & \text{if } 0 < \beta_i < \phi_i. \\ \alpha \leq f_i(\phi_i) \leq \alpha + g(\lambda). & \text{if } \beta_i = \phi_i. \\ \alpha > f_i(\phi_i). & \text{if } \beta_i > \phi_i. \end{array}$$

The sufficiency in (7): We can show the reverse by contradiction. (For example, assume that, for some i, $\beta_i > 0$ when $f_i(0) \geq \alpha + g(\lambda)$. Then from the above necessity we see that for i, $\alpha + g(\lambda) > f_i(0)$, which would contradict the assumption.) □

Note the following definitions in the optimal solution.

$$\begin{aligned} R_d &= \{i \mid \beta_i = 0\} \quad \text{(idle sources)}, \\ R_a &= \{i \mid 0 < \beta_i < \phi_i\} \quad \text{(active sources)}, \\ N &= \{i \mid \beta_i = \phi_i\} \quad \text{(neutrals)}, \\ S &= \{i \mid \beta_i > \phi_i\} \quad \text{(sinks)}. \end{aligned}$$

Property 2:

$$\begin{array}{ll} \beta_i = 0, & i \in R_d. \\ \beta_i = f_i^{-1}(\alpha + g(\lambda)), & i \in R_a. \\ \beta_i = \phi_i, & i \in N. \\ \beta_i = f_i^{-1}(\alpha), & i \in S. \end{array}$$

Proof: This is clear from the Tantawi–Towsley theorem. □

Property 3:

$$\lambda = \lambda_S = \lambda_R,$$

where

$$\lambda_S = \sum_{i \in S} [f_i^{-1}(\alpha) - \phi_i], \tag{8}$$

$$\lambda_R = \sum_{i \in R_d} \phi_i + \sum_{i \in R_a} [\phi_i - f_i^{-1}(\alpha + g(\lambda_S))]. \tag{9}$$

Proof: We see that (4) is equivalent to the equality $\lambda_S = \lambda_R$, if we use the definitions given by (8) and (9) and note that $\sum_{i \in R_d} \phi_i + \sum_{i \in R_a} \phi_i + \sum_{i \in N} \phi_i + \sum_{i \in S} \phi_i = \Phi$.

Recall the above definitions on node partition. By noting that $\sum_{i=1}^{n} \beta_i = \sum_{i=1}^{n} \phi_i = \Phi$, we have

$$\lambda = \frac{1}{2}\sum_{i=1}^{n} |\phi_i - \beta_i| = \sum_{i \in S} (\beta_i - \phi_i).$$

Therefore, by noting Property 2 we see that $\lambda = \lambda_S$. □

Let us consider the following definitions in the order shown below for an arbitrary α (≥ 0).

$$S(\alpha) = \{i \mid \alpha > f_i(\phi_i)\}. \tag{10}$$

$$\lambda_S(\alpha) = \sum_{i \in S(\alpha)} [f_i^{-1}(\alpha) - \phi_i]. \tag{11}$$

$$R_d(\alpha) = \{i \mid f_i(0) \geq \alpha + g(\lambda_S(\alpha))\}. \tag{12}$$

$$R_a(\alpha) = \{i \mid f_i(\phi_i) > \alpha + g(\lambda_S(\alpha)) > f_i(0)\}. \tag{13}$$

$$\lambda_R(\alpha) = \sum_{i \in R_d(\alpha)} \phi_i + \sum_{i \in R_a(\alpha)} [\phi_i - f_i^{-1}(\alpha + g(\lambda_S(\alpha)))]. \tag{14}$$

$$N(\alpha) = \{i \mid \alpha \leq f_i(\phi_i) \leq \alpha + g(\lambda_S(\alpha))\}. \tag{15}$$

From properties 1, 2, and 3 above, we see that, if an optimal α is given, the node partitions in the optimal solution are characterized as follows.

$$R_d = R_d(\alpha), \quad R_a = R_a(\alpha), \quad N = N(\alpha), \quad \text{and} \quad S = S(\alpha).$$

Thus, we see that the node partitions are determined only by the conditions on $f_i(0)$, $f_i(\phi_i)$, and α. Furthermore, for an optimal α, we see that

$$\lambda = \lambda_S = \lambda_R = \lambda_S(\alpha) = \lambda_R(\alpha).$$

On the basis of properties 1, 2, and 3 above, the following load balancing algorithm is derived. The computational requirements of this algorithm are also given.

ALGORITHM 1:

1. Order nodes. $O(n \log n)$
   Order nodes such that $f_1(\phi_1) \leq f_2(\phi_2) \leq \cdots \leq f_n(\phi_n)$.
   If $f_1(\phi_1) + g(0) \geq f_n(\phi_n)$, then no load balancing is required.
2. Determine α. $O(n)$ (See following remarks.)
   Find α such that $\lambda_S(\alpha) = \lambda_R(\alpha)$
   (by using, for example, a binary search),
   where, given α, each value is calculated in the following order.

$$S(\alpha) = \{i \mid \alpha > f_i(\phi_i)\}. \tag{10}$$

$$\lambda_S(\alpha) = \sum_{i \in S(\alpha)} [f_i^{-1}(\alpha) - \phi_i]. \tag{11}$$

$$R_d(\alpha) = \{i \mid f_i(0) \geq \alpha + g(\lambda_S(\alpha))\}. \tag{12}$$

$$R_a(\alpha) = \{i \mid f_i(\phi_i) > \alpha + g(\lambda_S(\alpha)) > f_i(0)\}. \tag{13}$$

$$\lambda_R(\alpha) = \sum_{i \in R_d(\alpha)} \phi_i + \sum_{i \in R_a(\alpha)} [\phi_i - f_i^{-1}(\alpha + g(\lambda_S(\alpha)))]. \tag{14}$$

TABLE I
ALGORITHM COMPILATION AND COMPUTATION TIMES

Algorithm	Compilation time (s)	Computation time (ms) $\varepsilon = 10^{-2}$	10^{-3}	10^{-4}	10^{-5}	10^{-6}
T & T	2.11	3.07	3.24	3.66	3.80	4.17
K & K 1	0.78	1.88	2.10	2.65	2.86	3.30
K & K 1′	0.92	1.82	1.98	2.39	2.53	2.87

3. Determine the optimal load. $O(n)$

$$\beta_i = 0, \qquad \text{for } i \in R_d(\alpha).$$
$$\beta_i = f_i^{-1}(\alpha + g(\lambda)), \qquad \text{for } i \in R_a(\alpha).$$
$$\beta_i = f_i^{-1}(\alpha), \qquad \text{for } i \in S(\alpha).$$
$$\beta_i = \phi_i, \qquad \text{for } i \in N(\alpha).$$
$$\text{where } N(\alpha) = \{i \mid \alpha \le f_i(\phi_i) \le \alpha + g(\lambda_S(\alpha))\}. \quad (15)$$

Remarks: The main process of the algorithm is determining α (in step 2). The single-point algorithm of Tantawi and Towsley [5] determines the node partitions before calculating α exactly. Our algorithm does not need to provide the node partition process separately and can, instead, determine the node partitions during the process of obtaining α. The computational requirements given above are those with respect to the number of nodes n. In step 2, however, the computational requirements must increase as the acceptable tolerance for the relative error of α decreases. For example, if a binary search is used, the computational requirement then is $O(n \log 1/\varepsilon)$ where ε denotes the acceptable tolerance.

Continuing, consider the following additional properties.

Property 4: $\lambda_S(\alpha)$ increases (from zero) and $\lambda_R(\alpha)$ decreases (from Φ) both monotonically as α increases.

Proof: It is simple to prove this from definitions (11) and (14) and the assumptions on $f_i(\alpha)$ and $g(\lambda)$. □

Remark: This property assures that the algorithm can determine an α that satisfies $\lambda_S(\alpha) = \lambda_R(\alpha)$.

Property 5: For an arbitrary α such that $\alpha_1 \le \alpha \le \alpha_2$,

$$R_d(\alpha) = R_d(\alpha_1), \qquad \text{if } R_d(\alpha_1) = R_d(\alpha_2).$$
$$R_a(\alpha) = R_a(\alpha_1), \qquad \text{if } R_a(\alpha_1) = R_a(\alpha_2).$$
$$N(\alpha) = N(\alpha_1), \qquad \text{if } N(\alpha_1) = N(\alpha_2).$$
$$S(\alpha) = S(\alpha_1), \qquad \text{if } S(\alpha_1) = S(\alpha_2).$$

Proof: We first show $R_d(\alpha) = R_d(\alpha_1)$ for an arbitrary α $(\alpha_1 \le \alpha \le \alpha_2)$ if $R_d(\alpha_1) = R_d(\alpha_2)$. Order nodes such that

$$f_1(0) \ge f_2(0) \ge f_3(0) \ge \cdots \ge f_n(0).$$

$R_d(\alpha_1) = R_d(\alpha_2)$ implies that $f_i(0) \ge \alpha_1 + g(\lambda_S(\alpha_1)) > f_{i+1}(0)$ and $f_i(0) \ge \alpha_2 + g(\lambda_S(\alpha_2)) > f_{i+1}(0)$ for the same i. From Property 4 and the assumption on $g(\lambda)$ we see that

$$\alpha_1 + g(\lambda_S(\alpha_1)) \le \alpha + g(\lambda_S(\alpha)) \le \alpha_2 + g(\lambda_S(\alpha_2)). \quad (18)$$

Thus, we see that

$$f_i(0) \ge \alpha + g(\lambda_S(\alpha)) > f_{i+1}(0)$$

which implies

$$R_d(\alpha) = R_d(\alpha_1) = R_d(\alpha_2).$$

The rest is shown similarly. □

Remarks: As an example, through a binary search, we can obtain a sequence $\alpha_1, \alpha_2, \cdots$ as candidates for optimal α. In the case where $(\lambda_S(\alpha_i) - \lambda_R(\alpha_i))(\lambda_S(\alpha_{i+1}) - \lambda_R(\alpha_{i+1})) < 0$ then α_{i+2} is selected between α_i and α_{i+1}. In this case we derive

$$R_d = R_d(\alpha_{i+j}) = R_d(\alpha_{i+1}), \quad j = 2, 3, \cdots \quad \text{if } R_d(\alpha_i) = R_d(\alpha_{i+1}).$$
$$R_a = R_a(\alpha_{i+j}) = R_a(\alpha_{i+1}), \quad j = 2, 3, \cdots \quad \text{if } R_a(\alpha_i) = R_a(\alpha_{i+1}).$$
$$N = N(\alpha_{i+j}) = N(\alpha_{i+1}), \quad j = 2, 3, \cdots \quad \text{if } N(\alpha_i) = N(\alpha_{i+1}).$$
$$S = S(\alpha_{i+j}) = S(\alpha_{i+1}), \quad j = 2, 3, \cdots \quad \text{if } S(\alpha_i) = S(\alpha_{i+1}).$$

On the basis of the above, we derive another version of the above algorithm that may be a bit more complicated but may require less computational time.

IV. COMPARISON OF LOAD BALANCING ALGORITHM PERFORMANCE

We wrote FORTRAN77 programs implementing Tantawi and Towsley's algorithm (T & T), Algorithm 1 (K & K 1) as above, and Algorithm 1′ (K & K 1′) remade from Algorithm 1 by incorporating Property 5. The numbers of program steps for Algorithms 1 and 1′ are about a third and about two fifths of that of Tantawi and Towsley's algorithm, respectively.

Table I compares the performance of the algorithms for a distributed computer system model that consists of ten hosts connected via a single channel. The channel is modeled as an M/M/1 queueing system. Each node is modeled as a central server model. Optimal node partition is such that one sink (S), two neutrals (N), five active sources (R_a), and two idle sources (R_d) exist. The compilation and computation times of each algorithm are shown. ε indicates the acceptable tolerance for the relative error of α.

As we see from Table I, for a large acceptable tolerance ε, the computational times of K & K 1 and 1′ are nearly equal to each other and much less than that of T & T. The difference between those of T & T and K & K 1, however, becomes less as ε decreases. On the other hand, the difference between those of T & T and K & K 1′ is rather independent of ε. Such a tendency can be observed for different parameter values. As such, if we desire a simple algorithm and accuracy (i.e., the size of ε) is not important, K & K 1 is recommended. If we require very high accuracy levels, K & K 1′ is recommended.

V. AN ALGORITHM FOR STAR NETWORK CONFIGURATIONS

We can also apply the above idea to the case of the star network configurations studied by Tantawi and Towsley [4], and derive the following load balancing algorithm. For notation and assumptions, see [4].

ALGORITHM 2:

1. Order nodes.
   Order nodes such that $h_1(\phi_1) \geq h_2(\phi_2) \geq \cdots \geq h_n(\phi_n)$.
   If $h_1(\phi_1) \leq f_0(\phi_0)$, then no load balancing is required.
2. Determine α.
   Find α such that $\lambda_0(\alpha) = \lambda_R(\alpha)$
   (by using, for example, a binary search),
   where, given α, each value is calculated in the following order.

$$\lambda_0(\alpha) = f_0^{-1}(\alpha) - \phi_0,$$
$$R_d(\alpha) = \{i \mid h_i(0) \geq \alpha\},$$
$$R_a(\alpha) = \{i \mid h_i(\phi_i) > \alpha > h_i(0)\},$$
$$\lambda_R(\alpha) = \sum_{i \in R_d(\alpha)} \phi_i + \sum_{i \in R_a(\alpha)} [\phi_i - h_i^{-1}(\alpha)].$$

3. Determine the optimal load.

$$\beta_i = 0, \qquad \text{for } i \in R_d(\alpha),$$
$$\beta_i = h_i^{-1}(\alpha), \qquad \text{for } i \in R_a(\alpha),$$
$$\beta_i = \phi_i, \qquad \text{for } i \in N(\alpha)$$
$$\text{where, } N(\alpha) = \{i \mid h_i(\phi_i) \leq \alpha\}.$$

VI. Conclusion

This paper has proposed simpler versions of single-point algorithms than those of Tantawi and Towsley [5], [4] for the models of distributed computer systems with a single communication channel and star network configurations. The idea underlying our algorithms may also be used for some other related models that have a single job class. It would seem difficult to develop algorithms that yield the optimal solution of multiple job class models without iterative calculation of loads. However, it would be easy to develop iterative algorithms for multiple job class models at each stage in the iteration utilizing these same ideas.

Acknowledgment

The authors are grateful to S. A. Fondacaro who suggested a lot of improvements in the presentation.

References

[1] S. C. Dafermos and F. T. Sparrow, "The traffic assignment problem for a general network," *J. Res. Nat. Bureau of Standards—B. Mathematical Sci.*, vol. 73b, no. 2, pp. 91–118 Apr.–June 1969.
[2] L. Fratta, M. Gerla, and L. Kleinrock, "The flow deviation method: An approach to store-and-forward communication network design," *Networks*, vol. 3, pp. 97–133, 1973.
[3] T. L. Magnanti, "Models and algorithms for predicting urban traffic equilibria," in *Transportation Planning Models*, M. Florian, Ed. Elsevier Science (North-Holland), 1984, pp. 153–185.
[4] A. N. Tantawi and D. Towsley, "A general model for optimal static load balancing in star network configurations," in *PERFORMANCE '84*, E. Gelenbe, Ed. Elsevier Science (North-Holland), 1984, pp. 277–291.
[5] ——, "Optimal static load balancing in distributed computer systems," *J. ACM*, vol. 32, no. 2, pp. 455–465, Apr. 1985.