스타형 컴퓨터 네트워크의 부하균형방향 정책

임 경 수† 김 수 정†† 김 종 근†††

요 약

스타형 컴퓨터 네트워크의 최적 부하균형 문제를 생각한다. 스타형 모델에서는 세가지 부하균형방향 정책을 생각할 수 있다. 첫번째는, 주변노드로 도착한 작업들은 자노드에서 처리되거나 자노드가 과부하인 경우 중앙노드에 보내져 처리될 수 있다. 두번째는, 중앙노드에 도착한 작업만이 자노드에서 처리되거나 중앙노드가 과부하인 경우 저부하의 주변노드로 작업이 전송되어 처리될 수 있다. 본 연구에서는 두번째 정책에 대하여 비선형 최적화 문제를 만들고, 최적해를 이용하여 최적 부하균형 알고리즘을 제안한다. 세번째는, 중앙노드 혹은 주변노드를 가리지 않고 도착한 작업들은 자노드에서 처리되거나 자노드가 과부하인 경우 저부하의 타노드로 작업이 전송되어 처리된다. 이 세번째 부하균형 정책을 위한 부하분산 알고리즘도 제안한다. 본 연구에서 대상으로 하고 있는 세가지 부하균형 정책을 수치실험을 통하여 비교분석 하였다. 수치실험 결과중 다음과 같은 몇 가지 재미있는 실험결과가 발견되었다. 세번째 부하균형 정책은 대부분의 경우 다른 두 정책보다 시스템 성능을 크게 향상시킨다. 두번째 부하균형 정책은 대부분의 경우 미약한 성능향상 밖에 보이지 않는다. 마지막으로 중앙노드의 처리능력이 주변노드보다 훨씬 큰 경우에는 첫번째와 세번째의 부하균형 정책은 동일한 성능향상을 보인다.

Load balancing Direction strategies in star network configurations

Kyung Soo Lim†, Soo Jeong Kim†† and Chong Gun Kim†††

ABSTRACT

Optimal static load balancing in star network configurations is considered. Three kinds of load balancing direction strategies are considered. First, a job arriving at the peripheral nodes may be processed either where it arrived(origin node)or transferred directly to central node. Second, a job arriving at the central node may be processed there, or transferred to lightly loaded peripheral nodes. A nonlinear optimization problem is formulated. Using the optimal solution, an optimal load balancing algorithm is derived for the second load balancing strategy. Third, a job arriving at the central node or a peripheral node may be processed either at origin node or transferred to another lightly loaded node (central or peripheral). A load balancing algorithm is derived for the third load balancing strategy. The effects of these three load balancing strategies are compared by numerical experiments. During the conduct of these in numerical experiments, several interesting phenomena were observed. The third load balancing strategy improved performance more than the first two other strategies. The second load balancing strategy, as a whole, resulted in only slightly improved performance. Finally, if the central node has larger processing power than the peripheral nodes, the first and third load balancing strategies produce equal performance improvement.

1. Introduction

In this study, we consider star network configurations where a central node such as a workstation, is connected by communication links to heterogeneous peripheral nodes. In this model, by balancing loads from overloaded node to lightly loaded node, the user of a heavy loaded node will be able to draw upon

† 정 회 원 : 연암공업전문대학 전자계산과 조교수
†† 정 회 원 : 영남대학교 전산공학과 박사과정
††† 정 회 원 : 영남대학교 전산공학과 조교수
논문접수 : 1994년 7월19일, 심사완료 : 1994년 11월19일

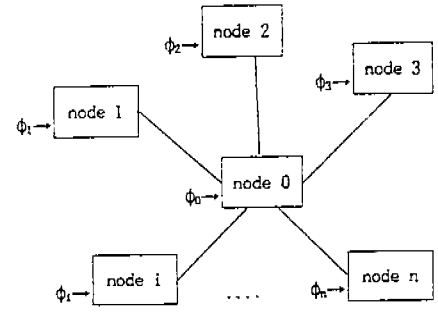
unused capacity residing elsewhere in the system and favorably feel that processing power and storage capacity of the node are increased. The overall purpose of load balancing, in this study, is to minimize mean job response time of the system.

Load balancing may be either dynamic[1, 6] or static. At first, it would appear that dynamic load balancing is more effective than static load balancing. However, upon closer examination, there are many costly, built-in drawbacks, e.g. realtime system information exchange and implementation in most cases. Static load balancing strategies, on the other hand, are based on the average behavior of the system[2, 3, 4, 5, 8].

There have been many studies on load balancing in star network configurations. But most of them have focused on flow control which is improving system performance by transferring overflow jobs on overloaded peripheral nodes to central node only[1, 2, 3].

An approximation for numerical analysis by the queueing theories for dynamic load balancing has also been studied[1]. This showed adaptive control techniques for controlling the flow of real time jobs.

In this paper, we consider three distinct load balancing direction strategies : First, where jobs arriving at an overloaded peripheral node are transferred only to the central node. (Flow Control algorithm 1 : FC 1). Several studies of load balancing in star network configurations are considered under this strategy[1, 2, 3]. Second, jobs arriving at the central node are transferred to lightly loaded peripheral nodes. (Flow Control algorithm 2 : FC 2). In this strategy, we formulate a nonlinear optimization problem and derive an optimal load balancing algorithm which minimizes the mean response time. Third, jobs arriving at either the central or a peripheral node are transferred to any node(central or peripheral) when the origin node is overloaded. (Load Balancing algorithm : LB)[8]. In this strategy a new load balancing algorithm is derived. Performance comparison of three proposed load balancing strategies, and the effects of changing system parameters on system behavior are also examined.



(Fig. 1) A star network configuration

2. System model

The model is a star network configuration which consist of a set of n heterogeneous peripheral nodes connected to the central node. The central node and the peripheral nodes may contain a number of resources such as CPU and Input/Output devices. Nodes may have different job arrival rate and processing power. Mathematical notations and key assumptions are as follows:

ϕ_i External job arrival rate to node i, $i = 0, 1, \ldots, n$

β_i Job processing rate (load) at node i, $i = 0, 1, \ldots, n$

λ_i Link traffic at link i, $i = 0, 1, \ldots, n$

x_{ij} Job flow rate from node i to j

β $[\beta_0, \beta_1, \ldots, \beta_n]$

φ $[\phi_0, \phi_1, \ldots, \phi_n]$

x $[x_{00}, x_{01}, \ldots, x_{0n}, x_{10}, x_{11}, \ldots, x_{1n}, \ldots, x_{n0}, x_{n1}, \ldots, x_{nn}]$

Φ Total external job arrival rate ($\Phi = \sum_{i=0}^{n} \phi_i$)

$F_i(\beta_i)$ Mean node delay of a job processed at node i — We assume that it is a differentiable, increasing and convex function

$G_i(\lambda_i)$ Mean communication delay of a job between node i and the central node- We assume that it is a differentiable, nondecreasing and convex function

Jobs arrive at each node according to time-invariant Poisson distributions. This job is then processed at the node or, if the node is overloaded, is transferred to a lightly loaded node which can be the central node or a peripheral node according to one of the strategies covered above. All jobs are processed either locally or remotely.

Nodes are classified as follows:

A. Idle source (Rd) : The node does not process any jobs and sends all its incoming jobs to another node. That is, $\beta_i = 0$.

B. Active source(Ra) : If overloaded, the node sends a part of the arriving jobs to other nodes, but does not receive jobs from other nodes.

C. Neutral(N) : The node processes jobs locally without sending or receiving jobs to or from other nodes.

D. Sink(S) : The node receives and processes jobs from other nodes but does not send any jobs out to other nodes.

The problem of minimizing the mean response time of a job is expressed as follows :

$$\text{minimize} \quad D(\beta) = \frac{1}{\Phi}\left[\sum_{i=0}^{n} \beta_i F_i(\beta_i) + \sum_{i=1}^{n} \lambda_i G_i(\lambda_i)\right] \qquad (1)$$

subject to

$$\sum_{i=0}^{n} \beta_i = \Phi,$$

$$\beta_i \geq 0, \quad i = 0, 1, \ldots, n$$

Formula (1) can be easily obtained by using the Little's result that [a mean number of jobs in the system = total job arrival rate × mean response time][7]. λ_i is the traffic flow of communication link i. Mathematically λ_i is expressed as the difference between external job arrival rate and job processing rate at node i. We assume that $G_i(\lambda_i)$ includes two communication delays : a delay caused by sending a job to a remote node and a delay caused by receiving the response back to the node of origin. λ_i is expressed as follows,

$$\lambda_i = |\phi_i - \beta_i| \qquad i = 1, 2, \ldots, n.$$

We also define the following two functions. $f_i(\beta_i)$ is incremental node delay and $g_i(\lambda_i)$ is incremental communication delay.

$$f_i(\beta_i) = \frac{\partial \beta_i F_i(\beta_i)}{\partial \beta_i}, \qquad (2)$$

$$g_i(\lambda_i) = \frac{\partial \lambda_i G_i(\lambda_i)}{\partial \lambda_i}. \qquad (3)$$

We may assign the flow rate from node i to node j as follows :

$$x_{ij} = \begin{cases} \dfrac{(\beta_j - \phi_j)}{\sum_{k \in R_a R_d} \lambda_k} (\phi_i - \beta_i), & i \in R_a, R_d \quad j \in S, \\ 0, & \text{in all other cases.} \end{cases}$$

3. The optimal solutions and optimal load balancing algorithms

3.1 Load balancing from peripheral nodes to central node (FC 1)

Tantawi and Towsley[2] proposed an optimal load balancing strategy which determines the optimal load at each peripheral nodes by balancing overloaded jobs from peripheral nodes only to central node. A constrain, $\beta_i \leq \phi_i$, $i=1,2,\cdots,n$, is added to formula (1) for the first strategy. They also derived the optimal load balancing algorithm. Theorem 1 is optimal solution derived by Tantawi and Towsley[2], but the formula is changed for convenience.

Theorem 1. The optimal solution to problem (1) satisfies the following relations when the first load balancing strategy is considered.

$$f_i(\beta_i) \geq \alpha + g_i(\lambda_i) \qquad \beta_i = o, \quad (i \in R_d),$$

$$f_i(\beta_i) = \alpha + g_i(\lambda_i) \qquad 0 < \beta_i < \phi_i, \quad (i \in R_a),$$

$$f_i(\beta_i) \leq \alpha + g_i(\lambda_i) \qquad \beta_i = \phi_i, \quad (i \in R_a), \quad i = 1,2,\ldots,n,$$

and

$$f_0(\beta_0) = \alpha, \tag{4}$$

subject to

$$f^{-1}{}_0(\alpha) + \sum_{i \in R_a} f^{-1}{}_i(\alpha + g_i(\lambda_i)) + \sum_{i \in N} \phi_i = \Phi, \tag{5}$$

where α is the Lagrange multiplier.

Proof. (See [2].[3])

Under the same model and the same assumptions as Tantawi & Towsley[2], Kim and Kameda[3] derived three properties from the Theorem 1. On the basis of these three properties, they derived an load balancing algorithm[3] that is easily understandable and more straightforward than that of Tantawi and Towsley[2]. We used Kim and Kameda's algorithm[3] for our experiments.

3.2 Load balancing from central node to peripheral nodes (FC 2)

The second load balancing strategy deals with overloaded jobs arriving at the central node which may be transferred to lightly loaded peripheral nodes. The optimal solution can be obtained by solving the following optimization problem:

$$\text{minimize} \quad D(\beta) = \frac{1}{\Phi}\left[\sum_{i=0}^{n} \beta_i F_i(\beta_i) + \sum_{i=1}^{n} (\beta_i - \phi_i) G_i(\beta_i - \phi_i)\right] \tag{6}$$

subject to

$$\sum_{i=0}^{n} \beta_i = \Phi,$$

$$\beta_i \geq \phi_i, \quad i = 1,2,\ldots,n,$$

$$\beta_i \geq 0, \quad i = 0,1,\ldots,n.$$

The following Theorem 2 is derived by using the Kuhn-Tucker conditions.

Theorem 2. The optimal solution to problem (6) satisfies the relations,

$$f_o(\beta_0) \geq \alpha, \qquad \beta_0 = 0, \quad (0 \geq R_d), \tag{7.a}$$

$$f_o(\beta_0) = \alpha, \qquad 0 < \beta_0 < \phi_0, \quad (0 \geq R_a), \tag{7.b}$$

$$\alpha - g_i(\lambda_i) \le f_i(\beta_i), \qquad \beta_i = \phi_i, \quad (i \in N), \qquad (7.c)$$

$$\alpha - g_i(\lambda_i) = f_i(\beta_i), \qquad \beta_i > \phi_i, \quad (i \in S), \qquad (7.d)$$

subject to the total flow constraint,

$$f^{-1}{}_0(\alpha) + \sum_{i \in S} f^{-1}{}_i(\alpha - g_i(\lambda_i)) + \sum_{i \in N} \phi_i = \Phi, \qquad (8)$$

where α is the Lagrange multiplier.

Proof. By multiplying the objective function $D(\beta)$ by the constant Φ and combining it with the constraints of problem (6), we obtain the Lagrangian function,

$$L(\beta, \alpha, \gamma, \varphi) = \Phi D(\beta) + \alpha(\Phi - \sum_{i=0}^{n} \beta_i) + \sum_{i=1}^{n} \gamma_i(\beta_i - \phi_i) + \sum_{i=0}^{n} \varphi_i \beta_i. \qquad (9)$$

The optimal solution satisfies the Kuhn-Tucker conditions.

$$\frac{\partial L}{\partial \beta_i} = f_i'(\beta_i) + g_i(\beta_i - \phi_i) - \alpha + \gamma_i + \varphi_i = 0, \quad i = 1,2,...,n \qquad (10)$$

$$\frac{\partial L}{\partial \beta_0} = f_0(\beta_0) - \alpha + \varphi_0 = 0, \qquad (11)$$

$$\frac{\partial L}{\partial \alpha} = \Phi - \sum_{i=0}^{n} \beta_i = 0, \qquad (12)$$

$$(\beta_i - \phi_i) \ge 0, \ \gamma_i(\beta_i - \phi_i) = 0, \ \gamma_i \le 0, \quad i = 1,2,...,n \qquad (13)$$

$$\beta_i \ge 0, \ \varphi_i \beta_i = 0, \text{ and } \varphi \le 0, \quad i = 1,2,...,n \qquad (14)$$

From condition (13), we note that $(\beta_i - \phi_i) = 1, 2, \cdots n$ may be either zero or positive. We consider each β_i case separately.

Case 1. $\beta_i = \phi_i$

It follows from conditions (13) and (14) that $\gamma_i \le 0$ and $\psi_i = 0$, respectively. Thus equation (10) becomes

$$f_i(\beta_i) \ge \alpha - g_i(\beta_i - \phi_i), \quad \beta_i = \phi_i. \qquad (15)$$

Case 2. $\beta_i > \phi_i$.

It follows from conditions (13) and (14) that $\gamma_i = 0$ and $\psi_i = 0$, respectively. Thus equation (10) becomes

$$f_i(\beta_i) = \alpha - g_i(\beta_i - \phi_i), \quad \beta_i > \phi_i. \qquad (16)$$

Case 3. $\beta_0 = 0$.

It follows from condition (14) that $\psi_i \le 0$. Thus equation (11) becomes

$$f_0(\beta_0) \ge \alpha, \quad 0 < \beta_0 < \phi_0 \qquad (17)$$

Case 4. $\beta_0 > 0$.

It follows from condition (14) that $\psi_i = 0$. Thus equation (11) becomes

$$f_0(\beta_0) = \alpha, \quad 0 < \beta_0 < \phi_0$$

which may be written as $\beta_0 = f^{-1}{}_0(\alpha)$. (18)

The Lagrange multiplier α may be obtained from the total flow constraint,

$$\sum_{i=0}^{n} \beta_i = \Phi$$

Once the node partition is determined, we derive equation (8) by substituting equation (16) and (18) into the above equation.

If β is the optimal solution of problem (6) and

$$\alpha = f_o(\beta_0) \qquad (19)$$

then, we can get following three properties from Theorem 2.

Property 1.

$f_0(0) \ge \alpha$, *iff* $\beta_0 = 0$,

$$\begin{aligned} &f_0(\phi_0) > \alpha > f_0(0), \quad iff \quad 0 < \beta_0 < \phi_0, \\ &\alpha - g_i(\lambda_i) \le f_i(\phi_i), \quad iff \quad \beta_i = \phi_i, \\ &\alpha - g_i(\lambda_i) > f_i(\phi_i), \quad iff \quad \beta_i > \phi_i, \\ &\qquad\qquad i = 1,2,...,n \end{aligned} \tag{20}$$

Proof. From our assumptions, $f_i(\beta_i)$ is a increasing function and $g_i(\lambda_i)$ is a nondecreasing function. It is clear from the relation (7) that $\alpha = \min_i f_i(\beta_i)$ $i=1, 2, \cdots, n$, which is the same as (19).

The necessity : We can get easily from relations (7).

$$\begin{aligned} &f_0(0) \ge \alpha, && \text{if} \quad \beta_0 = 0, \\ &f_0(\phi_0) > \alpha > f_0(0), && \text{if} \quad 0 < \beta_0 < \phi_0, \\ &\alpha - g_i(\lambda_i) \le f_i(\phi_i), && \text{if} \quad \beta_i = \phi_i, \\ &\alpha - g_i(\lambda_i) > f_i(\phi_i), && \text{if} \quad \beta_i > \phi_i, \\ &\qquad\qquad i = 1,2,...,n \end{aligned}$$

The sufficiency : It is proven by contradiction. For example, assume that $\beta_0 > 0$ when $f_0(0) \ge \alpha$. Then from the above necessity, central node may be $\alpha > f_0(0)$ in this case, which would contradict the assumption.

Property 2. If β is the optimal solution of problem (6), then we have

$$\begin{aligned} &\beta_0 = 0, && 0 \in R_d, \\ &\beta_0 = f^{-1}{}_0(\alpha), && 0 \in R_a, \\ &\beta_i = \phi_i, && i \in N, \\ &\beta_i = f^{-1}{}_i(\alpha - g_i(\lambda_i)), && i \in S, \quad i = 1,2,...,n \end{aligned}$$

Proof. This is clear from theorem 2.

Property 3. If β is the optimal solution of problem (6), then we have

$$\lambda_0 = \lambda_s,$$

where

$$\lambda_0 = [\phi_0 - f^{-1}{}_0(\alpha)], \tag{21}$$

$$\lambda_s = \sum_{i \in S} [f^{-1}{}_i(\alpha - g_i(\lambda_i)) - \phi_i]. \tag{22}$$

Proof. Equations (21) and (22) are clear from equation (7.b), (7.d) and (8).

Remark that equation (8) is equivalent to the equality $\lambda_0 = \lambda_s$.

On the basis of these three properties, a load balancing algorithm satisfying the problem (6) is proposed as follows:

ALGORITHM 1 : An algorithm for obtaining optimal solution of problem (6), (FC 2).

STEP 1. Order nodes.

Order nodes such that $f_1(\phi_1) \le f_2(\phi_2) \le \cdots \le f_n(\phi_n)$.

If $f_0(\phi_0) - g_1(\lambda_1) \le f_1(\beta_1)$, then no load balancing is required.

STEP 2. Determine α

Find α such that $\lambda_0(\alpha) = \lambda_s(\alpha)$ (by using, for example, a binary search), where, given α, each value is calculated in the following order,

$$S(\alpha) = \{i \mid \alpha - g_i(\lambda_i(\alpha)) > f_i(\phi_i)\},$$

$$\lambda_s(\alpha) = \sum_{i \in S} [f^{-1}{}_i(\alpha - g_i(\lambda_i(\alpha))) - \phi_i],$$

$$\lambda_0(\alpha) = [\phi_0 - f^{-1}{}_0(\alpha)].$$

STEP 3. Determine the optimal load.

$$\begin{aligned} &\beta_0 = 0, && \text{for } 0 \in R_d(\alpha), \\ &\beta_0 = f^{-1}{}_0(\alpha), && \text{for } 0 \in R_a(\alpha), \\ &\beta_i = \phi_i, && \text{for } i \in N(\alpha), \\ &\beta_i = f^{-1}{}_i(\alpha - g_i(\lambda_i)), && \text{for } i \in S(\alpha), \\ &&& i = 1,2,...,n \end{aligned}$$

3.3 Load balancing from any overloaded node to any lightly loaded node (Central or Peripheral) (LB).

As another solution to problem (1), we

consider the third load balancing strategy. This is where a job arriving at an overloaded source node (either central node or a peripheral node) is transferred to any lightly loaded node (again either central or peripheral). It is more difficult to fomulate and to solve an optimization problem for this model than that of FC 1 or FC 2. [8] shows a load balancing algorithm which can obtain optimal solution. We also derive a simple load balancing algorithm in which the FC 1 and FC 2 algorithms are used for subalgorithms. In this algorithm, ϕ'_i do the same role of ϕ_i in FC 1 and ϕ''_i do the same role of ϕ_i in FC 2.

Since the objective function is convex by the assumption and the feasible region is a convex set, any local solution point of the problem is a global solution point.

ALGORITHM 2 : An algorithm for optimal solution of problem (1), (LB).

STEP 1 Let L=0. (L: iteration number)

$$\beta_i = \phi_i, \quad i = 0,1,2,...,n$$

STEP 2 L=L+1.

$$\phi'_i = \beta_i, \quad i = 0,1,2,...,n$$

[FC1 algorithm]

$$\phi''_i = \beta_i, \quad i = 0,1,2,...,n$$

[FC2 algorithm]

STEP 3 (Stopping rule) Compare $D(\beta_{L-1})$ and $D(\beta_L)$.

If $|D(\beta_L)-D(\beta_{L-1})| < \varepsilon$, where $\varepsilon > 0$ is a properly chosen acceptance tolerance, then STOP. Otherwise go to STEP 2.

Numerical experiments may show hat the minimum of $D(\beta)$ can be obtained by using the above proposed algorithm.

4. Numerical experiments

4.1 The models used in numerical experiments

As before, our model is a star network configurations in which a central node is connected to n heterogeneous peripheral nodes. Each node is modeled as an M/M/1 queueing system that is a differentiable, increasing and convex function[7]. Mean response time at each node is given as follows,

$$F_i(\beta_i) = \frac{1}{\mu_i - \beta_i} \qquad \beta_i < \mu_i \qquad (23)$$

where μ_i is the mean service rate and β_i is the mean arrival rate at node i

Substituting the equation (23) into the equation (2), $f_i(\beta_i)$ may be expressed as follows,

$$f_i(\beta_i) = \frac{\mu_i}{(\mu_i - \beta_i)^2} \qquad \beta_i < \mu_i \qquad (24)$$

Each communication link is modeled as an M/G/∞ to simplify the experiments. Mean response time at each link can be expressed as $G_i(\lambda_i)=t$, $g_i(\lambda_i)=t$.

〈Table 1〉 The set of parameter values of system models

		node 0	node 1	node 2	node 3	node 4
CASE 1	μ_i	10->40	10	10	10	10
	ϕ_i	5	3	1	7	9
CASE 2	μ_i	10	10	10->40	10	10
	ϕ_i	5	3	1	7	9
CASE 3	μ_i	10	10	10	10	10->40
	ϕ_i	5	3	1	7	9
CASE 4	μ_i	10->40	10	10	10	10
	ϕ_i	8	3	1	4	5
	$G_i(\lambda_i)$	0.1	0.1	0.1	0.1	0.1

〈Table 1〉 shows the set of parameter values of system models. In this experiments, We set that processing capacity of each node is 10, while the job arrival rate of each node is different. To see the effect of changing

processing capacity, processing capacity of the central node(i.e. node 0, CASE 1 and CASE 4), lightly loaded peripheral node(i.e. node 2, CASE 2), and overloaded peripheral node(i.e. node 4, CASE 3) is changed from 10 to 40.

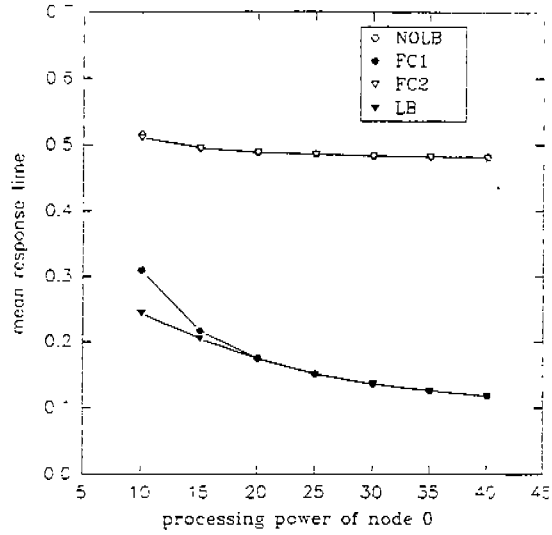
4.2 Experimental results

Performance should be favorably improved by any load balancing strategy. 〈Table 2〉 shows the number of iterations and the system wide mean job response times at each iteration steps by the ALGORITHM 2. L is the number of iterations for the algorithm, FC 1 is result of applying the first load balancing algorithm proposed by Kim & Kameda[3] and FC 2 is the result of applying the second load balancing algorithm proposed in 3.2 which are used in ALGORITHM 2 as subalgorithms. In 〈Table 2〉, as the number of iterations increases, the system wide mean job response time decreases gradually, proving that the minimum of $D(\beta)$ can be obtained by ALGORITHM 2.

〈Table 2〉 Mean job response times at each step of algorithm iterations

(μ_0 = 15, μ_i = 10, i = 1,2,...,n)

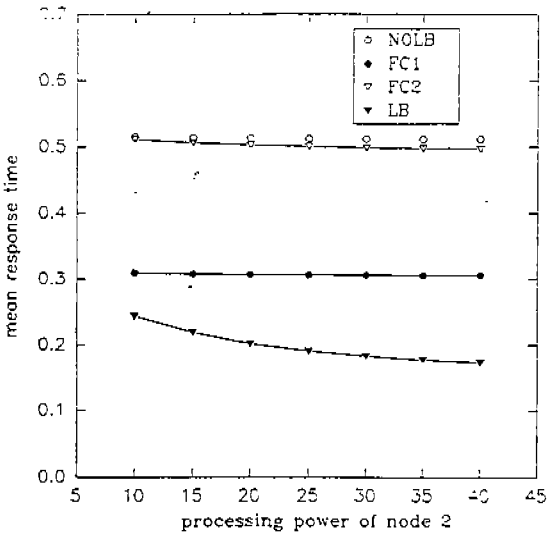
ALGORITHM 2	Subalgorithm	Mean job response time
NOLB		0.494921
L = 1	FC1 FC2	0.216797 0.209276
L = 2	FC1 FC2	0.205956 0.205057
L = 3	FC1 FC2	0.204576 0.204525
L = 4	FC1 FC2	0.204505 0.204497
L = 5	FC1 FC2	0.204494 0.204493
OPTIMAL		0.204491



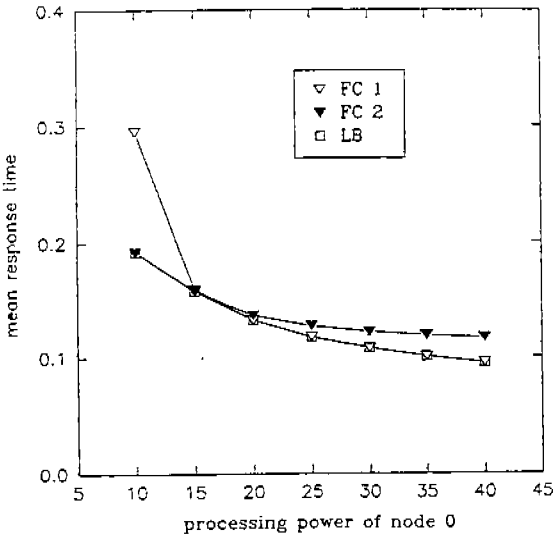
(Fig. 2) The effects of changing central node processing capacity (CASE 1)

(Fig. 2) shows the effects of central node's processing capacity on the behavior of system wide mean job response time when all three load balancing strategies are applied. The effect of no load balancing is shown for comparison. In (Fig. 2), it is observed that FC 2 has little performance improvement over that of No Load Balancing(NOLB), while FC 1 shows considerable performance improvement over that of FC 2. LB shows significantly more performance improvement than FC 1 when central node and peripheral nodes have the same processing capacity. However, LB shows little or no performance improvement over that of FC 1 when central node has larger processing capacity than peripheral nodes. Therefore, FC 1 is sufficient load balancing strategy to improve system performance when central node's processing capacity is larger than that of peripheral nodes.

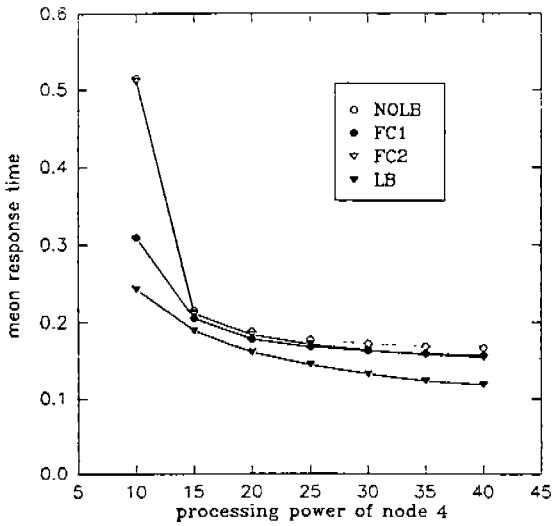
(Fig. 3) shows the effects of changing the processing capacity of lightly loaded peripher-



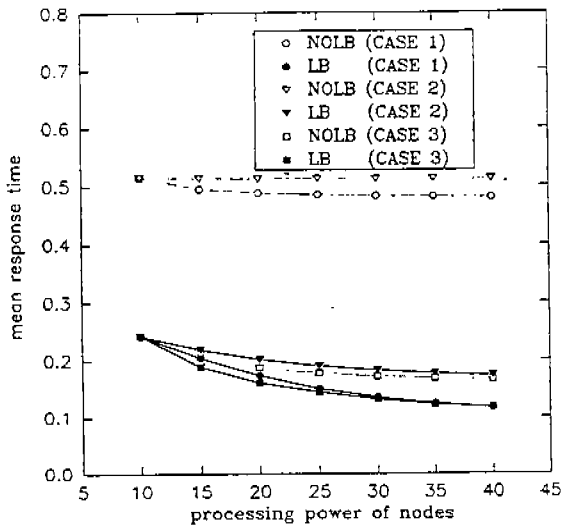
(Fig. 3) The effects of changing lightly loaded nodes' processing capacity (CASE 2)



(Fig. 5) The effect of changing central node processing capacity (CASE 4)



(Fig. 4) The effects of changing overloaded nodes' processing capacity (CASE 3)



(Fig. 6) The comparison of CASE 1, CASE 2 and CASE 3 in NOLB and LB

al node on the behavior of system wide mean job response time in CASE 2. FC 1 shows little improvement, while LB and FC 2 show some improvement on system performance.

(Fig. 4) shows the effects of changing the processing capacity of overloaded peripheral nodes on the behavior of system wide mean job response time in CASE 3. It is observed that application of FC 1 and FC 2 show little effect, but LB shows considerable perform-

ance improvement. An additional interesting phenomenon is that increasing the processing capacity of heavily loaded node shows apparent performance improving when applying NOLB.

(Fig. 5) shows the effects of central node's processing capacity on the behavior of system wide mean job response time in CASE 4. It is observed that FC 2 and LB show the same performance improving when central node and peripheral nodes have the same processing capacity, while FC 1 and LB show the same performance improving when central node's processing capacity is larger than that of peripheral nodes.

(Fig. 6) compares CASE 1, CASE 2 and CASE 3 on the system wide mean job response time only for LB and NOLB. It is shown that increasing the central node's or overloaded peripheral node's processing capacity is to improve the system performance effectively.

LB may show remarkable performance improvement when the central node and peripheral nodes have similar processing power and some lightly loaded peripheral nodes exist. But, since the central node's processing capacity is generally the more powerful than that of peripheral nodes and LB strategy is more complicated than the others. LB is not recommended except in the special case mentioned above. System performance can also be improved by not only balancing overloaded jobs but also improving overloaded node's processing capacity.

5. Conclusion

Three load balancing strategies were considered: 1) a job arriving at the overloaded peripheral nodes is transferred only to central node, 2) a job arriving only at the central node is transferred to lightly loaded peripheral nodes, and 3) a job arriving at an overloaded source node(either the central or peripheral nodes) is transferred to another node (also the central or peripheral nodes). The first strategy was considered in the several previous load balancing studies in star network configurations. Nonlinear optimization problem was formulated and from that an optimal load balancing algorithm was derived for the second strategy. An optimal load balancing algorithm was also derived for the third strategy. Performance comparisons of all three load balancing strategies, and that of no load balancing were also studied.

The third strategy is more effective than the first strategy when central node and peripheral nodes have similar processing powers. When the central node's processing capacity is greater than that of the peripheral nodes, the first and third strategies show equal system performance, and both are superior to the second strategy. However it must be kept in mind that the first strategy is simpler than the third, and therefore can be implemented more easily and cheaply. In special parameter cases, second and third strategies show the equal best performance improving.

This study has shown that system performance, in general, can be improved by any load balancing direction strategy. Of the three strategies considered, in this study, the third strategy works best all the time. However again keep in mind that the third strategy have worst complexity. Hopefully, this study will aid system designer to decide

which is the best strategy to apply to the practical condition of future star network configurations.

References

[1] A. Kumar, "Adaptive Load Control of the Central Processor in a Distributed System with a Star Topology", IEEE Trans. Software Eng., Vol. 38, No. 11, pp. 1502-1512, November 1989.

[2] A. N. Tantawi and D. Towsley, "A General Model for Optimal Static Load Balancing in Star Network Configurations", In E. Gelembe, editor, PERFORMANCE '84, Elsevier Science Publisher B. V. (North-Holland), pp. 277-291, 1984.

[3] C.-G. Kim and H. Kameda, "An Algorithm for Optimal Static Load Balancing in Distributed Computer Systems", IEEE Trans. Comp., Vol. 41, No. 3, pp. 381-384, March 1992.

[4] C.-G. Kim and H. Kameda, "Optimal Static Load Balancing of Multi-Class Jobs in a Distributed Computer System", The Trans. of IEICE, Vol. E 73, No. 7 pp. 1207-1214, July 1990.

[5] C.-G. Kim and H. Kameda, "Parametric Analysis of Static Load Balancing of Multi-class jobs in a Distributed Computer System", IEICE Trans. Inf. & Syst., Vol. E75-D, No. 4, pp. 527-534, July 1992.

[6] D. L. Eager, E. D. Lazowska, and J. Zahorjan, "Adaptive Load Sharing in Homogeneous Distributed Systems", IEEE Trans. Software Eng., Vol. SE-12, No. 5, pp. 662-675, May 1986.

[7] L. Kleinrock, Queueing Systems, Computer Applications, Wiley, New York, Vol. 2, 1976.

[8] C.-G. Kim, "Static Load Balancing in a Star-shaped Computer Network", Journal of KISS, Vol. 20, No. 6, pp. 872-879, June 1993.

임 경 수

1984년 경북대학교 공과대학 전자공학과(학사)
1986년 영남대학교 공과대학 전자과 계산기전공(공학 석사)
1993년 영남대학교 공과대학 전자과 계산기전공 박사과정 수료
1991년~현재 연암공업전문대학 전자계산과 조교수
관심분야 : 분산처리시스템, 컴퓨터망 등임

김 수 정

1991년 영남대학교 공과대학 전산공학과(학사)
1994년 영남대학교 공과대학 전산공학과(공학 석사)
1994년 현재 영남대학교 공과대학 전산공학과 박사과정 재학중
관심분야 : 분산처리시스템, 컴퓨터망 등임

김 종 근

1981년 영남대학교 공과대학 전자과(학사)
1987년 영남대학교 공과대학 전자과 계산기전공 (공학 석사)
1991년 (일본) 전기통신대학 정보공학과 소프트웨어 전공(공학박사)
1984년~91년 경북전문대학 전산과 전임
1991년~현재 영남대학교 전산공학과 조교수
관심분야 : 운영체제, 분산처리시스템, 컴퓨터망, 성능평가 등임