DEVELOPING A DECISION SUPPORT SYSTEM FOR DESIGNING DISTRIBUTED DATABASES ON A LOCAL AREA NETWORK

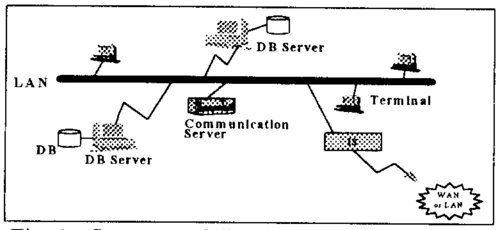
Heeseok Lee and Young-Ki Park
MIS, Graduate School of Management, KAIST, 207-43, Cheongyangri-Dong, Dongdaemoon-Gu, Seoul 130-012, KOREA

Abstract

This paper proposes a design methodology for distributed databases connected by a LAN. Two primary objectives of the methodology are (i) to allocate data files and workload among heterogeneous servers and (ii) to determine the number of servers to satisfy the response time required for processing each transaction. The file and workload allocation decision is formulated as a nonlinear zero-one integer programming problem. This problem is proven to be NP-complete. A heuristic is developed to solve this problem effectively. A decision support system is implemented and an example is solved to illustrate the practical usefulness of the system.

I. Introduction

A local area network (LAN) is an infrastructure which places the company on a secure footing in today's highly competitive business environment. A LAN is a communication network that is confined to a small area, such as a single building or a small cluster of buildings [Stallings, 1993]. Nowadays most organizations have established LANs and used them in various areas of work. Indeed, LANs play an important role in information sharing and office works in business, academia, and industry. Attached to the LANs would be personal computers, database servers providing various data, and communication servers for interacting with remote systems and downloading data to the local databases [Berman and Nigam, 1992]. Figure 1 shows the structure of distributed system on a LAN. An intermediate system (IS) is a device used to connect two subnetworks and permit communication between end systems attached to different subnetworks [Stallings, 1994]. Transactions are generated from terminals and routed to DB server having required data by communication server.

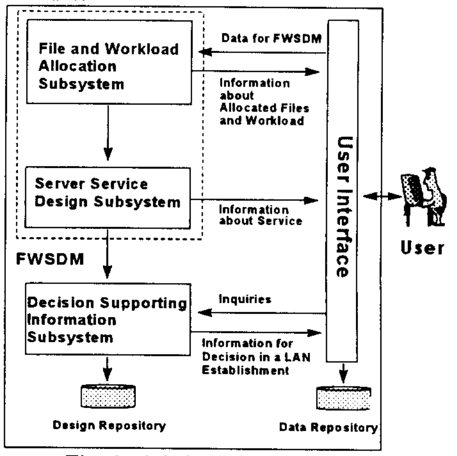


<Fig. 1> Structure of distributed system on a LAN

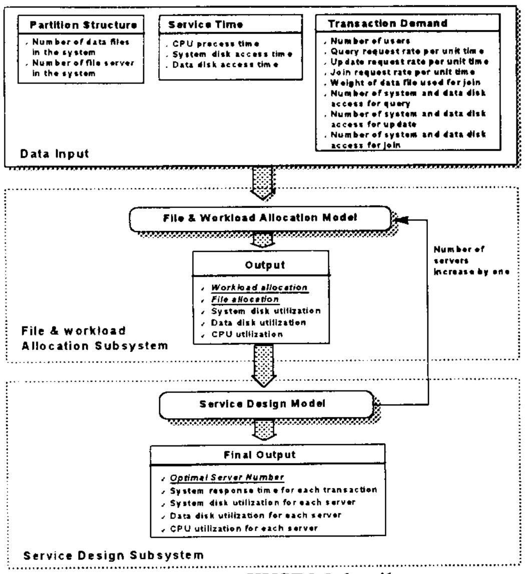
This paper proposes a methodology for determining file and workload allocation simultaneously on a LAN. This problem will be referred to as "file and workload allocation problem" (FWAP). In addition, the methodology determines the number of processing servers to satisfy the response time. Furthermore, a decision support system (DSS) is implemented to demonstrate the practical usefulness of the methodology.

II. Design Methodology

Our methodology consists of three subsystems: a file and workload allocation subsystem, a server service design subsystem and a decision supporting information subsystem. Figure 2 shows its architecture.


File and Workload Allocation Subsystem
Data for FWSDM
Information about Allocated Files and Workload
Server Service Design Subsystem
Information about Service
FWSDM
Decision Supporting Information Subsystem
Inquiries
Information for Decision in a LAN Establishment
User Interface
User
Design Repository
Data Repository

<Fig. 2> Methodology architecture


Partition Structure
. Number of data files in the system
. Number of file server in the system
Service Time
. CPU process time
. System disk access time
. Data disk access time
Transaction Demand
. Number of users
. Query request rate per unit time
. Update request rate per unit time
. Join request rate per unit time
. Weight of data file used for join
. Number of system and data disk access for query
. Number of system and data disk access for update
. Number of system and data disk access for join
Data Input
File & Workload Allocation Model
Output
✓ Workload allocation
✓ File allocation
✓ System disk utilization
✓ Data disk utilization
✓ CPU utilization
Number of servers increase by one
File & workload Allocation Subsystem
Service Design Model
Final Output
✓ Optimal Server Number
✓ System response time for each transaction
✓ System disk utilization for each server
✓ Data disk utilization for each server
✓ CPU utilization for each server
Service Design Subsystem
<Fig. 3> FWSDM details

FWSDM (file, workload and service design model) solves FWAP and determines the optimal number of servers. Figure 3 depicts details of FWSDM. The decision supporting information subsystem provides a user with many types of information for decision in a LAN establishment, for example, the placement of data files, service capacity satisfying the required response time, server utilization, what-if analysis information, etc. The methodology requires two repositories: input data is stored in the data repository and design results are stored in the design repository.

III. Modeling and Solution Procedure

In this paper, FWAP is modeled to minimize the response time for processing transactions. First, we assume that databases were already fragmented according to an affinity-based fragmentation policy. This policy partitions a set of data with the same properties, i.e. , access frequencies [Ceri et. al. 1982]. Thus, transactions with the same properties are routed to the same file server, and locality of data reference is achieved. A nonredundant file allocation policy is adopted. The workload allocation algorithm is static. Static policy is based on the average fluctuation in a long run, while dynamic policy is based on the short-term fluctuation of the system workload [Rahm, 1992].

The following notations are used in the model to present the problem in the form of nonlinear integer programming.

***System parameters*:**

N_s	Number of file servers in the system
N_f	Number of data files in the system
F_i	Data file i
S_k	File server k
λ	Total transaction request rate to the system
f	Total file service request rate to the system
q_i	Query accessing data file i (F_i)
u_i	Update accessing data file i (F_i)
J_S	Join requiring data file set S
f_{q_i}	Query request rate for data file i (F_i) per unit time
f_{u_i}	Update request rate for data file i (F_i) per unit time
f_{J_S}	Join request rate for data file set S per unit time
$w^i_{J_S}$	Weight of data file F_i which is used for join J_S
μ^S_k	System disk access time of server k
μ^D_k	Data disk access time of server k
μ^C_k	CPU process time of server k
S_{q_i}	Number of system disk access for query q_i
S_{u_i}	Number of system disk access for update u_i
S_{J_S}	Number of system disk access for join J_S
D_{q_i}	Number of data disk access for query q_i
D_{u_i}	Number of data disk access for update u_i
D_{J_S}	Number of data disk access for join J_S
N_u	Number of users
ρ^S_k	System disk utilization of sever k
ρ^D_k	Data disk utilization of sever k
ρ^C_k	CPU utilization of sever k
ρ_k	System utilization of sever k
ρ	System utilization
$R^{q_i}_k$	Response time for query i of server k
$R^{u_i}_k$	Response time for update i of server k
$R^{J_S}_k$	Response time for join J_S of server k
R_S	System response time

***Decision Variables*:**

λ_k	Total transaction rate assigned to S_k (i.e. workload allocation)
X_{ik}	Indicator variable of assignment of F_i to S_k (i.e. file allocation)

$$X_{ik} = \begin{cases} 1, & \text{if } F_i \text{ is assigned to the } S_k \\ 0, & \text{otherwise} \end{cases}$$

It is proposed to allocate data file to adequate server in such a way as to minimize system response time. We assume that the queuing system satisfies the conditions for M/M/1. In the M/M/1 queuing system, the utilization of server k is calculated as follows.

For the system disk utilization, the access rate per unit time is

$$\sum_{i=1}^{N_f} \{ (f_{q_i} S_{q_i} N_u + f_{u_i} S_{u_i} + w^i_{J_S} f_{J_S} S_{J_S} N_u) X_{ik} \} \cdot$$

Therefore, the system disk utilization of server k is

$$\rho_k^S = \mu_k^S \sum_{i=1}^{N_f} \{(f_{q_i} S_{q_i} N_u + f_{u_i} S_{u_i} + w^i_{J_s} f_{J_s} S_{J_s} N_u) X_{ik}\}.$$

In this way, the data disk utilization and CPU utilization are calculated as follows respectively.

$$\rho_k^D = \mu_k^D \sum_{i=1}^{N_f} \{(f_{q_i} D_{q_i} N_u + f_{u_i} D_{u_i} + w^i_{J_s} f_{J_s} D_{J_s} N_u) X_{ik}\}$$

$$\rho_k^C = \mu_k^C \sum_{i=1}^{N_f} [\{ f_{q_i} (S_{q_i} + D_{q_i}) N_u + f_{u_i} (S_{u_i} + D_{u_i}) + w^i_{J_s} f_{J_s} (S_{J_s} + D_{J_s}) N_u \} X_{ik}].$$

The system utilization is the summation of disk utilization and CPU utilization; i.e.,

$$\rho_k = \rho_k^S + \rho_k^D + \rho_k^C.$$

Since in a LAN we can ignore the communication overhead compared with I/O load, we can consider that all data files are stored at a single server. This idea allows us to regard a join as a simple query. Therefore, the summation of weights for a join is assumed to be one, and the total file request rate is equal to the total transaction rate; i.e.,

$$\lambda = \sum_{k=1}^{N_s} \lambda_k = \sum_{i=1}^{N_f} (f_{q_i} N_u + f_{u_i} + w^i_{J_s} f_{J_s} N_u) = f.$$

System utilization is a summation of all server utilization; i.e., $\rho = \sum_{k=1}^{N_s} \rho_k$.

Thus, according to queuing theory [Kleinrock, 1976], the system response time is

$$\frac{1}{\lambda} \times \frac{\rho}{1-\rho}.$$

Finally, FWAP is formulated as follows.

Minimize $\frac{1}{\lambda} \times \frac{\rho}{1-\rho}$

Subject to

$$\sum_{k=1}^{N_s} X_{ik} = 1, \ \forall i = 1, 2, \cdots, N_f$$

$$\lambda_k = \sum_{i=1}^{N_f} (f_{q_i} X_{ik} + w^i_{J_s} f_{J_s} X_{ik}) N_u + \sum_{i=1}^{N_f} f_{u_i} X_{ik}, \ \forall k = 1, 2, \cdots, N_s$$

$$\lambda = \sum_{k=1}^{N_s} \lambda_k$$

$$\rho < 1$$

$$\sum_{i=1}^{N_f} w^i_{J_s} = 1$$

$$0 \le w^i_{J_s} \le 1, \forall i = 1, 2, \cdots, N_f$$

$$X_{ik} \in \{0, 1\}, \forall i = 1, 2, \cdots, N_f, \ \forall k = 1, 2, \cdots, N_s$$

Next, we describe a model for designing the appropriate service capacity. The model determines the number of servers that satisfy the required service response time. First, we allocate all data files to a single server having the largest capacity. If any required service response time is not satisfied, allocation is done once again over two servers. Until all of the required service response times are satisfied, server number is increased by one and allocation is done again.

The service response time for each transaction is calculated as follows.

For the simple query routed to server k, the response time is

$$R_k^{q_i} = \frac{\{ S_{q_i} \mu_k^S + D_{q_i} \mu_k^D + (S_{q_i} + D_{q_i}) \mu_k^C \} X_{ik}}{1 - \rho_k},$$

where $S_{q_k} \mu_k^S + D_{q_i} \mu_k^D + (S_{q_i} + D_{q_i}) \mu_k^C$ is processing time for the query.

In this way, the response time for update routed to server k is

$$R_k^{u_i} = \frac{\{ S_{u_i} \mu_k^S + D_{u_i} \mu_k^D + (S_{u_i} + D_{u_i}) \mu_k^C \} X_{ik}}{1 - \rho_k}.$$

For a join, the response time of server k is

$$R_k^{J_s} = \frac{\{ S_{J_s} \mu_k^S + D_{J_s} \mu_k^D + (S_{J_s} + D_{J_s}) \mu_k^C \} \times \left(\sum_{i \in S} w^i_{J_s} X_{ik} \right)}{1 - \rho_k}.$$

Since the service response time for each transaction varies according to the placement of the data file, all of the response times are calculated again, according to the results of each allocation stage. Because of its NP-completeness, a heuristic is developed for solving the problem. We suggest that the heuristic should follow a well known result, *"The system time using a multiple processing system is worse than a single centralized one"* [Lee, 1994]. The following principles are derived from this result.

Principle 1: The data file having the larger request rate is assigned to the file server having the larger capacity.

Principle 2: A data file is allocated to a server in such a way as to balance load over the system.

The heuristic is summarized as follows.

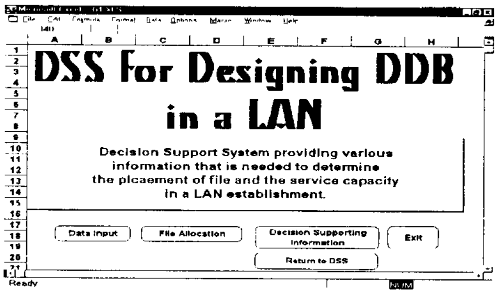
```
    Begin { heuristic }
Sort $f_1 \ge f_2 \ge \cdots \ge f_{N_f}$
$E_k \leftarrow \{\}, \forall k$
for  i = 1, 2, ... , Nf  do
   for  k = 1, 2, ... , Ns  do
      $U_k^S \leftarrow \mu_k^S \sum_{j \in E_k + \{i\}} \{(f_{q_j} S_{q_j} N_u + f_{u_j} S_{u_j} + w^j_{J_s} f_{J_s} S_{J_s} N_u)\}$
      $U_k^D \leftarrow \mu_k^D \sum_{j \in E_k + \{i\}} \{(f_{q_j} D_{q_j} N_u + f_{u_j} D_{u_j} + w^j_{J_s} f_{J_s} D_{J_s} N_u)\}$
$U_k^C \leftarrow \mu_k^C \sum_{j \in E_k + \{i\}} [\{ f_{q_j} (S_{q_j} + D_{q_j}) N_u + f_{u_j} (S_{q_j} + D_{u_j}) + w^j_{J_s} f_{J_s} (S_{J_s} + D_{J_s}) N_u \}]$
$U_k \leftarrow U_k^S + U_k^D + U_k^C$
   if  $U_k < 1$  then  $U_k \leftarrow U_k$  else  $U_k \leftarrow \infty$
   end for
      $k^* \leftarrow \{k \mid \min_k U_k\}$
         {adequate file allocation}
$X_{ik^*} \leftarrow 1$
$E_{k^*} \leftarrow E_{k^*} + \{i\}$
```

```
end for
    $\lambda'_k \leftarrow \sum_{l=1}^{N_f} (f_{q_l} N_u + w^l_{js} f_{js} N_u + f_{u_l}) X'_{lk}$   , $\forall k$
    {adequate workload allocation}
end { heuristic }
```

IV. DSS Implementation

To demonstrate the practical usefulness of our methodology, a DSS is implemented on an IBM PC by the use of Microsoft Excel [Nelson, 1992] and Visual Basic [Jennings, 1994]. This system consists of a data input subsystem, a data file allocation subsystem and a decision-supporting information subsystem. The example in the previous section is used to explain this system.

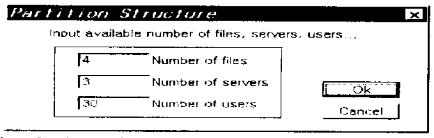
Figure 4 shows the main screen that presents three main subsystems.


DSS For Designing DDB in a LAN

Decision Support System providing various information that is needed to determine the plcaement of file and the service capacity in a LAN establishment.

<Fig. 4> Main screen

We start DSS by inputting data that are parameter values for the file allocation model (Fig. 5).

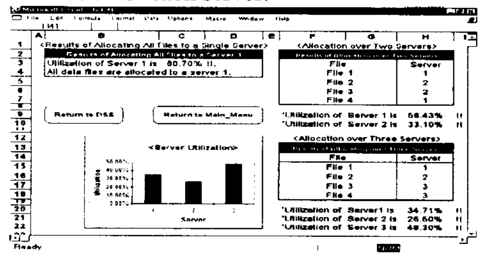

Partition Structure

Input available number of files, servers, users ...

4	Number of files
3	Number of servers
30	Number of users

<Fig. 5> Data input screen for partition structure

After the data input stage is completed, data files are allocated over available servers. Clicking on the 'File Allocation' button starts the data file allocation. Figure 6 shows the results of the file allocation. It shows the placement of each file and the utilization of each server.


<Results of Allocating All files to a Single Server>

Results of Allocating All Files to a Server 1

Utilization of Server 1 is 80.70% !!.
All data files are allocated to a server 1.

<Allocation over Two Servers>

File	Server
File 1	1
File 2	2
File 3	2
File 4	1

*Utilization of Server 1 is 66.43% !!
*Utilization of Server 2 is 33.10% !!

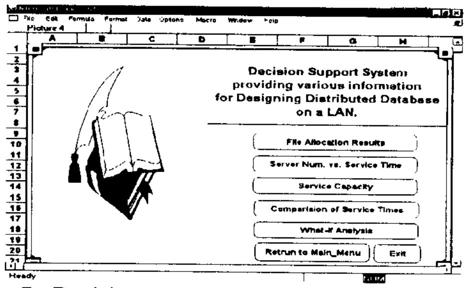
<Allocation over Three Servers>

File	Server
File 1	1
File 2	2
File 3	3
File 4	3

*Utilization of Server1 is 34.71% !!
*Utilization of Server 2 is 26.60% !!
*Utilization of Server 3 is 48.30% !!

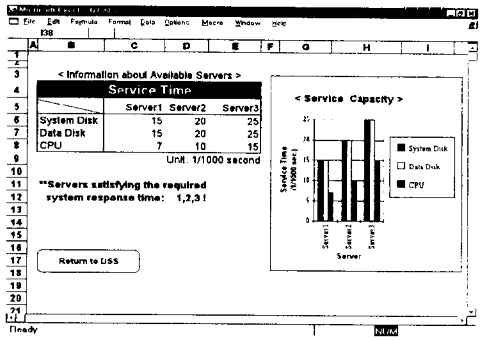
<Fig. 6> The results of file allocation

Our system provides many types of information needed for LAN establishment. These are highlighted in Figure 7.


Decision Support System providing various information for Designing Distributed Database on a LAN.

<Fig. 7> Decision supporting information main screen

The service capacity satisfying the required service response time for each transaction is shown in Figure 8, which also shows the comparison among server capacities.


< Information about Available Servers >

Service Time	Server1	Server2	Server3
System Disk	15	20	25
Data Disk	15	20	25
CPU	7	10	15

Unit: 1/1000 second

**Servers satisfying the required system response time: 1,2,3 !

<Fig. 8> Screen for service capacity determination

V. Conclusions

In a LAN establishment, the determination of file and workload allocation, and service capacities, is an important problem. This paper proposes a methodology for solving this problem. A heuristic is developed to solve the resulting NP-complete problem. A decision support system is implemented, which provides an interactive design capability. This feature is important because the decision problem is very complex. The system can help the user better design distributed databases in a LAN.

Note: Contact the author for the full paper including references.