유전적 알고리즘을 이용한 다목적 분산데이터베이스 설계

이 재 욱[†] · 고 석 범[††] · 조 정 복[†] · Mitsuo Geo[†††]

요 약

최근, 정보네트워크의 놀랄만한 확장과 함께 분산데이터베이스가 부가통신망(Value Added Network)상에서 구현되는 사례가 늘고 있다. 분산데이터베이스는 지역적으로 분산된 업무 환경에서 중앙집중식 구조에 비해 비용과 응답시간 면에서 큰 장점을 가진다. 그러나, 부적합한 설계는 불필요한 비용과 늦은 응답시간을 초래하게된다. 분산데이터베이스 설계에서의 주요한 문제는 각 노드에서의 1) 적합한 컴퓨터의 선택과 2) 단편화된 데이터를 적합하게 할당하는 것이다. 따라서, 본 논문은 부가통신망상에서의 최적인 컴퓨터의 선택과 데이터의 할당에 관하여 논한다. 또한, 공식화된 수학 모델은 1) 운용비용과 2) 투자비용으로서 두 개의 목적함수를 포함하고 경험적 탐색법 중의 하나인 유전적 알고리즘의 설계를 통해 최적인 분산데이터베이스 설계를 위한 해들을 탐색한다. 끝으로 수치예를 통해 각 성능을 평가할 것이다.

Multiobjective Distributed Database System Design using Genetic Algorithms

Jae-Uk Lee[†] · Suc-Bum Ko[††] · Jung-Bok Jo[†] · Mitsuo Gen[†††]

ABSTRACT

Recently, DDS (Distributed Database System) has been often implemented on VAN (Value Added Network) as we know the amazing expansion of information network. DDS can yield significant cost and response time advantages over centralized systems for geographically distributed organizations. However, inappropriate design can result in high cost and poor response time. In a DDS design, the main problem is 1) how to select proper computer, and 2) how to allocate data fragment into proper nodes. This paper addresses DDS design problem of selecting the proper class of computers and the allocating data files on VAN. Also, the formulated model includes two objectives, the operating and investment cost. GA (Genetic Algorithm) is developed to solve this mathematical formulation. A numerical experiment shows that the proposed method arrives at a good solution.

1. 서 론

Recently, DDS has been often implemented on VAN due to the amazing increase of Internet users. In DDS design, the main problem is 1) how to select a class of computers and 2) how to allocate data fragment into proper nodes. This research considers the operating cost and investment cost which have interrelation an optimization problem. The problem of DDS design is very difficult task because the network flow problem, the data allo-

† 정 회 원 : 동서대학교 컴퓨터공학과 교수
†† 정 회 원 : 부경대학교 대학원 전자계산학과
††† 비 회 원 : 일본 아시카가공업대학교 경영공학과 교수
논문접수 : 1998년 10월 22일, 심사완료 : 1999년 2월 25일

cation, and the computer location problem must be considered simultaneously.

The mathematical formulations which have been developed by many researchers [2, 3] in the past did not consider the difference in the cost of processing a transaction on the different classes of computers. Firstly, Mitrani and Sevick [8] and Buhr [7] addressed the problem of selecting the optimal speed of processors. Several studies have developed methods to minimize some measure of system cost as a function of database file locations [9, 10]. Another class of models has been developed for the design of communication network topology and allocation of data files[4, 11, 14, 15]. Dutta and Jain[12] addressed the combined problem of the processor selection, the file allocation, and the network design. Recently, March and Rho[1] addressed the data allocation problem as well as the operation allocation problem on the network which consists of the same grade computers.

In this research, the main purpose is to design the DDS model on the network which consists of five classes of computers. Also, we design the GA as a main solution algorithm to search the best compromised solution[5].

The main reason of using GA for solving the proposed mathematical model can be summarized by two points. Firstly, GA is powerful to solve a hard combinatorial problem such as the proposed mathematical model. Secondly, GA which is a kind of heuristic search method can produce a solution whenever the solution for the given problem is required. For the advantage, we will mention in section 4.

2. Formulation of DDS

We design DDS to be considered the data allocation and the computer location problem with two objectives, 1) the operating cost and 2) the investment cost.

The following notations will be used in describing the model :

S : A set of all network nodes
K : A set of all data files
M : A set of all classes of computers available
C : A class of jobs, represents the lowest class of computers on which it can be processed.

In this paper, each computer is classified as 1) personal computer (PC), 2) workstation (WS), 3) main frame (MF), 4) large main frame (LMF), and 5) super computer (SC) for the set M.

The parameters to formulate a mathematical model are defined as follows :

v_k : the size of file k in kilobytes.
λ_s : the arrival rate of jobs at node s.
b_{sc} : the probability that the job arriving at node s is a job in class c.
q_{ck} : the probability that a job in class c will result in updating file k.
p_{ck} : the probability that a job in class c will query file k.
o_{ck} : the average length of data transmission required to update a copy of file k by a job in class c.
t_{ck} : the average length of data transmission required to satisfy a query from file k by a job in class c.
e_c : the average primary storage required for job in class c.
g_c : the average length of job in class c.

This model has two decision variables. One implies the location of the classified computers to a proper node and the other implies the allocation of the data files to each node. It needs to notice that the duplicated allocation of data file is allowed. Each decision variables are shown as :

$$x_{ms}=\begin{cases}1; & \text{if the computer in class } m \\ & \text{is assigned to node } s. \\ 0; & \text{otherwise.}\end{cases}$$

$$y_{sk}=\begin{cases}1; & \text{if } k\text{th data file is located} \\ & \text{at the node } s. \\ 0; & \text{otherwise.}\end{cases}$$

2.1 Objective Functions

2.1.1 Operating Cost

The objective for operating cost includes two kinds of costs, 1) the transactions processing cost and 2) the transmission cost.

Specially, we must consider the transmission cost due to the following aspects :

a) Due to transactions executed at a remote node.

b) Due to updating files at a remote node.

c) Due to a query requiring data from files stored at a remote node.

The new parameters defined to formulate the total operating cost are shown as follows :

cp_m :	the operating cost of processing a machine instruction on a computer in class m.
β_m :	the average number of machine instructions executed for a transaction on a computer in class m.
α_c :	the average transmission required to remotely execute a job in class c.
b'_{sc} :	the total expected number of the class of job c processed at a node s per unit time.
c_T :	the data file transmission cost for unit time.

Then, the total operating cost per unit time $z_{ope}(\boldsymbol{x}, \boldsymbol{y})$ can be expressed as follows :

$$\begin{aligned} z_{ope}(\boldsymbol{x}, \boldsymbol{y}) = & \sum_{s \in S} \sum_{m \in M} \beta_m \, l_s \, cp_m \, x_{ms} \\ & + c_T \Big[\sum_{s \in S} \sum_{c=m+1}^{C} b_{sc} \alpha_c x_{ms} \\ & + \sum_{s \in S} \sum_{k \in K} \Big[\sum_{c \in C} b'_{sc} t_{ck} p_{ck} \\ & + \{ \sum_{c \in C} b'_{sc} (o_{ck} q_{ck} - t_{ck} p_{ck}) \} \, y_{sk} \Big] \Big] \end{aligned} \tag{1}$$

where the first term is to calculate the expected cost to process transactions and the second term to the last one is to calculate the total transmission cost per unit time according to three aspects.

Also, the average length of jobs processed at node s per unit time l_s is computed as :

$$l_s = \sum_{c \in C} b'_{sc} g_c$$

To calculate b'_{sc}, we must know where each job class is processed. It means that if the job class is less than or equal to the class of computer at the node, local processing is performed; otherwise, it is processed at a remote node.

In other words, a node with a computer to be included as high class, such as super computer, can perform any jobs locally, but a node with a computer to be included as low class, such as personal computer, has some limitation to process a job. So, the job should be processed at another remote node when the class of arriving job is higher than the class of computer.

2.1.2 Investment Cost

In the problem, the investment cost is calculated with the sum of the computer setup cost located in each node.

The expression of investment cost is :

$$\begin{aligned} z_{inv}(\boldsymbol{x}) = & \sum_{s \in S} \sum_{m \in M} (pp_m + pfm_m \cdot pm_m \\ & + psm_m \cdot sm_m) \cdot x_{ms} \end{aligned} \tag{2}$$

where, pp_m is the processor cost for the computer in class m, pm_m and sm_m represent the average cost of the primary storage capacity and the secondary storage capacity for the computer in class m, pfm_m and psm_m are the maximum avaliable size of the primary storage capacity and the secondary storage capacity for the computer in class m, respectively.

2.2 Constraints

Five constraints to ensure a feasible solution are described as follows :

1) Only one class of computer will be located at a node. There may be some nodes with no computers :

$$\sum_{m \in M} x_{ms} \leq 1 \quad \text{for} \quad s \in S \tag{3}$$

2) Primary storage capacity at a node should be sufficient to process arriving transactions :

$$\sum_{m \in M} pfm_m x_{ms} \geq \sum_{c \in C} b'_{sc} e_c \qquad (4)$$

3) Secondary storage capacity at each node should be sufficient to store all the copies of data file assigned to the node:

$$\sum_{m \in M} psm_m x_{ms} \geq \sum_{k \in K} y_{sk} v_k \quad \text{for} \quad s \in S \qquad (5)$$

4) Each database file must be allocated to at least one node, while multiple copies of a file are allowed:

$$\sum_{s \in S} y_{sk} \geq 1 \qquad \text{for } k \in K \qquad (6)$$

5) Whenever a node has some file/database allocated to it, there will be a computer allocated there along with the storage facility. In other words, if a node has storage capacity, it must have processing capacity:

$$\sum_{m \in M} x_{ms} \geq \min\{1, \sum_{k \in K} y_{sk}\} \quad \text{for } s \in S \qquad (7)$$

3. GA Implementation

3.1 Chromosome Representation

We design the chromosome representation to encode a solution of the proposed mathematical model. Allocation of the class of computer and location of data files are represented as Table 1. Also, Table 2 shows five kinds of computers to be located to each node in this research.

〈Table 1〉 Chromosome Representation

Computer Class	Allocated Files	0

〈Table 2〉 Classification Table for Computers

Class Number	Computer Class
1	Personal Computer (PC)
2	Workstation (WS)
3	Main Frame (MF)
4	Large Main Frame (LMF)
5	Super Computer (SC)

We did not use the matrix based chromosome representation for saving the memory spaces to be required. In Table 1, each locus represents each node. For the first location, a memory to store only one script is required while the memory for the second allocation is required as many as the number of allocated data files. The last field, 0, is to identify the end of node because 0 is not generated absolutely for all of cases.

In Figure 5, the chromosome, [174860 4768143250 11740 25361720 180 11740], means that PC, class 1, is located in node 1 and the data files 7, 4, 8, and 6 are also allocated into node 1. At the node 2, LMF, class 4, is located and the data files 7, 6, 8, 1, 4, 3, 2, and 5 are allocated.

3.2 Initialization

Basically, we must ensure the basic constraints, equations (3) and (6) for new chromosomes, *i.e.* they must be satisfied whenever chromosome is changed. Constraint (3) can be easily ensured by giving a random number between 1 and the number of classes of computer. But, for constraint (6), some check procedure is needed to prevent the duplicated allocation in the same node.

3.3 Genetic Operators

3.3.1 Crossover

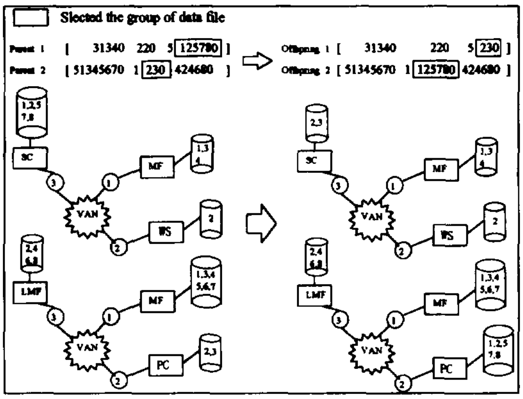
In the proposed crossover, we concentrated on the insurance of chromosome feasibility which is not to violate the basic constraints. For two selected nodes, only the genes represented a data file are partially exchanged each other. We give Figure 1 to understand the designed crossover operation in detail.

In the crossover, there is a possibility to search a narrow solution space. So, it needs to change a chromosome to broaden a searching space of a solution.

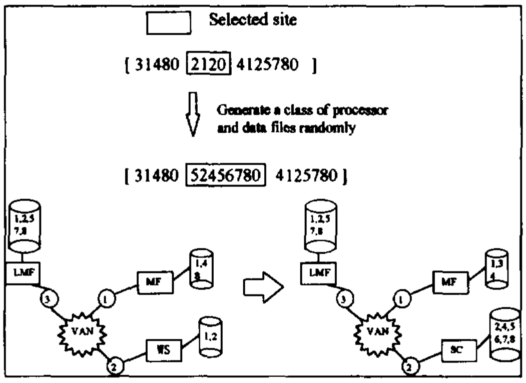
3.3.2 Mutation

In the proposed mutation, all genes in the selected node are regenerated by the creation rule shown in section 3.2. There is an example in Figure 2.

Obviously, the above genetic operators, crossover and mutation, do not yield an infeasible chromosome which violates the basic constraints.


Slected the group of data file
Parent 1 [31340 220 5 125780 1]
Parent 2 [51345670 1 230 424680]
Offspring 1 [31340 220 5 230]
Offspring 2 [51345670 1 125780 424680]
SC
MF
VAN
WS
LMF
PC

(Fig. 1) Partial Exchange Crossover



(Fig. 2) Rebuilding Mutation

3.4 Evaluation and Selection

We used the expression of Pareto optimal solution to solve our problem which consists of two objectives.

In the multicriteria optimization context, usually the Pareto optimal solutions are characterized as the solutions of the multiobjective programming problem [5, 6].

To evaluate two objective functions, we used the weighted sums method [5] to construct the evaluation function as follows :

Step 1 : At the t-th generation choose the solution points as follows :

$$z_q^{\min(t)} = \min_k \{ z_q^{\min(t-1)}, z_q^{(t)}(T_k) | k=1,2,\cdots, pop\_size \},$$
$$q=1,2$$

$$z_q^{\max(t)} = \max_k \{ z_q^{\max(t-1)}, z_q^{(t)}(T_k) | k=1,2,\cdots, pop\_size \},$$
$$q=1,2$$

where, $z_q^{\max(t)}$ and $z_q^{\min(t)}$ denote the maximum and minimum value of objective function q at generation t, respectively.

Step 2 : Solve the following equations to get weights for the evaluation function.

$$\delta_q = | z_q^{\max(t)} - z_q^{\min(t)} | , \qquad q=1,2$$

Step 3 : Calculate the value of evaluation function for each chromosome as follows if the solution is feasible :

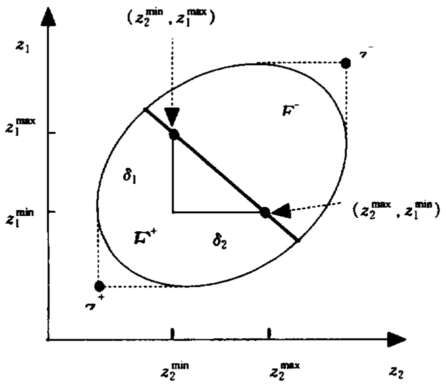
$$eval(T_k) = \frac{\delta_2 z_1(T_k) + \delta_1 z_2(T_k)}{\delta_1 + \delta_2}, \quad k=1,2,\cdots, pop\_size \tag{8}$$

Figures 3 gives an illustrative explanation of the objective.

The line formed with points $(z_2^{\min}, z_1^{\max})$ and $(z_2^{\max}, z_1^{\min})$ divides the criteria space into two half spaces : one containing the positive ideal solution and another containing the negative ideal solution. The feasible solution space F is correspondingly divided into two parts : F- and F+. It is easy to verify that all Pareto solutions lie within F+. At each generation, Pareto set E is updated and the two special points may be renewed. It means that along with the evolutionary process, the line formed with this two points will adapt gradually towards to Pareto frontier.

In the other words, this fitness function gives such selection pressure to force genetic search to exploit the nondominated points in the criteria space. The equation (8) can be easily generalized to n dimensional space to construct an adaptive hyperplane in the criterion space so as to force genetic search towards to exploiting the set of nondominated points.

To select the best chromosome for the next generation, we used the **elitist selection** that sort them in descending order as each value of evaluation function, equation (8), and then select the first chromosome as the new population.



(Fig. 3) Illustrative Explanation of Adaptive Line

4. Overall Algorithm

Let $P(t)$ is a population of chromosomes for iteration t and $C(t)$ is the generated chromosomes at iteration t. The overall procedure is summarized as follows :

```
Procedure for DDS :
begin
  $t \leftarrow 0$;
  initialize $P(t)$;
  evaluate $P(t)$;
  while ( $t < max\_gen$) do
     recombine $P(t)$ to generate $C(t)$;
     evaluate $C(t)$;
     select $P(t+1)$ from $P(t)$ and $C(t)$;
     $t \leftarrow t+1$;
  endwile
end
```

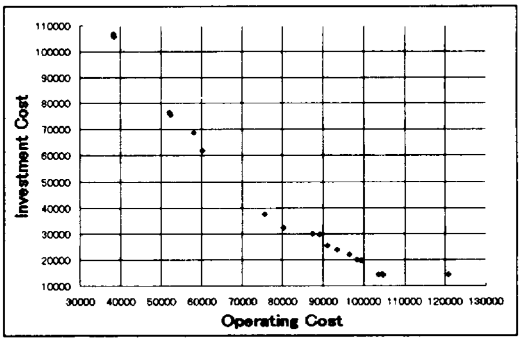
where, *max_gen* means the maximum generation number and t is needed to count each generation.

5. Numerical Example

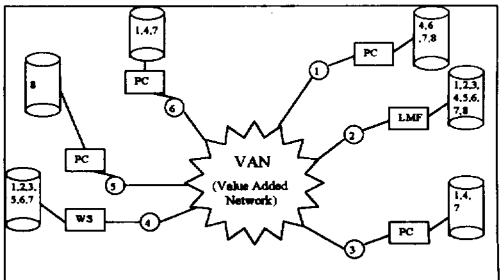
We deal with the problem which includes 6 nodes, 5 classes of computers and jobs, and 8 data files.

We executed GA on the evolutionary environment, $p_c = 0.3$, $p_m = 0.2$, $pop\_size = 20$, $max\_gen = 1000$. Here, p_c and p_m denote the crossover rate and the mutation rate, respectively. *pop_size* denotes the population size in each generation.

The Pareto solutions are shown in Figure 4. As shown in Figure 4, there is interrelation between the operating cost and investment one. In the face of using a superior computer, the network transfer cost can be saved due to the possibility of local process and the enough secondary memory to save many data files for user query originated in the node. But an expensive cost to set up a superior computer is required, simultaneously.



(Fig. 4) Pareto Solutions Obtained by GA



Chromosome : [174860 4768143250 11740 25361720 180 11740]
Operating Cost : 60104.3
Investment Cost : 61742.3

(Fig. 5) Illustrative Overall DDS

Figure 5 illustrates the overall DDS for the best compromise solution among the Pareto solutions, using the TOPSIS method. The TOPSIS method proposed by Yoon and Hwang[13] stands for technique for order preference by similarity to ideal solution, which is based on the concept that the chosen alternative should have the shortest distance from the positive ideal solution and the farthest from the negative ideal solution. For the detail information about the TOPSIS to determine the best compromise solution among obtained Pareto solution, see the paper[5, 13].

6. Conclusions

In this research, we dealt with the DDS design problem of selecting the classified computer and allocating the data file fragment while considering interrelation between the operating cost and the investment one.

In the proposed GA, we gained the efficiencies which can save a memory space and are able to check a violence of constraints to easier than the matrix based chromosome representation. As shown in the numerical example, the GA can find the best compromised solution.

In future research, the proposed model will be extended by more comprehensive model which considers other important factors such as the weighted response time and the network reliability.

Reference

[1] S.T. March and S.K. Rho, "Allocating Data and Operations to Nodes in Distributed Database Design," IEEE Trans. on Knowledge and Data Engg., Vol.7, No.2, pp.305-317, Apr. 1995.

[2] M. Ozsu and P. Valduriez, 'Principles of Distributed Database Systems', Prentice-Hall Inc., Englewood Cliffs; N.J. 1991.

[3] S. Ram and R. E. Marsten, "A Model for Database Allocation Incorporating a Concurrency Control Mechanism," IEEE Trans. on Knowledge Data Eng., Vol.3, pp.389-395, Sep.1991.

[4] H.K. Jain, "A Comprehensive Model for the Design of Distributed Computer Systems," IEEE Trans. on Software Eng., Vol.SE-13, pp.1092-1104, Oct. 1987.

[5] M. Gen and R. W. Cheng, 'Genetic Algorithms and Engineering Design', John Wiley and Sons, New York, 1997.

[6] Z. Michalewicz, 'Genetic Algorithms + Data Structures = Evolution Programs', second edition, Springer - Verlag, New York, 1994.

[7] R. J. A. Buhr and C. M. "Woodwide, Microscopic Economic Planning Models for Distributed Information Systems," INFOR, Vol.15, No.2, 1977.

[8] I. Mitrani and Sevcik, "Evaluating the Trade-off between Centralized and Distributed Computing," Proc. 1st Int. Conf. Distributed Computing Systems, pp.520-528, Oct. 1979.

[9] W. W. Chu, "Optimal File Allocation in Multiple Computer Systems," IEEE Trans. Comput., Vol. C-18, pp.885-889, 1969.

[10] S. Mahmoud and J. S. Riordan, "Optimal Allocation of Resources in Distributed Information Networks," ACM Trans. on Database System, Vol.1, No.1, pp.66-78, 1976.

[11] J. P. Ignizio, D. F. Palmer, and C. Murphy, "A Multicriteria Approach to Super System Architecture Definition," IEEE Trans. on Comput., Vol. C-31, pp.410-418, May 1982.

[12] A. Dutta and H. Jain, "A DSS for Distributed Computer System Design in the Presence of Multiple Conflicting Objectives," Decision Support System, Vol.1, No.3, pp.233-246, Sept. 1985.

[13] C. Hwang and K. Yoon, 'Multiple Attribute Decision Making Methods and Applications', Springer -Verlag, Berlin, 1981.

[14] M. Gen, Y. Tsujimura and S.B. Ko, "Allocation Strategy for Distributed Database System with Fuzzy Data Using Genetic Algorithm," 5th European Congress on Intelligent Techniques and Soft Computing (EUFIT'97), Vol.1, pp.737-742, Sep. 1997.

[15] S.B. Ko, Y. Tujumura and M. Gen, "Data Distribution Considered Communication Flow in Net-

work Using Genetic Algorithm," 2nd Inter. Conf. on Knowledge-Based Intelligent Electronic Systems, Vol.2, pp.264-271, Apr. 1998.

이 재 욱

e-mail : julee@kowon.dongseo.ac.kr

1975년 경북대학교 전자공학과(학사)

1981년 동아대학교 전자공학과(공학석사)

1989년 동아대학교 대학원 전자공학과(공학박사)

1987년～1988년 일본 아시카가공대 컴퓨터시스템 연구실 객원교수

1992년～현재 동서대학교 컴퓨터공학과 부교수

관심분야 : 신뢰성공학, 퍼지이론, 유전자 알고리즘

고 석 범

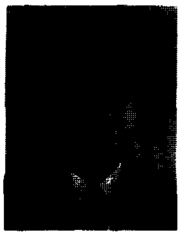
e-mail : sbko@dol.pknu.ac.kr

1996년 동서대학교 퓨터공학과 졸업(학사)

1998년 일본 아시카가공대 대학원 경영공학과(공학석사)

1999년～현재 부경대학교 대학원 전자계산학과 박사과정

관심분야 : 분산데이터베이스, 멀티미디어 데이터베이스, 유전자 알고리즘



조 정 복

e-mail : jobok@kowon.dongseo.ac.kr

1978년 경북대학교 전자계산기공학과(학사)

1986년 영남대학교 대학원 전자공학과(공학석사)

1996년 일본 동경도립과학기술대학교 대학원 연구공학과(공학박사)

1995년～현재 동서대학교 컴퓨터공학과 교수

관심분야 : 신경망이론, 퍼지대기행렬

Mitsuo Gen

e-mail : gen@genlab.ashitech.ac.jp

1969년 일본 공학원대학교 전자공학과(학사)

1971년 일본 공학원대학교 대학원 전자공학과(공학석사)

1975년 일본 공학원대학교 대학원 전자공학과(공학박사)

1973년～현재 일본 아시카가공업대학교 경영공학과 교수

1973년～현재 Computers and Industrial Engineering, System Engineering and Automation 국제저널 편집위원

관심분야 : 퍼지 다목적 프로그래밍, 유전자 알고리즘, 목표계획법