Using a Genetic Algorithm for Communication Link Partitioning

Ju-Hyun Lee Yanghee Choi Byoung-Tak Zhang Chong Sang Kim
Dept. of Computer Engineering, Seoul National Univ., Seoul 151-742, Korea
E-mail: jhlee@archi.snu.ac.kr

Abstract— **This paper addresses two instances of link partitioning problems in communication networks: dedicated and shared partition allocation problems. These problems belong to the class of nonlinear nondifferentiable integer optimization problems which are difficult for conventional nonlinear integer programming methods to find global optima. In this paper we use a genetic algorithm for solving these problems. Possible partitions of a communication link are represented as chromosomes to which genetic operators are repeatedly applied to find better solutions. Comparative experimental results show that the genetic method outperforms uniform partitioning and a conventional heuristic method for a wide range of offered load levels in multiclass calls.**

I. Introduction

Genetic algorithms (GA) are search methods inspired by natural evolution and population genetics [1], [2]. Due to their simplicity and generality, genetic algorithms are widely applied to global optimization problems. Recently, genetic algorithms have also been used in telecommunications. Davis and Coombs [3] applied a genetic algorithm to determine link capacity in packet networks. Palmer and Kershenbaum [4] described a genetic approach to finding optimal communication spanning trees. In this paper, we use a genetic algorithm for solving link partitioning problems in communication networks.

In communication networks which support multiclass calls, such as ATM (asynchronous transfer mode) networks [5], a link is partitioned to service multiclass calls, since link partitioning allows a simple call admission control and leads to cost reduction in call switching and network control.

In this paper, we consider two link partitioning problems, a dedicated partition allocation problem (DPAP) and a shared partition allocation problem (SPAP). DPAP is a kind of link partitioning problem in which the cost of incoming call blocking is to be minimized when the link is divided into dedicated partitions only. In SPAP, the goal is to minimize the cost of incoming call blocking when the link is divided into shared and dedicated partitions.

A heuristic method for solving the SPAP problem was proposed in [6]. In this heuristic, the size of the shared partition is kept fixed. The remaining link capacity is divided among n classes such that the size of dedicated partition of each class is proportional to the incoming rate and duration time of the corresponding call class. Although this heuristic finds a good solution for a heavy load, it does not operate well for a light load of calls. In this paper, we present a GA approach for the link partitioning problems and compare its results with those obtained by the heuristic method.

The paper is organized as follows. In Section II, the link partitioning problems are defined more formally. Section III describes a genetic algorithm for solving these problems. Section IV compares the GA solutions with the results obtained by uniform partitioning and the heuristic method. Section V provides the conclusions and future work.

II. Problem Formulation

In this section, we formulate link partitioning problems. In formulating the problems, the followings are assumed:

- Calls arrive to a link from n classes, with a class i call requiring integer bandwidth b_i, $i = 1, ..., n$.
- Each call is modeled as a Poisson process and the holding time of each call is exponentially distributed.
- All calls within a class have the same call characteristics.
- Without loss of generality, the classes are ordered so that $1 = b_1 \leq b_2 \leq ... \leq b_n$.

Variables to describe problems are defined as follows.

- a_i : offered traffic of class i
- b_i : bandwidth of i-th class call
- c_i : number of calls to be allocated to i-th class
- k_i : relative bandwidth of a i-th class call to unit bandwidth, that is, $k_i = b_i/b_1$
- C_{sh} : number of class 1 calls that can be allocated by shared capacity
- C : link capacity
- $(n_1,...,n_k)$: state that denotes the number n_i of calls for class i, $i = 1,..,k$, in shared partition.
- B(a_i,c_i) : blocking probability of i-th class calls, i.e.

$$B(a,c) = \frac{a^c/c!}{\sum_{n=0}^{c} a^n/n!}. \qquad (1)$$

Eqn. (1) is known as the Erlang B formula and used widely in evaluating the blocking probability of telephone networks [8].

A. The Dedicated Partition Allocation Problem (DPAP)

Given characteristics, such as call incoming rate and call duration time, of multiclass calls, the dedicated partition allocation problem is to find a way to divide the limited link into n dedicated partitions to minimize the cost of incoming call blockings. The cost due to blocking an incoming call is proportional to offered load, bandwidth, and blocking probability of the call. Thus, the problem can be formulated as:

$$LP1 = \min_{c_i} \sum_{i=1}^{n} a_i b_i B(a_i, c_i) \qquad (2)$$

subject to the link capacity constraint

$$\sum_{i=1}^{n} c_i b_i \leq C. \qquad (3)$$

Here c_i are nonnegative integers. Eqn. (3) specifies that the total bandwidth of all classes should not exceed the link capacity.

B. The Shared Partition Allocation Problem (SPAP)

The shared partition allocation problem (SPAP) is formulated as follows: given characteristics of n class calls, find a way to divide the limited link into dedicated and shared partitions to minimize the cost of incoming call blockings. The shared partition is used for the incoming calls which were blocked at their dedicated partition.

The probability that the shared partition has n_i calls for class i, $i = 1, \cdots, k$, is defined as follows [8]:

$$P(n_1, .., n_k) = \frac{\frac{a_1^{n_1}}{n_1!} \cdots \frac{a_k^{n_k}}{n_k!}}{\sum_{(n_1,..,n_k) \in S} \frac{a_1^{n_1}}{n_1!} \cdots \frac{a_k^{n_k}}{n_k!}} \qquad (4)$$

where S is the set of nonblocking states.

B_i^* is the probability that the total bandwidth of incoming calls and all calls using the shared partition exceed the size of the shared partition. Using Eqn. (4), we can define B_i^* as:

$$B_j^* = \frac{\sum_{(n_1,..,n_k) | C_{sh} - k_j + 1 \leq \sum k_i n_i \leq C_{sh}} P(n_1, .., n_k)}{\sum_{(n_1,..,n_k) | 0 \leq \sum k_i n_i \leq C_{sh}} P(n_1, .., n_k)} \qquad (5)$$

The cost by blocking an incoming call is proportional to offered load, bandwidth, and blocking probability for a shared partition. Thus the problem is formulated as follows:

$$LP2 = \min_{c_i} \sum_{i=1}^{n} a_i b_i B_i^*(a_i, c_i) \qquad (6)$$

subject to

$$\sum_i c_i b_i + C_{sh} \leq C, \qquad (7)$$

where c_i and C_{sh} are nonnegative integers, respectively. Eqn. (7) specifies that the sum of total bandwidth of all classes and the shared capacity should not exceed the link capacity.

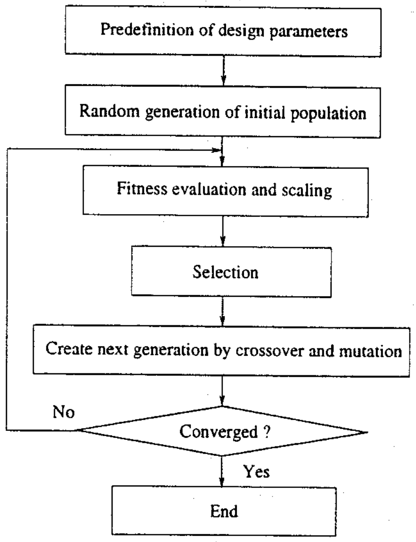
Predefinition of design parameters
Random generation of initial population
Fitness evaluation and scaling
Selection
Create next generation by crossover and mutation
No
Converged ?
Yes
End

Fig. 1. *Algorithm Summary*

III. A Genetic Algorithm for Link Partitioning

The problems described in the previous section are a nonlinear nondifferentiable integer optimization problem. Existing methods have difficulty in finding global optima for this kind of problems. In this section we present a genetic algorithm approach to solving the two link partitioning problems.

Genetic algorithms[7] are search methods based on natural evolution and genetics. Starting from a population of individuals generated at random, a genetic algorithm repeatedly evolves populations of fitter individuals. Genetic algorithms use binary strings to represent possible solutions, which is especially effective in solving optimization problems whose optimal solutions take integer or discrete values. Genetic algorithms are population-based search techniques and thus have a higher probability of finding global optima than point-based search techniques. Constraints such as continuity or differentiability of the search space is not required in genetic search. Due to the simplicity and generality of basic operations, genetic algorithms have been applied to a wide range of problems.

In using a genetic algorithm to solve the link partitioning problems, each value c_i representing the number of calls of class i is encoded as a binary string. Then, an indiviudal A is represented as a concatenation of n components:

$$A = (c_1, c_2, ..., c_n). \qquad (8)$$

The fitness of each individual is evaluated by first decoding the binary values into the corresponding integers and then inserting the values to the equation $LP1$ (Eqn. 2). In order to encourage the link capapcity constraint condition $\sum_i c_i b_i \leq C$ (Eqn. 3) to be satisfied, we add a penalty term $\{\min(C - \sum_i c_i b_i, 0)\}^2$ which has a positive value if

$C - \sum_i c_i b_i$ is nonzero. This leads to the fitness function:

$$F = LP1 + r \cdot \{\min(C - \sum_i c_i b_i, 0)\}^2 \quad (9)$$

where r is the penalty factor of positive constant. Similarly, the fitness of the individuals for problem $LP2$ is defined as:

$$F = LP2 + r \cdot \{\min(C - \sum_i c_i b_i - C_{sh}, 0)\}^2 \quad (10)$$

where $LP2$ is as defined in Eqn. (6) and C_{sh} is the capacity of the shared link.

The procedure for the proposed genetic algorithm is as follows (see also Fig. 1):

1. Initialization: A population of M individuals is created by randomly generating the bitstrings.
2. Fitness evaluation: The raw fitness of individuals is evaluated using either equation (2) or (6), depending on the problem. The raw fitness values are scaled to take a positive value within a defined interval: $F' = aF + b$ where a and b are constants.
3. Selection: Fitter individuals are selected to reproduce with higher probability than less fit individuals.
4. Creation: Genetic operators are used to produce the next generation of individuals. We use two-point crossover and mutation.
5. Termination condition: The steps 2 to 4 are repeated until the termination condition (in terms of specified number of evaluations or generations) are satisfied.

IV. Experimental Results

The effectiveness of the genetic algorithm was tested for the two link partitioning problems. The parameters used for GA were population size = 50, crossover rate = 0.9, and mutation rate = 0.01. We assumed that there are only two classes in a link and that calls arrived from two classes with bandwidth requirements $b_1 = 1$, $b_2 = 32$, and C = 1024. The results of the GA solutions are compared with those obtained by the heuristic method. For the purpose of further comparison, we also compared the results with optimal values obtained by exhaustive search of all possible combinations and the results obtained by uniform partitioning in which the link capacity is divided into partitions of equal capacity for all classes.

Fig. 2 shows the evolution of the best fitness for a run for the DPAP problem. It can be observed that the best-of-generation performance converges to the optimal solution within 20 generations. Fig. 3 shows the similar trend for the case of the SPAP problem, in which the optimal solution was found at the 26th generation.

Figs. 4 and 5 compare the performances of GA and uniform partitioning as the calls of class 2 gets heavier when the load of class 1 is fixed for the DPAP case. Fig. 4 shows the result when the load of class 1 is light. Fig. 5 is the result for a heavy load of class 1 calls. It is clearly observed that the GA solutions are superior to those by uniform partitioning irrespective of offered load levels of calls for DPAP.

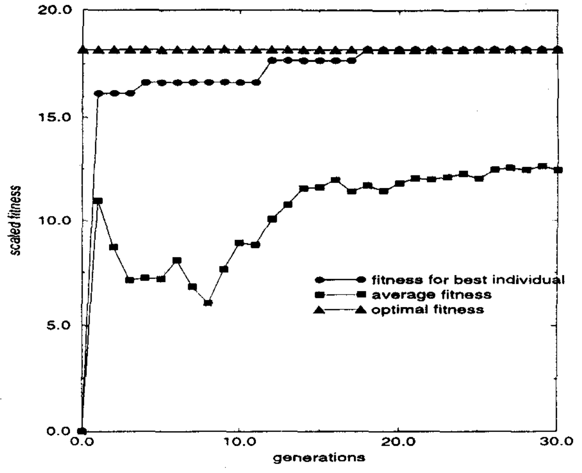

Fig. 2. *Evolution of the best fitness as a function of generation for DPAP.*

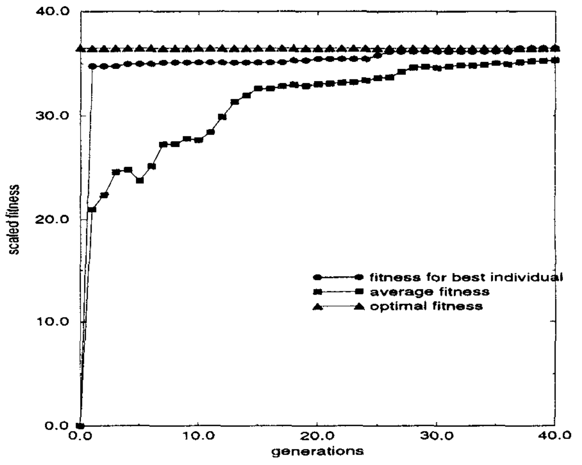

Fig. 3. *Evolution of the best fitness as a function of generation for SPAP.*

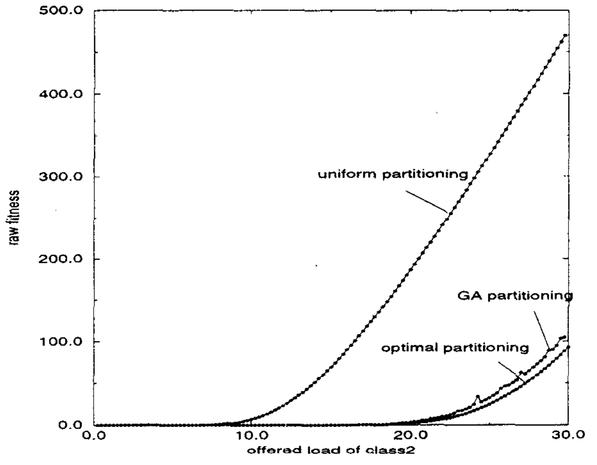

Fig. 4. *Performance of GA in comparison with that of uniform partitioning for DPAP when the load of class 1 is light.*

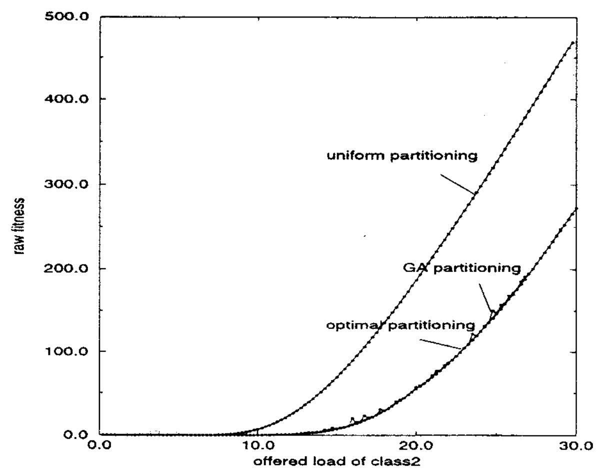

Fig. 5. *Performance of GA in comparison with that of uniform partitioning for DPAP when the load of class 1 is heavy.*

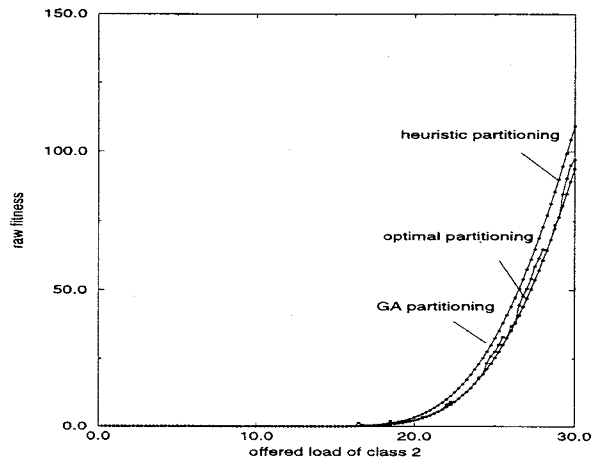

Fig. 6. *Performance of GA in comparison with that of heuristic partitioning for SPAP when the load of class 1 is light.*

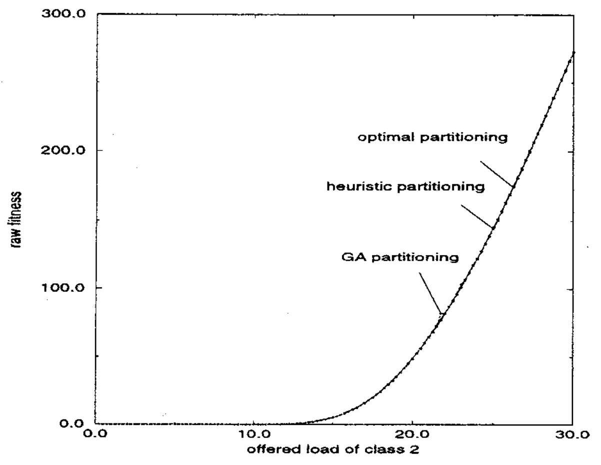

Fig. 7. *Performance of GA in comparison with that of heuristic partitioning for SPAP when the load of class 1 is heavy.*

A comparison of the GA method and the heuristic method for SPAP is shown in Figs. 6 and 7. The results indicate that the GA method is generally superior to the heuristic method for a light load. However, for some offered loads of class 2, the heuristic approach achieves better results. For a heavy load of calls, there is no significant difference in the quality of solutions obtained by both methods. Overal results suggest that genetic algorithms find optimal or near-optimal solutions for a wide range of load levels.

V. Conclusion and Further Work

In this paper, we formulated the link partitioning problems as nonlinear nondifferentiable integer optimization problems. A genetic algorithm was applied to these problems to find optimal or sub-optimal solutions. Experimental results show that the genetic algorithm is superior to uniform partitioning and the heuristic method for two-class cases. And, we can know that the genetic algorithm is applied well to instances of nonlinear nondifferential integer optimazation problems, the link partitioning problems.

Future work can be performed in two directions. One direction is to investigate the scaling properties of the genetic approach. Studies are in progress on the cases where calls from more than two classes arrive at the communication link. The second direction of extension involves the assumptions we made in this work. In this paper we have assumed that incoming calls are modeled as a Poisson process, and that the holding time of each call is exponentially distributed. Although both assumptions are reasonable for many problems in practice, characteristic parameters of objects requesting shared resources may have a different distribution in other resource allocation environments. Future study inludes the extension of the genetic approach for finding optimal resource partitions when the requests have different characteristics than the Poisson distribution.

References

[1] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, vol. 1, pp. 1-23, 1993.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning.* Addison Wesley, 1989.

[3] L. Davis and S. Coombs, "Genetic algorithms and communication link speed design: theoretical considerations," *Proc. Second International Conference on Genetic Algorithms*, 1987, pp. 252-256.

[4] C. C. Palmer and A. Kershenbaum, "An approach to a problem in network design using genetic algorithms," *Networks*, vol. 26, pp. 151-163, 1995.

[5] M. Kawarasaki and B. Jobbari, "B-ISDN architecture and protocol," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 9, pp. 1405-1415, 1991.

[6] S. Gupta and P. P. Gandhi, "Dynamic routing in multi-class nonhierarchical networks," *Proc. ICC'94*, 1994, pp. 1390-1394.

[7] J. H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: The University of Michigan Press, 1975.

[8] D. Bertsekas and R. Gallager, *Data Networks*, Prentice Hall, 2nd Edition, 1992.

[9] M. de Prycker, *Asynchronous Transfer Mode Solution for Broadband ISDN*, Ellis Horwood, 1993.

[10] K. T. Cheng and F. Y. Lin, "On the joint virtual path assignment and virtual circuit routing problem," *Proc. Globecom'94*, 1994, pp. 777-782.