INSTITUTE OF POLICY AND PLANNING SCIENCES

Discussion Paper Series

No. 920

Optimal Wavelength Converter Placement in Optical Networks by Genetic Algorithm

by

Johannes Hamonangan Siregar, Hideaki Takagi,
and Yongbing Zhang

April 2001

UNIVERSITY OF TSUKUBA
Tsukuba, Ibaraki 305-8573
JAPAN

Optimal Wavelength Converter Placement in Optical Networks by Genetic Algorithm

Johannes Hamonangan Siregar
Doctoral Program in Policy and Planning Sciences
University of Tsukuba
Tsukuba-shi, Ibaraki 305-8573, Japan
E-mail: siregar@shako.sk.tsukuba.ac.jp

Hideaki Takagi
Institute of Policy and Planning Sciences
University of Tsukuba
Tsukuba-shi, Ibaraki 305-8573, Japan
E-mail: takagi@shako.sk.tsukuba.ac.jp

Yongbing Zhang
Institute of Policy and Planning Sciences
University of Tsukuba
Tsukuba-shi, Ibaraki 305-8573, Japan
E-mail: ybzhang@shako.sk.tsukuba.ac.jp

April 23, 2001

Abstract

In optical networks, wavelength converters are required to improve the efficiency of wavelength-division multiplexing. In this paper, we propose a genetic algorithm to determine the optimal locations of the nodes where a given number of converters are placed. Optimality is achieved by the minimum wavelength blocking probability. Our algorithm is applied to realistic networks constructed from the locations of NTT offices in Ibaraki and Kanto areas, and is shown to reach the optimal solution in a limited number of generations. The computational time is compared with the exhaustive search algorithm.

1 Introduction

In communication networks, the use of the optical technology has been popular for the need of wide bandwidth, high-speed transmission and a large number nodes. For optical transmission, wavelength division multiplexing (WDM) is proposed which has the ability to spend many independent optical wavelengths on a single fiber link. One wavelength is dedicated to each channel between two adjacent nodes of the network. Interconnection between nodes is given a set of available wavelengths.

Because of the limitation on the number of the wavelengths in each direct link, there may not be a through wavelength between a pair of source and destination nodes of the path. The packets going to a destination node may reach their destination by hopping through a sequence of intermediate nodes. With wavelength conversion capabilities inside the network, the nodes are capable of routing different wavelengths, which can be reused throughout the network to establish all the required connections [1,2,3]. A wavelength converter is a device that can be placed on the node where the wavelength conversion mechanism is assigned. It converts data from one incoming wavelength to another outgoing wavelength. It can improve the utilization of wavelengths and minimize the network cost [3].

We consider the converter placement problem as a kind of combinatorial optimization problem with constraints. Converters are placed in distinct nodes because the number of converters is limited. Each placement yields a blocking probability value of the network. The optimal solution will be found if a combined placement of converters has the minimum blocking probability value. The method of searching the optimal solution in a combinatorial problem is problematic due to the regional search of feasible solutions. Exploring the search space of feasible solutions to get an optimal solution has brought several algorithms to develop.

Approaches for the converter placement problem have been developed in exact as well as heuristic algorithms. Exact algorithms use exhaustive search that delivers an optimal solution; an implementation is based on the construction of auxiliary graphs [4,5]. Heuristic algorithms are based on probabilistic performances with computational results such as random placement [1]. In [6], the authors explore the impact of placement of converters on the blocking probability performance for different numbers of converters whose nodes are obtained by genetic algorithm and for different conversion degrees.

We focus our study on using genetic algorithm that is of a heuristic type to determine the set of nodes with converters which gives the minimal blocking probability. We present our experimental results for small and large networks.

Genetic Algorithms (GAs) have been applied to various optimization problems [7,8,9]. The GA process uses a mechanism of natural selection from biology concept. This is an evolution of individuals from one generation to the next, based on the elimination of weak individuals and the reproduction of strong individuals. The individuals are analogous to the possible solutions of the problem. The stronger individuals are related to the nearly-optimal solutions that we search in optimization problems. These individuals will survive over a number of generations until the strongest one remains, that is, an optimal solution of the problem is obtained. For reproducing individuals in the population, genetic operators like selection, crossover, and mutation are used. These operators will explore more combinations of individuals which may lead to an optimal solution of the problem.

In this paper, we propose a GA mechanism for optimizing the converter placement. Section 2 contains a detailed formulation of the converter placement problem and evaluation of the blocking probability. Genetic algorithms in general are described briefly in section 3. The implementation of GA for the converter placement problem is given in section 4. Results of experiments with GA and comparison of the GA solutions to the solutions found by the exhaustive search algorithm are presented in section 5. In section 6 we make some concluding remarks.

2 The Converter Placement Problem

2.1 Network Model

Following [4], we introduce a network model of the converter placement problem using graph terminology. Consider a directed graph $G = (V, L)$, where V is the set of nodes and L is the set of directed links in the network. Let the nodes be numbered $1, 2, \ldots, N$ and let l_{ij} represent the directed link from node i to node j. There are F wavelength on each link, and each call requires a full wavelength on each link it traverses. Call requests arrive randomly at each source node and select any of the remaining nodes as their destination with equal probability. Each call uses a prespecified shortest path. All the shortest paths in the network are assumed to satisfy the optimality principle. This principle states that if a node x is on the optimal shortest path from node y to node z, the optimal shortest path from node x to node z follows the same route on that part of the path from y to z. If the prespecified shortest path is available, a lightpath is established between the source and destination nodes. Otherwise, the call may not be routed through an alternate path and is assumed to be blocked.

Let $A = [\lambda_{ij}]$ be the traffic matrix, where λ_{ij}, $i \neq j$, denotes the node load from node i to node j, and $\lambda_{ii} = 0$. Let the link load per wavelength be ρ_{ij} for link l_{ij}. Specifically, ρ_{ij} is the probability that a given wavelength is occupied by a lightpath on link l_{ij}. The link loads per wavelength per link can be obtained from the node loads by

$$(1)\qquad \rho_{ij} = \frac{\sum \lambda_{sd}}{F} \qquad \forall \text{ nodes } s, d$$

such that the path from s to d includes link l_{ij} provided λ_{sd} is so small that $\rho_{ij} < 1$ for all i and j. Once ρ_{ij}'s are obtained, we assume for the simplicity of computation that the load for any wavelength on a link is statistically independent of the link loads for all other wavelengths and on all other links.

2.2 Evaluation of Blocking Probability

Suppose that K is the number of converter nodes to be placed in the network. Let $C = (c(1), c(2), ..., c(K))$ be the converter placement vector such that $1 \leq c(i) < c(i+1) \leq N, 1 \leq i < K$. The entries of C denote the placement of converters among the nodes $1, 2, \ldots, N$. Now consider the path p of an end-to-end call from a source node s to a destination node d in the network. We define a segment to be the set of links on the path between two consecutive converter nodes or between the source (or destination) and a converter node. If the path contains no converter nodes, it consists of a single segment between the source and destination.

Let us assume that

$$(2)\qquad f(i,j) = 1 - (1 - \{\bar{\rho}_{ii_1}\bar{\rho}_{i_1 i_2} \cdots \bar{\rho}_{i_n j}\})^F$$

is the success probability for the segment from node i to node j on the path, where $\bar{\rho}_{xy} = 1 - \rho_{xy}$ for link l_{xy}, and $i_1, i_2, \ldots, i_n$ are the nodes in the segment between nodes i and j. Let k be the number of converters placed on the nodes (excluding the source and destination nodes) of path p, where $0 \leq k \leq K$. Given the converter placement vector C for the network, we have $d_{sd} = (d(1), d(2), ...d(k)) \subset C$ as the converter placement vector for path p such that the entries of d_{sd} correspond only to those nodes of path p which have converters on them. This divides path p into $k+1$ segments. Hence the success probability for the end-to-end call from source s to destination d with converter placement vector C is given by

$$(3)\qquad S_{sd}(C) = \prod_{i=0}^{k} f(s_i)$$

where $f(s_i) = f(d(i), d(i+1)), 1 \leq i \leq k-1$, is the success probability for the segment s_i between the converter nodes $d(i)$ and $d(i+1)$. $f(s_0)$ is the success probability for the segment between the source node s and converter node $d(1)$, and $f(s_k)$ is the success probability for the segment between the converter node $d(k)$ and the destination node d. The blocking probability for this call is

$$P_{sd}(C) = 1 - S_{sd}(C) \tag{4}$$

Thus we can calculate the blocking probabilities for all end-to-end calls in the network. We can then obtain the blocking performance $\Gamma(C)$ of the network with converter placement vector C by

$$\Gamma(C) = \frac{\sum_{\forall s,d} \lambda_{sd} P_{sd}(C)}{\sum_{\forall s,d} \lambda_{sd}} \tag{5}$$

The above formula will be used to calculate the blocking performance and to select the best converter placement that gives the minimum blocking probability in the network.

2.3 Exhaustive Search Algorithm

A simple way to solving the optimal converter placement problem is to use the method of exhaustive search and blocking performance computation. In this method, given the directed graph G with N nodes and the number K of converters to be placed, we initially obtain the set of all converter placement combinations $C_j^K, 1 \leq j \leq \binom{N}{K}$. Here $\binom{N}{K}$ is the number of combinations to choose K converter nodes out of N nodes. C_j^K is the jth converter placement vector and can be further expanded as $C_j^K = (c_j^K(1), c_j^K(2), \ldots, c_j^K(K))$ where $c_j^K(i), 1 \leq i \leq K$, is the node of the ith converter in the jth combination. For each converter placement combination, we calculate the blocking performance according to Equation (5). Then we select the converter placement combination which gives the minimum blocking probability as the optimal placement. Obviously, this method is not an efficient way to solve the problem unless N and K are small.

3 Genetic Algorithm

Genetic algorithm (GA) is based on the genetic processes from natural selection; it uses the terminology from natural genetics. In natural behavior, populations of the individuals evolve according to the principle of selection and survival of the fitness. Individuals that are more successful in adapting to their environment will have a better chance of surviving and reproducing. This means that the genes from the highly fit individuals will spread to an increasing number of individuals in successive generations. The combination of good characteristics from highly fit individuals will produce more fit offsprings as new individuals. In this way, species evolve to become more and more suited to their environment.

GA works with a population of individuals, which is represented by an array structure [10]. Each individual has a chromosome or a string of genes. Every gene has the values, called alleles, which are, in binary representation, 0 and 1. The initial population is created as a random number of strings.

GA process takes an initial population and applies genetic operators such as crossover and mutation in each reproduction, and a new generation is evolved. Each individual encoded into a string represents a possible solution of the problem. It is assigned a fitness value that is evaluated according to a given objective function. The fitness of each individual determines whether it will survive or not. After a number of iterations, or a stopping criterion is met, it is hoped that a nearly-optimal solution is found.

At each iteration, individuals in the population are selected for breeding by the selection procedure. It randomly selects individuals as parents in favor of the more fit individuals. The highly fit individuals have opportunities to produce new individuals as children who share some characteristic with their parent. There are several methods of selection. Tournament selection, roulette-wheel selection, and ranking selection are the main selection methods used in GA. In the tournament selection, the individual with the highest fitness wins the tournament, and the GA selects this individual to contribute to the reproduction. In the roulette-wheel selection, the ratio of an individual's fitness to the sum of all fitness values in the population determines its chance of being selected. In the ranking selection, the population of individuals is sorted according to an objective function.

Crossover combines two parents to form two new individuals as children with some features inherited from the parents. There are several types of crossover operators such as one-point crossover, two-point crossover, and uniform crossover. In the one-point crossover, a point called the crossover point is selected at random to separate each parent into two segements. The two segments, one from each parent, are then swapped to create two children. In the two-point crossover, two points selected at random break each parent into three segments, and a segment of one parent is exchanged with that of the other parent to produce two children. In the uniform crossover [11], two parents generate a single child by copying genes from the parents. Copying corresponding genes from one or the other parent creates a child. One of the parents is chosen according to a binary random number 0 or 1. If the random number is 0, the gene is copied from the first parent. Otherwise it is copied from the second parent.

After crossovers generate a child, the mutation is applied; it alters each gene with a small probability. By selecting one or more genes randomly, the bit values of the genes will change from 0 to 1 or 1 to 0. The genes of an individual are independently mutated. That is, the mutation of a gene does not affect the mutation of other genes. Mutation plays a role to restore lost genetic values when the population converges too fast.

By the genetic operators the new individuals are created to make a new population. The new population of individuals can be formed by the steady state approach or by the generational approach. The steady state approach chooses the best individuals out of the new as well as old individuals. The generational approach replaces the old population with the whole new set of individuals.

The population will evolve over successive generations so that the fitness of the best and average of individuals in each generation increases towards the expected solution of the problem. The GA process repeats until an expected solution is found.

A problem of using GAs for combinatorial problems occurs when the genetic operators breed individuals that produce an infeasible solution which does not satisfy the constraint of the problem. In such a case, a method of handling constraints must be added for satisfying the constraint and producing the feasible solution.

GAs are search algorithms often used to solve optimization problems. As a good search technique, GA has two abilities; exploring and exploiting. By crossover operators, GA explores the space of possible solutions, and creates more diverse members of the population. Exploitation makes use of the previous possible solutions for finding an optimal solution. Mutation may create individuals with the new fitness value which is better than the previous one.

A typical GA process consists of the following procedures:

- Initialize the population of individuals
- Evaluate the fitness value of each individual

do

- selection, to select parents from the population for reproduction
- crossover, to produce children from selected parents
- mutation, to alter some genes of selected children
- evaluate the fitness value for the children
- replace some or all individuals of the population by children

repeat *do* until a stopping criteria condition is satisfied.

More description of GA in general can be found in [10,11,12].

4 Genetic Algorithm for Converter Placement Problem

In this section, we present a GA for the converter placement problem. Each individual represents a possible solution to the problem as a string of bits. By a random procedure, we create the initial population. Genetic operators such as crossover and mutation will generate possible solutions. With constraint handling, we arrange the GA to satisfy the constraint and only feasible solutions are produced. When the stopping criteria condition is satisfied, we find a nearly-optimal solution to the problem.

4.1 Representation of Converter Placement

We use binary representation in encoding the possible solution. The converter placement is represented by an array of values 0 and 1. We can use this bit array for two alternative meanings.

In the first meaning of bit values, it represents the combination number in all converter placement combinations. For example, consider a network with 5 nodes and 2 converters to be placed. The number of combinations is $\binom{5}{2} = 10$, and $1 \leq j \leq 10$ corresponds to number j of the combinations. We construct the ordered set of 2-out-5 combinations as follows: number 1 is the converter placement at nodes 1 and 2, number 2 at nodes 1 and 3, and so on until number 10 at nodes 4 and 5. Thus we form the bit array of length 4. Since we have 10 combinations the size of search must not exceed 10. An infeasible solution occurs if the bit array has a value 0 or greater than 10.

The second meaning of bit values has a direct representation of the converter placement. For example, in a network with 5 nodes and 2 converters to be placed, vector solution $x = (x_j)$, $1 \leq j \leq 5$, means that if $x_j = 1$, node j is selected, and $x_j = 0$ otherwise. The length of the array is 5, since there are 5 nodes in the network. The bit array 01100 means that it selects nodes 2 and 3 for converter placement. An infeasible solution occurs when the number of 1-bits exceeds the number of converters to be placed. The weakness of this representation can be solved by a constraint handling method. Another problem is that it produces sparse vectors when the number of converters is small in a network with a larger number of nodes. For example, vector 0000011000 means that it selects nodes 6 and 7 out of 10 nodes in a network. Then this representation has high possibility to produce infeasible solutions.

In our implementation of GA to the converter placement problem, we may employ either representation depending on the case. If the number of converters is small, it tends to produce many infeasible solutions; we then use the first representation. If the number of converters is large or many 1-bit values occur, the number of infeasible solutions may not be too large; we can then use the second representation.

4.2 GA Process

Let us elaborate each procedure of the GA process for searching the optimal combination of nodes with converters.

Initialization

Each individual of the population is a possible solution to the problem. The GA starts with an initial population which is generated from the random seeds.

Evaluation of Fitness Value

The objective function to be minimized is the value of blocking probability given in Equation (5). To maintain uniformity over the problem domain, we use a fitness value for the objective function normalized to a convenient range. The normalized objective function indicates the fitness of an individual that the selection uses to evaluate. Individuals with good fitness value will be selected for the next generation. Our GA will search the vector solution that minimizes the blocking probability.

Selection

We use the tournament selection method for selecting two parents to produce new individuals for the next generation. This selection leaves only those individuals with highest fitness values in the population [10].

Crossover

We use the uniform crossover operator for exploration of diversity in the individual's bit [11]. The probability of crossover is given in the parameter setting. Using the sequence of random numbers generated with this probability, GA performs the uniform crossover on selected pairs of parents.

Mutation

The mutation procedure is applied after the crossover on each child independently. Every child has an opportunity for changing the bit value according to the probability of mutation. The probability of mutation is given in the parameter setting. This value determines the probability that each bit in the child is changed.

Replacement

The new individual replaces the old individual or their parent in order to maintain a fixed population size if the fitness value of the new individual is higher than that of their parent. If the fitness value of the new individual is lower than that of their parent, the new individual is discarded and the next generation employs the old individual.

4.3 Handling Constraints

In our GA process, the initialization, crossover, and mutation procedures will breed new individuals with new fitness values. If they do not satisfy the constraint conditions, the GA produces infeasible solutions. We avoid producing infeasible solutions by changing the bits of binary representation in accordance to the constraint.

Let us refer back to the representation examples in section 4.1. In the first meaning of representation, the feasible solution of bit array must be between the value of 0001 and 1010. If the new individuals does not fall in this region, the constraint is violated. To satisfy the constraint some value must be added if the individual has a bit array 0000; some value must be subtracted from a bit array greater than 1010. In the second representation, the number of 1-bits is the number of converters. If it exceeds the number of converters, the constraint is violated. To satisfy the constraint, some 1-bits must be changed to 0-bits. If the number of 1-bits is less than that of converters, some 0-bits must be changed into 1-bits. For changing the bit values, we choose random bit positions until the constraint is satisfied.

The method of handling constraints modifies the GA process. It is incorporated in the initialization, crossover, and mutation procedures in order to produce only feasible solutions. Then the space of feasible solutions may include the optimal solution.

The effect of using only feasible solutions to evaluate the fitness values of individuals depends on the probabilities of crossover and mutation. The crossover and mutation should maintain a certain degree of diversity in the population so as not to converge immediately to one feasible solution.

4.4 Stopping Criteria

A disadvantage in the optimization with GA is the difficulty of deciding when to stop [7]. Although statistical variables such as average and best fitness values are available in each generation, their values change almost unexpectedly as generations evolve. Stopping after a certain number of iterations with no improvement or when the change in average fitness is small may cause the algorithm to stop too early or too late.

Another stopping criterion may be that if the average fitness attains the value we expect then the iteration stops. In this case, the number of iterations may become large and the long computation time may be needed.

In the present study, we have used a stopping criterion based on the number of generations. Our algorithm stops when the generation counter exceeds the preset maximum number of generations.

5 Experimental Results

In this section we show some numerical results of using GA for the converter placement problem. From the source data in the NTT database of their office locations (http://www.ntt-east.co.jp/tariff/ryoukin/) we construct Ibaraki and Kanto Networks to test our GA.

First, by the Delaunay triangulation [13] for a given set of node locations, we get the links of the network. Since the network uses the optical transmission, all the distance values between two adjacent nodes are assumed to be the same. With Djikstra algorithm, the shortest paths between any two nodes are obtained.

5.1 Ibaraki Network

We first construct the Ibaraki Network from $N = 14$ office locations in Ibaraki Prefecture as shown in Figure 1.
In our experiment, we assume the optical network design and traffic requirement as follows:

- Number of wavelength $F = 3$
- Number of converters $K = 2$
- All the node loads are the same, and given by $\lambda_{sd} = 0.1$

The GA parameters for computation are as follows:

- Population size = 20
- Mutation probability = 0.00333
- Crossover probability = 0.6
- Maximum number of generations = 20

Our experiments run under Unix on a Digital Celebris GL workstation. The results are presented in Figure 2 which shows performance of the worst, best and average solutions. The x-axis represents the number of generations and the y-axis is the value of the objective function. GA starts with the initial generation which consists of 20 individuals according the parameter setting. Each individual represents a combination of converter nodes out of 14 nodes in the network. It has the fitness value related to the blocking probability. An individual with the best value in each generation is the one with the smallest value of blocking probability among all individuals in the population. The average in each generation is the average value over the 20 individuals in the population. In Figure 2, the initial generation starts with the average value 0.051178, the worst value 0.054491, and the best value 0.045410. In generation 4, all individuals exhibit the same value, which is said that GA has come to the *premature convergence* behavior without guaranteeing the optimal solution [10]. However, after generation 4, GA improves the individuals' values by exploiting the search space in the previous generation with mutation procedure. Then in generation 10 the optimal solution is reached. From the display in this figure we can see the performance of GA to obtain the optimal solution. Our GA has produced an optimal solution in a small number of generations.

5.2 Kanto Network

The Kanto Network with $N = 82$ nodes is constructed similarly. It is more complex as shown in Figure 3.

At first, the optical network design and traffic requirement are assumed as follows:

- Number of wavelength $F = 3$
- Number of converters $K = 2$
- All the node loads are the same, and given by $\lambda_{sd} = 0.01$

A plot of the blocking probability value versus the ordered number of node combinations for the converter placement computed from the exhaustive search algorithm is shown in Figure 4. In this figure, the minimum blocking probability value is found to be 0.486743 at the combination number 2102 out of $\binom{82}{2} =$ 3321 ways of converter placement. The combination number 2102 corresponds to the converter placement at nodes 33 (Mitsukaido City, Ibaraki Prefecture) and 39 (Kuki City, Saitama Prefecture).

The GA parameters which we use for the Kanto Network are as follows:

- Population size = 40
- Mutation probability = 0.00333
- Crossover probability = 0.6
- Maximum number of generations = 60

The performance of GA is shown in Figure 5, where the generation starts with the average value 0.510037, the worst value 0.513699, and the best value 0.503240. In generations 6 to 8, GA comes to the premature convergence behavior.

However, after generation 9, GA improves the individuals' values by exploitation. In generation 17, the premature convergence behavior appears again until generation 24. Then in generation 36, GA finds the optimal solution at 0.486743 with convereters at Mitsukaido City and Kuki City. This agree with the result of exhaustive search shown in Figure 4.

In Figure 6 we present the best value performance of GA for different numbers of converters (K = 0, 2, 3, 5, 10, 20, 50, and 82) for the Kanto Network with $F = 3$ and $\lambda_{sd} = 0.01$. The GA parameters are:

- Population size = 40
- Mutation probability = 0.00333
- Crossover probability = 0.6
- Maximum number of generations = 200

As expected, a larger number of converters can decrease the value of blocking probability. As the search space becomes larger, the convergence is attained more slowly. The value $K = 0$ corresponds to the case in which no converters are placed in the network. It can be used as the upper bound of the blocking probability. The value $K = 82$ corresponds to the case in which all nodes are equipped with converters. It provides the lower bound of the blocking probability.

Table 1 compares the computational time of our GA with the exhaustive search algorithm for the Kanto Network with various numbers of converters. The computational time requirement of the exhaustive search algorithm becomes large in proportion to the size of the search space which grows exponentially with the number of converters. In contrast, the computational time requirement of GA barely grows with the number of converters. This experiment shows that the computational time of GA achieves considerable improvement over the exhaustive search.

Table 1: Runtime of exhaustive search and GA for the Kanto Network.

K	Search Space	Exhaustive ($min.$)	GA ($min.$)
2	3321	1:06	5:02
3	88560	28:45	5:32
5	2.72853×10^7	7.45 days	6:08
10	2.13928×10^{12}	-	7:20
20	6.20877×10^{18}	-	9:02
50	5.93991×10^{22}	-	12:30

6 Conclusion

The purpose of our GA method is to produce a possibly optimal solution for a certain population of individuals in a limited number of generations. This method is tested to obtain the optimal converter placement for the Ibaraki and Kanto Networks constructed from the NTT office locations. We have compared the performance with the exhaustive search algorithm.

A major difference between GA and exhaustive search algorithm is in the search space for the optimal solution. Exhaustive search must work to compare the blocking probability values for all combinations of converter placement in a network. For large networks where the number of combinations grows, it needs explosive computational time for searching the optimal solution. On the other hand, since GA restricts trials to feasible solutions it does not need to compare all of the blocking probability values.

GA tries to reach an optimal solution for a certain population in a number of generations. Using probability model in crossover and mutation to produce more diversity in the population, it can bring more chance to produce the solution with optimal value. It has been shown by our experiment that GA can produce an optimal solution in a limited number of generations by spending less time than the exhaustive search method.

Acknowledgements
The authors wish to thank Dr. Kounosuke Kawashima of NTT Advance Technology, Traffic Research Center, for providing us with the web site url for the database of NTT office locations. They are also grateful to Dr. Sashisekaran Thiagarajan who kindly answered some questions by e-mail.

References

[1] B. Ramamurthy and B. Mukherjee, Wavelength conversion in WDM networking, *IEEE Journal on Selected Areas in Communications*, 16(7):1061–

1073, September 1998.

[2] K. C. Lee and V. O. K. Li, A wavelength-convertible optical network, *IEEE Journal of Lightwave Technology*, 11(5/6), 962–970, May/June 1993.

[3] R. Ramaswami and K. Sivarajan, *Optical Networks: A Practical Perspective,* Morgan Kaufmann, 1998.

[4] S. Thiagarajan and A. K. Somani, An efficient algorithm for optimal wavelength converter placement on wavelength-routed networks with arbitrary topologies, in *Proc. INFOCOM 1999*, 916–923, 1999.

[5] S. Subramaniam, M. Azizoglu, and A. K. Somani, On the optimal placement of wavelength converters in wavelength-routed networks, *Proc. INFOCOM 1998*, 902–909, 1998.

[6] C. Vijayanand, M. S. Kumar, K. R. V. and P. S. Kumar, Converter placement in all-optical networks using genetic algorithms, *Computer Communication*, 23, 1223–1234, 2000.

[7] C. Gazen and C. Ersoy, Genetic algorithms for designing multihop lightwave networks topologies, *Artificial Intelligence in Engineering*, 13, 211–221, 1999.

[8] I. E. Kassotakis, M. E. Markaki, and A. V. Vasilakos, A hybrid genetics approach for channel reuse in multiple access telecomunications networks, *IEEE Journal on Selected Areas in Communications*, 18(2):234–242, February 2000.

[9] T. Kuo and S. Y. Hwang, Using disruptive selection to maintain diversity in genetic algorithms. *Applied Intelligence*, 7, 257–267, Kluwer Academic Publishers, 1997.

[10] D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning,* Reading, MA: Addison-Wesley, 1989.

[11] L. Davis, *Handbook of Genetic Algorithms,* New York: Van Nostrand Reinhold, 1991.

[12] M. Mitchell, *An Introduction to Genetic Algorithms,* Cambridge, MA: MIT Press, 1996.

[13] J. O'Rourke, *Computational Geometry in C,* New York, Cambridge University Press, 1994.

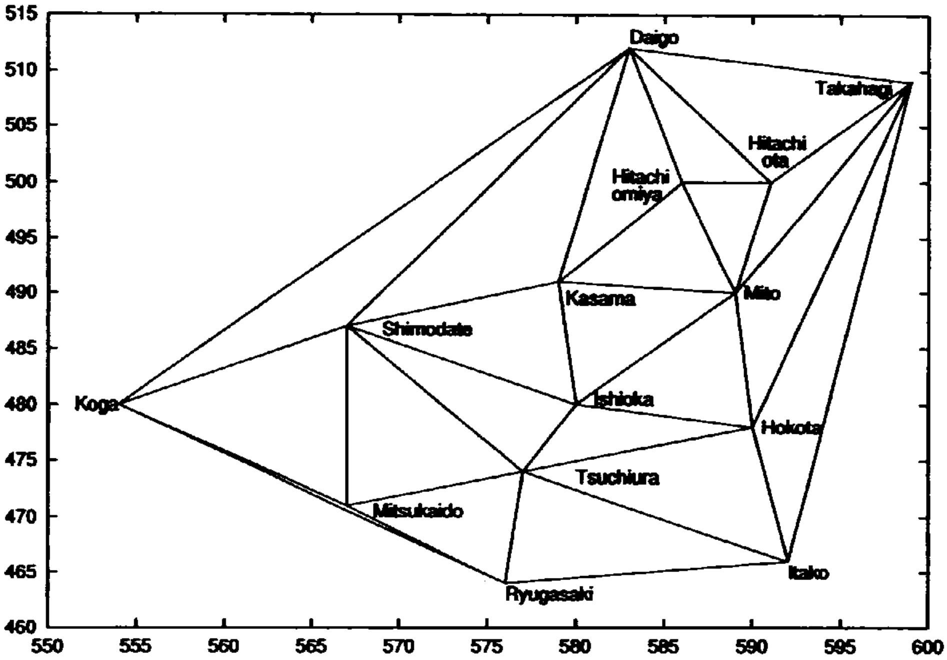

Figure 1: Configuration of the Ibaraki Network.

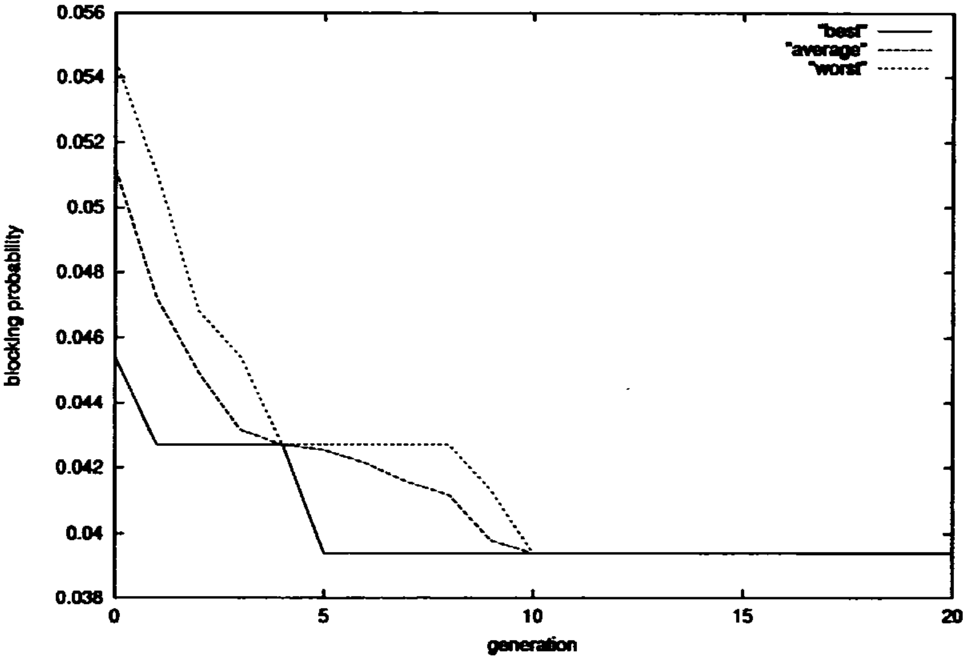

Figure 2: GA performance for the Ibaraki Network.

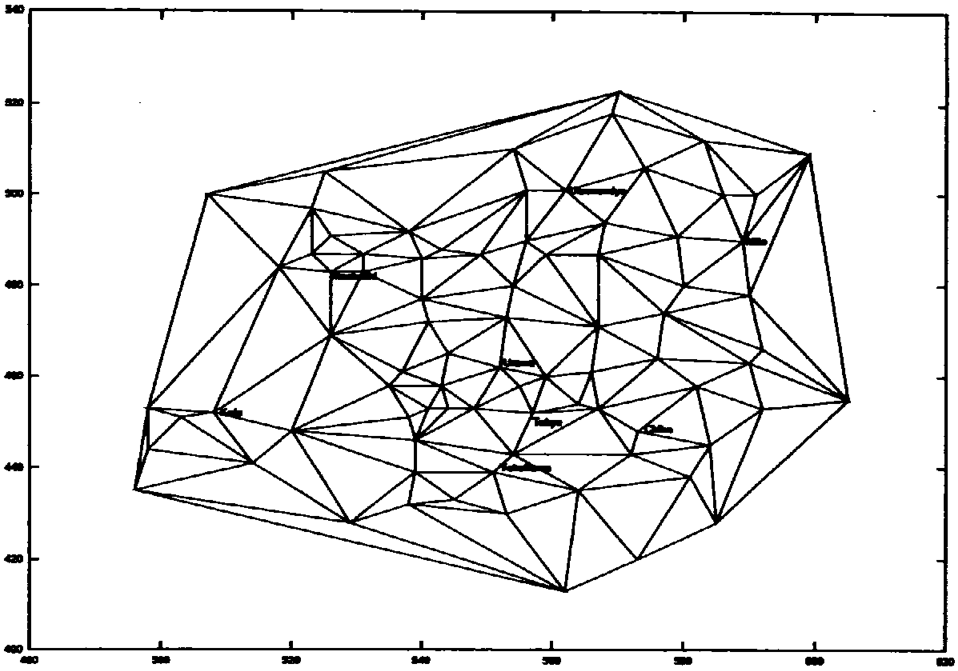

Figure 3: Configuration of the Kanto Network.

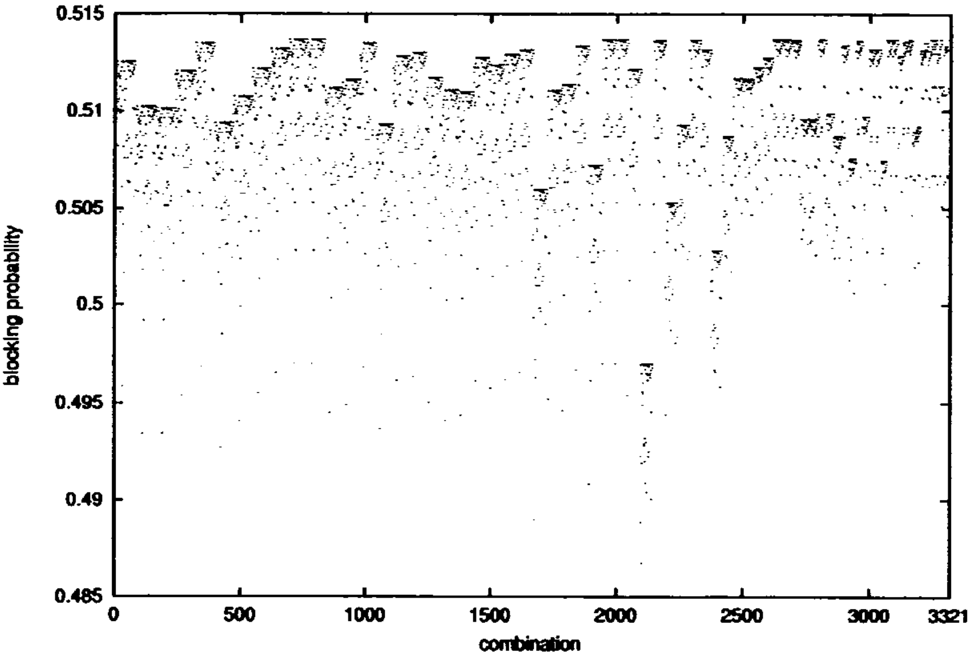

Figure 4: Exhaustive search for the Kanto Network with 2 converters.

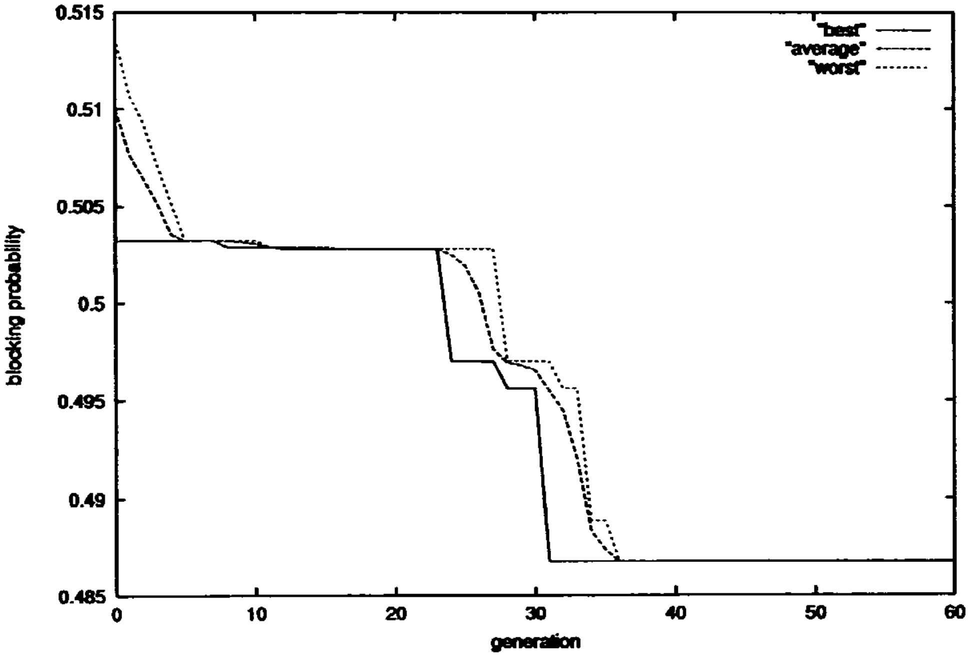

Figure 5: GA performance for the Kanto Network with 2 converters.

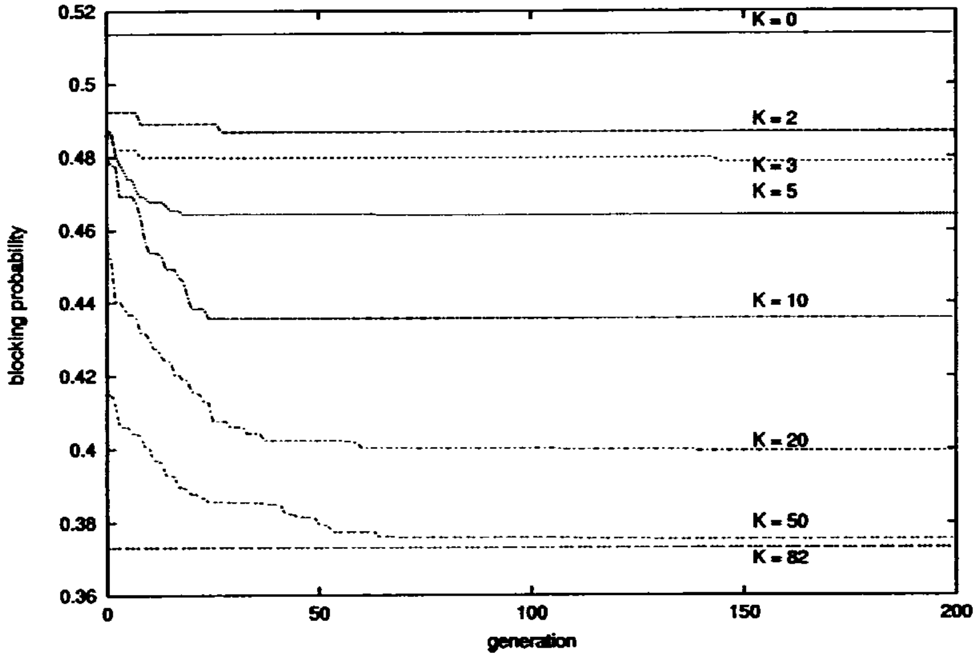

Figure 6: GA best value performance for the Kanto Network with K= 0, 2, 3, 5, 10, 20, 50, and 82 converters.