[Lip75] Steven A. Lippman. Applying a new device in the optimization of the exponential queueing systems. *Operations Research*, 23(4), July-August 1975.

[RT89a] K.W. Ross and D. Tsang. Optimal circuit access policies in an isdn environment: A markov decision approach. *IEEE Transactions on Communications*, 37(9), 1989.

[RT89b] K.W. Ross and D. Tsang. The stochastic knapsack problem. *IEEE Transactions on Communications*, 37(7), 1989.

[RY93] Christopher Rose and Roy D. Yates. Optimal call admission to single cells of a cellular mobile network. *Winlab Technical Report*, (TR-61), 1993.

[Yen] Aylin Yener. Finding good call admission policies for cellular mobile networks. *MS Thesis, Rutgers University.*

[ZY89] Ming Zhang and Tak-Shing P. Yum. Comparisons of channel assignment strategies in cellular mobile telephone systems. *IEEE Transactions on Vehicular Technology*, 38(4), November 1989.

Author Biographies

Aylin Yener was born in Istanbul, Turkey in 1971. She received the B.S. degrees in Electrical Engineering from College of Engineering and in Physics from College of Arts and Sciences of Boğaziçi University, Istanbul, in 1991. She is currently an M.S. student and a Graduate Assistant in Department of Electrical and Computer Engineering at Rutgers University, N.J. Her current research interests include analysis and optimization of system performance in telecommunication networks, in particular mobile communication networks.

Christopher Rose was born on January 9, 1957 in New York City. He received the B.S. (1979), M.S. (1981), and Ph.D.(1985) degrees all from Massachusetts Institute of Technology in Cambridge, Massachusetts. Dr. Rose's graduate work was supported by the AT&T Cooperative Research Fellowship Program and following graduate school he was with AT&T Bell Laboratories in Holmdel, N.J. as a member of Network Systems Research Department. He is currently an Assistant Professor of Electrical and Computer Engineering at Rutgers University, N.J. His technical interests include mobile communication networks, cellular automata and genetic algorithms as applied to optimization problems in communications. Dr. Rose is a Henry Rutgers Research Fellow at Rutgers University.

blocking probabilities of each customer class. We have modeled the system with a multi-dimensional Markov Chain and searched for call admission policies that will improve the system performance over the conventional admit-if-possible strategy.

Although in theory, it is possible to find the optimal call admission policies by means of Markov Decision Processes (MDP) for such Markovian systems, the state space of the system gets large very quickly and MDP becomes impractical. We have suggested a Genetic Algorithm (GA) for the search to find *good* call admission policies (as opposed to optimal) instead to overcome the limitation on the size of the problem.

We have shown that for small systems, GA performs comparably to MDP, and the resulting policies have better performances than AIP. The GA search blocked coding also proved useful in convergence rate and performance. Therefore, we suggest the method is extendable to larger systems where MDP is unable to attack the problem for searching an optimal policy.

To increase the efficiency of the search for an optimal policy, an essential issue is to find a compact coding for the policy. Upto this point, we have considered unconstrained state spaces while coding the policies. For larger systems, it is necessary to work on the constrained state space and code the policy accordingly to overcome the computational limits. We have used random block coding (described in Sec 4) in some of our experiments which worked successfully and increased the efficiency. Other coding methods such as Program codes and Lempel-Ziv are currently under investigation.

References

[BP89] Reiner Beck and Herbert Panzer. Strategies for handover and dynamic channel allocation in micro-cellular mobile radio systems. *Vehicular Technology Conference*, 1989.

[Gol89] David A. Goldberg. *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[GS83] I.S. Gopal and T.E. Stern. Optimal call blocking policies in an integrated services environment. *Conference on Information Sciences Systems*, 1983.

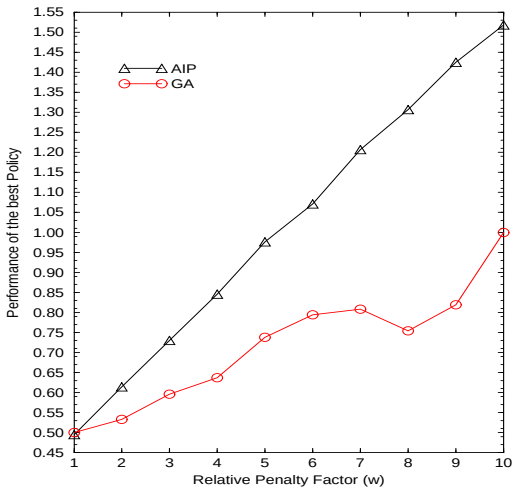
[GS93] M. Gopalakrishnan and M.K. Sundareshan. Optimal channel allocation policies for access control of circuit-switched traffic in isdn environments. *IEEE Transactions on Communications*, 41(2), 1993.

[How60] Ronald A. Howard. *Dynamic Programming and Markov Processes*. M.I.T. Press, 1960.

[Kau81] J.S. Kaufman. Blocking in a shared resource environment. *IEEE Transactions on Communications*, 29(10), 1981.

[KG78] T.J. Kahwa and N.D. Georganas. A hybrid channel assignment scheme in large-scale, cellular structured mobile communication systems. *IEEE Transactions on Communications*, COM-26(4), April 1978.

GRAPH 3



are the same for both classes, but the handoff rate is much faster for one class. Specifically, we looked at the system with:

$$\rho = 1.0 = \eta = 1.0$$
$$\gamma_1 = 1.0$$
$$\gamma_2 = 10 \times \gamma_1$$
$$\mu_1 = \mu_2 = 1.0$$
$$p_{split} = 0.5$$

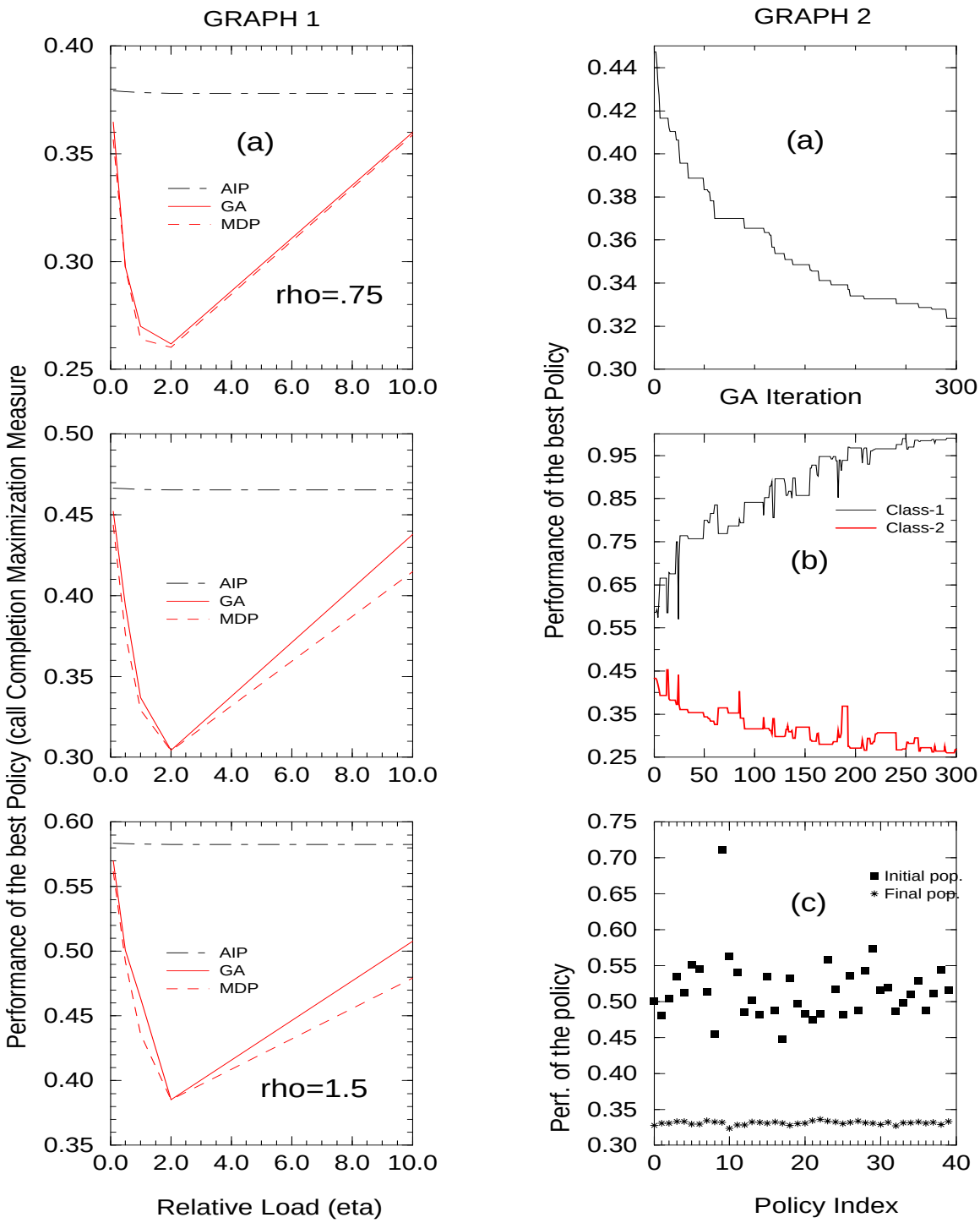
The model corresponds to a system with two mobility classes which we can interpret as the fast mobile users (users in motor vehicles, planes etc.) and the slow mobile users (pedesterians, bycycle riders etc.).

We have considered relative penalty factors (ω) in the range of [1,10].The community consisted of 100 policies and the block code representation was used with 1024 codewords of 27 bits each. This has resulted in policies of 4860 bits long (as opposed to the 13122 bits in complete representation) and increased the convergence rate towards the optimal policy. The GA was implemented on these communities for 200-300 generations.

The resulting performances of the best policies found by the GA is plotted in Graph 3 along with the AIP performances. It is observed that the policies found by the GA succesfully reduced the handoff blocking measure given by the AIP strategy for $\omega > 1$. The known optimal for call completion maximization ($\omega = 1$) is the AIP strategy itself which is also found by GA. As ω gets larger, accepting the handoffs over the new call arrivals becomes more favorable and as a result of this we have observed a convergence to a trivial policy where *no* new call is accepted to the system, and *no* handoff is blocked for large enough ω values, e.g. $\omega = 10$.

6 Conclusion and Future Work

We have considered the call admission problem for linear 1-dimensional cellular networks with two types of customers of differing characteristics such as handoff rates and call holding times. The system performance is defined as a linear combination of new call and handoff


GRAPH 1
(a)
AIP
GA
MDP
rho=.75
(a)
rho=1.5
Performance of the best Policy (call Completion Maximization Measure
Relative Load (eta)
GRAPH 2
(a)
GA Iteration
Performance of the best Policy
(b)
Class-1
Class-2
(c)
Initial pop.
Final pop.
Perf. of the policy
Policy Index

p_{split} is taken to be 0.5 for these experiments.

The call admission policies for this problem consist of binary decisions for each state of the system which are represented by single bits, i.e. we have used the complete representation approach. The performance measure used is the call completion maximization (minimization of the average blocking probability). Due to the nature of call completion maximization measure (since the measure does not favor any kind of arrival, whether it is a handoff or a new call arrival to the system) [Yen], we did not have to differentiate between new call and handoff arrivals in our policy representation. We also looked at an unconstrained state space representation and let the interference constraints decide whether the policy decision is given at a feasible state or not. In other words, for feasible states, the service requests are accepted or denied by the policy decisions, whereas for infeasible states, it is the *system*, not the policy, that denies the service. The motivation for this method is that for small systems the number of feasible states is close to the total number of states and computer implementation of the unconstrained state space is easier. The unconstrained state space has cardinality $(C+1)^{2N}$. The unconstrained policies are therefore $6 \times N \times (C+1)^{2N}$ bits long. However, since for call completion maximization we can combine the two handoff and new call arrival decisions in 1 decision (arrival), the policies for this problem are $2\mathrm{N}(C+1)^{2N}$ bits long (4374 bits/policy for N=3 and C=2).

For each of the ρ and η values above, a genetic algorithm search described in Section 4 is implemented for improving the performance of the system. A community of 40 policies where each policy is an *organism* of a single *gene* are fed to GA and allowed to evolve for 200 generations. 20 % of the community is mated in each iteration, so the set consists of policies searched by GA has cardinality 1640, whereas the unconstrained policy space has 2^{4374} policies. The new members of the community are replaced by the organisms that are chosen to die out if their performance is better than these older organisms. The fitness measure values ($\overline{P_b}$) are calculated using Monte Carlo simulations.

The best performances found by the GA, the optimal performances found by MDP and the admit-if-possible (AIP) strategy results are compared for these systems in Graphs 1(a)-(c) for ρ=0.75,1,and 1.5 respectively. The known optimal is a special case of a reservation policy by which no slower class (class 1) call is admitted to the system. The performances found by the GA are very close or identical to the optimal values which are lower than the admit-if-possible strategy. The structure of the best policies shows that, improvements are achieved by blocking the slower class more often than the faster class. As an example to show this fact, we have looked at the special case where the total loading ρ and the relative load η are both 1.0. GA was this time allowed to evolve for 300 generations with all other parameters kept the same with the previous runs. The resulting policy penalized the slower class (class I) much more severely relative to the faster class (class II). The best policy performances found in each iteration of GA and blocking probabilities for each class P_{b1} and P_{b2} are shown in Graph 2(a)-(b). It is also observed that the community after 300 generations is much smoother than the starting community, i.e. the performances of the individuals are much closer to each other and therefore to the best policy, supporting the idea of *the survival of the fittest* (Graph 2(c)).

We have also also performed experiments with the 3-cell, 2-channel system to search for call admission policies that minimizes the handoff blocking measure, P_{hp}. The parameter set this time was chosen to model a 2-class system where the arrival and call holding rates

measure of performance. The only information needed for the algorithm is the value of the performance of the candidate for the solution.

The major difference between MDP and GA lies in the fact that, MDP always finds the optimal solution for the problem *if it can handle the problem*, whereas GA usually *can handle the problem* but does not necessarily find the optimum. GAs search for the optimum from a set of candidate solutions, and try to improve on the best solution found in each iteration. The solution, after a finite number of steps, may not converge to the exact optimum. However, often the results obtained by GA are close to the optimal found by MDP. These encouraging results suggest the use of GAs for more complex problems that are not solvable by MDP.

5 Results

We first consider a small problem that can also be solved using MDP to compare the performances of the policies found by the GA to the known optimal, a linear highway system with 3-cells and 2 channels. The interference constraint dictates that the channel can not be used in the cell if it is in current use in any of the immediate neighboring cells, i.e. the *clique size* is 2.

The loading of the system, call holding times and the handoff rates of customer class j are assumed to be uniform over all the cells, i.e.

$$\begin{aligned} \lambda_{ij} &= \lambda_j, \quad \forall i, \quad j = 1,2 \\ \mu_{ij} &= \mu_j, \quad \forall i, \quad j = 1,2 \\ \gamma_{ij} &= \gamma_j, \quad \forall i, \quad j = 1,2 \end{aligned}$$

Offered load by class j to any cell is then defined as:

$$\rho_j = \lambda_j/\mu_j, \quad j = 1,2 \tag{10}$$

so that the total offered load to any cell becomes

$$\rho = \rho_2(1+\eta) \tag{11}$$

where $\eta = \rho_1/\rho_2$ is the *relative load*.

For this 3-cell linear system, the customers in the middle cell are allowed to request handoff to any of the two neighbors. Probability of requesting a handoff to the right (left) neighbor is a parameter denoted by p_{split} (1-p_{split}). The calls in the border cells can only request handoff to the middle cell. To test the genetic algorithm, we choose the following values for the handoff and call holding rates of the two class system:

$$\begin{aligned} \gamma_1 &= \mu_1 = 1.0 \\ \gamma_2 &= \mu_2 = 10 \times \gamma_1 \end{aligned}$$

We consider low, moderate and high loading values for ρ and vary η values:

$$\begin{aligned} \rho &\in \{0.75, 1.0, 1.5\} \\ \eta &\in \{0.1, 0.5, 1.0, 2.0, 10.0\} \end{aligned}$$

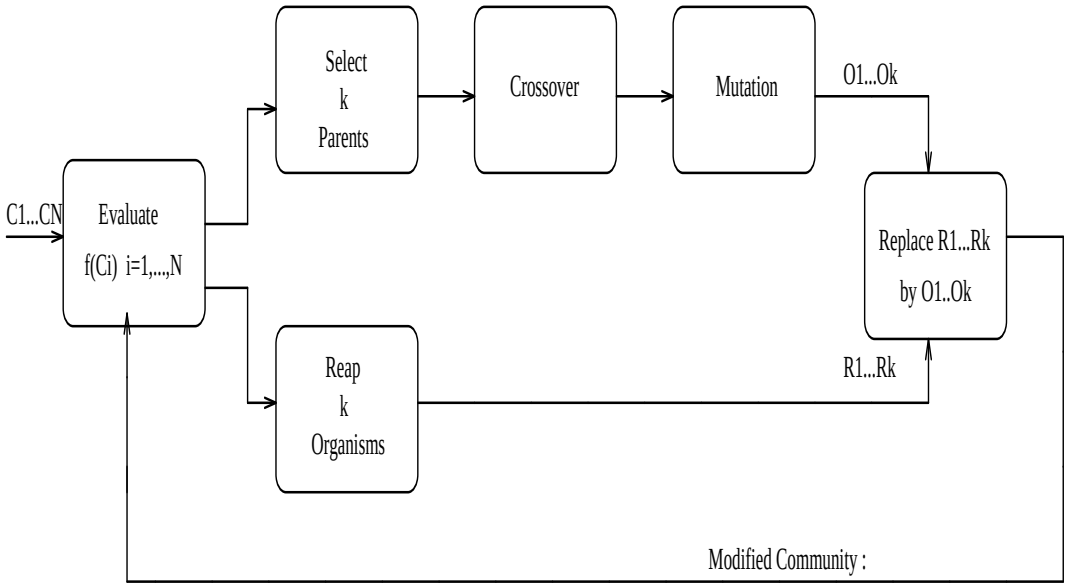


Figure 2: A simple Genetic Algorithm

representation is easy to implement and efficient for small systems, where the number of states is not very large.

2. For bigger systems, the *block coded* policy representation is used. This way a certain fraction of the policy space is searched to start with rather than the entire policy space. The method used for coding a policy is simply to use a code book of 2^G entries. Each entry is a different K-bit pattern where $G < K$. Codewords are constructed at random initially. The policy consists of different combinations of concatenated K-bit codewords. So, if L is the number of bits to specify the policy with the complete representation, number of blocks (codewords) to represent the policy is $\lceil \frac{L}{K} \rceil$. The coded policy representation is nothing but the concatenation of the indices of these blocks. Since each codeword index can be represented by G bits, the length of the coded policy is:

$$L_c = G\lceil \frac{L}{K} \rceil \tag{9}$$

Each organism, in this representation, consists of $2^G + 1$ genes; the policy code and the collection of K bit codewords that are to be mated at each GA iteration.

The analytical complexity of a GA is primarily the complexity of the policy evaluation procedure. Blocking probabilities should be computed for the performance measure, but unlike MDP, it is possible to use approximation methods or Monte Carlo simulations instead of solving for the state equations of the system. So, the complexity of the algorithm is much less than MDP and problems that can not be handled by MDP can be studied using GAs.

Another advantage of GAs is they can be used to find solutions for *any* optimization problem for which a set of candidates for the solution can be constructed and coded to binary strings. So, we do not have to confine ourselves to Markov Processes or to a linear

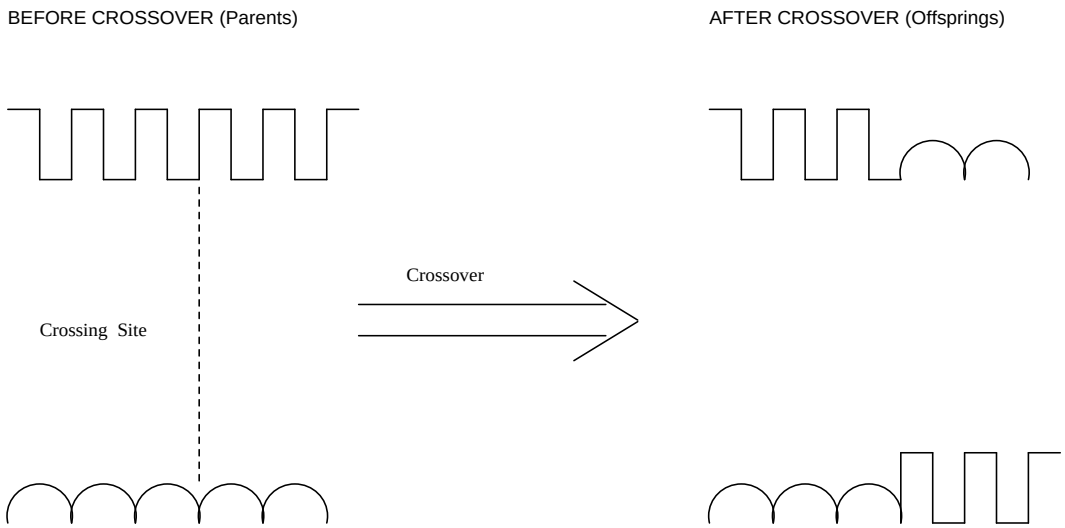


Figure 1: Crossover Operation

The basic method is to pick pairs of parents and exchange parts of their string codes. The position of where to cut the gene(s) is equally likely for all the positions. Once the position is selected, two offsprings are created by swapping the characters between the cut point and the end of the string(s). A simple single crossover operation between two strings is shown in Figure 1.

The organisms are allowed to mutate, i.e. change some of the characters, with a given probability. Mutation enables the community to utilize new gene information that is not present in the initial community and aids in the escape from local optima.

A typical two parent-two children Genetic Algorithm, shown in Figure2, shows these four basic steps. First a number of parents are chosen with the selection process to enter the mating pool. Then the crossover takes place on all the parents that are randomly paired. The offsprings of the crossover are the new entries to the population and are allowed to mutate before entering the community. Although not necessary, GAs usually assume a constant population size. Therefore, the offsprings replace some individuals from the old generation. The selection of these individuals to be replaced is achieved by the reaping process. The algorithm is allowed to run for a number of generations repeating these basic steps.

4.2 Application to Call Admission

In the call admission problem for cellular networks, we formulate the GA community as a collection of call admission policies, each policy being an organism. The fitness measure of each organism is one of the performance measures defined in Section 2. The policies are represented as bit strings in two ways:

1. Each Policy decision is represented with a bit, 1 for accepting and 0 for denying the service. The long bit string with $6 \times N \times S$ bits constitutes the single gene in the organism. We call this the *complete representation* of the policy. Complete

to any particular type of constraint. They only require the natural parameter set of the problem to be coded as a finite length string over some finite alphabet. The fact that GAs exploit coding similarities in a very general way results in overcoming most of the limitations of the functions to be optimized like continuity, derivative existence, unimodality etc.

With many optimization methods, we tend to evaluate the decision space on a point by point basis, i.e. the solution to the problem is searched by moving from one point to the other. This method can often result in locating false peaks in a multimodal space. GAs on the other hand, work with a number of points (a community) simultaneously which reduces the chance of finding local optima.

The initial community is a collection of coded individuals called *organisms* . Each organism contains group of characters called *genes* and corresponds to a solution candidate for the optimization problem; i.e., for call admission, the genetic code specifies the structure of the policy. The objective function of the optimization problem is called the *fitness measure.*

There are four basic steps in a simple Genetic Algorithm:

1. Selection
2. Crossover
3. Reaping
4. Mutation

Selection and reaping both use the *reproduction* operator. Reproduction is a process which copies organisms in the community according to their fitness measures. Each organism is evaluated and chosen with a probability that is assigned according to its fitness.

Selection determines the individuals that enter the *mating pool* for the crossover. A number of organisms are selected using the reproduction operator. If we seek a maximum fitness solution, then the probability of choosing candidate i is:

$$P_s(i) = \frac{f_i}{\sum_{j=1}^{n} f_i} \tag{7}$$

where f_i is the fitness of candidate i. So, the higher the fitness value of the candidate, the higher the chance to be chosen for the crossover.

Reaping is the process to chose the group that will die in the generation. The selection is again done by the reproduction operator, but this time the probability of being selected is inversely proportional to the fitness measure. So, for the maximization problem described above, probability of reaping candidate i is:

$$P_r(i) = \frac{1/f_i}{\sum_{j=1}^{n} 1/f_i} \tag{8}$$

The reader should note that for minimization problems, as in the problem studied in this paper, the fitness is inversely proportional to the amount of blocking $\overline{P_b}$ or P_{hp}.

Crossover is the mating process which generates new candidates for the solution. The organisms that are selected as *parents* yield offsprings by exchanging parts of their genes.

Thus, we will call it *Handoff Priority Blocking* measure, P_{hp}. In general, P_{hp} is not a probability measure, but rather a linear combination of the probabilities. Note that $\omega = 1$ reduces simply to the Average Blocking Measure.

3 Call Admission Policies and Previous Art

A call admission policy is simply a mapping of the system state to a binary decision rule. As mentioned in Section 2, system state is described by the number and type of users in each cell. A policy determines whether to accept (1) or deny (0) service based on system state when a customer attempts a call origination or handoff in a cell. Therefore, for the 2-class system, the policy has to make the decisions for each cell at each state:

- Admit/Reject a new call arrival of type I
- Admit/Reject a handoff request of type I from the right cell
- Admit/Reject a handoff request of type I from the left cell
- Admit/Reject a new call arrival request of type II
- Admit/Reject a handoff request of type II from the right cell
- Admit/Reject a handoff request of type II from the left cell

So, the policy is a collection of $6 \times N \times S$ binary decisions, where N is the number of cells and S is the number of states in the system. The total numer of possible policies is therefore 2^{6NS}.

In principle, the search for the optimal policy for this multi-dimensional Markov Chain can be done using a Markovian Decision Process (MDP) as has been in [Lip75] [Kau81] [GS83] [RT89b] [RT89a] [GS93] and [RY93]. The method requires solving a number of linear equations (equal to the number of states of the system) [How60]. This limits the practicality of using MDP to search for the policies of systems where the state space is large. In fact, it is observed that the state space grows rather quickly for the two class cellular problem, and MDP becomes less practical even for small systems with only a few channels.

However, the success of simple heuristic policies such as server reservation seems to indicate that good policies are not necessarily as complex as might be thought based on the size of the policy space. It may therefore prove useful to search for good heuristics in some systematic way. To this end we suggest the use of genetic algorithms.

4 Genetic Algorithms

4.1 Background

Genetic algorithms (GAs) [Gol89] combine the rules of natural selection and genetics to form a search algorithm for optimization problems. GAs are fairly different from the traditional search algorithms, in the sense that their ability to search for the optimum is not confined

The performance of the system is defined as a weighted measure of new call and handoff blockings. Therefore, improving the performance means minimizing this measure. We assume any blocked call will be cleared from the system immediately. If a new call connection is requested by customer type i and the access is denied (either by the system or the policy), event *type i new call blocking* occurs. Similarly, if a call of class i is dropped during a handoff attempt by the system and/or by the policy, *type i handoff dropping* occurs. The probability of new call blocking for user class i is denoted by P_{ni}. Probability of handoff blocking for type i is denoted by P_{hi}. The performance of the system in general is defined as:

$$T = [\alpha_1 \alpha_2] \begin{bmatrix} P_{n1} \\ P_{n2} \end{bmatrix} + [\beta_1 \beta_2] \begin{bmatrix} P_{h1} \\ P_{h2} \end{bmatrix} \tag{1}$$

where α_1, α_2, β_1, β_2 are the weight factors.

We will focus on two different special cases of equation 1:

2.1 Call Completion Maximization

For cellular systems, an important criterion is to provide successful service during the entire talk session. Thus, successful call completion probability is a reasonable performance measure to be maximized. Maximizing call completion probability, P_c, is equivalent to minimizing the new call and handoff blocking probabilities of both classes. The average blocking measure, $\overline{P_b}$, to be minimimized is then defined as:

$$\overline{P_b} = 1 - P_c = \frac{\lambda_1}{\lambda_1 + \lambda_2} P_1 + \frac{\lambda_2}{\lambda_1 + \lambda_2} P_2 \tag{2}$$

where P_1 and P_2 are the total blocking probabilities for class 1 and 2 respectively:

$$P_1 = P_{n1} + P_{h1} \tag{3}$$

$$P_2 = P_{n2} + P_{h2} \tag{4}$$

2.2 Handoff Priority Blocking Measure

Carrying the call during the conversation through different cell areas without dropping, is of crucial importance for the cellular system design. Therefore, one might also use a performance measure where dropping a handoff call is *penalized* more severely than blocking a new call. This measure, in Equation 1 format translates to the following weighted blocking measure:

$$P_{hp} = [\frac{\lambda_1}{\lambda_1 + \lambda_2} \ \frac{\lambda_2}{\lambda_1 + \lambda_2}] \begin{bmatrix} P_{n1} \\ P_{n2} \end{bmatrix} + [\frac{\omega_1 \lambda_1}{\lambda_1 + \lambda_2} \ \frac{\omega_2 \lambda_2}{\lambda_1 + \lambda_2}] \begin{bmatrix} P_{h1} \\ P_{h2} \end{bmatrix} \tag{5}$$

where ω_i is the *relative penalty* for dropping a handoff call of class i i($\omega_i \geq 1$). Although not necessary, we will assume both classes have the same relative penalty, i.e.

$$\omega_1 = \omega_2 = \omega \tag{6}$$

This measure is simply the average blocking and dropping probabilities of both classes with dropping probabilities weighted ω times more heavily than the new call blocking probabilities; i.e., handoff blocking probabilities have priority over new call blocking probabilities.

This paper considers the performance of 1-dimensional linear multi-cell systems with given interference constraints under state-based call admission policies. The performance is defined as a weighted measure of new call and handoff blocking. A genetic algorithm is used to search for *good* (as opposed to optimal) policies. Our motivation for using genetic algorithms (GAs) lies in the limited size of systems which may be considered using standard procedures such as Markov Decision Processes (MDP). Specifically, GAs offer the possibility of systematically deriving policies for networks with large state spaces.

Section 2 explains the mathematical model of the system and defines the optimization problem. Then, the formulation of call admission policies and the Markov Decision Theory Approach for searching the optimal policy are presented in Section 3. Section 4 describes the genetic algorithm used for finding good call admission policies. Finally, we present the performance of the policies found using the genetic algorithm for some examples and show that the policies improve the performance of the system significantly compared with the admit-if-possible strategy.

2 Assumptions and Problem Definition

We model a 1-dimensional linear cellular network and find call admission policies which improve the performance of the system. The results are extended in [Yen] to ring structured networks as well.

The network consists of N cells, and there are C channels that can be accessed by all cells. We assume a particular channel can be used in a given cell only if that channel is not used in any of the K closest neighboring cells; i.e. *cliques* of mutually interfering cells cannot share any of the the same channels. In [Yen], we prove that a valid channel assignment scheme can be found for such 1-dimensional linear cellular systems if all cliques have less than or equal to C channels currently in use. We also assume any rearrangement during the channel assignment is possible if needed.

We accommodate two types of users with differing characteristics such as handoff rates and call holding times. Users, in turn, may request two types of service in a cell; a new call set up, or a handoff from one of the neighboring cells. A new call set up is considered as an *arrival* to the system. Each customer type has an independent arrival process. Thus, cell i is assumed to have two types of Poisson arrival processes with intensities λ_{i1} and λ_{i2} for new calls. Calls in progress are subject to handoff to any of the neighboring cells, and the time spent in each cell by an active call is assumed exponential with rates γ_{i1} and γ_{i2} for type-1 and type-2 calls respectively. Note that, with this model, we do not require any information about the underlying motion model related to the handoff process. Since the exponential model implies a bidirectional *Brownian Motion* around the borders, it is most appropriate for smaller cells, i.e. microcells or picocells. Nevertheless, it is a good starting point, and we will adopt it for evaluation of the systems.

Call holding times for the calls also follow the usual exponential distribution with means $\frac{1}{\mu_{i1}}$ and $\frac{1}{\mu_{i2}}$. Therefore, the total rate of departure for class j from cell i is given by: $\mu_{ij} + \gamma ij$.

The system is modeled by a 2N-dimensional Markov Chain where the state is described as the number of active callers of each customer type in each cell. The number of feasible states in the system depends on the number of channels and the interference constraint.

Near-Optimal Call Admission Policies for Cellular Networks Using Genetic Algorithms

A. Yener C. Rose

Dept. of Electrical and Computer Engineering
Rutgers University
PO Box 909
Piscataway NJ 08855
e-mail: yener@mogli.rutgers.edu
crose@mogli.rutgers.edu

Abstract

It is well known that in stochastic service systems such as telecommunications networks, selective denial of service based on system state can enhance performance. For example, in a cellular mobile network, preference might be given to handoff calls over new calls when the system is more heavily loaded. When the service center can be modeled as a markov process, there exist means to calculate the optimal call admission policy (Markov Decision Processes, MDP). However, for systems with a large number of states MDP is impractical since it requires solving large systems of linear equations. Therefore we seek methods of calculating *good* policies when calculating the optimal policy is computationally prohibitive.

To this end we consider application of Genetic Algorithms to the call admission problem in 1-dimensional cellular networks. We test our methods on smaller systems and then offer evidence that the approach is extensible to substantially larger systems.

1 Introduction

The increasing demand and limited bandwidth available for mobile communication services require efficient use of radio resources to avoid blocking of new calls and dropping of calls in progress. It has long been known that these blocking probabilities can be reduced by allowing each cell access to all channels available for the system for low and moderate loadings [BP89]. Most such schemes [ZY89][KG78] deny access to the system only if it is impossible to assign a channel to the call. However, if the system is shared by users with different characteristics (such as differing handoff rates or call holding times), the *overall* system performance can be improved by denial of service requests even when the excess capacity exists. Such selective denial of service based on system state is defined as call admission.

The multiple class queue call admission problem has been widely studied, primarily in the context of call admission to multirate circuit switched networks [Lip75] [Kau81] [GS83] [RT89b] [RT89a] [GS93]. Call admission to a single cell system has been considered in [RY93].