A Co-Evolutionary Approach to the Call Admission Problem in Telecommunications.

Anil Seth

Centre for Computational Neuroscience and Robotics
University of Sussex
Brighton BN1 9QH, UK
anils@cogs.susx.ac.uk

Abstract

An important and difficult problem in the management of telecommunications networks is how good policies can be developed for admitting or blocking customer call requests for a network of limited server capacity. This paper describes a new co-evolutionary approach to this problem, which describes customers and servers as individuals within a 'technological ecology', and models their interaction using a modified version of the Iterated Prisoner's Dilemma (IPD). The model is also deployed to tackle the more complicated 2-class problem, in which there are two customer populations, each with different priorities and service requirements.

For both classes of problem, good results are achieved in terms of low call blocking rates, even in the face of system noise and server failure.

Although the present research is exploratory in character, it is proposed that this co-evolutionary approach could eventually be deployed in real-time in a telecommunications context, allowing a network to adapt automatically to changes and problems as they arise. More immediately, it is suggested that this kind of system could be used in simulation off-line, and combined with some degree of executive control in a hybrid system. Such a technology would be within reach of current engineering viability.

1 Introduction

1.1 Network Design, Network Management

In telecommunications, problems associated with network design and management are of central importance. The design problem concerns the development of an effective network, comprising of links with given bandwidths, so that it is capable of simultaneously routing traffic demand (see [6] for a comprehensive review). The management problem, in contrast, concerns controlling such networks once they have already been built, and the call admission problem is a standard problem in this latter area.

The basic call admission problem (see e.g. [8]) concerns a set of customers who require access to a routing network. Access to the network is gained through servers, and the customers compete for these servers, with system performance evaluated through the average blocking probability of calls. This problem assumes that the network design problem has already been solved, and concentrates instead on the admission of traffic to such a network.

An extension to this basic scenario is the 2-class problem in which there are two customer populations, each with different priorities, demand characteristics, and service requirements. The system must then develop effective policies to deal with the more complex situations that can arise in these environments.

1.2 GAs and the Call Admission Problem

Rose and Yates [8] have demonstrated the utility of genetic algorithms with regard to the call admission problem. They discuss methods for evolving 'policies' for the servers which dictate whether they admit or refuse call requests. The policies that they evolve are *centralised* in the sense that they treat the entire network as a single functional entity, rather than as an agglomeration of potentially autonomous units. They present successful candidates that display near optimal blocking probabilities, and they consider the relative merits of different types of genotype encoding schemes for their policies.

Two criticisms of the above approach can nevertheless be levied. Firstly, the way in which an optimal policy is evolved for implementation across all the servers presumes that these servers have, and will always have, up-to-date, complete, and accurate information about the global state of the system (including the states of all the other servers). This is not only implausible in large networks, but renders the method difficult to extend, as the overall population of servers may well increase (or decrease) over time. The related (and perhaps more significant) problem is that if the system environment is noisy, or if servers become dysfunctional, then in a policy determined in advance that is reliant on globally coherent behaviour, there is no guarantee that system behaviour will robustly adapt to changing circumstances and continue to deliver effective management.

In the present paper, a co-evolutionary approach is developed which describes a telecommunications network as an artifical ecology, thereby devolving control and management to the level of the individual servers. In this way, policies for call admission and blocking are implemented on a purely local basis, with a globally coherent network policy emerging out of the combination of many local interactions. A major advantage of this approach (and hypothesis of this paper) is that since there is no complex centralised policy, neither network extendability nor network robustness in the face of server failure should present serious problems.

The approach advocated here, as with many other co-evolutionary investigations, takes as a starting point the model for interacting agents provided by the Iterated Prisoner's Dilemma.

2 Co-Evolutionary Foundations

2.1 The Iterated Prisoner's Dilemma

The Prisoner's Dilemma has long been established as a tool of great value in co-evolutionary investigations, [2][3][7]. Essentially, it provides a framework for modelling non-trivial interactions between agents, where the maximisation of individual short term gain minimises the collective welfare, as illustrated by the following anecdote:

Imagine that you and an alleged accomplice have both been arrested, accused of a heinous crime. You are held in separate cells, and upon interrogation you can either *cooperate* by denying all knowledge, or *defect* by implicating your accomplice. You have no idea what your accomplice will do, but if you both cooperate, you will both be released (the reward, **R**), and if you both defect, then both of you will be jailed (the punishment, **P**). However, if you defect and she cooperates, then you will recieve a payoff (the temptation, **T**) and she will go to jail for longer (the sucker, **S**). But if she defects and you cooperate, then you yourself are the sucker. The paradox is thus evident, - in a single meeting you will

	player 2 cooperates	*player 2 defects*
player 1 cooperates	1:R=3 2:R=3	1:S=0 2:T=5
player 1 defects	1:T=5 2:S=0	1:P=1 2:P=1

Table 1: Prisoner's Dilemma Scoring Table

always do best to defect, in doing so either recieving the monetary payoff or avoiding being the sucker. But of course the logic is the same for your alleged accomplice, and if you both defect then you will both do worse than if you had both cooperated (see table 1).

Cooperation is thus unlikely to arise in a one-shot Prisoners Dilemma, but if players can meet time and time again, and retain some memory of previous interactions, then cooperation on any given move does become a rational strategy. It is this Iterated Prisoner's Dilemma (IPD) that forms the core of the present study.

Many researchers have used genetic algorithms to evolve strategies to play the IPD (see e.g. [2],[3]). In these studies, as in the present model, the genotypes comprise of binary character strings representing policies for playing the IPD, with the length of the genotype determining the number of preceding moves (the game history) upon which each individual can base its strategy. It has been repeatedly demonstrated that cooperative strategies can and do arise and persist in artifical ecologies populated by these evolving strategies.

2.2 The Modified IPD

The present paper uses a modified version of the IPD in order to model the interactions between servers and customers in a telecommunications network. The way in which the IPD model reflects the functional constraints of the call admission problem is a major contribution of this paper, but is predicated upon an extension to the basic paradigm that is not unique to the present investigation, concerning *preferential partner selection.*

In the standard formulation of the IPD, interactions are arranged in a very orderly fashion, usually in a 'round-robin' tournament where every individual interacts once with every other on each iteration. The principle of preferential partner selection removes this constraint by allowing individuals to have some control over who they interact with, and this extension can lead to the emergence of interesting new dimensions of emergent behavioural structure.

Stanley et al. [9],[1] have published extensively on the formation of *social networks* in an IPD context where agents can choose who they would prefer to interact with, and refuse overtures from partners they consider unsuitable. Choice and refusal is accomplished with reference to continuously updated expected payoffs that each agent maintains for every other agent in the population (Stanley calls this an IPD/CR mechanism). They demonstrate that cooperation is evolved rapidly in such circumstances and they discuss the emergence of various kinds of metastable networks displaying distinct patterns of connectivity.

In the present investigation, a variant of the IPD/CR mechanism is implemented so that servers can decide which calls to admit and which to refuse, and also so that customers can have some control over which servers they attempt to access.

2.3 Modelling the Call Admission Problem

In the simple call admission problem, a set of customers making call requests has to be accommodated by a set of servers. Thus, instead of a single population, we now have a bipartite world (servers and customers), with a member of the customer population *not* representing a person with a phone, but instead a distribution node for a set of calls that must obtain access to a server network. These distinct populations interact with each other according to a model derived from the IPD/CR, but evolve separately.

The customers make offers to the servers, based on (initially equal) expected payoffs. The servers, who likewise have (initially equal) expectations of the customers, evaluate the offers they have received and accept a quota of the most preferable, refusing the rest. These refusals constitute part of the blocked call measure. The accepted offers are then played out as an interaction modelled by a single IPD iteration, with a defection by a server constituting a further blocked call. Defections by customers are not generally allowed[1]. And so it goes on - the expectations of the servers held by the customers are updated (and vice-versa)[2], and new rounds of offers are made and played out.

After a certain number of rounds, breeding takes place in the distinct populations to form the next generation, with offspring deriving from the most successful individuals (in terms of IPD score) constituting the new populations[3].

[1]Complete customer cooperation was enforced by ignoring the customer genotype when it specified defection. However, when defection *was* allowed (by not ignoring the genotype), there was no appreciable difference in model behaviour (customers still cooperated most of the time anyway). Customer defections may have an interpretation in the model through representing the use of servers without subsequent payment.

[2]The following equation is used to update expecations:
$exp[i+1] = \gamma.exp[i] + (1-\gamma).payoff$
where 'γ' is a memory weight, and 'payoff' represents whatever IPD score is awarded.

[3]If customer defections are not allowed, then the customer strategies do not really evolve at all; they will always cooperate regardless of the constitution of their genotypes.

	server cooperates	*server defects*
type 1 customer coop.	c:R=3 s:R=3	c:S=0 s:T=5
(type 1 customer defects)	c:T=5 s:S=0	c:P=1 s:P=1
type 2 customer coop.	c:R=5 s:R=5	c:S=-2 s:T=8
(type 2 customer defects)	c:T=8 s:S=-2	c:P=-1 s:P=-1

Table 2: Call Admission Model Scoring Table

The central aspect of the server is then a combination of the genotype specifying the call admission policy (described in more detail in the following section), and a set of expected payoff values for each of the customers, which together determine whether calls are blocked or not.

Servers are also characterised by their *capacity,* specifying how many calls they can accept each iteration, and *status,* reflecting the operational state of the server. A finite probability can be set for servers to fail during the course of a given iteration, and the *down-time* (the number of subsequent iterations for which the server remains dysfunctional) can also be modified.

The customers also maintain expectations of the servers, as well as genotypes for playing the IPD. Customers are further defined by arrival rates, which can vary from iteration to iteration according to a probability distribution, reflecting the fluctuating call request profiles encountered in real networks. The customers *have* to make the required number of offers to the servers (as specified by the arrival rate) - they are not allowed to remain inactive (as is the case in the standard IPD/CR). And customers also demand particular service rates, such that high service rates require more server capacity than low service rates.

For the more complicated 2-class problem, customer priority can be modelled by different customer types delivering different scores (and penalties) during each IPD interaction, (through the use of modified versions of the scoring table, see table 2). The two customer classes may also have different arrival rate and service requirements.

With the situation set out as above, co-evolution should then lead to the servers adopting effective policies to maximise their individual fitnesses, which should require cooperation (as in the IPD/CR) and hence a decent solution to the call admission problem. The effects of failure, system noise, and their consequences for behavioural stability can then be assessed in this context.

2.4 Genotype Encoding Scheme

The encoding scheme employed in the present model is an extension of a system employed by Lindgren[4] [4].

[4]Lindgren considered infinitely iterated IPD games with regard to the dynamics of species complexity, and used mathematical

At the heart of each individual is a genotype, consisting of a string of `c`'s and `d`'s, which determine a strategy for playing the IPD. The longer the genotype, the more it can be influenced by the past history of interactions with other individuals - and the servers maintain separate game histories for each customer that they interact with (and likewise for the customers).

When the game is being played, each time a previous move is considered, half of the genotype is (temporarily) thrown away; one half if the move had been cooperative, and the other if it had been a defection. In this way, a genotype of length 16 can encode a strategy with a memory of 4 prior interactions, (after cutting a string of 16 characters in half 4 times, you are left with just a single character).

However, the genotype must be lengthened in order to specify the initial moves up until the memory limit is reached. A memory 4 strategy would require an extra 9 alleles to code for the initial 3 moves before the final 16 alleles can be used (thereafter, any number of moves can be made with the final 16 alleles determining the strategy). In the present model, a genotype length of 127 characters is used, providing a memory of up to 6 prior interactions.

Each time an iteration of the game is played, the moves made are stored in a large history array so that they can be accessed the next time the two players meet. In the present model this array is centralised, but in principle each server could maintain its own unique smaller array, as the servers only need to access those parts of game history in which they themselves took part.

The way in which any given server/customer interaction proceeds is therefore determined by a combination of the genotypes of the server and customer (which specify strategies for any possible game history of up to 6 prior interactions), and the actual history of interactions (if any) between the particular server/customer pairing that is being considered.

3 Implementation

The implementation of the model here bears certain similarities to the IPD/CR model of Stanley [9], in that customers choose servers on the basis of the expected payoff values that they maintain[5]. But whereas in the standard implementation of IPD/CR the agents have to play as many rounds of IPD as they have tolerable offers pending, in the call admission model the total number of rounds playable is restricted by the server's capacity and the service requirements of the customer call requests. Thus the servers must rank the received offers and refuse (or block) the surplus.

Breeding is implemented with simple generational tournament genetic algorithms, but it is ensured that the customer and server populations do not interbreed. Fitness is determined by the total score on the modified IPD, and it is the genotype coding for the IPD strategy (i.e. the call admission policy) that is modified in the breeding process. Population statistics are also calculated separately for each distinct population.

The main procedure of the model is implemented as follows:

```
randomly initialise bipartite populations
FOR EACH generation
    FOR EACH iteration
        servers fail according to the failure rate
        customers choose most preferable servers and
            make offers (according to variable
            arrival rates)
        servers rate offers and refuse least
            preferable ones that exceed capacity
            (taking account of different service
            requirements in the 2-class problem)
        FOR EACH tolerable offer for each server
            one round of IPD is played (with or
                without noise), and expectations
                are updated
        ENDFOREACH
        game history array and various scores are
            updated, blocking probabilities are
            calculated
        each server status evaluated and brought back
        on line if sufficient down time has elapsed
    ENDFOREACH
    new generations are created through bipartite
        breeding and tournament GAs
    every so often population statistics are
        calculated, and presented with graphics
ENDFOREACH
```

The model was coded in ANSI C and executed on a Sun Sparc workstation.

4 Results

4.1 Overall System Performance

The system typically evolves to very stable situations very quickly, with both customer and server cooperation very near to maximum all the time. This is a promising first observation as it suggests that even when control is completely localised and devolved, and even when short-term benefits can be gained through 'antisocial' behaviour, the system as a whole rapidly reaches an equilibrium beneficial to both customers and servers.

analysis to describe the behaviour of the ecology. The present model not only has completely different objectives, but also actually instantiates the 'ecology', therefore necessitating considerable alterations to Lindgren's scheme.

[5]Stanley, however, considers a single population rather than a bipartite populations of servers and customers

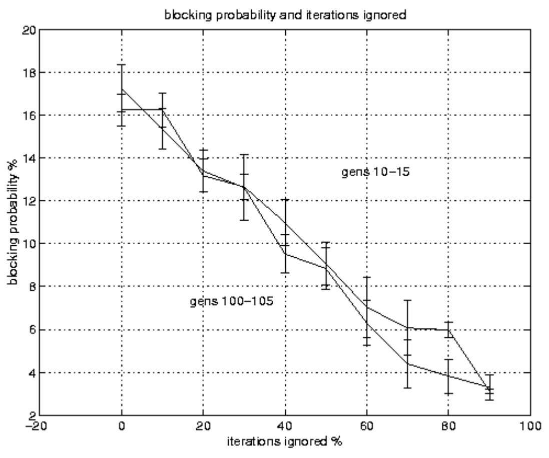

Figure 1: *the blocking rate falls off as each generation progresses (the horizontal scale determines the percentage of the earlier part of the generation that is igonred when calculating the statistic).*

The patterns of connectivity at the beginning of each generation are unpredictable due to the initial expectations all being equal, providing no basis for partner choice. But as each generation wears on, stable patterns of interaction develop, - the same customers being admitted by the same servers, with occasional alterations. This stability persists over many generations.

4.2 An Easy Problem

The model was initially explored with a very simple problem - with 20 customers (of a single type) and 10 servers, allowing each server to deal with 2 customers at each time step. With neither noise nor server failure, the blocking probability was typically very stable and very low. However, since the customer/server expectations are reset at the beginning of each generation, the blocking probability was always highest at these times, and would then fall off as the generation progressed. Figure 1 illustrates how the blocking probability is lower if earlier iterations are ignored (in a given generation), and suggests that in a steady-state system, without generational upheavals, the actual blocking probability would be very low indeed. Figure 1 also illustrates that the situation after 100 generations is very similar to that after just 10, implying that evolution acts very quickly to deliver a stable situation.

4.3 Noise and Failure

The effects of noise and server failure were also explored in the context of the simple problem outlined above, with noise referring to the probability with which the opposite action to that specified by the genotype of the server or

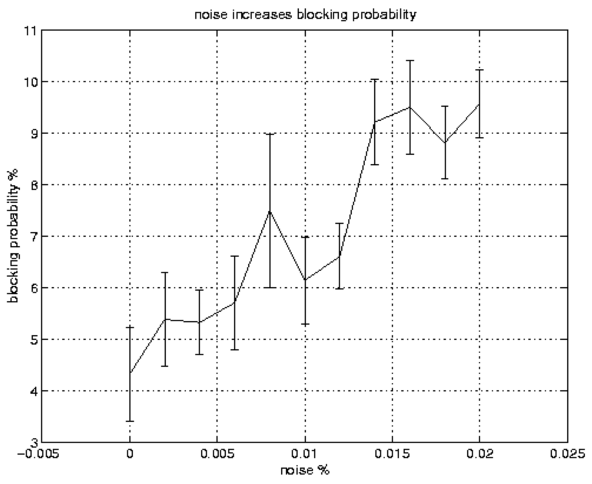

Figure 2: *noise degrades performance on the call admission problem, but in a non-catastrophic fashion.*

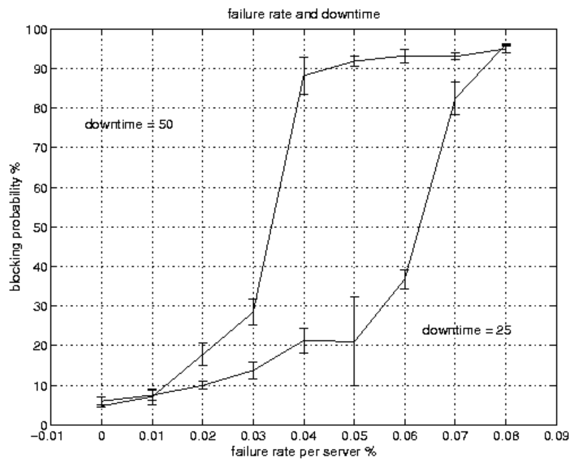

Figure 3: *low levels of server failure are successfully accommodated by the model, but high levels of server failure lead to system collapse - the critical point is dependent upon server down-time.*

customer is performed. In most cases this simply means the probability with which calls are accidentally blocked (although it is also the probability with which blocked calls are accidentally processed). Figure 2 shows that whilst system performance is degraded with increasing noise levels, this degradation is not catastrophic.

Figure 3 examines server failure, demonstrating that with low failure rates the system dynamically reconfigures the flow of call admissions in order to cope - but after a critical stage is reached this process breaks down and almost all calls are blocked. This critical point is dependent upon the server downtime; the longer each server is down for, the earlier the system performance collapses.

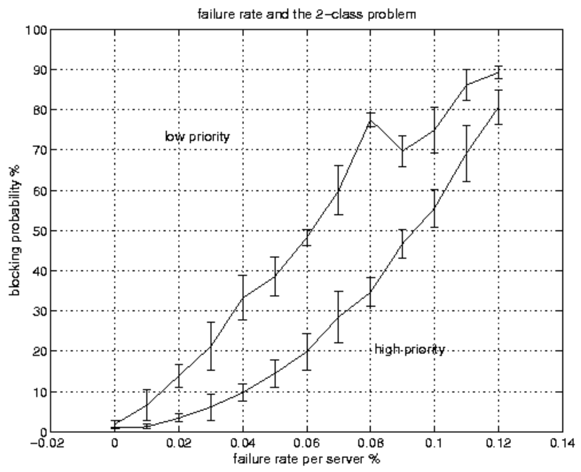

Figure 4: *low priority customers suffer more than high priority customers when servers fail in the 2-class problem.*

4.4 The 2-class Problem

Many runs were executed with various different parameters set for the relative numbers of high and low priority customers, their respective call arrival rates, service rates, and server capacities. In all cases, the general policy emergent in the system was to process all high priority calls, and then as many low priority calls as possible. If calls had to be blocked, the low priority customers suffered (even when high priority calls required more server capacity).

The preference for high priority customers is undeniably a rational policy, and is also evident in how the system reacts to server failure. Figure 4 illustrates that as server failure becomes more predominant, the low priority calls suffer more than those of high priority.

4.5 Extendability

The ability of the system to adapt well to extension was assessed through the introduction of additional servers. Ten extra servers were introduced after half the total number of generations had elapsed. The new servers were genotypically initialised randomly, just like the original population - and all had equal expected payoff values for all the pre-exisiting customers (as the customers themselves did for all the new servers). Nevertheless, the system proved able to dynamically reconfigure itself in order to take advantage of the extra capacity afforded by the new servers.

Figure 5 illustrates a single customer class call admission problem, in which the initial capacity is insufficient to admit all the call requests. After 50 generations, 10 new servers are introduced and the call blocking rate immediately shrinks. There is also a notable spike in the diversity of server genotypes as the initially random genotypes of the new servers come into play.

In the 2-class problem, a lack of sufficient server capacity is initially manifest in a high blocking rate for low priority calls (high priority calls are still getting through). With the addition of new servers, the difficult situation for the low priority calls is considerably ameliorated, as illustrated in figure 6.

5 Conclusions

The model described in this paper is possibly one of the first pure Alife models that has been blooded for a telecommunications application. The success enjoyed suggests that co-evolutionary/game-theoretic approaches to technological management, - either through the instantiation of completely autonomous interacting agents or in simulation as part of a hybrid system - herald a bright future.

In the context of the call admission problem, the application of the modified IPD/CR delivers good results in terms of call blocking probabilities and, in all cases, both servers and customers evolve highly cooperative strategies. Environmental noise proved detrimental to system performance, but in a non-catastrophic way. The system was also able to maintain stability in the face of server failure - although after a critical point blocking rates soared to nearly 100 percent (dependent on parameters such as downtime). Similarly, dynamic reconfiguration enabled the system to take efficient advantage of additional servers if they were introduced at later stages. No executive interference was necessary for this process to take place. This effectively demonstrates a major advantage of a continually evolving distributed system over a preprogrammed centralised system, in which central control programs would normally have to be updated carfeully in order to maintain effective management over any new situations.

With two customer classes, a simple policy of processing all high priority calls and as many low priority calls as possible was adopted, across a wide range of parameters. This is certainly a rational, and perhaps the *most* rational policy to adopt in these kinds of circumstances. And with server failure, the low priority calls bore the brunt of the extra call-blocking much harder than the high priority calls. However, the introduction of additional servers, in all cases, ameliorated the blocking of these low priority calls.

In sum, the model presented here delivers a new, robust, and effective principle for developing and deploying call admission strategies. The main question which then arises concerns the relationship that this model should have with existing and near-future telecommunications technology. There are two primary alternatives. In prin-

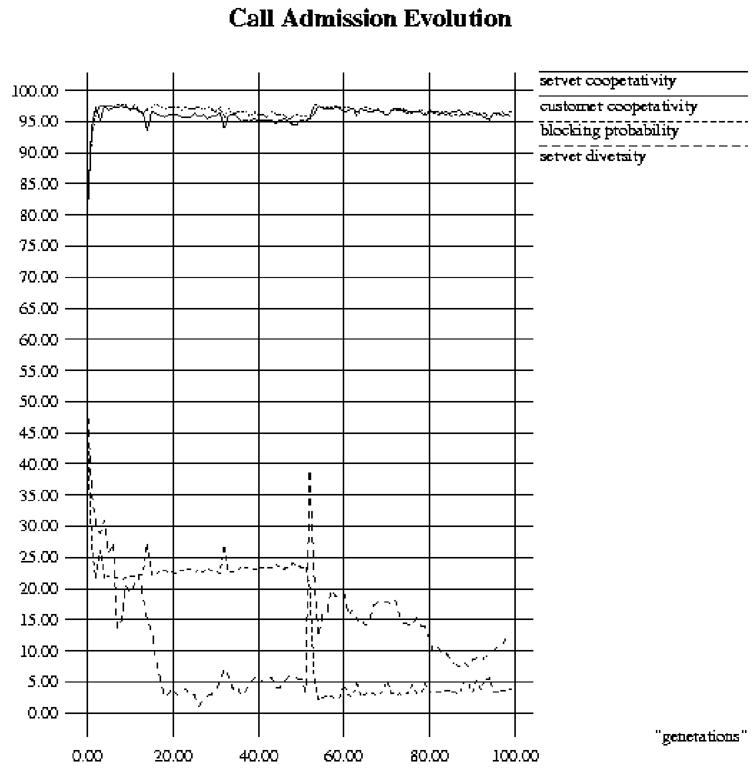

Figure 5: *introduction of extra servers leads to dynamic reconfiguration of the system and a concomitant reduction in the blocking rate.*

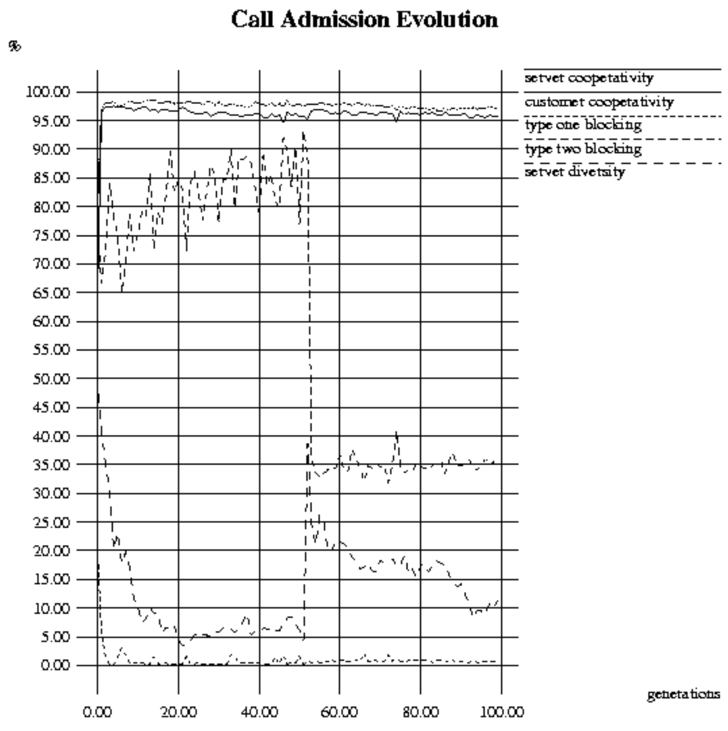

Figure 6: *introducing extra servers reduces the high blocking rate experienced by low priority customers in the 2-class problem.*

ciple, the system could be located on-line, with call admission strategies evolving in real-time in a real network. This would be a very strong interpretation of the work presented in this paper. Alternatively, the system could be used in simulation off line, and combined with a certain minimal degree of central control in a hybrid system.

There are two further possibilities here. The most simple and weakest interpretation would be to use simulations solely to guide designers of network technology in their quest to understand how networks might behave in all kinds of circumstances, much as car designers can now use virtual wind tunnels. A more interesting possibility would be for the policies evolved off-line to be periodically downloaded into the real-world system, and for real-world changes to be uploaded into the simulation. In this way, executive control can still be maintained over the global behaviour of the system, whilst allowing most of the advantages of the co-evolutionary system to manifest themselves, albeit with some small time lag.

One point must be raised in order to qualify the conclusions laid out above. Whilst it is true that the co-evolutionary approach does eliminate the need for executive control over the policies implemented by the system, it is *not* true to say that executive interference is *completely* abolished by this method. Both the system of offers and refusals, and the implementation of the genetic algorithm itself, require the synchronised and centrally co-ordinated interaction of individuals within the system. Practically, this is unlikely to present a problem, and indeed may even be of benefit in that it provides another way in which overall central control can be maintained without diminishing the benefits that accrue from this kind of system.

6 Future Work

6.1 Integration with Executive Control

Integration with a central controller is an obvious avenue for development, and would help bring ecological technology (if of a diluted variety) within the domain of engineering viability. Hybrid systems of the type mentioned in the previous section would provide a suitable direction for implementation.

6.2 Reinforcement Learning

Considerable interest has recently arisen in the possibilities afforded by the implementation of reinforcement learning techniques in telecommunications (see e.g. [5], perhaps since this provides another way of incorporating individual-level adaptive techniques without the abdication of all central control. The ecological call admission model itself makes use of reinforcement learning, in the way in which the servers and customers update their expected payoff values over time. Thus, learning theory could well provide a fruitful territory for bridging the chasm between centralised and ecological control methodologies that would reward further exploration.

6.3 Lamarckian Inheritance

Lamarckian inheritance could be introduced into the system, so that expected payoffs are not periodically reset but are passed down from parents to children. This would suggest the use of a continuous GA instead of a generational GA, and would clearly alleviate the problem of the surge in blocking rates at the start of each new generation.

Acknowledgements

This work owes a great deal to my supervisors Phil Husbands and Inman Harvey, and also to Tony White of Nortel Ltd. This work was funded in part by the ESPRC and in part by Nortel Ltd.

References

[1] D. Ashlock, M.D Smucker, E.A. Stanley, and L. Tesfatsion. Preferential partner selection in an evolutionary study of prisoner's dilemma. Economics report 35, Iowa State University, 1994.

[2] R. Axelrod. *The Evolution of Cooperation.* New York : Basic Books, 1984.

[3] C. Langton. *Artificial Life; an Overview.* MIT : Bradford Books, 1995.

[4] K. Lindgren. Evolutionary phenomena in simple dynamics. In C. Langton, J.D. Farmer, S. Rasmussen, and C. Taylor, editors, *Artificial Life II.* Addison-Wesley, 1991.

[5] M.L. Littman and J.A. Boyan. A distributed reinforcement learning scheme for network routing. In *Proceedings of the First International Workshop on Applications of Neural Networks to Telecommunications*, pages 45–51, 1993.

[6] T.C. Magnanti and R.T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18:1–55, 1984.

[7] R. May. *Stability and Complexity in model ecosystems.* Princeton University Press, Princeton, NJ, 1973.

[8] C. Rose and R. Yates. Genetic algorithms and call admission to telecommunications networks. *Computers and Operations Research*, 23(5):485–499, 1996.

[9] E.A. Stanley, D. Ashlock, and M.D. Smucker. Iterated prisoner's dilemma with choice and refusal of partners: Evolutionary results. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, *Advances in Artificial Life : Lecture Notes in Artificial Intelligence.* Springer-Verlag, 1995.