Adaptation in an Active Services Network

Ioannis Liabotis
Department of Electronic and Electrical Engineering,
UCL
Telecommunications Systems Group
{i.liabotis@ee.ucl.ac.uk}
Supervisor: Dr. Lionel Sacks

Abstract: *Active services networks will give to users the flexibility to have access to custom services created by them or third party service providers. The Internet will become much more attractive but also services will turn out to be more versatile. The complexity of such a system is expected to be very high making traditional network management strategies unable to cope with the organization of the whole network. New adaptive techniques must be introduced in order to handle the rapid and many small changes in the network state. Genetic algorithms and biological inspired models seem to make the network adaptive to unpredictable changes. Some simulation results prove that those models are promising for the management of the future networks.*

1. Introduction.

The number of services that today's Internet is offering is growing exponentially. Considering an active services network [1] that will give users the ability to add new services on demand and implement currently unknown protocols, the complexity of the internet is going to be unpredictably high. Active services are based on programs supplied by the users of the network. Those programs will run on equipment owned by the operators, enabling users to have access to custom services that will be managed by them without the operators' intervention. This will give flexibility to the user making the network even more attractive and the services more versatile. Such an approach expands the state space of the network making the precise identification of the current state impossible. Demand patterns are not easily predicted or are even unpredictable. Furthermore in an active services network resource requirements typically come in three categories: bandwidth, memory, and processing. Thus there is an increase in the parameters that must be optimized in order to achieve an efficient network operation, making the solution of the management problem harder to identify.

2. The management problem.

The current approach in network management considers the network as a system consisting of network elements that have a fully declared state. The used architecture is operator centric following the centralized or hierarchical approach. Such a system can deal with the few and slow changes in the network status because the communication and computational overhead increases as the changes become faster and more frequent. Thus, this approach does not scale in time as the system requirements move from large time scales to smaller ones. In an active services network, except from managing network devices we also have to manage services. Hence, the entities involved are increasing in number, the state of services is changing faster and more frequent.

On the other hand the dynamics of the resource usage and so the overall system evolution becomes unpredictable due to the non-linear behavior. Current research in WWW [2] traffic has shown that user demand has fractal statistics. As a consequence of this non-Gaussian behavior, average values from measurements do not give valid predictions for the patterns of the users demand.

The unpredictability of the future network status and the increased computational complexity, necessitate a new measurement and control methodology. Time scale invariant measurements are required. The fractal nature of the usage patterns allows the computation of the Hurst parameter using measurements of an attribute in one time scale [3]. Then it is possible to calculate the values of the same attribute, in different time scales. Thus, with one measurement methodology different levels of abstraction are obtained. On the other hand, the fast and frequent changes in the service plane, require local control using simple and fast algorithms with the ability to adopt to unpredictable circumstances. Genetic algorithms are adaptive methods that have been successfully applied to solve optimization problems of this type [4]. Over many generations, natural populations evolve according to the principles of natural selection and "survival of the fittest" [5]. By mimicking this process genetic algorithms evolve solutions to optimization problems. In nature, individuals in a population compete with each other for resources such as food or water. Those individuals which are most successful in surviving will have relatively larger number of offsprings, while poorly performing individuals will produce few or none offsprings. Bacteria use asexual reproduction and have the ability to reproduce very soon after their birth. These properties of bacteria are useful in an environment that very fast adaptation is needed. An active services network can be modeled as a population of bacteria trying to maximize their fitness while exchanging information through bacterial recombination [6].

3. The model.

The first results of the application of adaptive techniques to network management appear in the work of I. Marshall and C. Roadnight in [7]. The presented model is a simplification of the model that is described in their work. The network is represented by an array of nodes each of those randomly selected as alive or dormant. Alive nodes can process active services, while dormant ones only forward requests to neighbors. Each node of the network has some genetic material that represents the rules that govern its behavior. These rules apply to the execution of the various services that the node is called to service. Initially only two performance parameters are considered: The number of services that are in the server queue, and the load of the processor. Both parameters are represented as integers.

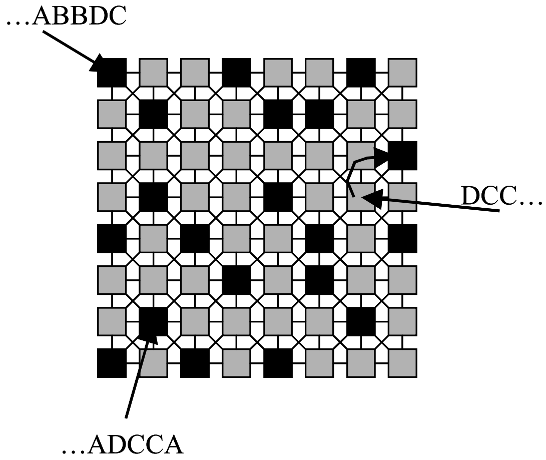

Figure 1: The network model.

The services are denoted using an alphabet letter. So for each service there is a rule of the form *{s,q,l}*. Each rule is interpreted as follows: Service s can be processed if the current queue length is less than *q* and the current processor load is less than *l*%. In the implementation, four services are supported, named *A, B, C* and *D*, while the max queue length equals to 200 and the maximum processor load is 100%. Each server has information for all the available services. As a future extension the introduction of new services should be supported. During the initialization process the values *q* and *l* for each service are randomly selected.

Request for services come randomly to the nodes as variable length strings, containing a random selections of the services *A, B, C* and *D*. Alive nodes put the services on their queue while dormant nodes forward the requests randomly to any neighbor. Figure 1 illustrates the above model, where the alive nodes are black, while the dormant are white.

Time is divided in periods called epochs. In each epoch a node is able to service 4 requests or forward 32. If a node services a request then the load of its processor increases. The load of the processor is calculated as follows: load = 0.8*old_load + 0.2*25*(number of serviced requests during the current epoch). Where old_load is the processor load calculated in the previous epoch.

Bacterial behavior is achieved by mutation, bacterial recombination and reproduction. Mutation is implemented by altering a value in the thresholds q or l of the rules that govern the node behavior, after the end of an epoch. For example, if a rule is {'A',80,30} it might become {'A',100,30} or {'A',80,70}. Bacterial recombination is achieved by transferring one or more rules from a healthy node to one of its neighbors. A node is considered healthy if after the end of an epoch it has managed to service 4 requests. Finally if the processor load of a node is high for a long period of time (many epochs) then the whole node is reproduced to a nearby node that is dormant. On the other hand if the processor load of a node is very low for a long period then it goes to the dormant state.

4. Results.

The experiment involved a network of 900 (30x30 grid) nodes having 25% probability of being alive. The scope of the simulations was to measure the quality of service that this network was providing to the services. Two methods of measuring the quality of service were used. The first one was the calculation of the mean length of the queues. The second one, was the calculation of the mean of services that were forwarded by the alive nodes. The behavior of the quality of service measures was examined, changing the demand on the network. Three different levels of demand were applied.

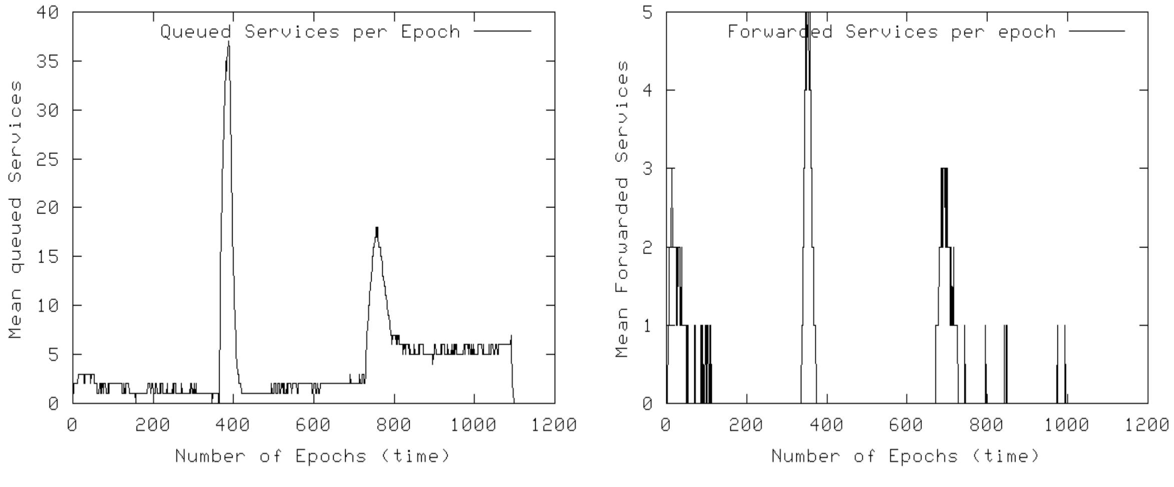

Figure 2: a) The mean queue length, b) the mean services forwarded, varying the demand level.

Figure 2, illustrates the quality of service levels, changing the levels of demand from 1X to 4X and finally to 8X. The moment that the level of demand increases there is a sudden increase in the mean value of services waiting in the queues, as well as in the mean value of forwarded services. But very soon the network adapts changing the number of alive nodes and the values of the rules parameters. The result is that the quality of service level returns to acceptable values. The time needed for the network to adapt is approximately 50 epochs. This demonstrates that the active services network can adapt rapidly to the fast changes in the users' behavior.

So far two parameters (q and l) were used to control the behavior of each node. The simulation was repeated using two more parameters, the memory utilization *m*, and the bandwidth utilization *b*. The calculation of those parameters was similar to the calculation of the processor load. Again the system managed to adapt after sudden increases in users demand. As a consequence of the usage of more control parameters, the number of epochs that where needed in order for the network to return to its stable state increased. In a more realistic simulation more parameters will be needed in order to control the network behavior and therefore the time needed for the whole system to adapt will depend on their number. Introducing learning techniques such as globally accessed gene pools, the system will be able to adapt faster to the changes of its environment.

From the simulation results, becomes clear that without any central control, active nodes, trying to maximize their own fitness, manage to maximize the quality of service offered to a large community of users as well.

5. Conclusions.

The Internet can be considered as a system where the behavior of the whole is much more complex than the behavior of its parts. Genetic algorithms and biological models reproduce the behavior of living organisms, giving mechanisms for reproduction, mutation, and death. A great advantage of living organisms is their ability to adapt to new and unpredictable circumstances. Network nodes can be modeled as organisms that compete for food that is represented by the demand that comes from the users. The node behavior is governed by simple rules regarding the services and the procedures of mutation and bacterial recombination. Initial results using this type of network model, show that adaptive management can meet the needs of an active services network.

References.

[1] I W Marshall, J Crowcroft, M Fry, A Ghosh, D Hutchison, D J Parish, I W Phillips, N G Pryce, M S Sloman, D Waddington "Application-level programmable internetwork environment". In BT Technology Journal, Vol. 17 No. 2 APRIL 1999.

[2] M. Crovella, A Bestavros "Self Similarity in World Wide Web Traffic: Evidence and Possible Causes", IEEE/ACM Transactions on Networking, 5, 6, Dec 1997, pp 835-846.

[3] Young. J, Sacks. L, Roadknight. "An Exploitation of the Dynamics of Auxiliary Network Resources", WTC/WSS2000

[4] D. Goldberg "Genetic Algorithms in Search Optimization and Machine Learning" Addison Wesley, 1989.

[5] C. Darwin "The Origins of the Species by Means of Natural Selection" New american Library, New York.

[6] I. Harvey "The microbial Genetic Algorithm" ftp://ftp.cpgs.suxs.ac.uk/pub/users/inmanh/Microbe.ps.gz, 1996.

[7] I.W.Marshall, C.M.Roadknight "Adaptive management of an active services network" To appear in BT Technol J special issue on "Biologically inspired computing", 18, 4, October 2000