Emergent Quality of service - A Bacterium Inspired Approach.

Ian W Marshall and Chris Roadknight
BT Adastral Park, Martlesham Heath, Ipswich, Suffolk, UK IP5 3RE
ian.w.marshall@bt.com

Abstract

A possible model for future network quality of service control is proposed. This is based on a community of bacterial strains, each organism handling network requests in the same way as bacteria metabolise energy sources. This model makes use of the unique methods that bacteria use to transfer and share genetic material, to create a more robust solution to the service provision problems associated with future data networks.

1. Introduction.

Despite the fact that existing Internet performance is inadequate for "real-time" communication, the number of services offered on the existing Internet is growing exponentially. As a result the Internet is probably one of the most complex machines built by mankind. The number of possible failure states in a major network like the Internet is so large that even counting them is infeasible. Deciding the state that the network is in at any time, with great accuracy, is therefore not possible. In addition, data networks such as the Internet are subjected to a mixture of deterministic and stochastic load [PAX95] [GRI97]. The network's response to this type of traffic is chaotic [ABR95], i.e. the evolution of network state is highly divergent, and accurate predictions of network performance require knowledge of the current state that is more accurate than can be obtained. This implies that current research, which focuses on enabling routers to adjust their forwarding behaviour to meet end to end Quality of Service (QoS) requirements, are likely to have limited success since the routers cannot predict the state of the network at remote points on the desired path. For example egress bursts originating from concurrent demand for a given web page cannot always be prevented by admission control at the network ingress points, since the demand at each ingress could be low, and the ingress points cannot know the total demand level.

Active networking [COV99] takes a different approach. The aim is to enable network clients to easily add new service components to network devices. Simplicity is achieved by restricting the action of the service components to a relatively small virtual network configured to meet the clients needs. Despite the virtual networking overhead, the resulting service performance will often be significantly better than if the component executed in the user's end systems (as at present). For example, a network based conference gateway can be located so as to minimise the latency of the paths used in the conference, whereas an end system based gateway will usually be in a sub-optimal location. Users of an active network supply the programs and policies required for their custom services in transport packets alongside their data.

However, active networking will increase overall network complexity, since it deliberately removes restrictions on what network devices can be used for, and therefore on their possible state. In other words active networking will introduce significant new degrees of freedom and make consistent QoS even harder to obtain.

Future networks, with more intelligence, will be even more complex and less tractable. A network control paradigm is therefore required that can maintain network performance in the face of fractal demands without detailed knowledge of the state of the network, and can evolve to meet unanticipated demands in the future.

Biologically inspired algorithms (eg. Genetic Algorithms, Neural Networks) have been successfully used in many cases where good solutions are required for difficult[1] problems of this type [ROA97] [GOL89]. They simulate evolutionary procedures or neural activation pathways in software, these then acting as problem solving tools. They can do this because:

[1] Here, the term 'difficult' is used to represent a problem that is computationally infeasible using brute force methods.

a. They take a clean sheet approach to problem solving.
b. They learn from successes and failures.
c. Due to multiple adaptive feedback loops, they are able to find optima in a fractal search space quickly.

The Darwinian mechanism of evolution involves a simple 'survival of the fittest' algorithm [DAR59]. While this undoubtedly served as a superb maxim for life on a slowly changing world, the lack of intra-generational exchange of information has obvious drawbacks when factors effecting fitness are varying more rapidly as in the Internet.

Bacteria are a set of metabolically diverse, simple, single-cell organisms. Their internal structure is simpler than most living cells, with no membrane-bound nucleus or organelles, and short circular DNA. Bacterial evolution 'transcends Darwinism' [CAL97], while asexual reproduction ensures survival of the fittest, a more Lamarkian[2] mechanism of evolution occurs in parallel, with individuals capable of exchanging elements of their genetic material during their lifetime. This plasmid migration allows much quicker reaction to sudden changes in influential environmental factors. These processes are commonly called adaptation. When a population of E. Coli is introduced to a new environment adaptation begins immediately, with significant results apparent in a few weeks [LEN94].

2. Proposed control Algorithm.

Our proposed solution makes each node within the network responsible for its own behaviour. The network is thus modelled as a community of cellular automata. Each member of the community is selfishly optimising its own (local) state, but this 'selfishness' has been proven as a stable model for living organisms [DAW76]. Partitioning a system into selfishly adapting sub-systems has been shown to be a viable approach for the solving of complex and non-linear problems [KAU94]. Figure 1 shows a diagram of a future network, some nodes are switched on (solid borders) some are switched off (dashed borders).

[2] Lamark was an 18th century French scientist who argued that evolution occurs because organisms can inherit traits, which have been acquired by their ancestors during their lifetime.

In this paper we describe an implementation that supports four services; A, B, C, D. The control parameters given below are examples provided to illustrate our approach. Optimisation of the algorithm for a particular application scenario is the subject of ongoing work. Our current implementation has 400 vertices connected on a rectangular grid. The system is initialised by populating a random selection of vertices with live nodes.

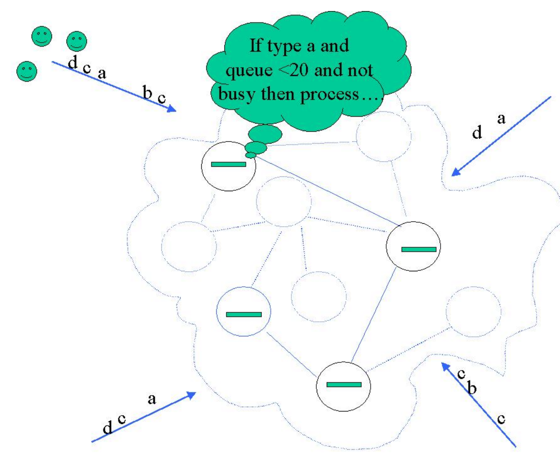

Figure 1. Schematic of proposed future network structure.

Each node has an amount of genetic material that codes for the rule set by which it lives. The initial nodes have a random selection of genes. This rule set defines how each node will handle requests for service. Currently each rule takes the form {x,y,z} where:
x. is a character representing the type of service requested
y. is an integer between 0 and 200 which is interpreted as the value in a statement of the form "Accept request for service [Val(x)] if queue length < Val(y)".
z. is an integer between 0 and 100 that is interpreted as the value in a statement of the form "Accept request for service [Val(x)] if busyness < Val(z)% "
Each node can accommodate up to 4 active rules.

Requests are input to the system by injecting sequences of characters (representing services) at each vertex in the array. If the vertex is populated by a node, the items join a queue. If there is no node the requests are forwarded to a randomly selected nearby node. Each node evaluates the items that arrive in its input queue on a FIFO principle. If the request at the front of the queue matches an available rule the node is rewarded and the request is deleted. If there is no match the request is forwarded and no reward is given. Only four requests can be processed per measurement period (epoch). The more time a node spends processing requests, the busier it is seen to be. The busyness is calculated by combining the busyness at the previous epoch with the busyness for the current epoch in a 0.8 to 0.2 ratio. For example, if the node has processed three requests this epoch (25 points each) it would have 75 points for this epoch, if its previous cumulative busyness value was 65 then the new cumulative busyness value will be 67. This method dampens any sudden changes in behaviour.

If we add rules for reproduction and evolution, and plasmid migration, it becomes possible to envisage each node as a bacterium and each request for a service as food. This analogy is consistent with the metabolic diversity of bacteria, capable of using various energy sources as food and metabolising these in an aerobic or anaerobic manner.

Genetic diversity is created in at least 2 ways, mutation and plasmid migration. Mutation involves the random alteration of just one value in a single rule, for example:

"Accept request for service A if node < 80% busy" to "Accept request for service **C** if node < 80% busy" or "Accept request for service A if node < **60**% busy".

Plasmid migration involves genes from healthy individuals being shed or replicated into the environment and subsequently being absorbed into the genetic material of less healthy individuals. If plasmid migration doesn't help weak strains increase their fitness they eventually die. If a node acquires more than 4 rules through interchange the newest rules are repressed (registered as dormant) so that no more than four rules are active. Currently, values for queue length and cumulative busyness are used as the basis for interchange actions, and evaluation is performed every five epochs.

If the queue length or busyness is above a threshold (both 50 in this example), a random section of the genome is copied into a 'rule pool' accessible to all nodes. If the node continues to exceed the threshold for four evaluation periods, it replicates its entire genome into an adjacent vertex where a node is not present. Healthy bacteria with a plentiful food supply thus reproduce by binary fission. Offspring produced in this way are exact clones of their parent. This replication is obviously exponential and enables rapid adaptation to load increases.

If the busyness is below a different threshold (e.g. 10), a rule randomly selected from the rule pool is injected into the node's genome. If a node is 'idle' for three evaluation periods, its active genes[3] are deleted, if dormant genes exist, these are brought into the active domain, if there are no dormant genes the node is switched off. This is analogous to death by nutrient depravation.

So if a node with the genome {a,40,50/c,10,5} has a busyness of >50 when analysed, it will put a random rule (e.g. c,10,5) into the rule pool. If a node with the genome {b,2,30/d,30,25} is later deemed to be idle it may import that rule and become {b,2,30/d,30,25/c,10,5}.

3. Initial Results.

A visualisation environment was created for our implementation. The environment provides an interface where load and other parameters can be varied in many ways, thereby stressing the system in a flexible manner. For instance, the ratio of requests for the 4 services can be made to vary over time, as can the overall number of requests per unit time. A 'petri dish' that can accommodate up to 400 bacteria/nodes was used to display the system state. Figure 2 shows what happens when an initial low load is increased and then reduced. Each strain that handles a single type of request is represented in the top image by a shade (White, Grey, Dark Grey). Strains with the ability to handle more than one service request type are allocated intermediate shades. When the load increases (centre image), the existing colonies increase in size, and new colonies appear due to mutation. Some of these thrive (eg. colony in centre of dish). The third image shows the response to a decrease in load. As in real communities a decrease in food causes a large amount of cell death, but also in increase in diversity (shown by the increase in the range of shades) as more plasmid migration occurs.

[3] Only some of the genes/rules in the genome are expressed, some lie dormant and are not used for decision making, this means they are not selected for.

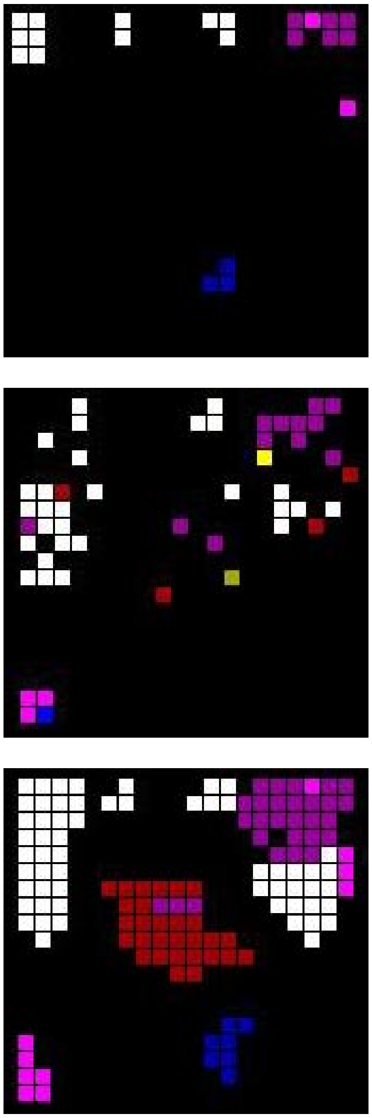

Figure 2. Three Stages of Bacterial Growth. Top, initial low load. Middle, response to increased load. Bottom, response to subsequent decrease in load.

A quality of service (QoS) measurement was introduced that measured the performance of the network. We measured the average age (expressed in epochs) of all requests on the network. The QoS was measured over time as the load on the network was increased in a series of significant steps, 1, 4, 12, 30, 45, 60 X the initial load. Figure 3 shows how the QoS reacts to these increases. At 60X the initial rate, the network is nearing its saturation point, yet performance has degraded very little. A short period of worsened service is observed as the network adapts in both size and heterogeneity, followed by a return to more acceptable QoS.

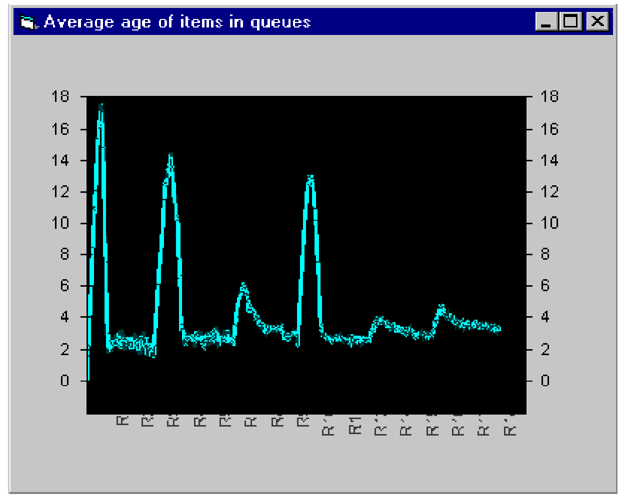

Figure 3. QOS levels under sequential changes in load.

4. Emergent approach to differential QoS

Currently there is great interest in enabling the Internet to handle low latency traffic more reliably than at present. Many approaches, such as intserv, rely on enabling the network to support some type of connection orientation. This matches the properties of older network applications, such as telephony, well. However it imposes an unacceptable overhead on data applications that generate short packet sequences. Given that traffic forecasts indicate that by the end of the next decade telephony will be approx 5% of total network traffic, and short data sequences will be around 50% of network traffic, it does not seem likely that connection orientation will deliver optimal results.

A recent alternative has been to propose differentiated services, an approach that is based on using different forwarding rules for different classes of packet, and maintaining the properties of the best class by admission control at the ingress to the network. There are difficulties however.

- Flow based admission control does not work well with short packet sequences
- The proposed algorithms assume Poisson burst intervals when real traffic is in fact fractional Gaussian and much harder to predict
- The performance benefits can only be obtained if the distribution of demand is such that only a small proportion of the traffic wishes to use the better classes

- The proposed classes typically propose a low loss, low latency class that uses a disproprtionate proportion of the available network resources

We propose a new approach, based on our emergent control algorithm. Users can request low latency at the cost of high loss, moderate latency and loss, or high latency and low loss by adjusting the ttl of the packets they send. To maintain reasonable statistics the ttl adjustment should be automated as part of the protocol stack (i.e. if loss is experienced the stack increases its expectation of latency). Two adaptations are possible; either the application sends less packets or the application persists until an application specific latency cut-off is reached and then terminates the session. Services such as telephony would use a low latency/high loss transport regime. This would require the application to be more loss tolerant than at present, however as mobile telephones demonstrate this is not hard to achieve. Interoperation with legacy telephones would be achieved by running loss tolerant algorithms (e.g. FEC) in the PSTN/IP gateway. Network performance is obtained by enabling routers to retain options in their routing table for fast path, medium path and slow path (i.e. high loss medium loss and low loss).

For this approach to work well the properties of the routers must adapt to local load conditions. Fast routers have short queues and high drop probabilities, slow routers have long queues and low drop probabilities. If most of the traffic is low latency the routers should all have short buffers and if most of the demand is low loss the routers should have long buffers. Adaptation of the buffer length can be achieved using an additional gene in our bacteria, and penalising routers by reducing their fitness whenever a packet in their queue expires. We will shortly demonstrate that this approach is feasible and begin to identify favourable operating parameters.

5. Conclusions.

Our initial results show that because of the long-term self-stabilising, adaptive nature of bacterial communities, a bacterial type network control algorithm might be a suitable approach to creating a stable network of autonomous nodes that can offer consistent end to end QoS. The methods used for adaptation and evolution are still in their infancy and the relative merits of different flavours of adaptation will need to be investigated further.

6. References.

Abrams M, Standridge C, Abdulla G, Williams S and Fox E. Caching Proxies: Limitations and Potentials. Proc. 4th Inter. World-Wide Web Conference, Boston, MA, Dec. 1995.

Caldwell D, Wolfaardt, Korber D and Lawrence J. (1997) Do Bacterial Communities Transcend Darwinism? Advances in Microbial Ecology. Vol 15. p.105-191

Covaci S (ed) "Active Networks", LNCS 1653, Springer (1999)

Darwin C. (1859) The Origin of the Species by Means of Natural Selection. New american Library, New York.

Dawkins R. (1976) The Selfish Gene. Oxford University Press.

Goldberg D. (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley.

Gribble S and Brewer E. (1997) System Design Issues for Internet Middleware Services: Deductions from a Large Client Trace. In Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS '97), December 1997.

Kauffman S, Macready W and Dickenson E. Divide and Coordinate: Coevolutionary Problem Solving. ftp://ftp.santafe.edu/pub/wgm/patch.ps

Lenski RE and Travisano M. (1994) Dynamics of Adaptation and Diversification. Proc. Nat Acad. Sci. 91: 6808-6814.

Marshall I, Roadknight C and Bilchev G. Performance implications of WWW traffic statistics. Submitted http://www.wtc2000.org

Paxson V and Floyd S. (1995) Wide Area Traffic: The Failure of Poisson Modelling. IEEE/ACM Transactions on Networking 3 (3) p. 226-244

Roadknight CM., Balls GR., Palmer-Brown D and Mills GE (1997). Modelling of complex environmental data. IEEE Transactions on Neural Networks. Vol 8, No 4. P. 852-862