Proceedings of the 5th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems, Madrid, Spain, February 15-17, 2006 (pp202-207)

Source Routing in the Internet with Reinforcement Learning and Genetic Algorithms

ZHIGUANG XU
Department of Math and Computer Science
Valdosta State University
1500 N. Patterson Street, Valdosta, Georgia, 31698
U.S.A.
http://www.valdosta.edu/~zxu

Abstract: Source routing of packets in the Internet requires that a path be selected in advance and stored at the source nodes. Path selection is typically based on Quality of Service (QoS) criteria like packet delay, delay jitter, and loss. A new protocol called the "Cognitive Packet Network" (CPN) [18, 19, 20, 21] has been proposed which is capable of dynamically choosing paths through a store and forward packet switching network like the Internet so as to provide best effort QoS to peer-to-peer connections. A CPN-enabled network uses smart packets to discover routes based on QoS requirements; acknowledgement (ACK) packets to deliver the routes back to source nodes; dumb packets to carry user-payload; and reinforcement learning to conduct path selection. We extended the path discovery process in CPN by introducing a genetic algorithm (GA) that can help discover new paths that may not have been discovered by smart packets [28]. In this paper, we further extend CPN with GA by prioritizing paths discovered based on their ages, adopting a progressive fitness evaluation system, and introducing a new genetic operator – mutation. The simulation topology has also been upgraded from a 10 by 10 grid to an arbitrarily connected network. We detail the design of the algorithms and their implementations, and finally report on resulting QoS measurements.

Key-Words: Routing, quality of service, packet switching, reinforcement learning, genetic algorithms

1 Introduction

Source routing protocols in a store and forward packet switching network like the Internet dynamically select paths for packets while they are being transmitted. The resulting paths are stored at source nodes for subsequent packets to use. Typically, routes are selected based on some pre-defined criteria of performance, for instance, "the shortest path", or "the smallest number of hops", or "the cheapest path"[14]. To fulfill the real-time requirements imposed by the multimedia applications, which are more and more popular nowadays, Quality-of-Service (QoS) type of considerations are taken into account when selecting a route, for example, "a quickest path", or "a path with the lowest packet loss rate", or both, etc. Furthermore, the route selection process has to be dynamic such that it can quickly adapt to the changes in the network [9,10,11,13]. Upon leaving the source, packets carrying the payload also carry a complete route that will lead it from the source to the destination. At each intermediate router, all what is left to do is to extract the next hop from the path and pass the packet down the road accordingly.

Cognitive Packet Network (CPN) has been proposed to provide best effort QoS to route user traffic. It uses *smart packets*, which are just a small fraction of the overall user traffic, to select routes based on the user's QoS "Goals". *Acknowledgement packets (ACK)* are the messengers who will carry the selected path along with QoS measurement data back to the source nodes. This path discovery and update process continues throughout the user's session, and the most recently found best path is used by the *dumb packets* that actually carry the user payload to the destination. The robustness of CPN to maintain traffic flow in the presence of node and link failures has been described in recent papers [20, 21, 22], while extensions to a wireless environment was first presented in [23]. It is the CPN-enabled routers who make the CPN intelligent. Those routers process both smart packets and ACK packets going through them by running reinforcement learning [16] based on the random neural network model (RNN) [3,4,5,8,24]. The weights of connections in the random neural network at each router are updated based on measurement data collected by smart packets and ACK packets, and the state of the network is computed in order to reward good routing decisions in the past and punish bad ones and make the best routing decisions for the subsequent packets.

An important improvement of CPN is to include a genetic algorithm (GA) that can help evolve new paths that may not have been explored by smart packets [27, 28]. Candidate routes found by smart packets and brought back by ACK packets are GA *individuals*. GA, which runs on the background at the source nodes only, selects *parents* from a pool of such individuals based on their fitness values with respect to user defined QoS requirements. Offspring individuals are generated as a result of GA crossover operation, and added back to the pool, from where the hosting node will pick the best route for the packets sent in the future.

In this paper, we further extend CPN with GA in the following ways. Since recent updates to a packet route tend to override and invalidate the old ones, GA individuals that represent packet routes are aged and thus prioritized. Consequently, new individuals have a better chance than old ones to be picked as parents for the subsequent reproduction. We then describe a progressive fitness evaluation system that evidently makes GA evolve better and quicker. In addition to crossover, a new GA operator – mutation is introduced. The randomness given by the mutation guarantees that GA is not trapped on a local optimum and converges quicker. We will conclude the paper by presenting the experiment results obtained on our simulator, which has also been upgraded from a simple 10-by-10 grid to an arbitrarily connected network. As witnessed by the results, both average packet delay and packet loss have been considerably improved thanks to the extensions.

2 Cognitive Packet Networks

The Cognitive Packet Network (CPN) architecture offers QoS-driven source routing to peer-to-peer user connections. Routers in the network that are CPN-enabled establish a virtual network on top of the existing IP network. CPN is designed to be QoS oriented, intelligent, and secure.

2.1 Packets in CPN

CPN contains three types of packets. S*mart packets* or *cognitive pac*kets route themselves and try to minimize the chances of being delayed or even destroyed by avoiding the congested areas. Smart packets learn from their own experiences about the network and from the experiences of other packets through *mailboxes* in the routers. They progressively refine their own model of the network (called Cognitive Map, or CM) and deposit QoS measurement data in the mailbox if needed as they travel through the network (See Fig. 1).

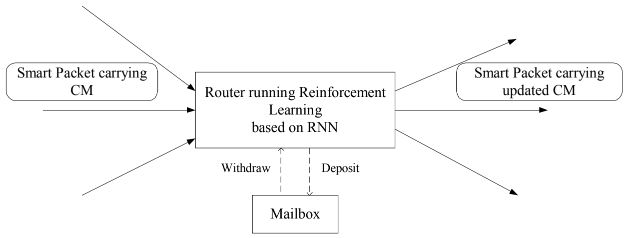

Fig. 1 Smart Packet and a CPN-enabled Router

Upon arriving the destination node, a smart packet will trigger to generate an ACK packet, which stores the reverse route and the measurement data collected by the smart packet. The ACK packet will travel along the reverse route back to the source, with loops on the path skipped, if any.

Dumb packets carry the user payload and use source routing. They use the best route so far discovered by smart packets and brought back by ACKs until another even better path is found. Such a selection is often made based on pre-defined QoS criteria like smallest packet delay and loss.

2.2 Reinforcement Learning based on RNN in CPN

In each router, a Random Neural Network (RNN) exists both for storing the weights and making decisions. When a smart packet arrives at a router searching for the best next hop, RNN extracts information like the QoS goal from the Cognitive Map (CM) of the smart packet and existing measurement data from the mailbox, updates the weights of the network with reinforcement learning algorithm, a trial-and-error type of unsupervised search, so that past decisions are reinforced or weakened depending on whether they have been observed to contribute to increasing or decreasing the accomplishment of the declared QoS goal. Suppose the packet delay and packet loss rate from a source node S to a destination node D are denoted as W and L respectively, the goal from S to D could be expressed as in the following equation.

$$G(S,D) = \frac{1-L}{W} \tag{1}$$

The adoption of the reinforcement learning in CPN was inspired by its uses in navigation in a maze [16].

The simulations, analysis, and implementations on test-beds have revealed that this simple approach appears very effective for dynamic routing of the smart packets [18,19,20,21]. New developments on CPN are presented in [25, 26].

3 Packet Routing with Genetic Algorithms

As we described so far, Reinforcement Learning algorithm based on Random Neural Networks (RNN) is running on all the CPN-enabled routers for smart packets to learn and react to the ever-changing and unpredictable conditions in the network. As a result, ACK packets keep bringing paths along with the corresponding QoS measurement data discovered by smart packets back to the source nodes. In order to optimize the best-path-selection decision made for subsequent dumb packets based on these candidate paths, Genetic Algorithm (GA) is adopted. GA runs on the background at the source nodes only and serves as an evolutionary mechanism that encodes the knowledge learned by smart packets and evolves optimized routes for dumb packets to use [28].

3.1 GA in General

A Genetic Algorithm (GA) is to simulate the natural evolution as a search and optimisation algorithm that operates over a population of encoded candidate solutions to solve a given problem [1,2,6,7,12]. The Key components that distinguish GA from other searching algorithms are:

- A *population of individuals* where each individual represents a potential solution to the problem. Individuals are typically binary strings of various lengths and each bit in the string is called a *gene*.
- A *fitness function* that evaluates the healthiness of each individual as a solution.
- A *Selection algorithm* that selects a pair of individuals for mating from the current population. Parent individuals are selected with a probability related to their fitness. The most popular selection algorithm is *Roulette Wheel Selection*.
- GA operators. As reproduction takes place, the *crossover* operator exchanges two individuals, whereas the *mutation* operator changes the gene value at some randomly picked location of an individual.

3.2 A GA Approach to Packet Routing

GA works as an enhancement of CPN packet routing. We will dedicate this section to the discussion of GA as a novice approach to packet routing. In the next section, we will see how GA is integrated into our CPN framework and works interactively with the other existing components in a CPN-enabled source node, which is typically located on the edge of the Internet.

3.2.1 GA Route Individuals

Every complete route discovered by smart packets and brought back by ACKs naturally becomes an individual (called *route individual*) in the GA population. Route individuals are of different lengths such as to allow GA more flexibility to the changes in the network. Each gene in the route individual represents a router along the route. For any nodes a and b, ab is a sub-sequence in a route individual w from S to D if and only if there is a link that connects a and b. Thus w always represents a viable path from S to D.

GA runs periodically on the background at source nodes. At the beginning of each run, the initial population of GA individuals is received from the traditional CPN part. By that, CPN keeps feeding GA with newly discovered routes along with QoS measurement data and the corresponding time stamps (for the purpose of aging routes). Note, in one population, route individuals for different destinations are all mixed (in section 3.3, we will discuss another pool of routes – *Dump Packet Route Depository*, where routes for different destinations are categorized into different sub-stacks). Also note that the QoS measurement data experienced by smart packets and brought back by ACK packets expressed in the format of goal function (Equation 1) allow us to collect at the source not only just the source to destination delay and loss, but also the delay and loss from any intermediate node the destination, and from source to any intermediate node, and also deduce the forward delay and loss from any node to any other node on the path. This is so significant because it allows possible the GA crossover and mutation operations that we will discuss in section 3.2.4 and justifies their correctness.

3.2.2 Prioritize GA Route Individuals by Their Ages

Suppose a route individual $R(S,D)$ which represents a route from S to D has the measured goal value $G(S,D)$ and time stamp Tg. Tg is defined as the time at which the ACK packet brought $R(S,D)$ back to S. Also say the current time is Tc. The following aging function $A(S,D)$ is used to prioritize $R(S,D)$:

$$A(S,D) = \frac{1}{T_c - T_g + 1} G(S,D) \qquad (2)$$

This is done to appreciate the fact that the fresher the information, the more valuable it is to our purpose of packet routing. When selecting parents for reproduction, we want to make up the decision based as much as possible on the most recent measurements.

3.2.3 Progressive Fitness Evaluation

Then, suppose the *A* value (expression 2 above) of the route currently being used by dumb packets from *S* to *D* is *C(S,D)*, the following progressive fitness evaluation function *f(S,D)* is used to evaluate the fitness of *R(S,D)*:

if $sigmoid(A(S,D) - C(S,D)) > 0.5$

$$f(S,D) = \alpha \times sigmoid(A(S,D) - C(S,D)) \qquad (3)$$

else

$$f(S,D) = \beta \times sigmoid(A(S,D) - C(S,D)) \qquad (4)$$

where α, β are largely separated constants (like α=10, β=0.1) and the sigmoid function is defined as following:

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \qquad (5)$$

One one hand, if *A(S,D)*, which essentially is the measured goal value of a route *R(S,D)* from *S* to *D* prioritized based on its age, is greater than the *A* value of the route that is actively being used now, the larger the difference between the two figures, the larger the fitness value we assign to *R(S,D)*, and the larger the chance it will be picked during the parent choosing process; on the other hand, if the route under study *R(S,D)* does not perform better than the one currently in use, we assign *R(S,D)* some small fitness value, and the larger the difference, the smaller the fitness *R(S,D)* gets, and the smaller the chance it will be picked as parent.

3.2.4 GA Operators

Once two parents w_1 and w_2 are selected with probability related to their fitness values (typically done through *Roulette Wheel Selection* Algorithm), the following GA operators are applied.

- Crossover with probability *Pc* (typical valued as 0.7~0.8). Suppose w_1 and w_2 share some intermediate node α. We have $w_1 = u_1 \alpha v_1$ and $w_2 = u_2 \alpha v_2$. The crossover will then generate offspring individuals $w_3 = u_1 \alpha v_2$ and $w_4 = u_2 \alpha v_1$. Since the goal values are additive, we have $G(w_3) = G(u_1 \alpha) + G(v_2)$ and $G(w_4) = G(u_2 \alpha) + G(v_1)$.
- Mutation with probability *Pm* (typical valued as 0.01~0.02). This is conducted on w_3 and w_4 in sequence. Let's take w_3 as an example. An intermediate node β (except the source and the destination) on w_3 is randomly picked, so w_3 can be expressed as $w_3 = u\theta\beta v$, where θ is the predecessor of β on w_3. Then we randomly choose one of θ's successors except β, say γ. Suppose one of the known downstream routes starting at γ is γ*x* and it is the one with the smallest observed goal value among all the known downstream routes, we mutate the original w_3 in the following way $w_3 = u\theta\gamma x$. Note, *x* could be empty.

The evolution process described above continues until some predefined termination condition is satisfied, for example, the maximum number of iterations has been exceeded. During the whole process, GA attempts to maintain the balance between the exploration for generating new routes and exploitation of discovered information. As a result, GA is expected to find some better routes that have not been found by CPN.

3.3 Integrate GA into CPN

The performance of GA described above depends largely upon the following two factors: the extent to which GA route individuals are able to interact with each other to produce effective offspring. This is mainly achieved through GA operators. Another one is the level of the population diversity, i.e., the number of different route individuals.

Although GA crossover and mutation are expected to result in generating new individuals and consequently diversify the GA population, the input from CPN is considered as a major and valuable source of diversity for the CPN population as well as an indication of the state of CPN (which could provide indirect pressure as to the direction in which GA should evolve next) [28].

Fig.2 summarizes how GA is integrated into and works together with the traditional CPN framework.

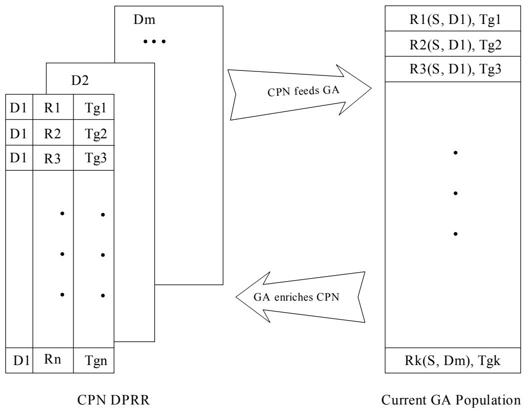

Fig. 2 GA interacts with CPN

On one hand, routes discovered by smart packets and brought back by ACK packets are stored in *Dump Packet Route Repository,* or *DPRR* at the source node, say *S.* DPRR is organized such that routes leading to the same destination are group together. In each group, routes are sorted based on their QoS performance, i.e., the observed goal values. Each route is also associated with a time stamp at which that route was added to the table. On the other hand, GA population is a mixture of discovered routes leading to different destinations.
CPN initiates GA population and keeps feeding it with newly found routes. Once started, GA works on the background to evolve new routes. It feeds back CPN with its best offspring routes and consequently enriches DPRR such that the hosting source node could have better and more alternatives to choose from when dump packets carrying user payload are ready to send.

4 Simulation Experiments

A network simulation program is used to demonstrate the effectiveness of the extensions we made to our previous CPN with GA. The simulator itself was upgraded from a 10-by-10 grid to a network with nodes and connections arbitrarily definable and modifiable. In order to make it even closer to the reality, areas in the network could be dynamically configured as either high congested areas, or areas with high packet loss rate, or both. Smart packets are observed being able to quickly react to the changes in the network, skip the dangerous areas, and manage to find best alternatives to their destinations.
We compare the performance of CPN with extended GA, CPN with GA, and traditional CPN on a network with 50 nodes that are arbitrarily and redundantly connected. Three areas with 4 nodes each are designated as high-traffic zones (packets traveling through these areas will experience big delays and even losses). Fig. 3 refers to average end-to-end delay versus normalized traffic rates. Fig. 4 plots the packet loss rate versus traffic rates.
Our results clearly reveal marked improvement in performance with respect to both delay and loss thanks to our extensions to CPN with GA.

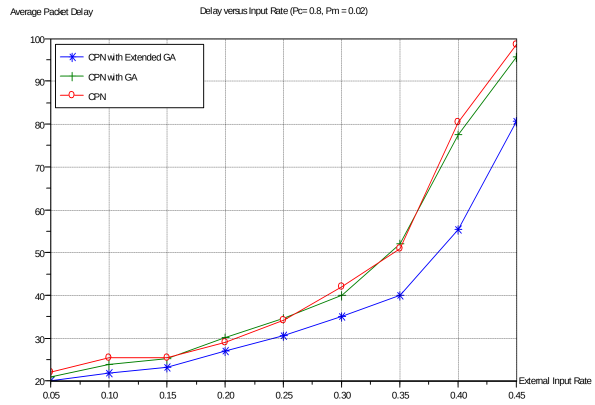

Fig. 3 Simulation Result: Average Packet Delay

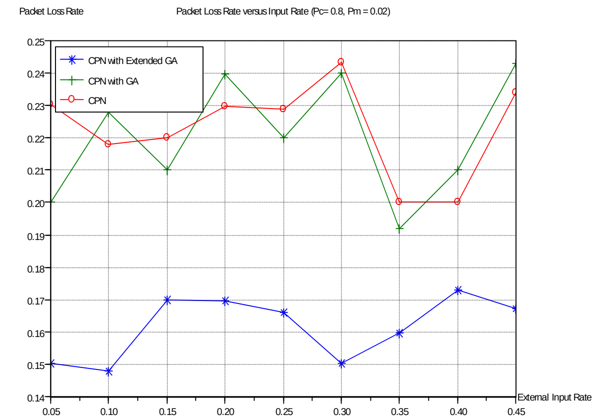

Fig. 4 Simulation Result: Average Packet Loss Rate

5 Conclusions

Extensions we made to the Cognitive Packet Network with Genetic Algorithms (CPN with GA) include: Prioritizing routes based on their ages such that when GA selects routes as parents for reproduction, newly discovered routes have better chances. Progressive fitness evaluation is adopted to deliver faster GA convergence. Also, mutation is introduced as a GA operator to generate offspring route individuals. As we showed in the QoS measurement results obtained from the simulations, thanks to theses extensions, we observed dramatic improvements with respect to both delay and loss, the two most important QoS expectations.

Future work include some refinement and optimizations of the Genetic Algorithms in CPN such as to minimize the impact in terms of computational speed at the source nodes caused by the execution of the GA process on the background.

References:

[1] J.H Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[2] R.S. Sutton, Learning to predict the methods of temporal difference, *Mach. Learning*, Vol. 3, 1988, pp. 9-44.

[3] E. Gelenbe, Random neural networks with positive and negative signals and product form solution, *Neural Computation*, Vol. 1, No. 4, 1989, pp. 502-510.

[4] E. Gelenbe, Réseaux neuronaux aléatoires stables, *Comptes-Rendus Acad. Sci., t.310, Série II*, 1990, pp. 177-180.

[5] E. Gelenbe, Stable random neural networks, *Neural Computation*, Vol. 2, No. 2, 1990, pp. 239-247.

[6] D.S. Burke, K.A. De Jong, J.J. Grefenstette, C.L. Ramsey, and A.S. Wu, Putting more genetics in genetic algorithms, *Evolutionary Computation*, Vol. 6, No. 1, 1998, pp. 387-410.

[7] G. Syswerda, Schedule optimisation using genetic algorithms, *Handbook of Genetic Algorithms*, 1991, pp. 332-349.

[8] E. Gelenbe, Learning in the recurrent random neural network, *Neural Comp.*, Vol. 5, No. 1, 1993, pp. 154-164.

[9] E. Gelenbe, X. Mang, R. Önvural, Diffusion based call admission control in ATM, *Performance Evaluation*, Vol. 27 & 28, 1996, pp. 411-436.

[10] E. Gelenbe, X. Mang, and Y. Feng, Diffusion cell loss estimates for ATM with multiclass bursty traffic*, Computer Systems – Science and Engineering, Special Issue: ATM Networks*, Vol. 11, No. 6, 1996, pp. 325-334.

[11] V. Srinivasan, A. Ghanwani, E. Gelenbe, Block cell loss reduction in ATM systems, *Computer Communications*, Vol. 19, 1996, pp. 1077-1091.

[12] E. Gelenbe, Genetic algorithms with analytical solution, *Robotics and Autonomous Systems*, Vol. 22, 1997, pp. 59-64.

[13] S. Chen and K. Nahrstedt, An overview of Quality-of Service routing for the next generation high-speed networks: Problems and Solutions, *Network Magazine*, Nov/Dec, 1998.

[14] C. Huitema, *Routing in the Internet, 2nd Ed.* Prentice Hall, 1999.

[15] E. Gelenbe, G. Pujolle, *Introduction to Queuing Networks*, John Wiley & Sons, 2000.

[16] U. Halici, Reinforcement learning with internal expectation for the random neural network, *Eur. J. Opns. Res.*, Vol. 126, No. 2, 2000, pp. 288-307.

[17] E. Gelenbe, E. Şeref, Z. Xu, Simulation with learning agents, Proc. IEEEE, Vol. 89, No. 2, 2001, pp. 148-157.

[18] E. Gelenbe, R. Lent, Z. Xu, Towards networks with cognitive packets, *Performance and QoS of next generation networking*, 2001, pp. 3-17.

[19] E. Gelenbe, R. Lent, Z. Xu, Towards networks with cognitive packets, Key-Note Paper, *Proc. IEEE MASCOTS Conference*, ISBN 0-7695-0728-X, San Francisco, CA, Aug/Sept, 2000, pp. 3-12.

[20] E. Gelenbe, R. Lent, Z. Xu, Design and performance of cognitive packet networks, *Performance Evaluation*, Vol. 46, 2001, pp. 155-176.

[21] E. Gelenbe, R. Lent, Z. Xu, Measurement and Performance of Cognitive Packet Networks, *J. Comp. Nets.*, Vol. 37, 2001, pp. 691-701.

[22] E. Gelenbe, R. Lent, Z. Xu, Networking with Cognitive Packets, *Proc. ICANN*, Madrid, Aug 2002.

[23] E. Gelenbe, R. Lent, Z. Xu, Mobile Ad-hoc Cognitive Packet Networks, *Proc. IEEE ASWN*, Paris, July 2002.

[24] E. Gelenbe, Cognitive Packet Networks: QoS and Performance, Key-Note Paper, *IEEE MASCOTS Conference*, San Antonio, TX, Oct 2002.

[25] E. Gelenbe, M. Gellman, P.Su, Self-awareness and adaptivity for Quality-of-Service, *Proc. 8th IEEE International Symposium on Computers and Communications*, Antalya, July 2003, pp. 3-19.

[26] E. Gelenbe, A. Nunez, R. Lent, Smart WWW traffic balancing, IEEE Workshop on IP Operations and Management, Kansas City, Missouri, 1-3 Oct. 2003.

[27] E. Gelenbe, P. Liu, J. Lainé, Genetic algorithms for route discovery, *SPECTS' 03, Summer Simulation Multiconference, Society for Computer Simulation*, Montreal, July, 2003.

[28] Z. Xu, A.S. Wu, Adhoc-like routing in wired networks with genetic algorithms, *Ad hoc networks 2*, Vol.2, No.3, July 2004, pp. 255-263.