In *IWANNT'93 International Workshop on Applications of Neural Networks to Telecommuniations*, pp. 183-187, 1993.

A Learning Model for Adaptive Network Routing

Tony R. Martinez and George L. Rudolph

Department of Computer Science
Brigham Young University, Provo, Utah 84602
martinez@cs.byu.edu
george@axon.cs.byu.edu

Abstract

Increasing size, complexity, and dynamics of networks require adaptive routing mechanisms. This paper proposes initial concepts towards a learning and generalization mechanism to support adaptive real-time routing. An ASOCS learning model is employed as the basic adaptive router. Generalization of routing is based not only on source/destination address, but also on such factors as packet size, priority, privacy, network congestion, etc. Mechanisms involving continual adaptation based on feedback are presented. Extensions to conventional addressing which can support learning and generalization are proposed.

1. Introduction

As networks increase in size and routing complexity, new types of mechanisms are required to support routing decisions. In particular, network routers must accommodate and adapt to rapid changes in network topology and usage.

This paper proposes a learning and generalization scheme to support adaptive real-time routing in computer networks. Traditional routing algorithms are based on tabular storage of destination/gateway tuples which decide where a packet should be forwarded upon arrival at a gateway. This method suffices when this tabular information is available and up to date. When not available,mechanisms such as default gateways, random forwarding, or message discarding are employed. We propose that generalization schemes may be used to forward packets in a more efficient and accurate manner when confronted with insufficient and inaccurate routing information.

2. Approach

Most current algorithms base routing decisions primarily on the destination address. Though destination address is obviously crucial, other information may be important in making a proper routing decision. For a packet, this information includes packet size, priority, preferences (latency, throughput, reliability, etc.), privacy classification, etc. For the network such information as congestion, current failures, load, etc., may be pertinent. We call these *packet features* and *network features* respectively. We refer to address, packet features and network features collectively as *routing features* in this paper. Ad hoc algorithms require unique handling of these different features. A more general learning scheme treats all of these uniformly, with weighting of importance taking place during learning, thus both simplifying and expanding the approach to routing.

McDonald [3] proposes an Adaptive Self-Organizing Concurrent System (ASOCS) model as an adaptive network router. ASOCS [4,5,6] are a class of connectionist learning models which seek computational efficiency as highly parallel and adaptive connectionist logic systems that map discrete input values to output values. An ASOCS can infer routing decisions from routing rules that have been expressed as discrete input variables. This paper seeks to extend this approach through more robust generalization techniques. Below, the adaptive mechanism used for learning routing information and doing actual forwarding is simply referred to as the *router*.

The approach proposed in [3] assumes a version of the learning bridge algorithm. A router observes the source address and routing features of each incoming packet and remembers on which interface it arrived. When the address and other features are later observed as a destination, then the router forwards the packet on the interface on which the address was earlier observed as a source address.

As new nodes are added to a network (or even with a network initialized with no routing information), these nodes can send out a *beacon* packet which is forwarded along all routing lines except that on which the beacon was received. This process "primes" the network with initial routing information which is learned as the network evolves. By adding distance information (number of hops, time lapse, etc.) to the beacon packet, an accurate assessment of efficient routing is captured. However, accuracy of information may still be faulty due to the following.

- The beacon process above must be limited because of excessive bandwidth requirements.
- Conditions at the time of the initial beacon were not representative of average case.
- A constantly changing network causes initial information to be inaccurate.
- The initial beacon is based only on source address and not on other routing features.

These and other factors cause the necessity of ongoing learning and adaptation to the network environment. The model above provides a reasonable initial state from which continual adaptation and refinement can occur. This adaptation requires some source of feedback to the router to allow update of its current state. Some potential approaches include:

- Feedback Protocol: This protocol would cause (optionally) the router which received the packet to send information back to the source address regarding the "cost" of receiving the packet. This cost is then used to update and improve the initial sending router. Note that cost in this context may involve not only time to arrival, but metrics relating to errors, privacy, reliability, etc.
- Piggy-Back Metrics: A feedback protocol will add to network traffic. Another option is to maintain the current cost metric with the packet. Upon arrival at a router, if this metric is excessive are not in line with a *predicted* cost stored at the receiving router, only then need this information be sent back to the source routers.
- Inquiry Packets: Discovery of improved routing can be initiated by a router through the sending of *inquiry* packets which are a more selective version of beacon packets. In particular, a router with poor predictive cost may seek improved routes to commonly occurring destinations. These can be initiated at times of lower network load so as to seek constant improvement without an adverse effect on network traffic.

3. Learning and Generalization

There are a number of learning and generalization models which could be used to support generalization of network routing. In the case where a gateway has seen an incoming source address and later needs to send to that address as a destination, the mechanisms of backward learning overviewed above could suffice. However, there are times when a source needs to send to an address from which it has not yet received a message and thus no information regarding that address will be found in the gateway's current routing tables. This lack could be due to newness of the destination site, error at the routing table, etc.

In this case the address given by the initial sender could not just be the standard symbolic name (i.e. jdoe@cs.univ.edu), since the name server is not yet aware of this address. The numeric name could be given and although the gateway has no record of this yet, generalization could be done to make a best guess at routing, rather than the standard return of host unknown. However, since numeric addresses are typically given randomly to new hosts, no interesting generalization can be made from the address number itself. For example, a learning model could note that addresses 123, 129 and 140 are best delivered to output line 1 and 124, 160, and 162 to output line 2. If a new address, 144, is the destination and it has not been previously seen, the learning model could only make an arbitrary guess as to the best output line. This is because the input feature is a random "address" which has no regularity or critical sub-features which can be learned to discriminate or join itself with/from other addresses. As mentioned above, other routing features can be used to aid in generalization, but without at least some candidate output lines based on address the other information is moot.

For any learning system to actually be able to generalize on inputs not seen before, the input must include features which have within them some kind of regularity which the learning model can extract. We propose that to support generalization on unknown addresses the address itself must contain some extractable regularities. For example, a uniquely identifying address (which could be a random set of bits) could be augmented to also contain geographical information, such as the approximate latitude and longitude of the address site. (Similar extensions having different motivation and advantages are proposed elsewhere [1,2]). In this case, there is extractable regularity to be learned from addresses both incoming and already available in the routing table. When a new address is the destination, one possible learning bias is that the packet should be sent on an output line which has been successful for addresses with a similar geographic location as the new one. At this point, deciding between candidates can be enhanced using other routing features which are a part of the packet itself.

Other regular features could be part of the basic address. One which is already part of current internet addresses is the organization type (i.e. edu, com, gov). This type of information can be used to aid in learning what types of lines (security, speed, size, etc.) are typically best suited to communication involving these classes of users.

While too much addition of regularity would lead to cumbersome addresses, a judicious extension to the basic address mechanism could allow generalization on routing to unknown locations with reasonable cost and accuracy.

4. Minimization

While not supporting full generalization as in the previous section, the learning bridge algorithm can lead to more concise routing information for addresses actually received. Hierarchical IP addresses (i.e. address prefixes, geographical information, or an equivalent labeling system) could provide hierarchical abstraction through the process of minimization. A pair of prefixes of length N can be minimized to a single prefix of length N-1 if the prefixes differ in only one bit (e.g. the prefixes 1010 and 1011 can be summarized to the prefix 101*, where * is a don't care variable). This can be repeatedly applied so that a set of longer prefixes can be combined into a single, much shorter prefix. (Note that this same philosophy can be used with the geographical extension of section 2 to allow the representation of a large number of routing tuples into a single rule based on geography or other feature.) This includes prefixes for both route "table" entries (internal minimization), and route update information sent across the network (external minimization). Minimization thus helps reduce growth and size of routing tables, as well as route information traffic in the network. If eight sites are attached to the same network regional router, and their prefixes can be minimized, then the regional router can store, and advertise, its reachability to all eight sites with a single prefix. As a result, the number of routing entries and routing information traffic will scale less than linearly with respect to the number of networks.

5. Conclusion and Current Work

This paper presents initial approaches which can support routing generalization for novel network addresses. While these ideas are still in an early stage, current work seeks to demonstrate potential through network routing simulation and generalization studies

References

[1] Ford, Peter S., Yakov Rekhter, and Hans-Werner Braun, Improving the Routing and Addressing of IP, *IEEE Network*, **Vol. 7**, #3, pp. 10-15, May 1993.

[2] Katz, Dave, and Peter S. Ford, TUBA: Replacing IP with CLNP, *IEEE Network*, **Vol. 7**, #3, pp. 38-47, May 1993.

[3] Mcdonald, K., T. R. Martinez, and D. M. Campbell, A Connectionist Method for Adaptive Real-Time Network Routing, *Proceedings of the 4th International Symposium on Artificial Intelligence*, pp. 371-377, 1991.

[4] Martinez, T. R. and Vidal, J.J., Adaptive Parallel Logic Networks, *Journal of Parallel and Distributed Computing*, **vol. 5**, 26-58, 1988.

[5] Martinez, T. R. and Campbell, D. M., A Self-Adjusting Dynamic Logic Module, *Journal of Parallel and Distributed Computing*, **vol. 11**, no. 4, pp. 303-313, 1991.

[6] Martinez, T. R. and Campbell, D. M., A Self-Organizing Binary Decision Tree for Incrementally Defined Rule Based Systems, *IEEE Transactions on System, Man, and Cybernetics,* **vol. 21**, 1991.