A Heuristic Routing Mechanism Using a New Addressing Scheme

M. Ravanbakhsh[1], Y. Abbasi-Yadkori[1], M. Abbaspour[1,3], H. Sarbazi-Azad[1,2]

[1]IPM, School of Computer Science, Tehran, Iran
[2]Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
[3]Department of Electrical and Computer Engineering, Shahid-Beheshti University, Tehran, Iran
y_abbasi@ce.sharif.edu, ravanbakhsh@ce.sharif.edu, maghsoud@ipm.ir, azad@{ipm.ir, sharif.edu}

Abstract-**Current methods of routing are based on network information in the form of routing tables, in which routing protocols determine how to update the tables according to the network changes. Despite the variability of data in routing tables, node addresses are constant. In this paper, we first introduce the new concept of variable addresses, which results in a novel framework to cope with routing problems using heuristic solutions. Then we propose a heuristic routing mechanism based on the application of genes for determination of network addresses in a variable address network and describe how this method flexibly solves different problems and induces new ideas in providing integral solutions for variety of problems. The case of ad-hoc networks is where simulation results are more supportive and original solutions have been proposed for issues like mobility.**

Keywords: Addressing Scheme, Heuristic Routing, Mobile Ad-hoc Networks, Pricing.

I. INTRODUCTION

Addressing and routing are two of the most important elements in modern network infrastructures. Recently, new problems like congestion in spite of disengaged paths [1], lack of a fair and perfect pricing mechanism [5], multicast and grouping overheads [7] and newborn problems of ad-hoc networks, have challenged the networking industry. There is a wide spread belief that because of an inherent unity between these problems in nature, providing particular solutions to each of them might prevent the others from becoming practical. In other words, there is a feeling of necessity for a unified solution for these problems, a solution that may act more flexible in confronting new concerns and introduction of new concepts. Previously, however, there have been some works, though considering simplified conditions, to achieve a general solution to the problem of routing efficiency and pricing using an approach inspired by the game theory [13].

In this paper, we first introduce an addressing scheme using variable addresses. Then, we propose an integral solution to some of prevalent problems including some in the area of ad-hoc networks like mobility and power-management, using a heuristic method named genetic routing based on the concept of variable addresses. Although the proposed solutions are not fully descriptive, they help clarifying the idea. As the topic of this article implies, rather than trying to describe a detailed algorithm, our attempt is to explain a mechanism of dealing with such issues, so there might be various implementations with different effectiveness for a particular usage.

The rest of the paper is organized as follows. Section 2 introduces the proposed addressing scheme. In section 3, a heuristic routing based on genetic algorithms is introduced. Section 4 discusses the routing in mobile ad-hoc networks. Some simulation results are reported in section 5, followed by section 6 which concludes the paper.

II. A NETWORK OF NODES WITH VARIABLE ADDRESSES

Consider a network in which every node has a long enough variable address to *reflect* the topology and the traffic of the time for that node. The exact meaning of the term *reflect* becomes clearer in the rest of the text. However, it is not wrong to think of node address as a replacement for keeping and exchanging the routing table's data. Then, routing is defined as follows: every node upon receiving a packet with a particular destination, applies a comparison function, with every neighboring node address and the destination address being its arguments, to determine where to forward the packet. For example if the comparison function is *similarity*, the routing method forwards the packet to a neighboring node which is the most similar to the destination address.

Let $G(V,E)$ be the network graph, where $V = \{v_i\}$ is the set of network nodes, and $E = \{e(v_i, v_j) | v_i, v_j \in V\}$ represents the set of instantaneous links costs. The link cost between two nodes u and v is assumed to be ∞ when no direct connection exists between them. Please note that we consider G as a snapshot of the network topology, in which the links costs represent the network traffic in that particular moment. Our aim is to assign an address $a(v_i)$ to each node v_i at the moment under consideration, where

$$a : V \to F, F = \{f : \{1,...,address\_length\} \to \{0,...,(Base-1)\}\},$$

Here, F denotes a set of all possible functions as defined above. Actually an address $a(v_i)$ denotes a sequence of specified length *address_length* that consists of integer numbers in the given base *Base*. The employed set notation leads to the simplicity of future definitions.

Both of the parameters *Base* and *address_length* are network parameters. A network parameter is a parameter that should be defined according to the network conditions including size, topology and routing algorithm in use (as previously mentioned, we can have different implementations).

When forwarding a packet, we use the evaluation function *eval* to determine the priority of the neighboring nodes, where $eval : F \times F \rightarrow \Re$, $eval(f_i, f_j) = \overline{\overline{(F_i \cap F_j)}}$, where F_i and F_j are the set representations of f_i and f_j (as a binary relation), and two parallel lines above a set means cardinality of that set. Please note that one could implement the function *eval* in any other way. We have used the intersection operation, because actually it is the one we use in the simplest implementation of genetic routing.

We also define a complete graph $C(V, E')$ as the *Distance-Graph*, in which *V* is the same as in *G*, and

$$E' = \left\{ e'(v_i, v_j) = eval\left(a(v_i), a(v_j)\right) \right\}.$$

Now, let us have a look at the problem from the opposite side. Having the value of the evaluation function for every two nodes, we may try to find the proper address for each node. These given values of the evaluation function are determined by a routing protocol. In fact, a routing protocol shall be defined as a method of mapping $G \rightarrow C$ in a distributed manner.

Now, the question is, having the graph *C*, how we can find a consistent address assignment function '*a*' if one exists. And what characteristics make an assignment feasible for a distance-graph.

This question defines a *Constraint Satisfaction Problem*. Although the problem seems to be NP-complete, it has a solution with a run time complexity of $O(n)$, where *n* is the number of nodes in the network. It is because the constraint network is strongly n-consistent if some weak conditions held. Thus, there is a solution without back-tracking for any variable ordering. Using the proposed notation, contemporary routing protocols correspond to some relatively simple and straightforward assignments of the evaluation function. Here, by *simple* we mean there is the possibility of more complex assignments by which we may have a better consideration of the traffic and topology. Also, introducing a distributed implementation of the assignment algorithm still remains a challenging problem. For example, the *OSPF* routing protocol, that uses time as the metric, can be simply defined as $e(v_i, v_j)$, i.e. the length of a shortest path between v_i and v_j.

Since similarity is a symmetric function, it seems that the proposed addressing scheme is not able to cause proper traffic effects in addresses when dealing with asymmetric traffic patterns. However, the evaluation function can be an asymmetric function in order to reflect asymmetric traffics.

The most crucial problem with variable address mapping is the issue of variability, which result in an important question: "How can one node address another node with a variable address?" In contrary to its difficult appearance, it's easy to solve because:

1) Address invariability is only necessary for end-nodes not for the intermediate ones (of course this is not true in the case of ad-hoc networks).

2) The address variation is only the result of traffic change. On the other hand, the address of a node is a representation of the topology as well, which is supposed to be constant (for MANETS this is not the case and the solution is discussed in section IV).

3) Tree-like structure of topology is common at the borders of networks. On the other hand, in a tree-like topology, the path remains the same, whatever the traffic is. Thus, the border nodes, including the end-nodes, can easily just reflect the topology.

4) In the worst case, the end-nodes can have permanent addresses that only reflect the topology, so there would be more burdens on the middle way routers to bring about flexibility.

III. THE PROPOSED ROUTING ALGORITHM

The *genetic routing* algorithm is based on variable address assignment, which exploits the concept of 'gene' in a similar way to its expected meaning. Although the algorithm, in its essence, is not like other genetic algorithms, for the sake of simplicity we call it genetic routing. The following example demonstrates how a heuristic method can be merged with the concept of variable addresses, and how it can contribute to the emergence of a new routing algorithm.

As we defined previously, in a variable address network, every node has a fixed-length variable address, which is the output of the function '*a*' having that node as the argument. Genes are responsible for making similarities in nodes by copying and pasting address segments. Genes are small packets (at most tens of bytes) that network nodes can distinguish them from data packets and treat them differently. Consider the graph *G(V,E)* as the network graph in which address mapping '*a*' is a *random* assignment of *f* to *v*, where $f \in F$ and $v \in V$.

If a gene *g* is generated in the node v_i, then it is a subset of F_i. After transmission of *g* to v_j, the node v_j applies that gene to itself, and then we have $g \subseteq F_j$. We say the gene has *affected* the node or has made the similarity. Suppose every node has a specific amount of genes. These genes are distributed among neighbors. Each neighbor after applying gene to itself (by applying, we mean affecting the node address by the gene), would carry out one of the following actions in a probabilistic manner:

It would alter the gene according to its own address and then send it back to the sender, or

(with the probability of distribution-percentile as a network parameter) send it unchanged to one of its other neighbors.

Reflecting the traffic: Node addresses during the convergence time somehow indicate the traffic at that time. During the last moments of convergence, packets can partially find their route, since there is a convergence of node addresses although not indicating the final path. On the other hand, consider reciprocation of genes on that link, in a particular duration. As said before, a node may alter a gene and send it back to its previous sender; this would cause the reciprocation

of some genes between neighbors. In a congested link, there would be less gene exchange in the same interval because genes should spend much of their time in the queue, while on a disengaged link, more similarity might be made between two neighbors as a result of gene exchange. But since there is no consideration for traffic asymmetry, this change will not be as effective as expected. Consider a link which is congested just in one way; the resulting similarity of the addresses would not affect the addresses in an asymmetric manner to show which way is congested and which is disengaged, but it works well when the traffic is nearly symmetric.

Elimination of genes: Since we need to eliminate genes from the network when enough convergence is obtained, we accompany every gene with a mask array where each cell corresponds to a number in the gene's address. More precisely: *mask(g) : g → {active, passive}*, determines whether the corresponding number in the gene is active or passive. When passive, a number can not participate in the copy and paste process. In each paste action, one mask bit is randomly selected and it becomes false if it is not false already. When all bits of a mask are false, a gene is no more effective and a node can neglect it.

A. Advantages of the proposed routing

Here, we describe some merits of the proposed genetic routing algorithm.

1) Its simplicity as because It is obvious that the routing process is solely based on node addresses without any peripheral information of the network. However forwarding process is simply a group bit comparison operation and if *Base* is chosen to be a power of two, a simple hardware implementation is possible. Also just like the forwarding process, interpretation and update of genes, is quite simple and makes fast hardware implementations possible After convergence, a loop free routing and correct delivery is guaranteed.

2) In contrary to the other protocols (see [8]), the drop of protocol messages, not only wont cause any problem but it is a merit since it reflects congestion..

3) Each end-node has a good estimate of its distance to the other end-node.

4) *Pricing*: Since the difference between the current and the destination node addresses of a packet is a good estimate of the network resources it uses, there is a possibility of exploiting a packet based charging mechanism (see [5]) which uses a reverse path charging.

5) Applying different kinds of *policies* is possible for each node in two forms:

   a) As an indicator of activity in the routing process, a node, by the amount of genes it produces, can decide to what extent it should influence other nodes, and by the amount of receiving genes it applies to itself, can decide to what extent it should be affected.

   b) An intermediate node can increase or decrease the amount of the traffic it receives from its neighbor x that should be transmitted to the neighbor y, by increasing or decreasing its own similarity to their common similarity.

6) There are variations of this routing mechanism that use asymmetric evaluation functions instead of intersection. They search for particular patterns in each node address and use that pattern to determine which node they should forward the packet to.

7) In case of link or node failure, which in turn is similar to some concurrent link failure, the corresponding packets can still be delivered to an available node which is the most similar to the destination; current prevalent protocols lack such capability [9].

8) The concept of genes provides the possibility of influencing local decisions by global conditions, which is a necessity for a forth generation routing protocol [3, 10].

IV. 4. GENETIC ROUTING IN MOBILE AD-HOC NETWORKS

The genetic routing method works well particularly for ad-hoc networks, in which we have locality in connections, in contrast to wired ones in which there is no necessity for locality of a connection. By locality in connections we mean some kind of geographical factors in deciding the connectivity of two nodes which finally causes a semi-planar graph [11].

Flexibility of the concept of genes in inclusion of new functionalities is evident in confronting the issues discussed below.

A. Mobility

Handling mobility in MANETs is a great challenge in designing new routing protocols. When a mobile node changes its place how it should be assigned a new address and how the other nodes can route to it, just having its old address? These are the main questions.

The mechanism is simple: a *periodic global converge*. The period of converge is determined by the amount of mobility in the network. A highly mobile network in which mobile nodes exit one region and join another one rapidly needs shorter periods while a more static network in which mobile nodes have lower speeds would be satisfied with longer convergence periods. But how a particular node is supposed to find another one when after a periodic global converge, nodes would have new addresses. The answer is: nodes will exploit their old addresses until the new convergence become complete, and then they try to switch to their new address while still keeping their own and their neighbor's old addresses. In this manner, every node would have a list of say *K* previous periodic global convergence addresses of its own and its neighbors. The case is trivial while nodes are in communication during the address switching since they would inform each other of their new address before switching. Now consider a node which wants to route to another one, but the latest address it has from that node is the address which belongs to *K'* period of global

convergence before. This node still can route to the destination in order to achieve its new address if K' < K, and then continue the communication using the new address.

B. Power management

Managing the overall network power to keep the network graph connected as long as possible, so using low power agents less than the high power ones in the process of routing , using lower cost paths with the power as metric [2], and determining the efficient power range of a node are important issues in MANETs. We won't solve them all, though the first and the second one are in opposition. As was mentioned in applying policies, one node can easily decide its involvement in routing, even between two particular neighbors which here may be in long geographical distance. So it's up to every node to manage its own power consumption regarding the other nodes in its neighborhood, since genes may act as an indication of a node's power too.

C. Address assignment

In MANETs, dynamic and fast address assignment, with low overhead is an important issue. In genetic routing, setting an address for a newly joined node is easy; two suggestions are:

o Wait until the next periodic global converge; then as a result of convergence, newly joined node would get a new address.

o If the node doesn't want to wait until the next convergence, it can get neighbors address and assign a mixture of their address manually to itself. This address although not precisely assigned, would work correctly, since the assumption is that the packets would route correctly to neighbors.

Both of these functionalities are easy to achieve using the method of periodic convergence. We don't describe them as the idea is obvious.

D. Grouping

Grouping and routing according to the common patterns is possible as mentioned before. In addition to providing stateless optimum multicast and anycast, this schema can provide facilities for exploiting dominating sets [4] in order to have lower power consumption noticing the advantages of genes in algorithms for finding dominating sets. For an example, a node can easily determine if its two neighbors are directly connected or not, calculating their similarity without any message exchange. Genes also provide facilities for selecting Virtual Backbones [12], since a backbone link should have a reasonable bandwidth and also should have stayed static long enough, for both of which we have genetic metrics in genetic routing.

V. SIMULATIONS RESULTS

We simulated a simple implementation of proposed genetic routing for both ad-hoc and wired networks of 10, 20 and 50 nodes for different topologies for each network size. Results for varieties of network parameters are shown in table 1.

In this table, c.t is the average convergence time, with all network links having 10 KBytes of bandwidth. Considering approximately linear relation of c.t with the inverse of bandwidth, because of negligible processing time required for genes, c.t can be improved so much in networks with high bandwidth. Efficiency (eff) is measured in comparison with SPF average network delay. n.g is the average amount of genes every node has. n.w is the average network width (maximum over every pair of minimum distance between two nodes, measured in number of hops). g.l is the gene length. a.l is address length in number of digits in Base b and finally n.s is the network size in number of nodes. As you may have noticed, there is a tradeoff between various desired characteristics.

TABLE I.   CONVERGE TIME AND EFFICIENCY FOR WIRED NETWORK.

n.s	c.t (ms)	n.w	Eff	g.l	b	a.l	n.g
10	8	3.2	1.350	1	16	50	2.5
10	10	3.2	1.000	1	16	50	3
10	11	3.2	1.000	1	16	50	3.5
20	650	6.2	1.005	10	32	50	7
20	420	6.2	1.002	8	32	50	7
20	369	6.2	1.010	8	32	50	6.2
20	295	6.2	1.014	7	32	50	6.2
20	260	6.2	1.092	7	32	50	5.0
50	429	6.8	1.000	7	32	100	6.0
50	360	6.8	1.046	7	32	100	5.28
50	300	6.8	1.000	6	32	100	5.28
50	2812	8	1.022	10	32	100	32
50	1474	8	1.105	5	32	100	43
50	1211	8	1.037	4	32	100	46

Protocol overhead is an important parameter in ad-hoc routing protocols. The measurement of protocol overhead during the convergence time has also been appended for ad-hoc networks.

Here, *p.o* is the protocol overhead per link during the convergence time (*c.t*) and *n.t* shows the density of nodes, dividing networks to sparse (*s*) and dense (*d*) types.

VI. CONCLUSION

The quick convergence and little protocol overhead for small networks are obvious from tables 1 and 2. It is enough to say that for small networks the convergence is possible even with genes of length one, which means the convergence occurs in one step, in comparison with RIP which needs n steps to converge (n is the number of nodes).

Another result of simulations is that genetic routing mechanism might be highly advisable in networks with locality in connections, like ad-hoc networks. Although convergence time increases fast as network size grows, protocol overhead grows reasonably. In the proposed method of periodic convergence for MANETs, by extending the convergence time to convergence period, the amount of protocol overhead would come even lower. Having all these

results with the merits mentioned in 3.1, reveal that the proposed mechanism of routing is really attractive.

TABLE II. CONVERGENCE TIME AND PROTOCOL OVERHEAD FOR GENETIC ROUTING IN AD-HOC NETWORKS.

n.s	n.t	c.t (ms)	n.w	p.o/link (Bytes/s)	g.l	b	a.l	n.g
10	s	12	4.3	277	1	16	50	5
10	s	4	4.3	333	1	16	50	2
10	s	9	4.3	296	1	16	50	4
10	s	13	4.3	307	1	16	50	6
10	s	10	4.6	357	1	16	50	5
10	s	9	4.6	238	1	16	50	3
10	s	9	4.6	317	1	16	50	4
20	s	52	7.0	720	2	32	50	10
20	s	49	7.0	942	3	32	50	5
20	d	11	6.4	275	1	32	50	5
20	d	23	6.4	263	1	32	50	10
50	s	771	17.0	970	4	32	100	50
50	d	540	8.0	439	3	32	100	50
50	d	215	8.0	444	3	32	100	20
50	d	545	8	503	4	32	100	30

Future work in this line of research can: a) search for a distributed evaluation function to optimize the variable address network's routing mechanism, b) look for possibility of finding a Nash-equilibrium which also maximizes the network throughput based on the proposed schema of reverse path charging, c) simulate alternative implementations of genetic routing with different evaluation functions, d) simulate efficiency of suggested optimum stateless multicasting and anycasting, e) further attempt in integrating ad-hoc issues in the model and verification of their functionality, and f) extensively run simulation experiments for evaluation of efficiency of proposed mechanism for power management.

REFERENCES

[1] S. Savage, A. Collins, E. Hoffman, J. Snell, T. Anderson. The End-to-End Effects of Internet Path Selection. Proc. 1999 ACM SIGCOMM Conference, August 1999, pp 289-299.

[2] S. Singh, M. Woo, C. S. Raghavendra. Power-Aware Routing in Mobile Ad Hoc Networks. In Proceedings of the 4th annual ACM/IEEE International Conference on Mobile Computing and Networking, 1998, pp 181-190.

[3] V. Houlden, N. Davies, J. McGinn, S. Cunningham, A Unified Framework for Optimal Routing, Proceedings of CSSI International Conference System Integration in Integrated Europe (ISIE), Liberec, Czech Republic, 8th-9th November 2004, pp 177-187.

[4] M. Cardei, M. Xiaoyan Cheng, X. Cheng, D.-Z. Du, Connected Domination in Multihop Ad Hoc Wireless Networks, JCIS, 2002, pp 251-255.

[5] M. Falkner, M. Devetsikiotis, I. Lambadaris, An overview of pricing concepts for broadband IP networks, IEEE Communications Surveys & Tutorials, 2nd Quarter, 2000, pp. 9-13.

[6] V. Heuven, P. Coppens, J. V. den Berghe, S. Colle, P. Demeester, Quantitative and theoretical analysis of recovery convergence in IP networks, 10th International Conference on Software, Telecommunications and Computer Networks. SoftCOM 2002, 2002, pp 209-13.

[7] A. Striegel and G. Manimaran, A Survey of QoS Multicasting Issues, IEEE Communications Magazine, June 2002, pp. 82-87.

[8] C. Labovitz, G.-R. Malan, and F. Jahanian, Origins of Internet Routing Instability, in Proc. of IEEE INFOCOM 1999, New York, NY, Mar 1999.

[9] A. Shaikh, L. Kalampoukas, R. Dube, A. Varma, Routing stability in congested networks: experimentation and analysis. Proceedings of ACM SIGCOMM 2000.

[10] D. Caro, G. Dorigo, An adaptive multi-agent routing algorithm inspired by ants behavior. In: Proceedings of PART98-- 5th Annual Australasian Conference on Parallel and Real-Time Systems. Springer-Verlag, Berlin, pp. 261-272.

[11] E. W. Zegura , K.L. Calvert, M. J. Donahoo, A quantitative comparison of graph-based models for Internet topology, IEEE/ACM Transactions on Networking (TON), v.5 n.6, 1997, pp770-783.

[12] S. Butenko, X. Cheng, D.-Z. Du, P. M. Pardalos, On the construction of virtual backbone for ad hoc wireless networks, volume 1 of Cooperative Systems, chapter 3, Kluwer Academic Publishers, January 2003, pp 43-54.

[13] E. Altman, T. Boulogne, R. El Azouzi, T. Jiménez, L. Wymer. A survey on networking games in telecommunications. Computers and Operations Research, in press, doi: 10.1016/j.cor.2004.06.005.