Adaptive Routing in Active Networks

Giuseppe Di Fatta[1,3], Salvatore Gaglio [1,2], Giuseppe Lo Re[1], Marco Ortolani[2]

[1] CERE, Centro di studio sulle Reti di Elaboratori,
Consiglio Nazionale delle Ricerche,
[2] DIAI, Dipartimento di Ingegneria Automatica e Informatica
[3] DIE, Dipartimento di Ingegneria Elettrica
Università di Palermo,
Viale delle Scienze, 90128 Palermo, Italy
e-mail {difatta, gaglio, lore, ortolani}@cere.pa.cnr.it

Abstract— **New conceptual ideas on network architectures have been proposed in the recent past. Current store-and-forward routers are replaced by active intermediate systems, which are able to perform computations on transient packets, in a way that results very helpful for developing and deploying new protocols in a short time. This paper introduces a new routing algorithm, based on a congestion metric, and inspired by the behavior of ants in nature. The use of the Active Networks paradigm associated with a cooperative learning environment produces a robust, decentralized algorithm capable of adapting quickly to changing conditions.**

Keywords— **Artificial life, Cooperative Learning, Active Networks.**

I. Introduction

PACKET loss in todays Internet is mainly due to congestion [1][2]. The nature of the environment, together with the consolidated use of end-to-end paradigm has focused the attention on transport level as the proper place to address congestion issues: overwhelmed by excessive traffic, routers are forced to discard packets and, after noticing a loss, sources usually react by reducing their flow. Indeed, this is a response to congestion, but it has the unavoidable side effect of reducing the throughput.

We propose to address this problem from a different point of view, namely at network level, by providing a routing protocol whose metric is based on the overall congestion level of links and routers. This means that we choose load balancing as a method to reduce congestion as opposite to flow control. These alternatives are not mutually exclusive, but rather complementary services, which may be used in combination to provide an efficient behavior at upper level.

Congestion may be thought of as a stochastic phenomenon in a highly dynamic environment; an adaptive algorithm thus appears to be a very promising approach.

Active Networks [3] provide a powerful means to develop it in a distributed fashion and the Ants computational paradigm [4] minimizes the undue oscillations in the routing tables such an algorithm is likely to produce, thanks to their characteristics of reinforcement learning. In the following sections we present the algorithm, after explaining the techniques that allow its realization.

The remainder of this paper is structured as follows: Section II presents the Active Networks architecture and their advantages. Section III describes the Ants computational paradigm. Section IV illustrates the algorithm as implemented in our project and Section V indicates some further developments. Finally Section VI concludes the paper.

II. Active Networks

Active Networks [5][6][7] provide programmers with the capability of analyzing and modifying messages flowing through the internal nodes of the network. The need for this new conception can be traced back to the difficulty experimented in developing new network protocols and deploying them in reasonable time as well as to the poor performance achieved by current protocols in terms of adaptability and throughput.

In Active Networks, active packets carry not only data but also suitable code portions that will be executed at intermediate nodes as they walk through the net. Several problems arise from this new approach, the most remarkable of which regards security, but they are balanced by the new possibilities offered by this new technology.

Active Networks may be regarded as the natural evolution of current ones. Usual packets might be already considered as program carriers, but their functions are so limited that basically they only accept a payload and specify an evaluation destination; moreover the evaluation consists only in the extraction of data. The payload is in fact made of passive data that are just passed to upper services; instead in an active environment it is replaced by a program (which may contain data as well). The network designer thus needs to provide the user with a programmable network API, a virtual machine suitable for the programmer to inject their active packets into the network. An active environment can thus be divided into a fixed part (the API) and a variable, user-defined one. If the API defines a complete Turing machine, then the programmer has virtually no limitations in determining the node behavior; on the other hand, the fixed part could accept only some predefined parameters and little variations from a standard behavior are allowed. In the former case it may be impossible to understand the influence of single nodes on the whole behavior of the net, while the latter one is more or less the common case in the current Internet, where the possible choice for parameters is very limited. Clearly a better choice lies somewhere in the middle.

The language we chose for our project, PLAN [8], provides only basic primitives for the delivery of packets, leaving to programmers the possibility of defining and deploying specialized services written in Ocaml [9], a functional language from the ML [10] family.

PLAN just acts as the unifying tool for user-defined protocols, but it also guarantees that the resulting programs will not break safety or security constraints by ensuring that they always terminate and that packets and their descendants visit only a fixed number of nodes [11].

All the discussion above is meant to introduce the idea that not only do Active Networks provide a new, more effective, way to implement usual protocols, but also they are suitable to introduce some form of Artifcial Intelligence in the network. The essential point is that distributed computation is made much easier and this is fundamental especially when the intelligent behavior is obtained via the use of cooperative agents, as is the case of the ants paradigm.

Given two nodes we have to find a route between them which does not suffer from congestion but which also optimizes the usage of resources so that other routes can be found with the same characteristics for other pairs of nodes. This task may involve a great number of nodes in a highly mutable environment (assigning a route also changes the situation as regards congestion), so that the use of a centralized algorithm is to be avoided.

The reason is due not only to the obvious problems arising from possible malfunctioning of the central control point, but also because information on the state of remote parts of the network would need to be up-to-date as more as possible to be useful. This is clearly not possible in a congested environment such as the one where we are supposed to operate into.

III. Ants in nature

A good model for a totally decentralized, yet robust and adaptive algorithm can be derived from the analysis of ant colonies in nature. Ants [4][12] are so simple insects that they can hardly be considered intelligent. The interest about them derives from their peculiar way of interacting with each other; in fact they just react to local environmental stimuli though in the end they reach a global optimal behavior.

They show great ability in such tasks as:

- regulating nest temperature within limits of 1C;
- raiding particular areas for food;
- sorting brood and food items;
- cooperating in carrying large items;
- finding the shortest routes from the nest to a food source;
- preferentially exploiting the richest available food source.

Ants do not have a central organization, nor do they directly communicate with each other, rather they use a form of indirect communication through the environment called stigmergy. As they walk from the nest towards the food source they deposit a volatile substance called pheromone, thus modifying the environment in a way not directly related to their task. Ants may thus lay these trails when going towards their destination, when coming back or in

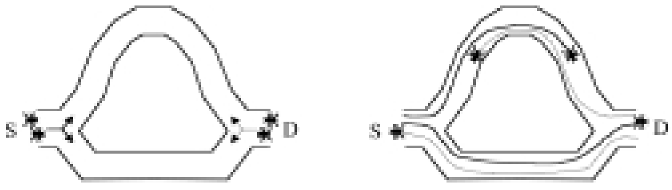

Fig. 1. Choosing a route randomly, ants that follow the shortest one have more pheromone released on it.

both ways; anyway the amount of pheromone laid will depend on the time taken to reach the food and it will not persist indefinitely, but it will evaporate unless it is not periodically renewed. On the other side, future choice of a direction will be influenced by the pheromone already present; this means that pheromone represents the attractiveness of a particular route to ants, but only within a short period. Let us consider figure 1.

This is a simplified abstraction of a common situation. At the beginning all ants start exploring the neighborhood, i.e. they choose the direction totally randomly; the ants that choose the "best" (i.e. the shortest) route will complete their round trip in a shorter time, thus depositing more pheromone per time unit along their route. Following ants will prefer that route because of the higher amount of pheromone, as a consequence less and less pheromone will be released on the remaining paths, so in the end all of them will converge to the optimal route.

IV. The Algorithm

As stated above, PLAN lets us define our own primitives in the form of services to be installed on every node. Our protocol will be a collection of such primitives resident on the active nodes and, since it will operate in a distributed fashion, communication will be carried on through the use of small software agents (i.e. PLAN packets representing artificial ants).

In traditional routing protocols, nodes basically maintain a table with one entry per each destination in the network. In this way, it is always possible to decide which is the neighbor to choose as next hop in order to reach a given destination, minimizing the total cost according to the fixed metric.

In this case, given a destination, the table must keep as many entries as the neighbors of the current node; every ant arriving to this node from one of its neighbors will increase the amount of pheromone related to that particular neighbor.

The underlying mechanism is a positive feedback: the characteristics of the final results of the system are magnified, thus very quick convergence or uncontrolled growth is obtained; this means that ants will need a global rule to manage the releasing of pheromone. In fact trail laying may be performed either on a hop-by-hop basis, with ants having no idea of the overall goodness of the route they are selecting (much as in a greedy search), or with a global view of the situation. In both cases sub-optimal solutions may be found, but the latter is by far more attractive, because it provides the necessary control to bias the solutions towards the best one.

The algorithm can be outlined as follows: every node periodically selects a random destination and sends an active packet (forward ant), which will record the nodes traversed and the arrival time at each node. When the final node is reached, the active ant will have another packet sent back to the source; this new ant will follow backwards exactly the same route as its parent according to the information received at creation time. At each node it will compare its arrival time with the time recorded by the forward ant, thus producing an estimate of the round-trip time, from the current node to the destination, that may be used to modify the pheromone table.

As underlined in [13], the updating of the pheromone in the tables needs some accuracy in order to be as insensitive as possible to meaningless fluctuations in the estimated times and some corrections may need to be applied. In this model ants are allowed to release pheromone only on their way back; the reason is that we can not assume to be working in perfectly symmetrical conditions: this may be the case when we think about the links (if they are bi-directional, they are congested in both ways), but congestion shows up inside the routers too.

Moreover, if the pheromone is deposited only after the destination is reached, then its amount may be a function of a global index (such as the round-trip time), thus providing the control required to direct the global performance of the ants.

A few points are noteworthy:

• *distribution:* no central intelligence is needed; actually there is not even one "intelligent" agent, because ants too are theoretically just simple pieces of software carrying only basic information about their "being an ant". The global intelligence arises from the interaction, thus it is spread all over the nodes in the network;

• *robustness:* this is another fundamental feature; we must assume the presence of congestion, so that communications (usually unreliable) are even more irregular. The loss of a single ant does not compromise the algorithm as a whole, since no fundamental information gets lost; at most convergence will be slower;

• *efficiency:* ants may require some computations inside routers, since pheromones must be somehow processed to obtain correct routing tables. However Active Networks aim exactly to provide the user with more computational capability inside the network (i.e. in intermediate nodes); moreover ants are good at saving bandwidth, which is our main concern.

At the end we obtain a coherent set of routing tables, which results in optimal exploitation of available resources; of course this cannot be a static assignation, because of the great variability in the state of the nodes. Nevertheless ants are designed to cope with this, as is evident when one considers two typical problems: the blocking problem and the shortcut problem [12]. Both of them occur when a change in network topology takes place, e.g. new paths become available or others go down; for example, when a router suffers a damage, all the routes passing through it must be changed.

This is actually done quite easily, because from ants point of view it is as if they have to face a wall; they automatically try to overcome it and adjust the pheromone tables as a consequence, so that future ants will follow new routes. When the damage is repaired, however, ants are forced to follow already established routes because of the reinforcement learning mentioned above, but in this way new more attractive routes would never be explored and the global state would not be the optimal one. The solution is hidden in the actual representation of virtual pheromone.

Pheromone tables replace routing tables in our case so one could imagine that, given a source and a destination, for each outgoing physical link at the source, ants using it deposit pheromone on it. The link with the largest amount of pheromone will be chosen. If we normalize all these quantities to the unit, then they can be regarded as the probability for an ant to choose a particular direction to its destination; the event that the route taken is another one is not impossible, but less probable, thus allowing for some exploration.

V. Further Developments

Simplicity is indeed one of the main features ants can offer; they do not carry any information, except about their nature, so that a new traditional communication protocol running over IP could be created. Nevertheless this approach has proved to be not very effective because of long deployment time and backward compatibility problems.

Currently no modification could be introduced in a routing protocol until general agreement is reached, whereas an active protocol may be designed so as to be able to coexist with older ones and could be deployed instantaneously.

If artificial ants are implemented as active packet, new functionalities can easily be added. For instance, in the Internet context such a protocol would be placed at the same level of traditional routing protocols (network layer), i.e. it uses the services provided by IP and supplies its services to upper transport protocols such as TCP or UDP.

Let us restrict to our case: ants as PLAN packets. A PLAN packet is evaluated only at the final destination in order to limit an excessive overhead, although at each node it may choose a routing function among the several ones offered as services. An ant, at its arrival at a node, would check if the proper routing function is present, otherwise it could resolve to use the default one; the ant would anyway update its own data, while it could not update any pheromone table when coming back. The algorithm would work anyway.

If the node were not active at all, the ant would only be tunnelled through it. We could also assume that the node initial knowledge about the network is restricted only to its neighborhood. We may slightly modify the ants to make them retrieve new information from the nodes they visit; namely they could report a list containing the neighbors of the routers encountered during the trip so that at the end every router would have a global view of the network.

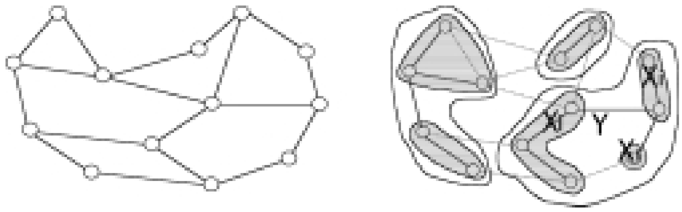

Fig. 2. A topology and the regions arising when clustering with respect to several levels of bandwidth. Colored regions represent nodes connected by links with higher free bandwidth (solid lines). Dashed lines indicate more congested links.

This simple change might bring interesting consequences, not limited to a better scaling up.

In [14] the authors suggest a new way to provide an active environment with some QoS. The idea is that some thresholds for bandwidth should be chosen so that nodes could be dynamically clustered in areas called blocking islands. Every couple of nodes belongs to the blocking island labelled with XMb/s, if at least one route with as much free bandwidth between them exists, see figure 2. If $Y < X_i$ then a blocking island labelled with YMb/s will group together all the X_iMb/s blocking islands; of course this division needs to be adaptive. As stated above, an ant will have the task of exploring the network to collect as many addresses as possible in order to widen the network view of its parent nodes.

Let us assume that the ant is aware of the bandwidth thresholds and consider two of these: X and Y. The ant could take as its search space only the nodes it can reach with routes of available bandwidth X; when coming back to the source it would report only about such nodes, so that a cluster of the lowest level would automatically be made. When the boundary of the blocking island is reached one can imagine that the ant simply returns and, if the destination was not reached, reports the corresponding information back to the source; what it actually could do is: generate a backward ant with the clustering information and at the same time a new upper level ant to deal with YMb/s blocking islands. In this way a hierarchy of ants is created; at each level the algorithm is the same and thus it is highly adaptive and robust.

VI. Conclusions

Routing is traditionally based on static metrics so that, even if the state of the links is taken into account, decisions are often taken regardless of the actual current state of the network. In a congested environment the situation becomes increasingly worse, because theoretically only instantaneously up-to-date information is required. We present an adaptive approach which, unlike usual AI techniques, is not too demanding in terms of computations in order to allow for the necessary speed; artificial ants are in fact sufficiently light-weighted and purport to obtain good performance.

The possible drawbacks are represented by oscillations in the suggested routes, which are likely to happen because of the stochastic fast changes in the state of the links; nevertheless [13] showed that, with some accuracy, this can be avoided.

Active Networks also allows us to get rid of the limitations deriving from the traditional end-to-end model. The network is forced to cooperate and some state is actually maintained inside it, even if we do not have to trust it, thanks to the robustness of the algorithm: the overall information on a route is used, but occasional failures in the intermediate nodes do not lead to inconsistency.

Also, the use of Active Networks is promising because future extensions are more naturally designed. It is in our intentions to explore these new possibilities as regards for instance a better scaling-up. We also aim to get a deeper insight in the mechanisms of active clustering, which appear to be suitable for an implementation with Ants; yet it deserves further analysis to solve problems like address resolution (adaptive schemes can not be easily used for this purpose).

References

[1] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397+, Aug. 1993.

[2] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 458+, Aug. 1999.

[3] D. L. Tennenhouse, J.M. Smith, W.D. Sincoskie, Wetherall D.J., and G.J. Minde, "A survey of active network research.," *IEEE Communications Magazine*, vol. 35, no. 1, pp. 80 – 86, Jan. 1997.

[4] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents. part b," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 26, no. 1, pp. 1–13, Apr. 1996.

[5] K.L. Calvert, S. Bhattacharjee, E. W. Zegura, and J. Sterbenz, "Directions in active networks," *IEEE Communications Magazine*, vol. 36, no. 10, pp. 72–78+, Oct. 1998.

[6] D. L. Tennenhouse and D.J. Wetherall, "Towards an active network architecture," *Computer Communication Review*, vol. 26, no. 2, Apr. 1996.

[7] D.J. Wetherall, U. Legedza, and J. Guttag, "Introducing new internet services: Why and how," *IEEE Network Magazine*, vol. 12, no. 3, May 1998, Special Issue on Active and Programmable Networks.

[8] M. Hicks, P. Kakkar, J. T. Moore, C. A. Gunter, and S. Nettle, "Plan: a packet language for active networks," in *Proc. of Third ACM SIGPLAN Interneational Conference on Functional Programming Languages*. sep 1998, pp. 86–93, ACM.

[9] Xavier Leroy, *The Caml Special Light System (Release 1.10)*, INRIA France, 1995.

[10] R. Milner, M. Tofte, and R. Harper, *The Definition of Standard ML*, MIT Press.

[11] M. Hicks, J. T. Moore, D. Scott Alexander, C. A. Gunter, and S. Nettle, "Planet: an active internetwork," in *Proc. of the Eighteenth IEEE Computer and Communication Society INFOCOM Conference*. IEEE, 1999, pp. 1124–1133.

[12] R. Schoenwoerd, O. Holland, J. Bruten, and L. Rothkrantz, "Load balancing in telecommunication networks," *Adaptive Behavior*, vol. 5, no. 2, 1997.

[13] G. Di Caro and M. Dorigo, "Mobile agents for adaptive routing," in *Proceedings of 31 Hawaii Internetional Conference on Systems Sciences*, The Big Island of Hawaii, jan 1998, HOCSS-31, The Software Technology Track.

[14] S. Wilmott and B. Faltings, "Active organizations for routing," in *Active Networks*, Stefan Covaci, Ed., Berlin, June/July 1999, number 1653 in Lectures Notes in Computer Sciences, pp. 262+, Springer.