A Weakly Coupled Adaptive Gossip Protocol for Application Level Active Networks

Ibiso Wokoma, Ioannis Liabotis, Ognjen Prnjat, Lionel Sacks, Ian Marshall
Department of Electronic and Electrical Engineering, University College London,
Torrington Place, London WC1E 7JE, England, UK
email:{ i.wokoma, i.liaboti, o.prnjat, lsacks, imarshall} @ee.ucl.ac.uk

Abstract

With the sharp increase in heterogeneity and distribution of elements in wide-area networks, more flexible, efficient and autonomous approaches for management and information distribution are needed. This paper proposes a novel approach, based on gossip protocols and firefly synchronisation theory, for the management policy distribution and synchronisation over a number of nodes in an Application Level Active Network (ALAN). The work is presented in the context of the IST project ANDROID (Active Network Distributed Open Infrastructure Development), which is developing an autonomous policy-based management system for ALAN. The preliminary simulation results suggest that with the appropriately optimised parameters, the algorithms developed are scalable, can work effectively in a realistic random network, and allow the policy updates to be distributed efficiently throughout the active network with a lower latency than other similar types of gossip protocols.

1 Introduction

The process of efficiently distributing large amounts of information in communication networks has always been a critical issue. Traditional methods of communication based on strong consistency protocols and flooding are becoming inefficient for quickly changing information or when faced with situations in which failures are inevitable. Similarly, the traditional centralised approach to network and service management does not work well in large complex heterogeneous networks, and the need for an automated, distributed and decentralised management approach is rising.

Gossip protocols [1] provide a robust technique for distributing replicated data in wide-area networks. Messages can be propagated from one node of a group to another until all the nodes in the group receive the message. This form of event message delivery means that the nodes are guaranteed to receive the message even if some nodes become disconnected. A modified form of TSAE (Time-Stamped Anti-Entropy) [2] is used in this paper to demonstrate how policies can be distributed in an active network. It is used in conjunction with an algorithm based loosely on the interaction of fireflies; the event synchronisation of their flashing lights when they form a group can be likened to the synchronisation of messages in a group of nodes.

This paper considers the issues of management policy distribution in the context of the IST project ANDROID (Active Network Distributed Open Infrastructure Development). ANDROID is developing a policy-based [3][4] management system for application-level active networks (ALAN) [5]. The ANDROID project has adopted a policy-based management approach to cater for this [6]. A management policy could apply to a group of nodes positioned in different areas of the network. Thus, the management policies, as they are introduced in the managed active network, need to be efficiently distributed, the rest of the group must be made aware of the change. This is achieved through efficient propagation of copies of the new policy within the group.

This paper focuses on mechanisms for policy distribution and synchronisation over a number of active servers that would allow the management system to become more adaptive and autonomous to overcome the shortcomings of centralised control. The proposed mechanisms can be applied to many systems requiring the same distributed peer-to-peer approach to information sharing such as applications in ubiquitous computing and sensor networks.

2 Background

2.1 Gossip Protocols

In systems where information is replicated in several places, updates must be propagated to all the nodes efficiently to maintain consistency, *i.e.* they all receive the same set of messages; but any trade-off with availability should be avoided. For example in mobile telephony, huge

centralised directories are required and lack of capacity or inconsistency causes denial of service. If strong consistency is not required the system can have higher availability. Weak consistency protocols can offer appropriate solution because after temporary divergence from the "correct" value, all the nodes will eventually be updated when the devices are reconnected. This is completely sufficient for many policy based management tasks. The advantages are that out-of-date nodes can still be accessed giving better availability and the delay in propagation allows the updates to occur when the system is less loaded. As a result, gossip can be described as a type of weak consistency protocol where update requests are propagated via delayed point-to-point communications between nodes in batches.

TSAE is a gossip protocol that gives the benefits of weak consistency while keeping the network traffic low and maintaining the updates at each node. Each node stores a log of the update information with a record of the time it was received. When a node updates its' neighbour, an anti-entropy session takes place where the content of their logs are exchanged similar to Bayou's anti-entropy protocol [7]. The nodes can determine the policies they do not have through the exchange of summary vectors and receive new policies exactly once thus minimising traffic. At the end of the session both nodes have the same logs.

Implementing TSAE involves considering the partner selection strategy where the node selects the neighbour it wants to communicate with. The partner selection strategy in this paper depends on the theory behind firefly flashing.

2.2 Fireflies

Self-organised criticality is one of the core ideas that emerge out of complex system theory where a system with dynamical non-equilibrium statistical properties evolves to a critical state without altering external parameters. Systems with these features can give rise to useful large-scale attributes (emergence) without rigid engineering and control. Many biological systems evolve in this way to have coherent behaviour out of seemingly random low-level activities making these systems useful analogies for analysing non-linear technical systems [8]. The biological model used here is the phenomena of firefly synchronisation. Fireflies are known to emit flashes at regular intervals when isolated but when they come together, they entrain the pulsing of their lights to converge upon the same rhythm as that of other fireflies in the group until synchronicity is reached [9].

Fireflies flash at a predetermined point in a periodic oscillation that can be modelled as a periodic counter driven by an internal clock. Synchronicity is achieved in a group of fireflies when the fireflies can observe the flashes of their neighbours. This is achieved through the simple mechanism of advancing the clock cycle whenever one sees a flash, unless one was also flashing. The fireflies are said to be pulse-coupled because one firefly affects the state of the others only when the firefly flashes.

This phenomenon is explained by Peskin's model [10], which gives an equation that models the interaction of two pulse-coupled oscillators and proves that the oscillators will almost always synchronize if they obey his equation. Renato E. Mirollo and Steven H. Strogatz [11] went on to prove that this was true for a network of any number of identical pulse-coupled integrate-and-fire oscillators by basing their proof on absorption, which occurs when two oscillators of different phase synchronize and remain locked in phase.

When relating this to a group of nodes in an active network, the synchronisation of flashes can be equated to the synchronisation of the rate of updates. This rate is likely to vary when new policies are added to the network but the nodes take steps on low-level basis to make the rate identical on a global scale. This is achieved at synchronization when the network is consistent with all the nodes having logs with the same policies.

3 Testing the Firefly/Gossip Model

The model consisted of a square lattice network of nodes that were each connected to their nearest neighbours. Each node had a timer that was decremented each time step, a store that held a record of the policies at the node and a group membership label that determined the subset of visible neighbours the node was authorised to communicate with. The nodes also had a flash interval that determined when the node was next going to talk to other group members to check for new policies, i.e. "flash", and an event record that allowed the node to select which group members to talk to according to how frequently they flashed. The network was simulated and the tests in the following sections were carried out.

3.1 Optimisation

A lattice network of 100 nodes was created and the optimal conditions for the network operation were found by changing variables to observe the effect two parameters. The first is the synchronisation time, which is the time it takes for a policy to spread throughout network, and the second is the network traffic, the amount of communication between flashing nodes and the surrounding neighbours.

The nodes had a maximum flash interval to limit the average flash interval increase due to the addition of new policies. If the average was below the maximum flash interval then it was because a new policy had been added to the network. The synchronisation time and network traffic was recorded for maximum flash intervals between 10 and 70 epochs to give the graph shown in Figure 1.

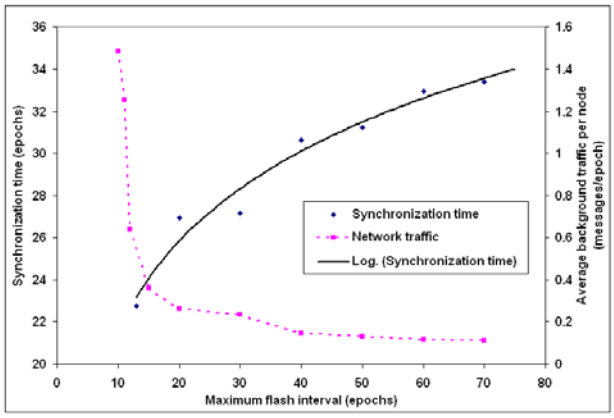

Figure 1: Graph to show the effect of varying the maximum flash interval between 10 and 70 epochs on the synchronisation time and network traffic for a lattice network of 100 nodes

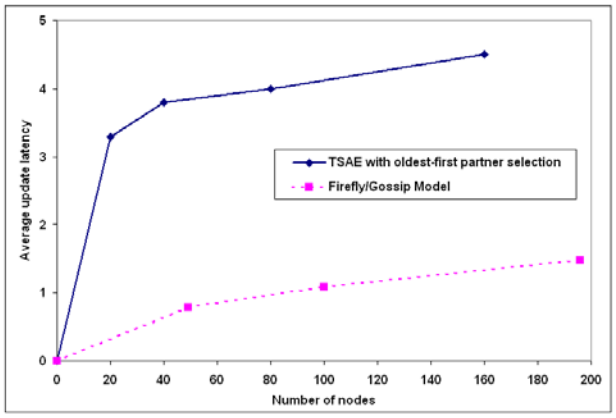

Figure 2: Graph to show how the average update latency varies with the number of nodes for the firefly/gossip model and for the TSAE with oldest-first partner selection strategy

Figure 1 shows the variation of the synchronisation time with the maximum flash interval. The network spends more time in a consistent state when the maximum flash interval is low but consideration must be given to the high amount of traffic caused by the low maximum flash interval.

When deciding on a good operating point for the network, the requirements for low synchronisation time and low background network traffic have to be achieved, hence, from Figure 1 a maximum flash interval of 20 epochs seemed to be a reasonable choice for this case. For higher flash intervals from Figure 1, there are very low reductions in traffic but higher synchronization times.

3.2 Scalability

The number of nodes on the lattice was varied between 7^2 and 13^2 nodes and the synchronization times for network were recorded and used to find the update latency. This is the synchronization time measured as a multiple of the maximum flash interval. The graph in Figure 2 shows how the update latency varies with the number of nodes for the firefly/gossip model and for TSAE experiments described in [2]. Different partner selection strategies and distributions of anti-entropy sessions were tested to find the most effective way of reducing the update latency. In [2], it was found that nodes choosing to have anti-entropy sessions with the most out-of-date nodes first worked the best. This oldest-first partner selection strategy gave reasonable update latencies when the time between two successive anti-entropy sessions was generated uniformly from the time interval [0.9T, 1.1T] where T is the flash interval.

Figure 2 shows the increase in the number of nodes leads to larger synchronisation times. The synchronisation time scales well with the logarithm of the number of nodes, so a mathematical relationship can be developed to predict the synchronisation times for networks of any size.

The partner selection strategy used in this paper is based on the timing of firefly flashing with the use of event records and Figure 2 shows that this strategy is more effective than the TSAE with the oldest-first partner selection strategy because of the lower update latency. The firefly/gossip model reduces the update latency by a factor of 4 because of the dynamic nature of the network that reacts quickly to the presence of a new policy.

3.3 Different Network Topologies

A square lattice of nodes is a simple topology but does not reflect a real network. The algorithm was tested on a random network based on the Autonomous System connectivity of the Internet. The connection matrix for the network was created using an Internet Topology Generator called Inet that gives networks of any size, the smallest containing 3037 nodes. The paper in [12] compares Inet to other generated topologies and Inet -2.0 was found to be suitable for modelling Internet-like characteristics making it good for testing how well the firefly/gossip model works in a real network.

A text file of the connections between 3037 nodes was generated using Inet -2.0 to give the random network to be compared to a lattice network of 3025 nodes. The percentage of nodes that receive the new policy over time for both types of networks is given in Figure 3. Policy synchronization was achieved in the random network by increasing the amount of time the nodes had to remember their neighbours and the same was condition was applied to the lattice network to keep the test fair.

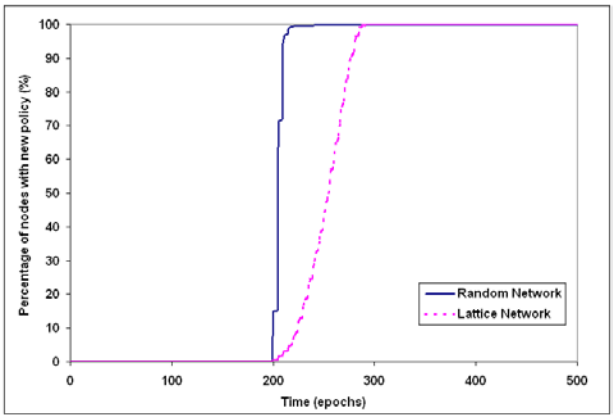

Figure 3: Graph to show the percentage of nodes that have received a new policy over time in a random network and a lattice network

Figure 3 shows that in the random network, the policy spreads very quickly in comparison to the lattice network. In the random network, the synchronisation time is 91 epochs whereas in the lattice network it is 640 epochs. Node 0 is a highly connected node that has a high number of neighbours so when a policy is added to it, it takes a short time for copies of the policy to spread to the surrounding neighbours. In comparison, the spread of the policy is more evenly distributed in the lattice network so it would take a longer time.

4 Conclusion

Policy synchronisation can be implemented by using a weakly coupled adaptive gossip protocol where the timing of the gossips is related to firefly flashing. The result is an algorithm that is effective at spreading policies in the network if certain system parameters are optimised. A maximum flash interval has to be applied to allow the nodes to keep in memory a list of the neighbours to communicate with. To achieve optimal performance of the algorithm, the maximum flash interval has to be set at a value where the background traffic and synchronization time are reasonably low.

The synchronisation time for networks of any size can be deduced from the results and can be used to program how long a node has to remember a policy before being able to delete it from its hash table. In comparison to other methods of partner selection strategies used in the TSAE, this firefly/gossip model is more effective at spreading the policies quickly.

The algorithm worked better in the random network approximating Internet connectivity than it did in a lattice network. To ensure that policy synchronization was achieved in the random network, the amount of time the nodes had to remember their neighbours was much higher for the random network than the lattice network. This means that more memory is needed for the nodes of a random network to maintain larger event records so that less connected nodes are guaranteed to receive copies of the new policy even if they do not flash frequently.

A limitation is that the experiments conducted are focused on a very simplified model of a complex network. For instance, it is difficult to determine what level of traffic is acceptable given that the nodes will have many other tasks to carry out. With more investigation this could be resolved and further work could also involve assessing how well the algorithm works in other network topologies such as ring, star, mesh, small worlds and scale-free topologies and investigating different policy arrival rates.

5 References

[1] R. A. Golding et al, “Modeling replica divergence in a weak-consistency protocol for global-scale distributed data bases”, Technical report UCSC-CRL-93-09, Computer and Information Sciences Board, University of California, February 1993.

[2] C. L. van Eijl, C. Mallia, "The time-stamped anti-entropy protocol- a weak consistency protocol for replicated data", BT document.

[3] “D1: Refined Architecture”, an ANDROID document, August 2000.

[4] M. Sloman, E. Lupu, "Policy Specification for Programmable Networks", Proceedings of IWAN '99 Conference, Springer-Verlag, 1999.

[5] M. Fry, A. Ghosh, "Application Level Active Networking", Computer Networks, pp. 655-667, 1999.

[6] Mike Fisher, Paul Mckee, et. al "Policy-based Management for ALAN-enabled networks", submitted for Policy 2002.

[7] K. Petersen et al, “Flexible Update Propagation for Weakly Consistent Replication”, Proceedings of ACM Symposium on Operating Systems Principles, pp. 288-30, October 1997.

[8] M. Resnick, “Turtles, Termites, and Traffic Jams”, MIT Press, 1997.

[9] U. Wilensky et al, “Learning Biology through Constructing and Testing Computational Theories”, Proceedings of the Second International Conference on Complex Systems, 1998.

[10] C. S. Peskin, “Numerical analysis of blood flow in the heart”, Journal of Computational Physics, pp 220-2521977.

[11] R. E. Mirollo, S.H. Strogatz, “Synchronization of pulse-coupled biological oscillators”, SIAM J. Applied Mathematics, 1990.

[12] C. Jin, Q. Chen, S. Jamin, “Inet: Internet Topology Generator”, University of Michigan Computer Science Department Technical Report, 2000.