IMPROVING THE LOCATION OF MOBILE AGENTS ON SELF-AWARE NETWORKS

Ricardo Lent
Intelligent Systems and Networks, Imperial College London, U.K.

Keywords: Mobile Agent Systems, Cognitive Packet Networks, Mesh Networking.

Abstract: Business applications are evolving on the Internet from centralised to distributed architectures demanding higher quality group communication facilities. Such communications are often time sensitive and prone to disruptions or unacceptable latency because of the high variability of Internet traffic and capacity across the network. Self-aware networks offer mesh networking facilities with adaptive routing able to cope with changes in network conditions, such as variations in traffic load and link or node failure. Further quality improvement can be achieved by relaying critical application tasks to software agents able to migrate to alternate hosts and take advantage of new location facilities for their communications. The paper examines the concurrent use of self-aware networking and software agent mobility to offer improved communication facilities to time-critical group communications. A host selection algorithm for agent migration is proposed to find suitable locations for agents and evaluated in a simulation study.

1 INTRODUCTION

A mobile agent is a type of software agent that can autonomously migrate from one computer to another to continue execution on the second host. Migration involves moving the mobile agent's code and data totally or partially, and in some cases also the execution state from one machine to another. Migration may offer interesting advantages to critical business applications that would benefit from changing their execution environment in response to a computing or communication degradation (Lange and Oshima, 1999). It would be of particular use for time-constrained applications. Because of the great variability of resources in the Internet, network performance in general can vary greatly from one network area to another and also over time. Significant network variability can introduce an undesirable uncertainty on the quality of the communications that may affect critical applications. Applications supported by mobile agents need not to be bounded by a fixed execution environment as they are able to relocate themselves and use a more advantageous location offering better communication quality.

For example, on a distributed auction system, auctioneers need to notify bidders after they receive a new offer to allow them react and send additional bids. Un-even latency and packet loss in the communications between seller and bidders may cause unfair advantage to certain participants. Another case is exemplified by the use of a central host to ensure total ordering in a chat system, whose location would determine the performance of the application. Both cases are good examples of critical real-time group communications, that could benefit greatly from the use of moving agents.

There are several comprehensive systems that support agent mobility that have been made available both from industry and academia. From the point of view of the form of mobility, those systems can be classified into one of two categories: weak mobility or strong mobility. The former form supports data and code mobility whereas the latter supports in addition a transfer of the control state of the process. The Java virtual machine natively supports weak mobility so many mobile agent systems are Java-based. Java offers portability with the use of bytecodes executed with just-in-time (JIT) compilation and host security that is enforced by virtual machines. Aglets (Tai and Kosaka, 1999), which have been developed by IBM, Voyager (http://www.recursionsw.com/,), Concordia (Wong et al., 1997) and Jumping Beans (http://www.JumpingBeans.com,) are examples of Java-based systems offering weak mobility. Tacoma (Sudmann, 1996) and Concordia are systems offering weak mobility. Tacoma offers support to multiple languages including Java. Some of the systems that offer strong mobility are D'Agents (Gray, 1995; Gray et al.,

Lent R. (2008).
IMPROVING THE LOCATION OF MOBILE AGENTS ON SELF-AWARE NETWORKS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 239-242
DOI: 10.5220/0001700802390242

2002) and Ara (Peine, 2002) both of which supporting multiple language support.

Self-aware networks are networks that exhibit a number of autonomic properties, which are useful to create effective mesh networks. These networks can dynamically reconfigure paths, both to improve communications quality and to avoid network element failures. Cognitive packet networks (CPN) is a self-aware network that utilizes reinforcement learning to dynamically adjust paths towards a user defined QoS goal. The paper's focus is close to the work of (Ranganathan et al., 1999), who investigated network-aware mobile programs, which are adaptable to network variations. Ranganathan et al. proposed a distributed latency monitor to probe the network and verify its status. Because of its structure, this work is applicable mainly to long-lasting communications and coarse-grained network variations. Other migration strategies were evaluate by (Gonzalez-Valenzuela and Vuong, 2002).

This paper tackles the problem of dynamically selecting migration hosts for mobile agents for time-critical group communications, which range from short-lived to long-lived communications. The case of short-lived communications is particularly challenging because network conditions tend to change all the time, so that long-running network monitors or static migration selection may not be effective. The paper explores the use of a self-aware network as the underlying communications infrastructure for agents, which is able to provide critical information to guide their migration. In particular, the paper uses CPN, which when coupled with mobile agents can together offer two levels of adaptation to applications. First, the CPN provides self-adaptable routing for improved quality of service (QoS). Second, the QoS of communications can potentially improve more by selecting an appropriate node from which to host the communications.

2 SELECTION OF MIGRATION HOST

Given a network represented by the graph $G = (V,E)$, where an agent located at node s has established a time-critical communication with N other nodes: $t_1,t_2,\ldots,t_N$ and assuming that any node is able to host mobile agents, the problem is to identify the most suitable host $v^* \in V$ for a particular agent that minimizes the cost of its group communication with nodes $t_1,t_2,\ldots,t_N$. The solution would require knowledge of the underlying network infrastructure along with an observation of the network conditions.

The use of CPN or comparable self-aware network as the supporting communications infrastructure would allow both route adaptation and to expose some of the information that agents would need to make a proper migration decision. In CPN, paths are constructed dynamically and updated over time. CPN nodes can temporarily store a number of last discovered paths (i.e. store the set $\Pi^q_{s,t} = \{P^q_{s,t} | P^q_{s,t}$ is a path from s to t built with routing goal $q\}$). A path $P^q_{s,t}$ defines a sequence of nodes that dumb packets visit: $P^q_{s,t} = v_1,v_2,\ldots,v_k$, where $k = |P^q_{s,t}|$ is the length of the path in number of hops and $v_k = t$.

Assuming additive cost metrics, the cost of path $q(P^q_{s,t})$ is:

$$q(P^q_{s,t}) = \sum_{i=1}^{k} q(v_i,v_{i+1})$$

where $q : (v_i,v_{i+1}) \to \Re$, so that $q(v_i,v_{i+1})$ is the observed cost on edge (v_i,v_{i+1}). The observed cost of the path is continuosly acquired by CPN packets.

Let Π_s be the set of all paths known by node s: $\Pi_s = \{\Pi^q_{s,t} | \Pi^q_{s,t}$ is a set of paths from s to t built with routing goal $q\}$ and $G' = (V',E')$ be the inferred graph from Π_s by making $V' = \{v | v$ is a vertex of a path in $\Pi_s\}$ and $E' = \{e | e$ are two successive vertices of a path in $\Pi_s\}$.

Likewise, let Q_s be the function $Q : (v_i,v_j) \to \Re$ such that $(v_i,v_j) \in E'$ and $Q(v_i,v_j) = sel\{q(v_i,v_j) : (v_i,v_j)$ is an edge of a path in $P_s\}$, where *sel* is a function selecting the desired cost to work with, for example: minimum, maximum, most recent, etc.

Group Distance. $\Phi : \{u,s,t_1,\ldots,t_N\} \to \Re$, such that

$$\Phi(u,s,t_1,\ldots,t_N) = \sqrt{c_0|P_{u,s}|^2 + \sum_{i-1}^{N} c_i|P_{u,t_i}|^2}$$

for a choice of constants c_i, which weight the importance of the path cost between the originating node (s) and group members (t_i). We will refer to s and $t_1,\ldots,t_k$ as the *reference nodes*. For $c_i = 1$, Φ represents the Euclidean distance from the selected migration host u to the reference nodes.

The problem is to find a node u in graph G' (and ultimately in G) such that:

$$\phi(u,s,t_1,\ldots,t_k) = \Phi(u,s,t_1,\ldots,t_N) : u \in G'\}$$

The value of Φ can be calculated from the computation of the shortest path cost from each reference node to all V' nodes. The computation has a complexity of $O(NE' + NV' logE')$ when done with the Dijkstra's algorithm implemented with a priority queue (Fibonacci heap).

Expected Trend. It is reasonable to expect that for a large migration rate τ_m^{-1} with respect to the group communication lifetime and a supporting self-aware network with reasonable active flows, within a finite time:

$$\phi(u,s,t_1,\ldots,t_k) \rightarrow \phi(v^*,s,t_1,\ldots,t_k)$$

that is, the agent will tend to move to the host in G offering the minimum distance. The case is studied in simulation in the next section.

3 EVALUATION

A simulation study was conducted to measure the potential benefits of introducing a selection of the migration host for agents exploiting the information gathered by a CPN. The simulation used INES (Lent, 2008).

The network topology in the study was a squared grid of 16 nodes per side (256 nodes in total) and full-duplex links with exponentially distributed propagation delays and transmission rates (with parameters 1 ms and 512 Kbps respectively). Each network interface had a buffer capacity of 100 packets. In the simulations, CPN was enabled to use external routing information (i.e. from a non-dynamic routing algorithm) to speed up the initial path setup.

The simulations used a 0.2 smart packet ratio with a 0.01 probability of random selection. The maximum number of hops was set to 64 and the routing goal of smart packets was low latency. A full explanation of these parameters can be found in the literature (Gelenbe et al., 2002; Gelenbe et al., 2001).

The group communication application consisted of a mobile agent sending traffic (10,000 packets of 256 bytes at 1 Kbps CBR) to a set of N destinations selected randomly at the beginning of the simulation. The mobile agent started at a random location and it tried to migrate to an alternate host after any packet transmission, but in not less than τ_m seconds. Parameter τ_m defines the minimum interval between migrations (τ_m^{-1} is the maximum migration rate). Note that it is not guaranteed that a migration will occur every τ_m seconds. A migration will occur only if a better location can be recognized with the information available from the CPN. Individual transmissions used unicast and they were sent one after another to the members of the group each time.

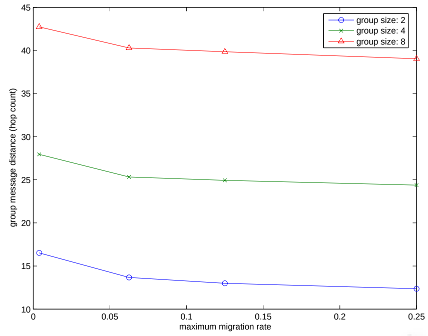

Figure 1: Average message group distance (hops) versus τ_m.

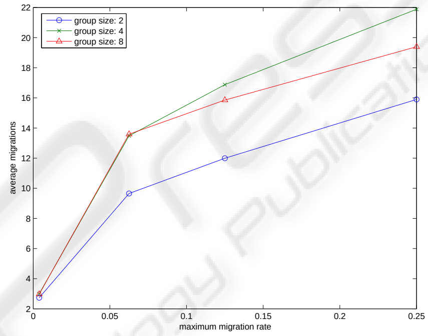

Figure 2: Average number of migrations versus τ_m.

The simulation allowed to store the last 20 routes discovered by smart packets. The stored routes were accessible to the mobile agent to build a partial view of the network to select an appropriate host to migrate. The experiments considered mobile agents that used migration to try to reduce the group distance calculated from the individual path delays to each of group members. The group size was 3 with $\tau_m = 2sec$.

Each message sent by the mobile agent to the group was tagged with an identifier and creation timestamp to allow the calculation of the group distance from the individual end-to-end delay observations. The average group distances that were observed during the simulations are depicted in Figure 1. The group distance tend to decrease with increasing values of the maximum migration rate.

The average number of migrations that occurred for a chosen value of τ_m^{-1} is shown in Figure 2. As expected, it was observed a clear increase in the number of migrations with larger values of the maximum migration rate. It was harder for the algorithm to find suitable migrations with larger groups as evidenced by the higher number of migrations observed for those groups. On the other hand, there was a small decrease in message throughput with larger migrations rates as a consequence of the migrations cost Figure 3.

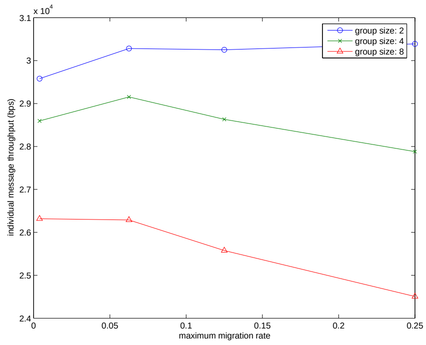

Figure 3: Message throughput (bps) versus τ_m.

4 CONCLUSIONS

As the volume of business traffic on the Internet continues to increase with a shift from centralised to distributed architectures, better solutions are needed to handle group communications under uneven network conditions. The paper has explored the use of a two-tier system to support QoS-critical group communications that often arise in distributed systems.

The use of a CPN-based mesh offers self-aware communications with adaptive routes that change over time and that attempt to take advantage of resources that may become temporarily available in the network. The paper suggested a way to exploit the paths discovered by CPN to determine appropriate migration hosts for mobile agents. The approach considered the progressive improvement of the location of mobile agents by making selective migrations, which are calculated from the focalised network information collected by CPN or comparable self-aware network.

Finally, a comprehensive simulation study showed the advantages of this approach in the context of improving a critical group communication.

ACKNOWLEDGEMENTS

The work presented in this paper was partially supported by the project CASCADAS (IST-027807) funded by the FET Program of the European Commission.

REFERENCES

Gelenbe, E., Lent, R., and Xu, Z. (2001). Measurement and performance of cognitive packet networks. *J. Computer Networks*, 37:691–701.

Gelenbe, E., Lent, R., and Xu, Z. (2002). Performance and measurements of cognitive packet networks. *Computer Networks*.

Gonzalez-Valenzuela, S. and Vuong, S. T. (2002). Evaluation of migration strategies for mobile agents in network routing. In *MATA '02: Proceedings of the 4th International Workshop on Mobile Agents for Telecommunication Applications*, pages 141–150, London, UK. Springer-Verlag.

Gray, R. S. (1995). Agent Tcl: A transportable agent system. In *Proceedings of the CIKM Workshop on Intelligent Information Agents, Fourth International Conference on Information and Knowledge Management (CIKM 95)*, Baltimore, Maryland.

Gray, R. S., Cybenko, G., Kotz, D., Peterson, R. A., and Rus, D. (2002). D'Agents: Applications and performance of a mobile-agent system. *Software— Practice and Experience*, 32(6):543–573.

http://www.JumpingBeans.com, O. Jumping beans.

http://www.recursionsw.com/, O. Voyager.

Lange, D. B. and Oshima, M. (1999). Seven good reasons for mobile agents. *Commun. ACM*, 42(3):88–89.

Lent, R. (2008). The integrated network-environment simulator (ines), online: *http://san.ee.ic.ac.uk/ines*.

Peine, H. (2002). Application and programming experience with the ara mobile agent system. *Softw. Pract. Exper.*, 32(6):515–541.

Ranganathan, M., Acharya, A., Sharma, S. D., and Saltz, J. (1999). Network-aware mobile programs. pages 567–581.

Sudmann, N. P. (1996). *TACOMA - fundamental abstractions supporting agent computing in a distributed environment*. PhD thesis.

Tai, H. and Kosaka, K. (1999). The aglets project. *Commun. ACM*, 42(3):100–101.

Wong, D., Paciorek, N., Walsh, T., DiCelie, J., Young, M., and Peet, B. (1997). Concordia: An infrastructure for collaborating mobile agents. In *MA '97: Proceedings of the First International Workshop on Mobile Agents*, pages 86–97, London, UK. Springer-Verlag.