GA-Based Parameter Tuning for Multi-Agent Systems

Search-Based Software Engineering

ABSTRACT

The problem of determining the desired type and number of mobile agents on a network is not trivial. In fact, this issue of agent population composition is critical for optimal performance of agent based software systems. Too many agents will congest and slow down the network, whereas too few will create processing bottlenecks. The problem of agent population management is exacerbated on wireless ad hoc networks where host mobility can result in significant changes in the network size and topology. Due to the dynamic nature of wireless networks, it is impractical to prescribe a static, precomputed solution. This paper proposes to utilize genetic algorithm (GA) to search for a satisfactory agent population. The current state of a real ad hoc network defines the parameters for the network simulator. The solution developed in simulation is then deployed over the real network.

1. INTRODUCTION

In an agent based system, a number of mobile agents work together to achieve a desired goal. The efficiency of these agents and the validity of their result depends on the number of agents deployed into the system. If too few agents are deployed, desired parallelism will not be achieved and as a result the system's efficiency will lessen. On the other hand, if too many agents are deployed, the system can become burdened with unnecessary overhead, and may cause significant delays. The task of finding the optimal number of agents required to achieve the desired effect is difficult and problem-specific. This is especially true if a multi-agent system (MAS) is deployed across a mobile ad hoc network (MANET). Due to the dynamic nature of MANETs the optimal agent population is likely to change as the topology and other properties of the MANET change.

In this paper, GA is proposed for on-line multi-parameter optimization on the agent population. A network simulator is used as the evaluation function, simulating the current state of the network and networked applications. When an agent population with acceptable performance is discovered, it is released into the real system.

The rest of this paper is organized as follows: Section 2 reviews related research topics (e.g., wireless ad hoc networks, multi-agent systems, and population control and stability). Section 3 describes the problem, and introduces the suggested solution. Section 4 discusses the experimental results and evaluates the approach.

2. RELATED WORK

2.1 Mobile Ad Hoc Networks

This approach assumes that an agent system is being implemented on a highly dynamic network. In a wireless network with both mobile physical hosts and mobile agents, the network topology may change with time giving the possibility for some route redundancy. Many efforts explore the complexities of wireless and mobile computing [8, 18, 5], few of these have involved mobile agents. The continuously changing nature of a wireless ad hoc network requires the consideration of dynamic state information [15]. The practice of using the current state of the network to determine the behavior of an agent is somewhat similar to current work in active networks [1, 12, 7, 14, 17].

2.2 Multi-Agent Systems

Each agent is an independent entity and each has the ability to collaborate with others to solve some common problem. The method presented in this paper assumes that the agent system engages in some form of collaborative data dissemination or information gathering task. Models of agent collaboration can be found in [4] and [2]. Under that condition, the agent replication techniques are often used to improve the performance and robustness of the system. This direction was taken by [6], [9], [11] and several other researches. This research focuses on the methodology of the agent/service replication, while others concentrate on mechanisms of maintaining the desired agent population [13]. In both cases, the authors acknowledge that system designer should decide up-front on which agents to replicate and how many copies to make. This paper proposes a technique to address that specific question.

2.3 GA for Parameter Tuning

GAs have been routinely used in conjunction with simulators to perform on-line optimization. One such example is the work of Wegener [19], where GA was used to optimize the vehicle parking process. Another example more related to

our work deals with optimizing a water distribution network by means of complex evolution [10].

3. PROBLEM FORMULATION

3.1 Motivation

A MANET is a challenging environment for software system designers due to its dynamism and unpredictable nature. Network links can go up and down and nodes can enter and leave the network depending on a variety of physical factors, such as movement of hosts, terrain, weather, interference, or available battery power. Agent based systems, with their runtime flexibility, can adapt to such an environment better than centralized systems [16].

On the other hand the tuning and control of the agent based system is more complicated, exactly due to the flexible and decentralized nature of the MAS. Since it is unlikely that the optimal agent population composition can be derived theoretically, a search based technique should be used to find acceptable suboptimal solutions rather then guaranteed optimal ones.

Many references to the example of information dissemination or collection by agent based system will be used throughout this paper. It is easy to envision a distributed system of mobile wireless devices used by the first responders: rushing to the cite of the natural disaster, a police unit in the crowd control situation or military urban operation unit where the information about the individual units should be collected and disseminated to the some subset of the team members [3]. Such information can include but is not limited to the status of the networked device (i.e., remaining battery life, CPU load and memory usage), sensory information or even biometrics of the users if device equipped with appropriate sensors. If units have GPS capabilities, then the problem of disseminating that information becomes so called "blue force tracking problem" i.e., problem of tracking friendly forces location on the battlefield. It is easy to see that optimal performance of such systems is of critical importance to it's users.

3.2 Proposed Approach

In order to address the on-line software configuration problem of that size, a GA based search with evaluations performed by an on-board network simulator was used. The management computing device reads in the current network topology (simulated at the current research stage) and forwards the data to the Network Simulator. GA performs the optimization of the agent population size and composition based on the fitness evaluated by the simulation runs. Each evaluation of the GA is the run of the simulator. Whenever desired levels of the parameter under optimization was achieved the population of agents is released onto the actual network. The Agent Manager block is in charge of the removing the old agent population from the network as well as introducing a new one. The overall design of the system is shown on the Figure 1.

In order to simulate normal network traffic, several steps were taken. Each link was given amount of available bandwidth and weight, and all pairs shortest path (Floyd-Warshall) algorithm was used to determine the routes on the simulated network. A typical load level was determined for each

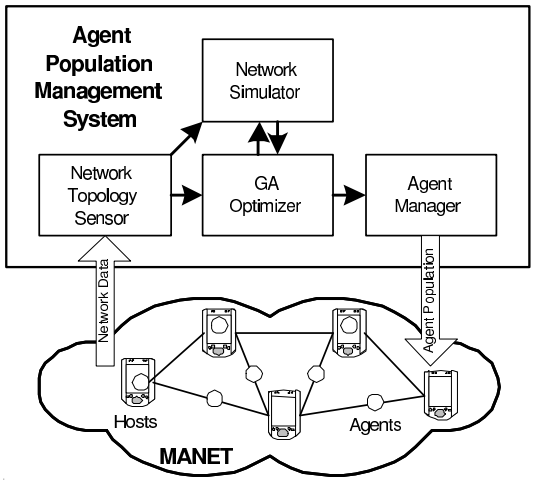

Figure 1: System Architecture

link based on the assumptions of shortest path routing and constant communication flow between the hosts. This level defined the amount of bandwidth left available on each link. We typically set the amount of bandwidth available to 20% on the most loaded link, and the rest of the links having a bandwidth usage inverse proportional to the number of communicating host pairs.

4. EMPIRICAL VALIDATION

In order to validate the proposed approach we considered the following information dissemination problem. Each host on the network is a source and sink for a specific kind of data. For example, each host independently obtains it's own coordinates via built in GPS card. It also needs to alert all other hosts about the updates in it position. In order to achieve that goal the swarm of randomly wandering agents can be used. That type of update, although non-deterministic, is rather scalable, robust to unexpected event and partial failures and completely decentralized [13].

Even more, it is reasonable to assume that each agent could hold information about a certain number of nodes (memory length), up to as many nodes as in the network. The more nodes an agent could remember, the larger its size, and thus the longer it takes to process and transmit this agent. Presumably, if there was a network of extremely large size, having only agents with a memory of that size would end up slowing down rather then improve the communication over the network. On the other hand, if agents with a memory of one were used exclusively, only pairs of nodes with a direct connection would receive information about one another. As we show later for some network topologies it is more efficient to use smaller, faster mobile agents with less complete record of the state of the world.

Evaluation.

In order to evaluate the process of the information dissemination by agents we created the measure of how up-to-date the information on each host is. Each host utilized a memory status list that contained the time of the last received update from every other host. The longest time without an update was considered to be the measure of how out of date

that host was. Average of these numbers over the set of all hosts was considered the measure of how out of date the whole system was:

$$\frac{\sum_{v \in H}(\max_{u \in H}^{v}(t_{upd}))}{|H|}$$

It was usually necessary to run the simulation for a short period of time before that number would stabilize.

Simulation

In order to run the simulations needed to evaluate the effectiveness of each agent population, this project used Lockheed Martin Advanced Technology Laboratories' CSIM [**?**]. CSIM is a discrete-event simulator for block diagram oriented systems. CSIM can be used for modeling wireless networks, computer networks, multi-processor systems, distributed systems, and other systems where the discrete event approach is applicable. The site for CSIM, where an overview of CSIM and a list of applications CSIM is used for, can be found on-line.

4.1 GA Setup

A population of twenty was used for each experiment. Agent population composition encoded as chromosome with allele corresponding to the number of hosts agent can store data for i.e., memory size, while the value of gene is the number of such agents in the population. Initial population was created at random. A uniform crossover was used. Mutation was implemented as follows: with uniform probability of one percent each gene was chosen to be mutated. If selected for mutation the gene value would be increased or decreased by one agent or by ten percent (what ever is larger). If the gene value was zero, it was mutated to the value of one.

Simulations were run to evaluate the population. Each simulation starts by triggering a thread for every agent in a given population. Agent threads last for the entire duration of the simulation i.e., no agents were terminated nor started during the simulation run. All agents were programmed to perform a random walk. Also each agent have a hash of host data updates of the size defined by the agent type. Every data update was time-stamped at the moment agent collected the data from the host. While agent preforms it's random walk across the network, its processing time is determined by its size, as well as the sizes and number of other agents currently on that host. While an agent is being processed, the host records the information provided by the agent about other nodes. If the agent has more up-to-date information about some other node, the host node overwrites its current information about that node with the information delivered by agent. After being processed, an agent is sent to its next randomly chosen destination. The transmission time is determined in a fashion similar to the process time. When an agent is in the process of being sent across a link, the amount of available bandwidth on the link decreases in accordance with the agent's size. If an agent is attempting to traverse a link and there is no available bandwidth, the agent is queued and attempts a retransmission every two CSIM seconds until it is successfully sent.

4.2 Experiment Data

Experiment 1 (Random Network)	
Network Size	10 nodes
Network Type	Random
Simulation Length	1500
Search Space	0-700
Initial Chromosome's Latency	750.656
Best Chromosome's Latency	311.151

Table 1: Setup and Latency

Optimal Chromosomes		
Memory Size	Agent Amount	
	Initial	Final
1	25	26
2	42	17
3	45	3
4	13	12
5	48	55
6	29	23
7	11	11
8	62	42
9	0	6
10	65	41

Table 2: Initial and Final Optimal Chromosomes

Currently experiments were performed on the generated topology with the intent to actively discover the current network configuration and import it into the simulator in the future.

In the first experiment, a randomly generated topology was used. This was done by adding one node at a time until every node had been added. Each time a node was added, it was assigned a random number of neighbors ranging from one to the current number of other nodes on the network at that time. The actual neighbor assignment was also done randomly.

4.3 System Behavior Over Time

The majority of the chromosomes in the initial population received the default worst fitness possible due to the fact

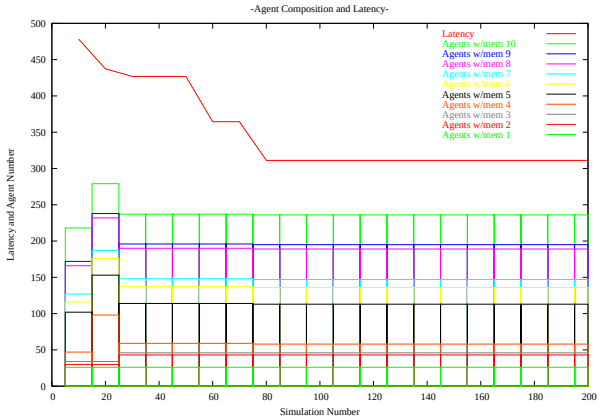

Figure 2: Experiment 1

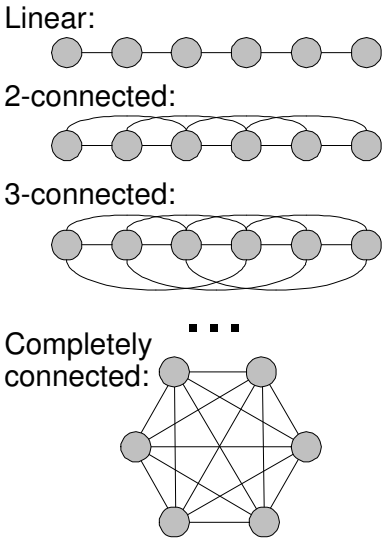

Figure 3: Spectrum of graph topologies

that the agent population of given composition failed to update some network hosts. With each generation, the number of good chromosomes increased until the chromosomes started converging to a particular composition. Eventually a new chromosome would be found that was better than the best of the last population, but each time it took longer for this to happen. The experiment was stopped after 200 generations, being how the fitness of the best chromosome would not improve enough to make the time spent simulating worth it.

4.4 A Topology's Effect on System Behavior

An idealized spectrum of network topologies, from linear to completely connected, corresponding to people moving in single-file formation, was used as the underlying topology for the experiments. In each subsequent experiment, the connectivity of the graph increased by one, i.e a connected linear graph is turned to 2-connected, 3-connected, ... , $(n-1)$-connected and completely connected graph, as shown in Figure 4.

As the connectivity of the network increased, there was a significant change in the optimal agent population composition. In the first network, made up of a single-file of nodes, the optimal agent population composition was made up of only two agents of the largest memory size permissible. The very small agent population was due to the fact that there was only one possible path between every pair of nodes. An increase in agents just meant possible queuing. The reason that there were two of this particular type of node was because no other agent would really be useful enough. An agent of any other type would never be able to carry information about a node from one end of the network to the other end. With each increase of connectivity, the use of agents with smaller memory sizes became more and more frequent. This was attributed to the fact that these agents could be transmitted and processed faster, and could also provide information to more nodes as the number of paths between nodes increases.

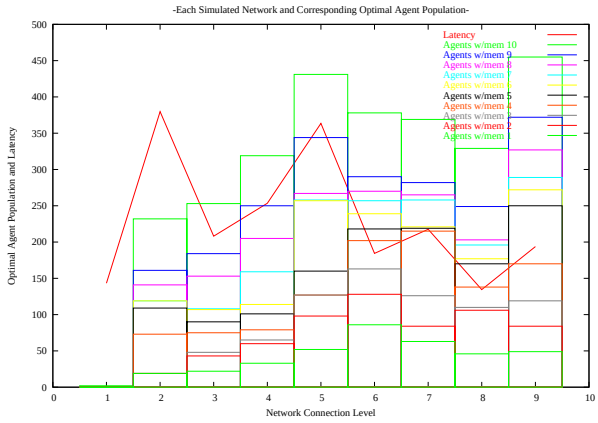

Figure 4: Optimal Populations For Each Network

5. FUTURE WORK AND CONCLUSIONS

5.1 Future Work

The next most important step in our research is to integrate the proposed approach with the fielded MAS deployed over the MANET. We are faced with some integration challenges but it is clearly implementable system. Another improvement necessary to the current state of the research is to investigate the scalability of the proposed approach. Performing evaluations using a simulator can get rather costly for the networks of large size. In order to mitigate this potential problem we are planning to investigate the operations that will allow us to achieve good optimization with fewer evaluations.

5.2 Conclusions

In this paper we presented a search based approach to the problem of software configuration system as it applies to MAS. We empirically validated it's correctness and applicability to the domain of mobile ad hoc networks and also showed some useful correlations between the network topology and the composition of the agent population performing the data dissemination tasks.

6. REFERENCES

[1] D. S. Alexander, W. A. Arbaugh, A. D. Keromytis, and J. M. Smith. Security in active networks. In *Secure Internet Programming: Issues in Distributed and Mobile Object Systems*. Springer, 1999.

[2] F. Brazier, P. van Eck, and J. Treur. Modelling competitive co-operation of agents in a compositional multi-agent framework. *Int'l J. Coop. Info. Sys.*, 6:67–96, 1997.

[3] V. A. Cicirello, M. Peysakhov, G. Anderson, G. Naik, K. Tsang, W. C. Regli, and M. Kam. Designing dependable agent systems for mobile wireless networks. *IEEE Intelligent Systems*, 19(5):23–29, September/October 2004. Special Issue on Dependable Agent Systems.

[4] J. E. Doran, S. Franklin, N. R. Jennings, and T. J. Norman. On cooperation in multi-agent systems. *Knowl. Eng. Rev.*, 12(3):309–314, 1997.

[5] G. H. Forman and J. Zahorjan. The challenges of mobile computing. *IEEE Computer*, 27(4):38–47, April 1994.

[6] F. C. Gartner. Fundamentals of fault-tolerant distributed computing in asynchronous environments. *ACM Computing Surveys*, 31(1):1–26, 1999.

[7] M. Hicks and A. D. Keromytis. A Secure PLAN. In *Int'l Working Conf. on Active Networks*, volume 1653, pages 307–314. Springer, 1999.

[8] J. Ioannidis, D. Duchamp, and G. Q. M. Jr. IP-based protocols for mobile internetworking. In *ACM SIGCOMM*, pages 235–243, 1991.

[9] S. Kumar, P. R. Cohen, and H. J. Levesque. The adaptive agent architecture: Achieving fault-tolerance using persistent broker teams. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS 2000)*, pages 159–166, July 2000.

[10] S.-Y. Liong and M. Atiquzzaman. Optimal design of water distribution network using shuffled complex evolution. *Journal of The Institution of Engineers*, 44:93–107, 2004.

[11] O. Marin, P. Sens, J. Briot, and Z. Guessoum. Towards adaptive fault tolerance for distributed multi-agent systems. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 2*, pages 737 – 744, 2002.

[12] S. Murphy, E. Lewis, R. Puga, R. Watson, and R. Yee. Strong security for active networks. In *IEEE Conf. on Open Arch. and Network Programming*, 2001.

[13] M. Peysakhov, V. A. Cicirello, and W. Regli. Ecology based decentralized agent management system. In *Proceedings of Formal Approaches to Agent-Based Systems III*, April 2004.

[14] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan. Detour: a case for informed Internet routing and transport. *IEEE Micro.*, 19(1):50–59, 1999.

[15] B. N. Schilit. *A System Architecture for Context-Aware Mobile Computing*. PhD thesis, Columbia, 1995.

[16] E. Sultanik, D. Artz, G. Anderson, M. Kam, W. Regli, M. Peysakhov, J. Sevy, N. Belov, N. Morizio, and A. Mroczkowski. Secure mobile agents on ad hoc wireless networks. In *The Fifteenth Innovative Applications of Artificial Intelligence Conference*, pages 129–36. American Association for Artificial Intelligence, Aug 2003.

[17] D. L. Tennenhouse and D. J. Wetherall. Towards an active network architecture. *Comp. Comm. Rev.*, 26(2), 1996.

[18] T. Watson. Application design for wireless computing. In *Wkshp on Mobile Comp. Sys. and Ap.*, 1994.

[19] J. Wegener and O. Bühler. Evaluation of different fitness functions for the evolutionary testing of an autonomous parking system. In *Proceedings of GECCO*, volume 2, pages 1400–1412, 2004.