行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※　　　　　網路代理程式演化計算　　　　　　　　　※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：■個別型計畫　　□整合型計畫

計畫編號：NSC 89－~~2313~~ 2213－E－032－004

執行期間：　88 年　08 月　01 日至　89 年　07 月　31 日

計畫主持人：葛煥昭 副教授

共同主持人：施國琛 教授

本成果報告包括以下應繳交之附件：

□赴國外出差或研習心得報告一份

□赴大陸地區出差或研習心得報告一份

□出席國際學術會議心得報告及發表之論文各一份

□國際合作研究計畫國外研究報告書一份

執行單位：

中　華　民　國　　89 年　　09 月　　26 日

行政院國家科學委員會專題研究計畫成果報告

國科會專題研究計畫成果報告撰寫格式說明

Preparation of NSC Project Reports

計畫編號：NSC 89-2213-E-032-004

執行期限：88 年 08 月 01 日至 89 年 07 月 31 日

主持人：　葛煥昭　淡江大學資訊工程學系

共同主持人：施國琛　淡江大學資訊工程學系

計畫參與人員：鄧有光　淡江大學資訊工程學系

林坤億　淡江大學資訊工程學系

陳俞先　淡江大學資訊工程學系

一、中文摘要

除了類神經網絡與基因遺傳演算法以外，生態系統的演化裡面還蘊含著其它重要的自然法則，食物鏈的概念便是我們可以從自然界裡學習到的另一種法則，此概念影射了計算理論中所謂的系統負載平衡問題。在此，我們提出了在網路上利用食物鏈的概念所建立的代理程式演化計算模式，同時，我們也提出了一些演算法來幫助模擬網路上多媒體搜尋代理程式的演化過程。這個技術的好處在於為達到分散式資訊擷取的情況下允許計算負載從本地端移到伺服端而平衡整個系統的負載水平，並且有效降低網路頻寬的需求。

關鍵詞：行動網路架構，行動代理人

Abstract

The ecosystem is an evolutionary result of natural laws. Food Web (or Food Chain) embeds a set of computation rules of natural balance. Based one the concepts of Food Web, one of the laws that we may learn from the natural besides neural networks and genetic algorithms, we propose a theoretical computation mobile agent evolution on the Internet. We define an agent niche overlap graph and the agent evolution states. We also propose a set of algorithms, which is used in our multimedia search programs, to simulate agent evolution. Agents are cloned to live on a remote host station based on three different strategies: the brute force strategy, the semi-brute force strategy, and the selective strategy. Evolutions of different strategy are discussed. Guidelines of writing mobile agent programs are proposed. The technique can be used in distributed information retrieval which allows the computation load to be added to servers, but significantly reduces the traffic of network communication. In the literature of software agents, it is hard to find other similar models. The results of this research only address a small portion of the ice field. We hope that, this problem should be further studies in the societies of network communication, multimedia information retrieval, and intelligent systems on the Internet.

Keywords: Mobile Network Architectures, Mobile Agent

2. Introduction

Communication over Internet is growing increasingly and will have profound implications for our economy, culture and society. Form mainframe-based numerical computing to decentralized downsizing, PCs and workstation computers connected by Internet have become the trend of the next generation computers. With the growing popularity of World Wide Web, digital libraries over Internet plays an important role in the academic, the business, and the industrial worlds. In order to allow effective and efficient information retrieval, many search engines were developed. However, due to the limitation of now-a-day network communication bandwidth, researchers suggest that distributed Internet search mechanisms should overcome the traditional information retrieval technologies, which perform the controls of searching and data transmission on a single machine. Mobile agents are software programs that can travel over the Internet. Mobile search agents find the information specified by its original query user on a specific station, and send back search results to the user. Only queries and result are transmitted over the Internet. Thus, unnecessary transmission is avoided. In other words, mobile agent computing distributes computation loads among networked stations and reduces network traffic.

3 Simulation

It is difficult to build our agent evolution computing environment or any agent systems form the scratch. Therefore, we choose JATLite as the supporting system for our simulation implementation. Another reason for choosing JATLite is that agent written in Java. Which can be achieve using JATLite, can run on different computer environments, since Java virtual machine is widely available and Java applet agents can be run from most Web browsers.

JATLite is a software library developed at Stanford University for building software agents, which communicate over the Internet. Developed in the Java language. JATLite used KQML (Knowledge, Query and Manipulation Language) as the agent communication language. KQML is well-developed and adapted to the agent research community. JATLite has five layers: the abstract layer, the base layer, the KQML layer, the router layer, and the protocol layer. The lowest layer is the abstract layer, which uses TCP/IP protocol. The base layer handles basic and store KQML messages. The router layer is charge of sending and receiving agent messages, as well as agent name registration. The protocol layer supports some standard Internet services, such as FTP, SMTP, HTTP, and POP3. Since our agent evolution model does not rely on any specific low communication protocol, most of the functions we used are in the router layer and the protocol layer.

The design of our agent simulation environment is illustrated in figure 4. Based on JATLite functions, it is convenient to develop agents, which pass messaged to each other. However, agent cloning is a technique problem. When an agent program (e.g. a search agent) is copied to a remote host station, the agent need to be "alive" on that station. However, the station where the original agent program lives can not initiate a process on a remote station. Therefore, in the simulation environment, we have docking agents to serve this purpose. A docking agent is an agent daemon run on a host station. The daemon is not part of agent evolution computing, but is designed to support the computing. A docking agent is constructed in JATLite, with the ability to launch another agent program (e.g., a search agent). Search agents are used as an example in our simulation. In our tested, we also consider other types of agents. For instance, agents intensively use network resources, such as advertising agents, which deliver multimedia information.

The agent cloning process includes a number of steps. Firstly, when a search agent is about to clone itself, the search agent send a query to a commercial search engine. The commercial search engine responses with a set of URLs. Based on these URLs, the search agent makes s copy of itself in a remote station through the help of the docking agent on that remote station (i.e., via sending a clone message). After that, the docking agent may start a new process to run the search program. Note that, a docking agent is installed in each host station of our simulation environment. One of the reasons of using docking agents is that when an agent program is copied, the docking agent can have a write access to the station where the agent program is duplicated.

Docking agents send host station messages to search agents. Thus, a docking agent may kill a search agent if necessary. Search agents may send agent messages to each other. Search results are send back to the agent who initiates the search goal via query result messages. Each host station may hold some search agents. But, holding a message router of a host station is optional. Message routers (or agent message routers) are programs provided in JATLite tool. Their responsibility includes allowing agent name registrations, maintaining agent message queues, and sending/receiving agent messages. A message router in our simulation environment handles the following types of messages:

- Agent Messages,
- Query Result Messages,
- Host Station Messages, and
- Clone Messages

  Since a message router should be an efficient to deliver messages, it is recommended to install a message router in station where an articulation search agent lives. Articulation search agents are connected to other search agents of the same societies. Therefore, they suggest efficient routes of communication. A message router, besides holding agent registration information and an message queue, also hold a repository which contain useful information such as network status.

3.1 Evaluation of the Simulation

Different agent distribution mechanisms have different impacts to the sizes of the agent societies and the overall performance. In the brute force agent distribution, agent societies are growing due to the duplication of agents, which are launched by the same user, as well as different users. Therefore, multiple copies of agents of the same priority exist in the ACN. The complexity of agent cloning algorithm is lower. But, network traffic as well as the size of agent societies are increased. In general, this cloning strategy has a bad performance. In semi-brute force agent distribution, agent societies increase their sizes due to different users launch the same goal of different priority. But agents from the same user will not increase their sizes upon the demand. Even the complexity of the agent-cloning algorithm is higher. But the amount of agents and thus network traffic are reduced. In general, this strategy achieves the best performance.

4. Conclusions

Mobile agent based software engineering is interesting. However, in the literature, we did not find any other similar theoretical approach to model what mobile agents should act on the Internet, especially how mobile agents can cooperate and compete. A theoretical computation model for agent evolution was proposed in this paper. Algorithm for the realization of our model were given, Consequently, our contribution in this paper are:

- We proposed a model for agent evolution computing based on food web, the law of natural balancing.
- We proposed s set of algorithm for the distributed computing of agent

programs.

- We implemented a simulation environment based on JATLite to support our theory.

However, there are other extensions to the evolution model. For instance, species in the nature world learn form their enemies. In our future model, agents can learn from each other. We can add a new state, the "learning" state, to the agent evolution states. When an agent is in the dangling state, it can communicate to other agents via some agent communication languages. Computing methods can be replicated from other agents. And the agent transits to the mutating state to wait for another new goal. In addition, when a station lacks of system resource, an agent in the suspending state can change its policy to admit to the environment before it transits to the searching state. These are the facts that agents can learn. On the other hand, in the cloning process, two agents on a station sharing a common goal can be composed to a new agent (i.e., marriage of agents). This agent states. But, the destination station where this new agent lives should be compromised.

The evolution of computers has changed form mainframe-based numerical computation to networked stations. In line with the success of Internet technologies, in the future, computation and information storage is not limited to a single machine. It is possible that, an individual buy a primitive computer that only has a terminal connected to Internet. Person data and the computation power are embedded within the Internet. Mobile agent and agent evolution computing will be very interesting and important. Our agent evolution model address only a small portion of the ice field, which should be further studied in the societies of network communication, automatic information retrieval, and intelligent system.

5. Reference

[1] P. Charlton, Y. Chen, E. Mamdani, O.Olsson, J. Pitt, F. Somers, and A. Wearn, "An Open Agent Architecture for Integrating Multimedia Services," in Proceedings of the Autonomous Agents 97 conference, Marina Del Rey, California, U.S.A., 1997, pp. 522-523.

[2] Chanda Dharap and Martin Freeman, "Information Agents for Automated Browsing," in Proceedings of the 1996 ACM CIKM conference (CIKM'96), Rockville, MD, U.S.A., 1996, pp. 296-305.

[3] Michael R. Genesereth, "Software Agents," communication of the ACM, Vol. 37, No. 7, Jul. 1994, pp. 48-54.

[4] Robert S. Gray, "Agent Tcl," Dr. Dobb's Journal of Software Tools for Professional Programmer, Vol. 22, No. 3, Mar 1997.

[5] S. Krause and T. Magedanz, "Mobile service agents enabling intelligence on demand in telecommunications," in Proceedings of the 1996 IEEE Global Telecommunications Conference, London, UK, 1996, pp. 78-84.

[6] Jide B. Odubiyi, David J. Kocur, Stuart M. Weinstein, Nagi Wakim, Sadanand Srivastava, Chris Gokey, and JoAnna Graham, "SAIRE - A Scalable Agent-Based Information Retrieval Engine," in Proceedings of the Autonomous Agents 97 conference, Marina Del Rey, California, U.S.A., 1997. Pp. 292 - 299.

[7] Joseph Tardo and Luis Valente, "Mobile agent security and telescript," in Proceedings of the 1996 41st IEEE Computer Society International Conference (COMPCON'96), Santa Clara, CA, USA, 1996, pp. 58 - 63.