行政院國家科學委員會專題研究計畫成果報告

電子商務之效率控管及訂單管理研究

Performance Control and Purchase Order Management on Electronic Commerce

計畫編號：NSC 87-2213-E-002-102
執行期限：86 年 8 月 1 日至 87 年 7 月 31 日

主持人：顏嗣鈞 教授　　台大電機系
(E-Mail: yen@cc.ee.ntu.edu.tw)

一、中文摘要

現在網際網路上各公司的電子商業系統，如雨後春筍般的出現，但是各個公司網址分散各處，消費者瀏覽產品需要花費許多時間在搜尋比較上，且各公司的安全系統並不一致，不獨消費者的權益保障不夠，同時也造成了公司、銀行與顧客的不便。所以整合型的網際網路電子商業系統漸漸流行，已成為電子商業新趨勢，如同超級市場取代傳統商店一般。隨著使用者的快速增加，這個全球性的傳播媒介正在逐漸面臨速度太慢的瓶頸。以往網際網路各公司的電子商業系統較小，且較侷限於區域性，內部效率管理的問題較不受重視，一旦整合型的網際網路電子商業系統漸漸流行，隨著加入的公司、銀行和顧客增加，與服務的區域擴大，效率管理問題會日趨嚴重，是故我們應提早因應。

本計畫目的是要研究電子商務訂單管理中之資料存取問題。電子商務系統是一個典型線上交易系統，消費者的需求必須線上即時處理，而銀行、廠商也必須線上即時反應。近年來，由於電腦通信網路的發達，造成了網際網路的蓬勃發展，越來越多的人連上網路去尋找他們所需要的資料，急速地增加網路傳輸上的負載。因此，如何安排資料伺服器在網路上的位置，以降低網路傳輸的負載便成為一個重要的問題。在這研究中，我們模擬了資料在網路的傳輸，並假設資料存取是有區域性(locality)，我們比較了數種演算法，這些演算法將會決定資料在存取後在網路上擺放的位置，我們同時比較了這些演算法的效能並予以分析。

未來我們的研究將整合於電子商務的訂單管理實作系統內，並與整個電子商務結合測試，希望能對整體運作效率的有所改進。

關鍵詞:電子商務、網際網路、資料存取、模擬分析。

Abstract

Electronic commerce on the Internet is a future trend of the daily transaction. More and more electronic commerce systems on the Internet are proposed by corporations. Since the dispersion of sites of corporations, consumers spend much of time to search different sites for comparing the products. Furthermore, the security systems of different corporations are inconsistent. These cause the inconvenience of all the consumers, merchants, and banks. So, integrated electronic commerce systems on the Internet are more and more popular, as supermarkets and malls replace the traditional shops.

In recently years, the Internet has become ubiquitous. The size of the network grows quickly and a tremendous amount of data flows on the Internet at any time during the day. Therefore, how to place data in the network to reduce the traffic load, which in

turn increases the utilization, is an important issue. We design several mechanisms to decide the best location at which data are stored in the network. To this end, we take advantage of many concepts and designs originated in, for example, the list update problem, the k-server problem and the paging problem. We also simulate these algorithms in order to analyze their performances. In our simulation, we study the behaviors of data accesses and movements among servers in a mesh network

Keywords: Electronic commerce, Internet, the server problem, simulation.

二、緣由與目的

In recently years, computer networks have become ubiquitous, and consequently, the research in this area is full of vitality. Million of people connect their computers by an immense network, namely the Internet. Using the Internet, people communicate with each other by sending e-mail or conducting in video-conferencing. Because more and more people are connected to the Internet. The size of the network grows quickly and a tremendous amount of data flows on the Internet at any time during the day. Therefore, how to place data in the network to reduce the traffic load which in turn increases the utilization is an important issue, which has a profound impact on the efficiency of the network.

There are many services provided by a variety of servers in the Internet. If one server provides popular data in the network and many users want to access it in particular when there is only one copy of the datum [12, 14], it is important to decide which server to store the data in order to reduce the total access cost. Such a server-related problem is going to be studied in depth in this thesis.

We design several mechanisms to decide the best location at which data are stored in the network [3, 4, 6, 10, 13]. To this end, we take advantage of many concepts and designs originated in, for example, the list update problem, the k-server problem and the paging problem. Many algorithms for these problems have been proposed in the literature [2, 3, 4, 6, 8, 10, 11]. Among them are the *Move to Front*, *Transpose*, *Move to Middle*, *BIT*, *BIT* and *Move to Middle* heuristics, for example. Since the server problem in real networks is much more complicated than the simplified k-server, list update and paging problems, it is very difficult to solve the server problem analytically. As a result, resorting to simulation in the analysis of the " general" server problem seems to be inevitable.

In our simulation, we study the behaviors of data accesses and movements among servers in a mesh network. In comparison with ring and star topologies commonly seen in local area networks, meshes represent an architecture which is much more different to analyze. Hopefully, the experiences and results learned from studying meshes will allow us to better understand the more complicated behaviors encountered in general network.

三、結果與討論

We assume that our system topology is an n×m mesh and each crossing point in the mesh, as shown in Fig.1.

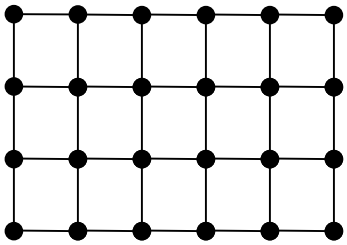

Fig.1 Mesh topology

There are two types of costs, namely **access cost** and **move cost**, in the cost model of our simulation. When server A wants server B to serve a request, server A first finds where the data is. After the access of the data is completed, one has to decide which server, server A or another server, is going to store the data. The cost involved in first step is mainly the access cost and the second step is concerned with the move

cost. Both access cost and move cost are defined by the distance between the two servers involved. The exchange cost has two kinds of modes: free exchange and paid exchange in the list update problem. The exchange cost in the free exchange mode is zero; others are paid exchanges. Such a classification, however, it is not appropriate in our network because data transmission between servers involves not only access cost but also move cost. So move cost is not free in network and we should not ignore it. But some move operations can be done when network flow is not heavy. For example, less frequently used data can be moved in the night or on holidays if they are accessed only one time. So we can define several kinds of mode to give different weights to access cost and move cost.

Server request sequences and **data request sequences** are two important factors in the simulation. A server request sequence indicates which server issues request in each time step. A data request sequence indicates which data is used in each time step. We give several situations that are of interest in practice. We have three kinds of sever request sequences:

(A). All servers have the same service frequency.

(B). Some servers are accessed more frequently than others.

(C). Accesses are carried out randomly.

The data request sequence is also an important factor in our simulation. It specifies the data to be accessed in each time step. We consider five types of data request sequences.

- All items in the data request sequence are the same.
- The data request sequence contains two kinds of items and both have the same weight.
- The data request sequence contains two kinds of items with different weight.
- Fixed amount of data occupies the major part of the access sequence.
- 80% accesses is selected from 20% data.

In the system we use five algorithms to simulate the one-server problem and three algorithms to simulate the two-server problem. In the two-server problem simulation, each datum has two copies in the network and any pair of data will not be in the same server. The five algorithms used for the one-server problem are M*ove to Front*, *Transpose*, *Move to Middle*, *Bit*, and *Bit and Move to Middle*. The algorithms for the two-server problem are *Balance*, *Greedy* and *harmonic*.

In our simulation, we give two different topologies for comparing their performances with respect to a variety of algorithms. Mode 1 is a 40×40 mesh and mode 2 is a 20×40 mesh, as shown in Fig. 11 and Fig. 12, respectively. Each mode contains some properties about the underlying network topology. Table 1 is the related parameters about two modes in the one-server simulation environment. Table 2 is the related parameters in the two-server simulation environment.

Both in the one-server and the two-server simulations, the width, the length, the number of servers, and the number of data types are all the same. They only differ in the number of data and the cache size. The two-server simulation gives each data two copies in the network and one-server mode only contains one copy. Therefore, all servers need double cache size in the two-server simulation. Furthermore, we initialize that the same data are set in different servers in the two-server simulation.

	Mode1	Mode2
Width	40	20
Length	40	40
Number of server	1600	800
Number of data	6400	6400
Data type	6400	6400
Cache size	4	8

Table 1 Parameters in the one-Server simulation.

	Mode1	Mode2
Width	40	20
Length	40	40
Number of server	1600	800
Number of data	12800	12800
Data type	6400	6400
Cache size	8	16

Table 2 Parameters in the two-server simulation.

Some of our simulation results for the two-server case are shown below.

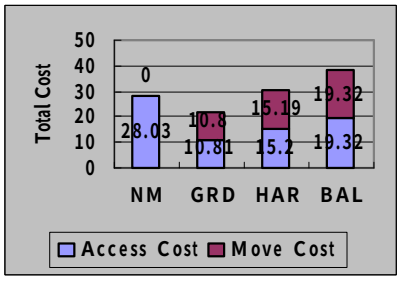

Fig. 2 Server generation rule A Data generation rule A

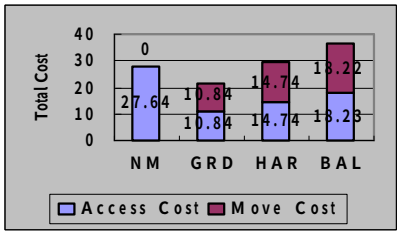

Fig. 3 Server generation rule A Data generation rule B

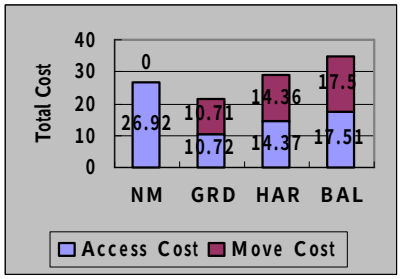

Fig.4 Server generation rule A Data generation rule C

Fig. 2 is the simulation results of rule A of the data request sequence generation rules. *Greedy* performs better than others in free exchange mode and paid exchange mode. It improves on *NM* 60.36% in free exchange mode and 18.22% in paid exchange mode. Fig. 3 and Fig. 4 have the same results with Fig. 2. In Fig. 29, *Greedy* improves on *NM* 60.2% in free exchange mode and 20.4% in paid exchange mode. In Fig. 4, it improves on *NM* 60.78% in free exchange mode and 21.56% in paid exchange mode improves. From our simulation result, if data access has higher degree of locality, *Greedy* can get more advantage.

四、計畫成果自評

In this research, we have introduced several algorithms to make decisions about where to put the data after accessing in a network for the purpose of decreasing future access cost. These algorithms come from three well-known on-line problems, namely the list update problem, the k-server problem and the paging problem, which are related to electronic commerce. We have also simulated these algorithms in order to analyze their performances.

Because the previous electronic commerce systems on the Internet are small and local, few systems consider the performance control issues. As the integrated electronic commerce systems on the Internet are popular, the consideration of performance control studied in this research will be helpful for enhancing the system performance.

五、參考文獻

1. A. R. Karlin, M. S. Mamasse, L. Rudolph and D. D. Sleator, Competitive snoopy caching, Algorithmica 3 (1988) 79-119.

2. A. R. Calderbank, E. G. Coffman, and L. Flatto, Sequencing problems in two-server systems,

Math. Oper. Res. 10, NO 4 (1985), 585-598.

3. J. H. Hester and D. S. Hirschberg, Self-organizing linear search, ACM Computing Surveys, Vol. 17, No.3 , Sep. (1985),295-311.

4. S. Irani, N. Reingold, J. Westbrook, and D. D. Sleator, Randomized competitive algorithms for the list update problem, Algorithmica 11 (1994) 15-32.

5. P. Raghavan and M. Snir, Memory versus randomization in on-line algorithms, Proc. ICALP'88, Lecture Notes in Computer Science, Vol. 372(1988) 687-703.

6. M. Chrobak and L. L. Larmore, HARMONIC is 3-competitive for two servers, Theoretical Computer Science 98 (1992) 339-346.

7. A. CALDERBANK, E. G. COFFMAN, and L. FLATTO, Sequencing problems on a sphere, Commun. Statist. –Stochastic Models 1, NO 1 (1985), 17-18.

8. R. RIVEST, On self-organizing sequential search heuristics. Commun. ACM 19, 2 (1976) ,63-67.

9. D. Knuth, A "self-organization" file, In The Art of Computer Programming, Memory versus randomization in on-line algorithm, Vol. 3 (1973), 398-399.

10. M. S. Manasse, L. A. McGeoch and D. D. Sleator, Competitive algorithms for server problems, Journal of Algorithms 11 (1990), 208-230.

11. M. Chrobak, H. Karloff, T. Payne and S. Vishwanathan, New results on server problems, Proc. 1st Annual ACM-SIAM Symp. on Disc. Algo. (1990).

12. M. Taylor, Building an architecture for multiservice networks, Tele-communications (International Edition) 30, 7, (1996).

13. D. W. Brubeck and L. A. Rowe, Hierarchical Storage Management in a Distributed VOD System, IEEE Multimedia 3, 3, (Fall, 1996) 37-47.

14. S. Venkataraman, M. Livny, and J. F. Naughton, Memory Management for Sca;able Web Data Servers, Proceedings of IEEE 12th International Conference on Data Engineering. (1997).

15. A. Silberschatz, P. Galvin, Operating system concepts, 4th ed., Addison-Wesley, (1994).

16. A. Tanenbaum, Computer networks, 3rd ed. ,Prentice Hall International, (1996).