行政院國家科學委員會專題研究計畫成果報告

網路資料庫之設計與實作

Design and Implementation of Network Database

計畫編號：NSC 87-2213-E-002-009
執行期限：86 年 7 月 1 日至 87 年 6 月 30 日
主持人：陳銘憲　　台灣大學電信中心

一、中文摘要

我們在此計畫中完成了一個網路資料庫系統。本系統可以透過網際網路瀏覽器對多個分散在網際網路上的資料庫做結合運算(join)操作。在資料庫的多重結合(multiple join)運算是高成本的運算元。在運算過程中，將會產生極為龐大的資料量。因此，在網路資料庫中執行多重結合運算，將會使得網路頻寬不足之問題更形嚴重。在此一計畫中，我們將針對減少多重結合運算資料量的問題加以深入研究。在我們的網路資料庫系統中，除了使用 Java 與 JDBC 的技術成功的整合了資料庫與瀏覽器外，也包含了一個詢問排程器（query scheduler）。此排程器採用了 semi-join 與 join sequence 的技巧來減少執行網路多重結合運算中所需的傳輸資料量。在此計畫報告中，我們將描述詢問排程器的排程機制並且將討論如何有效地在網路資料庫系統上實作。

關鍵詞：網路資料庫、詢問排程器、semi-join、join sequence

Abstract

In this project, we have finished a network database system. This system can execute join operations in distributed Internet databases via Web browsers. Since the cost of multi-join is high, multi-join operations involve a huge number of data. Therefore, the bandwidth in Internet is more insufficient. In our network database system, the database and browser are integrated successfully and this system exploits the technique of Java and JDBC. A query scheduler is also developed in this system. This scheduler adopts the mechanism of semi-join and join sequence to reduce the amount of data transmission required. We describe the mechanism of query scheduler and discuss how to implement this scheduler effectively in network database system.

Keywords: Network Database, Query Scheduler, semi-join, join sequence

二、緣由與目的

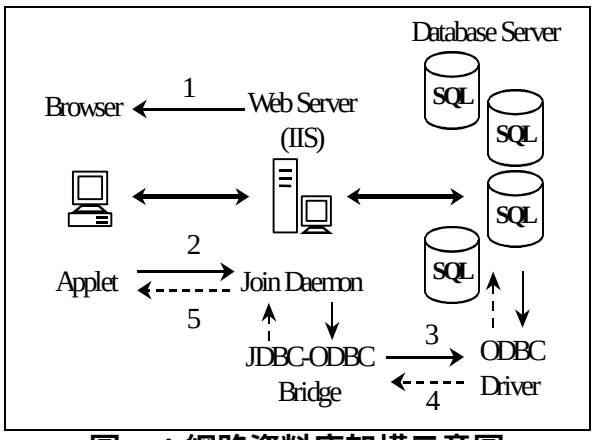
現今網際網路的快速發展，資訊界研發出許多網際網路上的工具和技術，包括瀏覽器之改進，伺服器功能之加強，與網路協定之修定等。 這些新的技術使得網路上的電腦主機在交換資訊時更為方便和多樣化。這些發展使得網路上的使用人口更是快速提升。以此觀之，網路實是個能以極低廉成本去達成高效率的訊息交換之場所。此外，現今的資料庫具有強大的資料管理的功能，使得目前的企業與政府機關均使用資料庫來幫助處理日常的工作。透過網際網路，使得遠端使用者方便存取與處理資料。因此，將網際網路的便利性與資料庫的強大處理能力結合，將是主要的趨勢。

然而，有部分的資料庫運算是屬於高成本的運算，多重結合運算即為其中之一。再執行多重結合運算的過程中，會產生極大的資料量。倘若所需結合的表格（relation）是分散在網路上不同的位置

（site）時，則需要大量的頻寬來傳送這些龐大的資料。這將使得已經壅塞的網路流量更加嚴重。

在此一計畫的主要目的在於整合網際網路瀏覽器與資料庫系統，完成一套網路資料庫系統。透過此系統，使用者使用瀏覽器經由方便的人機介面存取遠端資料庫的資料。除此之外，我們在此一研究計畫中，深入研究改善網路資料庫執行時所需頻寬的演算法。並且實作在本系統中。透過此演算法，網路資料庫將能夠更正確且更有效率的執行。

三、結果與討論



圖一：網路資料庫架構示意圖

我們所提出的網路資料庫系統的架構如圖一所示。在我們的系統中，我們使用的作業平台是 Windows NT 作業系統。使用的資料庫是 Microsoft SQL Server。所使用的 Web Server 是 IIS (Internet Information Server)。在這張示意圖中，遠端使用者利用瀏覽器來操作資料庫共有五個步驟。首先，使用者端的瀏覽器從 Web 伺服器下載網路資料庫系統的使用者介面。此介面是一個 Java applet 程式。其主要的工作有三：1.擷取使用者輸入的資料庫操作的 SQL 指令。2.與 Join Daemon 相連，並將指令傳到 Join Daemon 處理。3.輸出執行後的結果。

當 Join Daemon 接收到使用者的 SQL 指令後，將對指令做評估，並且依評估結果重新排序指令順序。Join Daemon 是一個 Java application 程式。因此，Join daemon 是透過 JDBC 介面來與資料庫做資料交換。然而，資料庫伺服器端與外部程式做資料交換是使用 ODBC 介面。因此，JDBC 介面必須再透過一個轉換介面—JDBC-ODBC Bridge—來與 ODBC 介面交換資料。當資料庫處理完 SQL 指令後，結果回傳給 Join Daemon。如果是最後的執行結果。Join Daemon 會將最後的結果回傳到使用者端的 applet 並且顯示出執行的結果。因此，透過這幾個步驟，使用者即能使用瀏覽器，方便針對資料庫的資料作存取的工作。

明確的說，我們所實作網路資料庫系統共有兩個重要元件。一是使用者端的使用者介面，其功能如上所述。二是伺服器端的 Join Daemon。Join Daemon 的功能除了接收使用者的 SQL 指令並將指令轉送到資料庫伺服器端執行外，還包含了一個重要的元件—Query Scheduler。如前所述，執行多重結合運算是一個高成本的運算。在執行過程中會產生極高的資料量。而增加網路傳輸的成本。因此，我們研發了 Query Scheduler algorithm 來降低傳輸成本。

Query Scheduler 的目的是找出最小傳出成本的 relation 結合執行順序。於是，我們使用兩種技術來降低所需傳送的資料量：semi-join 和 join sequence。在結合運算中，有部分的資料項沒有符合條件，而這些資料項不必在網路上傳輸。Semi-join 的方法可以過濾出這些資料項。消除這些資料項可以減少傳輸成本以及能減少執行的時間。

R1

I	P
i1	p1
i2	p2
i3	p2
i4	p4
i5	p4

R2

P	M
p1	m2
p2	m3
p3	m4
p5	m5

The result of join R1 and R2

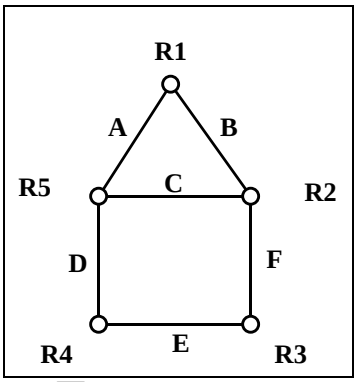
I	P	M
i1	p1	m2
i2	p2	m3
i3	p2	m3

圖二：結合運算的範例

在圖二中，R1 與 R2 分屬於不同的機器 S1 與 S2 中。要執行 R1 與 R2 的結合運算，最簡單的方式是把其中一個 relation 傳到另一台機器上執行結合運算。以此例為例，把 R2 傳到 S1 所需的傳輸資料項為 8。

若執行 semi-join 方式。R1 先將 P 做 projection，結果為{p1、p2、p4}。並把 projection 的結果傳到 S2。在 S2 中，R2 利用這個結果來過濾不需傳輸的資料項{p3、p6}。最後，R2 只需將{(p1, m2)、(p2, m3)}等四項資料回傳 S1 做結合運算即可得到正確結果。因此，在此例中，semi-join 只需傳送 3+4=7 項資料項。由此可知，semi-join 的方法可以有效的減少傳輸的資料量。

除了 semi-join 外，join sequence 也被使用來減少多重結合運算執行期間所產生的資料量。一個多重結合運算可以是用 query graph 來表示 relation 間的關係。每個端點即表示一個 relation。若端點與端點間有一條線相連，則表示這兩個 relation 存在一個結合運算。圖三為一個 query graph 的實例。

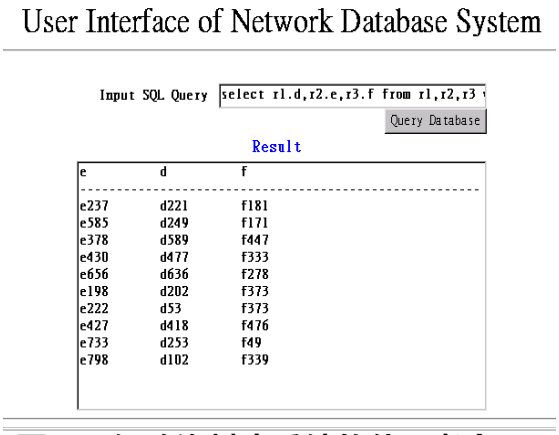


圖三：query graph

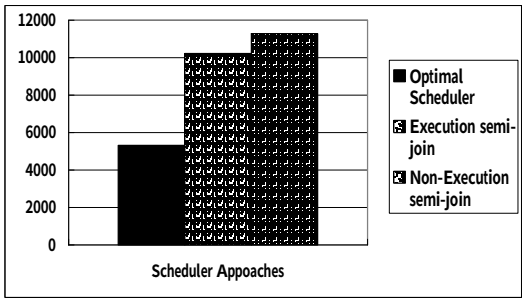
在一個多重結合運算中，可以找到不同的執行結合運算的順序。以圖三為例，假設 R3 為最後結果的存放處，我們可以找到 R1⇒R5、R5 ⇒R2、R2 ⇒R4、R2 ⇒R3 和 R1⇒R2、R2 ⇒R5、R5 ⇒R4、R4 ⇒R3 等執行順序。不同的執行順序所產生的結果會相同。但是，不同的順序，在執行期間所產生的資料量也不同。因此，利用多重結合運算的這種特性，可以減少網路的傳輸資料量。

在我們設計的 query scheduler algorithm 中，使用 semi-join 和 join sequence 兩種方法來降低網路資料傳輸量。當 Join daemon 接收到一個多重結合運算的指令時，query scheduler 會把它分解成子結合運算。並且會評估各種執行順序的傳輸量。在評估各個子結合運算的同時，也使用 semi-join 的方式更進一步的減少傳輸資料量。最後，query scheduler 會根據評估的結果找出最小傳輸資料量的執行順序。Join Daemon 會根據這個順序執行此多重結合運算指令。

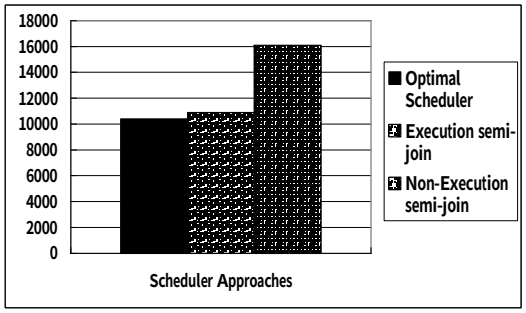
圖四是我們實作的網路資料庫系統的使用者介面外觀。此外，我們也對網路資料庫系統的效能作分析。使用亂數產生 3-relation 和 4-relation 結合運算的資料庫中資料項。並且分別執行三種不同的排程演算法。第一種是我們發展的 query scheduler。第二種是所有的子結合運算都執行 semi-join。第三種是所有的子結合運算都不執行 semi-join。圖五與圖六為執行後的結果。執行結果顯示出我們實作的網路資料庫系統的效能優於另外兩種排程。



圖四：網路資料庫系統的使用者介面



圖五：3-relation 結合運算執行結果



圖六：4-relation **結合運算執行結果**

四、計畫成果自評

在這個研究中，我們使用 JDBC **的技術成功的整合網際網路與資料庫系統。透過瀏覽器的介面可以很容易的操作資料庫系統。此外，由於網路資料庫的資料庫是分散在網路不同位址上。因此，對於網路頻寬之需求以及執行的效率等問題，我們也提出了解決方案。此一解決方案已經透過實作的方式加以驗證，並確認其可行性與正確性。我們確信這種方法可以有效的減少網路頻寬之需求以及降低執行的時間。使得網路資料庫系統能夠更有效率的執行。**

今日網際網路的相關技術演進十分迅速，新的工具或是新的技術將使網際網路與資料庫系統間的整合更為方便，也使網路資料庫系統的效能提升。由於此類網路資料庫的研究無論在實用上或學理上都具有其價值，我們將繼續利用新的技術對此類課題加以探討。

五、參考文獻

[1] P. A. Bernstein and D,-M. W Chiu, "Using semi-joins to solve relational queries," *Journal of ACM*, vol. 28, no. 1, pp. 25-40, Jan. 1981.

[2] P. A. Bernstein, N. Goodman, E. Wong, C. Reeve, and J. B. Rothnie, "Query processing in a system for distributed databases (SDD-1)," *ACM Trans. Database Syst.*, vol. 6, no.4, pp.602-625, Dec. 1981.

[3] P. A. Black and W. S. Luk, "A new heuristic for generating semi-join programs for distributed query processing," in *Proc. IEEE COMPSAC*, 1982, pp. 581-588.

[4] A. L. P. Chen and V. O. K. Li, "improvement algorithms for semijoin query processing programs in distributed database systems," *IEEE Trans. Comput.*, vol. C-33, no. 11, pp.959-967, Nov. 1984.

[5] M.-S. Chen and P. S. Yu, "Using combination of join and semijoins operations for distributed query processing," in *Proc. 10th Int. Conf. Distributed Comput. Syst.*, May 1990, pp. 328-335.

[6] M. S. Chen, "Using join operations as reduces in distributed query processing," in *Proc. 2nd Int. Symp. On Databases in Parallel and Distributed Systems*, July 1990, pp. 116-123.

[7] M. S. Chen, "*Interleaving a join sequence with semijoins in distributed query processing*," IEEE Trans. Parallel and Distributed Syst., vol. 3, no. 5, pp. 611-621, Sept. 1992.

[8] D.-M. Chiu, P. A. Bernstein, and Y.-C. Ho, "Optimizing chain queries in a distributed Database System," *SIAM J. Computing*, vol. 13, pp. 116-134, Feb. 1984.

[9] X.-M. Huang and M.-S. Chen, ``Design and Implementation for Join Operations in a Network Database," Proc. of Conf. on Electronic Imaging and Multimedia Systems, SPIE Photonics China '98 Symp., November 3-7, 1998.

[10] Y. Kambayashi, M. Yoshikawa, and S. Yajima, "Query processing for distributed databases using generalized semi-joins," in *ACM Proce. of SIGMOD*, 1982, pp. 151-160.

[11] H. Kang and N. Roussopoulos, "Combining joins and semijoins in distributed query processing," CS-TR-1794, Univ. Maryland, 1987.

[12] L. Lemay, C. L. Perkins, "Teach yourself JAVA in 21 days." Sams.net, 1996.

[13] P. Patel, K. Moss, "Database Programming with JDBC and JAVA." The Coriolis Group, Inc., 1997.

[14] G. Reese, "Database Programming with JDBC and Java." O'Reilly & Associates, Inc., June 1997.