行政院國家科學委員會專題研究計畫成果報告

高性能 I/O 架構與平行檔案系統設計之研究

The Design of High-Performance I/O and Parallel File System

計畫編號：NSC 87-2213-E009-053
執行期限：86 年 8 月 1 日至 87 年 7 月 31 日

主持人：張明峰 教授　　mgchang@csie.nctu.edu.tw
交通大學　資訊工程系

一、中文摘要

在計算能力與 I/O 速度差距很大的平行系統中，I/O 子系統成為電腦系統效能提昇的主要障礙。為了提高 I/O 子系統的效能，在平行檔案系統中，檔案以資料分割的方式分佈於多個資料伺服器上，使用者程式可以利用平行存取來提高 I/O 頻寬，增加 I/O 效能。為了解決平行環境下發展的程式移植性不佳、開發成本高的問題，四十幾個單位組織聯合制訂了 MPI(Message Passing Interface)標準。我們製作的 MPFS 平行檔案系統，就是架構在可移植性高的 MPI 環境上，根據 MPI-IO 規格製作使用者程式庫，提供程式設計者使用容易且高效能的存取函式。MPFS 除了有高度的移植性之外，系統並允許使用者彈性定義檔案的邏輯分割，指定檔案分佈方式與其他系統功能的設定，可以讓使用者依照應用程式的行為調整出最佳的系統設定。目前 MPFS 檔案系統的製作以工作站群組為實驗平台，以 100Mbits/sec Fast Ethernet 相連結，已經可以移植到 FreeBSD 及 SunOS 的作業系統上。

關鍵詞：MPI, 平行檔案系統，多執行緒

Abstract

The project developed a MPI (Message Passing Interface)-based parallel file system, MPFS. MPI-IO is an extension of MPI to support flexible logical file partition and physical file organization, as well as a rich set of file access functions. MPFS enables users to specify both logical file partition among user processes and physical file data layout across data servers. Present implementation includes full support of file data distribution, asynchronous I/O, shared file pointers, collective I/O operations, and limited support of data layout hints. Moreover, MPFS achieves portability by developing a user library that conforms a generic UNIX interface. However, MPFS does not support FORTRAN interface, error handling and profiling in MPI-IO specification. MPFS has been implemented on a workstation cluster connected by Fast Ethernet. The performance measurements of MPFS show that the message passing overhead is minimal but the performance is limited by the bandwidth of Fast Ethernet.

Keywords: MPI, parallel file system, multi-thread

二、緣由與目的

平行(parallel)系統的 I/O 在近幾年來引起愈來愈多的注意，因為 I/O 的效能(performance)已經成為限制整體系統效能的主要因素。I/O 會成為系統的主要瓶頸有幾個原因：當中央處理器(Central Processor Unit)的處理速度在過去幾十年來以驚人的速度成長的同時，I/O 的速度受到

機械裝置的限制，成長的速度相當的緩慢。舉例來說，處理器的運算速度每年以50-100%的速度成長，而磁碟的存取時間(access time)在十年內才減少三分之一[1]，這種趨勢在未來也不會改變。在平行及分散式系統中，結合了許多處理器同時運算，使得系統的運算速度和I/O的速度差距更大，而且平行應用程式不只需要快速的計算，也需要快速的I/O子系統來提供足夠的資料傳輸頻寬。再加上應用程式對I/O的要求愈來愈大，如影像處理、多媒體(multimedia)、電腦視覺(visualization)等，使得解決平行及分散式系統中的I/O瓶頸變得日益重要。

目前對I/O的改善有很多方法集中在磁碟子系統架構的研究上，較常見的如disk interleaving[2]、striping[3]、RAID[4][5]等等。這些方法基本上是利用多個磁碟同時動作以重疊磁碟存取的時間來增加I/O頻寬。但是這些利用低階層平行(low-level parallelism)的方法如果沒有好的演算法，以及編譯器(compiler)的配合，可以提升的效能有限。所以目前的研究方向轉向較高階層的平行機制：I/O系統架構以及平行檔案系統的設計。

I/O系統架構的優劣，將影響整體系統的效能表現。I/O系統包括I/O伺服器及磁碟機，磁碟機可能以一個或多個匯流排(bus)和I/O伺服器相連。I/O伺服器負責執行I/O要求，將I/O要求有效的分散處理以增加整體的I/O頻寬。平行檔案系統和分散式檔案系統(Distributed File System)以及並時檔案系統(Concurrent File System)並不相同。分散式檔案系統，如Coda、AFS和Sprite等，主要的目的在達到檔案資源共享，透過分散式檔案系統可以提供遠端檔案的存取服務，系統中的檔案可以被不同的使用者共享但不能被平行的存取(parallel access)或具有平行的視界(parallel view)，而且檔案伺服器很容易因為大量的檔案存取而飽和，檔案系統的效能會隨著系統的成長而降低。並時檔案系統，如CFS[10]、Bridge[11]和nCUBE[12]等，和平行檔案系統比較類似，檔案資料分散在多個I/O節點中，但是檔案由使用者的觀點看起來好像放在一個地方，由檔案系統決定資料放置的方式，並沒有提供使用者平行的視界以及檔案分割機制。而平行檔案系統中，檔案資料利用檔案散置(file declustering)的觀念分散在數個I/O節點之中，使得I/O動作能平行存取，並提供使用者檔案實際(physical)與邏輯(logical)的平行視界，使用者看到的是一個分散在各處的檔案，不但可以明瞭檔案的分佈方式，並可以加以控制。這樣的設計使得使用者可以針對自己的需求，定義出適當的檔案分割方式，寫出高效能的平行程式。

和平行應用程式相同，目前大多數的平行檔案系統一般都是在不同的系統平台上做成專屬的檔案系統，移植困難所以開發成本很高。不過自從MPI Forum推動MPI標準（Message Passing Interface standard）以來，平行系統上有了具高度移植性的訊息傳遞(message passing)系統，根據這個標準製作可移植的平行檔案系統變得甚為可行。我們利用MPI可移植性高的特性，設計及製作完成了符合MPI標準的平行檔案系統－MPFS(MPI-IO based Parallel File System)。MPFS包括一群檔案資料伺服器(Data Server)、一個或多個服務協調伺服器(Service Coordinator)與使用者程式庫三部份。檔案資料伺服器負責磁碟存取以及處理使用者程式的I/O要求，檔案資料以散置(striping)的方式放置於檔案資料伺服器上。服務協調伺服器負責管理file handle、檔案的matadata與系統控制動作。

使用者程式庫則參考 MPI I/O 的介面規格製作,提供使用者程式與 I/O 節點之間溝通的介面。

三、結果與討論

我們完成 MPFS 平行檔案系統的設計，其架構使用符合 MPI 標準的 MPICH 函式庫;資料存取函式則根據 MPI-IO 規格製作。隨著 MPI 標準的普及，MPFS 可以減低移植到不同機器平台上的成本，具有高度的可移植性，目前 MPFS 已經可以在 FreeBSD 以及 SunOS 等作業系統上運作。

在 MPFS 系統底層的資料傳輸上，相較於以往的系統,我們改用封送口 (socket) 取代 MPI 的訊息傳遞函式，以提升效能及減少 overhead，避免 MPI 系統對訊息的過度包裝，造成不必要的負擔；同時在 MPFS 系統的實作中，嘗試使用多執行緒來改善伺服器及 MPI-IO 函式庫的整體效能。經過實際的測量結果，驗證了以上兩個方法的可行性。而目前使用多執行緒所遇到的問題在於我們所使用的 MPICH 函式庫並不是 thread-safe 的；因此，使用多執行緒與 MPICH 函式庫所製作的 MPFS 系統，在目前只支援單一個委託者程式讀寫分散於多個磁碟上的檔案，而不支援多個委託者程式讀寫分散於多個磁碟上的檔案。這一點有待於將來 MPICH 函式庫對 thread-safe 的支援，而加以解決。

MPFS 目前建構在以 100 Mbits/sec 高速乙太網路連結的工作站群組上，給予平行程式設計者在設計程式時相當大的彈性。使用者程式可以設定的功能包括：散置區塊大小、檔案在檔案資料伺服器上放置的方式、系統設定的查詢。使用者可以透過 MPI 定義的介面自行定義合適的資料型態來達成檔案的邏輯分割功能，配合 MPI 所定義的資料存取函式，能有效的提高 I/O 動作的效能。根據我們測量的結果，雖然系統效能在兩個檔案資料伺服器時就受到網路頻寬的限制，但是我們相信在未來更高速網路所連結的多處理機系統上應該有更好的效能表現。實驗顯示非同步函式可以有效的將 I/O 動作時和計算時間重疊起來，但是這需要程式設計者對非同步存取函式特性的了解，以及應用程式的配合才能充分的利用。目前 MPFS 只有提供使用者 C 語言的連結，還缺少 MPI-IO 規格中 FORTRAN 的連結函式。而在可靠度、聚集式檔案存取函式及容錯能力方面也可以再做加強，在磁碟階層可以使用 RAID 磁碟陣列，利用 RAID 的同位元可以回復系統中的資料，並記錄各使用程序對系統的存取動作，可以讓系統回復到發生錯誤前的狀態。另一方面可以將檔案系統移植於不同的作業系統及機器平台上，以驗證系統的可移植性。

四、計畫成果自評

在計畫中，對於以 MPI 為基礎的平行檔案系統的設計與製作，舉凡：系統架構設計，平行檔案系統的發展，系統效能評估都以達成預期的目標。此計畫的研究成果可應用於平行檔案系統之設計，也相當適合發表於學術期刊上；本計畫成果將發表於國科會研究彙刊。

五、參考文獻

[1] J. Hennessy and D. Patterson, Computer Architecture: A Quantitative Approach, Morgan Kaufmann, San Mateo, CA, 1990.

[2] Y. Kim, “Synchronized disk

interleaving", IEEE Trans. Comp., C-35, 1986.

[3] Salem and H. Garcia-Molina, "Disk striping", in Proc. IEEE Intl. Conf. Data Eng., 1986.

[4] A. Patterson, et al., "A case for redundant arrays of inexpensive disks(RAID)", in Proc. SIGNOD, 1988.

[5] Stonebraker and G. A. Schloss, "Distributed RAID - a new multiple copy algorithm", in Proceedings of the 6th International Conference of Data Eng., pages 430-437, 1990.

[6] V. Bala and S. Kipnis, "Processes groups: a mechanism for the coordination of and communication among processes in the Venus collective communication library", technical report, IBM T. J. Waton Research Center, October 1992. Preprint.

[7] A. M. Steven and V. S. Sunderam, "A Parallel I/O System for High-Performance Distributed Computing", in Proceedings of the IFIP WG10.3 Working Conference on Programming Environments for Massively Parallel Distributed Systems, April, 1994.

[8] A. Purakayastha, et al., "Characterizing Parallel File-access Patterns on a Large-scale Multiprocessor", in IEEE Internal Parallel Proceeding Symposium, pp. 165-172, 1995.

[9] R. Floyd, "Short-term file reference patterns in a UNIX environment", technical report 177, Dept. of Computer Science, Univ. of Rochester, March 1986.

[10] P.Pierce, "A concurrent file system for a highly parallel mass storage subsystem", in 4th Conf. Hypercubes, Concurrent Comput.,&Appl., vol. I, pp.155-160, May 1989.

[11] Peter Dibble, et al., "Bridge: A High Performance File System for Parallel Processors", in Proceedings of the Eighth International Conference on Distributed Computer Systems, pp.154-161, 1988.

[12] Erik DeBenedictis and Juan Miguel del Rosario, "nCUBE Parallel I/O Software", in Proceedings of the Eleventh Annual IEEE International Phoenix Conference on Computers and Communications, pp.117-124, 1992.