行政院國家科學委員會專題研究計畫成果報告

多處理機作業系統設計之研發

A Study on the Design of Multiprocessor Operating Systems

計畫編號：NSC 87-2213-1009-954
執行期限：86 年 8 月 1 日至 87 年 7 月 31 日
主持人：謝續平教授　　國立交通大學資訊工程學系

一、中文摘要

我們在本年度中將特別著重在多處理機系統中虛擬記憶體保護架構的設計，根據單一定址空間的概念，研究如何有效的保護不同程序的位址空間與達成有效率的資源共享，我們更將依此設計之虛擬記憶體架構設計與製作一虛擬檔案系統，此一虛擬檔案系統將不同於傳統檔案系統的架構，而是與虛擬記憶體相互配合，以期達到高傳輸率與低等待的特性。

關鍵詞：虛擬記憶體，單一定址空間，虛擬檔案系統

Abstract

Conventional virtual memory schemes are used to support private address space operating systems, such as UNIX. They cannot handle virtual memory management of single address space operating systems on modern 64-bit architectures. In this project, we introduce a new memory protection model to manage single virtual address space, we also develop a file system, called distributed virtual file system (DVFS), based on the concept of the single address space and the one-level storage. The storage of the file system is built in the swap devices, therefore, when a file access is invoked, only one time of demand paging is required to transfer data between swap devices and user buffers, avoiding the conventional copy behavior from kernel space to user space in most of the current UNIX operating systems.

Keywords: Virtual Memory, Single Address Space, Virtual File System

二、計劃緣由與目的

現有的檔案系統在讀取檔案時，必須由系統核心(kernel)讀進系統內部區塊緩衝區(block buffer)，然後再從此處拷貝至使用者緩衝區(user buffer)（read 系統呼叫的第二個參數）。這層介於使用者和核心之間的傳遞雖然大部分 UNIX 的建置都儘量使其拷貝速度加快，但所耗掉的時間仍是相當可觀。此時六十四位元定址空間、單一定址模式的發展，給了我們一種想法：如果將「physical memory」和「整個 disk device」結合起來看成同一個 memory space（virtual address space），那 disk device 上的檔案資料等於是 virtual memory 的一部分，此時存取檔案就變成是對 memory 的存取，自然不必再透過核心緩衝區（kernel buffer）。因此我們決定製作一個檔案系統來模擬上述的情況，並且評估效能。

在 UNIX 作業系統上製作一個檔案系統（以下稱為“虛擬檔案系統”以便與原有的檔案系統區別）用以模擬單一定址空間的系統，並與原有的 UNIX 檔案系統比較效能。

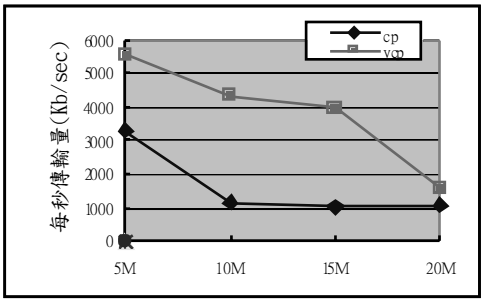
將虛擬檔案系統擴充，讓不同機器的使用者也可以使用到虛擬檔案系統的資源，我們初步計畫了以一個類似 FTP 的工

具來作測試。接著研究將不同機器上的虛擬檔案系統集中管理的可行性，讓使用者不必費心資源所在。

三、結果與討論

虛擬檔案系統最大的優點可以節省一次記憶體的拷貝動作，並且分成最佳及最差狀況討論，我們便對此進行測試及實驗評估來驗證理論的正確性。

由於最差狀況無法有效控制，加上我們的重點在於記憶體的拷貝途徑, 所以我們僅測其最佳狀況。且為了減少不必要的變因，我們僅比較資料從讀取到寫入的時間，也就是不包括 open 及 close。為了達到最佳狀況（也就是所要讀取的資料都在記憶體中）我們將拷貝動作連續執行幾次，如此一來所欲讀取的檔案內容便很有可能存在記憶體中，這時候再記錄幾次執行的時間取其平均值，得到之結果如下圖：



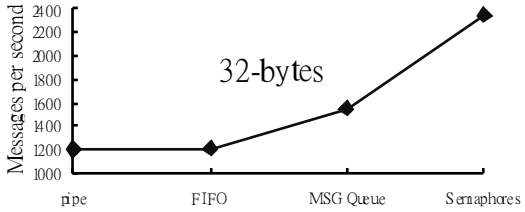
圖一

理論上，在最佳狀況之下，所花的時間只在於記憶體的拷貝動作，由於虛擬檔案系統節省了一次 memory to memory 的拷貝動作，故速度應為原系統的兩倍。從圖七中我們可以發現，5M 時 vcp 的速度大約是 cp 的兩倍（此刻兩者的資料大都還在 memory 裡），因此，與我們的理論相符合。

然而系統 buffer size 以及記憶體空間有限，當檔案一大時，資料不可能完全在 memory 裡。因此，將資料從磁碟拷貝到記憶體的動作在所難免，且這段時間所佔的比例將會隨著檔案增大而增加（趨近於最差狀況的情形），使得記憶體省下來的時間的影響大大減小，兩者的差距將逐漸縮小。這一點從實驗結果也可以得證實。

再來看看這兩條曲線的變化，兩條曲線都有一段傳輸速率急降的現象，這段急降應該就是磁碟拷貝動作介入的分界，明顯地看得出來，原本系統的分界比較早出現，造成中間兩者的差距大到不只兩倍，所以利用虛擬檔案系統的方式對於記憶的使用彈性比較大，當記憶體足夠時虛擬檔案系統的效能將更有提昇的空間。

我們所完成的虛擬檔案系統尚有幾個地方可以改善，在這裡提出來討論。將原本用 BSD socket 傳送 message 的方式改為 IPC 的方式，沒有必要用到 socket，而圖二顯示使用 semaphore 的 IPC 效果最好。



圖二、 在 Compaq 386(20 MHz Intel 80386 處理機)上執行 IPC 的績效

對於大部份的程式而言，檔案的動作多為讀取檔案資料，所以將檔案拷貝到使用者緩衝區實非必要，因此若我們可以直接取得檔案內容而不透過使用者緩衝區的話，這樣一來又省了一次記憶體的拷貝動作，見圖三。這對我們的系統而言是輕而易舉的事，只不過使用者必須改變原有的寫作方式，而改以下面的語法：

```
(char *) file_addr(int *byte);
```

file_addr 便是我們尚未發展的功能，其輸入的變數為一個整數指，function 結束之後會將可以處理的大小存於在些變數之中，而 function 的回傳值便是檔案的記憶體位置，若已至檔案尾則傳回 NULL。

```
char *data;
int size;
while ((data=file_addr(&size) != NULL)
{
     /* 此時便可以對 data[0]至 data[size-1]
        作處理 */
```

}

原本我們的目在於模擬驗證我們的理論，因此一些與我們的理論無直接關係的部分便省略了過去，如權限的認証。為了讓系統更為完善，應該把我們所省略的部分補上去。這樣一來，我們的系統就不只停留在模擬的階段，可以真正的實用。

四、計劃成果自評

在本年度的計劃中，針對虛擬檔案系統我們完成了以下的軟體元件：

程式庫：提供的 library：vopen、vcreat、 vclose、vread、vwrite、vlseek 等。只要在編譯時加入 libdvfs.a 這個 library，使用者便可以直接使用。

指令集：在 shell command 下，我們系統提供：vls、vdf，、vcp、 vrm 等指令。此外，還有 vput 及 vget，前者是將原本系統中的檔案，複製到虛擬檔案系統中；後者則是將虛擬檔案系統中的檔案，複製到原本的系統裡。

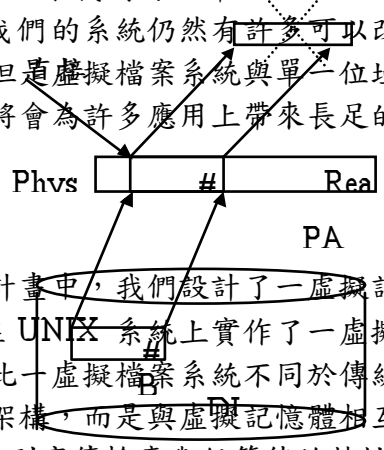
應用程式：vftp 功能與 ftp 類似。使用者可由遠端取得虛擬檔案系統裡的檔案。

在以上各部份的實作上，我們使用具有虛擬記憶體特性的共享記憶體(share memory) 作為單一定址模式的媒介。我們將系統的 share memory 作為虛擬檔案系統的空間，如此一來就可以直接對指定位置存取。當作業系統需要記憶體資源時，share memory 自然會被 page out 至 swap device，相當於單一定模式中 page 至 disk device 一樣；當我們要存取該位置時，再從 swap device page in 到記憶體，亦與單一定址相同。

另外，我們也採用主從式(client-server)的架構，服務端(server)負責管理虛擬檔案系統，其中包括虛擬檔案系統的初始化以及等待委託端的要求，委託端(client)負責使用者與虛擬檔案系統的溝通，其中包括將訊息傳遞給使用者，以即將使用者的要求傳遞給服務端(server)。

經由以上各部份軟體的實作，我們實現了將「physical memory」和「disk device」結合成同一個位址空間的觀念。透過這樣的單一位址空間，檔案存取就變成是記憶體之間的存取，存取速度和效能有了明顯的提升，驗證了我們的設計。

雖然我們的系統仍然有許多可以改進的部份，但是虛擬檔案系統與單一位址空間的觀念將會為許多應用上帶來長足的發展



五、結論

在本年度計畫中，我們設計了一虛擬記憶體架構並在 UNIX 系統上實作了一虛擬檔案系統，此一虛擬檔案系統不同於傳統檔案系統的架構，而是與虛擬記憶體相互配合，以期達到高傳輸率與低等待的特性。

六、參考文獻

[1] Maurice J. Bach, "The Design of The UNIX Operating System", Prentice Hall International Editions.

[2] W. Richard Stevens, "UNIX Network Programming", Prentice Hall International Edition.

[3] Marshall Kirk McKusick, Keith Bostic, Michael J. Karels, John S. Quarterman, "The Design and Implementation of The 4.4BSD Operating System", Addison Wesley.

[4] C. Shen, "Virtual Address Translation and Protection for Single Address Space Operating Systems on Wide Address Architectures"

[5] J. P. Jou, "Scheduling Schemes for Reducing the False Sharing in Cache Coherent Multi-processors"