第四章 實作與分析

4-1 測試環境與方式

硬體環境	軟體環境
CPU: P3-1G x 2 RAM: 512MB x 4 SDRAM HDD: 4.3G SCSI，250G PATA 硬體越快，當然執行的速度就越快，所需時間就越短。	OS: FreeBSD/4.10-RELEASE-p20 Apache/1.3.36 PHP/4.4.2 MySQL/3.23.58

表 6：測試環境軟硬體列表。

Apache 的基本調校已針對此硬體環境調整到最佳數值；PHP 的部分則不使用 Zend Optimizer，也不使用 Turck MMCache，以測試方案一之成效；MySQL 則使用套件中所附的 my-huge.cnf 設定檔案來運作，以期在記憶體用量充足的情況下，測試方案二與方案三再度提升效率的成果。

因需要測試 1000 人同時上線的情況，apache MaxClients 原始的上限是 512，經修改原始碼為 1024 來配合測試。

所有 PHP 程式碼均以 Byster.NET 所提到的效率較佳的方式來撰寫，減少不必要的 CPU 運算以提高測試數據準確性。

測試過程中均以 ab - Apache HTTP server benchmarking tool - 來進行執行時間的取得與計算，且都以 -c concurrency 參數進行同時上線人數的測試。

4-2 方案一：除冗碼的實際測試結果

這是在考試系統中，以導師功能的 PHP 程式(約 1500 行)來進行程式碼分而治之的處理。在各自保留 1/10/100/1000 行冗程式碼時，在人數 1/10/100/1000 人同時存取的情況下，每人平均需要等待多久的時間，才會看到傳回的網頁內容。

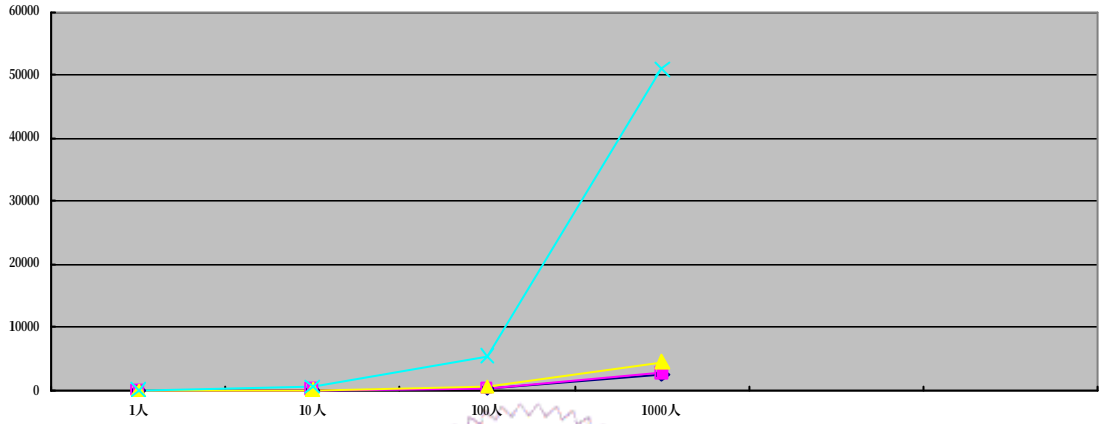

圖 7：冗碼與同時上線人數所花的時間比。

由這數據來看，將程式碼分而治之，減少執行時冗程式碼的數量，大幅減少 parser 和 syntax checker 執行的時間，加速了程式的執行與減少等待的時間，有效的提昇執行的效率。就算只有一人使用，減少 1000 行的冗程式碼也可以縮短十倍的執行時間，在連線人數更多的時候，效能提昇得更顯著。

4-3 方案二：減少 JOIN 的實際測試結果

考試系統中的學生功能中，查詢學生詳細資料功能中，在 SELECT 的時候就會有 JOIN 過來的資料表，像是所屬學校、所屬學校所在區域、所屬班級類型、所屬老師，在這種情況下就 JOIN 了四次。修改這部分的 SQL 指令和學生的資料表結構，各自以 0/5/10/15 次的 JOIN 項目來送出查詢的 SQL 指令。

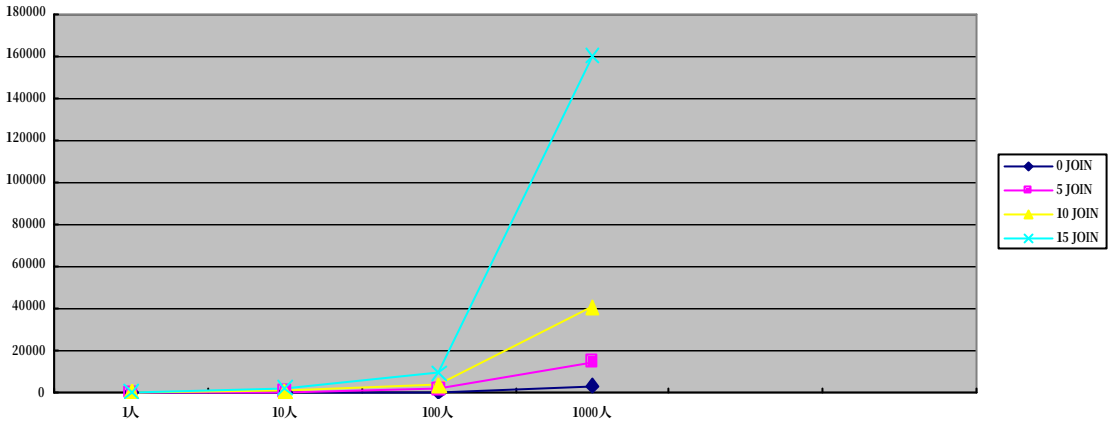

圖 8：JOIN 次數與同時上線人數所花的時間比。

在這測試過程中，MySQL 已經是使用 my-huge.cnf 的設定值在運作，所以 SQL 指令的所有 JOIN 的動作都是在記憶體中完成，並未動用到硬碟來做暫存空間，所以執行出來的數據就純粹是 CPU 處理 JOIN 時所使用的時間。

很明顯的每多五個 JOIN 的動作，所花的時間大約就變成原來時間的 3 倍左右，同時使用的人數越多，所需的時間可能超過 3 倍甚至更多。反過來說，在這種情況下只要減少 5 個 JOIN 的動作，執行時間就可以縮短為原來的 1/3 以上，同時使用人數越多時，效能提昇得更明顯。

4-4 方案三：降低資料搜尋空間的實際測試結果

考試系統中有一個巨大的題庫資料表，每一次舉辦考試都得依照考卷指定的範圍條件或是題號，從題庫中找出所需的題目來使用。這次準備了兩個題庫資料表，各自有 50 題和 1000000 題的數量，而這 50 題試從 1000000 題的題庫中指定題號分離出來的，再從中 SELECT 出相同題號的 50 題來進行測試。

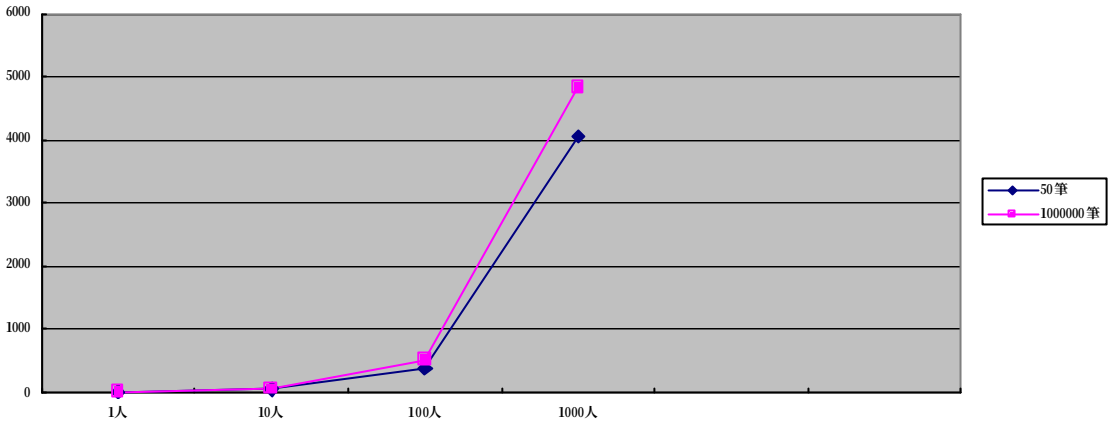

圖 9：資料空間數量與同時上線人數所花的時間比。

由於 MySQL 是以 my-huge.cnf 的設定值在運作著，所以兩個各 50/1000000 筆的題庫在執行一次後就會整個 cache 在記憶體中，而不需要再次從硬碟中讀取出來。先執行一次讓兩個題庫都 cache 在記憶體中之後，測出來的數據就純粹是 CPU 在處理 SELECT 指令的運作時間，不包含硬碟存取所需的時間。

就算是所有的資料都在 MySQL 的 cache 記憶體中進行 SELECT 的動作，資料總數量的不同還是有執行時間上的影響，而且越多人同時使用系統時越明顯，甚至能夠相差 1/5 的執行時間。