輔仁大學 107 年度高教深耕計畫

自主學習課程計畫

成果報告書

106 學年度第 二 學期

（雲端計算概論）

授課教師：林振緯

系所單位：理工學院 資訊工程系

中華民國 107 年 7 月

目錄

1. 課程指導成果說明 3

課程實際規劃與說明 4
具體教學成果與評估 4
課程遇到問題與困難 4
省思與未來的展望 4
優秀自主學習組別 4

2. 學生學習成果 5

2.1. 第一組 6

自主學習計劃書 6
成果報告 11

2.2. 第二組 15

自主學習計劃書 15
成果報告 21

2.3. 第三組 24

自主學習計劃書 24
成果報告 31

2.4. 第四組 34

自主學習計劃書 34
成果報告 42

2.5. 第五組 46

自主學習計劃書 46
成果報告 51

2.6. 第六組 54

自主學習計劃書 54
成果報告 59

2.7. 第七組 62

自主學習計劃書 62
成果報告 67

2.8. 第八組 69

自主學習計劃書 69
成果報告 73

3. 活動紀錄 78

演講（一） 79
演講（二） 80
演講（三） 81

4. 其他附件 82

學習問卷 82
演講心得 82

1. 課程指導成果說明

課程實際規劃與說明

學期前半由教授帶領學生先理解雲端的基本概念以及現有雲端服務的介紹，並安排兩次演講探討分散伺服器的佈署方法以及分散式資料的儲存與處理。學期後半則是著重讓學生分組閱讀論文或是實作論文內容，最後在台上發表成果。

具體教學成果與評估

報告理論的組別在報告時，有把論文內的演算法以及探討案例解釋詳盡。有實作的組別則是示範論文中的實驗內容，並與論文內部數據做比較。以報告成果來看，每一組都有達到理解雲端資料處理的一些解決方案。

課程遇到問題與困難

報告理論的組別很多都面臨到演算法理解的困境，但在指示研究方法以及時間的演進後有些許的改善，所以報告時沒有過多停頓。實作的組別則因為時間倉促，沒有足夠的時間在實驗結果上做縝密的分析，所以報告上沒有辦法聚焦在解決方法。

省思與未來的展望

大學生在研究與實作上確實沒有論文研究者謹慎，但在論文閱讀仍有可發展的空間。雖然我們對論文只求不脫離雲端主題，但是在報告內容中有些主題偏向架構部分，若能往解決方案以及擴展論文應用發展會有很大的發展空間。

優秀自主學習組別

實作組別我們遴選第四組，這組的報告主題為：A Kubernetes-Based Monitoring Platform for Dynamic Cloud Resource Provisioning，主要是因為這組的主題著重在該架構分析，也了解 kubernetes、docker 詳細運作方法，並在報告上也有根據他們實際觀察到的狀況作出解釋，並做出優缺點比較以及簡易的實作心得，為實作部分的優秀組別。

理論組別我們遴選第八組，這組的報告主題為：Efficient Storage of Multi-Sensor Object-Tracking Data，主要是因為這組的主題並重在架構與解決方案，在報告時有詳實說明該論文的演算法與架構如何解決實際遇到的問題，並對現有的解決方案作詳實比較。

2. 學生學習成果

2.1. 第一組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：GAE 的技術與實作

二、組別：第一組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：

姓名	工作內容	聯絡方式 (E-mail)
吳采諺	報告+查資料+做 PPT	stuffed567@gmail.com
許桓順	查資料+做 PPT	
黃欽偉	報告+實作+做 PPT	
林浚宏	報告+實作	
李明翰	查資料+彙整	

六、 計畫之動機與目的：

1. 計畫內容與行動方案： 把組員分成兩部分，一部分負責查資料，一部分負責實作，各組自己做好後要推派一個人上台報告。

2. 進度規劃

週次	規劃內容
第一週	討論分工
第二週	查資料+實作完成 25%
第三週	查資料+實作完成 50%
第四週	查資料+實作完成 75%
第五週	查資料+實作完成 100%

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期：107 年 6 月 18 日(一) 00:00~00:00

紀錄人：吳采諺

序號	工作事項	註記(心得/成果/問題 等)
	報告	
	查資料	
	PPT	

待辦事項/待解決問題/筆記欄

老師意見及回饋

1. 可查閱網路上碩博士論文但不能完全抄寫
2. 方向內容很不錯，撰寫論文的時候內容要詳細一點
3. 分工記得要明確

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期： 107 年　 6 月　 18 日

一、課程基本資料

(一)自主學習計畫主題：GAE 的技術與實作

(二)組別：第一組

(三)學生姓名：吳采諺、許桓順、黃欽偉、林浚宏、李明翰

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：每個禮拜我們都會約出來盯進度，執行狀況都很順利，每個組員都很盡責在時間內達成目標。

(二)成效說明與實際產出：見論文。

(三)自主學習歷程：內容方向不錯，內容可以參考網路文獻。

三、學習心得

林浚宏

在這堂選修課的學期末之前，教授讓我們五到六個人一組，使我們有個機會去自主學習、研究平常上課可能不會探討到的課程內容，我們這組以相關論文為主題，分工做出了一份報告。

起初大家在分工前都在一個一個尋找哪篇論文適合，需要不是太多技術層面的相關文獻，也需要符合課程內容的範圍，在這裡我們就費了一番苦心，其後在分工前，大家也都花了時間去研讀該篇論文的內容，我們便開始了分工、製作報告。但在過程中，原本我們是有想要照著論文的內容去跟著去做看看，能不能達到實作模擬的部分，但該篇論文的實作使用到的 POS 系統卻不是每個人都可以輕易拿到的，甚至在網路上的絕大部分皆須付費，因為那是一款商業用的系統，我與另一位負責呈現實作的組員有點不知所措，雖然沒有成品可以展現，但還是希望能藉由作者的實驗結果為大家展現使用 Google 引擎所帶來的功效。我想這是一個在未來人生中必須記取的一個小教訓，雖然作者內容有大部分的解釋步驟如何達成目標，但有些設備或是環境，都必須先自己試著做才知道能不能達成，才不會造成組員的困擾。

黃欽偉

我們組以 Google Engine 的架構來做學習與探討，從 Google 雲端方面著手，了解整個運作的方式以及實作來測試雲端運算及一般平台運算的差異。在還沒接觸雲端到了解其超強的資料處理能力，現在是資料爆炸真的是需要靠這種技術才能解決，但在實作方面，我們用了 POS 系統加上 Google 雲端來看它的強大，發現跟一般的平台在一千萬筆資料就差了 250 倍的數度，但也知道雖然雲端運算很強大，也是有其缺點及目前技術無法突破的地方，這也是我收穫作多的地方，首先是資料庫方面，大量不同的資料類型造成 Google 所使用的關聯式資料庫會慘聲很大的負擔來處理，加上一次只能接受幾千筆資料，對於每秒上億筆的資料量根本無法承受，還有設計系統方面，要怎麼配合 Google Engine 做出一套系統是整合過後的，也就是不只侷限設計給單一族群或是商家，這些讓我深深地學到設計一套軟體，雖然有強大的套件幫忙或是技術的協助，卻總是有更多的問題也要克服，也就是沒有一個完美的東西，只能讓他更加的好、更加地符合大眾的需求。

許桓順

雲端計算是個近期很熱門的項目，也因為如此，我以外系的身份修了這堂課。相對於一般的資工課程是在電腦前完成，雲端計算概論採用了不同的教學流程，扎實的知識訓練使專業核心能力能有效汲取，也給予學生口頭報告的機會，培養穩健的台風與對答的能力；並非只是坐在台下聽講，而是換為思考，練習台上表達。在報告摸索與探尋的過程中，除開將所學的知識用於實作當中，也練習資料搜尋、參考文獻的檢索、論文擇寫等等，為未來學術研究奠定基礎。其中對團隊合作的部分也有很多的練習，學習分工、討論、開會。最後期末報告能完成，特別感謝組員的付出與老師的指導。

李明翰

這堂雲端計算服務概論也差不多告一個段落了，其實雲端與我們資工息息相關，這堂課，當初選擇也是秒選的，畢竟很有幫助，除了老師自己講解，也請到外校的老師也幫我們做演講，算是非常的充實，再者，期末的報告，我們這組做了 google 相關的資料做為報告，分工合作也完成了我們的上台內容，算是非常的圓滿。

同時，邊做報告邊學習知識，上課為了什麼，想當然就是要應用，我們在報告裡實做，也學到很多新的東西，一舉多得，也是這堂課帶給我的感觸。

雲端的出現為我們帶來了許多的便利，但也對台灣的產業也造成了一些衝擊。但面對雲端的新興，雲端的產生，造成了伺服器的需求減少，對台灣生產國家來說不利。但我覺得，科技與產業不斷的創新，舊有的產業能維持到現在，代表已具備面對新興產業興起的能力，舊有的產業仍能調適、創新並找到屬於自己的市場。

希望我以後如果有能力接觸雲端的工作，也能盡一份我的心力去開發更好的功能!

吳采諺

一開始我們在想主題的時候其實蠻緊張的，因為雲端這個範圍太廣了不知道要從哪裡下手，翻了很多相關資料內容也都很深，也怕找到的主題無法完整的呈現出一個專題論文，後來在大家努力查資料下我們決定做關於 google 的 GAE。我覺得我應該是在這個組別裡面學到最多的，因為我同時負責查資料，撰寫論文，做 PPT 還有上台報告。查資料其實還沒有吸收到的感覺，就只是把收集到的資料整合輸出，後來決定上台報告後才開始認真地研究自己的資料，雖然過程有點緊張也覺得自己壓力很大，但報告完之後發現自己確實也吸收到很多也得到很多就覺得一切都值得了。雲端真的是一個未來的一個大趨勢，不僅跟我們的生活息息相關，而且還深深地影響我們現在的生活品質，很感謝這堂課讓我學到這麼多。

成果報告

雲端概論報告
Google App Engine 技術與實做

MapReduce 的執行過程
Step 1
將要執行的 MapReduce 程式複製到 Master 跟每一台 Worker 機器中。
Step 2
Master 決定 Map 程式與 Reduce 程式，分別由哪些 Worker 機器執行。
Step 3
將所有的資料區塊，分配到執行 Map 程式的 Worker 機器中進行 Map。
Step 4
將 Map 後的結果存入 Worker 機器的本地磁碟。
Step 5
執行 Reduce 程式的 Worker 機器，遠端讀取每一份 Map 結果，進行統整與排序，同時執行 Reduce 程進行式合併。
Step 6
將使用者需要的運算結果輸出。

研究動機
MOTIVATION
Google 為了因應雲端運算，也提供了一連串的服務，如日曆、文件、協作平台等等，可說是提倡雲端運算的先趨，Google 也為了雲端運算的概念提供了開發環境，也就是Google App Engine，透過 Google App Engine 開發出的應用，即是雲端運算的概念，未來「軟體即是服務」的概念也會相應而生，因此本篇論文將以 Google App Engine 為例，發展出以雲端運算為基礎的簡易 WebPOS 系統來完成本論文之研究。

Google
Bigtable

研究方法
APPROACH
研究方法採用文獻分析法，蒐集關於雲端運算與 Google App Engine 的相關資料，並加以分析，進而瞭解雲端運算的運作基礎與架構。利用 Google 所提供的 Google App Engine 平台實作應用程式，Google App Engine 可以讓開發者能更方便與直接開發基於雲端運算架構的應用程式。

Google Bigtable
何謂Bigtable
是一種壓縮的、高效能的、高擴展性的，基於Google檔案系統的資料儲存系統，用於儲存大規模結構化資料
資料庫
BigTable不是傳統的關係型資料庫，不支援JOIN這樣的SQL語法，BigTable更像今日的NoSQL的Table-oriented，優勢在於擴充功能性和效能。
應用
適用於雲端計算，現今已成為Google的應用程式。

GAE的技術
第一部分
MapReduce
第二部分
BigTable
第三部分
AJAX 非同步傳輸
第四部分
MVC 架構
第五部分
Google File System

AJAX
非同步傳輸

Google
MapReduce

AJAX非同步傳輸三步驟
近端（用戶端）的發送函數
遠端（伺服器端）的程式碼
近端（用戶端）的接收函數
Step 1
負責在接收主網頁的事件後，設定 AJAX 物件，並對伺服器發送 request 以啟動伺服器端程式碼的執行。
Step 2
通常是一個 ASP 網頁，負責在伺服器執行必要之步驟，例如檢查帳號密碼，或是對資料庫進行查詢等。
Step 3
負責接收伺服器的執行結果，並將結果以非同步的方式顯示在主網頁上。

MVC 架構

Google File System

GFS的應用

GFS專門為Google的核心資料即頁面搜尋的儲存進行了最佳化。

其他優點

它也是針對Google的電腦集群進行的設計和最佳化，這些節點是由廉價的「常用」電腦組成，這就意味著必須防止單個節點的高損害率和隨之帶來的資料遺失。

GFS的系統

這是一種分散式檔案系統，是一套能快速穩定及安全的資料儲存系統。

MVC 架構

Model

包含所有的邏輯、物件、內容管理。

Controller

盡量輕量，這裡盡量不撰寫邏輯與物件，而以路由以傳遞資料為主。

View

僅呈現，故盡量單純(笨笨的呈現即可。

Google App Engine 的功能

GAE 發展歷程 → GAE功能 特性&優點 → GAE的 API

MVC的優點

- 增加程式可用性
- 程式方便管理
- 程式擴充性高
- 有例於團隊開發

Google 發展歷程

MVC的缺點

- 不適用於小型專案
- 管理文件增加
- 嚴謹的系統架構與規劃
- 需要重覆的測試

Google 雲端發展歷程

- **2000** Google Adwords
- **2006** Google Talk, Pisca Web Album, Google Calender
- **2007** Google Apps
- **2008** Google App Engine

Google File System

Google App Engine 特性

Google App Engine 的特性

動態網頁服務，支援常見的網頁技術

可永久儲存查詢、排序及交易

自動擴充和負載平衡

容易建立、管理應用程式

可整合 Google 所提供之 API

全功能的本機開發環境

支援多種程式語言

支援異質性平台開發

Google App Engine API

資料存放 API

背後由 BigTable 所支援，使用 GQL 查詢語法。

影像 API

提供影像操作服務，以便在網頁要求期間執行簡單的影像轉換。

郵件 API

透過 Gmail 電子郵件與使用者溝通，或是將活動通知其他人。

Memcache API

記憶體快取服務，使用分散式記憶體中的資料快取，預先處理而永續的儲存。

URL 擷取 API

使用 HTTP 要求，透過網際網路與其他主機溝通。

使用者 API

與 Google 帳戶整合，應用程式可以讓使用者使用 Google 帳戶登入，並於工作期間取得知道是誰正在使用應用程式。

Gdata 服務 API

可以存取儲存在 Google 服務，如：日曆事件、GMail 通訊錄中的使用者特定資料。

Java API

提供給 Java 語言用的 API，讓開發人員可輕鬆使用 Java 語言開發雲端應用程式。

資源種類的限制

01 應用程式的佈署數量

02 CPU的使用時間

03 Cron排程工作的數量

04 URL擷取的頻寬限制

05 1GB的免費使用空間

06 API的呼叫次數

深入分析

Google App Engine 優點

一般伺服器平台與雲端運算平台效能比較

- 經由圖表可知PHP平台運算複雜的程式所耗費的時間相對雲端運算還要慢的很多。

實驗組 PHP 平台	對照組 Google 雲端運算平台	實驗組 PHP 平台
迴圈 10000 次	耗用 2 毫秒	耗用 0 毫秒	迴圈 10000 次	耗用 2 毫秒
迴圈 100000 次	耗用 29 毫秒	耗用 3 毫秒	迴圈 100000 次	耗用 29 毫秒
迴圈 1000000 次	耗用 321 毫秒	耗用 6 毫秒	迴圈 1000000 次	耗用 321 毫秒
迴圈 10000000 次	耗用 3644 毫秒（3.6 秒）	耗用 21 毫秒	迴圈 10000000 次	耗用 3644 毫秒（3.6 秒）
實驗組 PHP 平台	對照組 Google 雲端運算平台	實驗組 PHP 平台

Google App Engine 的優點

通用於所有語言社群

立即可用的 App Engine 支援多種語言。屬於這些語言社群的開發人員只要在熟悉的環境中新增程式碼，就能立即投入實際執行工作。

開放式的未來雲端服務

開發人員都能採用相同於特定平台的專有工具或是受到人為限制的技術。您可以提供自己的 Docker 映像檔或自行導入自訂套體堆疊，與為您執行執行階段、架構或第三方資料庫相容開發。

只需加入程式碼

不必擔心如何佈置高品質的程式碼。無論是開發應用程式的前端以處理流量、取得負載平衡、檢查及修正執行個體的健康狀態，以及在基礎的 OS 中套用更新。

Google App Engine

不斷成長的服務生態系統

只要執行 API 呼叫，就能透過 App Engine 費用預約不斷成長的雲端化服務生態系統。

視需要使用基礎架構

享受全方位管理型開發人員體驗，或視需要調整可無限控管的基礎架構。您可以直接與執行個體建立 SSH 連線，以即看自訂程式碼、管理佈署、或是將 App Engine 整合到您的開發運作程序中。

適用於任何工作負載的統對端計算

許多大型公司在混合式（但往往是多雲端）環境中執行既有或舊版應用程式。目可能將這些應用程式部署至 VM、容器或是全方位管理的平台。GCP 在這種多元運環中脫穎而出。

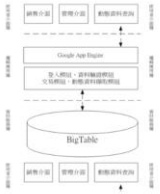
WebPOS系統介紹

- POS系統是很常見的銷售管理系統，運用了雲端運算，計算出每一筆買賣紀錄，並快速丟到資料庫做統整運算
- WebPOS以三層式架構來建構此系統，使用者只要透過瀏覽器就可以進行收銀或管理的動作。

Google App Engine API

WebPOS系統架構

- 使用者介面層: 透過Google Web Toolkit工具與伺服器做溝通
- 邏輯應用層: 包括許多動態模組來提供POS的資料處理
- 資料服務層: 以Google 的 BigTable資料庫做為WebPos使用，BigTable透過索引的方式紀錄資料存取，可以加快存取速度



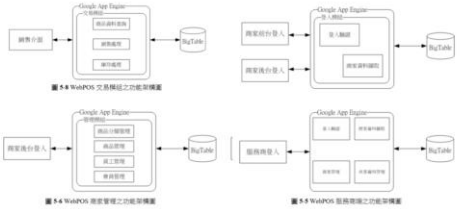
WebPOS 基礎架構

- 收銀端: 商家收銀用，只要透過web即可收銀，大部分的資料來源皆由商家管理端進行設定
- 商家管理端: 後台管理，所以商家個別做POS設定後，於收銀端應用，各家獨立運作，互不影響
- 服務商端: 設定所有給商家的整體設定，此端的設定項目，立即應用所有商家的POS服務，商家的權限也有此設定

結論
Conclusion

利用Google App Engine 結合Google Web Toolkit來發展WebPOS系統，可達到Uniquitous Network 與資料共享、各自維護、互相使用，主要是Google App Engine的雲端服務提供強大的資料處理能力與運算能力，而Google Web Toolkit則替代程式設計師減輕工作壓力，但在這過程中也發現許多待改進的部分。

POS系統功能模組



結論
Conclusion

① 資料放在共用的網路上：所有資料皆放在遠端伺服器上，無法完全自主管理，也有機率發生檔案損毀。

② 資料量的問題：一般關聯式資料庫都存在資料量的問題，過多資料造成關聯太過複雜，造成資料庫快速膨脹，而導致搜尋速度慢。而BigTable雖然採用分散式儲存及壓縮技術，但限制筆數只能1000筆，動則上億的零售業資料內容目前也無法解決

WebPOS介面介紹

※登入帳號

※設定要使用的資料表

結論
Conclusion

③ 系統靈活性：未來的系統必須走向整合性，來達成縮短作業時間、人力浪費、維持資料一致性及增加系統效率，如此一來才能在系統的發展有所創新，而POS系統橫跨各行各業，設計若不夠靈活，將無法符合多數業者的需求。

④ 無網路問題：雖然雲端運算提供強大且快速的服務內容，但無網路的那一刻，所有依靠雲端的企業將會造成相當大的損失。

WebPOS介面介紹

※設定商家、顧客資料

※ 服務端商家設定

雲端概論報告

THE END

WebPOS介面介紹

※商家資料管理

※ 收銀介面

2.2. 第二組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：探討 Maximum utilization allocation、Job splitable allocation 資源分配方法

二、組別：第二組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：(可以個人或團體方式執行，至多 5 人)

姓名	工作內容	聯絡方式 (E-mail)
陳馥鈴	查資料	open19961115@gmail.com
李婉妍	查資料、整理資料	dory860811@gmail.com
楊安宜	整理資料、製作 ppt、講稿	annielikebell@gmail.com
沈子齊	順講稿、上台報告、熟讀 ppt	mdoinktheclown@gmail.com
吳盈泰	查資料	h12020282@gmail.com

計畫之動機與目的：計畫內容與行動方案：一週開一次會，讓大家報告這週的進度及下週的目標，進而達到大家能過開會互相督促的效益。

進度規劃

週次	規劃內容
第一週	確定題目方向，了解相關背景知識
第二週	彙整資料，了解相關技術
第三週	討論 ppt 內容，開始製作 ppt，製作教學講稿，技術應用延伸
第四週	修訂 ppt 教學內容，找出問題癥結點，所有工作內容皆初步定案
第五週	試教學，教學內容、ppt、所有工作內容皆定案

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期：107 年 5 月 22 日(二) 09:00~12:00

紀錄人：楊安宜

序號	工作事項	註記(心得/成果/問題 等)
一	資料不夠齊全，導致討論沒有效率	陳馥鈴要找更多資料
二	PPT 感覺資料太少	本來就是想要很乾淨的 PPT，很難做取捨
三	試報告的不太好	很多細節因為太緊張都忘記了，還要再特別訓練口才
四	資料找太久，後續工作都延期	開會時被大家撻伐，找資料不能太懶惰
五	討論到後期，還是有組員不在狀況內	有加開更多次會議讓大家知道自己要幹嘛

待辦事項/待解決問題/筆記欄

我覺得跟和尚打水喝的故事一樣，越多人一起共事，就會有越多的冗員，但是這正是分組討論的意義，就是要試著跟大家一起合作，所以我覺得組長是個最重要的角色，當大家都不知道自己要做甚麼的時候，就是組長應該要分配工作的時候，一個團隊一定要有一個領導人，要不然就會像一盤散沙一樣，可能到要報告的時候甚麼東西都還沒做好吧，因為這次剛好有這個機會當上了組長，才能更深刻的體會到團結的重要性，但是我希望，大家都能主動的去幫忙，不要總是被動的希望有人會先跳出來，經過幾次討論後，我發現大家有漸入佳境，很感謝大家。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期：107 年　6 月 15 日

一、課程基本資料

(一)自主學習計畫主題：探討 Maximum utilization allocation、Job splitable allocation 資源分配方法

(二)組別：第二組

(三)學生姓名：楊安宜、沈子齊、吳盈泰、李婉妍、陳馥鈴

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

真善美聖

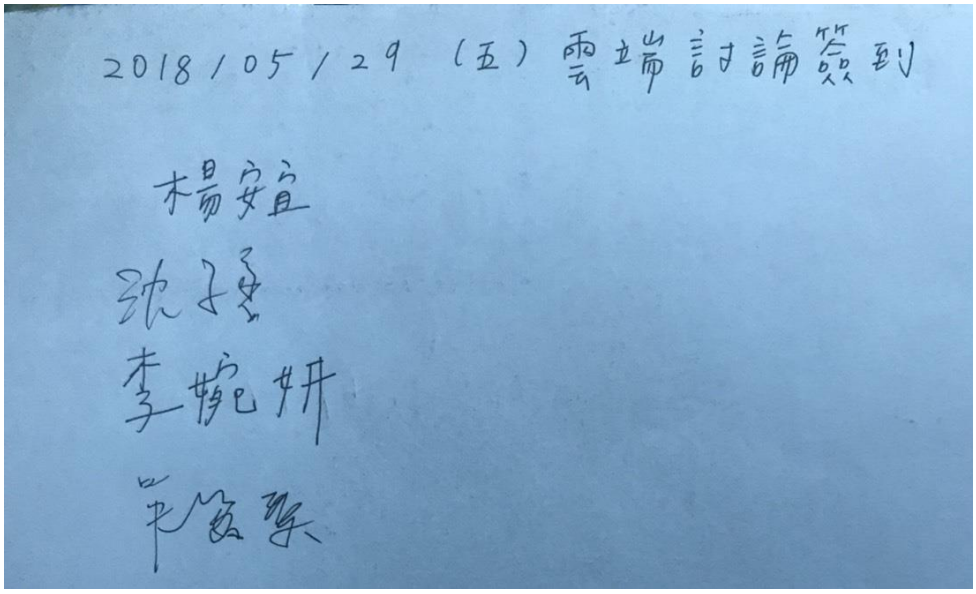
二、計畫成果

(一)整體計畫執行狀況描述：

資料碗給導致之後工作不太流暢，但跟組員深切討論後有所改善。

(二)成效說明與實際產出：

第二次開會簽到表。



(三)自主學習歷程：

三、學習心得

安宜：

我覺得這堂課安排的自主學習對我來說受益良多，不只能更加了解雲端計算相關知識，還能與很多朋友共同討論一個題目然後深入研究。在討論的同時，我們遇上了很多困難，比如有時候組員們常常會叫不動，這時身為組長的我很有警覺心，於是我就一直約大家出來討論，在前一天提醒大家要餓開會，而在這次開會時就先制定好下次的開會時間。雖然常常在群組裡只有我一個人在發話，而且常常被已讀。但只要不屈不撓的表示出積極的樣子，我想組員們都是看在眼裡的。最後，大家都有被我感化，拿出心力看待這份報告。我覺得很感動。

經過這堂課，我覺得我學到的不只是雲端計算的相關知識；還學到了自主學習，靠自己的力量去釐清答案，去學習知識；更學到了許多帶領同學、與人相處的技巧。我覺得緣分這個東西說來很奇妙，特別是在大學階段，來自各個地方的人，也讓我感受到不一樣的衝擊。很感謝有這次的機會與緣分讓我學習到這些。

子齊：

藉由這次自主學習的機會讓我學到很多很多，在其間我覺得最有收穫的一點是當我馬上有疑問的時候，我必須自己上網找出相關文獻然後自己去理解，而不是一直去詢問教授或者是同學，我覺得經過這次經驗之後，對我的畢業專題有很大的影響，讓我不會那麼擔心了，而且在上網尋找疑惑時，盡量用英文當關鍵字去尋找，相關文獻會多很多，剛好同時又讓我增加我的英文能力，覺得自主學習計畫真的是一舉多得。然後在尋找相關的文件的時候發現一件有趣的事情，只有不是資工系的論文，大多都不會有實作的部分，幾乎都是介紹工具，

這也正是林教授一而再再而三強調的重點，了解工具大家都會，重要的是需要自己去實作，才能知道自己有沒有透徹的了解，而不是懵懵懂懂。

盈泰：

在上完這學期的雲端計算課程後，我對這方面的了解又更多了，不再是簡簡單單的以為雲端只是提供擺放資料的地方。而且，雲端計算這個概念以後肯定會掀起大風暴，所以現在把它搞好一定有幫助。這個東西的好處多到講不完，但是壞處也不是沒有。用個比喻：把雞蛋放在同個籃子。一旦供應商掛掉，那是不是全部系統也會掛掉。

再來說說我這學期的感想，我覺得這門課蠻有趣的，帶領我們進入這個大世界裡，而且還有那個專題，讓我們一起討論一起研究，互相討論對雲端運算的看法，而且也交到一些朋友，就還蠻不錯的總而言之，這個課程讓我收穫很多

婉妍：

在這次的自主學習中，從一開始的閱讀論文、跟老師討論題目方向，到隔週跟組員們討論論文內容以及了解相關的背景知識，再跟組員們一起整理資料、討論簡報的報告內容，整件事情對我們來說是非常少有的經驗。平常的課程中大多都是個人的作業考試，很少有分組協力完成一個報告，也因為這次的經驗，讓我們更懂得如何在團隊中合作。而在這次的期末報告中，我們這組的題目是「節能雲端運算分配方法」，當然也因為此次的報告，去搜尋並了解了不少雲端運算方面的技術與應用，才知道原來在運作的背後，有這麼多種演算法，使得 Virtual Machine 的資源得到更好的配置。在整體運作過程中，如何節省時間成本並增進整體運算的效能，都是一門很大的學問，這也讓我了解到原來還有這麼多知識是需要去探索且學習的，非常有趣，並令我更有動力去了解新知，是一次難能可貴的經驗。

馥鈴：

這次報告準備的時間不太夠，導致我們沒什麼時間可以實作。但我們也從論文中學習到了跟課堂上不太一樣的知識。以前覺得雲端很便利是因為不需要帶隨身碟，只要在有網路的情況下就能得到很多的資料，相較於隨身碟，不用怕資料會不見。後來修到這門課後，才知道後台做了很多我都不知道的事情。雲端的出現為我們帶來很大的便利，但對於台灣的產業也造成不少的衝擊，像微軟當初會選擇在台灣設廠是因為台灣的水電便宜且有保障，但面對雲端的興起，微軟仍受到些微的衝擊。隨著科技與產業不斷創新，雖然雲端的興起對台灣來說算不利，但舊有的產業能夠維持到現在代表已具有面對新興產業興起的能力，不過舊有的產業仍須調適並創新來找到屬於自己的市場。

四、結論與具體建議

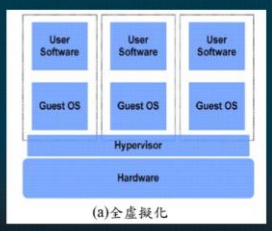
- 講稿應順過多次
- 工作要與表訂日期同步
- 要再開會前一天就被齊所有資料，不然要事先告知
- 資料應早點丟到群組，好讓後面的工作迅速開始。
- 報告當天應全數組員到場，都說好不遲到了。

成果報告

雲端運算之節能資源分配方法

404261397楊安宜
404261309李婉妍
404261593沈子齊
404261505陳馥鈴
404262248吳盈泰

全虛擬化



(a)全虛擬化

全虛擬化是未經過作業系統和應用程序上來運行的一項虛擬化技術。以基礎硬體設施作為底層架構，再往上架構虛擬化平台，就是圖片中的hypervisor，模擬出硬體來供給運算使用。這樣一個完整的虛擬化系統可以運行多個Guest OS。這樣使用者就可以很迅速地建立 Guest OS 不用再多餘的安裝和設定。

動機

由於系統匯集了數量龐大的資訊設備，因此電源消耗是一個相當重要的議題。我們要探討的是在雲端運算上資源分配的問題為了降低電源消耗以及分析工作任務執行

半虛擬化

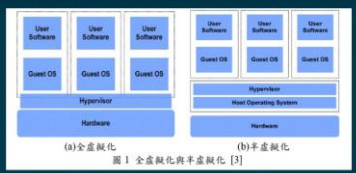
(b)半虛擬化

半虛擬化它無法完全模擬，必須在基礎硬體設施上安裝 Host OS 才可進行模擬，大家可以看投影片。它必須在Host OS上來安裝虛擬化平台(hypervisor)軟體，藉由此虛擬化平台軟體才可產生模擬Guest OS 來運行使用。

虛擬化

雲端運算加入了虛擬化技術的概念，使得整體運算效能提升並降低所需的成本開銷。

全虛擬化與半虛擬化的比較



(a)全虛擬化　(b)半虛擬化
圖 1 全虛擬化與半虛擬化 [3]

虛擬化技術下有兩種不同實現的模式。分別為：虛擬機監視器（Virtual Machine Monitor，VMM），給予虛擬機器一個運行的環境，包括提供虛擬資源（CPU、記憶體。
另一種是虛擬化平台（Hypervisor），負責虛擬機器的管理，不同於 VMM，它是直接 運行在硬體之上，因此績效上較好

虛擬化技術分兩類

1. 全虛擬化(Full virtualization)
2. 半虛擬化(Para-virtualization)

圖 2：虛擬化一個運行環境 VMM 示意圖　圖 3：一個虛擬化平台示意圖

虛擬化技術下有兩種不同實現的模式。分別為：虛擬機監視器（Virtual Machine Monitor，VMM），給予虛擬機器一個運行的環境，包括提供虛擬資源（CPU、記憶體、硬碟。
另一種是虛擬化平台（Hypervisor），負責虛擬機器的管理，不同於 VMM，它是直接 運行在硬體之上，因此績效上較好

虛擬化技術

Maximum utilization allocation

這是依據某一工作任務之所需使用率，並檢查每個實體機器的資源使用剩餘比例來作為分配考量，選擇剩餘比例最小且能容納該任務之實體機器並分配資源給此任務執行，主要目標是希望能將工作執行儘可能集中，以節省機器之使用。

綠能雲端運算

虛擬化技術就是將伺服器、儲存空間等運算資源予以統合。一臺採用虛擬化技術的伺服器，可以創造出一個虛擬化的環境，同時執行好幾臺宛如實體伺服器一般的虛擬伺服器。透過虛擬化管理工具，在幾分鐘內就可以建立一臺虛擬伺服器，而其運算資源，如處理器速度、記憶體容量、硬碟儲存空間，則可任意配置。

Job splitable allocation

以當時情況下選擇 PM 高使用率的依序將被拆解的工作任務逐一的分配讓其使用率達到 100% 再將剩餘的工作再次分配至次高使用率的 PM 上，依此類推，直至分配完所有的工作使用率為止。

目的

針對雲端運算的服務層級來進行資源分配的改善，藉由這樣的方式進而達到綠能省電的目的，一個雲端中心若能達到節能省電且達到處理效能的提升，不但讓使用者在使用提供的服務時很放心的去使用供應商所提供的服務，且供應商也能達到資源使用率的提高兼具滿足使用者的服務品質要求，同時亦能省下很多電力的開銷成本。

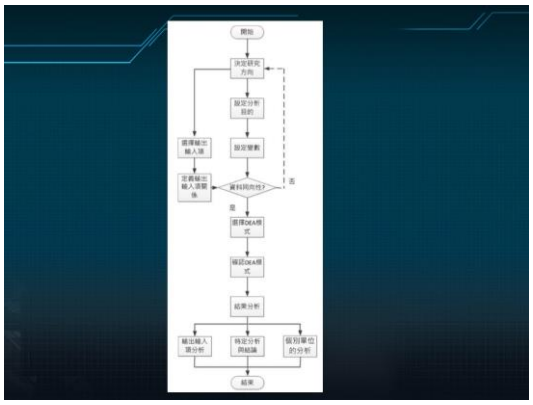
DRS（Dynamic Resource Scheduler）

可在已彙整為邏輯資源池的硬體資源叢集之間動態配置並平衡運算資源，DRS 亦能夠持續監控各個資源池使用率，並能夠反映商務需求以及不斷改變的考量因素的預先定義規則，在虛擬機之間彈性地配置可用資源。

分配方法

資料包絡分析法 Data Envelopment Analysis,DEA

是根據經濟學中柏拉圖最佳境界之觀念所提出的相對效率評估模式。所評估出來的效率值是在客觀環境下對受評單位最有利的結果。DEA是一種可以同時參考多項輸出輸入的分析方式，分析結果會是各項目之間的相對關係，依照 DEA 的特性，在資源分配問題中，將考慮多種可能影響的各種參數，分析結果將提供每位使用者及供應商的最合適的資源分配方針。



結語

天下沒有一種十全十美的分析方法，每一種計量分析技術的目的與應用情況都不太一樣，我們必須視研究目的與條件去決定要採用哪種方式，或搭配哪些方式，使研究的信度與效度達到最大。同時必須知道所使用的分析方式的限制與缺點，以便在判讀結果時有所注意，做出正確的結論。

優點

DEA對實務管理很有幫助，一來它可以化繁為簡，將眾多成效指標統合分析，並用效率分數呈現出來，讓決策者一目了然，而且DEA更提供了明確的標竿對象與經營或改善努力的目標與方向，這是其它很多計量分析方法所沒有的優點。

參考資料

- http://s90304a123.pixnet.net/blog/post/37324172-%E9%9B%B2%E7%AB%AF%E9%81%8B%E7%AE%97%E5%88%B0%E5%BA%95%E8%A9%B2%E5%A6%82%E4%BD%95%E7%AF%80%E8%83%BD%E5%BD%A2%E6%88%90%E7%B6%A0%E8%83%BD%E7%94%A2%E6%A5%AD%3F
- http://140.127.82.166/retrieve/18832/13.pdf
- https://www.ithome.com.tw/article/93014
- http://thchou.blogspot.com/2009/03/data-envelopment-analysis-dea.html

缺點

由於DEA所得到的效率分數是相對的數值，不是絕對的數值，因此DEA的結果敏感度可能很高，萬一我們的資料中有一個錯誤的異常的偏離值(outlier)，就可能對結果產生很明顯的影響；

謝謝大家

2.3. 第三組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：Hmfs - Efficient Support of Small Files Processing over HDFS

二、組別：第三組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：

姓名	工作內容	聯絡方式 (E-mail)
林明嫻	Part. 5	
張庭瑄	Part. 6	
黃政瑋	Part. 4	
江承昊	Part. 1, 2, 7 及統整	howard082921@gmail.com
蘇泰維	Part. 3	

計畫之動機與目的：與組員合作一同解析論文內容，再將論文製作成簡報，專題過程中了解 Hmfs 的概念、設計以及運作方式。

進度規劃

週次	規劃內容
第一週	各自找尋專題 統整意見並討論決定
第二週	組員各自閱讀兩篇論文
第三週	投票最終決定專題論文 並分配工作
第四週	組員各自製作自己分配的論文(簡報)部分
第五週	統整 並完成完整簡報

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期： 2018 年 5 月 15 日(二) 10:30~11:00

紀錄人：

序號	工作事項	註記(心得/成果/問題 等)
1	與老師討論	因內容不合邏輯 老師不建議使用該論文
2	詢問建議論文方向	老師提供兩篇論文能做為參考
3	決定論文	組內各自閱讀 5/29 決定最終使用論文

待辦事項/待解決問題/筆記欄

無。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期： 2018 年 5 月 29 日(二) 10:00~10:40

紀錄人：

序號	工作事項	註記(心得/成果/問題 等)
1	決定論文(Clustering…/Hmfs)	Hmfs
2	分配工作	按論文 Part 分開製作 統整後進行討論
3	繳交期限	6/10 12:00

待辦事項/待解決問題/筆記欄

無。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期：107 年 6 月 15 日

一、課程基本資料

(一)自主學習計畫主題：Hmfs - Efficient Support of Small Files Processing over HDFS

(二)組別：第三組

(三)學生姓名：林明嫻, 張庭瑄, 黃政瑋, 江承昊, 蘇泰維

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：

在尋找論文，檢查內容時花費較多時間，使其他部分在時間有些緊急。透過組員各自解讀，製作簡報，完成了論文講解簡報，並向同學以較容易了解的方式進行講解。

(二)成效說明與實際產出：

五週內完成了一篇論文講解簡報。簡報內對於論文各部分都有以中文與圖表解釋，透過簡報能清楚了解 Hadoop 的問題，Hmfs 的設計、運作方式及其效益。

(三)自主學習歷程：

尋找論文，討論決定，解讀論文，分開製作簡報，進行最後討論並上台報告。

三、學習心得(每位學生提供至少 300 字的學習心得)

林明嫻：

一開始我們選的論文是關於 Hadoop 的研究與改善，選擇的理由是因為它跟我們上課的內容有很密切的關係，但後來與老師討論後發現那篇論文似乎有觀念不太正確的一些部分，因此老師推薦了我們另外兩篇有關在 HDFS 中處理大量小檔案的英文論文，經過討論後，最後我們決定選擇 HMFS 這篇論文。

原本以為我讀英文論文應該還可以，因為我英文算不錯，後來發現還是稍微有點吃力，一些專有名詞都需要查一下，也還好我負責統整的部分不算太難，PPT 也用兩個流程圖簡潔的顯示。

報告當天其實蠻緊張的，因為看到老師都會追加問報告的人更細部的問題，感覺如果是我的話有可能答不上來，結果果然，因為對論文整體沒有研讀的那麼仔細，被老師問倒了，覺得有點慚愧，以後要更加認真仔細才行。

覺得這個自主學習的計畫很好，它讓我去閱讀平常不會主動去看的論文，並且使我發現其實論文並沒有想像中艱深，稍微花一點精力就可以讀懂，學到很多。

張庭瑄：

第一次參與學校的自主計劃，一開始摸不清頭緒，也不知從哪一種主題下手選擇論文，經過一番的討論，選擇了一篇國防大學的 Hadoop 雲端運算平台的評估與改善，但更進一步和老師討論時，發現了此篇論文有需多不合理的地方，且篇幅冗長不詳細，最後改選成老師所推薦的論文-HMFS，此篇論文探討小檔案是否適用於 HMFS，並且功效勝過 HDFS，HMFS 藉由非同步寫入機制和預先取回連結(prefetching)，以及快取策略(caching)，經過多次的實驗與必較，可以得知最後的結果，HMFS 比 HDFS 在儲存和取得小檔案有較快速的速度。在一整個自主計劃的過程中，長達三個星期，按照著進度，首先選擇論文，並討論出最適合的一篇，然後開始分配工作，做出 ppt，最後大家負責報告各自的部分，組員們也很盡責的完成自己的章節，上台報告完後，我也放下心中的大石頭，完成了這堂課重要的自主報告。

黃政瑋：

經由這次自主學習計畫，令我體會到自主學習的重要性，很多東西課堂上不一定會講道，或是輕輕帶過而已，透過自己自主學習，去網路上尋找相關資料，或者閱讀相關論文，就可以得到更深入的知識。這次我們這組透過老師推薦，選了一篇關於 HDFS 如何處理 small data 的論文，透過這篇論文，讓我知道了 HDFS 中間還可以用一個中介軟體 Hmfs 來幫忙處理問題，真的令我大開眼界。

自主學習還有一點很重要的是團隊合作，每個人負責一部份，做 PPT，最後上台報告，每個人都是這份報告很重要的因子，唯有大家一起合作、討論，才能收穫滿滿。

這個計畫其實就跟翻轉教室一樣道理的，之前大一大二有幾堂課就是用這種翻轉教室的方式，讓我們自己去學習，我覺得這樣的計畫是還不錯的，有蠻有趣的，也因為這次計畫，累積了一些經驗，未來再遇到這樣的事以後就不會手忙腳亂了。

江承昊：

這次的自主學習計畫，是我第一次嘗試解讀英文論文。許多陌生的專有名詞讓了解英文論文比想像中困難和費力，也由於製作的時間不多，過程有些匆忙，並沒有完全的了解論文每一部分的內容，當老師問到較為深入的部分時，沒辦法準確的應答。以後應該要更仔細的研讀論文。

這次的計畫在團隊合作的方面我也學到了很多，由於每個人都要解讀整篇論文十分耗神和耗時，我們決定每個人負責一部分，最後再將個人製作的簡報合併。比起孤軍奮戰，團隊合作使作業更有效率，也更能以各個面向解讀論文。每個組員用心完成自己的部份，並成功上台報告，讓我了解了合作的重要。

蘇泰維：

這次的自主學習計畫讓我更了解 HDFS，知道其中的問題和解決方法。

我們這組這次找的論文是用英文寫的，一開始有許多單字看不懂要查字典，但看著看著也能透過前後文了解大概的意思。透過一個中間程式-Hmfs 解決 Hadoop 小程式形成的效率低下的問題。我覺得這次自主學習最困難的還是一開始尋找論文和決定主題的部分，要從許多論文之中找到與

雲端課程相關的，不僅要了解論文內容，同時還要了解課程內容才能找到。

我認為透過這次的學習計畫，除了在課程方面，也在合作方面獲得很多收穫。

四、結論與具體建議

透過五週與組員共同學習，第一次解讀英文論文。除了在課程方面深入接觸 Hadoop、HDFS 和 Hmfs，同時也學習團隊合作。但五週的時間有些吃緊，沒辦法對於每個部分都透徹了解，也缺少與組員的會面討論時間，希望下次參與計畫時能有更充裕的時間。

成果報告

Hmfs

Efficient Support of Small Files Processing over HDFS

1. Introduction

HDFS (Hadoop distributed file system)

- Hadoop的分散式檔案系統
- 目的是對大量資料進行分析
- 處理大量小型文件時，會占用大量內存，且效率降低

3

Hmfs (Hadoop-based middleware for file system)

- 中介軟體 (Middleware)
- 將儲存索引數據在內存中，節省HDFS NameNode的空間
- 透過Asynchronous Write Mechanism來優化HDFS上的文件上傳作業
- 使用Prefetching和Caching Strategies來提高文件下載速度

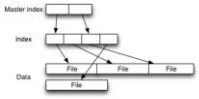
4

2. Related Work

HAR Files (Hadoop Archives)

- 在HDFS上構建一個層次化的文件系統
- 讀取HAR Files需先讀取兩層index文件後才能進行文件本身的讀取，效率不高

HAR File Layout



6

Sequence Files

- 使用filename作為key，file contents作為value
- 支援壓縮和解壓縮
- 將小文件轉換成Sequence Files速度較慢

SequenceFile File Layout

7

3. Small Files Problem in HDFS

Download

- T1 : 使用者發送存取請求給NameNode的時間
- T2 : NameNode在記憶體中尋找檔案的metadata的時間
- T3 : NameNode回傳檔案位置給使用者的時間
- T4 : 使用者從DataNode存取資料的時間

$$T_n = \sum_1^n (T_1 + T_2 + T_3 + \sum_1^m T_4)$$

9

Download

- T4a : 使用者傳送blockId到DataNode的時間
- T4b : DataNode得到blockId以及從硬體中取得檔案的時間
- T4c : DataNode回傳檔案給使用者的時間

$$T_4 = T_{4a} + T_{4b} + T_{4c}$$

10

Download

$T_n = \sum_1^n (T_1 + T_2 + T_3 + \sum_1^m T_4)$, $T_4 = T_{4a} + T_{4b} + T_{4c}$

檔案的大小小於Block(64MB)

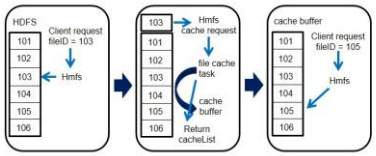
m=1

$T_n = \sum_1^n (T_1 + T_2 + T_3 + T_4)$

11

檔案預先存取及快取

- 預設檔案間無關聯並依照上傳時間合併
- 流程：



16

Upload

- T1 : 使用者發送存取請求給NameNode的時間
- T2 : NameNode在創建檔案的metadata的時間
- T3 : NameNode回傳data output stream給使用者的時間
- T4 : 使用者傳送資料到output stream以及在DataNode儲存資料的時間

$T'_n = \sum_1^n (T'_1 + T'_2 + T'_3 + \sum_1^m T'_4)$

12

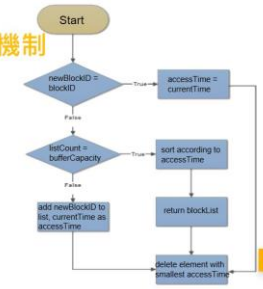
Buffer 替換機制

變數：

blockList<blockID, accessTime>,

newBlockID,

bufferCapacity



17

Upload

$T'_n = \sum_1^n (T'_1 + T'_2 + T'_3 + \sum_1^m T'_4)$

檔案的大小小於Block(64MB)

m=1

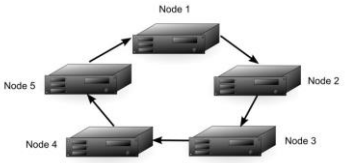
$T'_n = \sum_1^n (T'_1 + T'_2 + T'_3 + T'_4)$

13

6. Experimental Evaluation

4. The Design of Hmfs

Experimental Environment



19

5. Optimization Strategy

Memory Usage Analysis

由於較小的文件可以儲存在Hmfs的緩衝區中，因此在這三個之中較為勝出。

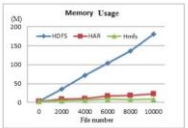


Fig. 5. NameNode memory usage of HDFS, HAR, and Hmfs

20

Memory Usage Analysis

Download :

下載的速度相似

Upload :

HAR花較多空間在儲存Metadata的個別資料

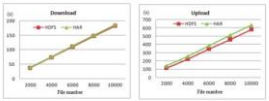


Fig. 6. Download and upload time of HDFS and HAR

21

Upload Efficiency Analysis

ARTPF (Average Response Time Per File) :

Hmfs為非同步寫入機制

AUTPF (Average Upload Time Per File) :

由於AUTPF必須同時考慮檔案上傳時間和檔案合併時間

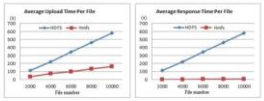


Fig. 7. Upload time of HDFS and Hmfs

22

Download Efficiency Analysis

Sequential Download

Single File :

利用非同步來快取資料，影響單一檔案的回復速度

Repeat 5 Times:

在Hmfs中，若為相同檔案，預先取回策略可以加快下載

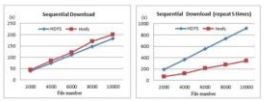


Fig. 8. Sequential download time of HDFS and Hmfs

23

Download Efficiency Analysis

Random Download

因為預測下一個檔案很困難，無法使用預先取回策略，

所以若要下載的檔案較大時，相對來說會較沒效率。

Fig. 9. Random download time of HDFS and Hmfs

24

7. Conclusion

Conclusion

- Hmfs能夠支援四種基本操作 - upload, download, update, delete，即使HDFS更新也可持續提供支援
- Asynchronous Write Mechanism使其可快速回應客戶端
- Prefetching和Caching Strategies提高了讀取文件的效率

26

2.4. 第四組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：A Kubernetes-Based Monitoring Platform for Dynamic Cloud Resource Provisioning

二、組別：第四組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：(可以個人或團體方式執行，至多 5 人)

姓名	工作內容	聯絡方式 (E-mail)
錡靖	技術實作	gin5020gin@gmail.com
王彥凱	投影片	z86125@yahoo.com.tw
鍾尚值	查資料	stj44783@gmail.com
林彥臻	找論文	jackt5566@gmail.com
黃湧文	論文整合	myong3@yahoo.com.tw

計畫之動機與目的：

1. 計畫內容與行動方案：

- 先上專題網站找適合的題目
- 找網路或手邊書籍輔助
- 盡早開始製作 PTT

2. 進度規劃

週次	規劃內容
第一週	搜尋論文
第二週	閱讀論文
第三週	討論論文
第四週	做投影片
第五週	準備講稿

七、預期效益及具體檢核方式：

大家能順利報告完成自己負責的部分之外，也要瞭解 kubernetes、docker 運作方法，這樣這次報告才有意義，每個禮拜定期討論&互相督促進度

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期： 2018 年 5 月 22 日(二) 10:30~11:00

紀錄人：王彥凱

序號	工作事項	註記(心得/成果/問題 等)
1	與教授討論題目	瞭解報告該著重在哪部分
2	組員間討論報告內容	集合大家回家查的資料

待辦事項/待解決問題/筆記欄

報告著重於:

- 介紹原本的 kubernetes
- docker container
- 這份論文 kubernetes 擴充的項目

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期： 2018 年 5 月 29 日(二) 10:30~11:00

紀錄人：鍾尚值

序號	工作事項	註記(心得/成果/問題 等)
1	整合論文內容	
2	開始做投影片	

待辦事項/待解決問題/筆記欄

- 分配工作
- 決定投影片內容大綱

  Docker

  Kubernetes

  動態資源監控&配置

  結論

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期：2018 年 6 月 19 日

一、課程基本資料

(一)自主學習計畫主題：

(二)組別：第四組

(三)學生姓名：錡靖、王彥凱、林彥臻、鍾尚值、黃湧文

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：

計畫進行順利，沒甚麼 delay，投影片也在報告一個禮拜前就完成

(二)成效說明與實際產出：

我們從這篇論文中學到很多有關於 docker 跟 kubernetes 的技術，製作投影片技術也有所提升，自我學習過程中，我們也成功架出了 kubernetes 叢集

(三)自主學習歷程：

因為升上大三有專題實驗室，大家很容易碰面，專題實驗室也是個很適合來討論與自我學習的地方，我們經常利用零碎的課餘及午休時間在專題教室討論，所以計畫進行得比預期順利很多。

三、學習心得(每位學生提供至少 300 字的學習心得)

404261153 資工三甲 錡靖:

此次的自主學習因為我們選的主題與我的畢業專題有關，所以在前面讀論文時沒有太大的困難，我們也是與論文作者使用同樣的工具，在之前就已經有所接觸。這次的論文研究進一步去探討 Kubernetes 系統的進階應用，是我在之前所不了解的，我們在研究過程中也清楚的去了解到 Kubernetes 這個工具的內部架構，而不是只是使用它的功能，除此之外，我們也了解如何去完成一個簡單的實驗，讓這個系統不是只有他自己預設的功能，而是利用其他的輔助工具來讓系統使用我們自訂的算法來運作。

比起聽老師講解技術的這種教學模式，自主學習讓我們更清楚的知道我們需要了解哪些東西，如果要有成果出來，必定是要經過自己的消化吸收，才能將所了解到的知識技術傳達給同學們。有時候死背硬記的去應付考試，反而學習效果會很差，考過的東西就忘，而自主學習出來的成果，起碼是自己有去親自實作體會過的。當然這些東西的前提是要熱心地去投入這個活動，如果沒有把這個活動放在心上，也只是又一次想混分數的期末報告而已。

404261268 林彥臻

自主學習對我來講並不太陌生，只有要修過郭文彥老師的課就知道他的教學模式幾乎就是自主學習的方法。不過找論文閱讀論文卻是我第一次。閱讀論文其實就是看別人做的研究實驗以及數據分析，我們小組討論後所選擇的主題是找 Kubernetes 相關的論文作為我們的主題。因為我們小組員裡有三人的畢業專題的主題就是有運用到 Kubernetes，並且有買相關的書籍學習研究，所以在讀論文時比較沒有障礙、不懂的地方會比較少，整體過程很順利，最後的成果發表也如預期完成。

其實自主學習是我滿喜歡的教學方式，除了不必早起、主要還因為有方向性、同學一起討論吸收得很快，而且因為要做報告，所以邊做的同時也能更加認識、可以好好講解報告。以小組方式進行、有的人懂得多、有人理解的較少，團隊合作下有不錯的學習氛圍。

404261531　資工三甲　鍾尚值:

從起初聽說過雲端共享平台，只知道是個網路上，提供用戶一個空間可以自由存取檔案，卻曾未去真的去了解過雲端平台的原理。我們的組員有三人的畢業專題是與雲端相關的，已經參考過相當多的資料與論文然而我並不熟悉這一塊，因此這次的報告提供給我一個機會去認識雲端，也透過自主學習的方式，尋找自己想研究的部分。雲端計算這堂課起初都只是學到了理論的部分，雖然課堂中間有其他講師來講課，但仍然只是模糊的概念，透過期末的報告讓我對於雲端有更深入的了解。

因為組員的畢業專題與這次報告主題相關，對於我來說只是個剛入門的新手，在這次報告雖然不敢說已經對於雲端有多深入的見解，但也學習到了些不一樣的知識，也從中學習到大家一起思考、討論、分工，如何在團隊中團隊合作與蒐集成員們的想法，做出一個有內容且令人看到了會清楚明瞭，會喜歡的投影片。這次報告學習到的東西，我將會用在我自己的畢業專題上，希望能呈現出更好的結果。

404261177 資工三甲 王彥凱:

起初會選擇這個題目，是因為我們專題所用的 kubernetes 跟 docker 就是與雲端技術息息相關的技術，所以就在教授提供的論文網上找有沒有使用類似技術的論文可以當報告題目，搜下去就發現原來有清大的論文是關於改良 kubernetes 的控管，覺得選擇這個題目應該可以對未來專題製作有更好的理解吧。

論文中也有對於 docker 的介紹，因為我做專題時對於 docker 著墨較多，所以報告時就負責這一部分。Kubernetes 在我做這次雲端報告前，其實是不太瞭解的，因為專題中 kubernetes 的架設主要是別的組員完成，但經過這次做報告的過程，我瞭解了 kubernetes 的基本架構與運作方式，對未來做專題會有不小的幫助。這次準備 ptt 內容與上台的講稿也是很特別的經驗，因為這次報告內容較為專業，而且主要是講給同學聽，寫內容時還要跟組員討論這樣會不會太難懂，以前報告的時候都不需要考慮這個呢。

404261103 資工三甲 黃湧文

這學期選修雲端計算概論，是因為對雲端相關技術有很大的興趣，也覺得這是一個趨勢，不會這項技術實在太落伍了。然而剛好老師提供了自主學習的機會，讓我們學生自己分組，利用幾週的上課時間，討論我們要報告的論文，打破了以往只有教授在課堂上教課的方式，我們透過自動自發的學習，可以學到的成效更大且更多，還要在課堂上發表我們所討論的論文，讓我們增加了上台報告的經驗，以及統整論文重點、內容的能力，也可以聽別組同學的報告，等於一口氣享受了許多論文的知識，實在收穫良多。

起初會選擇這個題目，是因為組員的畢業專題所用的技術有 docker，剛好有一篇論文與 kubernetes 相關，因此就選擇了此題目，了解到了 kubernetes、docker 的技術、優缺點。

如果以後還有機會可以有自主學習的機會，我一定很樂意去參與，也會推薦同學、學弟妹們一起來學習。

四、結論與具體建議

藉由此次活動，衍生學習到與課程主題相關的知識，我們懂得了自主學習的真諦，善用這次自主學習的經驗，可以活用在未來專題上

成果報告

A Kubernetes-Based Monitoring Platform for Dynamic Cloud Resource Provisioning

PART ONE Docker
傳統虛擬化
Docker
分鐘級
較弱
較多
一般為GB
一般幾十個
完全隔離
一般
啟動速度
性能
記憶體代價
硬碟使用
運行密度
隔離性
遷移性
秒級
接近原生
很小
一般為MB
單機支援上千個容器
安全隔離
優秀

第四組
404261153 錡　靖
404261177 王彥凱
404261268 林彥臻
404261103 黃湧文
404261531 鍾尚值

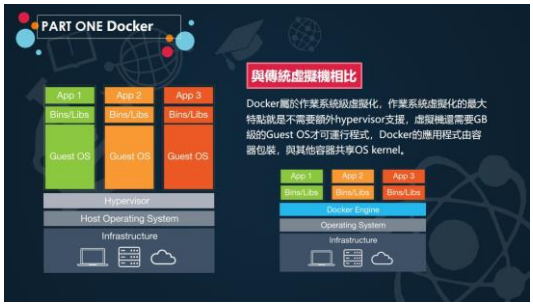
PART ONE Docker
Image
Container
Repository

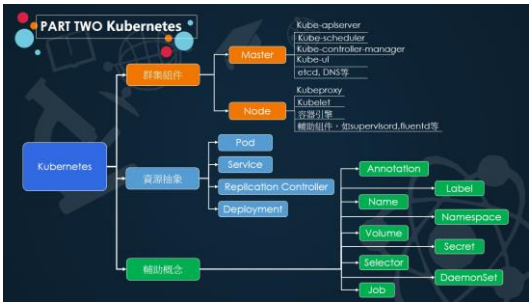
CONTENTS
1 Docker
2 Kubernetes
3 動態資源監控&配置
4 結論

PART TWO
Kubernetes

PART ONE
Docker

PART TWO Kubernetes
優秀的API設計
可拓展性佳
自動化管理
支援多種環境
kubernetes

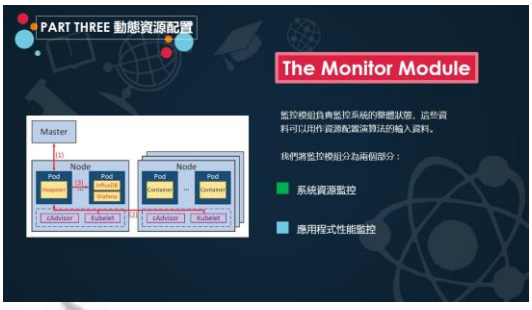

PART ONE Docker
與傳統虛擬機相比
Docker屬於作業系統級虛擬化，作業系統虛擬化的最大特點就是不需要額外hypervisor支援，虛擬機還需要GB級的Guest OS才可運行程式，Docker的應用程式由容器包裝，與其他容器共享OS kernel。
App 1
App 2
App 3
Bins/Libs
Guest OS
Hypervisor
Host Operating System
Infrastructure
Docker Engine
Operating System


PART TWO Kubernetes
Kubernetes
群集組件
Master
Node
Kube-apiserver
Kube-scheduler
Kube-controller-manager
Kube-ui
etcd, DNS等
Kubeproxy
Kubelet
資源抽象
Pod
Service
Replication Controller
Deployment
輔助概念
Annotation
Label
Name
Namespace
Volume
Secret
Selector
DaemonSet
Job

PART TWO Kubernetes
POD:
一個或多個容器組成一個POD，是Kubernetes系統中最基礎的部署單位，共用命名空間、儲存卷
Kubelet:
是Node上最重要的工作程式，負責具體運算工作與監控、維護POD
cAdvisor:
為一個open source容器資源監控工具
etcd:
分散式資料庫，用於儲存叢集所有節點的資源的狀態

PART THREE 動態資源配置
Monitor
The Monitor Module
Analyze
The Data Aggregator Module
Plan
The Resource Scheduler Module
Execute
The Pod Scaler Module

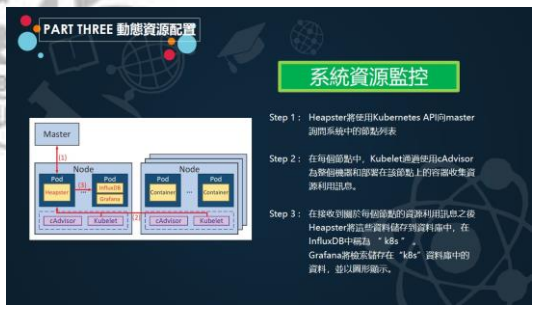
PART TWO Kubernetes
Scheduler:
負責主要資源調度工作，當收到資源後，依照策略選擇合適的NODE執行POD
API Server:
作為REST API伺服器端，負責接收來自使用者，其他元件的請求與更新etcd裡的資料
Controller-manager:
提供多種控制器服務，監視叢集狀態

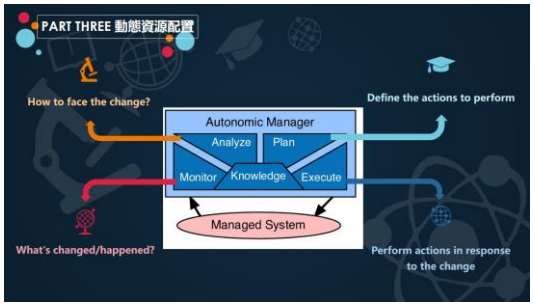

PART THREE 動態資源配置
The Monitor Module
監控模組負責監控系統的整體狀態，這些資料可以用作資源配置演算法的輸入資料。
我們將監控模組分為兩個部分：
系統資源監控
應用程式性能監控

PART TWO Kubernetes
Service 01
Replication Contoller 02
Deployment 03
Horizontal Pod Autoscaler 04
服務
由若干Pod組成的對外部提供某種功能的抽象概念，不隨Pod改變而變化，附帶唯一固定的存取路徑。
複製控制器
負責啟動Pod，並維護其預訂運行之狀態。是使用者控制管理Pod的工作。
部署
建立Pod，可根據參數自動建立管理Pod的複製控制器。
橫向Pod擴展器
根據Pod的使用率自動調整一堆部署裡面Pod的個數，保障服務之可用性。

PART THREE 動態資源配置
系統資源監控
為了實現系統資源監控
系統中使用了三種現有開源技術:
Heapster：是Kubernetes基於容器的叢集監控工具，他收集每個節點的資源利用資訊，將收集到的資訊導入後端儲存。
InfluxDB：是一個用來儲存時間序列數據的資料庫。在系統中，通過使用InfluxDB來儲存Heapster收集的資源利用資訊
Grafana：是一個用於時間序列資料可視化的應用程式。它將從資料庫查詢時間序列資料，然後以圖形顯示。

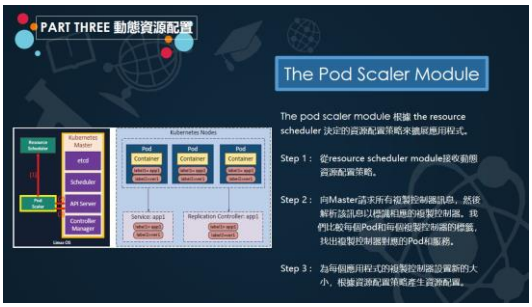
PART THREE
動態資源配置


PART THREE 動態資源配置
系統資源監控
Step 1：Heapster將使用Kubernetes API向master詢問系統中的節點列表
Step 2：在每個節點中，Kubelet通過使用cAdvisor為容器機器和部署在該節點上的容器收集資源利用訊息。
Step 3：在接收到關於每個節點的資源利用訊息之後，Heapster將這些資料儲存到資料庫中，在InfluxDB中稱為“k8s”。Grafana將檢索儲存在“k8s”資料庫中的資料，並以圖形顯示。

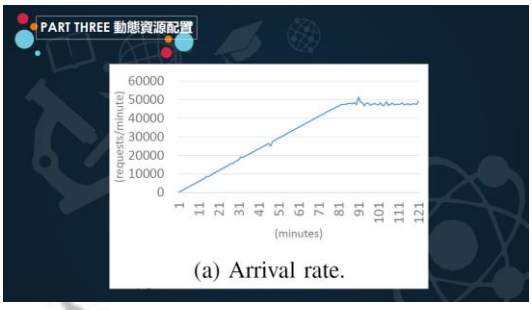

PART THREE 動態資源配置
How to face the change?
Define the actions to perform
Autonomic Manager
Analyze
Plan
Monitor
Knowledge
Execute
Managed System
What's changed/happened?
Perform actions in response to the change

PART THREE 動態資源配置
應用程式性能監控
我們將應用程式叢應時期作為應用程式性能指標。為了監視應用程式回應時間，我們在系統中安裝了Apache JMeter。
JMeter是為工作負載測試和多種伺服器的性能而設計的，使用者可以使用JMeter來分析測試應用程式的性能。
我們實作了一個名為JMeter Invoker的程式來調用Apache JMeter，並按照我們的期望來執行它的動作。

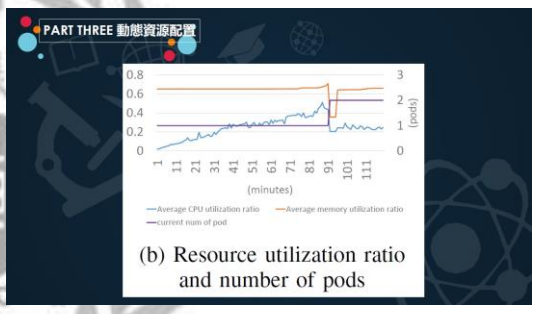
PART THREE 動態資源配置
應用程式性能監控
Step 1： 我們搜尋所有服務的當前訊息，以及從Master到整個系統的單應的當前IP位址和Port。
Step 2： 在我們的JMeter Invoker程式中，我們使用CSV資料集配置使JMeter向系統中的當前服務的IP位址發送請求。
Step 3： 我們定義了一個JMeter測試計劃文件。然後，JMeter將根據之前定義的JMeter測試計劃，向部署的應用程式發送請求。


PART THREE 動態資源配置
The Pod Scaler Module
The pod scaler module 根據 the resource scheduler 決定的資源配置策略來擴展應用程式。
Step 1： 從resource scheduler module接收最佳資源配置策略。
Step 2： 向Master請求所有複製控制器訊息。然後解析該訊息以檢識相應的複製控制器。找們比較每個Pod和每個複製控制器的標籤，找出複製控制器對應的Pod和服務。
Step 3： 為每個應用程式的複製控制器設置新的大小，根據資源配置策略產生資源配置。

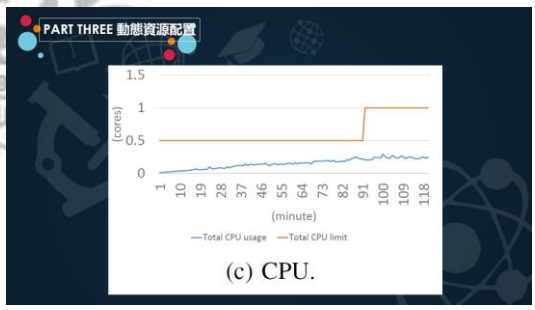
PART THREE 動態資源配置
應用程式性能監控
Step 4： 我們讓JMeter在CSV格式的指定文件中列出請求的結果。在此文件包括應用程式的URL、回應時間、回應碼等。
Step 5： JMeter完成發送請求並產生結果檔案後，JMeter Invoker程式將從JMeter結果檔案加載結果，並計算所有已部署應用程式的平均回應時間。
Step 6： 最後，JMeter Invoker程式以JSON格式將每個部署應用程式的平均回應時間輸出到一個文件中。

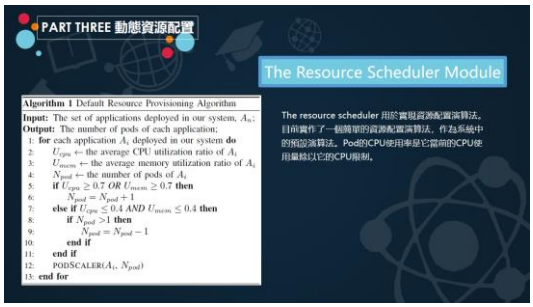

PART THREE 動態資源配置
60000
50000
40000
30000
20000
10000
0
(requests/minute)
1
11
21
31
41
51
61
71
81
91
101
111
121
(minutes)
(a) Arrival rate.

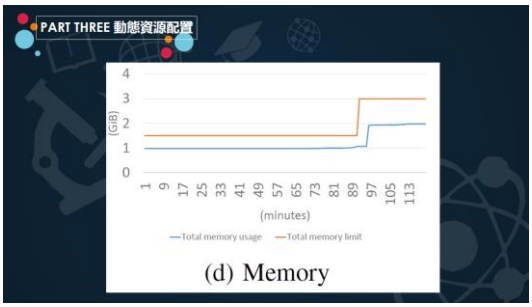
PART THREE 動態資源配置
The Data Aggregator Module
The data aggregator用於集成系統狀態(包括節點和pod)、系統資源和應用程式性能。
Step 1： 從整個系統中查詢有關節點、命名空間、容器和服務的訊息。對於每個名稱空間，我們比較每個pod和每個服務的標籤，然後記錄pod和服務的mapping。
Step 2： 列出每10秒的值，並查詢最新的值。


PART THREE 動態資源配置
0.8
0.6
0.4
0.2
0
3
2
1
0
(pods)
1
11
21
31
41
51
61
71
81
91
101
111
(minutes)
Average CPU utilization ratio
Average memory utilization ratio
current num of pod
(b) Resource utilization ratio and number of pods

PART THREE 動態資源配置
The Data Aggregator Module
Step 3： 從JMeter Invoker輸出文件中擷取應用程式性能指標。
Step 4： 將前面步驟收集的訊息集成到JSON結果文件中。集成的輸出文件分為兩部分—是系統中的伺服器訊息，另一是系統中的應用訊息。


PART THREE 動態資源配置
1.5
1
0.5
0
(cores)
1
10
19
28
37
46
55
64
73
82
91
100
109
118
(minute)
Total CPU usage
Total CPU limit
(c) CPU.


PART THREE 動態資源配置
The Resource Scheduler Module
Algorithm 1 Default Resource Provisioning Algorithm
Input: The set of applications deployed in our system, A_n;
Output: The number of pods of each application;
1: for each application A_i deployed in our system do
2: $U_{cpu} \leftarrow$ the average CPU utilization ratio of A_i
3: $U_{mem} \leftarrow$ the average memory utilization ratio of A_i
4: $N_{pod} \leftarrow$ the number of pods of A_i
5: if $U_{cpu} \geq 0.7$ OR $U_{mem} \geq 0.7$ then
6: $N_{pod} = N_{pod} + 1$
7: else if $U_{cpu} \leq 0.4$ AND $U_{mem} \leq 0.4$ then
8: if $N_{pod} > 1$ then
9: $N_{pod} = N_{pod} - 1$
10: end if
11: end if
12: PODSCALER(A_i, N_{pod})
13: end for
The resource scheduler 用於實現資源配置演算法。目前實作了一個簡單的資源配置演算法，作為系統中的預設演算法。Pod的CPU使用率是它當前的CPU使用量除以它的CPU限制。


PART THREE 動態資源配置
4
3
2
1
0
(GiB)
1
9
17
25
33
41
49
57
65
73
81
89
97
105
113
(minutes)
Total memory usage
Total memory limit
(d) Memory

真善美聖
仁
FU JEN CATHOLIC UNIVERSITY
SANCTITAS BONITAS
PULCHRITUDO
VERITAS

2.5. 第五組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：利用雲端計算加速深度學習的計算時間

二、組別：第五組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：(可以個人或團體方式執行，至多 5 人)

姓名	工作內容	聯絡方式 (E-mail)
張皓軒	上台報告	lcv31913@gmail.com
徐弘軒	投影片製作	
邱柏皓	上台報告	
游竣貿	資訊彙整	
許津凱	相關技術蒐集	a0970829528@gmail.com

計畫之動機與目的：

3. 計畫內容與行動方案：研讀論文中的相關技術，並依照時間進度規劃是否可以進行實作，因蒐集 Deep Learninig 所需的資料需要不少時間。

4. 進度規劃

週次	規劃內容
第一週	了解人工智慧起源
第二週	了解人工智慧相關演算法(CNN、R-CNN 等等)
第三週	了解何謂雲端計算
第四週	了解雲端計算如何幫助人工智慧加速運算
第五週	了解雲端計算如何幫助人工智慧加速運算

預期效益及具體檢核方式：

利用雲端計算加速對深度學習的運算時間，具體核檢方式因時間問題並未實做出來。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期：107 年　6 月　20 日

紀錄人：

序號	工作事項	註記(心得/成果/問題 等)
游竣貿	資訊整彙	因為要先了解這些相關技術，所以在整合時較困難
張皓軒	上台報告	相關技術太多，知識不足，準備期間較辛苦
許津凱	相關技術蒐集	許多專有名詞且大多數都是未接觸過的領域
徐弘軒	投影片製作	需要把大量的理論，做成較為簡短的投影片，讓大家快速理解，還蠻困難辛苦
邱柏皓	上台報告	理解和能講出來讓別人理解一項技術，是一項不簡單的挑戰

待辦事項/待解決問題/筆記欄

遇到的問題:

在閱讀這篇論文時,因為有很多專有名詞、演算法都是沒碰過的,所以要了解這篇論文前,要先查詢這些演算法,並且深入了解,也由於這篇論文結合了深度學習跟機器訓練的部分,所以也查詢了相關資訊才能將整篇論文融會貫通。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期： 107 年　6 月　14 日

一、課程基本資料

(一)自主學習計畫主題：利用雲端運算加速深度學習的計算時間

(二)組別：第五組

(三)學生姓名：張皓軒

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：

先將論文不懂的文字敘述翻譯成中文，組員每個人把論文看過後,提出自己所吸收到的資訊，再將每個人蒐集到的不同資料整合，並且透過網路、書、其他論文等做深入了解。

(二)成效說明與實際產出：

該篇論文中我們並沒有實做，只針對理論、技術部分做了解，因為僅有 3 周的時間，無法蒐集足夠的深度學習資料量。

(三)自主學習歷程：

因為老師給我們三個禮拜的時間去研究這個論文，以及找尋有興趣的主題，經過組員們討論後，覺得這篇論文感覺很有趣,但實際的去看完整份論文，發現有許多東西是我們現在沒有碰過的，所以在理解這份論文也花了不少時間，這樣讓我們深深體會了為什麼老師要給我們自主學習。

三、學習心得

游竣貿:

對於自主學習的感覺我覺得很好,因為可以讓我們學習雲端技術的相關運用,而因為挑選了本篇文章,也讓我知道了有關人工智慧演算法及 Deep Learning 這種相關的課外知識,在專題也有運用到這個工具,而對於不懂的問題也能跟組員們相互討論,整體的感覺我都覺得很好!還有可以聽到不同組別的人報告,也能了解了雲端技術在各方面的實作與應用,原本以為雲端就只是讓我們方便儲存檔案的地方,但沒想到原來他運用的範圍是那麼的廣大,且大多數也已成熟,而也認識了各種的平台服務模式像是 Iaas、Saas 及 Paas 等等……

張皓軒：

對於這次自主學習的主題，原本是想找利用雲端技術加速冗長的運算，然而在挑選論文時並未仔細篩選，導致題目偏題，雲端計算的技術成分相對少了很多，然而卻對深度學習有更深的了解，包括其演算法、運作過程等等，對於畢業專題的製作有很大幫助。而透過其他組別對於雲端計算的報告也了解到原來還可以用在這地方，像是加快編碼速度這個主題我個人就很感興趣，也十分感謝其他同學的報告，彼此互相將自己的知識做交換我認為是很好的學習方式，這三周雖沒上課但也絕無白費時間，吸收的知識量也不比上課來的少，也感謝老師與我們討論相關題目，讓我們獲得了十分有幫助的建議。

徐弘軒：

從小到大，我們的學習方式都是老師教什麼，我們學什麼。有些時候老師很認真地講了很多內容，但我們可能因為注意力不集中，或是對某些內容不感興趣，而並非完全吸收進去。

老師這次給我們這個機會來做自主學習，讓我們自己去找資料，自己去學習，然後做成報告，再跟大家分享。我們可以從中學到東西，學自己選的內容，必然會花很多的心力去進行，而其效果肯定是非常的好。其都歸功於心理因素，可能一開始我只是因為一點興趣，去學一點點，但我可能在學習的過程中，激發出更多的興趣，就學得更好了。

這次我們這組選的題目，是利用雲端運算來做深度學習，因為我們的專題也有用到深度學習這部分，感覺對我們的專題會有一些幫助，而過程中跟組員互相討論，學到了很多不同的思考方式。聽了很多其他組的報告，也讓我們了解了很多雲端技術的各種應用，像做檔案儲存、影像壓縮處理之類的內容，真的收穫很多。

邱柏皓：

平常在學校聽老師上課是被動地在接受資訊，但自主學習則是主動去找資料來自己學習，差別其實滿大的，我覺得自主學習有滿大的好處，因為可以找自己有興趣的知識來學習，再跟老師討論有問題的地方，對學習效率來說提昇滿多的。而這次的主題我們選擇的是「透過雲端計算來加速深度學習」，這個主題包含了兩種現在熱門的技術，可以學到如何把兩種技術結合在一起。雲端技術的部分問題比較簡單，理解上沒遇到太大的困難，但深度學習的部分，遇到比較多難題，畢竟是一個以前沒接觸過的領域，理解許多影像處理演算法的部分，就花了滿多時間，到後面可以理解的時候，真的很感謝許多前人整理好的重點，讓我可以順利的理解這項技術在做甚麼。總之透過這次自主學習計畫的機會，讓我對雲端計算和深度學習這兩個領域都有更深度的認識，也讓我學到如何將兩種不同的技術做結合的方法。

許津凱：

相較於我們已經習慣在平常上課老師給什麼我們就學什麼的模式，自主學習突然告訴我們自己才是學習的主人，老師只是提供資源和基本限制的角色，學什麼、怎麼學和學習進度規劃完全操控在自己，我認為這樣的教育方法會提高學習的積極性，因為當”想學什麼”變成自己的權利時，同時也會讓自己產生出較大的責任心以及更加落實的執行力。

在這堂課上老師要求選擇和雲端計算與運用相關的論文做自己的題目，雖然在選題時稍有跑題，但在自我查詢的過程中學到和 AI 相關的技術(deep learning 等)和目前瓶頸，讓我對 AI 有了更深刻的認識以及發現雲端計算與運用能讓許多技術在瓶頸上能有所突破，而在各個組別的報告中也了解到各個雲端技術的知識面和運用環境，而且也能聽到不同組別的不同見解和思路，我覺得自主學習讓我收穫良多。

四、結論與具體建議

這次的自主學習讓我們體會到了選擇正確論文的重要性，由於我們的主題在深度學習的部分占了一半的比例，所以在研讀方面就較為困難許多(專有名詞多)，也希望從這次的經驗可以讓我們之後在其他層面更加進步。

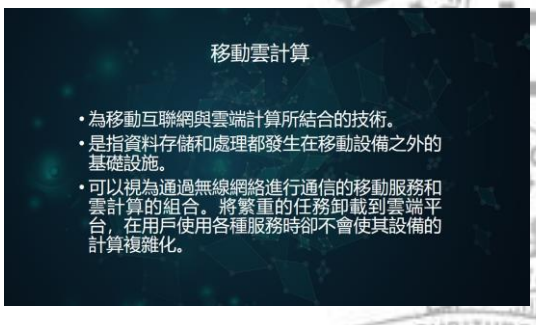
成果報告

雲端計算概論
• 組員：張皓軒、許津凱、游竣貿、徐弘軒、邱柏皓

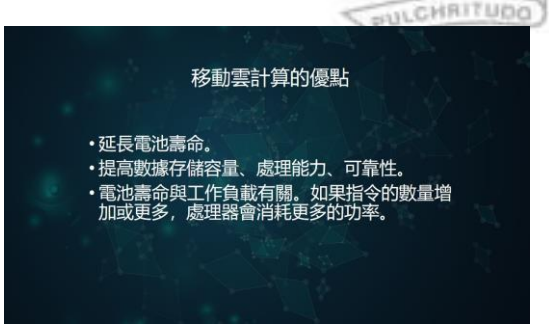
深度學習
• 分為兩個階段:
• 佈署
• 訓練
• 由一個輸入層，一個輸出層和多個隱藏層組成的神經網絡。

主題
• 利用雲端計算加速深度學習的運算機能。

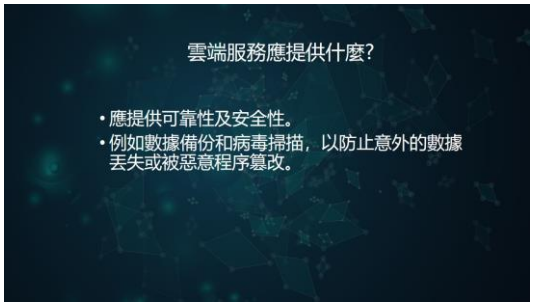
LeNet
• 出現在1960年代，因硬體設備技術不足而緩慢進展。
• 1970後誕生LeNet 。


移動雲計算
• 為移動互聯網與雲端計算所結合的技術。
• 是指資料存儲和處理都發生在移動設備之外的基礎設施。
• 可以視為通過無線網絡進行通信的移動服務和雲計算的組合。將繁重的任務卸載到雲端平台，在用戶使用各種服務時卻不會使其設備的計算複雜化。

何謂LeNet?
• 一種具有捲積和積層的深層神經網絡。
• 帶有用於數字識別的反向傳播算法。
• 最早將深度學習納入識別任務的工作。


移動雲計算的優點
• 延長電池壽命。
• 提高數據存儲容量、處理能力、可靠性。
• 電池壽命與工作負載有關。如果指令的數量增加或更多，處理器會消耗更多的功率。

OverFeat
• 通過深度神經網絡解決位置和分類問題的第一項工作。
• 使用回歸網絡和分類網絡來相應處理兩個視覺任務。

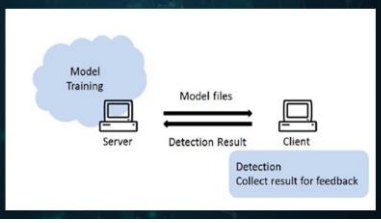

雲端服務應提供什麼?
• 應提供可靠性及安全性。
• 例如數據備份和病毒掃描，以防止意外的數據丟失或被惡意程序篡改。

R-CNN
• 將基於卷積神經網絡的方法推廣到對象檢測。
• R-CNN可以分為三部分:
• 候選人選擇
• 特徵提取
• 分類

Fast R-CNN

- 改進了R-CNN的缺點，例:
  - 測試時速度慢
  - 訓練時速度滿
  - 訓練所需的空間大

雲端計算和深度學習的結合

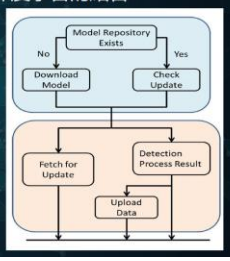


YOLO

- 具有統一檢測單個任務的功能。
- 僅使用具有24個卷積層和兩個完全連接層的單個網絡來預測圖像中的可能對象的類別和坐標。
- 可以在一秒鐘內預測45張圖像。

雲端計算和深度學習的結合

- 論文中設計了一個深度學習對象檢測系統，提供檢測服務、模型維護。



應用動機（一）

對於網絡來說，LTE網絡的中位數為5.64Mbps上傳頻寬。考慮到圖像處理，常見的行車紀錄器都有1080p分辨率，這也表明原始圖像大小超過6 MByte。在雲端上傳和處理圖像後檢索結果變得不可行。即使先壓縮文件，不穩定的無線網絡可能會阻止客戶端的一些服務。

運作過程（一）

當系統啟動時，將先執行版本檢查。如果客戶端在本地磁盤存儲器中沒有模型文件，它將連接到服務器並獲得所需的模型。否則，系統會檢查更新版本是否可用。

應用動機（二）

許多汽車都配備了相機來記錄駕駛過程中的事件。但是，設備存儲的容量是有限的，錄製視頻的許多部分是不感興趣的對象。利用物體檢測能力，行車紀錄器可以找出可能具有感興趣對象的視頻片段。這些感興趣的視頻片段可以上傳到雲平台進行進一步分析，並且可以刪除不需要的視頻片段以節省存儲空間。

運作過程（二）

- 第二階段，有兩個thread：更新thread和檢測thread。
- 更新thread會定期檢查新的模型版本。此thread僅負責模型檢查和下載。

雲端計算和深度學習的結合

基於這些事實，論文中設計了一種特殊架構，只將訓練階段卸載(offload)到雲端，並將經過訓練的模型保存到移動設備上進行識別。

客戶端的另一個可能的任務是搜集資料並回傳給服務器。從用戶端搜集的各種資料可以幫助改進深度模型。另外，服務器端可以通過分析預測結果來評估模型的性能。

安全性考慮（一）

- 考慮到安全性問題，模型不會被更新後的版本替換，除非系統重新啟動。

安全性考慮（二）

檢測thread處理資料搜集，如果可能的對象出現在輸入圖像中，檢測thread將搜集網路輸入和輸出。同時，它保留一個計數器來控制收集的文件大小。一旦計數器達到指定上限值，檢測thread將創建另一個用於上傳資料的thread ，上傳後搜集的資料將被銷毀以節省本地存儲空間。

網路傳輸

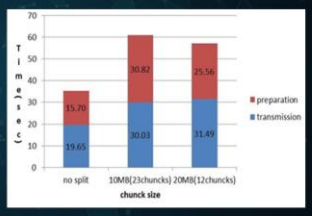
考慮到在下載過程中可能面臨網路故障，重新連接後需要重新啟動文件傳輸。限制重複傳輸的一種方法是將整個模型分割成較小文件大小的塊。以這種方式，重複傳輸的資料大小在故障時將被限制為區塊大小。另一方面，它需要更多的準備時間來傳輸。而且，文件之間的傳輸具有延遲。這兩個因素為傳輸分割模型帶來開銷。

實作

- 利用OverFeat處理滑動視窗，由兩個網路分別預測對象和位置。
- Faster R-CNN使用RPN來定位ROI 。
- YOLO將檢測統一到一個階段的任務。
  - RPN是一個全卷積網路，也就是沒有全連接層。這個網路的輸入是任意大小的圖片，輸出是矩形目標後選區域和區域得分。
  - ROI(Region of Interest)是指僅針對感興趣的部分影像作處理。

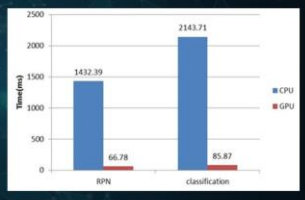
網路傳輸

- 圖為論文中分割一個大小為10MB和20MB的224MB模型並比較結果。



CPU與GPU的比較

- 圖中顯示了平均CPU和GPU的運作時間。



SANCTITAS BONITAS
PULCHRITUDO
VERITAS

2.6. 第六組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：

二、組別：第六組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：(可以個人或團體方式執行，至多 5 人)

姓名	工作內容	聯絡方式 (E-mail)
李建宏	論文閱讀、研討	d789456123851208@gmail.com
陳正宏	論文閱讀、報告	theokleintw@gmail.com
張博逸	論文選擇、論文閱讀	a101013322@gmail.com
賴志團	論文閱讀、簡報製作	alex910144@gmail.com

進度規劃

週次	規劃內容
第一週	論文方向討論、論文選擇
第二週	確認論文、研讀
第三週	研讀論文、討論
第四週	討論、提出問題
第五週	討論、簡報製作

七、預期效益及具體檢核方式：

這次是我們全組組員第一次接觸論文方面的自主學習計畫，針對雲端計算技術結合不同領域的學術性論文做研讀，在自己看不懂的地方和同學、老師討論，可以讓我們更加了解雲端計算在各領域的應用，未來想考研究所的人也能夠對於論文的架構、形式有初步了解。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期： 2018 年 6 月 4 日(二) 00:00~00:00

紀錄人：李建宏

序號	工作事項	註記(心得/成果/問題 等)
1	論文方向討論	大致確認論文選擇方向
2	論文參考	網上搜尋了 4 片論文並大致閱讀
3	論文選擇	確認論文為編碼之雲端計算研究
4	論文閱讀	詳讀論文
5	與老師討論	

待辦事項/待解決問題/筆記欄

簡報製作

提出問題

與老師討論

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期： 2018 年　 6 月　 4 日

一、課程基本資料

(一)自主學習計畫主題：編碼之雲端計算研究

(二)組別：第六組

(三)學生姓名：李建宏、陳正宏、張博逸、賴志團

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：

在論文的研討上整體還算順利，沒有太多不理解的部分，組員配合度都很好。

(二)成效說明與實際產出：

了解如何將影片壓縮技術利用分散式計算來加快速度。

(三)自主學習歷程：

整體分為三個部分，論文選擇、論文閱讀、以及提出問題討論

三、學習心得(每位學生提供至少 300 字的學習心得)

李建宏

這學期的雲端計算自主學習計畫是我們大部份同學第一次在課堂中所接觸的，對於自主學習或許仍然不習慣，但也因為這次的經歷讓我們更加了解赤主學習對於任何事物的重要性。

我們所選擇的題目是關於 H.265 影片壓縮的論文，由於現在的影片因畫質、像素等原因使檔案越來越大，在影片壓縮的加速上便是工程師要解決的問題之一。作者將原本的 H.265 技術利用分散式計算的架構來使龐大的資料量得以減輕，並藉由兩種不同的運算方式來加速。最後雖然實驗結果還算理想，但

若想真正做到同步壓縮，還有一大段的距離。

這樣的計劃讓我們不只對於論文的形式、內容能有初步概念及了解，也讓我們瞭解到，一項技術、一門課的所學知識是必須結合多方領域才能擁有好的發展，而不是只有學好一樣東西，就能夠做到很多事，因此必須更加廣泛的學習不同領域無論是程式、網路、數學等等，都是我們將來會使用到的工具。對於雲端計算而言，能應用的範圍實在太大，但也因為這樣藏著無限的發展可能。

陳正宏

這次的雲端計算自主學習選擇的主題是一篇關於 H.265 影片壓縮加速的論文。隨著現在的影片畫素越來越高，影片的壓縮傳輸便顯得十分重要，但要在不損失畫質的前提下還要有良好的壓縮率，伴隨兒來的便是龐大複雜的運算。這篇論文使用的方式即為基於現有的壓縮演算法設計一個雲端的平行化計算服務，而非僅有單機的多執行緒執行。

論文前半段的重點著重在介紹現存兩種影片的壓縮平行計算方式：slice-based 和 wave-front，這兩種方式分別有其優缺點，論文中也針對兩種方式分別實驗餅討論最後的速度優缺點。

針對平行化計算的部分我根據論文的架構圖也實作了一套 client & server 架構的壓縮程序，不過論文並未過多討論程式碼層面的問題，而且我也沒有完全套用 H.265 的壓縮方法，僅將目標針對放在資料的平行化計算處理以及資料傳輸同步的部分，所以無法比較最終的實驗數據。若再加上根據 H.265 開發的 libde256 開源壓縮函式庫，基本上應和論文作者開發的程序相去不遠。

張博逸

我們選擇的論文是把 H265 編碼的方式，用分散式雲端系統的架構，來達到減少花費時間的效果。會選這篇論文是因為它將現有的技術做改良，要能夠實作一個概念一定要有相當的水準。閱讀論文的過程中遇到很多問題，以前從來沒有想過要認識某種檔案編碼方式，更何況 H265 是在 2013 年才成為一種標準，也算是相當新的技術。

這樣的學習方式有別於傳統的「教」與「學」，老師只負責與雲端計算有關的部分，剩下的技術必須依靠自己與組員一同研究，因此更強調學習的主動性。能理解某個技術的概念是一回事，真正拿出實作的成果又是另一回事了，我覺得根據論文提出的研究產生實作的例子真的是另外一個境界。這樣的學習方式強迫學生自己搜尋並了解相關資料，我藉由這樣的經驗更了解自己的能力，對我的生涯規畫有很大的影響。

賴志團

我們這組選的是清華大學的論文，研究有關雲端 H.265 的編碼方式，新一代的 H.265 比前一代的 H.264 可以獲得 50%的壓縮效率提升，這個編碼是用在視訊與影像壓縮方面的技術標準，讓我了解到原來一張圖片和一段影片處理起來是很複雜的，現在的影像畫質越來越高伴隨著檔案越來越大，所以更需要進行壓縮讓檔案小一點，目前 4K 畫質已經越來越普遍了，H.265 最高可以支援到 8K x 4K 的解析度，壓縮的目的是增加壓縮的影像品質並降低儲存所需要的位元，但伴隨而來的是複雜的運算和大量的運算時間需求，這時候就可以使用平行計算的概念 Slices Tiles 為基準的雲端平台來達到平行計算，使用雲端平行計算技術可以獲取有效率的加速效果並達到編碼速度的提升，有關影像編碼的部分就已經很複雜了，更何況這只是雲端應用的一部份，讓我體會到雲端運算的博大精深。

四、結論與具體建議

自主學習計畫的立意非常好，但我們認為在學期中進行自主學習計畫或許並不是個好的時機，與老師討論的時間也不多，並非每一個學生都能對於自己所選的論文有充分了解。也許自主計劃若能是自發性質的才是最好的方式吧。

成果報告

雲端計算概論自主學習計畫
H.265 編碼之雲端計算

404262107 資工三乙 李建宏
404262339 資工三乙 陳正宏
404262092 資工三乙 張博逸
403261699 資工四甲 賴志憲

Encoding Order

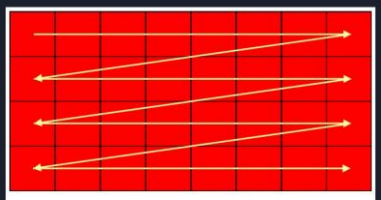


Fig. 2 Z-scan order

Reference

- LIU GUAN-YU, H.265 編碼之雲端計算研究, *Cloud Programming for the H.265 Encoder: Balanced Slice-based and Wavefront-based*, 2014

Coding Tree Structure

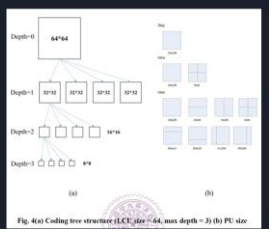


Fig. 4(a) Coding tree structure (LCU_size = 64, max depth = 3) (b) PU size

綱要

- 論文簡介
- HEVC (H.265)
  - Slice-based
  - Wave-front
- 平行化編碼方式
- H.265 Parallel Process Structure

Coding Tree Structure

- intra
  - 尋找現在的 frame
- inter
  - 尋找之前的 frames

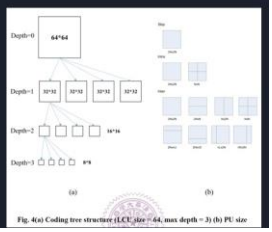
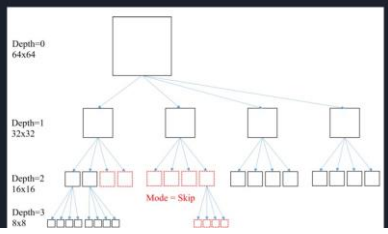


Fig. 4(a) Coding tree structure (LCU_size = 64, max depth = 3) (b) PU size

論文簡介

- 有別於原先單機多執行緒的編碼方式，探討利用雲端平行計算技術來提升 H.265 的編碼速度

Early CU Termination



HEVC (H.265)

- High Efficiency Video Coding (H.265)
- 最高支援解析度 8192x4320（8K）
- H.264 2 倍的壓縮率

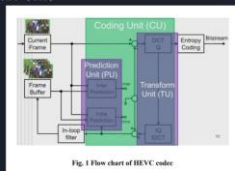
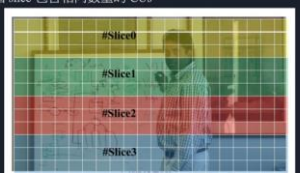


Fig. 1 Flow chart of HEVC codec

Fig. 5Coding tree structure in HEVC

Slice-based

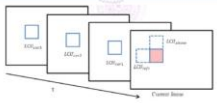
- 把一個 frame 切成一樣大的 slice
- 每個 slice 包含相同數量的 CUs



Slice-based Parallelism

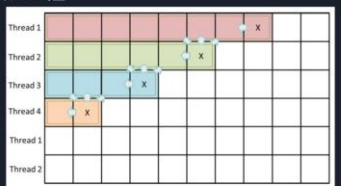
均勻各個 Slice 所需的時間：

1. 檢查 CU 是 前景 或 背景，壓縮時間放進 candidate set
2. 刪除
   a. 前景：最小值
   b. 背景：最大值
3. 分別計算前景背景的平均時間
4. 根據 CU 的預測時間分配 Slice

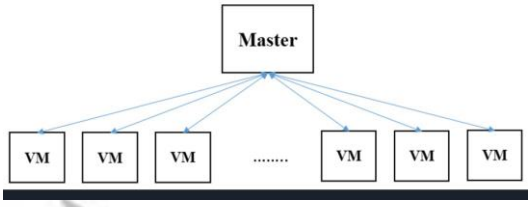


Wave-front

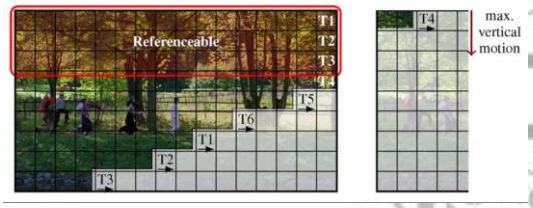
- 以 row 為主



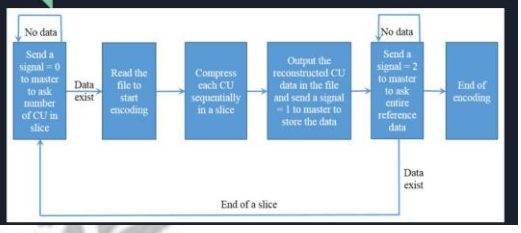
Cloud Computing



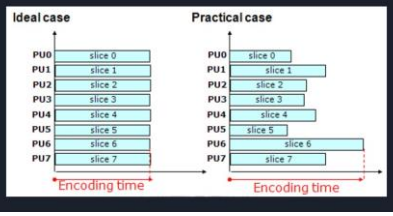
Wave-front



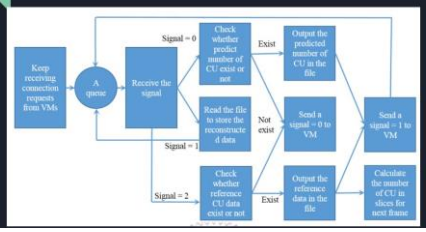
Slice-based: Client



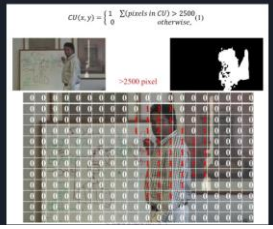
Slice-based Parallelism



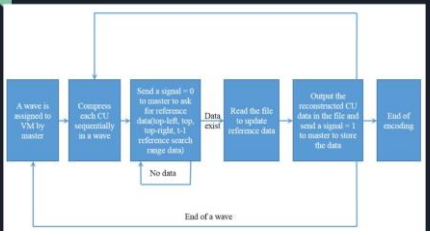
Slice-based: Server



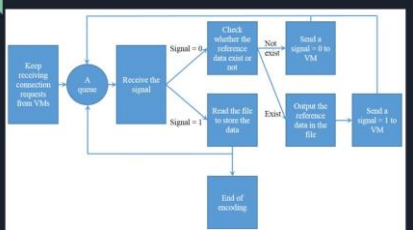
Region of Interest (ROI)



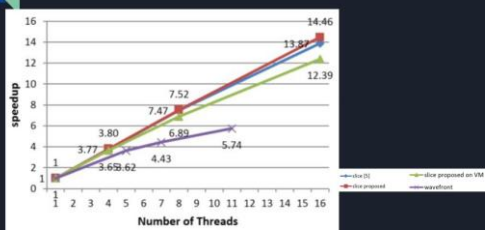
Wave-front: client



Wave-front: server



Experimental Result



Experimental Result

	CPU	RAM	Core	Software System
Master	Intel E5645 2.40 GHz	4G	1	Win7 64bits
VM	Intel E5645 2.40 GHz	1G	1	Win7 64bits

Conclusion

- 即時 8K 影片編碼還無法實現
  - (30 frames encoded in 7200 seconds)
- 升級 client & server 的硬體規格
- 縮短壓縮預測的運算時間

2.7. 第七組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：雲端運算 Hadoop 架構下離峰時間資料讀取節點管理

二、組別：第七組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：(可以個人或團體方式執行，至多 5 人)

姓名	工作內容	聯絡方式 (E-mail)
陳昶安	投影片製作	Chenchaungan0916@gmail.com
黃冠惟	報告	osdsdo@hotmail.com
周俊立	報告	Zhq123333@gmail.com
謝秉宏	論文統整、寫計劃書	404262169@mail.fju.edu.tw
陳新宇	投影片製作	tmhsu@gmail.com

計畫之動機與目的：

計畫內容與行動方案：分配工作後每個禮拜討論並回報進度

進度規劃

週次	規劃內容
第一週	搜尋本課程相關的論文、選定主題
第二週	組員閱讀並且了解論文內容
第三週	與教授討論主題內容，提出問題
第四週	製作投影片，寫自主學習計畫書
第五週	上課報告，完成自主學習成果報告

七、預期效益及具體檢核方式：

了解論文內容討論後製作投影片，藉由報告和與教授討論的方式來檢視是否有不了解的地方

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期：　107 年　6 月 12　日(二)

紀錄人：謝秉宏

序號	工作事項	註記(心得/成果/問題 等)
1	搜尋論文	由課程中的 Hadoop 主題下去搜尋
2	深入了解論文內容	有組員提出別篇論文,後來決定與教授討論
3	與教授討論	與教授討論過後決定論文
4	製作投影片	實驗結果的部分節錄較少
5	報告	報告的同學有些疑問,後來經由討論順利解決問題

待辦事項/待解決問題/筆記欄

與教授討論的過程與問題

本來我們有兩篇論文，經由和教授討論後決定採用”雲端運算 Hadoop 架構下離峰時間資料讀取之節點管理”這篇論文，因為該篇內容比較簡單，也有一些關於節點佈屬的主題與本課程相關。除此之外，教授也詢問有關自主計畫書的部分，也都有如期進行。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期：107 年 6 月 15 日

一、課程基本資料

(一)自主學習計畫主題：雲端運算 Hadoop 架構下離峰時間資料讀取之節點管理

(二)組別：第七組

(三)學生姓名：謝秉宏

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：

整體情況良好，少數問題如:主題選定、論文有些部分不懂，後來經由討論都順利解決

(二)成效說明與實際產出：（可附加佐證資料、文書記錄、照片或相關計畫運作情形資料等）

了解有關 Hadoop 架構下之離峰時間的節點節能部屬管理，藉由投影片使我們更加了解並推廣

(三)自主學習歷程：（請回顧「自主學習日誌」之內容並摘要寫出歷程）

搜尋論文

深入了解論文內容

與教授討論

製作投影片

報告

三、學習心得(每位學生提供至少 300 字的學習心得)

周俊立

這次報告是我第一次認真地閱讀學術性論文，在這之前我對一篇論文的結構以及書寫手法等並不是很清楚，在這次報告之後我對論文有了基本的了解。除了基本認識論文結構之外，論文的內容和我們的課程內容相關。我們選擇的論文內容和節能有關，老師在上課時為我們介紹了 HDFS 的架構，而我們選擇的論文討論了在 HDFS 基礎上改進的 greenHDFS，我對老師教授內容有了更深的了解，不僅僅拘束于課堂上的東西。綜上所述，這次報告與學習使我獲益良多，也希望下次還有機會參與類似的學習活動。

黃冠惟

這一次的自主學習，我們選擇了 Hadoop 的專題去做介紹，在這自主學習的過程當中，我又更加了解雲端的操作方式，也從中知道雲端系統也需要節省能源，從 HDFS 還有衍伸出 GreenHDFS，從 GreenHDFS 衍伸出 Cold Zone 和 Hot Zone，並依照資料類型的不同去對兩區塊去做處理，由於的資料不同所以也有硬體配置藉達到節點，兩個區塊去做處理，由於資料不同所以也有硬體配置藉達到節點的效果。

自主學習讓我更深入了解雲端運算，這項研究也讓我更深入得體會軟體和硬體得結合需要有精實思維的概念。

而這次自主學習的過程中，這團隊合作也是一項重大的學習經驗，有些組員草草了事，而有些組員很認真地看待這項任務，從中體會到一樣米養百樣人這句話的涵義，團隊合作就像小型社會一樣，不能因為個人的懶惰影響到別人。

最後我要謝謝老師給我們這次的機會去做學習，謝謝助教在這過程中的幫忙，謝謝組員和組長在這一次合作中的包容和指教。

陳昶安

閱讀這篇論文可以了解到兩大主題，覆蓋子集與 GreenHDFS 的基本觀念。為了提高資源的分配與效率，而有了覆蓋子集的提出。覆蓋子集是含括每一個 block 的子集，在減少節點數量來提高效率的情況下，保證每一個 block 不因節點的關閉而使得資料消失。GreenHDFS 是在 MapReduce 的 Map 階段前的節能部署策略，將節點群分成 Hot Zone 及 Cold Zone，並將資料在兩個 Zone 之間做轉換及部署。自主學習的過程中，遇到最大的困難是閱讀理解能力的不足。初次閱讀時誤解了文字敘述的意思，而有了覺得前後文敘述有互相牴觸的感覺。後來根據這篇論文所提供的參考資料稍微找尋相關的敘述閱讀，比對相關的敘述，才大致了解到敘述原本的意思。經過這一過程，學習到了閱讀一篇論文所需做的準備。在閱讀主題前需要先了解基礎知識，閱讀主題時或許需要與其他參考資料搭配與比較。

陳新宇

在這次的分組報告後，不管是學習上還是在合作上我都受益良多。這次的報告主要在於如何將從雲端概論獲得的知識，用於讀懂他人的論文，像這次我們選的這篇論文，其中將 HDFS 在效能無法提高的情況下，找出減少耗能的方法。這篇論文的作者在使用個種不同的判斷式後，成功找出一種能在 hot zone 切換到 cold zone 減少約五分之一的耗能。也許五分之一看似並沒有很多，但在大量資料，大量處理的情況下，耗能的基數會非常的大，而相對能節省的耗能也會相對的大。

在合作這方面我也學到很多。像是在於分工方面，組長必須將工作分配好，且組長必須看出各個組員的擅長而非興趣，只有讓組員各司其職才能達到最大的效益，且報告一定要提早做完，剩下的時間才能用於檢查報告中的錯誤，還有讓上台報告的人瞭解報告的內容。

謝秉宏

這次自主學習獲益良多，透過上課時教授教的 Hadoop 架構下的 HDFS 主題下去搜尋，找到算是課堂延伸的這篇論文，由這篇論文我們了解到 GreenHDFS 的概念，是將節點分為 Hot Zone 和 Cold zone 並藉由得到溫度變數來進行區域轉換達到節能效果，我們所選的論文算是簡單，而且我之前就有進行過類似的論文閱讀，所以算是很快的理解本篇論文的重點，唯一可惜的可能是我們無法將論文實驗成果呈現，而其中我們也遇到許多問題，像是有些組員不太明白論文內容，後來經由討論之後，除了解決問題外，也能加深自己對論文的了解，論文前段有些和上課內容稍微重疊，我也經由此論文加深以前上課內容的印象，藉由參考別人的論文寫作我將來在寫論文時的也有所幫助，提升我寫論文的能力，最後要謝謝跟我一起努力的組員，我身為組長雖然有些辛苦但能順利完成報告後也有所成長學習。

四、結論與具體建議

經過這一過程，學習到了閱讀一篇論文前須先具備基本的相關知識背景。閱讀時或許需要與其他人討論或是參考相關資料。或許將來除了節錄論文作成投影片報告外，可以小組寫一篇小論文。

成果報告

雲端運算 Hadoop 架構下離峰時間資料讀取之節點管理

組員:
404290520 黃冠惟 資工三 404262444 陳新宇 資工三
404262169 謝秉宏 資工三 404012150 周俊立 資工三
403262681 陳昶安 資工四

Hadoop 架構下節能之相關技術

- Hadoop 中包含了較為常用的儲存系統Hbase、分散式檔案系統HDFS、MapReduce框架
- 平行的運算方式處理這些資料，增進資料處理的效率。
- Hadoop 的節點可以進行資料備份的安全措施，避免資料遺失或損毀而造成無法使用的情況。

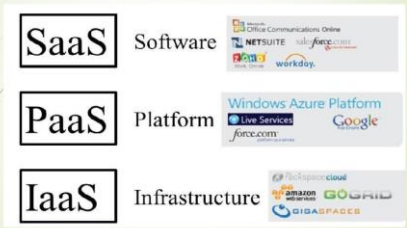
摘要

- 由於尖峰時段資料讀取量過高，若是將節點休眠可能不但沒有達到節能的效果，反而要付出更多的能源代價，所以針對使用者資料讀取離峰時間下的節點進行重新分配管理以節約能源。
- 實驗結果表明，比起原本的GreenHDFS執行任務時的耗能，在離峰時間下將可能閒置節點進入休眠狀態之後，可以有效的降低整體的21%能源消耗。

HDFS

- 硬體錯誤偵測及恢復
- 檔案分散式儲存系統
- 串流式的資料讀取
- 可移植性
- 一致性的模型，"一次寫入，多次讀取"

雲端部署模型



MapReduce 運算框架

在 HDFS 架構下利用MapReduce 框架來進行大量資料的運算，Name Node 負責決定將這些資料區塊分別交給底下的Data Node 來處理，Data node 接收到資料塊後進行處理，處理完依照Map 的標籤進行排序然後傳給Reduce 做合併然後將資料輸出。

雲端部署模型

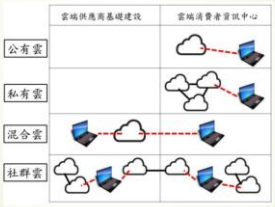
依供應商與消費者的關係分成四種部署模型

- 公有雲
  使用者藉公開的雲端平台達到使用目的
- 私有雲
  提供的服務與公有雲相似，雲端平台內部不對外開放
- 混合雲
  公有雲及私有雲的結合
- 社群雲
  多個組織共享資源池，以服務共同訴求，共同使用雲端資源

覆蓋子集

- 大量的節點來提高處理效率，但是當效率提高的時候，節點就有了閒置的可能，這樣就會造成系統資源的浪費，所以提出了一個覆蓋子集模型。
- 子集內有足夠的節點確保使用者能夠正常使用，其他子集外的節點將被關閉。
- 覆蓋子集大小佔整個群集的10%~30%

雲端部署模型



GreenHDFS

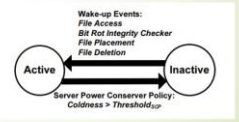
- 針對HDFS的節能佈署模型
- 資料區塊分成Cold Zone 及 Hot Zone
- 資料分類根據溫度變數進行分類成冷或熱資料，並移至Cold Zone 或 Hot Zone

Hot Zone

- 處理重要或時常被讀取的資料
- 配置高性能、高功率、高成本的CPU
- 高耗能 -> 無節能策略
- 佔全部節點的70%

Sever Power Conserver Policy

- Cold Zone內的節點作節點狀態控制。
- 利用溫度變數與閥值作比較，以此來控制節點。
- 當休眠狀態節點內的資料被動作，節點將被喚醒
  - 讀取
  - 刪除
  - 儲存
  - 位元衰退完整性檢查



Cold Zone

- Cold Zone 則負責處理那些比較不重要的或是不常被讀取的資料
- 不需要高效能處理，並且在耗能方面可以啟用低功率消耗甚至是無功率模式，達到節能的效果。

節能部署策略架構及步驟

- 為了降低能源的消耗，在資料進行MapReduce 前，判斷處理資料的時間是否為網路使用的離峰時段
- 當資料進入 MapReduce 後，對資料進行冷熱的判斷，資料依照冷熱度分別進入冷資料區塊或是熱資料區塊進行資料切割。
- Map 階段：將切割過後的input 給予(key,value)值，然後分配到節點上做處理。
- Shuffle 階段：將處理完的資料依照map 給的(key, value)值來進行排序。
- Reduce 階段：將排好順序的資料進行合併。
- Output 階段：將合併的資料輸出。

溫度變數

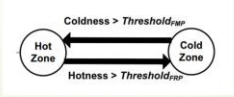
- 1.FileLifeSpan CFR(Create First Read)：定義為資料建立後到第一次資料被讀取的時間。
- 2. FileLifeSpan CLR(Create Last Read)：定義為資料建立後最後一次被讀取的時間。
- 3. FileLifeSpan LRD(Last Read Delete)：定義為資料最後一次被讀取到資料被刪除的時間。
- 4. FileLifeSpan FLR(First Last Read)：定義為資料第一次被讀取到最後一次被讀取的時間。

節能部屬策略File Migration Policy

- $T_s = T_c - T_l$
- T_s為閒置時間
- T_c為讀取資料並開始處理的時間
- T_l為上一次處理資料時最後被讀取的時間
- 當T_s >= $Threshold_{FMP}$,資料被判斷為冷資料,資料將被移至Cold Zone

區塊轉換管理機制

- Cold Zone 或Hot Zone 內的資料並不是永久存放在同一區塊內，避免造成Hot Zone 中的資料可能閒置過長時間但是卻沒被讀取，或是Cold Zone 中的資料突然頻繁的被讀取。
- 在區塊定時偵測內部的資料是否要轉換區塊，利用每筆資料的**溫度變數**可以知道資料是否為經常被讀取的資料、資料的閒置時間、資料在目前區塊儲存的時間長短用來判斷資料的溫度。



結論

- 針對GreenHDFS 在離峰時段可能發生的節點閒置、節點數量過多的問題進行改善，GreenHDFS 中冷資料區塊在離峰時間的情況下處理資料，由於冷資料區塊中的資料並不經常被讀取，而且在網路使用的離峰時段中，非常有可能會發生讀取資料量小而導致的節點閒置，當閒置的節點數量過多的時候，就會造成大量的能源浪費，所以在進入離峰時段後就會將可能閒置的節點進入休眠狀態，當資料量過高或是進入尖峰時段才會去喚醒睡眠狀態的節點。

File Reversal Policy

- For every file i in Cold Zone
- $Num\_accesses_i$ 為資料讀取次數
- 當$Num_accesses_i$ >= $Threshold_{FRP}$, 資料被判斷為熱資料,將被移至Hot Zone

2.8. 第八組

自主學習計劃書

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習計畫書

一、自主學習計畫主題：Efficient Storage of Multi-Sensor Object-Tracking Data

二、組別：第八組

三、課程名稱：雲端計算概論

四、指導老師：林振緯

五、學生姓名與工作分配：(可以個人或團體方式執行，至多 5 人)

姓名	工作內容	聯絡方式 (E-mail)
許孟傑	論文分析以及效能比較	404262614@mail.fju.edu.tw

計畫內容與行動方案：

閱讀相關論文，探討論文中提到的 SensorFS 對於感測器資料的彙集以及改善感測器與 HDFS 之間的不適用性。並將論文中新的檔案系統與直接將資料送進 HDFS 比較，看是否能改善 HDFS 在小資料讀寫上的浪費。

進度規劃

週次	規劃內容 (填寫大約計畫即可)
第一週	探討大小資料比較(Block Chain or Sensor Data for HDFS)
第二週	尋找小資料匯集方法
第三週	SensorFS、DMFS 與其他記憶體資料庫比較
第四週	SDG Based Algorithm 研究
第五週	效能比較

七、預期效益及具體檢核方式：

釐清此資料結構所獲得的效益以及整理論文中所提供的數據比較。

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習日誌

填寫日期：107 年 06 月 05 日(二) 10:00~12:00

紀錄人：許孟傑

序號	工作事項	註記(心得/成果/問題 等)
1	論文分析	新的 FS 要改善的問題
2	演算法研究	小資料要怎做匯集
3	效能比較	需要額外研究其他檔案系統原理

待辦事項/待解決問題/筆記欄

1. HDFS 對於小型資料的不適用性案例探討(Sensor Data)
2. 新資料型態（SensorFS）的規劃與比較
3. 小資料集合的演算法研究
4. 效能比較

輔仁大學 107 年高教深耕計畫【自主學習課程補助計畫】

學生自主學習成果報告

撰寫日期：107 年 06 月 21 日

一、課程基本資料

(一)自主學習計畫主題：Efficient Storage of Multi-Sensor Object-Tracking Data

(二)組別：第八組

(三)學生姓名：許孟傑

(四)課程名稱：雲端計算概論

(五)指導老師：林振緯

二、計畫成果

(一)整體計畫執行狀況描述：

在比較現有的記憶體資料系統時花費不少時間，演算法的部分有額外定義兩項規則要做分析。

(二)成效說明與實際產出：

論文中的做法在模擬上確實改善了 HDFS 的浪費，演算法的部分雖採用貪婪的方式求解，但相對於資料讀寫的時間上沒有太多的損害。

(三)自主學習歷程：

為了降低空間浪費，小資料在進入最小單位較大的資料系統前要做簡易的彙整，所以要考慮在 HDFS 上建立一個檔案系統。此系統需要能改善資料進入 HDFS 的時間，並嘗試改善 HDFS 的資料吞吐量。

三、學習心得(每位學生提供至少 300 字的學習心得)

在研究此論文前，我想研究有關小資料(block chain)與小資料(sensor data)的聯合使用，但是實際上前者只用於驗證資料的正確性。教授提議先研究小資料在雲端平台上的讀寫效能，於是就決定研究此篇論文。

此篇論文的亮點在於他是使用獨立在 HDFS 外的節點單位來做資料匯集，比起從 HDFS 改善讀寫效果更明顯，我們知道機械硬碟上的 4K 讀寫與大檔讀寫會有很大的差異（但在記憶體上不會）。與其在選擇一個適當的記憶體檔案系統上，這篇論文建立了一個確實具備 DMFS 功能的 SensorFS，而且也針對 SensorFS 做出了一個匯集資料用的演算法。

從成效與結論上，我們可以得知小資料進到 SensorFS 所消耗的時間遠低於小資料直接進 HDFS 的時間。當小資料在 SensorFS 匯集到比較大的資料後，這部分就是 HDFS 的強項了，並且也成功省下不少時間以及增大資料吞吐量。

四、結論與具體建議

在 SensorFS 上工作的伺服器並不只有一台。在不同伺服器的資料傳遞中，資料一致性參考自 HDFS，這部分需要再研究其他方法。論文中的演算法使用的是 Greedy 解法，雖有效率證明但沒有說明該演算法是否為最佳解法。

PULCHRITUDO VERITAS

成果報告

Efficient Storage of Multi-Sensor Object-Tracking Data

Source: IEEE TPDS vol. 27 no. 10, 2016/10
Presenter: Hsu, Meng Chieh, Team 8, Junior of CSIE, FJU

SensorFS and DMFS

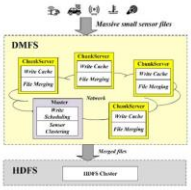
Agenda

- Problems for sending Sensor Data to Cloud
- SensorFS and DMFS
- Clustering methods
- Performance
- Conclusion

SensorFS

An small data unit generated by sensors.

- Extend from DMFS
- Distributed write caching
- Parallel file merging
- Sensor clustering

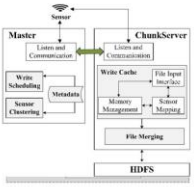


Problems for sending Sensor Data to Cloud

DMFS

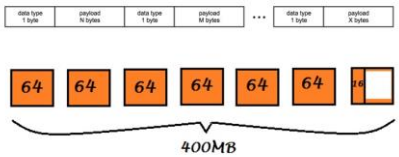
Distributed Memory File System

- Distributed
- Concurrent Write
- File Clustering
- File Merging



Data size compare

HDFS block size vs Sensor Data size



Reference: How exactly the data is stored on HDFS Cluster?
Sensor Data Format - Camera Media Server

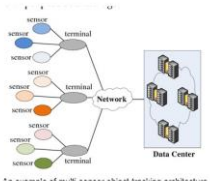
File System Compare

DMFS vs Existing Memory File Systems

	Redis	Memcached	Tachyou	RAMCloud	DMFS
Distributed	X	X	V	V	V
Concurrent Write	X	V	V	X	V
File Clustering	X	X	X	X	V
File Merging	X	X	X	X	V

Problems to be solved

- Massive data(from sensors) to be stored in data center
- Massive small data should be merged for analysis



Parameters for Efficiency Analyzing

Number of Sensors	10^5
File Size	10KB
Number of Files(n)	10^7
Number of DataNodes in HDFS(k)	6
Block size of HDFS	64MB
T_{cache}	70us
Run time of one Sensor Clustering	8s
Sensor-Clustering Frequency	Every 10^6 files
$T_{NameNode}$	1ms
$T_{DataNode}$	250us(10KB)
$T^{Block}_{DataNode}$	300us(64MB)

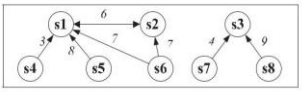
Efficiency Compare

HDFS vs HMFS vs SensorFS

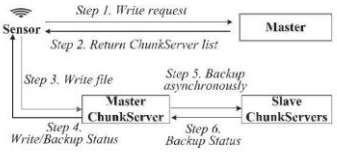
Type	Formula	Result(sec)
HDFS	$T_{HDFS} = \sum_{1}^{n}\left(T_{NameNode} + \frac{T_{DataNode}}{k}\right)$	10,700
HMFS	$T_{HMFS} = \sum_{1}^{n}(T_{cache}) + \sum_{1}^{m}\left(T_{NameNode} + T_{DataNode}^{Block}\right)$	1,170
SensorFS	$T_{SensorFS} = T_{clustering} + \sum_{1}^{n}\left(\frac{T_{cache}}{k}\right) + \sum_{1}^{m}\left(T_{NameNode} + \frac{T_{DataNode}^{HDFS\_Block}}{k}\right)$	315

Sensor-Dependence Graph

A directed weighted graph representing sensor distance.



Reliability of SensorFS



Sensor-Dependence Graph

The sensor-dependence graph has the following properties.

1. A dependence edge can **only point to one root node**. If two nodes have sensor dependence between each other, choose smaller sensor identifier as the root node.
2. A correlation **edge between root nodes is bidirectional**, which is introduced to reflect the sensor similarity.
3. A sensor-dependence graph is **not required to be a connected graph**.
4. For each sub-graph in an SDG, there are **only two levels of nodes connected by dependence edges**. So there is at most one dependence edge between any root node and any leaf node.

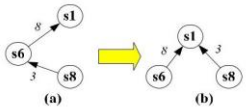
Sensor Clustering

Sensor-Dependence Graph

Lemma 3. B→A is equivalent to the following.

$$Jaccard(O_A, O_B) = \frac{|O_B|}{|O_A|}$$

Simplification Example



Sensor Similarity

Let A and B be sets of sensors.

O_A and O_B be the object set sensed by A and B.

$$Sensor\_Similarity(A, B) = |O_A \cap O_B|$$

$$Sensor\_Similarity(A, B) = \frac{Jaccard(O_A, O_B)}{1 + Jaccard(O_A, O_B)} \cdot (|O_A| + |O_B|)$$

$$Jaccard(O_A, O_B) = \frac{|O_A \cap O_B|}{|O_A \cup O_B|}$$

Algorithm 1. SDG_Construction(T)

Part 1. prepare initial 0(not sensed)/1(sensed) matrix
row: objects, col: sensors

```
1. M1 ← Sensor-Object-Matrix(T);
2. M2 ← Minhash(M1);
3. SP ← LSH(M2);
4. SS ← Sensor-Set(M1);
5. for each(Si, Sj) ∈ SP do
6.   if Sj ∉ SDG.leaf then
7.     if(Si ⇀ Sj and Sj ↛ Si) or (Si↔□Sj and Si < Sj) then
8.       SDG.leaf = SDG.leaf ∪ {Sj};
9. SDG.root = SS - SDG.leaf;
```

Sensor Dependence

Let A and B be sets of sensors.

O_A and O_B be the object set sensed by A and B.

If $O_B \subseteq O_A$, B depends on A(B→A).

Lemma 1. If B→A and C→B, then C→A.

Lemma 2. If B→A, then for any sensor set C, we have

Sensor_Similarity(B, C) ≤ Sensor_Similarity(A, C)

Algorithm 1. SDG_Construction(T) Cont.

Part 2. detect dependency edges
use Lemma 1 & 2 to simplify SDG to satisfy Property 4.

```
10. for each Si ∈ SDG.root do
11.   for each Sj ∈ SDG.leaf do
12.     if Si ⇀ Sj then
13.       SDG.d_edge = SDG.d_edge ∪ {Si⇀ Sj(weight = |Sj|)};
14. for each s, w ∈ SDG.root do
15.   d = Sensor_Similarity(s, w);
16.   if d > 0 then
17.     SDG.c_edge = SDG.c_edge ∪ {s ↔□ w (weight = d)};
18. return SDG
```

SDG-Based Sensor Clustering

For a SDG, we create an algorithm to

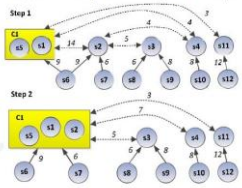
- Utilize the sensor dependence relationships
- Find sensor pairs that have the maximal sensor similarity.

Lemma 4. Let v be a leaf node in an SDG, and r be the node that has the maximal sensor similarity with v in the SDG, **r must be a root node of v**.

Lemma 5. A node pair that has the maximal positive value of sensor similarity in an SDG **must have a dependence edge** or **a correlation edge between them**.

Clustering Example Cont.

Processing s5 → s1(C5 → C1), C1 → C2(C1 → s2).



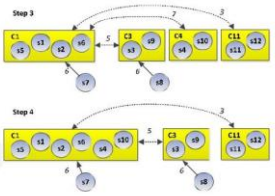
Algorithm 2. Sensor_Clustering(T)

Part 1. Initialize each node as cluster, pick each pair by Lemma 5

1. SDG ← SDG_Construction(T)
2. n ← get the number of sub-graphs in SDG;
3. **if** (n > MIN_CLUSTERS) **then**
4. C ← each sub-graphs is made a cluster;
5. **return** C;
6. **for** each node in SDG **do**
7. Initialize each node as cluster in C;
8. **while** |C| > MIN_CLUSTERS **do**
9. (c1, c2) ← the cluster pair with the maximal sensor similarity in SDG;

Clustering Example Cont.

Processing s12 → s11, s6 → C1, s9 → s3, s10 → s4 and C1 ↔□ C4.



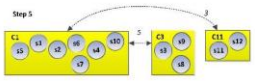
Algorithm 2. Sensor_Clustering(T) Cont.

Part 2. if dependency edge: merge front to back
if correlate edge: merge and calculate edge

9. **if** c1 → c2 ∈ SDG.d_edge **then**
10. Merge c1 into cluster c2;
11. Remove c1 → c2 from SDG;
12. **else if** c1 ↔□ c2 ∈ SDG.c_edge **then**
13. **for** each c ↔□ c2 ∈ SDG.c_edge and c != c1 **do**
14. **if** c ↔□ c1 ∉ SDG.c_edge **then**
15. Add c ↔□ c1 with the weight of c ↔□ c2 into SDG;
16. **else**
17. Compute the mew weight of c ↔□ c1 in SDG;
18. remove c ↔□ c2 from SDG;

Clustering Example Cont.

Processing s7 → C1, s8 → C3.



Algorithm 2. Sensor_Clustering(T) Cont.

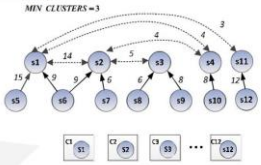
Part 2. if dependency edge: merge front to back
if correlate edge: merge and calculate edge

19. Merge c2 into cluster c1 and make all the leaves of c2 be c1's leaf nodes;
20. **for** each w ↔□ c1 ∈ SDG.c_edge **do**
21. Compute the new weight of c ↔□ c1 in SDG;
22. **return** C;

Performance Evaluation

Clustering Example

Generate cluster by SDG.



Experimental Setting

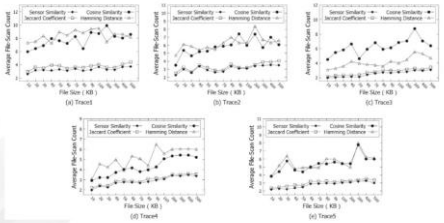
Parameters	Value
Server OS	Ubuntu 12.04 64bit
CPU	i5-4440 @3.10 GHz
RAM	Kingston 8GB DDR3
Hard Disk	Seagate ST1000DM003 1TB 7200 RPM
Server Role	1 * Master DMFS 1 * HDFS NameNode 5 * Nodes (for DMFS ChunkServer nodes & DataNodes of HDFS)

Features of the Five Traces

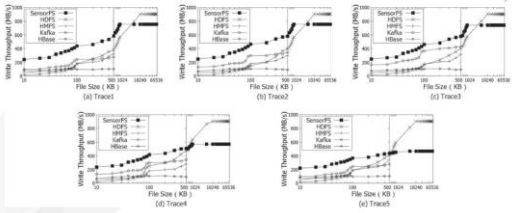
Generated by IndoorSTG, 50 buildings * 20 floors * 30 minutes for each traces.

Trace	Root-Node Ratio	Sensors	Objects	Files (10KB)	Files (500KB)
1	1%	100,000	2,608,800	22,010,048	292,028
2	5%	100,000	2,272,400	10,276,045	261,973
3	17%	100,000	2,195,200	10,171,187	228,047
4	26%	100,000	1,986,300	9,751,757	249,763
5	45%	100,000	1,991,900	9,856,614	265,804

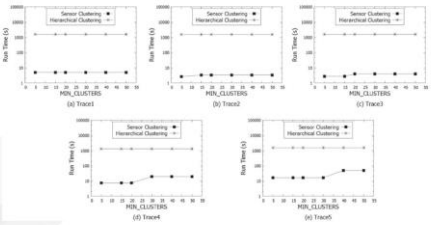
Average file-scan counts (sensor clustering)



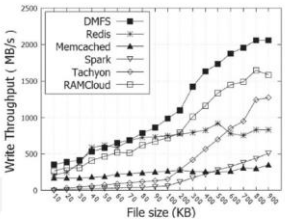
Disk-write throughputs



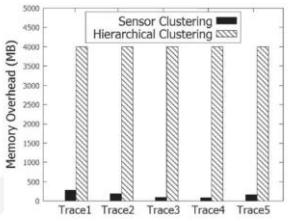
Time overhead



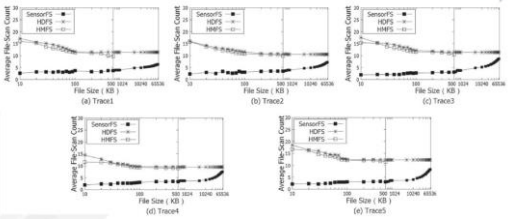
Memory-write throughputs



Memory overhead



Average file-scan counts



Conclusion

Average file-scan counts (sensor clustering)

$$Jarrcard\ Coefficeint = \frac{|O_A \cap O_B|}{|O_A \cup O_B|} = \frac{|O_A \cap O_B|}{|O_A| + |O_B| - |O_A \cap O_B|}$$

$$Cosine\ Similarity = \frac{|O_A \cap O_B|}{\sqrt{|O_A|} \cdot \sqrt{|O_A|}}$$

$$Hamming\ Distance = |O_A \oplus O_B| = |O_A \cup B| - |O_A \cap B|$$

Conclusion

We did improved the write throughput of HDFS.

SensorFS:
On top of HDFS to solve multi-sensor object-tracking files.

Sensor Similarity and Sensor Dependence:
Two concepts for sensor clustering.

SDG-Based Sensor Clustering algorithm + SensorFS:
Efficiency for performance between other FS.

Conclusion

There are several research directions in future work.

1. Reliability of SensorFS.
2. Improving the sensor-clustering algorithm.
3. Account the consistency of the merging files.

Q&A

Reference

Here are some proper noun in this paper.

1. Hmfs: Efficient Support of Small Files Processing over HDFS.
   Available: https://link.springer.com/chapter/10.1007/978-3-319-11194-0_5
2. In-Memory File System - Apache Ignite.
   Available: https://ignite.apache.org/features/igfs.html
3. Jaccard index.
   Available: https://en.wikipedia.org/wiki/Jaccard_index

3. 活動紀錄

演講（一）

4/17 胡永立 教授演講

演講主題：Virtualization Technology for Cloud Computing

雲端在現代發展是最為重要的指標之一，隨著資料量、運算、儲存等大幅提升，虛擬化的技術顯得更重要，胡教授在這場演講中明確的點出雲端虛擬化技術的重點以及技術使用，讓全班受益良多。

演講（二）

5/8 胡永立 教授演講

演講主題：Database Technology in Cloud Computing: Case Study on HBase

在資訊大爆炸的時代，不論是儲存還是搜尋都不能使用以往的方式，要如何在這麼龐大的資訊裡找到符合的項目，是這堂課胡教授想告訴我們的。

演講（三）

5/8 黃慶育 教授演講

演講主題：Software and Service Reliabilities Engineering：From State of the Art to the Best Current Practice

黃慶育教授帶給我們許多程式相關的新知識，像是程式的妥善率數據、軟體的測試方法等等，讓大家知道一個軟體的開發、檢測、上市甚至是維運都是非常重要且困難的。

4. 其他附件

學習問卷

演講心得