應用於雲端運算系統預測 DSRF 改善優先排程機制之研究

鍾弘哲*[a]、李維聰[a]、楊堯強[a]、王秀桂[b]
淡江大學電機工程學系[a]
中山科學研究院[b]
*通訊作者:chunghungche@gmail.com

對於 MapReduce 的演算法在先前的文獻研究中有提出了 DSRF Algorithm[1]的改善方案，但此排程機制會因為使用者的數量增加到一定數量以上的時候，會因為切換的頻率過多，反而造成系統效能的降低。本研究因參數控制，將原本的系統運算效能因為單一運算機制，使得系統負荷增大，運算時間加長的問題，改善並預測系統的最大使用人數，提前通知是否增加伺服器的數量，因此減少了不必要的時間浪費，在提升人數，反而使得效能低於傳統演算法的效能。

一、 前言

由於雲端技術[2]的蓬勃發展，許多高運算、高儲存量以及高彈性的資料目前都是透過雲端的技術來提升用戶的使用效能，降低用戶們的使用成本。Google 在雲端技術上提供了一個平行分散式的架構來處理龐大的資料搜尋或是資料的結合等等的程序[3]，這個架構稱之為 MapReduce。

MapReduce 提供了 Map 和 Reduce 兩個功能，讓使用者可以輕易的將待處理的大量資料自動的完成。而 Hadoop 是將 MapReduce 這個架構由概念變成實作的產物。由於目前 Hadoop 的應用絕大部分都還是用在如搜尋(Sort) 、資料統計等等複雜度較低且運算密度較高的程序上[4]-[6]。在先前的文獻研究中，有許多是針對改善其效率這部份的研究，在其中有對於 Reduce 的演算法提出了 Dynamic Seitch of Reduce Function (DSRF) Algorithm 的改善方案，因此減少了 Reduce 的閒置時間，但此排程機制會因為使用者的數量增加到一定數量以上的時候，會因為切換的頻率過多，反而造成系統效能的降低,甚至無法達到原本 Hadoop 所提供出來的效能品質。本研究提出透過斜率的計算來提供預測系統人數最佳效率的方法，藉此避免系統因人數過多後造成效率的降低。本篇論文第一章節為前言、第二章節為相關技術與背景、第三章節本為論文所提出之方法、第四章節為結果與分析、第五章節為結論與未來方向。

二、 相關技術與背景

2.1 MapReduce

圖 2.1 是 Google 為了需要處理大量的密集性資料，並且在平行式分散系統上面運行，所提出的 MapReducer 架構，其提供了使用者簡易的程式撰寫方法，讓程式的開發者方便且簡易的開發所需要的數據處理程式。MapReduce 主要是由三個部分所組成，分別是 Master 端、Map 功能和 Reduce 功能。

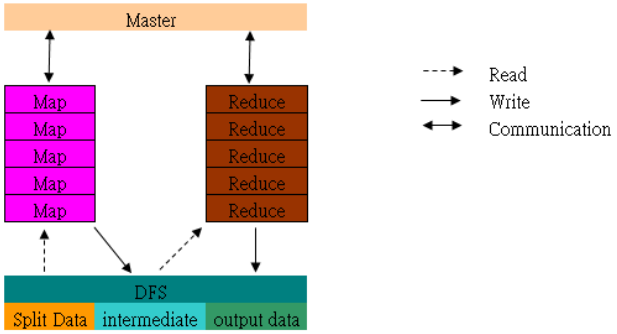

圖 2.1： MapReduce 架構圖[1]

2.1.1 Master 端

Maser 端主要的功能在於管理 Map 和 Reduce 這兩個功能，適時的提供這兩個功能所需的運算資料，並且指派那個 Map 或是 Reduce 功能到閒置的伺服器上直行運作。

2.1.2 Map function

Map 功能最主要的是把使用者所輸入的資料轉換成能夠讓 Reduce 功能接受的 Intermediate 資料，這其中包含了 Key 和 Value。Key 是代表著每一份資料之間的相關性，Value 則是由 Map 處理完的資料內容。

2.1.3 Reduce function

Reduce 主要就是將 Map 所處理好的資料，依據使用者的定義來處理這些資料，並將處理好的資料再生成新的 Key 和 Value，最後再交由 Maser 通知使用者來取得運算結果。

2.1.4 Google file system

GFS 的分散式的檔案儲存架構，是由 Google 所提出的，最主要是被使用在分散式和大量資料運算的程序上，使用者可以透過一般的個人電腦就可以達到如企業等級的高階設備所作的資料運算功能。

GFS 檔案系統會將系統上的資料切割成數個小容量的檔案，以依序讀取的方式來方便系統後續的處理。

2.2 Hadoop MapReduce

雖然 Google 提出了 MapReduce 的架構，但是並沒有提供相關的開發平台。但是 Apache 利用 Java 發展出具有 MapReduce 概念的應用軟體 Hadoop，而 Hadoop MapReduce 顧名思義就是建置在這個軟體下的 MapReduce 功能。如圖 2.2 為其架構。

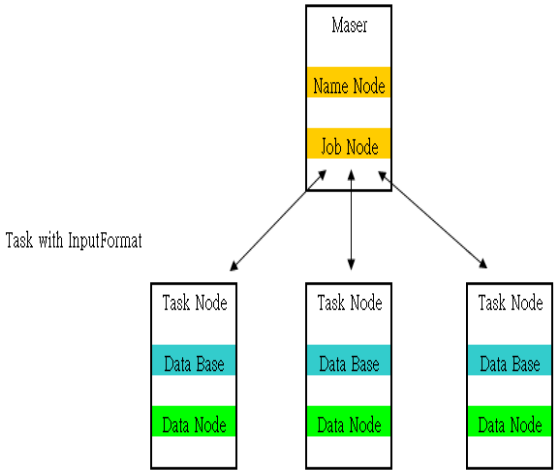

圖 2.2： Hadoop MapReduce 架構圖[1]

2.3 DSRF Algorithm

DSRF Algorithm 是用來改善傳統 Hadoop MapReduce 的執行效率所提出的方法。

Hadoop MapReduce 使用了 FIFO 的排程來進行資料的處理及運算，如圖 2.3。這使得系統大部份的時間都會因為等待而處於空閒的狀態，進而浪費了系統的資源。

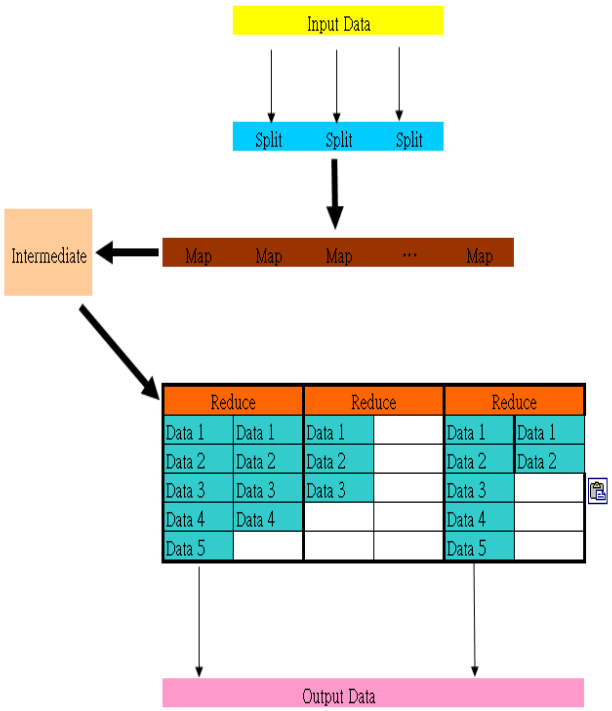

圖 2.3： 傳統使用 FIFO 架構圖

經由加入 DSRF Algorithm 的 MapReduce，如圖 2.4 所示，由於使用了動態切換的機制，會因此減少了系統等待並浪費時間的問題，進而增加其效率。

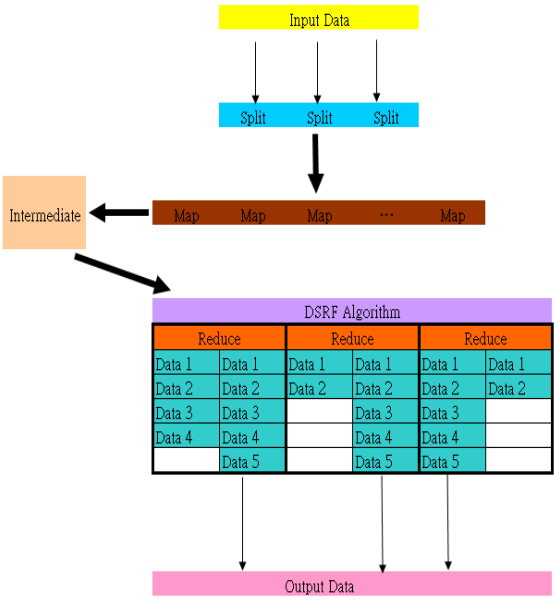

圖 2.4： 使用 DSRF 架構圖[1]

但使用 DSRF 這個架構的系統，仍然會因為使用者人數的增加後，因為動態切換的頻率增加，導致整個系統的時間延遲增加，進而拖累了系統的效能。此人數若是達到一定數量時，原本應有的優勢反而會讓原本使用 FIFO 的效能又超越 DSRF，如圖 2.5 所示。

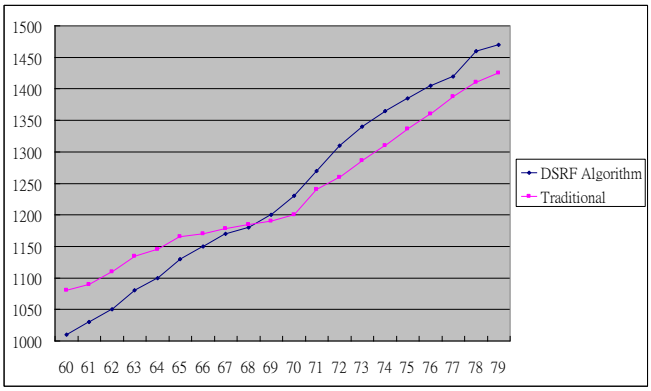

圖 2. 5： 使用者對平均運算時間的影響[1]

三、 動態偵測並預測系統最大人數機制

本論文透過兩點間斜率以及不同斜率的兩條線必有一交點的特性來預測此系統何時會遇到最大人數，事先預防並改善系統的效能。

由前述的兩種架構，我們可以假設兩條方程式，如下(1)、(2)式。

$$Ta = MaX + Ca \quad (1)$$

$$Tb = MbX + Cb \quad (2)$$

這兩式中的 x 為系統的使用者人數，Ta、Tb 分別為該使用人數所需花費的平均時間，Ma、Mb 為斜率，Ca、Cb 是常數。

藉由系統在運行的過程中，我們將可以記錄得知不同使用者人數時，該系統所需要的平均運行時間。因此取得兩個不同人數的結果後(t1、t2)，我們就可以知道此兩人數(x1、x2)之間的斜率(M)。如下公式(3)。

$$M = \frac{(t2 - t1)}{(x2 - x1)} \tag{3}$$

此時將斜率、人數、平均時間代回(1)、(2)式，就可以得到常數 Ca、Cb，因此就可以得知兩個完整方程式。我們可以預期在斜率不變的情形下，未來隨著人數的增加，此兩方程是必會有一交點。

由於此交點的產生，這兩個方程式就可以相等，因此我們就可以得到此交點的數值，這既為我們所推測出的系統最大人數。

隨著系統人數的增加，我們取樣計算的次數也會跟著增加，但由於斜率的不同，所以得到的最大人數也會不同。因系統接近最大人數時，其斜率的變化將隨之變小，因此我們可以根據人數結果出現的頻率，加上一個人數機率比例的參數(p)，就可以確認此預測的人數已經接近實際系統的最大人數。參數(p)的機率公式如(4)式。

$$p = \frac{(X\ \max - x)}{X\ \max} \tag{4}$$

其中 Xmax 為預測出的最大人數，x 為目前系統的實際人數。

四、 結果與分析

透過前面章節所介紹的公式，我們實際運用在 DSRF 和傳統 mapreduce 的系統中來做驗證。透過實際的資料數據，如表 I 所示。其中 69 人時以為系統最大效率的人數。

表 I： 人數與系統運行時間

	系統平均運行時間	
人數	DSRF Algorithm	Traditional
10	110	110
20	300	310
30	475	490
40	610	690
50	805	890
60	1010	1080
61	1030	1090
62	1050	1110
63	1080	1134
64	1100	1145
65	1130	1165
66	1150	1170
67	1170	1178
68	1180	1185
69	1200	1190
70	1230	1200
71	1270	1240
72	1310	1260
73	1340	1286
74	1365	1310
75	1385	1336
76	1405	1360
77	1420	1388
78	1460	1410
79	1470	1425
80	1500	1450

透過公式的運算，我們將此表的每兩列人數計算一次斜率及預測人數，結果如表 II，我們可以看到預測人數在 63 人開始已經預測到最大人數是接近 69 的數值。

表 II： 各人數間的斜率與所預測的最大人數

人數	斜率 DSRF	斜率 Traditional	預測最大人數
10	19	20	10
20	17.5	18	0
30	13.5	20	27.7
40	19.5	20	-120
50	20.5	19	106.7
60	20	10	67
61	20	20	69
62	30	24	72
63	20	11	69
64	30	20	68.5
65	20	5	67.3
66	20	8	67.7
67	10	7	69.7
68	20	5	68.3
69	30	10	68.5
70	40	40	69
71	40	20	69.5
72	30	26	59.5

73	**25**	**24**	**19**
74	**20**	**26**	**83.2**
75	**20**	**24**	**87.3**
76	**15**	**28**	**79.5**
77	**40**	**22**	**75.2**
78	**10**	**15**	**88**
79	**30**	**25**	**70**

藉由所預測出來的最大人數與實際人數，經過機率參數公式得到的結果，發現這些時候的 p 值已經相當的小，如表 III。這意味著，當目前的實際人數已經開始接近系統運行效能最好的最大人數，因為若是目前人數離最大人數一段差距時，此 p 值將會接近 1 的數值，因此這兩個條件成立的情況下，我們找到此系統的最大人數。

表 III： 人數與機率參數 p

人數	最大人數	p 值
63	**69**	**0.086957**
64	**68.5**	**0.065693**
65	**67.33333**	**0.034653**
66	**67.66667**	**0.024631**
67	**69.66667**	**0.038278**
68	**68.33333**	**0.004878**
69	**68.5**	**-0.0073**

五、 結論與未來方向

藉由前面章節所介紹，原本使用 DSRF Algorithm 所得到的效能會因為系統最大人數的限制導致效能低於傳統的 MapReduce 效能，所以我們可以藉由斜率計算的方法發現系統運行效能的最大人數限制，提前增加伺服器的數量，來避免系統效能的降低。但在計算的過程中若是發現有不相交的狀況發生時，我們可以選擇忽略此人數所預測的結果，因為最大預測人數必須是在多個人數計算出來的結果，而這個結果所預測出來的數值都要是接近的，才可以說此數值為系統最大人數。在未來，希望能夠有其他的改善方式來加強提前預測系統最大人數的上限值，以利增加伺服器的準備時間，並且足以保持系統維持一定的效能。

參考文獻

[1] 郭玲裳, “基於MapReduce的影像處理系統加入DSRF優先排程機制,” 淡江大學電機工程學系碩士論文,中華民國一百零一年六月.

[2] NovQinlu He,Zhanhuai Li’Xiao Zhang, “Study on Cloud Storage System based on Distributed Storage Systems”, Computational and Information Sciences(ICCIS), 17-19 Dec 2010, pp. 1332 - 1335

[3] Mingyue Luo,Gang Liu, “Distributed log information processing with Map-Reduce: A case study from raw data to final models”, Information Theory and Information Security(ICITIS), 17-19 Dec. 2010, pp.1143-1146

[4] Chen Zhang ,De Sterck, H.,”CloudBATCH: A Batch Job Queuing System on Clouds with Hadoop and HBase”, Cloud Computing Technology and Science (CloudCom), Nov. 30 2010-Dec. 3 2010, pp. 368-375

[5] Jorda Polo, David Carrera, Yolanda Becerra, Malgorzata Steinder, and Ian Whalley., “Performance-driven task co-scheduling for mapreduce environments”,Network Operations and Management Symposium (NOMS) , 19-23 Apr 2010, pp.373 –380

[6] Thomas Sandholm and Kevin Lai., “Dynamic proportional share scheduling in hadoop”, Job Scheduling Stragies For Parallel Processing (JSSPP), vol. 6253, 2010, pp. 110-131