資料格網中可調適的資料複製策略

Adjustable File Replication Strategy for Hierarchical Data Grid

王逸民
靜宜大學資訊管理學系
ymwang@pu.edu.tw

游瑄富
靜宜大學資訊管理學系
g9771015@pu.edu.tw

摘要

Data Grid 提供了強大的運算能力和大量的資料儲存空間，而複製檔的運用更可以減少資料存取時間和減少頻寬的消耗。張瑞雄等人提出了 Latest Access Largest Weight (LALW)的演算法，LALW 演算法記錄了檔案存取的次數，從中找出熱門的檔案，進而進行檔案的複製。LALW 的演算法減少了執行的時間和增加了網路的可用性。在這篇論文中我們提出 Adjustable File Replication(AFR)以改良 LALW 演算法，讓複製檔的選擇更有彈性，增加了複製檔的選擇範圍不再侷限只複製一個檔案。這樣可以增加複製的效率和避免複製到重複的檔案。模擬的結果顯示，改進後的演算法在選取範圍為 90%和 80%時分別增加了 4%和 11%的網路可用性在工作執行時間上也分別減少了 6%和 13%。

關鍵詞: 資料格網; 資料複製

Abstract

To shorten the file access latency and to reduce the network bandwidth consumption for data grid environments, many dynamic file replication strategies have been studied extensively recently. For example, Chang et al. proposed Latest Access Largest Weight (LALW) strategy for locality file access pattern on hierarchical data grid. Based on file access records, LALW first chooses the most popular file among replicated files, and computes the number of copies should be made for that popular file. Then LALW locates the suitable nodes for those copies. In this paper, we propose a new dynamic file replication strategy, Adjustable File Replication (AFR), to improve LALW. As the differences of accessed times among the leading popular files are insignificant. Instead of choosing only one popular file,AFR suggests that more popular files should be selected. Simulation results show that AFR improves the effective network usage and reduces the remote file access time on hierarchical data grids.

Keywords: Data Grid; Data Replication

1. 前言

隨著科技的進步，對於運算能力和資料儲存的需求越來越高，而 Data Grid 的出現提供了我們解決這些問題的方案[4,5]，舉例來說高效能的科學計算，如:生物和天氣方面，需要很高的運算能力和儲存空間，Data Grid 的出現讓這些問題不需要再依靠超級電腦[1]。歐洲核子研究委員會(CERN)的大型強子對撞機(LHC)實驗便是使用了 Data Grid 來當作資料儲存和運算的平台[2,12]。 LHC 一年會產生

15PB(PetaByte, 2^{15})的資料，而 Data Grid 方便的擴充性，強大的運算能力和龐大的儲存空間，非常適合 LHC 的使用。

增加資料的複製檔不只可以減少檔案存取時間的花費，更可以增加資料的可用性。增加複製檔的方法有靜態和動態兩種，靜態的檔案複製方法需要手動複製檔案，所以較無彈性，無法適應變動的網路環境。而動態的檔案複製方法在網路環境有變動時可以經過演算法來做改變，以適應改變後的網路環境[6, 10, 11, 13]。如何選擇較適合的檔案來做複製是增加資料複製檔中一個很重要的問題，在 Cascading Replication[6]當中針對檔案需求的次數來找出需求量較大的檔案來做複製。張瑞雄等人提出了一個動態的複製演算法 Latest Access Largest Weight (LALW) [9]，此演算法相信一個較為熱門的檔案在未來仍有較大的機會被使用到。LALW 演算法會收集存取記錄的歷史資料，然後在記錄裡面找出哪一個檔案是較為熱門的，接著就會複製較為熱門的檔案給 client。此方法可以有效的降低執行時間，增加網路的可用性還可以有效的節省儲存空間。但是在此方法當中如果有些檔案彼此的存取次數相近便會造成某些選擇上的困難，或是當有某個檔案的存取次數產生異常，便會對之後的數據計算上產生很大的影響。

在這篇論文當中，我們提出了 Adjustable File Replication(AFR)策略，在複製檔的選擇方面增加了動態的機制，增加了複製檔的選擇範圍，不再只侷限於只複製一個檔案，在記錄檔方面改善了大量的存取次數所產生的問題，讓存取次數的評估和計算方面更為合理，也可以防止後續數據計算上的影響。實驗結果顯示改進後的演算法增加了網路的可用性，也減少了工作執行的時間，讓整體的效能更為提升。

本篇論文分為以下各節:第二節介紹相關研究的部份，在第三節中提出改進的演算法，第四節中會介紹模擬的環境和結果，第五節則是本篇論文的結論。

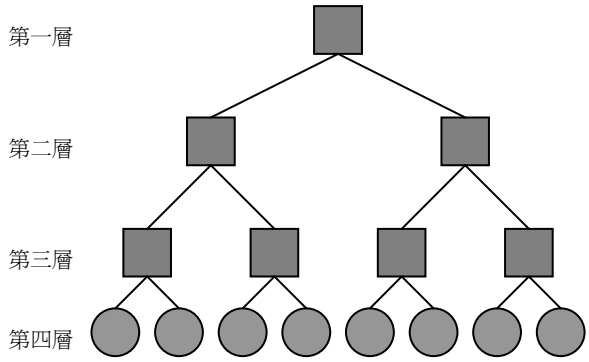
2. 相關研究

增加複製檔案的方法是為了能夠減少傳輸時間的花費也可以減少頻寬的消耗，其中靜態的資料複製方法可以達到上述的某些目標，但是他無法適應眾多使用者的需求變化，在一個龐大的 Data Grid 當中，靜態的複製方法是不可行的。在動態的檔案複製上可以依照使用者的需求來做不同的複製檔選擇，在使用者的需求有所改變的時候動態的檔案複製方法也可以適應使用者需求做變更。Ranganathan, K. and Foster, I. 在[6]整理出幾項常見的檔案複製策略: 1. No replication or caching 2. Best Client 3. Cascading Replication 4. Plain Caching 5. Caching plus Cascading Replication 6. Fast Spread. 這些策略會藉由三種不同的資料存取模型來做評估。 1.亂數的存取模型，表示檔案的需求是沒有規律的(Random access pattern)2.在一定的時間內同個節點會有相同的檔案需求(Data contain a small amount of temporal locality)3.在一定時間上某個節點和其他附近的節點會有相同的檔案需求(Data contain a small amount of geographical and temporal locality)。

上述方法中，No replication or caching 是完全不做任何的檔案複製，而表現較佳者有 Cascading 和 Fast Spread 兩種，這兩種方法是基於多層的 Data Grid 環境。圖一為多層的 Data Grid 架構。原始檔案會放在 root 當某個檔案的請求數量超過了門檻值，便會把檔案往下一層複製。 Fast Spread:此方法會將檔案複製在請求點路徑上的所有節點，可以讓檔案的傳播更為快速。Fast Spread 的方法在亂數的存取模型中可以得到比較好的效果，而 Cascading Replication 在一定時間上某個節點和其他附近的節點會有相同的檔案需求時可以得到最好

的效能回應時間[6]。

另一個比較特別的動態資料複製方法是 Simple Bottom-Up(SBU) 和 Aggregate Bottom-Up(ABU)[7]。這兩種方法也是基於多層的 Data Grid 環境如圖一所示。



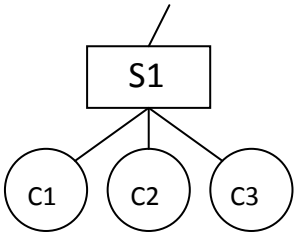
圖一. 階層式的 Data Grid 環境架構

SBU 基本概念是盡可能將檔案複製到最接近其存取頻率已超過某預設門檻值的點，SBU 會設定一個門檻值，如果存取的次數達到了門檻值但是檔案已經存在父節點則不會進行複製。反之如果達到了門檻值但是檔案並沒有存在於父節點則會進行檔案的複製。ABU 跟 SBU 的不同點在於 SBU 只參考單次的紀錄而且不考慮其他的記錄，但是 ABU 則會結合相同檔案需求的紀錄來做判斷，在蒐集了這些紀錄之後會把所有的紀錄聯合起來計算，如果超過門檻值就會複製，除了這點之外其他的概念就跟 SBU 演算法類似。

舉例來說，圖二中 c1、c2、c3 皆為 S1 的子節點，其中 c1 存取檔案 X 的次數有 8 次，c2 存取了檔案 Y 有 10 次，c3 存取檔案 X 有 5 次。在 SBU 的演算法中假設門檻值為 10，c2 存取檔案 Y 是次數最多的有 10 次而且超過了門檻值，所以會複製檔案 Y 到 S1。

但是在 ABU 演算法中會把圖二的檔案傳輸次數表結合成(S1、X、13)，(S1、Y、10)，由修改過後的表格顯示出檔案 X 才是較為熱門的檔案。

FileID	ClusterID	Access Times
X	C1	8
Y	C2	10
X	C3	5



圖二. 檔案需求次數和節點關係範例

Latest Access Largest Weight (LALW)[9] 演算法是建立在階層狀的網路架構上，張瑞雄等人假定熱門的檔案在未來會有較大的機會被存取，利用 LALW 的演算法找出較為熱門的檔案然後進行複製。 LALW 演算法有三個階段，每過一段時間便會執行 LALW 演算法。

第一階段:找出哪個是熱門的檔案，在 LALW 的演算法中，每個節點會有一個表格來記錄存取的歷史紀錄，<FileID、ClusterID、Access Times>，FileID 代表了此節點傳送了哪個檔案，ClusterID 是此節點把檔案傳到哪個 Cluster 上，Access Times 代表了檔案存取的次數。

首先，會蒐集各個表格的歷史記錄資料，把各個記錄做加總找出存取次數最多的檔案，存取次數最多代表他是最為熱門的檔案。為了減少老舊記錄檔造成的影響，LALW 演算法提出了一個遞減的機制，每過一個階段舊的歷史紀錄便會減少一半的次數，利用此方法可以減低老舊記錄的影響。

第二階段: 計算要複製多少個檔案，在 LALW 演算法當中，會利用和其他檔案的平均傳輸次數來做計算，找出此檔案要複製多少個。

第三階段:決定要複製到何處，當找出要複製哪一個檔案時會根據紀錄檔中此檔案在

Cluster 1 Header Record			Cluster 2 Header Record		
FileID	ClusterID	Access times	FileID	ClusterID	Access times
A	C3	9	C	C1	5
B	C3	5	D	C1	10
B	C4	6	E	C4	8
D	C2	10			
Cluster 3 Header Record			Cluster 4 Header Record		
FileID	ClusterID	Access times	FileID	ClusterID	Access times
C	C1	9	A	C2	10
C	C4	6	A	C3	12
E	C1	5			

Aggregated Record(t1)		
FileID	Access Times	Come from
A	31	C2:10;C3:21
B	11	C3:5;C4:6
C	20	C1:14;C4:6
D	20	C1:10;C2:10
E	13	C1:5;C4:8

圖三. LALW 第一時間階段範例

前一個時間階段

Aggregated Record(t1)		
FileID	Access Times	Come from
A	31	C2:10;C3:21
B	11	C3:5;C4:6
C	20	C1:14;C4:6
D	20	C1:10;C2:10
E	13	C1:5;C4:8

最新的時間階段

Aggregated Record(t2)		
FileID	Access Times	Come from
A	11	C2:11
B	20	C2:8;C3:12
C	14	C4:14
D	11	C1:5;C2:6
E	8	C4:8

Aggregated Record		
FileID	Access Times	Come from
A	$31*2^{-1}+11*2^{0}=26.5$	C2:10/2+11=16
		C3:21/2+0=10.5
B	$11*2^{-1}+20*2^{0}=25.5$	C2:0/2+8=8
		C3:5/2+12=14.5
		C4=6/2+0=3
C	$20*2^{-1}+14*2^{0}=24$	C1:14/2+0=7
		C4:6/2+14=17
D	$20*2^{-1}+11*2^{0}=21$	C1:10/2+5=10
		C2:10/2+6=11
E	$13*2^{-1}+8*2^{0}=14.5$	C1:5/2+0=2.5
		C4:8/2+8=12

圖四. LALW 演算法第二時間階段範例

哪個 Cluster 的執行次數較多，來決定檔案要複製到何處。

參考[9]的例子，圖三當中的 Aggregated Record(t1)表格結合了 Cluster1 到 Cluster4 的記錄表，由 Aggregated Record 中可以找出傳輸次數最多的檔案是 Flie A，所以決定複製檔案 A。此為 LALW 演算法的第一階段。

在第二階段中，會計算要複製多少個檔案 A，在圖三中由 Aggregated Record 裡的 Access times 算出五個檔案平均的存取次數為 19 次，代入檔案 A 的存取次數經由 floor(31/19)=1 表示檔案 A 只複製一份，計算出要複製多少個檔案 A 之後，接著是要決定檔案要複製到哪個 Cluster。在 Aggregated Record 裡的 Come from 可以看出，Cluster2 對檔案 A 的需求次數最多，在第三階段會針對要複製的檔案個別的 Cluster 讀取次數來做計算，會將檔案複製到存取次數最多的 Cluster。

最後的結果顯示出在第一階段時間會選擇檔案 A 來做複製，複製的份數為一份，複製的點則會選擇 Cluster3。

上面所述為第一時間階段，在圖四中可以看出進入下一個時間階段時 LALW 演算法使用存取次數減半的方法來減弱老舊記錄檔的影響力，由 Aggregated Record 裡的 Access times 中每前一個時間階段的檔案存取次數都會做減半的動作，越舊的記錄檔遞減的次數就越多。在第二時間階段找出檔案 A 為最熱門的檔案，複製份數為一份，複製到 Cluster2。

3. Adjustable File Replication (AFR)

在張瑞雄等人的方法中，雖然利用了遞減的機制來減少對後續的影響，而若有數量較大的存取，仍在會在後續的幾個階段中造成決定

前一個時間階段

Aggregated Record (t1)		
FileID	Access Times	Come from
A	10	C2:10
B	11	C3:5;C4:6
C	20	C1:14;C4:6
D	20	C1:10;C2:10
E	13	C1:5;C4:8

最新的時間階段

Aggregated Record (t2)		
FileID	Access Times	Come from
A	11	C2:11
B	20	C2:8;C3:12
C	14	C4:14
D	11	C1:5;C2:6
E	8	C4:8

Aggregated Record		
FileID	Access Times	Come from
A	$10*2^{-1}+11*2^{0}=16$	C2:10/2+11=16
B	$11*2^{-1}+20*2^{0}=25.5$	C2:0/2+8=8
		C3:5/2+12=14.5
		C4=6/2+0=3
C	$20*2^{-1}+14*2^{0}=24$	C1:14/2+0=7
		C4:6/2+14=17
D	$20*2^{-1}+11*2^{0}=21$	C1:10/2+5=10
		C2:10/2+6=11
E	$13*2^{-1}+8*2^{0}=14.5$	C1:5/2+0=2.5
		C4:8/2+8=12

圖五. AFR 第二時間階段範例

性的影響。由圖四中可以看出，在第二階段時間會複製檔案 A，但是在第一階段時間已經複製檔案 A 到 Cluster3 在時間計算上仍然把 Cluster3 需求檔案 A 的次數計算進去這是有問題的。所以在 AFR 中會把複製過的檔案在記錄檔裡的存取次數記錄做清除，以減少對於後續的影響，因為當 Cluster 被複製新檔案時代表在一定的時間內此 Cluster 不需要遠端存取此檔案，如果把不需要的次數加入計算便會有誤差。

舉例來說，圖五可以看出 AFR 在第二階段的存取時間計算上扣除掉之前已複製檔案的需求次數時便會造成不同的結果，扣除已複製過的存取次數可以發現計算出來的存取次數和圖四有明顯的差距，由此可見此差距對後續的次數計算上會造成影響。

另一方面，在 LALW 的方法中，在複製檔的選擇方面只會複製一個最熱門的檔案，但是如果有存取次數相近的情況發生或存取次數相同時只選擇一個檔案來複製便會造成疑慮，由圖四中可以發現在第二時間階段中檔案 A 跟檔案 B 的存取次數分別為 26.5 和 25.5，以如此相近的存取次數卻只複製檔案 A 似乎是有些問題的。所以在 AFR 中會利用一個比例，假設當其他檔案的存取次數達到最熱門檔案存取次數的 80%就一併複製，這樣可以增加檔案複製的速度也可以改善存取次數相近造成的影響。

在檔案的選擇上 AFR 對於熱門檔案的存取次數做選擇範圍的設定，舉例來說，使用 90%或 80%的設定範圍會有不同的效果，使用 80%的選擇範圍在複製檔的數量上便會比使用 90%的範圍來的多。

不同的選擇範圍在檔案散佈的速度上便不相同，由圖六可以看出使用不同比例的選擇範圍(80%和 90%)做選擇時產生的結果。

在圖六中檔案 B 為最熱門的檔案，依照 LALW 演算法只會複製檔案 B。AFR 會把檔案 B 的存取次數分別取 90%或 80%當門檻值，其數值分別為 22.95 和 20.4 。存取次數超過檔案 22.95 的有檔案 B 和 C，存取次數超過 20.4 的則有檔案 B 檔案 C 和檔案 D。

Aggregated Record		
FileID	Access Times	Come from
A	$10*2^{-1}+11*2^{0}=16$	C2:10/2+11=16
B	$11*2^{-1}+20*2^{0}=25.5$	C2:0/2+8=8
		C3:5/2+12=14.5
		C4=6/2+0=3
C	$20*2^{-1}+14*2^{0}=24$	C1:14/2+0=7
		C4:6/2+14=17
D	$20*2^{-1}+11*2^{0}=21$	C1:10/2+5=10
		C2:10/2+6=11
E	$13*2^{-1}+8*2^{0}=14.5$	C1:5/2+0=2.5
		C4:8/2+8=12

比例	門檻值
80%	25.5*0.8=20.4
90%	25.5*0.9=22.95

選擇複製檔	
80%	90%
B	B
C	C
D	

圖六. AFR 以 80%和 90%範圍來做檔案選擇範例

4. 模擬結果

在模擬方面，我們使用 C 語言亂數產生 80 組檔案需求然後來做一個簡單的模擬。模擬的參數在表一當中。

表一. 模擬參數表

Cluster 數量	5
檔案數量	10
檔案需求次數	80

在效能評估方面 Effective Network Usage (ENU)[3]可以評估網路的可用性，其中

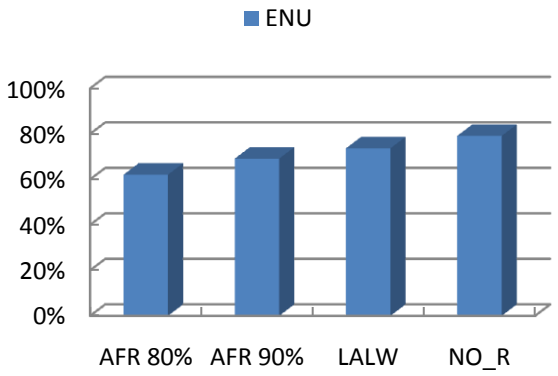
$$ENU = \frac{N_{\text{remote file access}} + N_{\text{file replication}}}{N_{\text{file access}}}$$

N remote file access 表示遠端的檔案存取， N file replication 為檔案複製的次數， N file access 則為所有檔案存取的次數。較低的 ENU 表示網路使用率較低，效能較好。

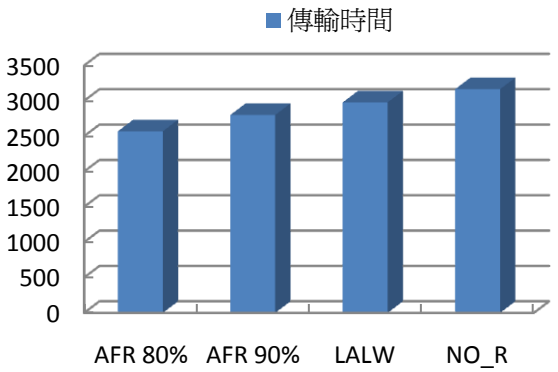
在實驗中設定遠端存取所花費的時間為 50 秒，進行檔案複製的時間為 80 秒。由此來計算檔案遠端存取和複製的傳輸時間。NO_R 為不進行檔案複製，以此來做比較。

在圖七中可以看出在 NO_R 中因為不進行檔案的複製所以大量的檔案要進行遠端存取所以其 ENU 達到了 79%在 LALW 中 ENU 為 73%。AFR80%和 AFR90%分別為不同比例的選擇範圍，其 ENU 分別為 62%和 69%。由此可見 AFR 可以有效的降低 ENU。

圖八為檔案的傳輸時間，由此可以看出其檔案在做遠端存取和複製的時間，可以發現 AFR90%比 LALW 演算法降低了約 6%的時間，AFR80 比 LALW 演算法降低約 13%的傳輸時間。由此可見 AFR 可以有效的降低檔案遠端存取和複製的時間。



圖七. Effective Network Usage



圖八. 平均檔案遠端存取和複製的傳輸時間

5. 結論

在本篇論文當中，我們使用 Adjustable File Replication (AFR)策略，改善了 Latest Access Largest Weight 演算法中大量檔案需求次數所產生的問題，設定了一個範圍來選擇要複製哪些檔案，AFR 可以有效的降低 Effective Network Usage 和工作的執行時間。在未來會考慮加入檔案一致性的研究，考慮在檔案內容有變更時，如何去更新其他的複製檔。

致謝：本論文經費部分由國科會計畫提供(NSC 97-2221-E-126-004,NSC 98-2221-E-126-004)在此致謝。

參考文獻

1. A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, S. Tuecke (2000) *The data grid: towards an architecture for the distributed management and analysis of large scientific datasets.* J Netw Comput Appl 23:187–200.
2. CERN. http://public.web.cern.ch/public/
3. D. G. Cameron, R. C. Schiaffino, P. Millar, C. Nicholson, K. Stockinger, F. Zini (2003) *UK grid simulation with OptorSim.* e-science all-hands meeting, Nottingham, UK, September 2003
4. I. Foster, C. Kesselman, *The Grid: Blueprint for A New Computing Infrastructure,* Morgan Kaufmann, San Fransisco, 2004.
5. I. Foster, *The grid: A new infrastructure for 21st century science,* Physics Today 55 (2002) 42–47.
6. K. Ranganathan, I. Foster., *Identifying Dynamic Replication Strategies for a High Performance Data Grid.* in International Workshop on Grid Computing, (2001).
7. M. Tang, B. S. Lee, C. K. Yeo, X. Tang (2005) *Dynamic replication algorithms for the multi-tier data grid.* Future Gener Comput Syst 21:775–790
8. M. Tang, B. S. Lee, X. Tang, C. K. Yeo (2006) *The impact of data replication of job scheduling performance in the data grid.* Future Gener Comput Syst 22:254–268
9. R. S. Chang, H. P. Chang, *A dynamic data replication strategy using access-weights in data grids.* J. Supercomputing, 10.1007/s11227-008-0172-6,(2008).
10. R. S. Chang, P. H. Chen, *Complete and fragmented replica selection and retrieval in Data Grids,* in: Future Generation Computer Systems 23 (2007) 536–546
11. S. M. Park, J. H. Kim, Y. B. Ko, W. S. Yoon, *Dynamic Data Grid replication strategy based on internet hierarchy,* Second International Workshop on Grid and Cooperative Computing, GCC'2003, Shanghai, China, Dec 2003.
12. The Large Hadron Collider. http://public.web.cern.ch/Public/en/LHC/LHC-en.html
13. Y. L. Yuan, Y. W. Wu, G. W. Yang, F. Yu, *Dynamic Data Replication based on Local Optimization Principle in Data Grid,* in: 2007 The Sixth International Conference on Grid and Cooperative Computing