4C-5

ストレージ省電力化手法RAPoSDAに対するデータ複製数の増加が与える影響

引田 諭之† レー ヒェウハン† 横田 治夫†

†東京工業大学大学院情報理工学研究科計算工学専攻

1 はじめに

近年における情報量の爆発的増加に伴い大規模化するストレージシステムにおいては，増大する消費電力量の削減は重要な課題である．我々はこれまでに，ストレージシステムの省電力化手法である RAPoSDA(Replica Assisted Power Saving Disk Array)[3] を提案してきた．RAPoSDA では信頼性および可用性を確保するためにデータをプライマリ・バックアップ構成で二重化し，個々のディスクドライブの回転状況を考慮したアクセス制御により省電力化を実現している．

データの複製数を増やすことは，ストレージの信頼性と可用性を高める主要な方法であり，実運用における大規模ストレージシステムである Google File System(GFS)[2] や Hadoop Distributed File System(HDFS)[1] ではデフォルトで三つの複製データを別々のノードに格納している．

RAPoSDA においても，信頼性および可用性をより高めるためにデータの複製数を増やすことは重要である．本報告では RAPoSDA におけるデータの複製数を増加する際のデータ配置方法を検討し，そのデータ配置方法が RAPoSDA の省電力効果に与える影響について考察する．

2 RAPoSDA の概要

2.1 構成

RAPoSDA[3] はキャッシュメモリとディスクドライブで構成され，ディスクドライブは更に少数のキャッシュディスクと多数のデータディスクに分けられる．キャッシュメモリは個別の電源系統に接続された複数のメモリからなり，UPS 等で断電対策されているものとする．キャッシュディスクは常に回転しており，クライアントからの読み出し要求に対するキャッシュとして働くのに対し，データディスクは一定時間アクセスがなければ回転を停止し，それによりストレージシステム全体での省電力化を実現している．

また，キャッシュメモリとデータディスクはプライマリ・バックアップ構成によりデータを二重化している．RAPoSDA は一つのキャッシュメモリと複数のデータディスクを紐付けてこれを論理的なグループとして扱う．あるグループに所属するデータディスク群をディスクグループ（Disk Group: DG）と呼び，一つの DG に含まれるデータディスクの数は固定値とし，これを N_{DG} で表す．

2.2 データ配置

これまで報告してきた RAPoSDA では，データはキャッシュメモリとデータディスクのそれぞれで二重化されている．キャッシュメモリでは，プライマリデータは予め格納先ディスクを割り当てられており，そのディスクが所属する DG に紐付くキャッシュメモリのプライマリデータ格納領域に書き込まれる．バックアップデータはそれとは別のキャッシュメモリのバックアップデータ格納領域に書き込まれる．一方，データディスクでは Chained declustering[4] を採用しており，i 番目ディスクのプライマリデータに対応するバックアップデータは，$(i+1) \bmod N_{DD}$ 番目のディスクに格納される．ここで，N_{DD} はデータディスクの総数である．

3 省電力を考慮したデータ複製数増加時におけるデータ配置

本報告ではデータを三重化させた場合のデータ配置方法について述べる．

3.1 データディスクにおける三重化時のデータ配置

データディスクにおける三重化時のデータ配置は，Chained declustering を次のように拡張した方式を採用する．i 番目ディスクのプライマリデータ P_i に対する r 番目の複製データ（$r = 0$ はプライマリとする）R_i^r は，$i + r \bmod N_{DD}$ 番目のディスクに格納される．

3.2 キャッシュメモリにおける三重化時のデータ配置

キャッシュメモリ上のデータ配置方法を工夫することでデータディスクへのアクセスタイミングを制御し，省電力効果の維持を図る．データ配置では Chained declustering を応用するが，ここでは以下のようにプライマリデータに対応するキャッシュメモリの割り当て方法が異なる二つのアプローチを検討する．

(a) Disk Group Aggregation（DGA）

(b) Cache Striping（CS）

(a) の DGA では，同じ DG に所属するプライマリデータは同一のキャッシュメモリのプライマリ領域に書き込まれる．複製データはキャッシュメモリのプライマリ領域の単位で Chained declustering により配置先のキャッシュメモリが決定される．すなわち，i 番目ディスクの複製 r（$r = 0$ はプライマリとする）のデータが格納されるキャッシュメモリの番号 j とその複製領域 r は，次の式によって求まる．

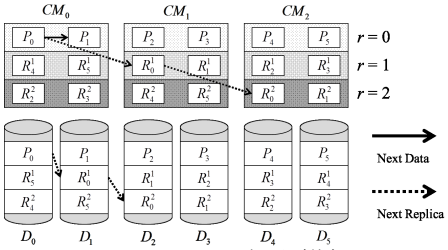
A Study of Effects of Increasing Data Replicas on RAPoSDA

Satoshi HIKIDA† Hieu Hanh LE† Haruo YOKOTA†

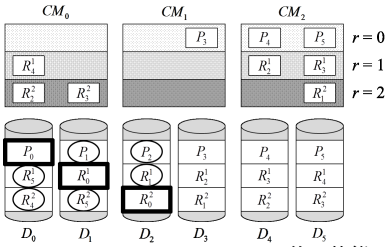
†Dept. of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology

†hikida@de.cs.titech.ac.jp †hanhlh@de.cs.titech.ac.jp

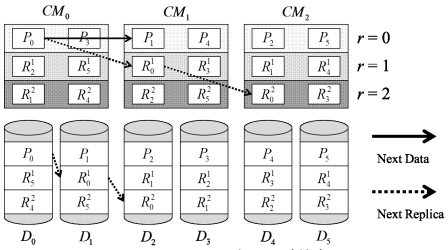
†yokota@cs.titech.ac.jp



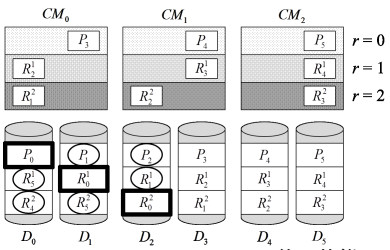
(a) DGA のデータ配置



(b) DGA のオーバーフロー後の状態



(c) CS のデータ配置



(d) CS のオーバーフロー後の状態

図 1: DGA と CS のデータ配置例.（$N_{CM} = 3$，$N_{DG} = 2$，$N_{DD} = 6$，データは三重化されている）

$$j_r = \begin{cases} \lfloor i_0 / N_{DG} \rfloor \bmod N_{CM} \, (r = 0) \\ j_{r-1} + 1 \bmod N_{CM} \, (otherwise) \end{cases} \quad (1)$$

ここで，i_0 は r 番目の複製データに対するプライマリデータが格納されているディスクの番号であり，N_{CM} はキャッシュメモリの総数である．

(b) の CS では，ディスクグループのデータ毎にまとめることはせず，キャッシュメモリへの割り当てはストライピングによって決定される．すなわち，i 番目ディスクの複製 r（$r = 0$ はプライマリとする）のデータが格納されるキャッシュメモリの番号 j とその複製領域 r は，次の式によって求まる．

$$j_r = \begin{cases} i_0 \bmod N_{CM} \, (r = 0) \\ j_{r-1} + 1 \bmod N_{CM} \, (otherwise) \end{cases} \quad (2)$$

4 キャッシュオーバーフローの違いによる影響の考察

図 1(a)，図 1(c) はそれぞれ DGA および CS のデータ配置例を示している．図中の実線矢印は次のプライマリデータの位置を示しており，破線矢印は r 番目の複製データの位置を示している．また，図 1(b)，図 1(d) では，キャッシュメモリ上のデータがオーバーフローし，P_0 およびその複製データ（R_0^1，R_0^2）に対応するディスクをスピンアップさせてバッファデータを書き込んだ後の状態を表している．図中の矩形枠はその領域のデータが契機となってディスクのスピンアップが発生していることを表し，円形で囲まれている部分は，スピンアップのタイミングで同時に書き込まれたデータ領域を表している．

図より CS では全てのキャッシュメモリのすべてのバッファ領域でバランス良く空き容量が確保出来ているのに対し，DGA では CM_0 の $r = 2$ 領域および CM_2 の $r = 0, 1$ 領域のデータはそのまま残ってしまうため，これらの領域が直ぐにでもオーバーフローしディスクアクセスが頻発してしまう可能性が高い．そのため DGA では CS に比べ不要なスピンアップも増えてしまう可能性が高く，省電力効果が減少してしまうことが予想される．

5 まとめおよび今後の課題

本報告では我々が提案しているストレージ省電力化手法である RAPoSDA について，データを三重化させる際のデータ配置方法を検討し，それが RAPoSDA に与える影響について考察をおこなった．

今後の予定としては，提案したデータ配置方法を RAPoSDA に適用した場合に，省電力効果と応答性能性に与える影響についてシミュレーション実験による評価を行う．更にデータの複製数を三重化以上に増加させた場合についても同様の評価を行う予定である．

謝辞

本研究の一部は，日本学術振興会科学研究費補助金基盤研究 (A)(# 22240005) の助成により行われた．

参考文献

[1] Apache Hadoop Project. `http://hadoop.apache.org/`.

[2] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google file system. *SIGOPS Oper. Syst. Rev.*, Vol. 37, No. 5, pp. 29–43, 2003.

[3] Satoshi Hikida, Hieu Hanh Le, and Haruo Yokota. A Power Saving Storage Method That Considers Individual Disk Rotation. In Proc. *DASFAA*, Vol. 7239/2010, pp. 138–149, April 2012.

[4] Hui-I Hsiao and David J. DeWitt. Chained declustering: A new availability strategy for multiprocessor database machines. In Proc. *ICDE*, pp. 456–465, Washington, DC, USA, 1990. IEEE Computer Society.