1G-03

格納データの継続的な増加に対応した大規模ストレージシステムの省電力化手法

細岡　晟†　　長谷部　浩二‡　　加藤　和彦‡

†筑波大学 情報学群 情報科学類　　‡筑波大学大学院 システム情報工学研究科 コンピュータサイエンス専攻

1　背景と目的

近年、クラウドコンピューティング技術の発展に伴い、YouTube や Flickr 等に代表される大規模ファイル共有サービスが普及している。一方で、それらの運用基盤となるデータセンタにおける運用コスト、特にストレージシステムの電力消費の増加が大きな問題になっている。

これまで数多くの省電力化手法が提案されてきた。しかしながら、大規模ファイル共有サービスに見られるような、大量のファイルが継続的にアップロードされ、またそのアクセス頻度が時間とともに急激に変化するような環境は、十分想定されてこなかった。こうした課題に取り組んだ研究として論文 [1] がある。しかしながらこの手法では、扱うファイルは固定サイズのみと仮定していた。また、応答性能を向上させるために広く用いられるキャッシュが導入されていなかった。そこで本研究では、論文 [1] の手法をもとに、ファイルサイズが異なる場合にも対応できるような改良を加える。また、キャッシュを導入することにより応答性能の向上を図り、より実用的な省電力ストレージの構築を目指す。

2　関連研究

これまでに提案されてきたストレージシステムの省電力化手法の多くは、アクセス頻度が高いデータを特定のディスクに集約し、アクセス頻度の低いディスクをスピンダウンさせることにより、消費電力を削減するというものである。代表的なものとして MAID（Massive Arrays of Idle Disks）[2] や PDC（Popular Data Concentration）[3] といった手法が挙げられる。

大規模なファイル共有サービスなどで用いられるストレージシステムでは、多数の利用者によりデータのアップロードが継続的に発生する。また格納ファイルは、時間経過とともに急激にアクセス頻度が低下する。これまでに提案された手法の多くは、このような特徴について十分に考慮していなかった。

こうした課題に応える試みとして、研究 [1] が挙げられる。その基本的なアイデアは、クライアントからアップロードされたファイルはディスクアレイ上に逐次追加される。格納データのアクセス頻度はアップロードからの時間経過ともに低下していく。また、定期的に、別のディスクに格納されているアクセス頻度の低いファイルと、アクセス頻度が高いファイルを交換することにより、アクセス頻度が低いファイルをディスクに集約していく。しかしながら、こうしたファイル交換によるアプローチは、ファイルサイズが全て同一サイズであると仮定したものである。また、キャッシュの導入など、実用性を向上させるための課題が残されていた。

3　提案手法

本手法では、長谷部らの手法をもとに、ファイルをアクセス頻度別に新規ディスクへ移動することにより、ファイルを集約する手法を提案する。ファイルを新規ディスクに移動することでさまざまなファイルサイズについて対応できる。また応答性能の向上させるため、キャッシュとして SSD（Solid State Drive）を組み合わせる。

本提案手法で想定するストレージシステムの構成を図 1 に示す。ここでは大規模ファイル共有システムの利用を想定し、数千から数万台規模のディスクがシステムに搭載される。

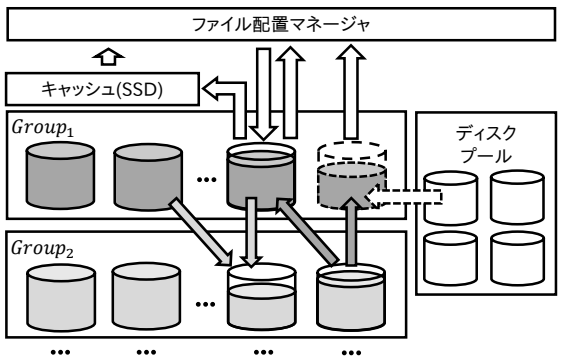

図 1　システム構成

システム内のディスクは、複数のディスクからなるディスクアレイと空のディスクからなるディスクプー

Power Saving in Large-Scale Storage Systems for Dealing with Continuous Increase of Stored Data

† Sei HOSOOKA University of Tsukuba College of Information Science
‡ Koji HASEBE University of Tsukuba Department of Computer Science
‡ Kazuhiko KATO University of Tsukuba Department of Computer Science

ル、キャッシュの 3 つの集合から構成される。さらにディスクアレイは格納ファイルのアクセス頻度別に 1 から N のグループに分類する。システム内のディスクとクライアントから発せられる I/O 要求を全てファイル配置マネージャが管理する。またファイル配置マネージャは、ファイルのハッシュ値とディスクの固有 ID、ファイルに対するアクセス回数を組み合わせたインデックスを作成し、それを用いてファイルとディスクを管理する。

クライアントから書き込み要求を受けると当該のファイルはグループ 1 に属するディスクへ書き込まれる。そのグループに属するディスクの容量に空きが無い場合は、ディスクプールより新たなディスクをグループに追加し、そのディスクへ書き込む（図 1 破線矢印)。

また定期的に格納ファイルに対するアクセス頻度を算出し、それに基づいたファイルの再配置を実行することで省電力化を図る。ここでは、[1] で提案されたアップロードからの経過時間とこれまでの累積アクセス回数より将来のアクセス頻度を予測する手法を用いる。ファイル再配置は、アクセス頻度が大きくなると予測されたファイルの移動と、将来的にアクセス頻度が低下するファイルをグループごとに集約する 2 つの手続きからなる。グループごとにアクセス頻度の許容範囲上限と下限を表す閾値 $Th_i^{up}(i = 1, ..., N)$ と $Th_i^{low}(i = 1, ..., N)$ を設定する。Th_i^{up} を超過した場合は、その閾値に応じたグループへファイルを移動する（図 1 色付き上矢印)。また Th_i^{low} 未満のファイルは、その閾値に応じたグループへ移動される（図 1 色付き下矢印)。移動先のグループに空き容量があるディスクが存在する場合は、そのディスクへ書き込み、空き容量があるディスクが存在しない場合は、新たにディスクを追加し、そこへ書き込む。

またシステムの応答性を向上と HDD の過負荷状態を防止するために、アクセス頻度が高い一部のファイルを SSD 上にキャッシュすることで省電力化を図る。キャッシュは LRU(Least Recently Used) アルゴリズムにより管理する。

4 シミュレーションによる評価

シミュレーションから本手法の評価を行う。表 1 にシミュレーションの設定項目を示す。ストレージシステム全体で HDD の容量が不足すると随時 HDD を追加する。またキャッシュディスクは SSD1 台のみとした。

表 1　設定項目

設定項目	設定値
ファイルサイズ	2.65MB
HDD 容量	1.0TB
SSD 容量	1.2TB

シミュレーションのワークロードは、[1] で用いられている写真共有サービス Flickr より収集したアクセスパターンを用いる。

シミュレーションでは省電力性能と応答時間を評価した。省電力化性能は HDD の状態遷移をシミュレートし、動作時間に占めるスタンバイ状態の比率を省電力化性能として評価する。

4.1 評価結果

提案手法と、提案手法と同様にファイルを格納し再配置を行わない方法を比較したところ、最大で 8.3% の省電力化を確認できた。またキャッシュを用いることによりファイルの再配置による応答速度の低下を最大で 1.5 秒に抑えた。

5 結論と今後の課題

本研究では、論文 [1] の手法をもとにデータの再配置を、ファイルの交換によらずに行う方法を提案するとともに、新たにキャッシュを導入する手法を提案した。実システム上のアクセスパターンを用いたシミュレーションにより、提案手法の省電力性と応答性を評価した。

今後の課題として、シミュレーションのモデルを詳細化してより実環境に近い評価を行うことが挙げられる。また省電力効果の高い閾値を見出す方法についても検討したいと考えている。

参考文献

[1] K. Hasebe, J. Ookoshi, and K. Kato. Power-saving in storage systems for cloud data sharing services with data access prediction. *IEICE Transactions on Information and Systems*, Vol. 98, No. 10, pp. 1744–1754, 2015.

[2] D. Colarelli and D. Grunwald. Massive arrays of idle disks for storage archives. In *Proceedings of the 2002 ACM/IEEE Conference on Supercomputing*, SC '02, pp. 1–11, 2002.

[3] E. Pinheiro and R. Bianchini. Energy conservation techniques for disk array-based servers. In *Proceedings of the 18th Annual International Conference on Supercomputing*, ICS '04, pp. 68–78, 2004.