動的に記録されたアクセス関連性に基づくデータ配置による省電力ストレージシステムの消費電力と性能に対する影響

入谷　優† 　横田　治夫†

† 東京工業大学 大学院情報理工学研究科　〒 152–8552 東京都目黒区大岡山 2–12–1
E-mail: †iritani@de.cs.titech.ac.jp, †yokota@cs.titech.ac.jp

あらまし　ストレージシステムで扱われるデータ量の増大に伴い，データを保持するハードディスクの消費電力の増加が問題となっている．この解決策の 1 つとして，頻繁に利用されるデータを一部のディスクに集約し，他のディスクをスピンダウンさせることでストレージシステムの消費電力を削減する手法が提案されている．しかし，スピンダウンしたディスクに対するアクセスは，スピンアップによる消費電力と遅延の増加を引き起こす．この問題は，近い時間にアクセスされるデータがスピンダウンしたディスクの多くに分散配置されている場合に顕著となる．そこで本論文では，スピンアップによる消費電力と待ち時間を削減することを目的として，同時に使われやすいデータを動的に発見・記録し，その情報を用いてそれらのデータを集約配置することを可能にする仕組みを提案する．また，ソフトウェアシミュレーションにより提案手法が消費電力と性能に与える影響について評価を行う．

キーワード　省電力ストレージ，ファイルシステム，関連性抽出

The Effect of Data Placement based on Dynamically-Logged Access Relations on Energy and Performance of Energy-Saving Storage System

Masaru IRITANI† and Haruo YOKOTA†

† Graduate School of Information Science and Engineering, Tokyo Institute of Technology
2–12–1 Ookayama, Meguro, Tokyo, 152–8552 Japan
E-mail: †iritani@de.cs.titech.ac.jp, †yokota@cs.titech.ac.jp

1. 概　　要

近年，コンピュータシステムによって扱われる情報の量が爆発的に増加し，それらの情報を保持するための消費電力の増大が大きな問題となっている．例えば，EPA [1] の報告では 2010 年に全米のエンタープライズサーバで利用されているハードディスクの台数は 2004 年と比べて 3.7 倍以上になっていると予測されている．また，ディスク数の増加に伴い 2000 年から 2006 年の間にエンタープライズサーバに取り付けられているハードディスクによって消費されるエネルギーは 3 倍になったと推測されている．このようなハードディスクによる消費電力の増大は，データセンターやファイルサーバなど多くのハードディスクを利用するコンピュータシステムでは特に大きな問題となる．

ハードディスクによる消費電力を削減するために，利用頻度の偏りを利用したデータ配置手法が幾つか提案されている．それらの手法はハードディスクを 2 つのグループに分け，一方のグループに属するディスクには比較的よく参照されるデータを，他方のグループに属するディスクにはあまり参照されないデータを配置する．これにより，ファイルアクセスに伴うディスクアクセスは前者のグループに属するディスクに集中し，殆どアクセスされなくなる後者のグループに属するディスクに関しては回転を停止させてその消費電力を削減することができる．

しかしながら，回転を停止したディスクに格納されたデータにアクセスする際には，ディスクの回転を再開させるスピンアップと呼ばれる操作が必要である．スピンアップには電力と待ち時間が必要であるため，多くの回転を停止したディスクに対してアクセスが発生すると，深刻なアクセス遅延や消費電力の増大を招く可能性が有る．この問題は，近い時間に使われるファイル同士が回転を停止しているディスクの多くに分散して配置されている場合に顕著となる．

この問題を解決する方法として，我々はこれまでアクセスログから抽出されたファイルアクセスの関連性を利用して近い時間に使われているファイル同士を同一ディスクに配置する手

法 [19] を提案してきた．しかしながら，この手法ではファイル同士の関連性を抽出するのに時間が掛かるという問題や，利用傾向の変化に追従することが難しいという問題が有る．

そのため今回我々は，ストレージシステムの省電力化及び高速化を目的とし，近い時間に利用されているファイル同士を動的に記録し管理する手法を提案する．本論文では，その手法の概要について述べた後，ソフトウェアシミュレーションにより手法を適用した際のストレージシステムの消費電力と性能に対する影響について評価を行う．

2. 関 連 研 究

ここでは関連研究について述べる．

2.1 省電力ストレージ

複数ディスクによって構成されるストレージシステムを省電力化する手法については，これまでに多くの研究が為されている．その一例として MAID (Massive Array of Idle Disks) [3] が良く知られている．MAID では一部のディスクをアクセスされたデータを一時的に格納するキャッシュディスクと呼ばれるディスクに割り当てる．多くのアクセスをそれらのディスクに集中させることで他のデータディスクと呼ばれるディスクへのアクセスを削減し，データディスクをスピンダウンさせて消費電力を削減する．RAPoSDA [11] は MAID を拡張した手法であり，プライマリ・バックアップ構成により冗長性を確保しながら，ディスクの回転状況を考慮することで信頼性の向上と省電力化を実現する．

これらの省電力ストレージシステムでは，アクセス頻度によるデータ配置により省電力化を実現しているが，データやファイル同士の関連性については利用していない．これらの手法では，本手法を適用してアクセス関連性に基づくデータ配置を行うことで，より性能及び省電力効果を高めることができると考えられる．

2.2 ファイル先読み

ファイル同士のアクセスパターンをもとに関連の有るファイル同士を見付け，ファイルシステムの性能を向上させる手法としてファイルの先読み手法が研究されている．

ファイルシステムの先読みでは，あるファイルの次にアクセスされたファイル (Successor) を利用する手法 (last-successor) が広く用いられている．[8] この手法では，ファイルアクセスで同じパターンが繰り返され易いという特徴を利用し，各ファイルについてそのファイルの次にアクセスされたファイルを記録しておく．そして，あるファイルがアクセスされた際に，前回のアクセスで次にアクセスされたファイルを事前に読み出すことで，同一アクセスパターンが発生した場合にファイルシステムの性能を向上させることができる．

Griffioe らはこの手法を元に，重み付きのグラフ構造を利用する改良手法が提案している．[5] この手法では，確率グラフによってアクセス系列を記録することで，複数ファイルとの関連性を記録して利用することが可能である．

Noah [2] も last-successor を改良した手法である．Noah では，Successor の信頼度を記録するカウンタを用意し，信頼度が一定の基準に到達するまで Successor の書き換えを保留することで，一度しか現れないようなアクセス系列による影響を抑えることができる．

ファイルの属性から関連性を抽出する手法としては FARMER [14] が挙げられる．FARMER はログに記録されたアクセス履歴からアクセス時刻やパス，ユーザ等の情報を収集し，ファイル同士の関連性を抽出する．

これらのファイル先読み手法は，アクセスパターンから近い将来アクセスされる可能性の高いファイルを予めディスクから読み出しておくことによって，ファイルシステムのアクセス速度を向上させる．その一方で，ストレージシステムにおけるデータ配置や消費電力の削減についてはその対象とされていない．

2.3 関連性に基づくデータ配置

ファイル同士の関連性を抽出する手法の 1 つとして FI 法 [17] [13] が有る．この手法ではファイルアクセスログから一定時間毎に区切り，その 1 つ 1 つをトランザクションと見做してデータマイニングアルゴリズムを適用することで，近い時間にアクセスされたファイル同士を見付けることができる．

この FI 法によって抽出されたファイル同士の関連性をストレージシステムの省電力化に用いる手法としては，XAPC [15] と PLECO [10] [19] が挙げられる．XAPC は，複数ディスクに跨がる巨大な XML ファイルへのクエリ要求に対して FI 法を用いた関連性抽出を行う．それにより同時に参照され易いと判断された XML の部分木を同一ディスクに格納することで，参照時に必要なディスクのスピンアップを抑制する．PLECO はファイルサーバのアクセスログに対して FI 法を適用し，関連性の高いファイル同士を集約配置することで，省電力ストレージシステムの消費電力と性能を向上させる．

これらの手法は，提案手法と同様に関連性に基づくデータ配置を行うことで，ストレージシステムの省電力化と性能向上を実現する．その一方で，多数のアクセスが発生しているような場合では，ログに対するデータマイニング処理に長時間掛かってしまう可能性が有る．本手法は，ファイル同士の関連性を動的に記録することによって，これらの手法における問題点を解決する．

2.4 ストリーム処理におけるデータマイニング

動的にアイテム同士の関連性を抽出する手法として，ストリーム処理におけるデータマイニング手法が数多く提案されている．[6] 例えば，Karp ら [7] は，最小サポート値 θ に対して $O(\frac{1}{\theta})$ の空間計算量でストリームデータから頻出アイテム集合を抽出する手法を提案している．また，Lossy Counting [9] や FPDM [16] は，一定の誤差を認めることにより出現頻度が一定以上であるようなアイテム集合をストリーム処理によって抽出することを可能にする手法である．

3. ストレージシステム

ここでは，本手法を適用する前提となるストレージシステムについて述べる．

本手法の適用対象となるストレージシステムは，回転状況の

制御が可能な複数のハードディスクによって構成されるものとする．ストレージシステムはそれを構成するディスクを，比較的高い頻度でアクセスされるデータを格納するためのディスクとあまりアクセスされないデータを格納するためのディスクに二分する．本論文では前者を高頻度ディスク，後者を低頻度ディスクと呼ぶ．低頻度ディスクに関しては，一定時間以上アクセスが無ければスピンダウンさせて電力消費を抑える．

ストレージシステムはファイル毎に一定時間内のアクセス回数を記録しており，低頻度ディスクに有るファイルのアクセス回数が閾値を越えた場合は高頻度ディスクに移動させる．逆に，一定時間アクセスの無かったファイルが高頻度ディスクに存在する場合には低頻度ディスクへ再配置する．

4. 提案手法

アクセス頻度を元にデータ配置を行うことによってストレージの省電力化を実現する既存の手法では，低頻度ディスクにおけるデータ配置については十分に考慮されていない．このため，近い時間帯で利用されやすいデータが複数の低頻度ディスクに分散して配置される結果，多数のスピンアップが必要になる可能性が有る．多数のスピンアップはスピンアップ自体による消費電力の増加とスピンアップ待ちによるアクセス遅延の増大を引き起こす．

近い時間で利用されやすいファイル同士を同じディスクに配置することができれば，この問題による影響を最小限に抑えることができる．今回我々は，ファイルのアクセス系列からアクセス関連性の高いファイル同士を動的に抽出する新しい手法 DACE を提案する．

4.1 アクセス関連性

プロジェクト管理やソフトウェア開発などのある作業に使われるファイル同士は，近い時間に参照されやすいという性質を持っている．[17] [13] 例えば，あるソフトウェア開発プロジェクトにおいて，ソフトウェアのコードファイルとそれをレビューする際に用いられる文書ファイルがほぼ同時に参照される傾向が有る，というような事例が考えられる．本論文ではこの性質をアクセス関連性と呼び，この事例のようにあるファイルに対して別のあるファイルが近い時間に参照されやすいとき，2 つのファイルはアクセス関連性が高い，と言う．

あるファイルに対するアクセスの前後 T 秒以内でアクセスが発生することを，提案手法ではアクセスの共起と呼ぶ．ただし，T は事前に設定される時間の閾値である．

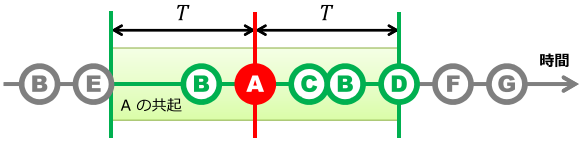


図 1　アクセスの共起

提案手法では，あるファイルとアクセスが共起したファイルをアクセス関連性の高いファイルとして扱う．どのような場合にアクセスが共起したと見做すかについて，その例を図 1 及び図 2 に示す．図中の円はファイルシステムに対して要求された 1 つのファイルアクセスを表しており，円内部のアルファベットが同じものは同一のファイルに対するアクセスを表現している．また，右にあるアクセスほど新しい．

図 1 中央の A に対するアクセスのように前後 T 秒に自身（この例では A）に対するアクセスが存在していない場合，その前後 T 秒で発生したファイルアクセス全てを共起したものと見做す．なお，同一のファイルが複数回共起していた場合でも，1 回共起したものとして扱う．ファイルの一部分に対して非連続的なアクセスが行われる場合や NFS のようなステートレスなファイルシステムが利用されている場合においてはアクセスが多く発生することがあるが，このような原因で発生した多数のファイルアクセスの存在は必ずしもそのファイルとのアクセス関連性が強いことを意味しないからである．図の例では B, C, D の各ファイルがそれぞれ 1 回ずつファイル A とアクセスが共起したものと見做される．

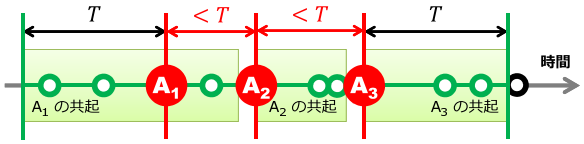


図 2　連続する複数アクセスでの共起

図 2 の A_1, A_2, A_3 のように単一ファイル（この例では A）に対して T より短い時間間隔で複数回アクセスが発生している場合は，各アクセスについて次の自身へのアクセスまでに発生したアクセスを共起したものと見做す．この規則により，単一のファイルアクセスが共起したアクセスとして複数回記録されることを防ぐことができる．

4.2 再配置バッファ

提案手法では，ストレージシステムを構成するディスク毎に再配置バッファと呼ばれるバッファ領域を用意する．ストレージシステムはファイルの再配置を行う際に，再配置先のディスクを決定した直後に再配置処理を行うのではなく，一旦再配置先のディスクに対応する再配置バッファにそのファイルを登録し，再配置の実行を保留する．ファイルの再配置はシステムがクライアントからのファイルリクエストを処理しておらず移動元と移動先のディスクが共に回転状態にある場合にのみ実行される．これにより，ファイルの再配置を実行することにより本来不必要なスピンアップが発生してしまうことを防ぐことができる．

4.3 データ構造

表 1　アクセス関連性テーブル

ファイル識別子	共起回数
B	18
D	8
E	4
I	2
L	5
⋮	⋮

表 2　単共起ファイルリスト

単共起ファイルリスト
C
A
K
F
G
⋮

各ファイルに対して，アクセス関連性テーブルと単共起ファイルリストという 2 つのデータ構造を用意する．アクセス関連性テーブルは，表 1 で示したような構造を持ち，2 回以上共起したファイルの識別子とその回数を記録するために用いられる．1 回のみ共起が発生したファイルの識別子は，表 2 に示したような単共起ファイルリストによって別に管理される．このリストはキューとして管理されており，挿入された順番が保持されている．

ファイル識別子に 8 バイト，回数の記録に 4 バイトそれぞれ必要であると仮定すると，アクセス関連性テーブルは 1 レコードあたり 12 バイト，単共起ファイルリストは 1 要素あたり 8 バイトを必要とする．例えば，アクセス関連性テーブルと単共起ファイルリストが共に 128 ファイルまでを保持する場合には約 2.5 キロバイトの領域が必要である．この例からも分かるように，これらのデータ構造を保持するために必要な領域は小さいため，ファイルシステムの未使用領域や空き領域，もしくはメモリ上で管理することも十分可能であると考えられる．

4.4　アクセス関連性の記録

ファイルシステムに対するアクセスを管理するファイルサーバは，直近 $2T$ 秒に発生したアクセスについてその対象となったファイルと時刻を記録しておく．ファイルサーバはアクセスされてから T 秒経過したファイルについて，記録されている前後 T 秒のアクセスを共起したアクセスと見做し，アクセス関連性テーブルまたは単共起ファイルリストに記録する．

共起したファイルの識別子がアクセス関連性テーブルに記録されている場合，対応するレコードの共起回数の値をインクリメントする．アクセス関連性テーブルに記録されていないが単共起ファイルリストには記録されている場合，そのファイルの識別子を単共起ファイルリストから削除し，代わりにアクセス関連性テーブルに新しいレコードを追加する．その際，共起回数の値は 2 とする．アクセス関連性テーブルと単共起ファイルリストの何れにも対応するエントリが存在しない場合には，共起したファイルの識別子を単共起ファイルリストに追加する．

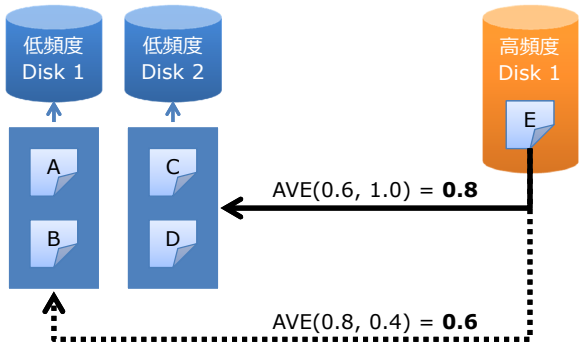
アクセス関連性テーブルや単共起ファイルリストを保持するためのディスク領域またはメモリ領域に制限が有る場合も考えられる．アクセス関連性テーブルのレコード数が上限を越えた場合は，Karp [7] らの提案した頻出アイテム抽出手法で用いられるアルゴリズムを利用して追い出し処理を行う．すなわち，全てのエントリの共起回数の値をデクリメントし，共起回数が 1 となったレコードが有ればそれを取り除き，代わりに単共起ファイルリストにそのファイル識別子を追加する．単共起ファイルリストに登録されているファイル識別子の数が上限を越えた場合は，要素数が上限と等しくなるまで最も古い要素から順に削除する．

また，短い時間で同じファイルに対してアクセスが発生した場合，そのファイルと共起したファイルが複数回重複して記録されてしまう可能性が有る．そのため，アクセス関連性テーブルと単共起ファイルリストの更新処理に関して次のような制約を設ける．

T 秒前から 0 秒前までのアクセスについては古いアクセスから新しいアクセスの順に処理を行う．もしその処理で自分自身がアクセスされていた場合，それ以降に発生したアクセスの記録を中止する．また，$2T$ 秒前から T 秒前までの間に自分自身に対する別のアクセスが有った場合，$2T$ 秒前から T 秒前までのアクセスは全てそのアクセスによって記録されているはずであるため，$2T$ 秒前から T 秒前までのアクセスについては記録を行わない．これらの制約により，連続する同一ファイルへのアクセスが共起の記録を受け持つ範囲は図 2 と同様になり，1 つのアクセスが同一ファイルに対して重複して記録されることを防ぐことができる．

4.5　アクセス関連性の適用

あるファイルに対して記録されたアクセス関連性は，そのファイルが高頻度ディスクから低頻度ディスクに対して再配置される際に，その再配置先のディスクを決めるために利用される．



E のアクセス関連性テーブル

ファイル識別子	共起回数	関連性スコア
A	4	**0.8**
B	2	**0.4**
C	3	**0.6**
D	5	**1.0**
トランザクションカウンタ		5

スコア計算

図 3　アクセス関連性の適用

アクセス関連性の適用方法を図 3 に示す．まず，それぞれの低頻度ディスクについて，その再配置バッファに存在するファイルとの共起回数の平均値を計算する．対応する共起がアクセス関連性テーブルに存在しない場合には，その共起回数の値は 0 であるものとして計算される．そして，このようにして得られた共起回数の平均値が最も大きくなった低頻度ディスクをそのファイルの再配置先ディスクとする．

表 3　評価に用いたハードディスクモデル

参考モデル	HGST Deskstar 7K2000 [12]
容量	2 TB
キャッシュサイズ	32 MB
キャッシュアルゴリズム	LRU
消費電力 (ACTIVE)	11.1 W
消費電力 (IDLE)	7.5 W
消費電力 (STANDBY)	0.8 W
スピンダウン消費エネルギー	35.0 J
スピンアップ消費エネルギー	450.0 J
スピンダウン時間	0.7 秒
スピンアップ時間	15.0 秒

このとき，単共起ファイルリストに登録されているファイルに関しては，共起回数への算入及び関連性の適用を行わない．これは，共起が 1 回のみであるファイルは，その共起が偶然発生した可能性も高く，必ずしもアクセス関連性が存在することを意味しないからである．

なお，関連性の適用は再配置バッファに存在するファイルに対してのみ行われる．これは，ファイルシステムで管理されている全てのファイルについて格納されているディスクを調査した上で関連性を計算すると，特にファイル数が膨大である場合に処理時間が非常に長くなってしまうためである．再配置バッファに存在しているファイルはファイルシステムに格納されているファイル数と比べるとごく一部ではあるが，アクセス関連性が有るファイルは近いアクセスパターンを示し易いため同じタイミングで再配置バッファに格納されている可能性が高く，再配置バッファに登録されているファイルとの関連性だけでも十分効果的なファイル配置が可能である．

5. 評　　価

提案手法がストレージシステムの消費エネルギーとアクセス時間に与える影響について，ソフトウェアシミュレーションによる評価を行った．シミュレーションには，引田ら [18] によって開発されているディスクシミュレータを用いた．消費エネルギーはファイルサーバで利用される全ハードディスクによる消費エネルギーの総和，アクセス時間はファイルが要求されてからその処理が完全に完了するまでの平均時間によって計測した．

5.1 評価条件

評価では，ファイルサーバをハードディスク 16 台で構成し，そのうち 2 台を高頻度ディスク，残りの 14 台を低頻度ディスクとした．また，低頻度ディスクについては 60 秒以上アクセスが無ければスピンダウン処理を行うものとした．評価に用いたハードディスクの仕様を表 3 に示す．

評価に用いるワークロードとしては，ハーバード大学の EECS で利用されていた NFS サーバのログ [4] を利用した．この評価では，ログから 2001 年 9 月 1 日から同月 10 日までに発生したファイルに対する READ 及び WRITE の要求を抽出しワークロードとして利用した．このワークロード中には，36,722 ファイルに対する 59,660,976 アクセスが含まれている．

ファイル配置の違いによる影響のみを評価するため，ファイルは各ディスクがほぼ同数を保持するよう最初にハッシュ値で分散配置された後はディスク間での交換によってのみ移動されるものとした．また，他のファイル再配置を行う手法においても，再配置バッファを用いて DACE と同様にディスクの回転状況に応じた再配置の実行を行うものとした．

5.2 評価対象

次の 5 つの手法について評価を行った．

STRIPE　ファイルを各ディスクに対してハッシュ値によりほぼ均等になるよう割り振り，初期配置から移動しない，最も基本的なストレージシステム．

FREQ　STRIPE において，60 分で 2 回以上アクセスが有れば高頻度ディスクに，逆に 60 分で全くアクセスが無ければ低頻度ディスクに再配置を行う．配置先のディスクはファイル識別子のハッシュ値を基に決定される．

SUCCESSOR　FREQ において，ファイルの先読み手法として広く用いられている last-successor をファイル配置決定に適用させたものである．last-successor は全てのファイルについて，そのファイルの次にアクセスされたファイル（successor）を記録しておき，あるファイルに対してアクセスが発生した際にそのファイルの successor を予めディスクから読み出しておくことで，ファイルに対するアクセス遅延を削減する手法である．今回の評価ではこの手法を応用し，そのファイルの次にアクセスが有ったファイルをアクセス関連性が高いファイルと見做して PLECO や DACE と同様のアクセス関連性適用を行う．あるファイル A の successor が B の場合，A に対する B の関連性スコアが 1.0 と見做して提案手法と同様のファイル配置決定を行う．

PLECO　アクセスログ解析により抽出した関連性を利用する手法である [10]．ファイルアクセスログを 600 秒毎に区切り，その 1 つ 1 つをトランザクションと見做して相関抽出手法である Apriori アルゴリズムを適用し，長さ 2 の相関ルールを抽出する．仮に $\{A\} \rightarrow \{B\}$ というルールが抽出された場合，A に対する B の関連性スコアがそのルールの信頼度と等しいものと考えて DACE と同様にアクセス関連性を適用し，ファイル配置を決定する．相関ルールは 24 時間毎に再計算され，最新のものに更新される．最小サポート値は 0.004，最小信頼度は 0.8 とした．これは，最小サポート値を 0.004 または 0.006，最小信頼度を 0.2, 0.4, 0.6, 0.8 から選択した組み合わせのうち，最も平均アクセス時間が小さくなった組み合わせである．

DACE　提案手法に基き，FREQ において前後 600 秒で共起したアクセスを記録し，その情報に応じて低頻度ディスクを決定する．なお，アクセス関連性テーブルの最大レコード数は 256 ，単共起ファイルリストの最大要素数は 409 である．単共起ファイルリストの最大要素数は，ファイル名，ファイルサイズ，ファイル先頭位置，アクセス関連性テーブルなどと共に 8KB で保持可能な最大の数を算出して得られた数字である．

5.3 評価結果

各手法において，ディスクがスピンアップした回数を評価した結果を図 4 に示す．単純にディスクをスピンダウンさせた場合，43,869 回のスピンアップが発生してしまう．それに対

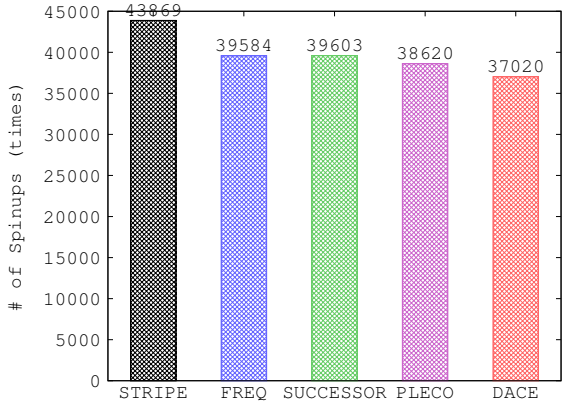


図 4　スピンアップ回数

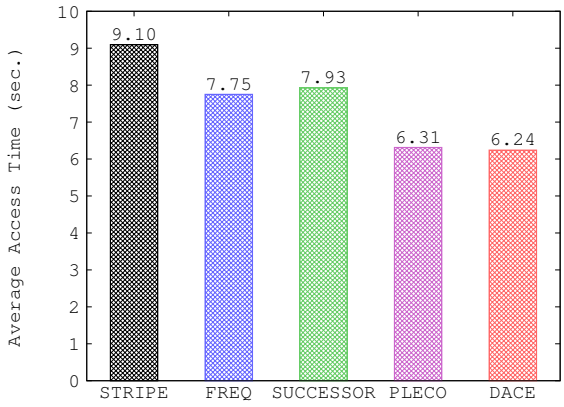


図 6　平均アクセス時間

して，FREQ はアクセス頻度に応じて再配置を行うことでスピンアップ回数を 9.8% 削減している．SUCCESSOR では直後のアクセスに基づくアクセス関連性を利用しているものの，FREQ に比べて僅かながら逆にスピンアップ回数が増加する結果となった．一方で PLECO は FREQ に比べて 2.4% スピンアップ回数を削減できている．今回の提案手法である DACE は 5 つの手法の中で最も低いスピンアップ回数を示した．これは，FREQ に対して 6.5% の削減である．

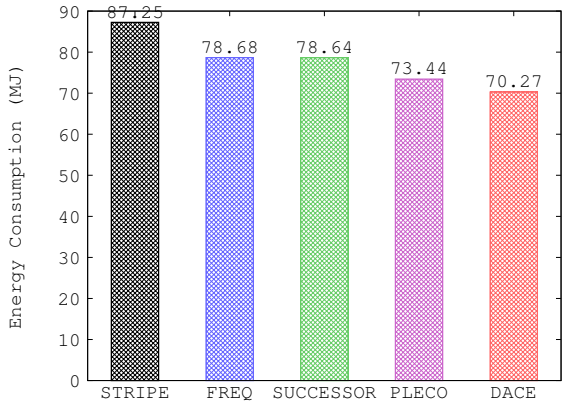


図 5　消費エネルギー

このようなスピンアップ回数の傾向が，省電力ストレージシステムの消費エネルギーにどのような影響を及ぼすかについて評価を行った．各手法において，全てのディスクが消費したエネルギーの総和を図 5 に示す．

最も消費エネルギーが高かったのは STRIPE であり，その消費エネルギーは 87.25 MJ となった．FREQ は STRIPE に比べて 9.8% 少ない 78.68 MJ，SUCCESSOR は FREQ より僅かに低く 78.65 MJ であった．アクセス関連性を用いる PLECO は FREQ に比べて 6.7% 低い 73.44 MJ となった．提案手法である DACE はここでも最も小さい値である 70.270 MJ を示した．これは，FREQ に比べて 10.7% の削減である．

手法の違いによるストレージシステムの性能への影響についても評価を行うため，平均アクセス時間についても比較を行った．平均アクセス時間は，クライアントがアクセスを要求してからその処理が完了するまでの平均時間によって定義される．その結果を図 6 に示す．

平均アクセス時間についても STRIPE が最も大きな値を示し，9.10 秒であった．FREQ ではその時間が 14.8% 削減され，7.75 秒であった．SUCCESSOR では FREQ よりも平均アクセス時間が 2.3% 伸びてしまう結果となった．アクセス関連性をファイル配置に用いる PLECO と DACE は共に他の手法と比べて良い結果を示した．PLECO と DACE は FREQ と比較して，それぞれ 18.6%，19.5% アクセス時間を削減することができた．

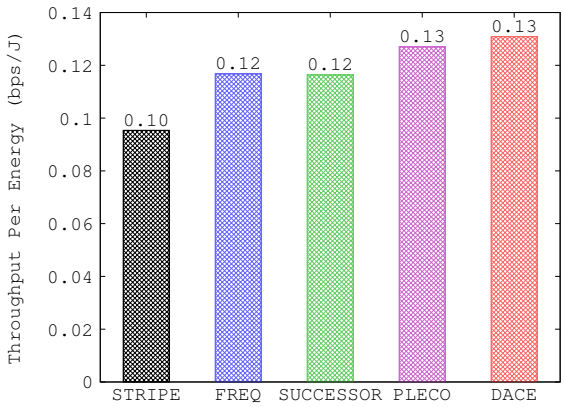


図 7　エネルギー比スループット（TPE）

省電力ストレージシステムでは，消費電力と性能はトレードオフの関係である．この両者のバランスについて評価を行うため，電力比性能の指標としてエネルギー比スループット（TPE）を算出した．エネルギー比スループットは，単位エネルギーあたりストレージシステムがどの程度のスループットを提供できているかを測るもので，消費エネルギーを e，リクエスト数を n，$i(1 \leqq i \leqq n)$ 番目のファイルリクエストで s_i ビットのファイルに対し，要求を開始してから t_i 秒でアクセスできたとすると，次の式で算出される．

$$TPE = \frac{1}{en} \sum_{i}^{n} \frac{s_i}{t_i} \tag{1}$$

この結果を図 7 に示す．

ここでは，DACE が最も高い値となった．このことから，今回比較した手法の中では DACE が最も高い電力比性能を示したと言える．また，PLECO についてもエネルギー比スループットは他の手法と比べて高い値となっており，アクセス関連性をファイル配置に適用することで，適用することにより省電力ストレージシステムの電力比性能を高められることが分かった．

5.4 考　　察

図 5 で示した消費エネルギー，図 6 で示した平均アクセス時間は何れも，図 4 で示したスピンアップ回数と近い傾向を示している．このことから，省電力ストレージシステムにおいては低頻度ディスクのスピンアップ回数を抑制することがストレージシステム全体の電力消費の削減に繋がることが分かった．

何れの評価においても PLECO と DACE は今回評価した全ての項目において，他の手法と比べて良い結果を示した．このことから，PLECO や DACE のように前後一定時間で発生した複数のアクセスからアクセス関連性を抽出し，低頻度ディスク間でのファイル配置の決定に利用することが，省電力ストレージシステムの省電力効果と性能を更に向上させるために効果的であることが示された．

PLECO と DACE を比較すると，今回の評価で用いた全ての評価項目について DACE の方が良い結果を示した．これには幾つかの要因が考えられる．

その理由の 1 つとして，DACE が動的にファイル同士のアクセス関連性を抽出できる，という特徴の存在が考えられる．

PLECO では，ファイルシステムのアクセスログにデータマイニングアルゴリズムを適用することで，アクセス関連性を抽出している．しかしながら，この方法で一度アクセス関連性を計算してしまうと，それ以降に発生したアクセスを反映させてアクセス関連性を最新のものに更新することは難しい．このため，PLECO でファイルのアクセス関連性を抽出する場合には，一定時間毎にアクセス関連性を全て再計算する必要がある．

その一方で，特にファイル数やアクセス数が非常に多い場合，データマイニングアルゴリズムの適用処理には非常に長い計算時間が必要となってしまう．実際に，100 万アクセスから，最小サポート値 0.01，最小信頼度 0.1，トランザクション時間 60 分という条件でアクセス関連性の抽出を AMD Opteron 4184 (2.8GHz x12) + 32GB RAM の環境で行った場合，ディスクアクセスの時間も含めて 442 秒の時間が必要であった．更にアクセスログが長大になった場合には，長時間に渡るアクセス関連性の計算がストレージシステムの性能に影響を与えてしまう可能性が有り，PLECO で頻繁にアクセス関連性の再計算を行うことは難しいと考えられる．

このことを踏まえて，今回の評価で PLECO は 24 時間毎にアクセス関連性を再計算することとした．しかし，計算の合間で発生したアクセスについてはアクセス関連性の計算に反映することができないため，結果として DACE と比べて十分にアクセス関連性を抽出及び適用できなかったことが DACE と PLECO の性能差に影響したのではないかと考えられる．

DACE が PLECO と比べて良い結果を示したもう 1 つの理由として，各ファイルに一定数のアクセス関連性が抽出されるという DACE の特徴が影響している可能性が挙げられる．

PLECO では最小サポート値と最小信頼度という 2 つの閾値によって抽出するルールに一定の基準が設けられる．低頻度ディスクに移動されるようなアクセス頻度の低いファイルに関するルールも抽出するには，ルールの出現頻度の閾値である最小サポート値は低い値にすることが望ましい．その一方で，最小サポート値を低くすることで，偶然発生したような有効性の低いルールまで抽出されてしまう．

このため，比較的アクセスされやすいファイルについてはノイズような相関ルールが多数抽出される一方で，数回しかアクセスされないようなファイルについては相関ルールが全く抽出されないという場合が発生してしまう．DACE では各ファイルについて関連性の高いファイルが一定数抽出される仕組みになっているため，ノイズとなる関連性が多数抽出されたり，逆に全く関連性が抽出されなかったりすることが少ない．このような特性が，DACE の方がより多く適切なアクセス関連性を適用できたことが，PLECO と比べて DACE が良い結果を示した要因の 1 つである可能性が有る．

SUCCESSOR も直後のアクセスのみを共起したものと見做し，アクセス関連性としてファイル配置に用いる手法である．しかしながら，PLECO や DACE が FREQ に対して良い結果を示している一方で，SUCCESSOR においてはスピンアップ回数が FREQ と比べて逆に増加し，ストレージシステムの消費電力とアクセス遅延を増大させてしまう結果となってしまった．このことから，直後にアクセスされたファイルのみを関連性の高いファイルと見做すことは，ストレージシステムにおけるデータ配置に利用する際には必ずしも十分とは言えないことが分かった．

今回の評価に先立って，予備評価として単共起ファイルリストを用いない場合についても同様の評価を行った．紙面の都合上それらの詳細については割愛したが，FREQ と比較して殆ど効果を上げることができなかった．それに対し，単共起ファイルリストを用いた DACE は比較対象とした他の全ての手法に対して一定の効果を示している．このことから，単共起ファイルリストを用いることでアクセス関連性テーブルのレコードが 1 回だけしか発生していない新しい共起によって追い出されることを防ぐことが，より良いアクセス関連性の抽出と適用に効果的であることも分かった．

6. まとめと今後の課題

今回我々は，省電力ストレージの性能向上と更なる省電力化を目的として，アクセス関連性として近い時間に共起したアクセスを動的に記録し，その情報に基いて利用頻度が低くなったファイルの再配置先ディスクを決定する新しい手法 DACE を提案した．この手法は，アクセス関連性の高いファイル同士を同じディスクに配置することで，それらのファイルに近い時間でアクセスが発生した場合にスピンアップが必要なディスクの数を抑制し，多数のスピンアップによる消費電力と遅延の増大を防ぐ．また，ソフトウェアシミュレーションによる評価を行い，提案手法が省電力ストレージの消費電力と遅延の両方を改

善することが示された．

最後に今後の課題について述べる．

ワークロードによって，ファイルのアクセス関連性の数やその強さなどは異なることが考えられる．その違いによって提案手法の省電力効果や性能にどのような影響が生じるかについては今回の評価では十分に検討できていない．これを解明するために，より多くのワークロードについて評価を行い，ワークロードの違いによる変化や提案手法に影響を与える要因について調査することが必要である．

また，本提案手法を実際の省電力ストレージシステムに対して実装し，実機による消費電力及び性能の評価を行うことも必要である．

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．

文　　献

[1] Report to congress on server and data center energy efficiency public law 109-431. Technical report.

[2] A. Amer and D.D.E. Long. Noah: low-cost file access prediction through pairs. In *Performance, Computing, and Communications, 2001. IEEE International Conference on.*, pp. 27 –33, apr 2001.

[3] Dennis Colarelli and Dirk Grunwald. Massive arrays of idle disks for storage archives. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, Supercomputing '02, pp. 1–11, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[4] Daniel Ellard, Jonathan Ledlie, Pia Malkani, and Margo Seltzer. Passive nfs tracing of email and research workloads. In *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, pp. 203–216, Berkeley, CA, USA, 2003. USENIX Association.

[5] J. Griffioen and R. Appleton. Performance measurements of automatic prefetching. In *Proceedings of the ISCA International Conference on Parallel and Distributed Computing Systems*, p. 16, 1995.

[6] Ruoming Jin and Gagan Agrawal. Frequent Pattern Mining in Data Streams. In Charu C. Aggarwal, editor, *Data Streams*, Vol. 31 of *Advances in Database Systems*, chapter 4, pp. 61–84. Springer US, Boston, MA, 2007.

[7] Richard M. Karp, Scott Shenker, and Christos H. Papadimitriou. A simple algorithm for finding frequent elements in streams and bags. *ACM Trans. Database Syst.*, Vol. 28, pp. 51–55, March 2003.

[8] Thomas M. Kroeger and Darrell D. E. Long. The case for efficient file access pattern modeling. In *Proceedings of the The Seventh Workshop on Hot Topics in Operating Systems*, HOTOS '99, pp. 14–, Washington, DC, USA, 1999. IEEE Computer Society.

[9] Gurmeet Singh Manku and Rajeev Motwani. Approximate frequency counts over data streams. In *Proceedings of the 28th international conference on Very Large Data Bases*, VLDB '02, pp. 346–357. VLDB Endowment, 2002.

[10] Iritani Masaru and Yokota Haruo. Effects on performance and energy reduction by file relocation based on file-access correlations. In *The 1st workshop on Energy Data Management*, March 2012.

[11] Hikida. Satoshi, Hieu Hanh. Le, and Yokota. Haruo. A power saving storage method that considers individual disk rotation. In *Database Systems for Advanced Applications*, 2012.

[12] Hitachi Global Storage Technologies. *Deskstar 7K2000 & Ultrastar A7K2000 Specification v1.4*.

[13] Y. Wu, K. Otagiri, Y. Watanabe, and H. Yokota. A file search method based on intertask relationships derived from access frequency and rmc operations on files. In *Database and Expert Systems Applications*, pp. 364–378, 2011.

[14] Peng Xia, Dan Feng, Hong Jiang, Lei Tian, and Fang Wang. Farmer: a novel approach to file access correlation mining and evaluation reference model for optimizing peta-scale file system performance. In *Proceedings of the 17th international symposium on High performance distributed computing*, HPDC '08, pp. 185–196, New York, NY, USA, 2008. ACM.

[15] Jiang Xuehua, Watanabe Yousuke, and Yokota Haruo. Data allocation based on xml query patterns to reduce power consumption. In *InternationalConference on Cloud and Green Computing*, 2011.

[16] Jeffery Xu Yu, Zhihong Chong, Hongjun Lu, and Aoying Zhou. False positive or false negative: mining frequent itemsets from high speed transactional data streams. In *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, VLDB '04, pp. 204–215. VLDB Endowment, 2004.

[17] 小田切健一, 渡辺陽介, 横田治夫. 頻出ファイル集合のアクセス時間を考慮した仮想ディレクトリ生成手法. 第 2 回データ工学と情報マネジメントに関するフォーラム (DEIM2010), 2010.

[18] 引田諭之, LE Hieu Hanh, Koh Kai Hung, 横田治夫. ストレージシステムにおける省電力効果検証のためのシミュレータ. 情報処理学会研究報告, 2010.

[19] 入谷優, 横田治夫. アクセス関連性によるストレージデータ配置の性能と消費電力. 第 152 回 データベースシステム・第 103 回情報基礎とアクセス技術合同研究発表会, 2011.