DEIM Forum 2011 C10-4

アクセス頻度情報を利用したインデックスによる消費電力への影響

入谷　優†　横田　治夫†

† 東京工業大学大学院情報理工学研究科計算工学専攻　〒 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　コンピュータシステムで扱われるデータ量は増え続けており，近年ではそれを維持するストレージの消費電力が大きな問題となっている．この問題に対し，アクセスの頻度に応じてデータを再配置することでシステム全体のストレージによる消費電力を削減する手法が提案されている．すなわち，アクセス頻度の高いデータと低いデータを別々のディスクに配置することによって，アクセス頻度の低いデータを保持するディスクの動作を抑え，ストレージの電力消費を抑える手法である．しかし，この手法についてファイルシステムのインデックスと関連付けた研究は十分になされていない．そこで我々は，ファイルシステム上に存在するインデックスにアクセス頻度に関する情報を付与することで，ストレージの消費電力を削減する新しい手法を提案する．本論文では，提案手法のアーキテクチャと手法を適用した場合の処理の流れについて述べる．また，ソフトウェアシミュレータによる評価を行い，提案手法を用いてファイルの再配置を行うことでストレージの消費電力を削減できることを示す．

キーワード　ファイルシステム, 省電力化, インデックス

Effects on Power Consumption of Storages by an Indexing Method Utilizing Access Frequency

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

Abstract　The growing amount of data handled by computer systems makes the energy consumption of storages a serious problem. In order to solve this problem, some methods which locate data depending on its access frequency have been proposed. These methods suppress movements of disks for unpopular data. However, researches which associate themselves with indice of file systems are not enough. In this paper we propose a new indexing method which maintains additional information about the access frequency of data. We show our index structure to hold the access frequency. We also evaluate our method with a software simulator and show that the file relocation using our method reduces the energy consumption of disks.

Key words　File system, Power saving, Indexing

1. はじめに

ファイルシステムが扱うファイルの数は，コンピュータの扱う情報量の増加に伴って増え続けている．その結果として，データセンターにおいては大量のファイルを扱うファイルサーバを維持するためのストレージの消費電力が大きな問題となっている．また，環境意識やエネルギーコストに対する意識の高まりも，ストレージの省電力化をますます重要にしている．

ファイルシステムで扱われるファイルの中には，頻繁にアクセスされているものとそうでないものが有り，アクセス頻度がファイル毎に大きく異なる．例えば，大量のログファイルを扱うようなファイルサーバでは，最新のログファイルには頻繁な読み書きが有る一方で，古いログファイルに対しては統計処理に使うためのアクセスがごくたまに発生するような状況が考えられる．また，研究室や企業等のグループで共同利用されているファイルサーバにおいても，現在の作業に必要な一部のファイルが頻繁にアクセスされる一方で，それ以外の大半のファイルについては殆どアクセスがなされないような状況が考えられる．

このようなデータに対するアクセス頻度の偏りを利用するこ

とでストレージの省電力化を実現する手法 [1] [2] が提案されている．また，Information Lifecycle Management (ILM) の考え方を利用して，ファイルに対して事前に定めたポリシーを適用することで配置先ストレージを決定し，ストレージ資源を有効活用することも考えられる．これらの手法は，頻繁にアクセスされているデータとそうでないデータを別々のディスクに配置し，あまりアクセスされていないデータを配置したディスクの動作を抑えることで，頻繁にアクセスされるデータへのアクセス速度を維持しつつストレージの消費電力を削減できる．

これらの手法はストレージの省電力化においては有効であるが，その一方でファイルシステムにおけるインデックスと関連させた研究は十分になされていない．ファイルのデータに対して再配置が行われた場合，ファイルシステム上のファイルに対して作られたインデックスの情報を更新する必要が有るが，再配置に伴うインデックスの更新処理に対する考察は十分なされていない．また，インデックスから得られる情報を利用したより高度なデータの再配置も可能であると考えられる．

そこで我々は，ファイルシステムにおけるインデックスに対してファイルやディレクトリへのアクセス頻度に関する情報を付与する新しい手法を提案する．

アクセス頻度情報を付与する対象としては，ファイルシステム上のファイルに対して作られたインデックスに対して付与する場合と，ファイルシステムのツリー構造に対して付与する場合が考えられる．本稿では，インデックスに付与したアクセス頻度に関する情報を利用してデータ再配置を行う場合について，その方法を述べる．また，ソフトウェアシミュレータによる評価を行い，アクセス頻度に応じてファイルの配置を変更することによってどの程度アクセス速度やディスクの消費電力に影響が生じるかについて示す．

2. 提案手法の概要

提案手法は，数台のディスクが接続されているファイルサーバを対象とする．それらのディスク上に配置されたファイルに対する検索を高速化するために，B-Tree によるインデックスが作られているものとする．このインデックスの葉ノードは各ファイルに対応し，それ以外のノードは B-Tree の他のノードに対するポインタを持つ．ポインタはディスクを越えてデータを参照することも可能であるとする．

このインデックスにおける各ノードに対してアクセス頻度情報を付与する．インデックスを利用したファイルアクセスが発生する度に付与した情報を更新及び参照し，その値の高低に応じてデータを配置するディスクを振り分ける．その結果，アクセス頻度の低いデータを保持するディスクの動作が抑えられ，ストレージの消費電力を削減する．

インデックスに対してアクセス頻度情報を付与することにより，インデックスから得られる情報をデータ再配置に用いることが可能になるため，LRU のような一般的なキャッシュ置換アルゴリズムを利用してデータの再配置を行った場合と比べて，より高度なデータ再配置が可能になる．具体的な利点としては，ファイルサイズの情報を利用することによってファイル単位でのデータの再配置が可能になることや，インデックスにおいて隣接するデータに対して同じ再配置動作を行うことでより効率的なデータの再配置が可能になることが挙げられる．

3. 提案手法のアーキテクチャ

提案手法では，ファイルシステム上のファイルへのインデックスに対して，未アクセスノード集合と呼ばれるノードの集合を付与する．また，インデックスの各ノードについて，アクセス頻度情報と呼ばれる情報を付与する．ここでは，それらの概要と利用する際の手順について述べる．

3.1 システム構成

提案手法の構成図を図 1 に示す．

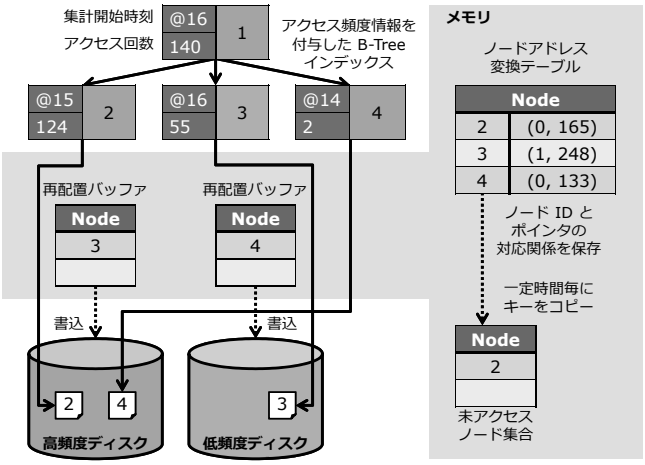

図 1　提案手法の構成図

ファイルに対する B-Tree を用いたインデックスの各ノードに対し，アクセス頻度情報と呼ばれる各ファイルのアクセス頻度に関する情報を付与する．また，各ディスクに対して再配置するデータを一時的にバッファリングする再配置バッファをディスク毎に用意する．更に，インデックスを構成する全てのノードについてその ID とディスク上での場所の対応を管理するノードアドレス変換テーブルと，一定期間でアクセスが存在しなかったノードの集合である未アクセスノード集合を付加情報として利用する．

これらの情報をもとにして，ディスク間でデータの再配置を行う．ディスクは，比較的頻繁にアクセスされているファイルを保持するディスクである高頻度ディスクと，あまりアクセスされていないファイルを保持するディスクである低頻度ディスクに分けて利用される．

インデックスの各ノードにアクセス頻度情報を配置することによって，インデックス構造やノードの持つ情報を同時に利用して高度なデータ配置を行うことが可能になる．本論文ではインデックスのノードとして，ファイルシステム上の B-Tree の各ノードを利用した場合について取り上げるが，ファイルシステムのツリー構造を構成する各ノードに本手法を適用した場合には，インデックスにアクセス頻度情報を付与することでデータ配置の判断に必要な情報をより効率的に取得できるようになると考えられる．また，アクセス頻度情報が各ノードに分散し

ていることにより，複数ディスクが並行してアクセス頻度利用した処理をしやすくなることも利点として挙げられる．

以下では，これらの構成要素の詳細について述べる．

3.2 アクセス頻度情報

インデックスの各ノードに対してアクセス頻度情報を付与する．アクセス頻度情報の構成は様々なものが考えられる．今回はその一例として，アクセスを集計し始めた時刻である集計開始時刻と，集計開始時刻からそのノードに対してアクセスがなされた回数を示すアクセス回数からなるアクセス頻度情報を考える．

3.3 再配置バッファ

ディスク間でデータを再配置する際にどのデータを再配置するかを記憶するため，再配置バッファと呼ばれるバッファをディスクの台数と同じ数だけメモリ上に用意する．このバッファを利用することで，同じディスクに対する再配置をまとめてディスクのスピンアップや書き込み動作の回数を抑えることや，書き込み動作を行う時間帯をファイルサーバに余力の有る時間帯まで遅延させる等が可能となる．

このバッファは，再配置対象となるノードの ID を保持する．

3.4 ノードアドレス変換テーブル

インデックスのあるノードが別のノードを指し示す際には，実際にノードが存在するポインタを保持するのではなく，代わりに各ノードに割り振られたユニークなノード ID と呼ばれる識別子を利用する．このノード ID と実際にノードが配置されている場所へのポインタとの対応関係を保持するのがノードアドレス変換テーブルである．このテーブルはハッシュテーブルによって実装され，メモリ上に配置される．

インデックスのノードを辿る場合には，ノード ID からこのテーブルを引いて実際にノードが配置されている場所を得ることになる．このようにノード ID によって他のノードを参照することで，再配置の際にノードが移動することに伴うポインタの書き換えをする必要を無くすことができる．

3.5 未アクセスノード集合

未アクセスノード集合は，一定期間内でアクセスが無かったノードのノード ID を保持するために，メモリ上に確保された集合である．一定時間毎にこの集合に属する対して再配置を行うことで，アクセスが無かったデータについても再配置を行うことが可能になる．

4. 提案手法の流れ

ここでは，提案手法を適用してノード及びそれに関連するファイルデータの再配置を行う方法について述べる．

4.1 ファイル新規作成時

ファイルシステムにファイルが新しく作成された結果，インデックスにノードが追加される．この時，ノードにアクセス頻度情報を初期化して付与する．集計開始時刻はノードが追加された時点での時刻，アクセス回数は 0 に設定される．また，ノード集合に追加されたノードに対してユニークな ID を割り振り，その ID とノードへのポインタの対をノードアドレス変換テーブルに登録する．

4.2 ファイル削除時

ファイルシステムのファイルが削除されることでノードが削除された場合，ノードアドレス変換テーブルと未アクセスノード集合からそのノードに対応するエントリを削除する．また，各再配置バッファにそのノードが登録されている場合には，それについても削除を行う．

4.3 ファイルアクセス時

ファイルシステム上のファイルに対してインデックスを利用したアクセスが発生した場合，インデックスのルートノードからポインタを辿ることによってファイルの探索が行われる．その際，ノードが保持する子ノードのキーやポインタに関する情報を読み取るのと同時に，ノードに付与されているアクセス頻度情報についても読み取りを行う．ノードに記録されている集計開始時刻から一定時間以上が経過していた場合，ノードのアクセス回数を参照して再配置するべきかどうかを判断する処理を行う．

アクセス回数が現在配置されているディスクで設定されている基準値と比較して極端に差が有る場合には，その値に応じて新しいノードの配置先ディスクを決定し，そのディスクに対応する再配置バッファにノード ID を登録する．再配置が行われているかどうかにかかわらず，ノードの集計開始時刻をその時点の時刻で上書きし，アクセス回数を 0 にリセットする．

低頻度ディスクから高頻度ディスクにノード移動が行われる場合，低頻度ディスクの再配置バッファで保持されているノード 1 つを低頻度ディスクへ移動する処理を同時に行う．低頻度ディスクで再配置が実行されるのはこの場合だけとする．これは，低頻度ディスクから高頻度ディスクへ移動するファイル数と高頻度ディスクから低頻度ディスクへ移動するファイル数を等しく保つことによって，ディスク間でのファイルの極端な偏りを抑え，各ディスク資源を有効に利用するためである．

ノードの集計開始時刻から一定時間が経過していなかった場合には，アクセス回数を確認する．アクセス回数が 0 であったなら，そのノードを未アクセスノード集合から削除する．その後，アクセス回数をインクリメントしてから次のノードまたは要求されたデータを参照してアクセス処理を続行する．

4.4 再 配 置

再配置バッファに移動対象のノードが追加されても，その時点ではすぐに再配置を行わない．再配置は，再配置バッファが対象とするディスクに対してアクセスが発生した時点で，再配置バッファに一定数以上のノードが存在する場合に行われる．これは先程述べたように，別のデータに対するアクセスと同時にデータを再配置することによって，再配置で引き起こされるスピンアップや書き込みの回数を削減するためである．また，再配置バッファを利用して夜間などのアクセス頻度の低い時間帯になるまで再配置を遅延することで，昼間などのファイルサーバの利用頻度が高い時間帯における再配置での性能低下を防ぐことも可能となる．

しかしながら，再配置を実行する条件をこれに限った場合，特定のディスクに対するアクセス頻度が極端に低い時にそのディスクへの再配置が行われなくなることが考えられる．この

問題に対応するため，再配置バッファに追加されたノードが規定量に達した場合には，無条件でノードの再配置を行うこととする．

なお，低頻度ディスクに対する再配置の実行は低頻度ディスクから高頻度ディスクへの移動が有った場合に行われる．これは先述の通り，ファイル数を極端に偏らせないための措置である．

再配置が実行されると，再配置対象の各ノードが移動先のディスクにコピーされる．インデックスのリーフノードを再配置したときには，そのリーフノードが指すファイルのデータについても同様に再配置を行う．最後に，ノード及びファイルデータを元ディスクから削除し，再配置バッファの当該エントリを抹消する．

4.5 未アクセスファイルの処理

ノードに対するアクセスが有った場合とは別に，一定時間毎に未アクセスノードについて再配置処理を行う．これによって，アクセスが無かったノードについても適切な再配置が一定時間毎に行われることが保証される．未アクセスノードに存在するノードは一定時間でアクセスが無かったノードであるため，最もアクセス頻度が低いデータを配置するディスクの再配置バッファに登録する．

その後，未アクセスノード集合をノードアドレス変換テーブルのキー集合を用いて初期化する．

5. 評　　価

提案手法を適用した場合のストレージのアクセス速度と消費電力について，ソフトウェアシミュレータによる評価を行う．ここでは，評価の概要とその結果について述べる．

5.1 評 価 方 法

評価に用いるワークロードとして，統計モデルに基いて人工的に生成されたファイルに対するアクセスパターンを利用する．ワークロードの生成に用いた各種パラメタを表 1 に示す．これらのパラメタをもとにして生成されたファイルアクセスは，全てファイルシステム上のファイルに対して作られたインデックスを利用して行われるものとする．

表 1　ワークロードの各パラメタ

対象ファイル数	10000 ファイル
ファイルサイズ	各ファイル 1 MB
アクセス分布	均一ランダムまたは Zipf 分布（偏り 1）
到着時刻分布	ポアソン到着
ワークロード時間	604800 秒（7 日間）
アクセス回数	604800 回（平均毎秒 1 回）

ソフトウェアシミュレーションには，本研究室において開発されているソフトウェアシミュレータ [5] を用いる．このソフトウェアシミュレータは，複数ディスクに対するアクセスパターンからストレージの消費電力とアクセスを処理するのに要した時間を計算する．このシミュレータから得られたアクセス時間とストレージの消費電力を評価項目とする．シミュレータに与えるハードディスクの各種パラメタを表 2 に示す．

表 2　ハードディスクの各パラメタ

参考モデル	HGST Hitachi Deskstar 7K2000
容量	2 TB
プラッタ数	5
ディスク回転数	7200 RPM
ディスクキャッシュ	4 MB
データ転送速度	134 MB/s
アクティブ時消費電力	11.1 W
アイドル時消費電力	7.5 W
スタンバイ時消費電力	0.8 W
スピンダウン時消費エネルギー	35.0 J
スピンアップ時消費エネルギー	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

5.2 評 価 項 目

今回の評価では次の 3 項目について評価を行う．

アクセス待ち時間　ファイルリクエストが到着してから実際にリクエストの処理が開始するまでの所要時間．

ファイルアクセス時間　ファイルリクエストの処理が開始されてから，ファイルの内容を完全に取得し終わるまでの所要時間．

ストレージ消費電力　最初のファイルリクエストの到着から最後のファイルリクエストの処理を完了するまでに，ファイルを保持する全てのストレージが消費した電力の合計．

5.3 評 価 対 象

提案手法を適用し，インデックスのノードに付与したアクセス頻度情報を参照してディスク間でデータの再配置を行った場合（PROPOSAL）を評価の対象とする．低頻度ディスクから高頻度ディスクへの移動は，低頻度ディスクにあるデータに対してアクセスが有ればその時点で無条件に行われる．また，再配置の基準となる時間間隔は 86400 秒（24 時間）で固定とする．

また，比較対象として，複数ディスク間で均等にファイルを配置し再配置などを行わない場合（NORMAL）についても評価を行う．

PROPOSAL では，10 秒以上アクセスのなかったディスクはスピンダウンさせる．一方，NORMAL においてはスピンダウンは行わず，全てのディスクが回転している状態を常に維持する．

5.4 評 価 結 果

ストレージ消費電力についての評価結果を図 2 に示す．このグラフは，最初のファイルアクセスが発生してから最後のファイルアクセスが完了するまでのディスクの消費電力をディスク毎に表している．

ファイルアクセスが全てのファイルに対して均一な確率で発生する場合では，PROPOSAL が NORMAL に比べて 4.6% 程度低い消費電力を示した．更に，ファイルアクセスが偏り 1 の Zipf 分布に従う場合には PROPOSAL が NORMAL に比べて約 10.2% の消費電力削減を実現していることが分かった．この結果から，特にファイルアスセスに偏りが生じている場合には，ファイルに対するアクセス頻度に関する情報を利用して，

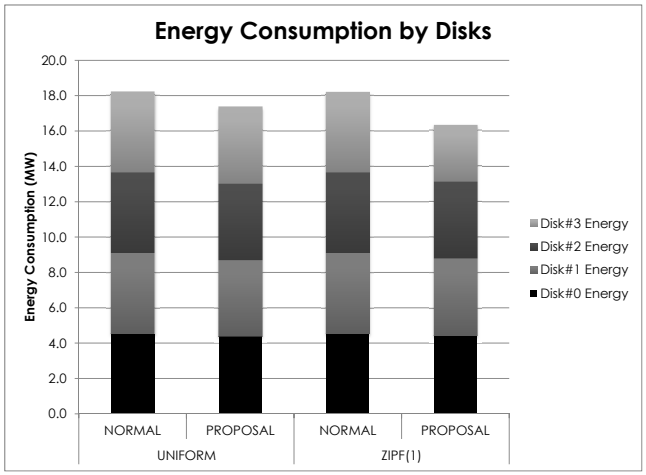

図 2　ストレージ消費電力の比較

ディスク間で適切にファイル配置を行うことでストレージの省電力化が実現できることが分かった．

次に，アクセス待ち時間について評価した結果を図 3 に示す．PROPOSAL が NORMAL と比較して 1 秒程度遅い結果となっている．これは，PROPOSAL がスピンダウンを行うことで省電力化を実現しているため，スピンダウンしているディスクにあるファイルにアクセスが発生した場合，スピンアップ時間だけアクセス処理の開始が待たされることが原因であると考えられる．

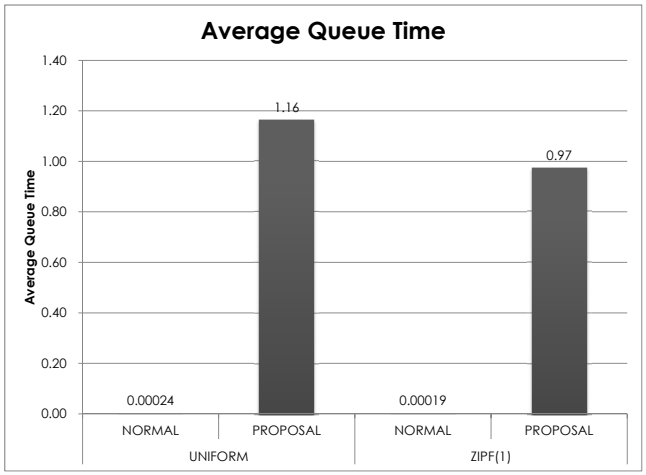

図 3　アクセス待ち時間の比較

表 3　提案手法におけるアクセス待ち時間の統計

アクセス偏り	0	1
平均値	1.163	0.9725
最小値	0.000	0.000
第 1 クォータ	0.000	0.000
中央値	0.000	0.000
第 3 クォータ	0.000	0.000
最大値	16.039	16.0461

提案手法におけるアクセス待ち時間についての詳細な統計を表 3 に示す．最大値がスピンアップ時間と同程度の 16 秒前後になっている一方で，第 3 クォータまでは待ち時間無しであることを示す 0 秒になっている．この結果から，ごく一部のファイルアクセスがスピンアップ待ちによって平均アクセス待ち時間を引き上げており，大部分の他のファイルアクセスは殆ど待たされることなく実行されていることが分かる．

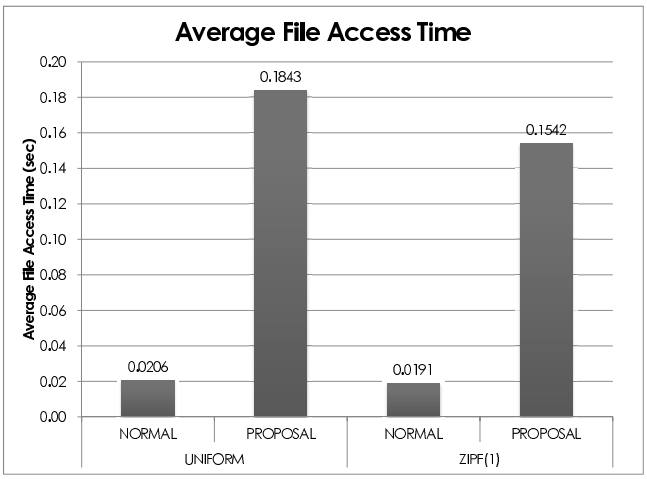

図 4　ファイルアクセス時間の比較

ファイルアクセス時間について評価した結果を図 4 に示す．アクセスに偏りが有る場合でも無い場合でも，PROPOSAL が NORMAL と比較して 0.15 秒程度遅い結果となった．これは，ファイルアクセスの発生によってファイルの再配置が行われる場合が有ることが主な原因であると考えられる．

表 4　提案手法におけるファイルアクセス時間の統計

アクセス偏り	0	1
平均値	0.18427	0.15416
最小値	0.00180	0.00180
第 1 クォータ	0.01925	0.01925
中央値	0.01925	0.01925
第 3 クォータ	0.01925	0.01925
最大値	16.0480	16.15770

提案手法におけるファイルアクセス時間についての詳細な統計を表 4 に示す．こちらの結果も，最大値がスピンアップ時間と同程度の 16 秒前後である一方で，第 3 クォータまでは 0.01925 秒以下の値を示している．よって，ファイルアクセス時間もアクセス待ち時間と同様に，一部のファイルアクセスがスピンアップ待ちによって平均アクセス時間を引き上げており，大部分のファイルアクセスについては比較的短い時間で処理できていることが分かる．

これらの結果から，提案手法は，アクセス速度に関しては一部のファイルアクセスにおいて通常の複数ディスク構成のファイルシステムより低下するものの大部分のファイルアクセスについては殆ど影響が無く，その一方で，アクセス頻度に偏りが有る場合においてストレージの消費電力を削減できていると言える．

6. 関連研究

様々な種類のデータを数多く保持している場合にデータの重要度などの属性やアクセス頻度によって複数ディスク間でデータの配置を行う手法として，Information Lifecycle Management (ILM) が存在する．ILM では，利用のされ方に応じてデータ毎に事前に定めたポリシーを適用し，データを格納するストレージを決定することでストレージ資源の有効活用を実現する．多くの ILM では，様々な種類のストレージを用いた大規模なストレージシステムを前提として，サーバによってポリシーやストレージを管理する構成となっている．

データのアクセス頻度をもとにデータの再配置を行うことでストレージの消費電力を削減する手法としては MAID [2] や RAPoSDA [1] が挙げられる．これらの手法では，データアクセスの局所性を利用して少数のディスクをキャッシュに用いることで大多数のディスクに対するデータアクセスを抑制し，省電力化を実現する．しかし，これらの手法においてはファイルシステムやデータベースにおけるインデックスと関連付けた研究はなされていない．

データベースのインデックスにおいて，データのアクセス頻度を利用する手法として LRU-K [3] がある．通常の LRU では直前のアクセス時刻によってキャッシュの置換を行うが，この手法ではデータに対するアクセス頻度 として k 回前のアクセスがあった時刻を利用することで，LRU が苦手とする全スキャンのようなアクセスパターンの場合や，アクセスパターンが一定でない場合においても適切なキャッシュ置換を実現する．しかしながら，この手法はファイルシステムにおけるファイルをその対象とはしていない．

インデックスを利用した SSD などのフラッシュメモリストレージでの省電力化手法としては BFTL [4] がある．この手法ではフラッシュメモリへのアクセス層である FTL と論理的な B-Tree インデックス層との間に BFTL という階層を新しく追加する．BFTL においてストレージに対する書き込みをまとめることで，フラッシュメモリにおける頻繁な書き込みを抑制し，アクセス速度を向上し消費電力を削減する．

7. まとめと今後の課題

本論文では，ファイルシステムにおけるインデックスに対してアクセス頻度に関する情報を付与することで，ファイルシステムを維持するストレージの消費電力を削減する新しい手法を提案した．また，この手法を適用した場合にアクセス速度とストレージの消費電力にどのような影響が与えられるか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い，アクセス頻度情報を用いてファイルを再配置することによって，一部のファイルアクセスの待ち時間が長くなる一方で，ストレージの消費電力を削減できることを示した．

今後の課題について述べる．現段階では，この提案手法は利用するストレージが十分信頼できるものであることを前提にしており，この手法自体は耐故障性を向上させる方法を提供していない．また，故障時におけるリカバリ処理についても，通常のファイルシステムにおける処理以外に必要となる処理については検討が必要である．ストレージの故障率を考慮に入れたより信頼性の高い手法の確立については今後の検討課題である．今回は，アクセス頻度情報として集計開始時刻とアクセス回数から構成されるものを考えたが，これ以外のアクセス頻度情報の構成を利用することも検討課題として挙げられる．また，多くのパラメタ値について評価を行い，適切なパラメタ値とその決定方法について調査することも必要である．

謝　　辞

本研究の一部は，文部科学省科学研究費補助金特定領域研究 (#21013017)，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] 引田論之, 横田治夫「プライマリ・バックアップ構成を有効利用したストレージシステムの省電力化手法の提案」, DEIM Forum 2010 E6-4, 2010.3.

[2] Dennis Colarelli, and Dirk Grunwald “Massive arrays of idle disks for storage archives”, Proc. of the 2002 ACM/IEEE conference on Supercomputing, pp. 1-11, 2002.

[3] Elizabeth J. O’Neil, Patrick E. O’Neil, and Gerhard Weikum “The LRU-K page replacement algorithm for database disk buffering”, ACM SIGMOD Record vol. 22 no. 2, pp. 297-306, 1993

[4] Hyun-Seob Lee, Sangwon Park, Ha-Joo Song, and Dong-Ho Lee “‘An efficient buffer management scheme for implementing a B-tree on NAND flash memory”, Embedded Software and Systems, pp. 181-192, 2007

[5] 引 田論之, 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, 2010.8.