デバイス性能モデルを用いた制御による階層型ハイブリッド・ストレージ・システムのアクセス性能向上

飯澤　健† 　相坂　勇気‡ 　横田　治夫‡ 　小沢　年弘†

†株式会社富士通研究所〒211-8588 神奈川県川崎市中原区上小田中 4-1-1
‡東京工業大学〒152-8550 東京都目黒区大岡山 2-12-1

E-mail: †{iizawa.ken, t.ozawa} @jp.fujitsu.com, ‡aisaka@de.cs.titech.ac.jp, yokota@cs.titech.ac.jp

あらまし SSD と HDD を組み合わせ，低コストで高速なストレージ・システムを実現するハイブリッド・ストレージ・システムが注目されている．階層型と呼ばれる方式では，データの移動単位毎に収集したアクセス数等の統計情報に基づき，定期的に SSD/HDD 間でデータを移動する．従来，性能を最大化するデータ配置の選択はヒューリスティクスにより行われていた．本研究では，アクセス・パターンから実行時の性能を予測するデバイス性能モデルを用いて，統計情報から性能を最大化するデータ配置を直接選択する手法を提案する．複数のワークロードに対してシミュレーションによる評価を行い，従来手法と比較した結果を示す．

キーワード ハイブリッド・ストレージ，階層制御，デバイス，モデリング

1. はじめに

ビッグデータのビジネスへの活用が進むにつれ，企業の情報システムに蓄積されるデータの量は増加の一途を辿っている．また，有用な情報を短時間で抽出するためには，蓄積された大量のデータに対して高速にアクセスできる必要がある．したがって，現代においては，大容量と高速なアクセス性能を単一のストレージ･システムが同時に提供することが求められている．従来，ストレージ・システムの主要な構成デバイスとして利用されてきた HDD は，低コストで大量のデータを蓄積できる反面，十分なアクセス性能を提供できないことが多い．

一方，近年は NAND 型フラッシュメモリを用いた SSD の普及が進んでいる．SSD は HDD に比べアクセス性能，特にランダム・アクセス性能に優れる．SSD の容量単価は下落傾向にあるものの，ビッグデータの全てを SSD に格納するのはまだコスト面で現実的でないことが多い．

これらの問題の解決策として広く使われているのが，SSD と HDD を組み合わせ，低コストで高速なストレージ・システムを実現するハイブリッド・ストレージ・システムである．そのうち，階層型と呼ばれる方式では，データの移動単位毎に収集したアクセス数等の統計情報に基づき，最適なデータ配置を選択し，定期的に SSD/HDD 間でデータを再配置することで，システム全体のアクセス性能向上を図る．最適なデータ配置の選択はヒューリスティクスにより行われることが多い（例．アクセス数の多いデータは SSD に配置する，等）．

しかし，２章で詳述するように，ヒューリスティクスに基づくデータ配置の選択には次の問題がある．

SSD は任意のアクセス・パターンに対して HDD よりも高い性能を発揮するわけではない．一般に，SSD に対する上書きアクセスはブロック単位の消去が必要となるため，単純な書き込みアクセスや読み出しアクセスに比べ,アクセス性能が低いことが知られている．そのため, HDD をログ構造化デバイスとして利用する場合には，一般的な SATA HDD であっても，ミッドレンジの SSD と遜色ない書き込み性能を発揮するという報告もある[12].

あるいは，あるアクセス・パターンに対して SSD の方が HDD より高い性能を発揮するとしても，性能差が許容範囲ならばデータを HDD に配置した方がコストの面で有利となる．

従来のヒューリスティクスに基づくデータ配置の選択は，上記のような SSD/HDD のデバイス特性を考慮していなかったために，両者の性能を十分に引き出すことができていなかった．

そこで本研究では，ワークロードのアクセス・パターンから，当該ワークロードを各デバイスで実行した場合の性能を予測することで，性能を最大化するデータ配置を選択する手法を提案する．複数のワークロードに対してシミュレーションによる評価を行い，従来手法と比較した結果を示す．

本論文の構成は以下の通りである．２章で従来の階層型ハイブリッド・ストレージ・システムが抱える問題点を明らかにする．３章で本研究における提案手法を示す．４章では複数のワークロードに対してシミュレーションによる評価を行った結果を述べる．５章で評価結果に対する考察を行う．６章で関連研究の紹介を行い，階層型ハイブリッド・ストレージ・システムの分野における本研究の位置づけを示す．最後に，７

章で今後の課題を述べ，まとめとする．

2. 従来技術

ハイブリッド・ストレージ・システムは，SSD の利用方法により，階層型とキャッシュ型に大別される．表 1 に両者の違いをまとめる．

表 1　階層型とキャッシュ型の違い

特性	階層型	キャッシュ型
データの冗長性	なし．データは SSD/HDD のいずれか一方に存在．	あり．SSD に存在するデータは HDD にも存在．
データの移動単位	大（例．64MiB）	小（例．4KiB）
データ再配置のタイミング	一定期間毎（例．30 分毎）	アクセスの都度
データ配置の選択方法	一定期間の統計情報	LRU 等

キャッシュ型は従来の RAM を用いたキャッシュと同じく，全てのデータを HDD に配置する方式である．データがアクセスされる都度 HDD から SSD へとデータをコピーする．データコピーの単位は一般に 4KiB 程度と小さい．SSD へコピーするデータは，LRU 等の置換アルゴリズムによって選択される．キャッシュ型のハイブリッド・ストレージ・システムとしては，OSS の flashcache や NetApp の FlashCache [10]が広く知られている．

一方，階層型は，データを SSD と HDD に分けて配置する．データのコピーは行わず，SSD と HDD 間でデータの移動を行う．データの移動は一定期間毎に行われる．以降，この一定期間を epoch と呼ぶ．データ移動の単位は一般に 64MiB 程度と大きい．以降，このデータ移動の単位を extent と呼ぶ．SSD へ移動するデータは，extent 毎に記録された前 epoch のアクセスの統計情報に基づき選択される．階層型のハイブリッド・ストレージ・システムは，IBM [13], EMC [16]等のベンダーにより商用化されている．

階層型はキャッシュ型に比べ，書き込みが多く，アクセスの空間局所性が低く，SSD 容量が小さい場合に高い性能を示すというメリットがある[4][5].

１章で述べた通り，階層型ハイブリッド・ストレージ･システムは以下に述べるような問題を抱えている．

従来技術として，前 epoch の各 extent のアクセス数に基づき，最適な extent の配置を選択する配置選択アルゴリズムを想定する．性能指標として，SSD/HDD に対して行われる全アクセスの平均レイテンシを想定する．

一般に HDD はランダム・アクセスよりシーケンシャル・アクセスが高速である．今，前 epoch で HDD に配置されていた２つの extent があり，前 epoch におけるアクセス数が同一であったとする．かつ，一方はランダム・アクセス，もう一方はシーケンシャル・アクセスが支配的だったとする．前 epoch で各 extent に対して行われた全アクセスの合計レイテンシを考えると，前者の方が後者よりもずっと大きかったはずである．したがって，アクセス・パターンが前 epoch と次 epoch で変化しないと仮定するなら，前者の extent を SSD に移動する方が性能改善効果は大きい．しかし，従来のアクセス数に基づく配置選択アルゴリズムでは，このような判断をすることができない．

同様に，SSD については，一般に書き込みアクセス（特に上書きアクセス）より読み出しアクセスが高速であることが知られている．今，前 epoch で HDD に配置されていた２つの extent があり，前 epoch におけるアクセス数が同一であったとする．かつ，一方は書き込みアクセス，もう一方は読み出しアクセスが支配的だったとする．仮に前 epoch において，各 extent に対する全アクセスの合計レイテンシが同一だったとしても，（アクセス・パターンが前 epoch と次 epoch で変化しないと仮定するなら）後者の extent を SSD に移動する方が性能改善効果は大きい．従来のアクセス数に基づく配置選択アルゴリズムでは，同様にこのような判断もすることができない．

上記は極端な例であり，実際にはヒューリスティクスを配置選択アルゴリズムに組み込むことで，ある程度デバイス特性を考慮した階層型ハイブリッド・ストレージ・システムも提案されている．例えば Guerra ら[1]は HDD へのシーケンシャル・アクセスを検出するヒューリスティクスを組み込んだ．

ただし，ヒューリスティクスによるデバイス特性の考慮にも次の問題がある．一口に HDD と言っても，プラッタの数やキャッシュメモリサイズ等の内部機構や，RAID 等の利用方法により，その性能特性は大きく変わる．SSD についても同様で，性能特性は素子の種類や FTL(=Flash Translation Layer)の作りに大きく依存する．したがって，ヒューリスティクスによりデバイスの性能を引き出すためには，デバイスを購入する都度，人手による性能特性の理解とチューニングが必要となる．データ分析の速度が企業の競争力を左右する昨今のビジネス環境においては，新たなデバイスをいち早く導入し，その恩恵を受けることが重要となる．人手による性能特性の理解とチューニングの工程は，新デバイスを導入する上でのボトルネックとなる．

3. 提案手法

3.1. 概要

本研究では，ワークロードのアクセス・パターンから，当該ワークロードを各デバイスで実行した場合の性能を予測することで，平均レイテンシを最小化するデータ配置を選択する手法を提案する．再配置後の性能を正確に予測することで，従来のヒューリスティクスによる配置選択アルゴリズムに比べ，より性能向上効果の大きい extent を選択可能になることが期待される．

本研究の狙いに最も近い先行研究として，Park ら[9]の研究がある．

Park らは，複数の仮想ディスクと複数のストレージ・デバイスが存在する仮想化環境における，負荷分散アルゴリズムを提案した．デバイス間で負荷の偏りを検知すると，仮想ディスクのマイグレーションを行い，偏りを解消するというものである．各デバイスの性能モデルに基づいた予測を行うことで，従来手法に比べ，より最適に近い仮想ディスクの配置を選択できることを示した．

ただし，Park らの性能モデルを階層型ハイブリッド・ストレージ・システムに適用するためには，以下の変更が必要となる．

Park らはデバイス全体のモデル化を行い，当該デバイスに配置する仮想ディスクの組み合わせのそれぞれに対して，性能の予測を行った．一つのデバイスに配置される仮想ディスクの数が高々数個なのに対し，extent は数百～数千個配置されることになるため，同一デバイスに配置されうる extent の組み合わせは膨大な数に上る．

そこで本研究では，デバイス全体ではなく extent のモデル化を行う．このモデルは入力として，ある epoch である extent に対して行われる全アクセスのアクセス・パターンを表現する特徴量を受け取り，それらのアクセスを実行した場合の合計レイテンシの予測値を出力する．このモデルは各 extent の性能予測を extent 毎に独立して行うことができるため，最適な extent 配置を選択するための探索空間を大幅に削減することができる．配置選択アルゴリズムの詳細は後述する．

表 2 に提案手法で使用する用語を定義する．

表 2　用語集

用語	意味
特徴量	ある epoch で，ある extent に対して行われた全アクセスのアクセス・パターンを表現する複数の統計情報の組．
性能モデル	ある epoch，ある extent の特徴量を入力として受け取り，そのワークロードを実行した場合の合計レイテンシの予測値を出力するモデル．

なお，あるデバイスに属する全 extent の性能特性が同一であるという仮定の元，作成する性能モデルの数は 1 デバイスにつき 1 つとした．

3.2. 特徴量

[2][3][9]の先行研究に倣い，本研究では特徴量として表 3 の統計情報を採用した．

表 3　特徴量

特徴量	意味
アクセス数	ある epoch でこの extent に対して行われた合計アクセス数
読み出し比率	この extent に対する合計アクセス数における読み出しアクセス数の割合．
平均 アクセスサイズ	一回のアクセスで転送されたデータ量の平均．
平均 アクセス間 距離	連続するアクセス間のアクセス開始アドレスの差分の平均．大きいほどアクセスのランダム性が強いことを意味する．
平均 未完了 I/O 数	この extent に対して発行されたアクセスのうち未完了のアクセス数の平均．extent 毎に存在する仮想的なコマンドキューの平均キュー長を意味する．

3.3. 性能モデル

本研究においては，評価の都合上，SSD のモデル化は行わず，HDD のモデル化のみを行う．

性能モデルは，ある epoch である extent に対して行われた全アクセスから算出された特徴量のうち「アクセス数」以外を入力して受け取り，平均レイテンシを予測値として出力する．この予測値と「アクセス数」の積を，次の epoch で当該 extent に対して行われるであろう全アクセスの合計レイテンシと見做す．

HDD の性能モデルについては，解析的な手法を用いたモデルから統計的な手法を用いたモデルまで，様々なものが先行研究により提案されている[11][14][2][3][9]．本研究では，平均レイテンシを予測するモデルとして回帰木を採用した．回帰木は実数値を取る関数の近似に用いられる．学習時には，特徴空間を再帰的に分割し，分割された各矩形領域に出力値（＝提案手法の場合は平均レイテンシ）を割り当てる．予測時には，与えられた特徴量が属する矩形領域を選択し，当該領域に割り当てられた出力値を予測値として出力する．

回帰木は学習・予測に要する計算量が少なく，予測精度が高いため，Wang ら[14]を始め複数の先行研究において HDD の性能モデルとして採用されている．

3.4. 配置選択アルゴリズム

次のアルゴリズムにより，extent の配置を選択する．

① epoch 切り替え時，全 extent について，前 epoch の特徴量を SSD/HDD 双方の性能モデルに入力し，当該 extent を双方のデバイスに配置した場合の合計レイテンシを予測する．ここでは，前 epoch の特徴量が次 epoch でも持続することを前提としている．なお，extent が現在配置されている方のデバイスについては，予測値として前 epoch で実測された合計レイテンシをそのまま使用する最適化が可能である．

② 合計レイテンシの予測値から当該 extent の性能向上度（＝HDD に配置した場合の合計レイテンシ―SSD に配置した場合の合計レイテンシ）を求める．

③ SSD が一杯になるまで，性能向上度が大きい extent から順に SSD に配置する．残りの extent は HDD に配置する．

以下に具体例を示す．

SSD/HDD 合計で 6 個の extent が存在する場合を考える．前 epoch では extent=1, 2 が SSD に，extent=3, 4, 5, 6 が HDD に配置されていたとする．各 extent において次 epoch で予測される合計レイテンシが表 4 のようになったとする．

表 4　合計レイテンシの予測値の例

	SSD に配置した場合の合計レイテンシ	HDD に配置した場合の合計レイテンシ
1	500[ms]	3000[ms]
2	3500[ms]	4000[ms]
3	500[ms]	4000[ms]
4	2000[ms]	3500[ms]
5	500[ms]	1000[ms]
6	2000[ms]	1500[ms]

次に，extent 毎に性能向上度を求めると，図 1 のようになる．

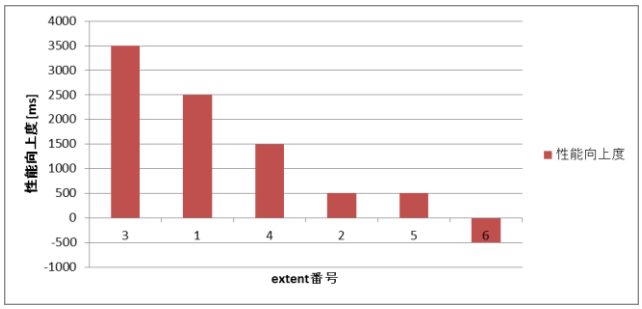

図 1　extent 毎の性能向上度の例

extent3 と extent1 の性能向上度が最大なので，提案手法ではこの 2 つの extent を SSD に配置する．実際には，既に extent1 は SSD に配置されているので，extent3 を SSD に，extent2 を HDD に移動する．

4. 評価

4.1. 評価方法

複数のトレースを対象としてシミュレーションによる評価を行う．提案手法により extent 配置を選択した場合の平均レイテンシと，従来手法により extent 配置を選択した場合の平均レイテンシを比較し，提案手法による性能向上効果を検証する．

4.2. ワークロード

MSR Cambridge トレース

MSR Cambridge トレースは，Microsoft Research Cambridge の Narayanan らによって採取され，一般公開されているファイル群である[7]．Microsoft Research Cambridge のデータセンタに設置されている 13 台のサーバから，各サーバに接続されているボリュームへのアクセスが，ブロック層で記録されている．トレースの採取期間は全ボリューム共に 2007/2/22 から 2007/3/1 までの約一週間である．

トレースには表 5 の情報が含まれる．

表 5　利用したトレースの形式

フィールド	意味
Timestamp	タイムスタンプ
Hostname	サーバ名
DiskNumber	ボリューム番号
Type	Read/Writeの種別
Offset	アクセスされたアドレス
Size	アクセスされたサイズ
ResponseTime	レスポンス時間

表 6 に 13 台のサーバの用途を示す．

表 6　ワークロードの種類

サーバ名	用途
usr	ユーザのホームディレクトリ
proj	プロジェクトのディレクトリ
prn	プリント・サーバ
hm	ハードウェア監視
rsrch	研究プロジェクト
prxy	ファイア・ウォール／Webプロキシ
src1	ソースコード管理
src2	ソースコード管理
stg	Webステージング
ts	端末サーバ
web	Web／SQLサーバ
mds	メディア・サーバ
wdev	テスト用Webサーバ

各サーバは 1 台のシステム・ボリュームと，複数台のデータ・ボリュームを持つ．システム・ボリュームは内蔵 HDD を RAID 1 でグループ化したボリュームで，データ・ボリュームは共用の外付けストレージ・アレイから RAID 5 でグループ化されたボリュームを切り出したものである．

本研究では，各サーバのシステム・ボリュームを対象として分析を行った．対象は hm_0, mds_0, prn_0, proj_0, prxy_0, rsrch_0, src1_2, src2_0, stg_0, ts_0, usr_0, wdev_0, web_0 の全 13 ファイルである．

VDI トレース

VDI トレースは筆者らが実運用環境で採取したトレースである．ある IT 企業で約 300 人のユーザによって使用されている仮想デスクトップ・インフラストラクチャ(VDI)において，2015/9/1 から 2015/9/6 までの 6 日間にわたり採取された．

この環境には 6 台のサーバと 1 台のストレージが存在する．ストレージ上には 6 個のボリュームが構成され，それぞれが各サーバに接続されている．ユーザはボリューム上に配置された仮想ディスクへアクセスを行う．

本研究では，6 個のボリュームのうちの 1 個を対象として分析を行った．

4.3. 分析方法

分析に使用した性能モデルと特徴量は 3 章で述べた通りである．

同じく MSR Cambridge トレースを評価に使用した Guerra ら[1]に倣い，extent のサイズを 64MiB，epoch の長さを 30 分とした．Guerra らによれば，extent のサイズは，メタデータ（＝各 extent の統計情報）のオーバヘッドを考慮しつつ，なるべく小さいサイズが選択された．同様に，epoch の長さは，extent 移動に必要な帯域が，利用可能な帯域の 10%程度になるように選択された．

MSR Cambridge トレースについては，前半の 2007/2/22 から 2007/2/25 を訓練用データ，後半の 2007/2/26 から 2007/3/1 をテスト用データとして使用した．

VDI トレースについては，2015/9/2 から 2015/9/3 を訓練用データ，2015/9/4 から 2015/9/5 をテスト用データとして使用した．

SSD の容量は，訓練用データのワーキング・セット・サイズに基づき設定した．ここで，ワーキング・セット・サイズとは，各 epoch でアクセスされた extent 数を意味している．SSD の容量は，訓練用データの平均ワーキング・セット・サイズの 10%に設定した．

なお，シミュレーションによる評価を行う都合上，以下の仮定を置いた．トレースが採取されたデバイスを低速デバイス（＝HDD）と見做し，extent の移動先として架空の高速デバイス（＝SDD）を仮定する．このデバイスは無限のアクセス性能を持つ，つまり，アクセス・パターンによらず任意のアクセスのレイテンシが 0 [ms]であるものとする．

ここで，提案手法の効果は次の二種類の要因の影響を受けることに注意する必要がある．

①予測モデルの精度

特徴量からどれくらい正確に性能を予測することができるか．精度が高いほど，合計レイテンシの大きい extent を正確に予測することができるので，再配置による平均レイテンシの向上が期待できる．

②特徴量の時間変化

前 epoch の特徴量に対応する性能を完全に予測できたとしても，次 epoch で特徴量が変化したとすれば，性能の予測値と次 epoch における性能の実測値には誤差が生じる．特徴量の時間変化が小さいほど，再配置後の次 epoch の性能を正確に予測できるので，より最適な extent 配置の選択が可能となる．

提案手法の効果に対するこれらの要因の影響を区別するため，評価は次の二段階に分けて行った．

連続する epoch で特徴量が変化しない場合

以下の手順で評価を行う．

①ある epoch における特徴量から各 extent の合計レイシを予測する．

②予測値の大きい extent を SSD（＝架空の高速デバイス）に移動したものと見做す．

③②で HDD に残した extent について，①と同一の epoch で実測されたレイテンシとアクセス数から平均レイテンシを算出する．

これは，前 epoch と全く同じアクセス・パターンが

次 epoch でも繰り返されると仮定した場合に，再配置により平均レイテンシがどれくらい向上するか評価していることに相当する．特徴量の時間変化の影響を排除することが本評価の目的である．

連続する epoch で特徴量が変化する場合

以下の手順で評価を行う．

①ある epoch における特徴量から各 extent の合計レイシを予測する．

②予測値の大きい extent を SSD（＝架空の高速デバイス）に移動したものと見做す．

③②で HDD に残した extent について，①の次の epoch で実測されたレイテンシとアクセス数から平均レイテンシを算出する．

これは，連続する epoch でアクセス・パターンが変化する現実的なシナリオについて評価していることに相当する．

4.4. 結果

連続する epoch で特徴量が変化しない場合

提案手法と従来手法を適用した場合の平均レイテンシ[ms]，および従来手法に対する提案手法の平均レイテンシの削減率[%]を表 7 に示す．

表 7　平均レイテンシの比較
（特徴量が変化しない場合）

ワークロード	提案手法[ms]	従来手法[ms]	削減率[%]
hm_0	1.12	1.17	3.98
mds_0	2.42	2.51	3.39
prn_0	0.325	0.374	13.1
proj_0	15.9	16.1	0.985
prxy_0	0.0761	0.0561	−35.6
rsrch_0	0.0968	0.108	10
src1_2	1.25	1.35	7.2
src2_0	0.375	0.388	3.23
stg_0	0.172	0.185	6.92
ts_0	2.85	2.87	0.871
usr_0	1.76	1.87	5.93
wdev_0	0.147	0.156	6.19
web_0	1.45	1.47	1.65
VDI	1.11	1.12	0.849

14 ワークロード中，13 ワークロードで平均レイテンシの向上が見られた．削減率の平均は 2.0%であり，従来技術に比べて改善した．

連続する epoch で特徴量が変化する場合

提案手法と従来手法を適用した場合の平均レイテンシ[ms]，および従来手法に対する提案手法の平均レイテンシの削減率[%]を表 8 に示す．

表 8　平均レイテンシの比較
（特徴量が変化する場合）

ワークロード	提案手法[ms]	従来手法[ms]	削減率[%]
hm_0	1.35	1.39	3.04
mds_0	2.48	2.56	3.1
prn_0	0.617	0.629	1.87
proj_0	19	19.2	1.23
prxy_0	0.0817	0.0596	−37
rsrch_0	0.193	0.188	−2.64
src1_2	1.74	1.73	−0.436
src2_0	0.417	0.41	−1.79
stg_0	0.199	0.198	−0.203
ts_0	3	3.02	0.535
usr_0	2.07	2.05	−0.884
wdev_0	0.257	0.252	−1.88
web_0	1.81	1.8	−0.324
VDI	3.03	2.96	−2.27

14 ワークロード中，5 ワークロードで平均レイテンシの向上が見られた．削減率の平均は-2.7%であり，従来技術に比べて悪化した．

5. 考察

評価の結果，提案手法による性能改善効果は軽微なものに留まった．本章では，その原因を 3 つの観点から考察する．

5.1. ワークロードの特性

アクセス数のみによって，extent の合計レイテンシを十分な精度で予測できるようなワークロードについては，提案手法による性能改善の余地が少ない．

この仮説を検証するため，連続する epoch で特徴量が変化しないと仮定した場合について，次の追加評価を行った．

従来技術を適用した場合の平均レイテンシと，理論上最適な再配置アルゴリズムを適用した場合の平均レイテンシを比較した．ここで，理論上最適な再配置アルゴリズムとは，合計レイテンシの実測値が多い上位 extent を SSD に移動するアルゴリズムである．これはまた，100%の精度で性能予測が可能な場合の提案手法と同一のアルゴリズムでもある．結果を表 9 に示す．

表 9　従来技術と理論上最適なアルゴリズムの比較

ワークロード	最適法[ms]	従来手法[ms]	削減率[%]
hm_0	0.975	1.17	16.6
mds_0	2.27	2.51	9.52
prn_0	0.285	0.374	23.8
proj_0	15.5	16.1	3.81
prxy_0	0.0418	0.0561	25.6
rsrch_0	0.0838	0.108	22.1
src1_2	1.04	1.35	23.1
src2_0	0.335	0.388	13.5
stg_0	0.164	0.185	11.4
ts_0	2.65	2.87	7.55
usr_0	1.68	1.87	10.5
wdev_0	0.134	0.156	14.1
web_0	1.37	1.47	7.3
VDI	0.91	1.12	18.1

削減率の平均は 14.7%だった．提案手法による削減率の平均は 2.0%だったので性能改善の余地はあるものの，これらのワークロードでは劇的な性能改善は見込めないことになる．

この理由として，次の二点が考えられる．

① 評価に使用したトレースが，合計アクセス数のみで合計レイテンシが決まるデバイスで採取された．

② 評価に使用したトレースでは，合計アクセス以外の特徴量が extent 間でほとんど差がなかった．

今後は評価結果の分析を進め，上記を明らかにする予定である．

5.2. モデル化の粒度

モデルをデバイス全体ではなく extent に対して作成することで，予測精度が下がることが予想される．これは以下の理由による．

① extent を跨ぐアクセスが「平均アクセス間距離」に反映されなくなる．

② 本来はデバイス全体で一つ存在するコマンドキューが，extent 毎に存在するものと見做される．

モデル化の粒度による影響を調べるため，デバイス全体をモデル化した場合と extent をモデル化した場合の予測精度を比較する．デバイス全体をモデル化した場合は，各 epoch で予測したデバイス全体の合計レイテンシの絶対誤差を算出した．extent をモデル化した場合は，各 epoch で予測した各 extent の合計レイテンシの絶対誤差を算出した．

比較結果を表 10 に示す．数値は全絶対誤差[%]の中央値である．

表 10　モデル化の粒度と予測誤差

ワークロード	デバイス全体[%]	extent[%]
hm_0	36.9	68
mds_0	3.36	38.8
prn_0	28.4	67.4
proj_0	28.2	68.2
prxy_0	5.37	126
rsrch_0	7.98	36.2
src1_2	74.1	167
src2_0	12.6	34.6
stg_0	30.5	31.6
ts_0	9.41	31.8
usr_0	10.6	74.5
wdev_0	13.6	52.6
web_0	5.04	38.4
VDI	13	103

デバイス全体をモデル化した場合は平均 20%, extent をモデル化した場合は平均 67%と，大幅な予測誤差の増大が見られた．

解決策として，デバイス全体のアクセス・パターンを表現する特徴量の導入等が考えられる．

5.3. 特徴量の時間変化

連続する epoch で特徴量が変化する場合，連続する epoch で特徴量が変化しないと仮定した場合に比べ，提案手法による性能改善効果の悪化が見られた．これは先述の通り，連続する epoch で特徴量が変化したことが原因と考えられる．

解決策として，提案手法と特徴量の変化を予測するアルゴリズムを組み合わせることが考えられる．前 epoch の特徴量をそのまま性能モデルへの入力として使用するのではなく，過去の epoch の特徴量から次 epoch の特徴量を予測し，予測値を性能モデルへの入力として使用することで，より正確な性能予測が可能になると思われる．

6. 関連研究

6.1. 階層型ハイブリッド・ストレージ・システム

階層型ハイブリット・ストレージ・システムは，IBM [13]や EMC [6]等，複数のベンダーから商品化されている．公開されているドキュメントから判断する限り，それらの多くはアクセス数に基づく配置選択アルゴリズムを採用している．

Guerra ら[1]はアクセス数に加え，アクセスのシーケンシャル性も考慮した配置選択アルゴリズムを提案した．

アクセス数の予測精度向上を目的とした研究としては，Oe ら[8]による研究がある．Oe らはアクセスが集中する領域を予測し，当該領域のデータを事前に

SSD に移動するアルゴリズムを提案している.

6.2. 性能予測に基づくデータ再配置

ストレージ・システムの性能向上を目的として，性能予測に基づいたデータ再配置を行う手法が提案されている.

Gulati ら[2][3]や Park ら[9]は複数の仮想ディスクと複数のストレージ・デバイスが存在する仮想化環境において，性能予測に基づいた負荷分散を行う手法を提案した.

7. おわりに

本研究では，ワークロードのアクセス・パターンから，当該ワークロードを各デバイスで実行した場合の性能を予測することで，性能を最大化するデータ配置を選択する手法を提案した．複数のワークロードに対してシミュレーションによる評価を行い，従来手法と比較した結果を示した.

今後は，特徴量の追加や，特徴量の変化を予測するアルゴリズムとの組み合わせ等により，性能モデルの精度向上を進める.

参 考 文 献

[1] J. Guerra, H. Pucha, J. Glider, W. Belluomini, R. Rangaswami. Cost Effective Storage using Extent Based Dynamic Tiering. In Proc. of FAST, 2011.

[2] A. Gulati, C. Kumar, I. Ahmad. BASIL : Automated IO Load Balancing Across Storage Devices. In Proc. of FAST, 2010.

[3] A. Gulati, G. Shanmuganathan, I. Ahmad, C. Waldspurger, M. Uysal. Pesto: Online Storage Performance Management in Virtualized Datacenters. In Proc. of SoCC, 2011.

[4] S. Hayashi, N. Komoda. Evaluation of SSD Tier Method and SSD Cache Method in Tiered Storage System. In Proc. of ACIS, 2013.

[5] H. Kim, I. Koltsidas, N. Ioannou, S. Seshadri, P. Muench, C. L. Dickey, L. Chiu. How could a flash cache degrade database performance rather than improve it? Lessons to be learnt from multi-tiered storage. In Proc. of INFLOW, 2014.

[6] B. Laliberte. Automate and Optimize a Tiered Storage Environment FAST! ESG White Paper, 2009.

[7] D. Narayanan, A. Donnelly, A. Rowstron. Write off-loading: Practical Power Management for Enterprise Storage. In Proc. of TOS, 2008.

[8] K. Oe, T. Nanri, K. Okamura. On-The-Fly Automated Storage Tiering with Proactive and Observational Migration. In Proc. of CANDAR, 2015.

[9] N. Park, I. Ahmad, D. J. Lilja. Romano: Autonomous Storage Management using Performance Prediction in Multi-Tenant Datacenters. In Proc. of SoCC, 2012.

[10] M. Peters. Netapp’s solid state hierarchy. ESG White Paper, 2009.

[11] C. Ruemmler, J. Wilkes. An introduction to disk drive modeling. IEEE Computer, 27(3), 1994.

[12] G. Soundararajan, V. Prabhakaran, M. Balakrishnan, and T. Wobber. Extending SSD lifetimes with disk-based write caches. In Proc. of FAST, 2010.

[13] Taneja Group Technology Analysts. The State of the Core Engineering the Enterprise Storage Infrastructure with the IBM DS8000. White Paper, 2010.

[14] M. Wang, K. Au, A. Ailamaki, A. Brockwell, C. Faloutsos, G. R. Ganger. Storage Device Performance Prediction with CART Models. In Proc. of MASCOTS, 2004.