社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DBS－145 (22)
2008－ＦＩ－ 91 (22)
2008／6／20

メニーコア時代における OS レベルでの I/O 最適化

上田 高徳[†] 平手 勇宇[††] 山名 早人[†††,‡]
† 早稲田大学基幹理工学研究科 †† 早稲田大学メディアネットワークセンター
††† 早稲田大学理工学術院 ‡ 国立情報学研究所
{ueda,hirate,yamana}@yama.info.waseda.ac.jp

OS Level I/O Optimization in the Many-Core Era

Takanori UEDA[†], Yu HIRATE[††], Hayato YAMANA[†††,‡]
† Graduate School of Science and Engineering, Waseda University †† Media Network Center, Waseda University
††† Science and Engineering, Waseda University ‡ National Institute of Informatics

1. はじめに

大容量のデータを保存・解析するために，ストレージは必要不可欠の存在である．ストレージの性能はシステム全体の性能に大きく関係するが，アクセス速度で不利なハードディスクドライブ（HDD）が，いまだにストレージとして利用されている．HDD はその物理的な構造から，アクセス性能の向上が難しく，CPU の性能向上に遅れをとっている．1996 年から 2006 年までの 10 年間で CPU の性能が 30 倍になったのに対し，HDD の性能向上は 1.3 倍に留まっているとされている[1].

近年，1 チップ上に複数のコアを持つマルチコア CPU が登場してきている．今後，1 つのチップに十数コアを搭載したメニーコア CPU が登場することは確実である．このようなメニーコア環境下では，アプリケーションが並列に動作するため，ストレージへのアクセス集中が問題になる．並列アクセスが原因で，HDD が特に不得手とするランダムアクセスの頻度が増え，ストレージがますますボトルネックになると考えられる．

そこで我々は，ストレージのボトルネックをソフトウェア的に軽減することを考えている．OS レベルに実装することで，ハードウェアおよびアプリケーションの修正を回避し，ボトルネックを低コストで軽減することを目指す．

2. OS レベルでの I/O 最適化

ストレージのボトルネック軽減，すなわち I/O の最適化を，アーキテクチャのどのレベルで実現するか，以下の 3 段階で捉えることができる．

1. アプリケーションレベル
2. OS レベル
3. ハードウェアレベル

アプリケーションレベルでは，アプリケーションの I/O 動作に適した最適化を実現できる．ただし，I/O 最適化を個々のアプリケーションに実装する必要がある．ハードウェアレベルでは，高性能なストレージデバイスを用いて性能を向上させることができる．例えば，近年普及が進んでいる Solid State Drive (SSD) を用いたり，大容量キャッシュメモリを搭載したハイブリッドディスクを用いたりすることが考えられる．しかし，新規ハードウェア導入などのコストが必要となる．

これに対し，OS レベルで実現すれば，アプリケーションおよびハードウェアの変更は必要ない．また，多くのアプリケーションが並列動作するメニーコア環境を考えると，OS レベルでのアプリケーション横断な I/O 最適化が必須と考えられる．そこで我々は，OS のストレージ管理機構を改良することで，ストレージのボトルネックを軽減することを考えている．例えば OS レベルでは，アクセスログからアプリケーションの I/O の性質を抽出することができる．

これまでに我々は，ファイルアクセスログから抽出したアクセスパターンを用いることで，キャッシュ性能を向上できる可能性を示している[2]．[2]の実験では，TPC-C 相当のベンチマークのアクセスログをシステムコールレベルで取得し，データマイニングのアルゴリズムを用いてアクセスパターンを抽出した．シーケンシャルアクセスをフィルタすることで，マイニングに掛る時間を小さくすると同時に，良質なアクセスパターンを抽出可能なことを確認した．[2]では，アクセスパターンを用いて LRU リストを並び替えることで，キャッシュミス率を低減できることも確認している．

現在の OS のディスクキャッシュ機構は LRU を用いたものが基本であり，アクセスパターンが考慮されていない．そこで我々は，アクセスパターンを考慮したディスクキャッシュ機構を Linux に実装し，実機で評価することを最初の目標にしている．その際はストリームマイニング的に処理を行い，オーバーヘッドを抑えることが必須であると考えている．

3. おわりに

ここ数年，SSD の普及など，ストレージアーキテクチャの変革を予感できる．SSD の性質に，読み込みと書き込み速度が異なる非対称性があり，この性質は I/O 最適化の上で無視できない．今後は SSD といったフラッシュメモリの性質も考慮にいれて，I/O 最適化を考える必要がある．本研究を行うにあたっては実用化を念頭に置き，最終的にはカーネルパッチとして広く配布することを目指す．ワークショップでは，これまでの研究概要と既存研究について概要を述べると共に，最新の研究成果について述べ，今後の研究指針を示す．

参考文献

[1] “Overcoming Disk Drive Access Bottlenecks with Intel® Turbo Memory,” Intel, May. 2007.
[2] 上田高徳，平手勇宇，山名早人，“システムコールレベルのアクセスログに対するディスクアクセスパターンマイニング，” DEWS2008，Mar. 2008，D5-3.