B-017

SSD キャッシュの制御によるシーケンシャルアプリケーションの性能向上
Improving Sequential I/O Performance with SSD Cache Replacement Control

中島 健司† 近 丈一郎† 山口 実靖†
Kenji Nakashima Joichiro Kon Saneyasu Yamaguchi

1. はじめに

DNA データなどのビッグデータを利活用するには，プライバシーの保護のための暗号化が不可欠である．データの暗号化は，データ量の増大に繋がることがある．そのため，ビッグデータ処理基盤には高い I/O 速度が求められる．大規模データの I/O 性能を向上する方法の 1 つに，SSD をキャッシュとして用いる手法がある．本稿では SSD キャッシュに着目し，SSD データキャッシュの置換の改善による秘匿検索アプリケーションの I/O 性能改善手法を提案する．そして，性能評価によりその有効性を示す．

2. Flashcache

Flashcache は Facebook 社が開発したオープンソースのキャッシュモジュールである．

Flashcache は Linux Device Mapper より実装され，SSD のようなアクセス速度が高い記憶装置をキャッシュとして用いることで I/O 性能の高速化を図る．キャッシュ置換アルゴリズムには FIFO(default)と LRU が実装されており，キャッシュモードとしては Writethrough，Writearound，Writeback を選択できる．キャッシュブロックは初期設定では 4KB で管理される．

3. ゲノム配列の秘匿検索アプリケーション

バイオインフォマティクス分野の研究では，ゲノム配列の検索が頻繁に行われる．しかし，個人ゲノムの利用はプライバシー保護への対策が必須であり，暗号化したままでゲノム配列の検索処理を行う必要がある．これを実現するための手法として，PBWT 離散データ構造と完全準同型暗号を組み合わせた手法が提案されている．本稿では，PBWT 離散データ構造と完全準同型暗号を用いたゲノム秘匿検索手法 [1] に焦点を当て，そのアプリケーション性能について考察を行う．

当該手法を実装したアプリケーションは，クライアント(検索者)がサーバ(データ保有者)に対してクエリ内容を秘匿し，サーバはクライアントに対してクエリの検索結果以外の情報を秘匿する．当アプリケーションの現在の実装では，サーバは長さ N のゲノム配列が M 本登録されているデータベースを持つ．

4. 提案手法

本章では，Flashcache の動作を考慮し，前章のゲノム配列の秘匿検索アプリケーションの I/O 性能を向上させる手法を提案する．

† 工学院大学大学院 工学研究科 電気・電子工学専攻
Electrical Engineering and Electronics, Kogakuin University Graduate School

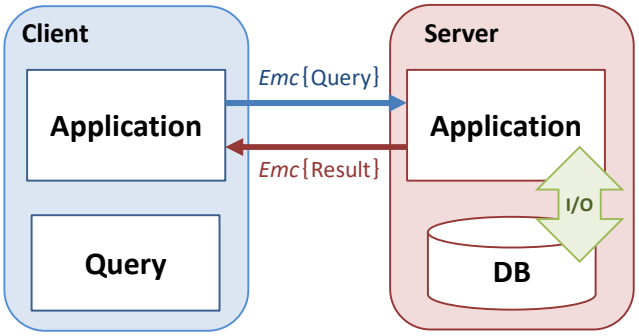

図 1 ゲノム配列検索の概略図

ゲノム検索アプリケーションはデータにアクセスする順序が既知であり，この特性を利用することで性能向上を図る．提案手法では，アプリケーションが CPU インテンシブ処理を実行中に次にアクセスされるデータを Flashcache キャッシュ領域である SSD に格納する．

図 2 に通常手法の動作，図 3 に提案手法の動作を示す．通常手法では，アプリケーションが HDD から対象のデータを読み出し，読み出し完了後に CPU 処理を開始する．一方提案手法では，事前に対象データを HDD から SSD に格納しておき，SSD からアプリケーションに読み込ませる．図 3 の Helper thread は，アプリケーションによる data No. n+1 の読み出し終了直後に，data No. n+2 の読み出しを開始する．図 3 に示すように，data No. n+1 の処理と data No. n+2 の読み込みが並列に実行される．Flashcache 環境にてデータにアクセスすると，そのデータがキャッシュ領域(SSD)に格納される．アプリケーションによる data No. n+2 の読み込み開始時には当該データは既に SSD に格納されており，I/O 速度の向上を図ることができる．

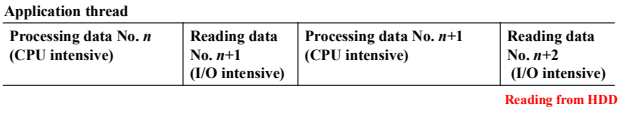

図 2 通常手法

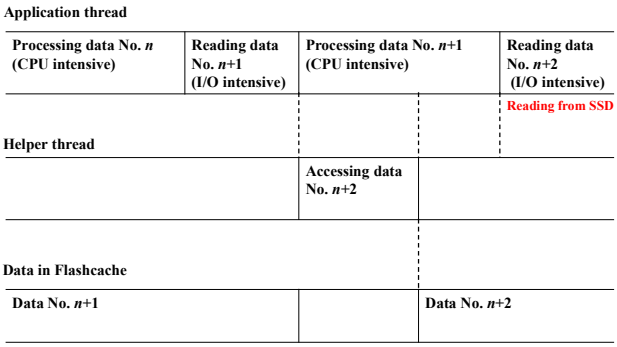

図 3 提案手法

5. 性能評価

5.1 測定方法

本節にて，提案手法の性能評価を行う．HDD，SSD，Flashcache(非提案手法)，Flashcache(提案手法)においてゲノム検索アプリケーションを実行し，データの読み込み時間を比較した．HDD や SSD における実行は，データを HDD や SSD に格納してから読み込みを行う．測定は，HDD，SSD，Flashcache(非提案手法)に関しては図 4 の手順で実行し，Flashcache(提案手法)に関しては図 5 の手順で実行した．図 4 と図 5 に示すように 2 個のファイルの読み込み時間を測定した．図 5 に示すように提案手法は No. 2 のデータを読み込む時間に影響を与え，No. 1 のデータの読み込み時間には影響を与えない．ファイル 1 個当たりのデータサイズは 2GB である．

測定に用いた物理計算機のメモリサイズは 16GB であり，メモリからデータを読み込まないように Flashcache(提案手法)の data No.2 を読み込む前にドロップキャッシュを実行している．

測定に用いた Flashcache の構成は，HDD が主記憶装置と，SSD がキャッシュとなっている．キャッシュサイズは 8GB，キャッシュ置換アルゴリズムは FIFO，キャッシュモードは Writeback，キャッシュのブロックサイズは初期設定の 4KB とした．

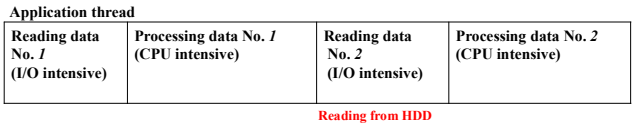

図 4　通常手法の測定概要

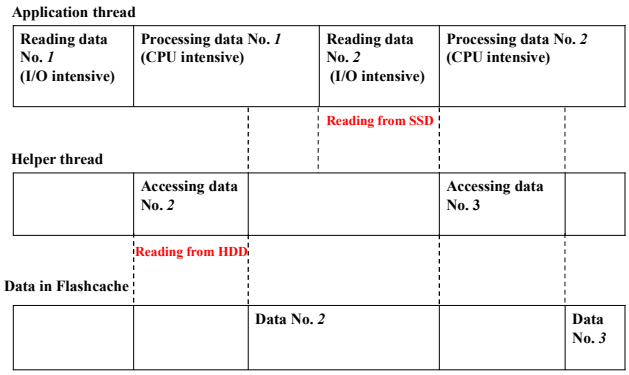

図 5　提案手法の測定概要

5.2 測定結果

測定結果を図 6 に示す．縦軸は各データの読み込みにかかった時間である．

図 6 において，読み込み時間は HDD では平均 16.2 秒，SSD では平均 10.6 秒，Flashcache(非提案手法)は平均 47.8 秒となっている． Flashcache(提案手法)の読み込み時間は data No. 1 で 48.6 秒，data No. 2 で 13.6 秒となっている．

Flashcache(非提案手法)の性能は，他の手法に比べて著しく低くなっている．これは同じデータに再度アクセスをしないため，すべてのアクセスにおいてキャッシュミスが発生していることが原因であると考えられる．具体的には，

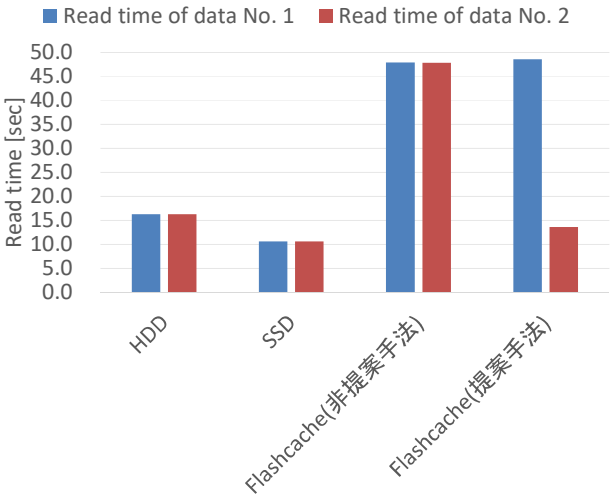

図 6　測定結果

キャッシュミスに伴うデータのキャッシュ(SSD)への書き込み処理による増加であると考えられる．

Flashcache(提案手法)では，Flashcache による data No. 2 の読み込み時間が大幅に改善していることが分かる．data No. 1 の読み込み時間に比べて data No. 2 の読み込み速度は約 3.6 倍に向上していることが確認できる．

HDD の読み込み時間と提案手法の data No. 2 の読み込み時間を比較すると，約 19.5%の性能向上している．これは，単純に HDD を使用した場合よりも Flashcache(提案手法)は I/O 性能が高いことを示している．

本研究では大量のデータの処理を想定しており，すべてのデータをメモリや SSD に格納することはできない状況を想定している．よって SSD の性能は提案手法の理想的な性能であり，提案手法の性能の上限となる．Flashcache(提案手法)の性能は，単純に SSD や HDD を用いて測定した性能の中間程度の性能になっている．Linux カーネルを改変し，本実験におけるカーネル内の物理ストレージデバイス(HDD と SSD)へのアクセス要求を調査したところ，data No. 2 へのアクセスの一部がキャッシュミスとなり，HDD から読み込んでいることが分かった．これが SSD と同等にならない原因であると考えられる．

6. おわりに

本稿では，ゲノム配列の秘匿検索アプリケーションの I/O 性能についての考察を行い，Flashcache を用いた事前読込手法を提案した．そして，性能評価を行い，提案手法の有効性を確認した．

今後は，SSD への事前読み込み要求をサポートしたシステムの実装を行う予定である．

謝辞

本研究は，JST，CREST JPMJCR1503 の支援を受けたものである．

本研究は JSPS 科研費 26730040, 15H02696, 17K00109 の助成を受けたものである．

参考文献

[1] 石巻優, 清水佳奈, 縫田光司, 山名早人, “完全準同型暗号を用いた高速なゲノム秘匿検索”, 2016 Symposium on Cryptography and Information Security (SCIS2016), 2A2-2 (2016).