Gfarm のための CDC による重複排除キャッシュ機構の実装と評価

村上 じゅん† 石黒 駿 † 大山 恵弘†,††

1. はじめに

近年のデータの大規模化に伴い，大規模データを扱うアプリケーションやその実行基盤の研究が盛んに行われている．そのような研究として，Gfarm[1] が挙げられる．Gfarm は広域分散ファイルシステムであり，大規模データ解析に利用可能である．ゲノムデータを始めとする大規模データの中には，重複するデータが多く含まれる．しかし，現在の Gfarm の実装にはこれらの重複データを排除する機構は存在しない．そこで本研究では Gfarm に対し重複データを排除するようなキャッシュ機構の導入，及び評価を行った．

2. Gfarm

Gfarm は大規模データ解析のための広域分散ファイルシステムである．Gfarm は単一のメタデータサーバ，複数の IO サーバおよびクライアントから成る．クライアントはファイルアクセスを行う際，まずメタデータサーバに問い合わせ，当該ファイルを格納する IO サーバについての情報を得る．その後はメタデータサーバを介さず IO サーバに対して直接ファイルデータを要求する．本研究では，このクライアントと IO サーバの間の通信プロトコルを変更することで実装を行った．

3. CDC

CDC (Content-Defined Chunking) とは，ファイルをその内容に基づいて可変長のチャンクに分割する方式であり，LBFS[2] により用いられたのが最初である．図 1 は CDC によるチャンク分割法を示したものである．CDC ではチャンクの境界位置を定めるために固定長（典型的には 48 バイト長）のウインドウが用いられる．この方法では，ウインドウをファイルの先頭から 1 バイトずつスライドさせ，ウインドウに含まれるデータをハッシュ値に変換する．このハッシュ値の下位数ビットが特定の値と一致したときに，該当するウインドウの終点をチャンクの境界とみなすことでファイルの分割が行われる．このときに下位何ビットまでを用いるかによってチャンクのサイズの平均が決まる．これは平均チャンクサイズと呼ばれる CDC のパラメータの一つである．

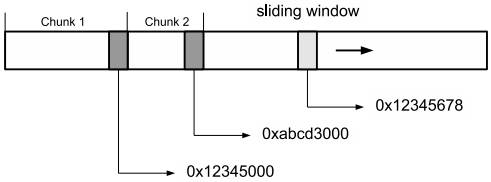

図 1 CDC によるチャンク分割

4. 提案機構

本機構導入後の Gfarm におけるファイルアクセスの流れを以下に示す．まずクライアントはファイルオープン時にハッシュ表を IO サーバに要求する．ハッシュ表とは，ファイルを構成するチャンクのリストであり，各チャンクのファイル内オフセット及び SHA-1 ハッシュ値をエントリに持つ．ハッシュ表は事前に計算され，IO サーバ上にファイルとして保存される．そのファイル名はハッシュ表が表すファイルの世代番号を元に作成される．クライアントはハッシュ表を元に read 対象部分を含むチャンクを求め，それらの SHA-1 ハッシュ値からチャンクファイルがクライアントのローカルディスク上に存在するかどうかを調べる．チャンクファイルは SHA-1 ハッシュ値をファイル名とするファイルであり，その中身がチャンクデータを表す．チャンクファイルがローカルディスク上に存在する場合にはそのファイルを読み，存在しない場合だけ IO サーバにそのチャンクデータを要求する．また IO サーバから新たに得られたデータは，チャンクファイルとしてクライアントのローカルディスクに保存される．ファイルへの書き込み時の処理については，現在設計と実

† 電気通信大学
The University of Electro-Communications
†† 独立行政法人科学技術振興機構， CREST
JST，CREST

表 1 実験環境

CPU	Intel Core i7 3.60GHz
Memory	16GB
OS	CentOS 5.7 64bit
kernel	ver 2.6.18
Gfarm	ver 2.4.2

表 2 提案機構導入前後の read 実行時間の比較

ファイル内容	ファイルサイズ	導入前	導入後	前後比
ランダム	10MB	0.25s	0.28s	1.12
	100MB	1.3s	1.15s	0.885
ゼロのみ	2GB	24.4s	11.9s	0.488

装を進めているところである．

5. 評　　価

提案機構導入による read 性能の上限を調べるための実験を行った．Gfarm ファイルシステム上のファイルを先頭から末尾まで 1MB ずつシーケンシャルに読み込むのにかかる時間を計測した．計測は内容・サイズの異なる複数のファイルに対してそれぞれ複数回行った．読み込まれるファイルがメモリに載った状態での性能を調べるため，最初の数回を除いた平均値を求めた．提案機構導入後の計測は，ファイルを構成するチャンクが全てクライアントのローカルに存在する状態で行い，全てのチャンクがローカルから読み出される場合の性能を調べた．また，チャンク分割における平均チャンクサイズは 4KB とした．実験環境は表 1 の通りである．

実験結果を表 2 に示す．ファイルサイズが小さい場合は導入前後における性能の変化はそれほど大きくない．ファイル内容が全てゼロのファイルの読み込みにおける実行時間比は 0.488 であった．この場合，ファイルを構成するチャンクはただ 1 つであり、読み出し時には単一のチャンクのみが読み出されている．この結果より，理想的な場合では実行時間が導入前後でおよそ半減することが分かった．

6. 現状と今後

CDC を用いた重複排除を行うキャッシュ機構の実装・評価を行った．今後はより効率的なハッシュ表の更新を行えるようシステムを改良する．また実用的なアプリケーションを用いた評価も積極的に行う．

謝辞　本研究を行うにあたって，有益な助言を頂いた筑波大学建部研究室の方々に深く感謝する．また本研究は，科学技術振興機構戦略的創造研究推進事業（JST CREST）の研究課題「ポストペタスケールデータインテンシブサイエンスのためのシステムソフトウェア」の支援を受けている．

参 考 文 献

1) O. Tatebe, K. Hiraga and N. Soda : Gfarm Grid File System, *New Generation Computing*, Ohmsha, Ltd. and Springer, Vol. 28, No. 3, pp. 257–275 (2010).
2) A. Muthitacharoen, B. Chen and D. Mazieres : A Low-bandwidth Network File System, In *Proceedings of the 18th ACM SOSP* (2001).