Hash-based Metadata Management for Parallel File System

Keiji Yamamoto*, Yoshiyuki Ohno*, Atsushi Hori*, and Yutaka Ishikawa*†
* Advanced Institute of Computational Science, RIKEN
† Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo

I. 背景

HPC では大量の I/O を高速に扱うために並列ファイルシステムが用いられている。一般に並列ファイルシステムではファイルはデータとメタデータに分けて保存される。データは複数のストレージに並列に読み書きされるため、高スループットが得られる。メタデータはメタデータ管理ノードに負荷が集中すると、性能上のボトルネックとなる。

近年、数万以上のプロセスから構成されるジョブも実行されるようになり、それらジョブの I/O によりメタデータ管理ノードの負荷が無視できないようになってきた。現在 HPC でよく利用されている Lustre File System は、メタデータを単一ノードで管理する。このため、メタデータサーバに負荷が集中し、ファイルシステムの性能はスケールしない。一方、メタデータを複数のノードで分散して保存する Ceph File System がオープンソースで開発されている。しかし、Ceph はメタデータをディレクトリ単位で分散するため、特定のディレクトリで多くのファイルアクセスを行う多くの HPC アプリケーションでは負荷が単一のメタデータ管理ノードに集中し、ファイルシステムのボトルネックとなる。

II. 提案手法

メタデータ管理ノードの負荷を分散するため、ファイル単位でメタデータを複数のメタデータ管理ノードに分散配置する手法を提案する。メタデータ管理ノードはハッシュ関数を用いてメタデータから求める。同じディレクトリに多数のファイルがあっても、それらのメタデータはメタデータ管理ノード群に分散配置され、メタデータ管理ノードの負荷は均等になる。

一方で、多数のクライアントがひとつのファイルにアクセスする場合を考える。この場合、ひとつのメタデータ管理ノードに多数のクライアントからのアクセスが集中し、ハッシュを使ったとしても性能上のボトルネックとなる。よって、クライアントからのメタデータアクセスを集約するプロクシノードを設ける。

メタデータ管理ノードはメタデータを読み書きするストレージを持つが、ストレージが HDD の場合はディスクのシーク時間 (約 10ms) を考慮すると高々毎秒 100 アクセスが上限である。このようなディクスのボトルネックは OS のファイルキャッシュにより無視できる場合も多いが、性能がメタデータ管理ノードの空きメモリ量に依存することになる。そこで、キャッシュノードを設け、メタデータ管理ノードのメモリ量に依存しないキャッシュを実現する。

図 1 にこれらの構成要素を示す。MDS はメタデータ管理ノードである。各クライアントはメタデータ操作を行う場合には、すべてプロクシノードにリクエストを投げる。クライアントが接続するプロクシノードは事前に決めておき、プロクシノードの負荷が片寄らないようにする。プロクシノードはメタデータのハッシュを計算し、メタデータがどのキャッシュノードにキャッシュされる求め、キャッシュノードにクライアントからのリクエストを転送する。キャッシュノードはプロクシノードからのメタデータリクエストがキャッシュにヒットし、副作用もなければキャッシュの内容を返す。キャッシュにヒットしなければ、メタデータのハッシュを計算し、メタデータ管理ノードを求め、プロクシノードからのリクエストをメタデータ管理ノードに転送する。メタデータ管理ノードからの返事は、キャッシュノードが適切にキャッシュしリクエスト元のプロクシノードそしてクライアントノードに転送する。

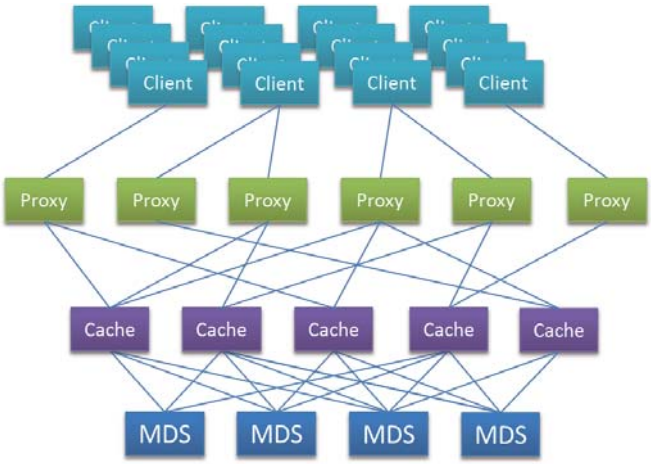

Fig. 1. Design of Metadata Management Node

本手法の利点は、プロクシノードとキャッシュノードを負荷によって適切に用意することで、スケーラビリティを確保していることである。一方で、クライアントとメタデータ管理ノード間にプロクシノードとキャッシュノードを 2 層はさむため、ファイルオープン時のレイテンシが大きくなる。これはファイルの逐次アクセス (例えば tar ファイルの展開) で顕著である。