2E-02

汎用分散処理フレームワークによる高精細映像符号化処理の検証

小田 周平† 黒住 正顕† 遠藤 洋介†
日本放送協会 放送技術研究所†

１．はじめに

インターネットによる動画配信システムを構築する際には、近年の配信対象端末の多様化に伴い、複数の配信用ファイルフォーマット生成を考慮する必要がある。また、システム構築後にも、新たに現れる未知のフォーマットへの対応が求められる。これに加え、番組制作用映像は従来のハイビジョン信号（2K 解像度）に加え 4K, 8K 解像度の高精細映像が普及することが予想され、映像信号の処理負荷はさらに増大する見込みである。以上を背景に、動画配信システムには映像信号の圧縮符号化及び多重化処理構成部分が柔軟に変更でき、拡張性があることが求められている。

従来の配信システムでは、出力フォーマットごとに専用の符号化装置を設置し、新たなフォーマットによる配信が必要となるたびに新規に符号化装置を追加している。これに対し、符号化処理を汎用サーバ上で動作するソフトウェアで構成することで、新フォーマットへの対応が容易となる。処理速度が遅い点については、処理を分散させて並列に符号化処理を行うことで改善できる。映像の符号化処理は、映像を時間単位で区切って処理するため、処理の分散化が容易である。これらの分析の元、我々は汎用の分散処理フレームワークである Hadoop[1]を利用した、高精細映像分散処理システムの構築を検討し、検証を進めている[2]。今回、8K 高精細映像の分散処理による高速符号化処理の実現可能性を把握するため、1000 台のノードによる分散処理を実行し、ノード数増加に対する処理速度向上の関係について検証した。また、実行結果から処理速度が期待値から低下する要因を分析し、ボトルネックとなっている処理割り当て時間の削減による速度向上と、ディスク入出力性能による高速化の限界を確認した。

２．検証内容と方法

分散処理フレームワーク Hadoop による、映像信号分散処理の実現可能性を検証するため、映像信号のフレーム圧縮処理を想定し、BMP 原画像の JPEG 圧縮処理を分散処理フレームワーク上に実装した。1000 台規模のノード上で大容量の超高精細映像データの分散処理を実行した事例はなく、今回初めて、1000 台のノードによる分散処理の実現の可否を含め、ノード数増加に対する処理速度向上性を調査した。

２．１ ノード数拡張性の検証

仮想サーバ提供サービスである、Google Compute Engine[3]上に 1000 台の仮想サーバを立ち上げ、Hadoop 分散処理ノードを稼働させた。また、8K 画像としてピクセル数 7680x4320 の BMP 画像（人物を含む自然画）を用意し、Hadoop File System (HDFS)上で 22 万枚に複製した。22 万枚の画像は、フレーム周波数 120Hz の映像再生時間 30 分間相当のデータとなる。Hadoop の処理ノードを 60 台から 1000 台まで 4 パターン増加させたときの 22 万枚の JPEG 圧縮処理完了時間を図 1 に示す。参考として処理完了までの速度がノード数の増加に反比例すると仮定した期待値の曲線を合わせて示した。なお、圧縮処理の実行にあたっては同じデータを繰り返し処理することでキャッシュヒット率が向上し処理速度が速くなってしまう影響を避けるため、処理ごとにキャッシュクリアを行っている。

本実験により Hadoop を利用して 1000 台の処理ノードによる分散処理が可能であることが実証できた。また、今回の分散処理系では、600 台程度の処理ノードで分散処理を行うことで 8K/120Hz 高精細映像のフレーム圧縮処理を映像再生の実時間以下で完了できることがわかった。一方、処理ノード数の増加に半比例して、処理速度が高速化するものの、処理速度の向上度は徐々に低下する傾向にあることが判明した。ロ

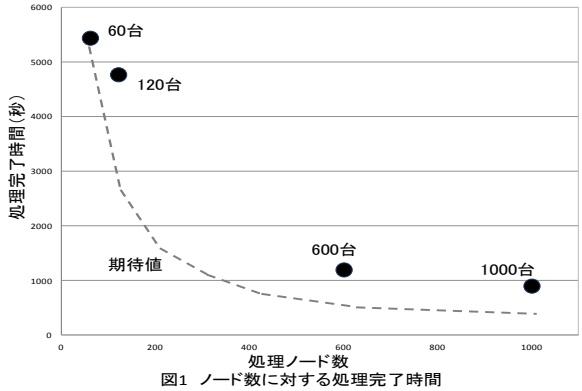

図1 ノード数に対する処理完了時間

Verification of ultra-high definition video coding on distributed processing framework
†Shuhei Oda, †Masaaki Kurozumi, †Yosuke Endo
†Japan Broadcasting Corporation (NHK)

グ解析の結果、処理ノード数の増加に対し、1 台のマスターノードによるタスク割り当て処理がボトルネックとなることが原因と推測された。図 1 に示したノード数の増加に対する処理完了時間の関係を以下の式 1 であると仮定した場合、

$$T = (A(n) + P) \times S / n \quad -(式1)$$

T: 処理完了時間, $A(n)$: 処理割り当て時間

P: 実処理時間, S: 処理枚数, n: 処理ノード数

式 1 の処理割り当て時間$A(n)$がノード数nの増加にしたがって増加するために処理完了時間が期待値ほど向上していないと考えられる。

２．２　処理速度向上手法の検証

1 枚の画像ごとに各ノードに対して圧縮処理を割り当てると、式 1 に示すように、実処理時間に加え、処理割り当て時間が必要となり、処理完了時間増大の要因となる。そこで割り当て時間を削減することによる処理完了時間の短縮方法を考案した。あらかじめ複数の画像を連結させて一つのファイルとしてから HDFS 上に展開の上、JPEG 圧縮プログラム内で複数の画像を一括して読み込み、同一の分散処理プログラム内で連続して複数画像の圧縮処理を行うことで、分散処理フレームワークによる圧縮処理割り当て回数を削減し、処理速度を向上させることを検証した。2K,4K,8K 各画像について、一括して圧縮処理を割り当てる画像数を増加させた際の 30 台のノードによる分散処理完了時間の計測結果を図 2 に示す。2K,4K 画像では 10 枚程度、8K 画像では 5 枚程度の画像を連結、一括して処理することで顕著な処理時間短縮効果が見られた。一方、10 枚以上画像を連結して一括処理しても処理速度は改善されず、8K 画像の場合はむしろ処理時間が増大する結果となった。ログ解析の結果、連結画像数を増やした場合、処理ファイルサイズが大きいためにディスクからの読み込みに時間がかかり、CPU アイドル時間が増加、処理が停滞していることが判明した。

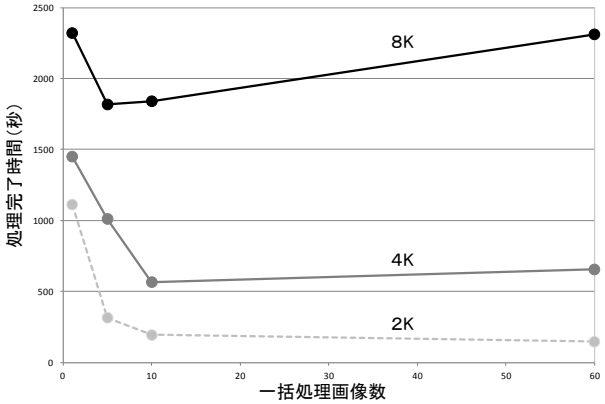

図2 一括処理画像数に対する処理完了時間

３．リソース利用率最適化に向けた考察

３．１　複数クラスタ再分散

処理ノード数を増加させた際に処理速度向上性が低下する結果は、Hadoop の設計上、処理ノードに対し処理を割り当てるマスターノードが常に 1 台であり、割り当て処理がボトルネックとなっていることに起因する。実行する処理が高負荷で、割り当て時間より十分長い処理時間を必要とする場合、すなわち、式 1 にて$P \gg A(n)$の際には割り当て時間は問題とならないものの、今回の処理のように$A(n) > P$となる際には、目立った性能劣化要因となる。

解決方法として、分散処理系をクラスタ化することが考えられる。1 台のマスターノードが管理する処理ノード数を制限し、複数のマスターノードからの結果を再統合するように分散処理系を階層化することで、ノード数増加による処理割り当て時間の増大を回避することが可能となる。

３．２　入出力最適化

あらかじめ画像を結合し、処理一括割り当てによりタスク割り当て回数を削減し処理の高速化を図ったところ、一定の結合枚数まで顕著な改善が見られたものの、ディスク読み込み時間がボトルネックとなり改善が頭打ちとなった。CPU コア数に合わせてディスク並列化を行い、スループットを向上させることで、CPU 利用率を上げ、処理時間を短縮できると予想される。すなわち処理ノードを構成する CPU,ディスク,ネットワークインターフェースなど、ハードウェアリソースの I/O スループットを揃えることが分散処理システムのリソース利用率向上に貢献する。

４．まとめ

Hadoop により 1000 台の大規模分散ノードで大容量 8K 高精細映像処理が実行可能であること、処理ノードのスペックにもよるが、目安として数 100 台程度の汎用サーバと分散処理フレームワークにより、映像再生時間以下の処理時間で 8K/120Hz 高精細映像のフレーム圧縮処理が可能であることを実証した。

参考文献

[1] Hadoop, “http://hadoop.apache.org/”
[2] 黒住ら, “大規模データ分散処理基盤を用いた映像トランスコード処理の検討, 映メ冬季大会研究報告,” 2014
[3] Google Compute Engine, “http://cloud.google.com/”