Web ブラウザ上での符号化処理の高速化

D.Mend – Amar† 村尾　裕一†
電気通信大学大学院　情報理工学研究科　情報・通信工学専攻

1　はじめに

近年、ネットワークの普及に伴ってクラウドサービスの発展が凄まじく、様々な用途に利用されている。クラウドサービスを使用するために、Web ブラウザ上からデータを直接サーバにアップロードするという方法がある。しかし、1 つのファイルの最大アップロード容量に制限があったり、アップロードするのに時間がかかり過ぎるなどの問題がある。そこで本稿で、Web ブラウザ上で高速にかつリアルタイムで符号処理の高速化を検討した。

2　LZSS 圧縮法

可逆データ圧縮の代表的なアルゴリズムとして LZSS 法がある。LZSS 法は実装が容易で高効率を誇るため LHA や ZIP といった様々な圧縮形式で用いられている [2]。LZSS 法は最近登場したパターンを辞書に登録し、同じパターンが登場した場合にそれに短い符号を振ったものを出力することでデータを圧縮する。辞書の長さは一定とし、辞書が満たされた場合には古いデータを削除する。

3　ブラウザ上での高速化法

ブラウザ上で符号化処理を高速化する際には、幾つかの方法が考えられる。本稿は、WebGL と Web Worker を用いて LZSS 圧縮手法の実装を検討した。現在、WebGL と Web Workers の両方が主要なブラウザにほとんど対応されていて、上手く活用することでとても高速な処理が可能になる。

4　設計と実装

読み込んだデータをスレッド数分に分割し、各スレッドに部分データに対する符号化処理を行わせる手法で並列化する。辞書は、スレッド毎に作成する。複数のスレッドでデータを細かく分割した場合に、圧縮率が逐次実行の場合と比較して低下してしまう問題がある。そこでスレッド開始時に辞書にスレッドの担当領域の直前のデータを与えることにする。以下の図 1 に示す。

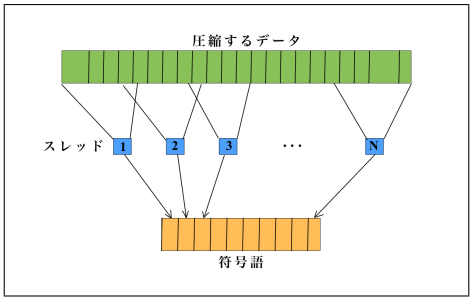

図 1: 並列化手法

5　性能評価

ファイルのサイズを変化させ、シングルススレッドとマルチスレッドによる並列処理時間の比較を行った。以下の図 2 より、CPU の場合はスレッド数が大きくなるほど実効時間が短縮化され、符号化処理が高速化されていることが分かる。

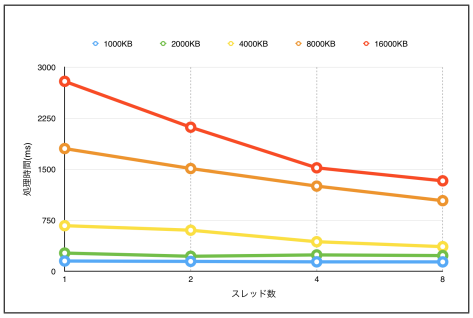

図 2: 並列処理の性能評価

6　まとめ

本稿で、Web ブラウザ上で符号処理の高速化を図り、通常のシングレススレッドで符号化処理行うより一定の高速化が得られた。高速にかつ高い圧縮率を目的に、可逆圧縮の代表的なアルゴリズム LZSS を用いた。最終的の結果としては、CPU 1 スレッドでの逐次処理より最大で 2.1 倍の速度向上を達成した。

参考文献

[1] A.L.V. Nicolaisen. 「Algorithms for Compression on GPUs」, Tecnical University of Denmark, DTU Compute.

[2] 昌達 K'z,「圧縮アルゴリズム：符号化の原理と C 言語による実装」, ソフトバンクパブリッシング, 2003.