B-006

# ベクトル量子化圧縮のコードブック生成のための Aggressive PNN 法に対する GPGPU 実装

## GPGPU implementation of VQ codebook generation using Aggressive PNN method

村上 明男† 若谷 彰良‡
Akio Murakami Akiyoshi Wakatani

## 1. はじめに

近年の GPU は高性能化が進み，汎用プロセッサの演算性能を大きく超えている．従来の GPU は専用ハードウェアで構成されていたので，グラフィック処理や画像処理に特化され，数値計算やデータベース処理などの汎用アプリケーションには用いられなかった．しかし，最近の GPU は CUDA (Compute Unified Device Architecture) [1] などにより，汎用プログラミング言語で GPU をプログラムできるようになっており，GPGPU (General-Purpose computing on Graphics Processing Units) と呼ばれる．GPGPU はデータ依存の少ない並列アプリケーションを容易に実装できるが，データ依存のために細かい制御が必要なアプリケーションへの適用が難しい．

一方，VQ (ベクトル量子化) 圧縮は画像や音声を含むマルチメディア情報を高圧縮率で圧縮するための手法の一つである．高圧縮率を実現するためには，少ないビット量で原情報を表現するための効率的なコードブックを生成することが重要である．本稿では，分散メモリ計算機向きに既に提案されている Aggressive PNN 法 [2] を元に，間接ベクトルを用いて GPGPU 向きに改良し，そのアルゴリズムに対する GPGPU の有効性について検討し，評価する．

本稿で用いる GPU*では，8 個の SP (Streaming Processor) を含む MP (Multi Processor) が演算実行を行う．演算はスレッド単位で行われ，複数のスレッドからなるスレッドブロックは shared memory を共有する．それ以外のメモリは全てのスレッドから読み書きでき，CPU と GPU は global memory を共有する．shared memory へのアクセス速度は高速であるが，それら以外は低速であり，global memory はキャッシュされない．そこで，global memory からのデータ転送には，同じ MP 内の複数の SP からの，連続したアドレスの global memory とのデータ転送を 1 個のメモリアクセス命令に coalesced (合体) した coalesced 通信を利用する．一方，MP に対して 32 スレッドがまとめて実行される．この実行単位をワープ と呼ぶ．

## 2. VQ 圧縮におけるコードブック生成

### 2.1 PNN 法と tau PNN 法

PNN (Pairwaise Nearest Neighbor) 法 [3] は 1) 原情報からあらかじめ決められた数(= T) の初期トレーニングベクトル (ベクトル) を選択し， 2) すべての 2 つのベクトル間の距離を計算し， 3) 距離が最小となるベクトルのペアをみつけてそれを融合 (マージ) し， 4) ベクトル数があらかじめ決められた数 (= K) になるまで 2 と 3 のステップを繰り返す方法である． ここで 2 のステップを距離計算ステップ，3 のステップをマージステップと呼ぶ．また，それぞれのベクトルの最近傍ベクトルをあらかじめ計算して PNN 法の計算量の軽減を図る tau PNN 法 [4] もあるが，アルゴリズムが逐次的であり，その速度向上は限定されたものである．

### 2.2 Aggressive PNN 法とその GPGPU 化

Aggressive PNN 法では， 距離計算ステップで計算した距離の中で，距離の小さいベクトルペアをあらかじめ決めた個数だけ選び，このベクトルペアをすべてマージする． この個数を AG 値 (Aggressive parameter) と呼ぶ． AG 値が大きくなれば，生成したコードブックの性能が悪くなる可能性がある．これについては後述の実験により精度の劣化を評価する．

Aggressive PNN 法は 2 つの方法で GPGPU 化する． 第 1 の方法 (aPNN1) では距離計算ステップは各ベクトルを一つの SP が担当し， 他のベクトルとの距離計算と最近傍ベクトルの検出と距離の最小値を計算する．また，マージステップはベクトルのマージを逐次に行う必要があるので原則として単一スレッド，すなわち単一 SP で実行するが，ベクトルの要素を shared memory にロードする際は複数 SP を用いる coalesced 通信で実現する．

第 2 の方法 (aPNN2) は間接ベクトルを用いる．例えば，A[0]，A[2]，A[9]，A[10] を連続した配列要素として参照するために，K[0]=0，K[1]=2，K[2]=9，K[3]=10 となる間接ベクトルを用意すれば A[K[0]]， A[K[1]]， A[K[2]]，A[K[3]] で参照できる．また，距離計算ステップでの距離計算は直前のマージステップに依存して計算すべきベクトルが決定される．

## 3. 実験結果と考察

PNN 法の実装に対し，compute capability 1.1 の GeForce 9500 GT を使った GPGPU システム (GPGPU1) と，compute capability 1.3 の Tesla C1060 を使った GPGPU システム (GPGPU2) において，T=2048 のトレーニングベクトルから始めて K=512 のコードブックを生成する場合の実行時間を計測し，評価した．GPGPU1 では，GPU は NVIDIA GeForce 9500 GT (32 コア) で，CPU は Intel Core 2 Duo E8400 (3 GHz)，メモリは 4 GB を用いる．また，OS は Windows 7 Professional 版を，CUDA toolkit は 3.2 を用いる．GPGPU2 では，GPU は NVIDIA Tesla C1060 (240 コア) で，CPU は AMD Phenom IIx4 945 (3 GHz)，メモリは 4 GB を用いる．また，OS は Windows 7 Ultimate 版を，CUDA toolkit は 3.2 を用いる．

†甲南大学大学院自然科学研究科

‡甲南大学知能情報学部

*本稿では扱わないが，現在 Fermi (NVIDIA GeForce GTX 590) での実行を検討中である．

GPGPU1 と GPGPU2 において，tau PNN 法の CPU および GPU の実装および Aggressive PNN 法の間接ベクトルを用いる場合 (aPNN1) と用いない場合 (aPNN2) の，それぞれの実行時間を図 1，図 2 に示す．

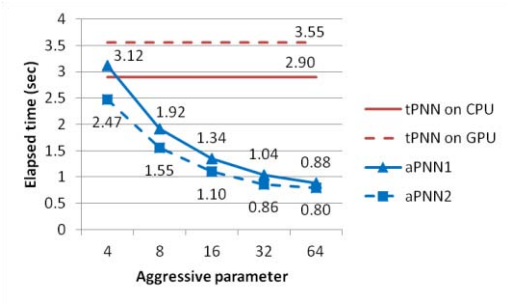


図 1: GPGPU1 における実行時間比較

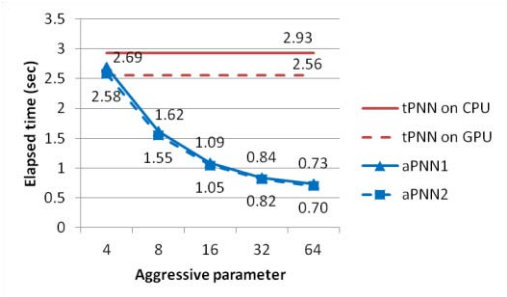


図 2: GPGPU2 における実行時間比較

図 1 より，GPGPU1 では，tau PNN 法では GPGPU 化するためのオーバーヘッドと並列度の低さにより，CPU 実装よりも GPGPU 実装のほうが遅い．Aggressive PNN 法では，AG 値を大きくすると実行時間が短縮され，AG 値が 64 のときに tau PNN 法の CPU と比べて 3 倍以上速い．これは，AG 値を上げることで並列度が高くなったことや，マージステップを行う回数が減少することで逐次実行の実行時間が減少するからである．

図 2 より，GPGPU2 では，GPGPU1 の場合と異なり，tau PNN 法では GPGPU 実装よりも CPU 実装のほうが遅くなる．また，Aggressive PNN 法では，AG 値が 64 のときに CPU と比べて最大で 4.01 (=2.93/0.73) 倍速くなる．この理由は GPGPU1 と同様である．

図 1，図 2 より，間接ベクトルを用いる aPNN1 と用いない aPNN2 の実行時間を AG 値別で比をとったグラフを図 3 に示す．GPGPU1 では，いずれの AG 値においても aPNN2 が aPNN1 を上回るが，AG 値の増加に伴い，aPNN2 の優位性が減少する．aPNN1 において，それぞれの MP で 16 個のワープが実行される．一方，aPNN2 において，128 個のスレッドまでは MP で実行ワープ数が 1 回になる．また，AG 値が高くなると並列度が高くなり，aPNN1 とのワープ回数の差が小さくなる．よって，AG 値を大きくすると，aPNN2 の優位性が減少する．

一方，GPGPU2 では，aPNN1 に対する aPNN2 の優位性が GPGPU1 の場合よりも小さい．aPNN1 において，それぞれの MP は 2 個もしくは 3 個のワープが実行される．一方，aPNN2 において，960 個のスレッドまでは MP は 1 回のワープ実行で済む．また，aPNN2 は間接ベクトルを用いた global memory へのアクセスを行う．よって，aPNN1 との実行時間の差は小さくなる．

また，tau PNN 法および Aggressive PNN 法による圧縮画像の画質を PSNR (Peak Signal Noise Ratio) で比較すると，AG 値が 16 までは 0.05 dB 程度の低下であるが，それ以上では，0.15 dB 程度の低下になる．AG 値を大きくすることで画像品質がさらに低下する可能性があるため，必要な画質を考慮し，AG 値を調整することが必要である．

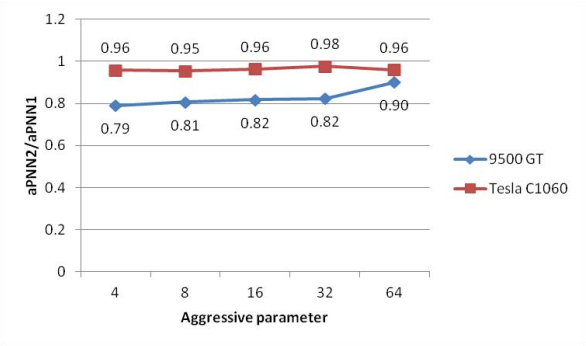


図 3: 間接ベクトル利用による実行時間の減少率

## 4. 終わりに

ベクトル量子化圧縮は効果的なコードブックを生成することが重要であるが，一般的にコードブック生成は計算量が大きく，膨大な実行時間を必要とする．そこで，複数のマージに同時に行う Aggressive PNN 法に対して，GPGPU 向けに間接ベクトルを適用した GPGPU 向け Aggressive PNN 法を提案し，圧縮性能を保ちながら並列性能の向上を図った．GPGPU 向け Aggressive PNN 法は tau PNN の CPU 実行に対して，最大で 4.01 倍性能が向上し，間接ベクトルを用いない Aggressive PNN と比較して 20%程度性能が改善した．

## 謝辞

本研究の一部は平成 24 年度私立大学等経常費補助金特別補助「大学間連携等による共同研究」によるものである．

## 参考文献

[1] “Parallel Programing and Computing Platform | CUDA | NVIDIA,” http://www.nvidia.com/object/cuda_home.html.

[2] Akiyoshi Wakatani, “Parallelization of VQ codebook generation using lazy PNN algorithm,” ParallelComputing: Software Technology, Algorithms, Architectures & Applications, Editors: G. Joubert et al.,Elsevier Science, pp. 415-422, 2004.

[3] Timo Kaukoranta, Pasi Franti and Olli Nevalainen, “Fast and space efficient PNN algorithm with delayed distance calculations,” in Proc. 8th International Conference on Computer Graphics and Visualization, pp. 219-224, 1988.

[4] William Equitz, “A new vector quantization clustering algorithm,” IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 37, no. 10, pp. 1568-1575, 1980.