

GPU を用いた Web データの並列処理

松本 翔飛[†] 山口 実靖[‡][†]工学院大学 工学部 情報通信工学科

1. はじめに

現在、Web 上には膨大な数のページが存在している。2008 年にページ数は 1 兆を超え、その後も拡大を続けている。この膨大な Web 空間から Web ページのコミュニティを抽出することにより、Web ページの分類や関連ページの検索が可能となり、効率の良い情報収集が可能になるといわれている。

しかし、大量の Web ページ群の中からコミュニティを抽出する処理には膨大な処理時間がかかる。そこで、本研究ではその処理を GPU を用いて行う手法に着目し、今回はデータが VRAM に格納できる範囲の抽出処理に適応したときの効果について考察を行う。

2. GPGPU

GPU を描画演算処理以外の汎用計算処理をさせる技術を GPGPU (General-purpose computing on graphics processing units) という。本研究では、著名な GPGPU 環境の一つである NVIDIA 社が提供している CUDA (Compute Unified Device Architecture) を使用する。CUDA は GPGPU 向けの統合開発環境であり、C 言語を拡張した言語仕様となっていることから描画機能の専門知識を必要とせずに GPU を用いた汎用計算を行うことが可能となっている。

CUDA にはスレッド、ブロック、グリッドという概念がある。スレッドは演算処理の最小単位であり、ブロック、グリッドはスレッドを管理するための概念である。スレッドをまとめたものをブロックと呼び、ブロックをまとめたものをグリッドと呼ぶ。

また、GPU は直接ホストのメインメモリにアクセスすることができないため、GPU に処理を行わせる場合はデータを一度 GPU のメモリ (VRAM) に転送させる必要がある。

3. Web コミュニティ抽出

Web コミュニティとは関連のある web ページをまとめたものである。コミュニティ抽出法にウェブページ間のリンク構造を用いた方法があり、本研究ではその一つである Reddy らの DBG 手法 [1] を用いた。

Parallel Web Date Processing using GPU

Tsubasa Matsumoto[†], Saneyasu Yamaguchi[†]

[‡]Department of Information and Communications Engineering, Kogakuin University

表 1 測定環境

	OS	CPU	GPU	Memory
Full-CPU	windows 7	Intel Core i7 950 -3.07GHz		8GB
hybrid_1bl	Fedora10 Linux(2.6.27.5)	Intel Core i7 920 -2.67GHz	Tesla C1060 -1.30 GHz	1GB
hybrid	Fedora10 Linux(2.6.27.5)	Intel Core i7 920 -2.67GHz	Tesla C1060 -1.30 GHz	1GB
Full-GPU_1bl	Fedora10 Linux(2.6.27.5)	Intel Core i7 920 -2.67GHz	Tesla C1060 -1.30 GHz	1GB
Full-GPU	Fedora10 Linux(2.6.27.5)	Intel Core i7 920 -2.67GHz	Tesla C1060 -1.30 GHz	1GB

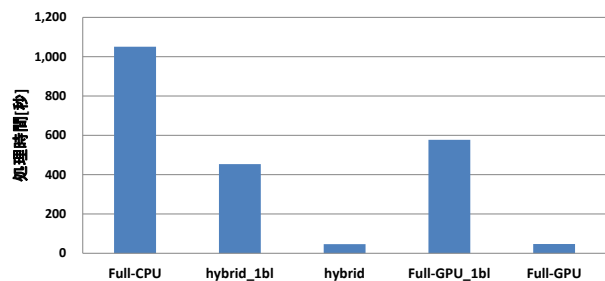


図 1 抽出時間

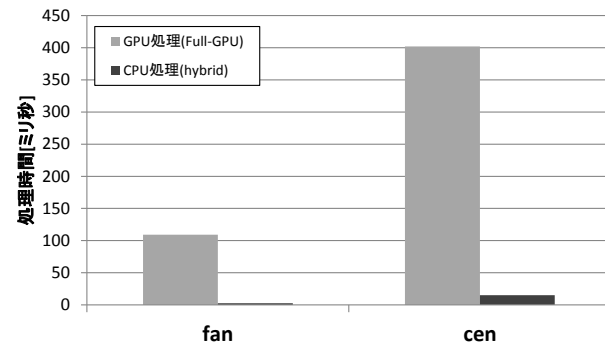


図 2 集計処理時間

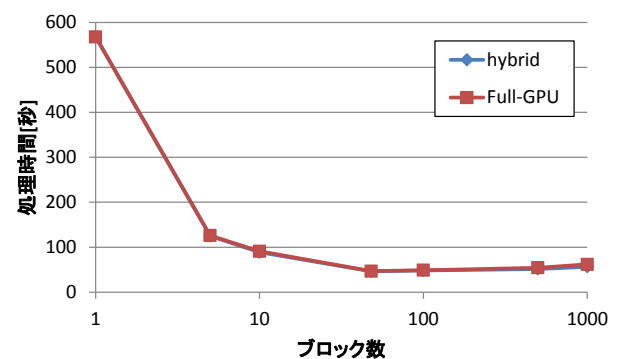


図 3 ブロック数と処理時間

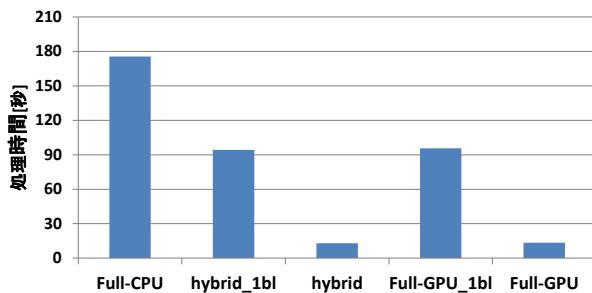


図4 抽出時間(閾値変更)

本手法では、コミュニティを centers、「コミュニティ内のページヘリンクを出すページ」の集合を fans と呼ぶ。まず初期の centers となるページ群を与える。「この centers に対して指定数以上のリンクを出すページ」の集合を fans と定め、初期 centers より fans を抽出する。次に、「この fans から指定数以上のリンクを受けるページ」の集合を次の centers とし、fans より centers を抽出する。以下同様に、centers からの fans の抽出と、fans からの centers の抽出を繰り返し、centers を更新していき、コミュニティを作成する。

4. 性能評価

上記 DBG 手法を CPU および GPU で実行し、その性能の比較を行った。測定には livedoor が提供している livedoor クリップ データセット[2]を用いた。その中で、US_ID を fans、URL_ID を centers とみなし処理をした。US_ID は 55,279 種類あり、URL_ID は 401,416 種類ある。US_ID と URL_ID の組数は 2,967,376 組ある。

評価実験では、(1)CPU のみを用いた場合の性能(Full-CPU)、(2)処理の中に CPU 処理と GPU 処理を混在させ VRAM-ホストメモリ間のデータ転送を繰り返しながら処理をした場合の性能(hybrid)、(3)途中の VRAM-ホストメモリ間のデータ転送を排除し全ての処理を GPU で行った場合の性能(Full-GPU)、の3種類を測定した。

hybrid では、同一データへのアクセスを伴い高い並列度での処理ができない集計計算部のみを CPU で行い、それ以外は GPU で行う。一時的に CPU による処理を行う場合は、一旦 CPU ヘデータを転送し、CPU で処理を行い、結果を GPU に送るといった処理が行われる。

Full-GPU は、hybrid にて CPU に処理させていた集計処理を並列化せずに GPU で行い、データ転送を省いたものである。性能評価は表1の装置を用いて行った。

図1に測定結果を示す。GPU を用いる手法(hybrid 手法と、Full-GPU 手法)では、ブロック

数1による処理と、ブロック数48による処理の両方を行った。1ブロックあたりのスレッド数は、いずれの場合も512である。

図1より、CPUによる処理が最も性能が低く、GPUを用いる処理(hybrid, Full-GPU)は1ブロック処理でCPU処理の約2倍、48ブロック処理でCPU処理の約20の高速化ができていたことが分かった。また、hybrid 手法と Full-GPU 手法を比較すると、1ブロック処理では hybrid の方が約23%性能が高く、48ブロック処理では同等の性能となった。両手法で集計計算部に要した時間は図2の通りであり、この処理(データ転送を含む)が全体に占める割合は fans 時で0.1%以下、centers 時で約1%である。よって、CPUにて集計処理を行うことにより集計処理の時間を大幅に削減することが可能であるが、集計処理は全処理の中の小さな構成要素に過ぎずその影響が限定的であることがわかる。

図3に hybrid, Full-GPU の、ブロック数と処理時間の関係を示す。図より、ブロック数48付近で最も高い性能が得られることがわかるが、ブロック数48以上における性能に大きな差はなく、十分な数のブロックを用意すれば良いと言える。

また、centers と判定する指定数を増やし、図1と同一の実験を行った結果を図4に示す。図4より、centers の数に依らず同等の結果が得られることがわかる。

5. まとめ

本研究では Web コミュニティの抽出処理を GPU を用いて行う手法に着目し、その処理性能の評価を行った。測定結果より、GPU を用いることによる大きな高速化が見られた。

今後は、さらなる大規模データへの適用について考察していく予定である。

謝辞

本研究は科研費(22700039)の助成を受けたものである。

参考文献

- [1] P. Krishna Reddy and Masaru Kitsuregawa, "An approach to relate the web communities through bipartite graphs," Proc. of the 2nd International Conference on Web Information Systems Engineering, 2001.
- [2] 「livedoor クリップ データセット」
<http://labs.edge.jp/datasets/>