POLITECHNIKA WARSZAWSKA
WYDZIAŁ MATEMATYKI
I NAUK INFORMACYJNYCH

PRACA DYPLOMOWA MAGISTERSKA

INFORMATYKA

Optymalizator kompresji szeregów czasowych na GPU

Autor:

Karol Dzitkowski

Promotor: dr inż. Krzysztof Kaczmarski

Warszawa, luty 2016

....

podpis promotora podpis autora

Abstrakt

Wiele urządzeń takich jak czujniki, stacje pomiarowe, czy nawet serwery, produkują ogromne ilości danych w postaci szeregów czasowych, które następnie są przetwarzane i składowane do późniejszej analizy. Ogromną rolę w tym procesie stanowi przetwarzanie danych na kartach graficznych w celu przyspieszenia obliczeń. Aby wydajnie korzystać z GPGPU przedstawiono szereg rozwiązań, korzystających z kart graficznych jako koprocesory w bazach danych lub nawet bazy danych po stronie GPU. We wszystkich rozwiązaniach bardzo istotną rolę stanowi kompresja danych. Szeregi czasowe są bardzo szczególnym rodzajem danych, dla których kluczowy jest dobór odpowiedniej kompresji wedle charakterystyki danych szeregu. W tej pracy przedstawię nowe podejście do kompresji szeregów czasowych po stronie GPU, przy użyciu planera budującego na bieżąco drzewa kompresji na podstawie statystyk napływających danych. Przedstawione rozwiązanie kompresuje dane za pomocą lekkich i bezstratnych kompresji w technologii CUDA.

Abstract

Many devices such as sensors, measuring stations or even servers produce enormous amounts of data in the form of time series, which are then processed and stored for later analysis. A huge role in this process takes data processing on graphics cards in order to accelerate calculations. To efficiently use the GPGPU a number of solutions has been presented, that use the GPU as a coprocessor in a databases. There were also attempts to create a GPU-side databases. It has been known that data compression plays here the crucial role. Time series are special kind of data, for which choosing the right compression according to the characteristics of the data series is essencial. In this paper I present a new approach to compression of time series on the side of the GPU, using a planner to keep building the compression tree based on statistics of incoming data. The solution compresses data using lightweight and lossless compression in CUDA technology.

Rozdział 1

Wstęp

Poniższa praca zawiera opis implementacji optymalizatora kompresji szeregów czasowych, bazującego na dynamicznie generowanych statystykach danych. Pomysł opiera się na tworzeniu drzew kompresji (kompresja kaskadowa) oraz zbieraniu statystyk o krawędziach takich drzew - jak dobrze dana para kompresji sprawdza się dla napływających danych. System będzie również dynamicznie zmieniał - korygował, takie drzewa w zależności od charakterystyki kolejnych paczek danych. W założeniu system ma umożliwić kompresję dużych ilości danych przy wykorzystaniu potencjału obliczeniowego współczesnych kart graficznych.

1.1 Procesory graficzne

Procesory graficzne stały się znaczącymi i potężnymi koprocesorami obliczeń dla wielu aplikacji i systemów, takich jak bazy danych, badania naukowe czy wyszukiwarki www. Nowoczesne GPU posiadają moc obliczeniową o rząd większą niż zwykle, wielordzeniowe procesory CPU, takie jak AMD FX 8XXX czy Intel Core i7. Dla przykładu flagowa konstrukcja firmy NVIDIA - GeForce GTX Titan X osiąga moc 6600 GFLOPS (miliardów operacji zmiennoprzecinkowych na sekundę), przy $336GB/s$ przepustowości pamięci, podczas gdy najszybsze procesory takie jak Intel Core i7-5960x osiągają niecałe 180 GFLOPS, przy przepustowości $68GB/s$. Kartom graficznym dorównują tylko inne jednostki typu SIMD, na przykład karty oblicze-

niowe Xeon Phi. Pomimo tak oszałamiających wyników, programowanie na jednostkach SIMD jest o wiele trudniejsze, jak również ograniczone przepustowością szyny PCI-E, która wynosi w porywach $8GB/s$, co dodatkowo przemawia za użyciem kompresji przy przetwarzaniu szeregów czasowych, choćby w celu przyspieszenia kopiowania danych z i na kartę graficzną w celu wykonania obliczeń.

1.2 Szeregi czasowe

Terabajty danych w postaci szeregów czasowych są przetwarzane i analizowane każdego dnia na całym świecie. Zapytania i agregacje na tak wielkich porcjach danych jest czasochłonne i wymaga dużej ilości zasobów. Aby zmierzyć się z tym problemem, powstały wyspecjalizowane bazy danych, wspierające analizę szeregów czasowych. Ważnym czynnikiem w tych rozwiązaniach jest kompresja oraz użycie procesorów graficznych w celu przyspieszenia obliczeń. Aby przetwarzać dane na GPU bez konieczności ich ciągłego kopiowania poprzez szynę PCI-E, powstają bazy danych po stronie GPU (najczęściej rozproszone), takie jak MapD lub DDJ (zaproponowana między innymi przeze mnie w poprzedniej pracy - inżynierskiej). Innymi rozwiązaniami są koprocesory obliczeniowe GPU, wspomagające działanie baz takich jak Cassandra, HBase, TepoDB, OpenTSDB czy PostgreSQL. Charakterystyka danych wielu szeregów wskazuje, że przy odpowiedniej obróbce mogą być kompresowane z bardzo dużym współczynnikiem, szczególnie jeśli byłoby możliwe kompresowanie za pomocą dynamicznie zmieniających się ciągów (różnych) algorytmów kompresji i transformacji danych. Dla przykładu, jeśli jakiś fragment szeregu jest stały, z nielicznymi wyjątkami, warto byłoby usunąć wyjątki, a resztę skompresować jako jedną liczbę - uzyskując współczynnik kompresji rzędu długości danych.

1.3 SIMD i lekka kompresja

Bazy danych przechowujące szeregi czasowe są najczęściej zorientowane kolumnowo oraz stosują metody lekkiej kompresji w celu oszczędności pamięci. W tych przypadkach stosuje się metody lekkiej kompresji, takie jak kodowanie słownikowe, delta lub stałej liczby bitów, zamiast bardziej skomplikowanych i wolniejszych metod, które często zapewniłyby lepszy poziom kompresji. Systemy te ładują swoje dane do pamięci trwałej paczkami, które mogą być kompresowane osobo i być może przy użyciu różnych algorytmów, zmieniających się dynamicznie w czasie. Takie kolumny wartości numerycznych tego samego typu wspaniale przetwarza się przy użyciu procesorów typu SIMD, co daje wielokrotne przyspieszenie w stosunku do tradycyjnych architektur. Okazuje się że większość algorytmów lekkiej kompresji może z dużym powodzeniem być w ten sposób zrównoleglona. Również dynamiczne generowanie statystyk napływających danych może być przyspieszone z użyciem SIMD, co otwiera możliwość implementacji wydajnych systemów, dynamicznie optymalizujących użyte kompresje w celu zwiększenia współczynnika kompresji danych. Dodatkowo użycie kaskadowej kompresji może wielokrotnie wzmocnić poziom kompresji, pod warunkiem stworzenia dobrego planu kompresji, właśnie na podstawie wygenerowanych statystyk. Ogromna moc obliczeniowa procesorów graficznych może pozwolić wygenerować taki plan w rozsądnym czasie. Takie użycie jest możliwe np. w bazach danych po stronie GPU, gdzie jest to niezmiernie ważne z powodu ścisłego limitu pamięci na kartach i ich wysokiego kosztu.

1.4 Zawartość pracy

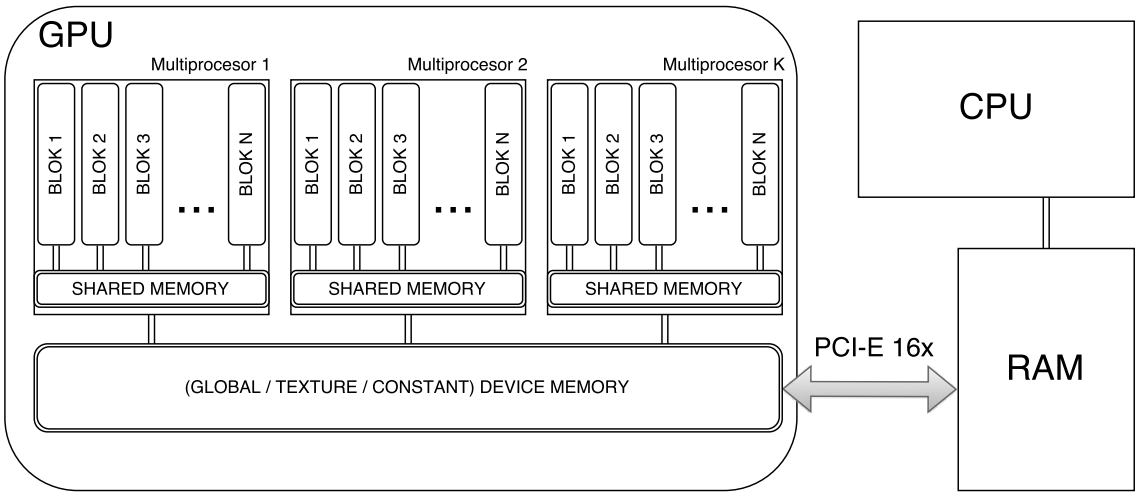
W tej pracy przedstawię planer kompresji (optymalizator) kompresujący napływające paczki danych. Zaprezentuję nowe podejście, planera budującego drzewa kompresji i uczącego się ich konstrukcji na podstawie na bieżąco generowanych statystyk węzłów takich drzew (jak również statystyk napływających danych). Przedstawię również zaimplementowane środowisko oraz użyte algorytmy lekkiej kompresji. W

ramach tej pracy stworzone zostały 4 biblioteki, wykorzystujące technologię NVIDIA CUDA, tworzące framework optymalizatora kompresji oraz program w sposób równoległy kompresujący kolumny podanego szeregu czasowego. Rozdział 2 zawiera opis wcześniejszych prac prowadzonych w tych tematach, a także krótki opis architektury CUDA. W rozdziale 3 omówię stworzony framework oraz metody lekkiej kompresji, ze szczególnym uwzględnieniem kompresji FL oraz GFC. Rozdział 4 jest w całości poświęcony optymalizatorowi kompresji oraz generowaniu drzew i statystyk. Następnie przedstawię wyniki prac i eksperymentów. Ostatni rozdział (szósty) to podsumowanie oraz zakres przyszłych prac i optymalizacji.

Rozdział 2

Wcześniejsze prace

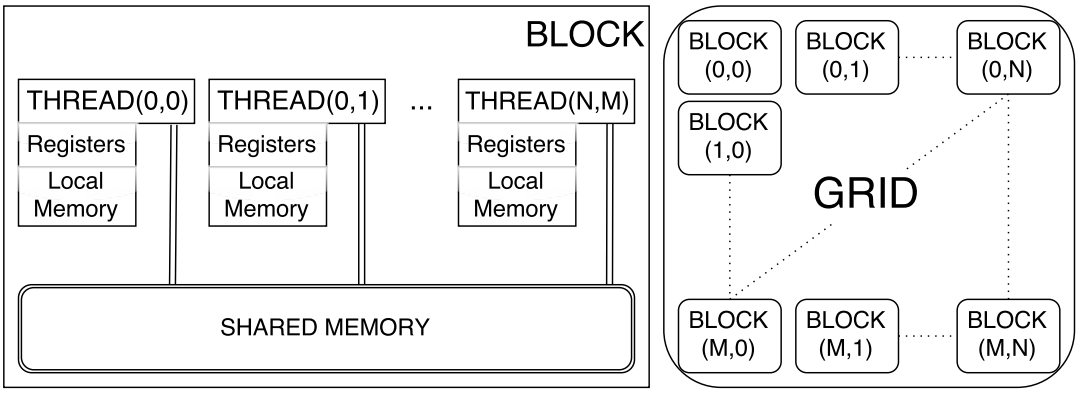
2.1 CUDA



Rysunek 2.1: Architektura NVIDIA CUDA

Jedną z wielu zalet architektury kart graficznych jest to, że składają się z kilku multiprocesorów (SMs - Streaming Multiprocessors) architektury SIMD. Jest to właściwie architektura typu SIMT, gdzie multiprocesor wykonuje wątki w grupach o liczności 32 zwanymi *warps*. Architektura ta jest zbliżona do SIMD z tą różnicą, że to nie organizacja wektora danych kontroluje jednostki obliczeniowe, a organizacja instrukcji pojedyńczego wątku. Umożliwia on zatem pisanie równolegle wykonywanego kodu dla niezależnych i skalowalnych wątków, jak i dla wątkuch koordy-

nowanych danymi. Wszystkie wątki wykonują ten sam kod funkcji kernela. Ponadto CUDA tworzy abstrakcję bloków wątków, które zorganizowane są w siatkę (GRID) i współdzielą zasoby multiprocesora. Ważna jest również hierarchia pamięci, w której część przydzielana jest wątkom w postaci pamięci lokalnej i rejestrów, oraz pamięci współdzielonej przez wątki z tego samego bloku (Shared Memory) - te pamięci muszą być znane w trakcie kompilacji kernela. Najwolniejsza jest pamięć globalna (Device Memory), wspólna dla wszystkich wątków, wszystkich bloków, na wszystkich multiprocesorach. Dokładny opis tej architektury można znaleźć bezpośrednio na stronie producenta.



Rysunek 2.2: Abstrakcja bloków i siatki w CUDA

Rozdział 3

Opis systemu

Rozdział 4

Optymalizator kompresji

Rozdział 5

Wyniki

Rozdział 6

Podsumowanie

Bibliografia

[1] Stanisław Białas, *Macierze. Wybrane problemy*, AGH Uczelniane Wydawnictwa Naukowo-Dydaktyczne, Kraków, 2006.

[2] Nicholas J. Higham, *Accuracy and stability of numerical algorithms*, SIAM, Philadelphia 1996.

[3] Nicholas J. Higham, *Functions of Matrices. Theory and Computation*, SIAM, Philadelphia 2008.

[4] Maksymilian Dryja, Janina i Michał Jankowscy, *Przegląd metod i algorytmów numerycznych, część 2*, Wydawnictwa Naukowo-Techiczne, Warszawa 1982.

[5] Mostak, T., 2013. *An overview of MapD (massively parallel database). White paper*, Massachusetts Institute of Technology, Cambridge, MA.

Spis treści

Warszawa, dnia .....

Oświadczenie

Oświadczam, że pracę magisterską pod tytułem: „Tytuł pracy”, której promotorem jest prof. dr hab. Jan Wybitny, wykonałem/am samodzielnie, co poświadczam własnoręcznym podpisem.

.....