Szczecin, 09 kwietnia 2024

Dr hab. Inż. Marek Pałkowski, prof. ZUT
Wydział Informatyki
Zachodniopomorski Uniwersytet Technologiczny
w Szczecinie

RECENZJA

rozprawy doktorskiej mgr Beaty Dmitruk

„Równoległe i wektorowe algorytmy rozwiązywania trójdiagonalnych układów równań liniowych typu Toeplitza na współczesnych architekturach wieloprocesorowych"

Promotor: dr hab. inż. Przemysław Stpiczyński

Rozprawa doktorska mgr Beaty Dmitruk dotyczy zrównoleglenia i wektoryzacji algorytmów rozwiązywania trójdiagonalnych układów równań liniowych za pomocą współczesnych komputerów wieloprocesorowych. Autorka skupia się na algorytmach numerycznych z macierzą współczynników trójdiagonalną typu Toeplitza oraz dokładniejszych metod sumujących. Wynikiem pracy badawczej są sformułowane algorytmy wraz z ich implementacją w języku C z wykorzystaniem procesorów wielordzeniowych włącznie z układami graficznymi. Poruszony jest także problem kolumnowego i wierszowego formatu danych wraz z poprawą lokalności kodu programowego. Praca badawcza poparta jest licznymi badaniami oraz przyzwoitym cyklem publikacji w specjalistycznych czasopismach oraz konferencjach międzynarodowych.

We wstępie pracy poruszany jest kluczowy problem optymalizacji kodu algorytmów numerycznych w ujęciu sprzęt - algorytmy - kompilator. Celem tego procesu jest uzyskanie efektywnego kodu, przy czym efektywny oznacza szybciej i dokładniej. Następnie autorka uzasadnia wybór badanych algorytmów numerycznych w rozdziale drugim. Zastosowanie układów z macierzą współczynników Toeplitza odnaleźć można w numerycznych rozwiązaniach warunków brzegowych równań różniczkowych zwyczajnych lub cząstkowych oraz drugiego rzędu, różnicach skończonych dla problemów 1-D oraz 2-D, szybkich metodach rozwiązywania równań Poissona, funkcjach sklejanych sześciennych i wielu innych. W kolejnych punktach tego rozdziału omawiane są dotychczasowe uznane algorytmy tj. algorytm Thomasa, który jest specjalnym przypadkiem eliminacji Gaussa, metoda rekurencyjnego podwajania, metoda cyklicznej redukcji i metoda Wanga. Opisany jest też stan istniejącego

oprogramowania ze szczególnym uwzględnieniem znanej biblioteki LAPACK i funkcji eliminacji Gaussa z częściowym wyborem elementu głównego, *dgtsv*.

Kolejne rozdziały prezentują sformułowane algorytmy wraz z ich metodą zrównoleglenia oraz wektoryzacją i częścią badawczą na wybranych maszynach równoległych. W rozdziale trzecim zaprezentowano dwie wersje zrównoleglonego i zwektoryzowanego algorytmu typu *dziel i zwyciężaj* dla rozwiązywania układów równań liniowych postaci (1,t2,t3). W implementacji rozwiązania algorytmu Thomasa wykazano jego sekwencyjną naturę, tzn. jest to szereg pięciu pętli programowych, których zależności danych uniemożliwiają zrównoleglenie. W celu wektoryzacji tego podejścia skorzystano z przekształcenia do tzw. macierzy blokowych umożliwiających na efektywne zrównoleglenie rozwiązania w formacie kolumnowym. Wynikiem są dwie wersje zoptymalizowanego algorytmu Thomasa przy użyciu pragm OpenMP. Zbadano czas i przyspieszenie opracowanego kodu dla różnych rozmiarów problemu z uwzględnieniem architektury MIC. Obie wersje cechują się także podobnym, w przybliżeniu dwukrotnie mniejszym poborem mocy w trakcie obliczeń w porównaniu do sekwencyjnej implementacji.

W następnym rozdziale dokonano optymalizacji ogólniejszej postaci rozwiązywania układów równań liniowych (t1, t2, t3) dla dwóch algorytmów typu *dziel i zwyciężaj*. Implementację rozszerzono o układy graficzne i standard OpenACC. Rozdział też poszerza analizę formatu danych o postacie kolumnowe i wierszowe oraz ich cztery wersje zapisane w jednowymiarowej tablicy. Analiza badawcza porusza także problem dokładności wyników i przeprowadzono próbę predykcji parametrów programu. Pierwszym algorytmem, którego dokonano optymalizacji kodu jest metoda Wanga, w której ponownie użyto macierzy blokowych i odpowiedniej dekompozycji podobnie jak w rozdziale poprzednim. Drugi algorytm zaproponowany przez zespół Liu został także zrównoleglony i zwkektoryzowany za pomocą równoważnej formy blokowej. Po odpowiednich przekształceniach wersji sekwencyjnej wyznaczenie niezbędnych wektorów i rozwiązania układu odbywa się w sposób równoległy. W obu przypadkach w pracy przedstawiono niezbędne dowody na poprawność zastosowanych przekształceń. Rozdział przedstawia listingi obu kodów w wersjach kolumnowych i wierszowych wraz z ich modyfikacjami oraz implementacją na wiele urządzeń GPU i GPU oraz GPU i CPU. Wyniki przedstawiono na pięciu platformach NVIDIA dla kodów kolumnowych, wierszowych oraz z blokiem cache oraz dokonano porównania ze wspomnianą wcześniej funkcja *dgtsv* z biblioteki LAPACK włącznie pod kątem dokładności wyników. Na uwagę zasługuje bardzo rozbudowany pakiet badawczy, który przeprowadzono w tym rozdziale.

W rozdziale piątym zaproponowana została optymalizacja kodów algorytmów sumowania z poprawkami. Sumowanie iteracyjne może być potraktowane jako szczególny przypadek rozwiązania układu równań typu Toeplitza poprzez wyliczenie sum częściowych. Na uwagę zasługuje fakt szczegółowego analizy literatury pokrewnej opisującej rozwiązania,

które dają dużo lepszą dokładność niż proste sumowanie iteracyjne opublikowane w ciągu ostatniej dekady. Przeanalizowano prace rekurencyjnych algorytmów sumowania z z wyliczanymi wartościami poprawek ze szczególnym uwzględnieniem dwóch znanych podejść algorytmów Kahana i Gilla-Mollera. Oba algorytmy zostały zrównoleglone i zwektoryzowane pomimo ich sekwencyjnej natury, uwzględniono dokładność oraz wprowadzono dodatkowo precyzję mieszaną. Zrównoleglenie uzyskano dzięki redukcji OpenMP oraz przedeklarowania tej operacji funkcjami wbudowanymi. Zoptymalizowane kody zbadano przy użyciu instrukcji Intel AVX-512 i maszyny z dwoma procesorami Intel Xeon Gold i kompilatorami OneAPI z wysokowydajną biblioteką numeryczną MKL oraz włączonym najwyższym stopniem optymalizacji O3. Na uwagę zasługuje fakt zbadanie konfiguracji maszyny gniazdo, rdzenie, wątki - najlepszym ustawieniem okazał się jeden procesor w gnieździe z 24 jednowątkowymi rdzeniami. W badaniach interesujące jest to, że konfiguracja obliczeń podyktowana jest niekoniecznie dwoma skorelowanymi czynnikami, przyspieszeniem i dokładnością. Ostatecznie przedstawiona zostaje analiza porównawcza algorytmów Kaha i Gilla-Mollera z dodatkową wersją precyzji mieszanej, która wzmacnia znacząco dyskusję badań.

Omawiane podejścia następnie zostały wdrożone w rozdziale szóstym do obliczeń całek numerycznych metodą prostokątną, trapezową i Simpsona oraz do rozwiązywania układów równań przy wykorzystaniu uproszczonej wersji eliminacji Gaussa bez pivotingu do macierzy trójdiagonalnych, tj. za pomocą omawianego wcześniej algorytmu Thomasa. Implementacje algorytmów do obliczania całek numerycznych zostały porównane z klasycznymi podejściami, gdzie zamiana przedziału wykonywana jest przez sumowanie lub mnożenie. W drugim problemie badawczym porównano dokładność z klasycznym podejściem w wersji sekwencyjnej i równoległej. Badania przeprowadzono na nieco starszym układzie graficznym Keplera (bodajże architektura ta wypuszczona była w latach 2012-2018), ale wykorzystując już kompilator NVC do obsługi pragm OpenACC. W badaniach zostały podane wartości przyspieszenia około 8,5 razy dla pojedynczej precyzji oraz 6 razy dla podwójnej.

Praca liczy 128 stron, a zatem została napisana zwięźle bez zbędnych rozszerzeń, co implikuje że jest monografią łatwą w odbiorze przez zaawansowanego jak i średnio zaawansowanego czytelnika. Praca została złożona profesjonalnie w systemie Latex, wszystkie matematyczne wzory i macierze zostały z dużą starannością zaprezentowane. Układ stron i spis treści nie budzi zastrzeżeń. Na część badawczą składa się kilkadziesiąt tabel i wykresów, co pokazuje szeroki wachlarz badań przeprowadzonych na poczet udowodnienia celu i tezy pracy. Do każdego rozwiązania przedstawiony został czytelny listing z kodem, a dodatkowo cała implementacja rozwiązań umieszczona w publicznym repozytorium, który pozwala na popularyzowanie i weryfikację prezentowanych rozwiązań. Podsumowując, rozprawa została wykonana edytorsko na wysokim poziomie, pozbawiona wszelkich wad składu treści oraz napisana w poprawnym języku polskim.

Rozprawa zawiera poprawną strukturę. We wstępie postawiono w sposób prawidłowy cel i tezę rozprawy wraz z jej postulatami. Dwa pierwsze rozdziały wprowadzają do tematyki obliczeń równoległych oraz układów równań liniowych. Rozdziały 3-6 prezentują tematykę badawczą dysertacji i są jej najważniejszą częścią. W podsumowaniu wykazano osiągnięcie celu oraz udowodniono tezę odpowiednimi sformułowaniami. Zasygnalizowano także kierunki kontynuacji badań.

Po wnikliwej analizie rozprawy, uważam, że postawione w rozprawie cel, tj. opracowanie wydajnych i dokładniejszych algorytmów do rozwiązywania układów równań liniowych z macierzą trójdiagonalną typu Toeplitza oraz udowodnieniu tezy, że rozwiązania te znacznie przyspieszają obliczenia na współczesnych architekturach wieloprocesorowych włącznie z konfiguracjami hybrydowymi zostały osiągnięte i poparte licznymi badaniami oraz cyklem publikacji. Mocną stronę rozprawy jest jednotematyczny zestaw problemów oraz wyznaczenie dziedzin i rozwiązań, w których znajdują one zastosowanie. Tematyka badawcza jest zatem kompletna, rozwijana z kolejnymi rozdziałami i publikacjami zgodnie z ich chronologią. To pokazuje, że idea dysertacji została dobrze przemyślana oraz zaplanowana. Początkowe pierwsze opracowywane algorytmy znajdują zastosowania w kolejnych problemach badawczych. Rozdział szósty konkluduje zastosowania proponowanych rozwiązań w popularnych algorytmach numerycznych. Jakość badań oraz tematyka zostały zaprezentowane na wysokim poziomie naukowym. Jednocześnie rozprawa napisana jest w języku przystępnym, a zastosowana terminologia techniczna nie budzi zastrzeżeń.

Uwagi / polemika

W temacie pracy pojawia się pojęcie współczesnej architektury wieloprocesorowej. W rozdziale 1 dokonałem próby odnalezienia definicji tego pojęcia. Na stronie 15 opisano, że ich cechą są wielordzeniowa budowa oraz wektoryzacja i rozkazy z rodziny AVX. Na następnej stronie dopisano, że nowe architektury wyposażane są w coraz bardziej złożoną strukturę pamięci. Na stronie 19 pojawia się podrozdział 1.3 architektury współczesnych komputerów, jednak nie podaje on definicji. Przytoczono zdanie i słusznie, że producenci dążą do m.in.. integracji procesorów z układami graficznymi (rozwiązania heterogeniczne). Rozdział I sądzę, nie wyczerpuje pojęcia przytoczonego w temacie dysertacji. Po pierwsze, brak informacji na temat procesorów AMD Epyc, które stanowią ważną konkurencję do układów Intela i z których zbudowany jest wciąż najmocniejszy superkomputer na świecie, Frontier. Układy AMD cechują się większą ilością pamięci podręcznej i rdzeni z jednoczesnym mniejszym poborem energii oraz mniejszą ceną. Z pewnością przeprowadzanie badań z ich wykorzystaniem mogłoby poszerzyć wiedzę o efektywności proponowanych rozwiązań. Warto też było wspomnieć o bieżących układach Intela Platinium, nowych CPU Max (druga maszyna z listy

Top500) oraz ARC, jak kształtują się takie charakterystyki jak ilość rdzeni, zegar, ilość pamięci Cache, litografia, nakłady energii. Wzmocnieniem pracy byłoby zatem nakreślenie jak zmieniają się procesory wielordzeniowe i jakie są kierunki rozwoju współczesnych architektur w ciągu ostatniej dekady.

Na stronie 20 do opisu zastosowanych architektur warto było zatem dopisać jaka jest litografia, wielkość pamięci cache, wspierane instrukcje wektoryzacji, czy rok produkcji. Użyty procesor Intel Xeon Gold jest 8 lat młodszy od układów Xeon Phi zbudowany z dwa razy mniejszych tranzystorów. Autorka podaje zegar 2.8GHz, lecz procesor w trybie turbo osiąga 3.5GHz. Pozostawiam także do dyskusji czy układ Intel Xeon Phi jest procesorem czy koprocesorem.

Drugą uwagą, czy raczej tematowi do dyskusji zostawiam słuszność zastosowania standardu OpenACC. W przypadku maszyn NVIDIA bardziej wydajną opcją wydawać się może zastosowanie niskopoziomowego programowania z zastosowaniem biblioteki CUDA. Docelowy kod będzie oczywiście bardziej skomplikowany i wyposażony o niewygodne instrukcje alokacji czy transferu danych. Jednak standardowi OpenACC zarzuca się mniejszą wydajność kosztem czasu implementacji. Wybór OpenACC w pracach doktorantki wydaje się dobry do udowodnienia tezy i szybkim opracowaniem kodu (skupiając się bardziej na algorytmach) zwłaszcza w miejscach gdzie region i gangi otwierane są wielokrotnie oraz możliwością wykonania na CPU. Pozostaje niemniej pytanie o koszt wydajności tego standardu. Czy bazując na znanej autorce literaturze można oszacować poprawę wydajności z zastosowaniem biblioteki CUDA do OpenACC? Niewątpliwie zastosowanie „surowej" biblioteki CUDA będzie ciekawym wątkiem kontynuowania badań nad wydajnością. Poza tym wciąż możliwe jest skorzystanie z alternatywnego rozwiązania, tj. pragm OpenMP dla kart GPU, które są wspierane od wersji 4.5 i dokonanie porównania wydajności obu standardów. W badaniach nie rozważono tego rozwiązania.

W rozdziale pierwszym nie przytoczono terminologii mierników jakości kodu zoptymalizowanego, a przecież proponowane rozwiązania znacznie poprawiają te parametry. Po pierwsze, nie wyjaśniono pojęcia lokalności kodu, który jest celem przy migracji danych obliczeń do pamięci podręcznej. Nie wyjaśniono pojęcia skalowalności obliczeń ze względu na liczbę procesorów i rozmiar problemu. W sumowaniu z poprawkami rozmiar problemu jest kluczowym parametrem do wyznaczenia dokładności rozwiązań. Nie wyjaśniono także czym jest efektywność energetyczna oraz jak działa Intel RAPL. Dodatkowy opis wzmocniłby rozdział pierwszy.

Poniżej umieszczono kilka zapytań z prośbą o ich doprecyzowanie:

- W rozdziale 5 podano stopień optymalizacji kompilatora -O3. W wcześniejszych badaniach nie podano tej opcji, czy była włączona?
- Jaki sposób weryfikacji poprawności implementacji został wykonany w badaniach? Jakie funkcje zostały zastosowane do pomiaru czasów?
- W standardzie C++ 20 przedstawiono nowe implementacje std::accumulate i std::reduce. Jaka jest przewaga proponowanych algorytmów sumowania z korekcją nad tymi funkcjami?
- W rozdziale 2.5 nie przytoczono funkcji z powszechnie znanego pakietu Wolphram Mathematica. Czy posiada ona ograniczenia podobne do funkcji z Matlab?

  https://reference.wolfram.com/language/ref/ToeplitzMatrix.html
- Badania z rozdziału 6.2 przeprowadzono na architekturze C czyli karcie Turing, dlaczego nie wybrano nowszej karty Ampere?
- Uzasadnić wybór problemu testowego 5.14.

Poniżej zawarto drobne usterki edytorskie

- Brak definicji SIMD, s.11.
- Na stronie 81 podano istotny schemat nazewnictwa algorytmów w przypisie. Tabela byłaby czytelniejszym formatem.
- W listingu 1.2 przedstawiono kod niedeterministyczny.
- Ilustrację 1.1 warto było wykonać w języku polskim.

Konkluzja

Recenzowana rozprawa stanowi oryginalne rozwiązanie jednoznacznie sformułowanego zagadnienia naukowego oraz istotny wkład w aktualnej tematyce badawczej w zakresie przetwarzania równoległego. Mgr Beata Dmitruk wykazała w tej rozprawie umiejętności prowadzenia badań naukowych, a także ich prawidłowej i wnikliwej interpretacji oraz publikacji wyników w uznanych przez fachowców czasopismach i konferencjach międzynarodowych. Wymienione powyżej uwagi krytyczne nie mają znaczącego wpływu na ogólną pozytywną ocenę pracy.

Konkludując uważam, że rozprawa doktorska mgr Beaty Dmitruk zdecydowanie spełnia wymagania stawiane w odpowiednich przepisach rozprawom doktorskim i wobec tego stawiam wniosek o jej dopuszczenie do dalszych, przewidzianych ustawą, etapów postępowania o nadanie stopnia doktora.

Patkowski Marek