Alexandru PÎRJAN

SOLUȚII INOVATOARE DESTINATE OPTIMIZĂRII PROCESĂRII DATELOR

PRO UNIVERSITARIA
BUCUREŞTI
2013®

Descrierea CIP a Bibliotecii Naţionale a României
PÎRJAN, ALEXANDRU
Soluţii inovatoare destinate optimizării procesării datelor / Alexandru Pîrjan. - Bucureşti : Pro Universitaria, 2013
Bibliogr.
ISBN 978-606-647-611-9

004(075.8)

PREFAȚĂ

Conținutul științific al acestei cărți reprezintă o parte din rezultatele obținute în urma cercetărilor desfășurate pe parcursul programului de studii doctorale din cadrul Academiei de Studii Economice din Bucureşti, sub îndrumarea științifică a domnului Prof. univ. dr. Ion Lungu [PiTe12]. Teza de doctorat „Soluții de optimizare a procesării datelor" a fost susținută pe data de 12 septembrie 2012, la Academia de Studii Economice din Bucureşti, iar ulterior, prin Ordin al ministrului educației, cercetării, tineretului și sportului, autorului i-a fost atribuit titlul de doctor în domeniul Informatică Economică.

Pentru cercetarea stadiului actual al cunoașterii în domeniul procesării datelor au fost utilizate o serie de publicații științifice consacrate: reviste cotate ISI, reviste de circulaţie internaţională, standarde, metodologii consacrate, cărţi publicate de edituri internaţionale recunoscute (Springer, Elsevier), cât şi volumele conferinţelor naționale şi internaţionale de profil.

Scopul cercetării a constat în identificarea de soluții pentru îmbunătăţirea performanței funcțiilor algoritmice de bază ce intervin frecvent în paşii de calcul ai unui număr mare de algoritmi sau aplicații de procesare a datelor, a căror optimizare implică resurse computaționale şi economice semnificative. Odată identificate funcțiile, a fost studiată posibilitatea implementării acestora în arhitectura de procesare paralelă CUDA (Compute Unified Device Arhitecture) pentru a beneficia de puterea computațională imensă oferită de aceasta. Au fost dezvoltate o serie de soluții de optimizare a performanței funcțiilor algoritmice de bază în CUDA, construind astfel o bibliotecă de funcții algoritmice de bază (BFAB) [PiTe12], ce facilitează dezvoltarea optimă a algoritmilor şi a aplicațiilor, implicit optimizarea procesării datelor în ansamblu. În consecință, procesarea datelor a fost optimizată prin dezvoltarea soluţiei BFAB ce conține funcțiile algoritmice, structurate ca blocuri funcționale de bază, ce intră în componența unui număr mare de algoritmi și aplicații de procesare a datelor. În această carte sunt prezentate doar o parte dintre funcțiile algoritmice de bază din BFAB și dintre aplicațiile lor, dezvoltate și prezentate în cadrul tezei de doctorat a autorului [PiTe12].

Motivația cercetării este dată de actualitatea şi importanța temei abordate, ce rezultă din modul cum a evoluat tehnologia modernă privind procesarea datelor. Una dintre problemele curente este absența unor metode performante și accesibile din punct de vedere economic care să permită implementarea și dezvoltarea de soluții de optimizare a procesării datelor bazate pe cele mai noi arhitecturi hardware şi software.

În ultimii ani, creșterea fără precedent a volumului de date ce trebuie procesat, cât și dezvoltarea aplicațiilor de procesare de date în diverse domenii (comercial, industrial, administrativ, etc) au determinat intensificarea interesului cercetătorilor cu privire la dezvoltarea soluțiilor de optimizare a procesării datelor. Un factor determinant ce a dus la creșterea impresionantă a volumului de date într-o varietate de domenii economice și științifice, îl reprezintă evoluţia continuă și rapidă a tehnicii de calcul, a dispozitivelor și mediilor de stocare cât mai ales a arhitecturilor de procesare paralelă. Pentru procesarea acestor volume imense de date sunt necesare resurse costistoare şi greu accesibile. În majoritatea situațiilor este esențial ca durata de timp necesară procesării datelor să fie cât mai mică.

Procesarea eficientă a datelor este extrem de utilă în numeroase domenii de activitate: în domeniul afacerilor, în predicția datelor financiare, în telecomunicații, în neuroştiinţe, în analiza datelor medicale, în explorarea resurselor naturale, în crearea de aplicații inovatoare în domenii precum recunoaşterea imaginilor, redarea grafică de înaltă definiție în timp real şi codificare. Astfel, pentru a planifica o strategie economică eficientă, se impune utilizarea unor tehnici adecvate pentru a procesa şi analiza datele disponibile. Cele mai multe companii utilizează volume mari de date în activitățile lor economice și în consecință necesită soluții eficiente de procesare a datelor. Pentru a supravieţui, pentru a se dezvolta și a obține profit în domeniul competitiv al afacerilor, companiile trebuie să gestioneze și să proceseze cât mai eficient posibil datele disponibile, obținând astfel strategii economice adaptate nevoilor și cerințelor pieței.

Eficientizarea componentei software și utilizarea optimă a resurselor hardware disponibile sunt cei mai importanți factori în dezvoltarea soluțiilor eficiente de procesare a datelor, în identificarea modelelor și în extragerea interdependenţelor dintre diferitele categorii de date. Aceste metode ajută la prezicerea viitoarelor modele ale tendinţelor pieței și oferă astfel companiilor ce le folosesc un avantaj competitiv. De exemplu: datele bursei de valori

sunt procesate în vederea elaborării de predicții; motorul de căutare Google utilizează tehnici de procesare a datelor pentru a selecta, din milioanele de pagini disponibile pe Internet, paginile Web cele mai relevante pentru căutarea cerută de client; în vederea stabilirii locației, datelele geospațiale necesită procesare și interpretare.

În privinţa proiectelor de cercetare ştiinţifică, este important ca analiza datelor experimentale să valideze ipotezele ştiinţifice pentru care sunt proiectate experimentele. Un exemplu de demers ştiinţific actual ce foloseşte procesarea datelor este acceleratorul de particule CERN de lângă Geneva, "Large Hadron Collider" (LHC), care generează zilnic o cantitate imensă de date, a căror analiză și procesare este realizată automat, prin utilizarea unor aplicații software şi a unor sisteme hardware performante.

Progresele înregistrate de tehnologia informaţiei în ultimii ani au determinat companiile să acumuleze volume imense de date din diverse domenii științifice sau economice. Costurile ce decurg din stocarea acestor volume de date au impus dezvoltarea corespunzătoare a tehnicii de calcul, a unor dispozitive și medii de stocare, a unor instrumente noi care să permită și să faciliteze atât accesul la date cât și procesarea acestora.

Deşi puterea de calcul evoluează odată cu dezvoltarea noilor tehnici și instrumente, volumul de date crește din ce în ce mai mult și se constată că timpul alocat procesării datelor are, de asemenea, tendinţa să crească. În vederea reducerii costurilor și a duratei de timp necesare procesării datelor, se impune dezvoltarea unor metode avansate și performante de procesare a datelor pentru a gestiona în mod corespunzător volumul imens de date, precum și a unor soluții de optimizare a procesării datelor, folosind atât funcții algoritmice de bază eficiente cât și arhitecturi hardware performante.

Pe baza problemelor identificate, în cadrul cercetării au fost atinse următoarele obiective principale:

- identificarea funcțiilor algoritmice cu cele mai bune performanțe, pentru a fi utilizate în dezvoltarea aplicațiilor privind procesarea datelor;
- cercetarea arhitecturilor hardware şi a mediilor de programare care să ofere soluţiile optime de implementare a funcțiilor algoritmice de bază;
- dezvoltarea de soluții pentru îmbunătăţirea performanței funcțiilor algoritmice de bază ce intervin frecvent în paşii de calcul ai unui număr mare de algoritmi sau aplicații de procesare a datelor. Aceste soluții au fost proiectate astfel încât să permită obţinerea

unor rezultate net superioare din punct de vedere al performanței și costurilor față de cele ale metodelor utilizate până în prezent, implementate în cadrul arhitecturilor clasice care folosesc unitățile de procesare centrală.

Pe baza concluziilor rezultate din experimentele efectuate, au fost identificate soluții de optimizare a funcțiilor algoritmice de bază prin adaptarea acestora în vederea executării în paralel a anumitor paşi de execuție, paralelizarea execuției fiind favorizată și recomandabilă întrucât rularea funcțiilor algoritmice are loc pe unități de procesare grafică, ce implementează arhitectura CUDA (Compute Unified Device Architecture).

În urma testărilor efectuate, a rulărilor experimentale pe procesoare grafice, s-au obținut rezultate deosebit de promițătoare, ceea ce conferă soluţiilor dezvoltate un înalt nivel de performanță și utilitate. Soluţiile au fost proiectate astfel încât să fie auto-ajustabile, să aloce resursele în funcţie de arhitectura de procesare folosită (Tesla, Fermi sau Kepler), ceea ce le conferă un grad ridicat de originalitate. Pe baza volumului mare de date de intrare generate, a unei game variate de arhitecturi, a numărului mare de iteraţii ale testelor efectuate, s-a obţinut o analiză detaliată a caracteristicilor funcțiilor algoritmice de bază folosite în optimizarea procesării datelor.

Testele au fost rulate pe următoarea configuraţie: Intel i7-2600K la 4,6 GHz cu 8 GB (2x4GB) de memorie cu frecvența 1333 MHz, DDR3, rulând în modul dual channel. Unitățile de procesare grafică utilizate au fost GeForce GTX 280, GTX 480 și GTX 680. Pentru a programa și accesa unitatea de procesare grafică s-a utilizat CUDA toolkit 4.1, versiunile de driver Nvidia 270.81 pentru GTX 280 şi pentru GTX 480 şi versiunea Nvidia 301.10 pentru GTX 680. Sistemul de operare utilizat în cadrul testelor experimentale a fost Windows 7 pe 64 biţi. În plus, toate procesele legate de interfaţa grafică a utilizatorului au fost dezactivate pentru a reduce traficul extern al unității de procesare grafică. În măsurătorile efectuate nu au fost incluși timpii necesari transferurilor de date între unitatea de procesare centrală (CPU) și unitatea de procesare grafică (GPU), întrucât funcțiile algoritmice de bază au fost proiectate pentru a fi folosite în componența unui număr mare de aplicații ce rulează pe procesoare grafice, astfel încât timpii de transfer variază în funcţie de complexitatea şi specificul aplicaţiei. Pentru a obţine o evaluare fiabilă a timpului de execuție necesar rulării funcțiilor algoritmice de bază pe unitățile de procesare grafică, au fost folosite evenimentele de înregistrare a ştampilelor de timp din cadrul interfeţei de programare a aplicaţiei (API) CUDA.

Experimentele au fost efectuate pe cele mai noi și reprezentative platforme hardware cu diferite arhitecturi și configuraţii. Rezultatele experimentale obținute sunt prezentate în această lucrare atât sub formă tabelară, cât şi sub formă grafică, împreună cu o serie de observaţii aferente acestor seturi de teste. Prin intermediul rulărilor experimentale au fost măsurate și analizate atât timpul mediu de rulare cât și lăţimile de bandă, folosind 10.000 de iteraţii pentru fiecare test al funcțiilor algoritmice.

În urma analizării rezultatelor cercetării prezentate în acest material, se obține o imagine de ansamblu a performanţelor oferite de funcțiile algoritmice de bază implementate pe unitățile de procesare centrală dar și a soluțiilor de optimizare a aplicațiilor dezvoltate în CUDA. Analizând această imagine, se ajunge la concluzia că o îmbunătăţire substanţială a timpilor de execuție aferenţi procesării datelor se obține prin rularea funcțiilor algoritmice de bază pe procesoare suplimentare, care conlucrează cu procesorul dedicat în operaţiile de procesare a datelor. Pentru atingerea acestui scop există diverse soluții, ca de exemplu:

- utilizarea mai multor procesoare centrale CPU, ceea ce implică costuri suplimentare considerabile cu privire la platforma hardware, necesitând modificări substanţiale, cum ar fi placa de bază multi-soclu, etc.
- folosirea unei unități de procesare grafică de scop general (GPGPU) ceea ce nu necesită modificări substanţiale ale arhitecturii hardware folosite (placa de bază, memorii, etc).

Din motive economice a fost preferată a doua variantă, iar rezultatele experimentale obținute au confirmat faptul că aceasta reprezintă o soluție performantă. Pentru a optimiza funcțiile algoritmice de bază folosite în procesarea datelor, s-a pornit de la principiul utilizării unităților de procesare grafică cu mai multe nuclee de procesare care să suporte ca mediu de dezvoltare arhitectura paralelă CUDA (Compute Unified Device Architecture). Această arhitectură permite utilizatorilor să folosească limbajul CUDA C, iar din punct de vedere operaţional, procesorul grafic devine un coprocesor al unităţii de procesare centrală. În vederea procesării în paralel a codului sursă, au fost identificate toate operaţiile ce pot fi paralelizate în cadrul arhitecturii CUDA. De asemenea, a fost acordată o atenţie deosebită modului de acces la diversele tipuri de memorie: privată, partajată sau globală, luând în considerare caracteristicile și timpii de acces specifici acestora. Tehnicile de proiectare utilizate pe parcursul cercetării, în

contextul rulării pe unități de procesare grafică, sunt aplicabile pe o gamă variată de procesoare ce incorporează mai multe nuclee de execuție.

Cartea este structurată în 6 capitole, cuprinzând de asemenea un capitol de concluzii, bibliografie şi anexe. În dezvoltarea cercetării s-a urmărit o abordare graduală ce corespunde următoarei structuri pe capitole:

După ce în **Prefață** este prezentata motivația, scopul și conținutul cercetării, în **Capitolul 1** este descrisă arhitectura CUDA (Compute Unified Device Architecture) de optimizare a procesării datelor, pornind de la un scurt istoric al evoluției de la unități de procesare centrală, la unități de procesare grafică. Sunt prezentate apoi câteva elemente referitoare la CUDA C, primul limbaj proiectat special de o companie producătoare de procesoare grafice pentru a facilita calculele cu scop general pe unitățile de procesare grafică. Sunt descrise o serie de domenii în care au fost implementate cu succes limbajul CUDA C şi arhitectura CUDA, precum și cele trei niveluri de abstractizare oferite de mediul de programare CUDA şi componentele de bază ale arhitecturii CUDA.

Sunt prezentate apoi o serie de concepte de bază ale modelului de programare CUDA: ierarhia firelor de execuție, ierarhia memoriei, echipamentul hardware gazdă și cel specializat. De un deosebit interes în cadrul acestui demers științific a fost comparația celor mai importante arhitecturi de procesoare grafice ce implementează CUDA. De asemenea sunt analizate principalele avantaje şi limitări ale utilizării arhitecturii CUDA în dezvoltarea aplicațiilor software, oportunitatea dezvoltării unei soluții de optimizare a procesării datelor în CUDA, iar apoi sunt identificate o serie de domenii cheie în care au fost implementate cu succes limbajul CUDA C şi arhitectura CUDA.

Aspectul esențial din cadrul acestui capitol se referă la identificarea celei mai importante facilităţi oferite de arhitectura CUDA: pentru a beneficia de puterea de procesare a unităților GPU, dezvoltatorii au posibilitatea să scrie aplicaţiile în diverse limbaje de programare, iar pentru dezvoltarea cu succes a aplicațiilor performante, programatorii trebuie să aiba la dispoziţie biblioteci specializate de funcții puternice dezvoltate în CUDA, ce pot fi apelate cu uşurinţă în cadrul aplicațiilor şi au ca efect creşterea substanţială a performanței acestora. Aceasta a fost și direcția de cercetare abordată în cadrul acestui demers științific, anume dezvoltarea unei soluții de optimizare a performanței funcțiilor algoritmice de bază, construind astfel o bibliotecă de funcții algoritmice de bază (BFAB) [PiTe12] în CUDA, apelabile în

programarea aplicațiilor de procesare de date. În finalul capitolului sunt prezentate oportunitatea, utilitatea şi importanța dezvoltării soluţiei BFAB în procesarea datelor, evidenţiind interfaţa de programare (API) a acesteia. În această carte sunt prezentate următoarele funcții algoritmice de bază, din cadrul soluţiei BFAB [PiTe12]: însumarea paralelă prefixată, reducerea paralelă şi compactarea fluxurilor de date.

În **Capitolul 2** sunt prezentate soluții de îmbunătăţire a performanței funcției algoritmice de însumare paralelă prefixată în CUDA. Este prezentată mai întâi proiectarea acestei funcții în cadrul warp-urilor, apoi în cadrul blocurilor de fire de execuție, iar în final la nivel global. A fost dezvoltată o variantă optimizată a funcției algoritmice de însumare prefixată în CUDA, au fost identificate şi aplicate o serie de soluții de îmbunătăţire a performanței acesteia. Analizând şi interpretând rezultatele experimentale obținute, se constată că acestea confirmă eficiența funcției algoritmice de însumare prefixată din BFAB, ce oferă rezultate optime în diverse situaţii și astfel, acesta este implementabilă într-o gamă largă de algoritmi și aplicații de procesare de date.

În **Capitolul 3** sunt prezentate soluții de îmbunătăţire a performanței funcției algoritmice de reducere paralelă în CUDA. Este prezentată funcția algoritmică de reducere paralelă în CUDA și variante de optimizare ale acesteia printr-o serie de soluții de îmbunătăţire a performanței. Eficiența funcției algoritmice de reducere paralelă din cadrul soluţiei BFAB a fost confirmată prin analiza și interpretarea rezultatelor experimentale obținute. Se constată că funcţia algoritmică de reducere paralelă oferă rezultate optime în diverse situaţii și, întrucât aceasta nu depinde de operatorul binar utilizat, poate fi implementată într-o gamă largă de algoritmi și aplicații.

În **Capitolul 4** sunt prezentate soluții de îmbunătăţire a performanței funcției algoritmice de compactare paralelă a fluxurilor de date în CUDA. Pentru început este descrisă proiectarea acesteia, apoi este dezvoltată o variantă optimizată a funcției algoritmice de compactare a fluxurilor de date în CUDA, utilizând o serie de soluții de îmbunătăţire a performanței acesteia. Rezultatele experimentale obținute confirmă eficiența funcției de compactare din BFAB, ce oferă rezultate optime în diverse situaţii și, astfel, funcția este implementabila într-o gamă largă de algoritmi și aplicații de procesare de date.

În cadrul **Capitolului 5** sunt analizate o serie de considerații economice privind oportunitatea soluțiilor de optimizare a procesării datelor

în CUDA. Pentru început, sunt evidențiate o serie de avantaje economice ale utilizării GPU legate de îmbunătăţirea considerabilă a eficienţei energetice în industria de calculatoare. Sunt prezentate apoi câteva dintre domeniile în care utilizarea sistemelor bazate pe GPU a adus o reducere semnificativă a timpului de lucru şi a cheltuielilor aferente consumului de energie. Această analiză reliefează importanţa din ce în ce mai mare a unităților de procesare grafică în întreaga industrie de calcul, ajungând astfel la concluzia că GPU reprezintă soluții de calcul eficiente din punct de vedere economic. În continuare este prezentat un studiu economic privind alegerea tehnologiei CUDA în implementarea soluţiei BFAB, punând în evidenţă, pentru început, o serie de aspecte ce trebuie luate în considerare atunci când se evaluează aspecte economice referitoare la utilizarea procesoarelor grafice ce implementează tehnologia CUDA. Pe baza acestor aspecte se remarcă oportunitatea soluţiilor de optimizare a procesării datelor în CUDA. În final este prezentată o analiză referitoare la avantajele economice ale soluţiei de optimizare a procesării datelor în CUDA, luând în considerare costul consumului de energie electrică și cel al echipamentului hardware. Rezultatele experimentale confirmă avantajele economice ale soluţiei propuse.

În **Capitolul 6** este propusă o soluție software de lansare, rulare şi testare experimentală a funcțiilor algoritmice de bază din BFAB, dezvoltată în Windows Presentation Foundation (WPF), folosind limbajul declarativ Extensible Application Markup Language (XAML) şi limbajul de programare C#. Folosind documentaţia oficială Microsoft, precum şi literatura de specialitate, pentru început este analizat şi prezentat modelul de programare WPF, principalele sale caracteristici, avantajele separării între interfaţa grafică şi funcţionalitatea aplicaţiei, tipurile de aplicaţii ce pot fi dezvoltate în WPF, tipurile de navigare suportate de modelul de programare WPF, arhitectura WPF, componentele majore ale acesteia, ierarhia claselor frecvent utilizate în WPF, posibilitatea de legare la sursele de date, o serie de caracteristici şi facilităţi ce au fost apoi implementate în dezvoltarea soluţiei software de lansare, rulare şi testare experimentală a funcţiilor algoritmice de bază din BFAB. Sunt prezentate apoi elemente legate de proiectarea acestei soluţii, de validarea datelor introduse, precum şi un exemplu de rulare a soluţiei dezvoltate în WPF. Soluţia dezvoltată în cadrul acestui capitol are un grad ridicat de originalitate şi este utilă în lansarea, testarea şi rularea funcţiilor algoritmice de bază prezentate şi analizate în capitolele anterioare.