Colonistii
Responsabil de temă: Eliana Tirsa
Data publicării: 22.10.2012
Termenul de predare: 4.11.2012, ora 23.55

1.Introducere

In nxn regiuni patratice identice (asezate sub forma de matrice) traiesc nxn colonisti. In intreg tinutul se comercializeaza doua tipuri de resurse, A si B. Fiecare colonist este specializat in producerea unui tip de resursa (A sau B), pe care o vinde anual (specializarea se poate modifica). Fiecare colonist are propriul pret pe unitatea de resursa, pe care il modifica anual, in functie de niste indicatori, insa exista limite legale pentru pretul minim si maxim(limite egale pentru cele doua resurse). Limitele stabilite de legea tinutului nu se modifica de la un an la altul. Resursele A si B sunt complementare, asa ca cei care produc A vor trebui sa cumpere anual B si viceversa. Fiecare colonist cumpara resursa complementara cu cel mai mic cost, cost care depinde de pretul de vanzare al resursei complementare si de distanta pana la regiunea unde e produsa. Fiecare colonist are un buget (pe care-l poate depasi) pe care-l ajusteaza anual, in functie de cheltuielile din anul precedent.

Fie Colij – colonistul care locuieste in regiunea cu coordonatele (i,j) din tinut, cu 0<=i<n, 0<=j<n

Rij – tipul de resursa pentru Colij intr-un anumit an; Rij poate fi 0 sau 1.

1-Rij – resursa complementara, pe care Colij trebuie s-o cumpere intr-un anumit an

Pij – pretul de vanzare al resursei Rij in anul curent

Pmin, Pmax – limita minima, respectiv maxima pentru pretul de vanzare al unei resurse (de orice tip)

Costij - costul cu care Colij cumpara resursa complementara in anul curent

Costij = mini1,j1 (Pi1,j1 + dist(Colij ,Coli1,j1)), cu Ri1,j1 = 1 - Rij ; e clar ca (i,j) si (i1, j1) nu se pot referi la acelasi colonist

CostResij - costul minim al unei resurse de tipul Rij, vazut din perspectiva lui Colij

CostResij = mini1,j1 (Pi1,j1 + dist(Colij ,Coli1,j1)), cu Ri1,j1 = Rij ; in acest caz, (i,j) si (i1, j1) se pot referi la acelasi colonist

dist(Coli,j ,Coli1,j1)) = |i - i1| + |j – j1| (Distanta Manhattan)

Bij – bugetul colonistului Colij in anul curent

Anual, au loc urmatoarele activitati:

1. Fiecare colonist cumpara cea mai ieftina resursa complementara (luand in calcul pretul resursei si lungimea drumului pana acolo)

2. Daca costul resursei complementare(Costij) depaseste bugetul alocat in acel an, pentru anul viitor va creste bugetul, dar si pretul resursei proprii, astfel:

Bij (an + 1) = Costij (an)

Pij (an + 1) = Pij (an) + Costij (an) - Bij (an)

3. Daca costul resursei complementare(Costij) este sub bugetul alocat in acel an, pentru anul viitor va scadea bugetul(respectand limita minima legala), dar si pretul resursei proprii, astfel:

Bij (an + 1) = Costij (an)

Pij (an + 1) = Pij (an) + (Costij (an) - Bij (an)) / 2

Aici “/” este impartire intreaga; rezultatul e partea intreaga a catului.

4. Daca costul resursei complementare(Costij) este egal cu bugetul alocat in acel an, pentru anul viitor se va modifica numai pretul resursei proprii, bugetul va ramane la fel:

Bij (an + 1) = Costij (an)

Pij (an + 1) = CostResij (an) + 1

5. Daca in urma actualizarilor de la punctul 3, pretul devine mai mic decat pretul minim admis de legea tinutului, pretul se actualizeaza astfel:

Pij (an + 1) = max (Pij (an + 1) , Pmin)

6. Daca in urma actualizarilor de la punctele 2 sau 4, pretul devine mai mare decat pretul maxim admis de legea tinutului, colonistul se re-specializeaza (produce resursa complementara), iar pretul si bugetul se actualizeaza astfel:

Rij (an + 1) = 1 - Rij (an) (pentru punctele 2-5, Rij (an + 1) = Rij (an))

Bij (an + 1) = Pmax

Pij (an + 1) = (Pmin + Pmax) / 2;

Aici "/" este impartire intreaga; rezultatul e partea intreaga a catului.

2. Cerintele temei

Se va studia evolutia pietei din tinut, pe un numar de ani. Mai exact, pentru fiecare an se vor calcula (in paralel pentru punctele 3 si 5 de mai jos):

- numarul de colonisti care produc resursa A (codificata cu 0)

- numarul de colonisti care produc resursa B (codificata cu 1)

- pretul maxim al resursei A

- pretul maxim al resursei B

Valorile cerute mai sus se vor tine in structuri uni sau multi-dimensionale (vectori sau matrici), cu lungimea egala cu numarul de ani. Abia dupa sfarsitul regiunii paralele se vor afisa.

In plus, se vor retine si afisa in fisierul de iesire, variantele finale (dupa ultimul an) al matricilor de tipuri de resurse, de preturi si de bugete.

Detaliere cerinte/punctaj

Va trebui sa aveti un fisier Readme(cu explicatii pentru surse – explicatii detaliate in cazul surselor optimizate, si cerintele de la punctul 4), precum si Makefile cu reguli de compilare si clean. Pentru variantele paralele, va trebui sa aveti un exemplu de script cu care ati rulat in cluster. Temele vor fi testate in clusterul facultatii. Lipsa makefile si script rulare (functionale), arhivarea cu alta extensie(in afara de zip), nerespectarea formatului de intrare/iesire (sau nume fisiere hardcodate in surse) ingreuneaza corectarea temei si duc la scaderea punctajului cu pana la 2 puncte (din 10).

Implementarea de baza (neoptimizata) seriala (secventiala) in C/C++ a acestei probleme. (2p)

Implementarea paralela utilizand OpenMP. Pentru obtinerea punctajului de corectitudine, trebuie sa calculati in paralel si cele 4 valori agregate mentionate la inceputul sectiunii Cerintele temei. Implementarile paralele trebuie sa fie corecte si din punct de vedere al concurentei/sincronizarii (daca e cazul) si al scope-ului variabilelor (private, shared, reduction...). (4p daca e corect + 1p daca scaleaza - performanta creste o data cu cresterea numarului de threaduri si varianta paralela cu 2 threaduri e mai rapida decat varianta seriala de la punctul 2)) Scalarea se puncteaza numai in cazul obtinerii punctajului total la corectitudine.

Calculati complexitatea teoretica a solutiilor implementate de voi (neoptimizata si optimizata – vezi punctul 5). Analizati performantele problemei implementate/paralelizate pentru fiecare caz: neoptimizat(punctele 2 si 3 – N=50, 500 iteratii si N=100 50 iteratii) si optimizat (punctul 5 – N=50, 5000 iteratii si N=100, 2000 iterati), variind numarul de threaduri (1, 2, 4, 8 in cluster, ruland cu qsub). Testati cu mai multe tipuri de scheduling si chunk. Pentru fiecare fisier de test cerut, va trebui sa scrieti in Readme numele testului(fisierul de input si numarul de iteratii), timpii pentru cea mai buna executie pentru fiecare numar de threaduri(cel putin pentru 1, 2, 4, 8) si speedup-ul obtinut – ca raport de timpi de executie (vezi definitie laborator 3). Mentionati de asemenea cu ce tip de scheduling si valoare chunk le-ati obtinut. (3p) Pentru a nu ocupa foarte mult clusterul si a da posibilitatea mai multor colegi sa-l foloseasca, va rog nu rulati in cluster scripturi a caror executie dureaza mai mult de 2 minute pe calculatorul personal. Pentru dimensiunile cu care vi se cere sa testati la acest punct, executiile ar trebui sa se incadreze in aceasta limita.

Bonusuri

Optimizati algoritmic varianta seriala astfel incat timpul de executie in cluster (cu qsub) sa nu depaseasca valorile de mai jos(fisierele de intrare de mai jos se regasesc intr-o arhiva zip in resurse_t1). Explicati in readme cum ati optimizat, si mentionati ce timpi ati obtinut (vom verifica). (2 p)

./optim 1000 in50_1.txt out50_1000.txt sub 3 secunde

./optim 1000 in100_1.txt out100_1000.txt sub 7 secunde

2. Paralelizati programul de la punctul 5.1 astfel incat sa obtineti o imbunatatire vizibila fata de varianta seriala optimizata(4.1). Explicatii si timpi / speed-up in readme. (1p) Aceasta varianta paralela se puncteaza numai daca scaleaza (si evident e si corecta)

Nu se vor puncta optimizarile de compilator (O2, O3, etc...). Implementarile se vor rula si pe teste private(cu N maxim 100); nu se accepta optimizari ne-algoritmice. (de genul if fisin=x, print 20)

Testele de la punctul 4 se vor face intotdeauna cu numarul de threaduri <= numarul de procesoare fizice disponibile(maxim 8 in coada ibm-quad.q). Inainte de a testa, fie algoritmul secvential, fie pe cel paralel, scoateti instructiunile de afisare nenecesare; in nici un caz sa nu aveti astfel de instructiuni intr-o regiune paralela.

3. Formatul datelor de intrare/ieşire. Rulare. Verificare

Exemplu de executie:
./serial T fisin fisout
./paralel T fisin fisout
./optims T fisin fisout
./optimp T fisin fisout
Unde T este numarul de iteratii(ani).
E de preferat sa respectati numele executabilelor de mai sus (numele surselor ramane la latitudinea voastra).

Veti avea cate o sursa C/C++ pentru fiecare din punctele (2, 3, 5.1, 5.2) abordate. Cu alte cuvinte, sursele paralele vor fi in fisiere diferite fata de cele seriale, iar sursele optimizate in fisiere diferite fata de cele neoptimizate. Daca implementati cerinta de la punctul 5, trebuie neaparat sa aveti si surse+analiza performantelor pentru punctele 2 si 3.

In cazul versiunilor paralele, numarul de threaduri nu e un parametru la rulare, dar se va specifica intr-o variabila de mediu, la fel ca scheduling-ul(eventual). Numarul de threaduri va fi identificat de variabila de mediu OMP_NUM_THREADS (nu setati in codul sursa numarul de threaduri). Tipul de scheduling poate fi setat in interiorul programului la static, dynamic, guided sau runtime. Daca il setati “runtime”, el va fi specificat (impreuna cu chunk-ul) de variabila de mediu OMP_SCHEDULE. Folosire:

export OMP_SCHEDULE="tip[,chunk_size]"

unde tip se refera la tipul de schedule, chunk_size la dimensiunea chunk-ului(care e optionala; nu puneti paranteze drepte)

Exemple de fisiere de intrare si iesire gasiti in directorul resurse_t1.

Formatul fisierului de intrare este:

pe prima linie, 3 numere intregi strict pozitive: Pmin, Pmax, n (vezi explicatii in prima parte a enuntului)

o linie goala

o matrice nxn, cu valori 1 si 0: n linii cu cate n elemente fiecare (matricea de tipuri de resurse)

o linie goala

o matrice nxn, cu valori intregi intre Pmin si Pmax: n linii cu cate n elemente fiecare (matricea de preturi)

o linie goala

o matrice nxn, cu valori intregi mai mari decat Pmin: n linii cu cate n elemente fiecare (matricea de bugete)

Formatul fisierului de iesire este:

pe primele T linii se afiseaza cele 4 valori agregate cerute pentru anii [1..T] (in ordinea: numarul de colonisti care produc resursa A (codificata cu 0), pretul maxim al resursei A, numarul de colonisti care produc resursa B (codificata cu 1), pretul maxim al resursei B); configuratia din fisierul de intrare corespunde anului 0

o matrice nxn, cu triplete de forma (Rij,Pij,Bij): n linii cu cate n triplete fiecare, reprezentand configuratia tipurilor de resurse, a preturilor si a bugetelor in anul T; configuratia din fisierul de intrare corespunde anului 0

Surse care dau erori la compilare sau rulare (sau se agata/blocheaza la rulare) vor fi punctate cu 0.

Corectitudinea implementarilor se va verifica, in aceasta ordine:

- comanda diff (vezi exemplul din mini-tutorialul de rulare in cluster) pe fisierele de iesire (intre outputul vostru si cel de referinta)

- inspectarea codului sursa

Daca verificarea cu diff intoarce diferente, punctajul obtinut pe implementarile seriale va fi neglijabil (0 in cazul punctului 5). In cazul implementarilor paralele, depunctarea depinde de gravitatea erorilor; atentie ca e posibil ca programul paralel sa treaca de verificarea cu diff si totusi sa nu fie corect.

4. Testare pe cluster cu qsub. Masurarea timpului de rulare

In directorul resurse_t1 aveti un document text cu un mic tutorial despre cum puteti rula in clusterul facultatii.

Indiferent daca testati local sau pe cluster, cronometrarea se va face cu comanda (din consola) time:

time ./executabil [parametri]

Din outputul comenzii time ne intereseaza valoarea din dreptul cuvantului "real":
real 0m1.287s
user 0m1.270s
sys 0m0.010s