========== README ==========

Detalii implementare tema 1 APD

Musat Andreea Alexandra, 333CA

Varianta seriala

In cazul in care programul nu este rulat cu cele 3 argumente in linia de comanda (input_file.txt no_of_iterations output_file.txt), atunci se afiseaza un mesaj de eroare si executia se termina. In cazul in care argumentele au fost oferite corect, programul va continua rularea, astfel: se incearca deschiderea fisierului de intrare; in cazul in care acesta se deschide cu succes, se incepe citirea datelor de intrare. Intrucat matricea care contine starea automatului necesita bordare, prima si ultima linie (si coloana) vor fi initial goale. Asadar, citirea se va face incepand cu indecsii 1 si 1. Cand este citit un spatiu, acesta este ignorat. Daca un caracter newline este citit, atunci indicele liniei se incrementeaza, iar indicele coloanei este setat la 1. Cand un caracter valid ('X' sau '.') este citit, acesta este stocat pe pozitia curenta specificata de cei doi indecsi. Dupa inchiderea fisierului de intrare se incepe procesarea efectiva intr-un loop cu numarul de pasi egal cu numarul de iteratii. La fiecare interatie se face mai intai bordarea matricei (se copiaza pe prima linie elementele de pe penultima, pe ultima linie elementele de pe a doua etc). Apoi, pentru fiecare element din matrice care nu este situat la extremitati, se calculeaza numarul de vecini si se aplica regulile, stabilind ce stare va avea celula la iteratia urmatoare. In final, se copiaza elementele din matricea care reprezinta starea automatului la iteratia viitoare in matricea care reprezinta starea curenta si se trece la iteratia urmatoare.

Varianta paralela

In plus fata de varianta seriala, la varianta paralela se foloseste o structura de work-sharing (#pragma omp parallel for), partajandu-se iteratiile for-ului peste toate thread-urile (numarul lor fiind setat prin export=OMP_NUM_THREADS=nr_thread-uri). Desi in mod teoretic ar fi mai eficient sa existe un singur bloc paralel in care sa existe bariere si sectiuni executate de un singur thread (pentru a evita crearea multipla a thread-urilor), in practica am obtinut timp mai mici pentru versiunea in care fiecare for a fost paralelizat separat. In rezultate, se observa ca in cazul paralelizarii pe un singur thread, timpul de executie este mai mare decat cel pentru versiunea seriala, incat se adauga timpul crearii thread-ului. Pentru 2, 4 sau 8 thread-uri, insa, scaderea este evidenta (atunci cand si testele au dimensiuni mai mari; in cazul testelor cu dimensiuni mici, timpul “economisit” prin paralelizare este compensat de timpul folosit pentru crearea thread-urilor, iar uneori este posibil ca timpul total de executie sa fie chiar putin mai mare decat cel serial – de exemplu pentru in4.txt cu 100 de iteratii se obtin timpii: 0.028 pentru varianta seriala, iar pentru paralel: 0.028, 0.030, 0.022, 0.024 pentru 1, 2, 4, respectiv 8 thread-uri, diferenta fiind insa insignifianta.). Pentru un test cu dimensiuni de intrare 2000 x 2000 si 100 de iteratii, programul ruleaza de 1.4 ori mai repede pe 2 thread-uri, de ~4 ori mai repede pe 4 thread-uri si ~7.8 ori mai repede pe 8 thread-uri.