Arhitektura računara
Projektni zadatak – grupa 2 – mikroarhitektura I paralelizam

Zadatak 2.2 – Optimizacija algoritma

(15) U proizvoljnom programskom jeziku realizovati paralelizibilan algoritam koji vrši netrivijalnu obradu podataka. Algoritam odabrati na kursu predmeta.

Program treba da, kao argumente komandne linije, prihvata putanju do ulaznog fajla, putanju do izlaznog fajla, kao i vrijednosti parametara algoritma. Obezbijediti smislene podrazumijevane vrijednosti za sve argumente komandne linije. Analizirati, uporediti, dokumentovati i grafički predstaviti mogućnosti ubrzavanja datog algoritma korištenjem kompajlerskih optimizacija i bar 2 navedena pristupa:

1. (a) SIMD programiranje ili (b) optimizacije za keš memoriju.
    - Za (a) je dozvoljeno koristiti SIMD optimizacije urađene u zadatku 1.2 (asemblerski program za obradu podataka), ako je odabran isti algoritam i za ovaj zadatak.
    - Za (b) je potrebno izvršiti mjerenja (npr. upotrebom *cachegrind* alata) i pokazati da keš optimizacija stvarno doprinosi keš performansama (procentu keš pogodaka).
    - Za (b) je dozvoljeno da se kao inicijalni algoritam iskoristi varijanta algoritma sa lošijim keš performansama (bez dodavanja nepotrebnog koda).
2. Paralelizacija na višejezgarnom procesoru (npr. OpenMP ili sopstveno rješenje).
3. Distribuirano procesiranje (npr. OpenMPI ili sopstveno rješenje).
4. GPGPU programiranje (npr. OpenCL ili CUDA).

(10) Kombinovati dva navedena pristupa u cilju još većeg ubrzanja i dokumentovati rezultate. Pri tome je potrebno zabilježiti i grafički predstaviti rezultate prije i poslije primjene optimizacija i paralelizacije.

Obezbijediti nekoliko primjera ili jediničnih testova kojima se demonstriraju funkcionalnosti iz stavki u specifikaciji projektnog zadatka. Na pravilan način pokazati da će optimizovane varijante algoritma proizvoditi tačan rezultat.

Pri mjerenju vremena izvršavanja:

- Izvršiti mjerenja za veličine ulaza N različitog reda – npr. $N \in \{10^3, 10^4, 10^5, 10^6, 10^7, \dots\}$.
- Izvršiti zagrijavanje prije mjerenja, tj. prije mjerenja izvršiti algoritam više puta bez mjerenja.
- Ako vrijeme izračunavanja algoritma zavisi od sadržaja ulaznih podataka, izvršiti mjerenja za različite slučajeve (npr. nasumični ulazni podaci i granični slučajevi).
- Za svaki ulaz izvršiti mjerenje veći broj puta, a kao konačan rezultat mjerenja dokumentovati prosječnu vrijednost i varijansu.

Pridržavati se:

- principa objektno-orijentisanog programiranja,
- SOLID principa,
- principa pisanja čistog koda i
- konvencija za korišteni programski jezik.