

Paralelna implementacija Merge sort algoritma

Milin Ivan E1-79/2023

Sekvencijalna implementacija algoritma u C programskom jeziku preuzeta je sa interneta, kod se originalno sastojao samo od funkcija *merge*, *mergeSort* kao i glavnog, *int main* dela programa.

U taj kod je dodata funkcija *printArray*, *isSorted*, *saveArrayToFile*, a u *main* delu samog programa ubačeno je i merenje vremena koliko je sekundi potrebno da se izvrši funkcija *mergeSort*.

- Funkcija *printArray* generisani niz ispisuje u terminal ukoliko se promenljiva *print* postavi na 1.
- Funkcija *isSorted* proverava da li je niz sortirani nakon što se pozove funkcija *mergeSort*
- Funkcija *saveArrayToFile* generiše fajl i u njega upisuje nizove pre i nakon sortiranja.

Paralelna implementacija koda ne odstupa puno u odnosu na sekvencijalnu, kod je takođe implementiran u C programskom jeziku uz korišćenje biblioteke **OpenMP** za paralelno programiranje.

Unutar funkcije *mergeSort* dodate su dve *pragma omp task* direktive za oba poziva rekurzije. Na ovaj način se omogućava kreiranje zadataka/tast-ova koji će biti izvršavani na nitima procesora.

Dodavanjem *if (n > TASK_SIZE)* sprečava se kreiranje malih taskova koji mogu usporiti čitav program, u ovom slučaju *TASK_SIZE* je postavljen na *100000*, taj broj se eksperimentalnim testiranjem ispostavio kao najefikasniji.

Direktiva *#pragma omp taskwait* služi za sinhronizaciju, čeka da se prethodna dva taska završe kako bi mogla funkcija *merge* da se pozove.

U glavnom *main* delu programa na početku se prvo podešava koliko niti će biti uposleno što je postignuto pomoću *omp_set_num_threads(numThreads)*, a zatim se unutar *#pragma omp parallel* poziva funkcija *mergeSort* i pomoću *#pragma omp single* vodi se računa da samo jedna nit može da pozove tu funkciju a zatim unutar funkcije *mergeSort* ostale niti preuzimaju deo posla na sebe.

Vreme paralelnog izvršavanja *mergeSort* funkcije meri se iznad i ispod direktive *#pragma omp parallel*.

U tabeli prikazano je vreme izvršavanja kao i ubrzanje koje je postignuto paralelizovanjem merge sort algoritma pomoću određenog broja niti, niz nad kojim je testiran program sastoji se od **10 miliona** nasumičnih brojeva.

| IMPLEMENTACIJA PROGRAMA | TRAJANJE PROGRAMA [S] | UBRZANJE PROGRAMA |
|-------------------------------|-----------------------|-------------------|
| Sekvencijalni program | 1.536594 | 1 |
| Paralelni program (1 nit) | 1.964474 | 0.782491 |
| Paralelni program (2 niti) | 0.984528 | 1.560741 |
| Paralelni program (3 niti) | 0.972822 | 1.579522 |
| Paralelni program (4 niti) | 0.750643 | 2.047037 |
| Paralelni program (5 niti) | 0.654915 | 2.346249 |
| Paralelni program (6 niti) | 0.616985 | 2.490488 |
| Paralelni program (7 niti) | 0.587905 | 2.613677 |
| Paralelni program (8niti) | 0.545633 | 2.816167 |

I Vreme izvršavanja i ubrzanje