

1. Dat je naredni zadatak

Za svaki element iz niza brojeva potrebno je odrediti naredne karakteristike:

- da li je broj prost?
- da li je broj složen?
- da li je jednak zbiru kvadrata svojih cifara?
- da li je deljiv sa sumom svojih cifara?
- da li ima paran broj delilaca?
- da li ima neparan broj različitih prostih faktora?
- da li je jednak zbiru kubova svojih cifara?

Navesti četiri različite mogućnosti paralelizacije rešenja ovog zadatka i objasniti za svako rešenje koji dodatni uslovi treba da budu ispunjeni da bi izabrali baš to rešenje (tj kada predloženo rešenje daje najbolje rezultate).

(a) **Paralelizacija zadataka:**

Jedna nit nad svakim elementom niza provjerava da li je broj prost, druga nad svakim elementom niza provjerava da li je broj slozen i tako redom. Svaka nit ima odredjeni zadatak koji primjenjuje nad svakim elementom datog niza. Nedostatak ovog pristupa je sto ne skalira dobro ukoliko imamo vise procesora a takodje mozemo uociti zavisnosti izmedju zadataka koje ovdje nismo iskoristili. Dakle, navedeni postupak daje najbolje rezultate ukoliko radimo sa manjim brojem procesora i nezavisnim zadacima.

(b) **Paralelizacija podataka, prvi način:**

Svaka nit obradjuje jedan dio niza, npr. ako niz podijelimo na 3 jednaka dijela, svaka nit je zaduzena za po jedan dio. Niti koriste zavisnosti medju pojedinim zadacima (niti medjusobno nezavisne). Npr. nakon sto je neka nit odredila cifre datog broja, koristeći dobijene medjurezultate lako provjerva da li je broj djeljiv sumom svojih cifara, jednak sumi kvadrata tj. sumi kubova svojih cifara. Slicno, moze se uociti povezanost izmedju zadataka za provjeru da li je broj prost odnosno slozen, da li ima paran broj djelilaca ili neparan broj prostih faktora. Ovakav postupak je pozeljno koristiti ukoliko imamo veliki broj procesora i obradjujemo veci skup podataka, na taj nacin svaki procesor dobija dio podataka koji obradjuje i postize se bolja efikasnost. Takodje u ovom slucaju bitna je i uniformnost podataka, tj. pretpostavljamo da ce svaka nit imati dovoljno posla sa svojim dijelom niza.

(c) **Paralelizacija podataka, drugi način:**

Ukoliko nemamo uniformnost podataka, npr. ukoliko su neki djelovi niza jednocifreni brojevi za njih se lako vrsi svako od izracunavanja i u skladu sa tim svaka nit obradjuje jedan po jedan element niza. Svaka nit uzima prvi sledeci element u nizu koji je slobodan tj. koji nije obradjen kako bi sve niti bile podjednako zaposlene. U ovom slucaju niti moraju medjusobno da komuniciraju da bi se znalo koji element nije obradjen.

(d) **Paralelizacija podataka, treći način:**

Modifikacija prethodnog pristupa. Jedna nit uzme npr. prvih slobodnih 10 elemenata, druga narednih slobodnih 10 itd. Ovaj nacin je brzi nego prethodni jer niti ne moraju da komuniciraju za svaki pojedinačni element. Predlog je koristiti ovaj nacin ukoliko nemamo uniformnost podataka.

2. Potrebno je napisati aplikaciju koja vrši različita matematička izračunavanja. Najzahtevniji deo aplikacije je izračunavanje koje se vrši nad elementima veoma dugačkog niza velikih brojeva (svaki broj sadrži više od 100 cifara). Svaki element niza potrebno je pomnožiti redom sa brojevima 2, 4, 6, 8 i 12 i sačuvati sve dobijene proizvode. Kako je najbolje paralelizovati ovo izračunavanje i zašto baš tako?

Kako je u pitanju niz sa velikim brojem elemenata najbolje je koristiti paralelizaciju podataka. Svaka nit obradjuje jedan dio niza. Takodje, mozemo primijetiti da su zadaci medjusobno zavisni tj. nakon sto smo neki element x pomnozili brojem 2, dobijeni rezultat mozemo iskoristiti i pomnoziti sa 2 da bismo dobili $4*x$ odnosno sa 3 da bismo dobili $6*x$ i tako redom koristiti rezultate prethodnih izracunavanja i dodatno ubrzati proces.