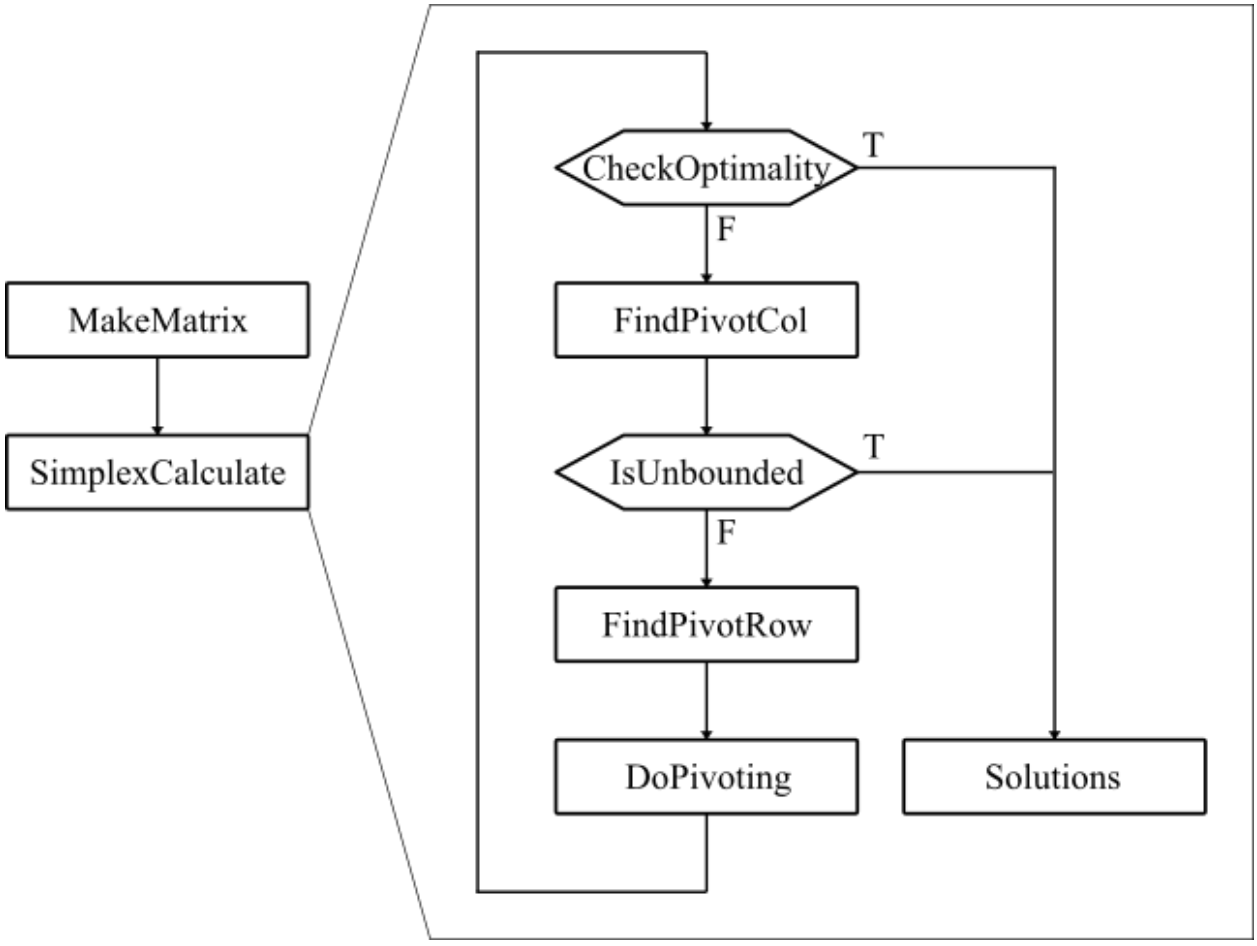
Paralelizacija simpleks algoritma

Opis algoritma

Simpleks algoritam je popularan algoritam koji se koristi u oblasti linearnog programiranja. Dijagram simpleks algoritma je prikazan na slici 1. U *makeMatrix* funkciji se učitava ulazna matrica i modifikuje da bi odgovarala algoritmu. U *simplexCalculate* funkciji se vrši glavni program.

Analizom algoritma se može videti da za dovoljno velike ulazne matrice, za koje ima smisla uvoditi paralelizaciji, najveći deo vremena se potroši na *doPivoting* funkciju, specifično na poslednju petlju u ovom delu. Za ulaznu matricu sa 1200 promenljivih i 1200 ograničenja ova petlja čini preko 90% ukupnog vremena i, prema tome, je glavni kandidat za paralelizaciju.



Slika 1. Dijagram simpleks algoritma

Paralelizacija

Na listingu 1 je prikazan kod petlje koju je paralelizovana.

```
#pragma omp parallel for num_threads(tc)
    for(int j=0;j<ROWSIZE;j++)
    {
         if(j==pivotRow)
         {
            for(int i=0;i<COLSIZE;i++)
               {
                    wv[j][i]=newRow[i];
               }
         }
         else
         {
            for(int i=0;i<COLSIZE;i++)
               {
                    wv[j][i]=wv[j][i]-newRow[i]*pivotColVal[j];
               }
        }
    }
```

Listing 1. Petlja koja se paralelizuje

Korišćen je model sa deljenom memorijom. Zaključuje se da petlja sa iteratorom j pripada klasi DOALL paralelizma jer ne postoji zavisnost između njenih iteracija. Svaka iteracija se može dodeliti posebnoj niti. Korišćen je OpenMp API.

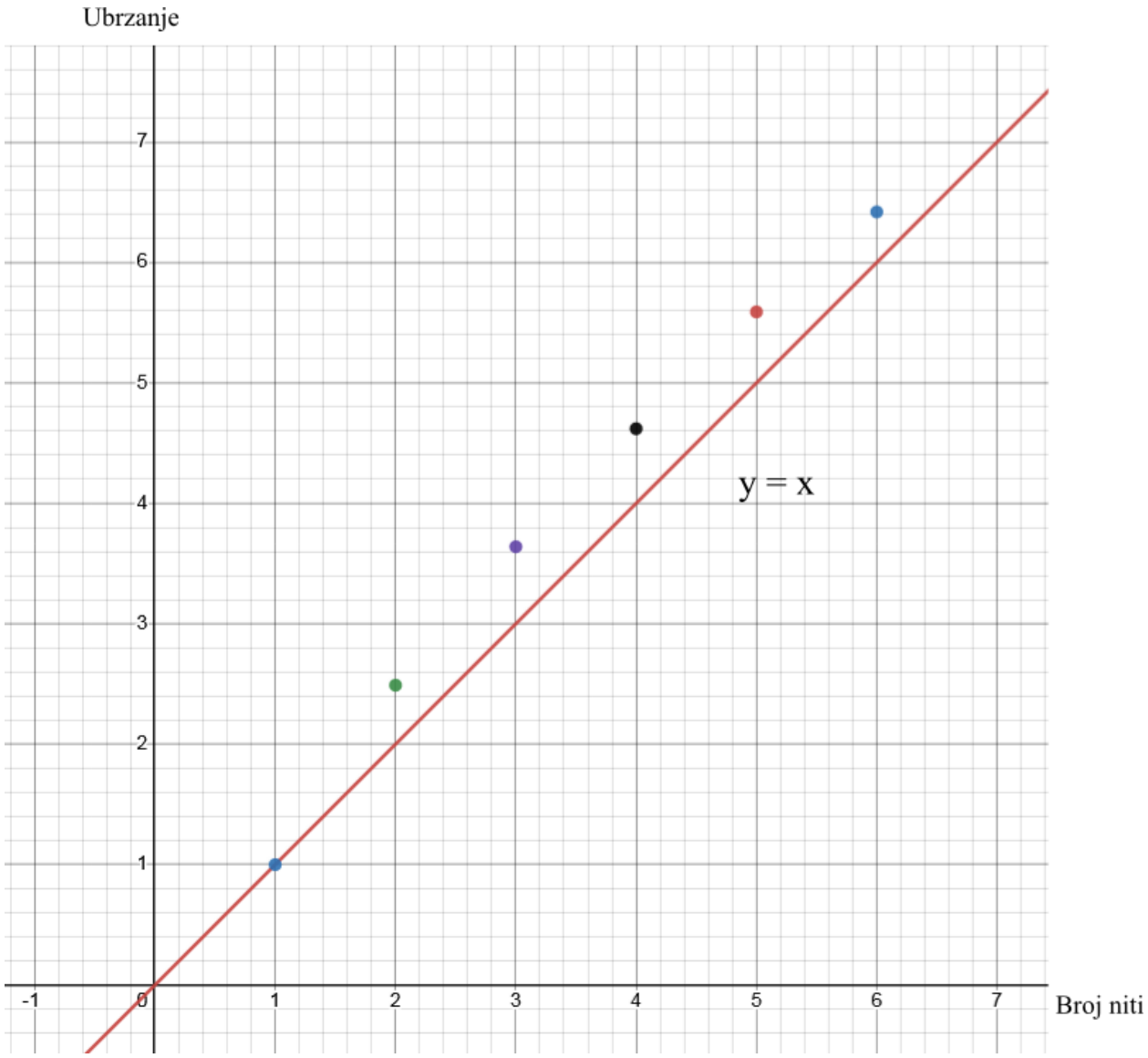
Rezultati

Sekvencijalan algoritam je napisan u *main.cpp* (aplikacija *mainomp*) a paralelizovani u *projekat.cpp* (aplikacija *projekat*). Ovi kodovi su kompajlirani sa O2 optimizacijom. Mereno je vreme izvršavanja čitave aplikacije. Rezultati sa vremenima i ubrzanjem u odnosu na broj korišćenih niti se nalaze u tabeli 1.

Način izvršavanja/broj niti	Sekvencijalno	2	3	4	5	6
Vreme izvršavanja [s]	99.03	39.67	27.18	21.45	17.72	15.42
Ubrzanje	1	2.49	3.64	4.62	5.59	6.42

Tabela 1. Poređenje sekvencijalnog izvršavanja i izvršavanja sa više niti

Dobijeno ubrzanje u odnosu na broj korišćenih niti je predstavljeno na grafiku na slici 2. Vidi se da je dobijeno superlinearno ubrzanje, što je posledica postojanja keša.



Slika2. Grafik dobijenog ubrzanja u odnosu na broj korišćenih niti i poređenje sa pravom y = x