Druga domaća zadaća

Igra Connect4

1. Implementacija

Upute: U ovom odjeljku potrebno je opisati ključne dijelove funkcionalnosti koristeći isječke programa i snimke zaslona. Obavezno uključite sljedeće elemente s odgovarajućim komentarima:

- *Isječak programa koji prikazuje pripremu poslova na glavnom (master) procesu.*

```
while True:
    print("Dubina: " + str(iDepth))
    dBest = -1.0
    iBestCol = -1.0
    for iCol in range(B.columns):
        if(B.moveLegal(iCol)):
            if iBestCol == -1:
                iBestCol=iCol
            if(dubinaDijeljenja==3):
                movesList[iCol]=1.0
            B.Move(iCol,2)
            if(size>1):
                if(dubinaDijeljenja==1):
                    movesList[iCol]=1.0

                    dResult ={'col':iCol,'Current':B.copy(), 'LastMover':2,'iLastCol':iCol,'iDepth':iDepth-1}
                    taskList.append(dResult)
                else:

                    EvaluateParalel(B.copy(), 2, iCol, iDepth-1)
                B.unodMove(iCol)
            else:
                dResult = Evaluate(B.copy(), 2, iCol, iDepth-1)
                B.unodMove(iCol)
                if dResult>dBest or (dBest == dResult and random.randint(0,2)==0):
                    dBest= dResult
                    iBestCol=iCol
                print("Stupac:"+ str(iCol))
                print("vrijednost: " + str(dBest))
```

Program kreira zadatke po uzoru na zadani primjer u C++-u, ako je broj procesora Jednak 1 algoritam se izvršava slijedno kao u primjeru, ali ako je zadani broj procesora veći od 1 tada se zadatci stvaraju ovisno o zadanoj dubini dijeljenja, ako je zadan dubina 1 onda se zadatci kreiraju još u petlji a ako je veći od toga kreiraju se u funkciji evaluateParalel. Svi kreirani zadatci se pohranjuju u listu „taskList“ iz koje glavni proces dijeli zadatke sporednim procesima koji obavljaju funkciju evaluate nad zadanim podatcima. U while petlji se kreira maksimalno 7 zadataka po iteraciji.

```
def EvaluateParalel(Current,LastMover,iLastCol,iDepth):
    dTotal=0.0
    bAllLose = True
    bAllWin = True
    iMoves=1
    if(Current.gotovo(iLastCol)):
        if(LastMover==2):
            return 1.0

        else:
            return -1.0
    if(iDepth==0):
        return 0.0
    iDepth-=1
    if(LastMover==2):
        NewMover = 1
    else:
        NewMover = 2
    for iCol in range(Current.columns):
        if(Current.moveLegal(iCol)):

            Current.Move(iCol,NewMover)
            if(dubinaDijeljenja==2):

                iMoves+=1
                dResult ={'col':iLastCol,'Current':Current.copy(), 'LastMover':NewMover,'iLastCol':iCol,'iDepth':iDepth}
                taskList.append(dResult)
            elif(dubinaDijeljenja==3):


                for jCol in range(Current.columns):
                    if(Current.moveLegal(jCol)):

                        iMoves+=1
                        if(NewMover==2):
                            NewerMover = 1
                        else:
                            NewerMover = 2
                        Current.Move(jCol,NewerMover)
                        dResult ={'col':iLastCol,'Current':Current.copy(), 'LastMover':NewerMover,'iLastCol':jCol,'iDepth':iDepth-1}
                        taskList.append(dResult)
                        Current.unodMove(jCol)
            Current.unodMove(iCol)
    movesList[iLastCol]=iMoves
```

Ovdje je funkcija evaluateParalel u kojoj se kreira maksimalno 49 ili 343 zadatka koji se pohranjuju u listu „taskList“ te se broji broj poteza koji su napravljeni da se izračuna prosjek slično kao što je to bilo u originalnom algoritmu iz primjera.

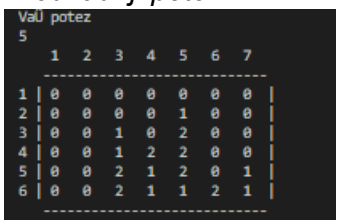
- *Isječke programa koji pokazuju kako se zadaci prenose s glavnog (master) procesa na radničke (worker) procese.*

```
if(size>1):
    for i in range(0,len(taskList),size-1):
        for a in range(1,size):

            if(i+a-1)<len(taskList):
                #print(a)
                print(i+a-1)
                comm.send(taskList[i+a-1], dest=a)
        for a in range(1,size):

            if(i+a-1)<len(taskList):
                data=comm.recv(source=a)
                resultList[data["col"]]+=data["rez"]
```

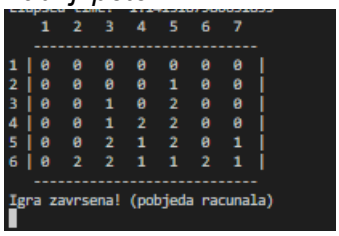
Dakle glavni proces iterira po listi task list i dijeli svakom sprednom procesu po zadatak i onda čeka odgovore od svakog te im opet dijeli zadatke dok ne prođe cijelu listu.

- *Snimku zaslona koja prikazuje posljednja dva koraka igre u kojoj računalo pobjeđuje.*

Predzadnji potez



Zadnji potez



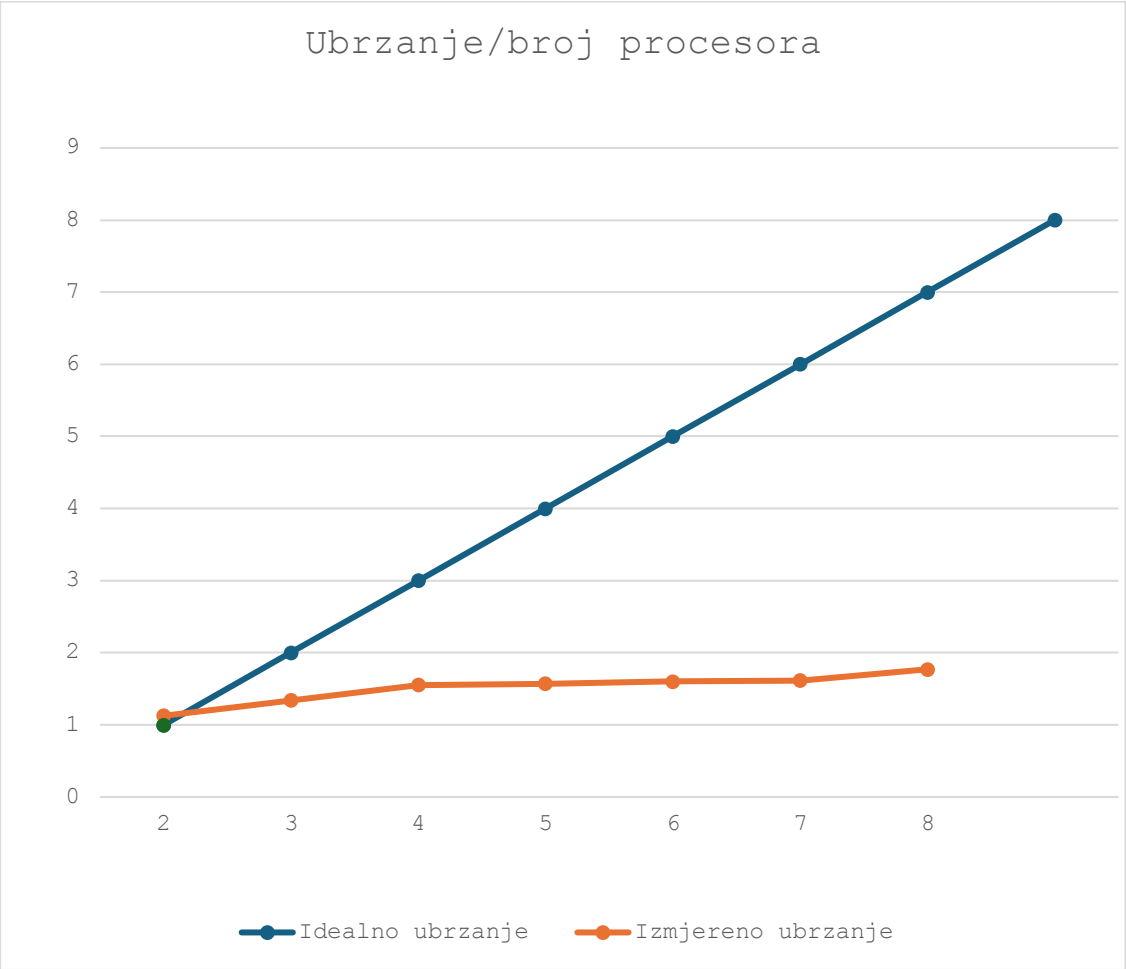
2. Kvantitativna analiza

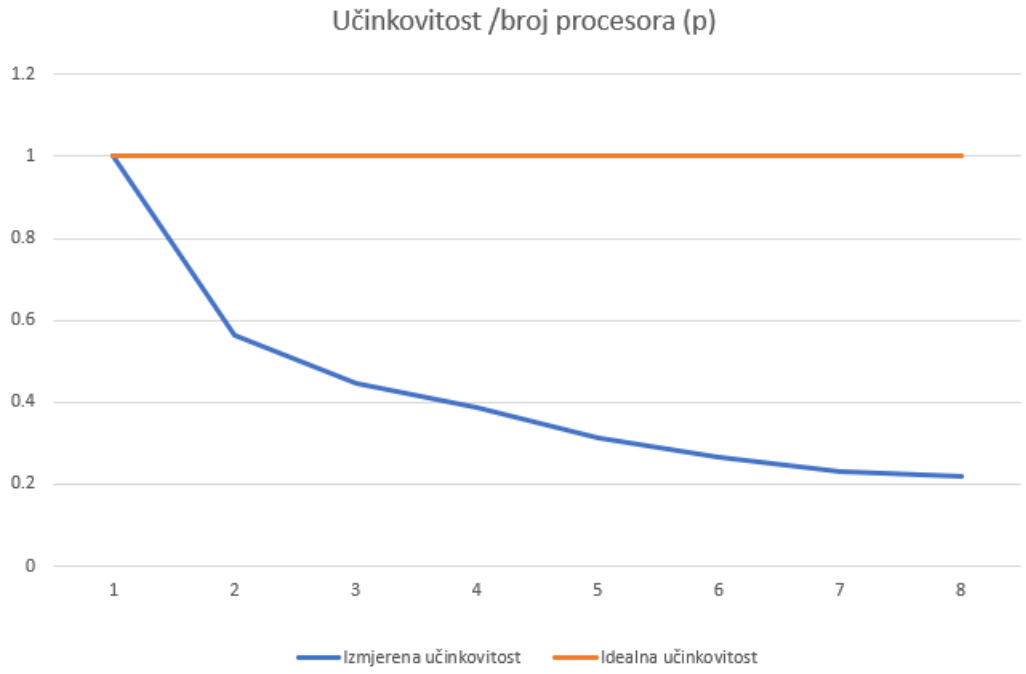
*Upute: U ovom dijelu potrebno je priložiti **tablice s rezultatima mjerenja te grafove ubrzanja i učinkovitosti** za tri različita scenarija: kada paralelni algoritam ima 7, 49 i 343 zadatka (uz aglomeraciju na dubini 1, 2 i 3). Mjerenja treba provesti tako da je najmanje mjereno trajanje (za 8 procesora) reda veličine barem **nekoliko sekundi** (definirajte potrebnu dubinu pretraživanja). Uz grafove, dodajte kratki komentar koji opisuje kako broj zadataka utječe na ubrzanje i učinkovitost (uzevši u obzir utjecaj zrnatosti zadataka, komunikacijskog overhead-a, te udjela programa koji se ne može paralelizirati).*

Napomena: Računalo na kojem su vršena mjerenja ima dvije fizičke jezgre, nisamo bio u situaciji da koristim drugo računalo.

a) 7 zadataka

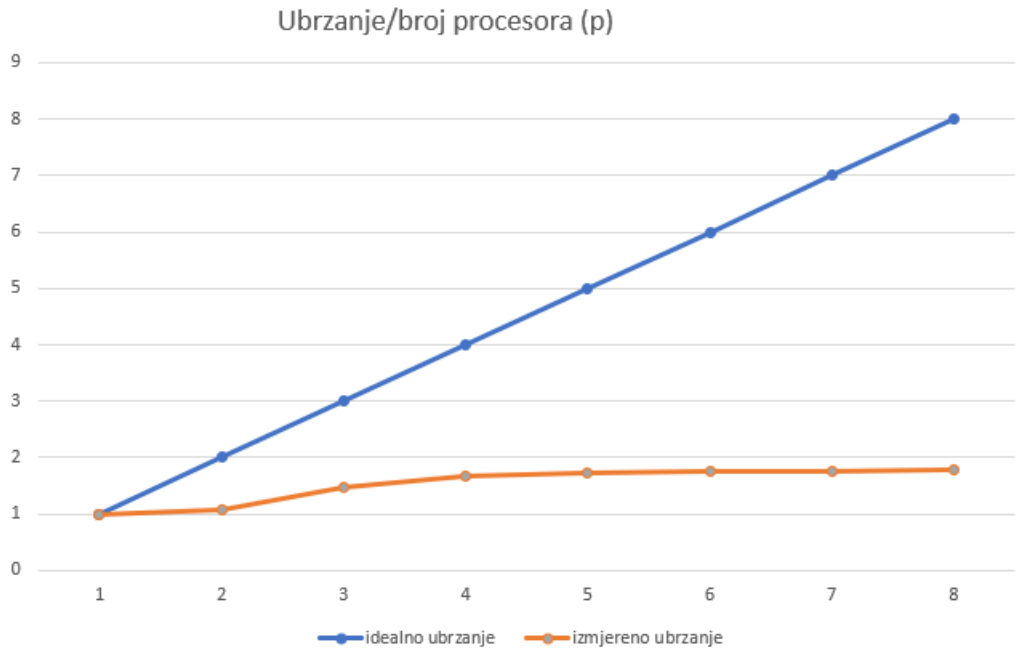
BROJ PROCESORA(p)	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Trajanje (s)	*25.41*	*22.56*	*18.97*	*16.34*	*16.17*	*15.87*	*15.71*	*14.35*
Ubrzanje	1	1.126	1.339	1.555	1.571	1.601	1.617	1.771
Učinkovitost	1	0.563	0.446	0.389	0.314	0.267	0.231	0.221

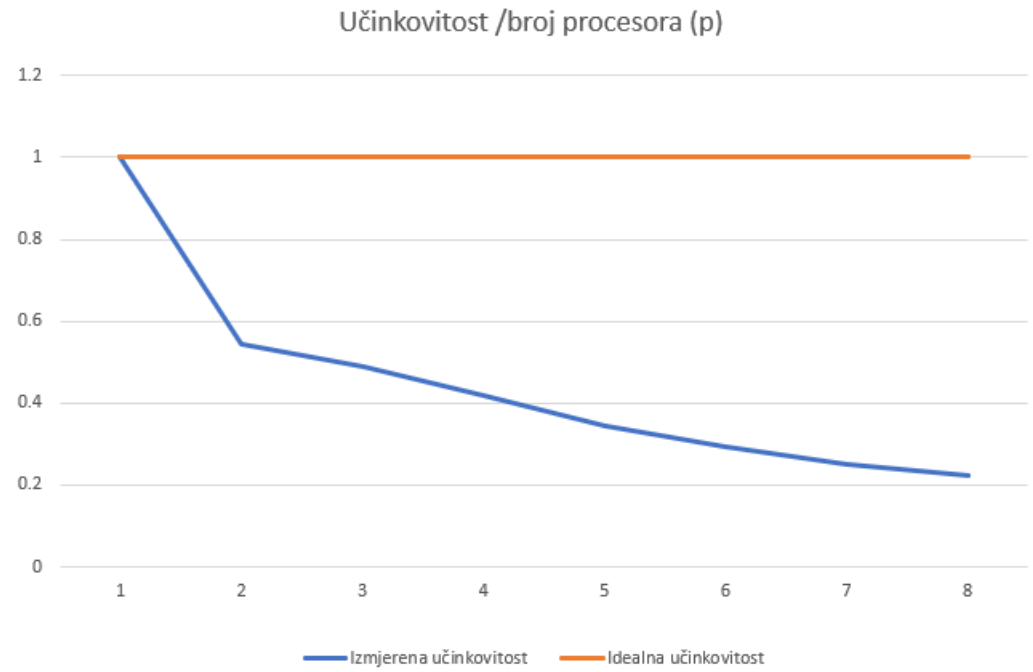




b) 49 zadataka

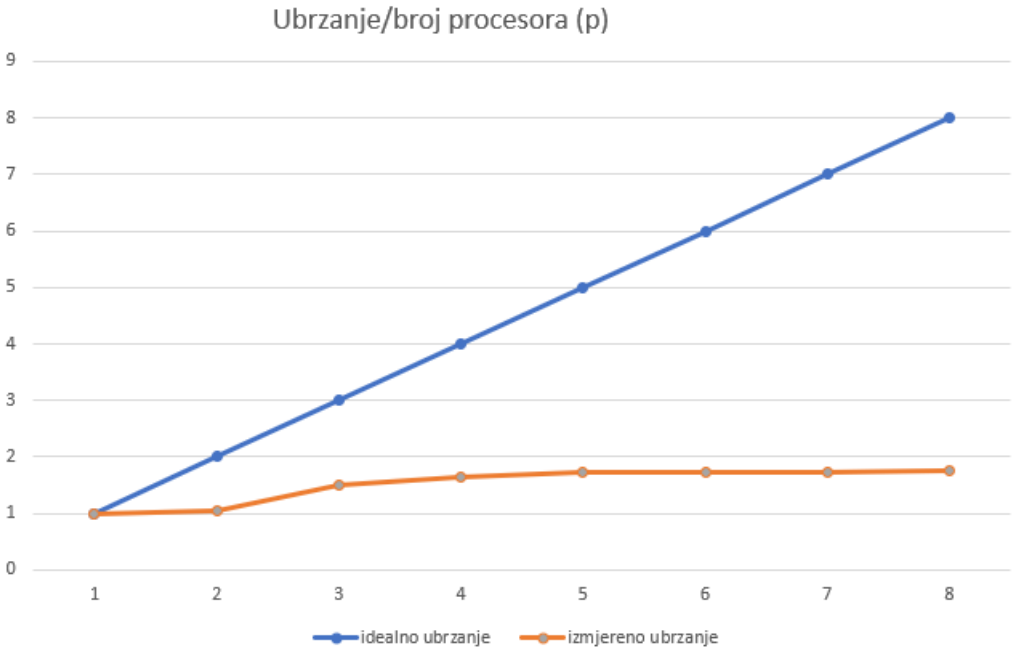
BROJ PROCESORA(p)	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Trajanje (s)	*24.82*	*22.9*	*16.94*	*14.79*	*14.41*	*14.08*	*14.22*	*13.87*
Ubrzanje	1	1.084	1.465	1.678	1.722	1.763	1.745	1.789
Učinkovitost	1	0.542	0.488	0.42	0.344	0.294	0.249	0.224

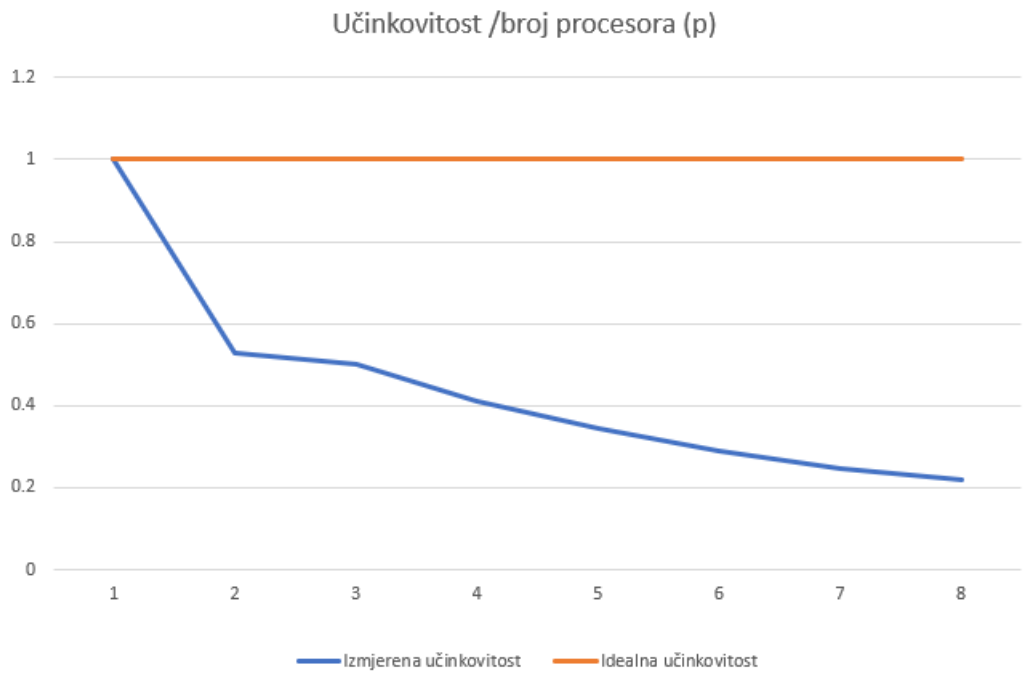




c) 343 zadataka

BROJ PROCESORA(p)	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Trajanje (s)	*25.57*	*24.16*	*17.06*	*15.53*	*14.87*	*14.78*	*14.72*	*14.67*
Ubrzanje	1	1.058	1.499	1.646	1.72	1.73	1.737	1.743
Učinkovitost	1	0.529	0.5	0.412	0.344	0.288	0.248	0.218





a) *Mjerenja u kojima se dijeli 7 zadataka su krupnozrnata jer se veći zadatci dijele jednom, odnosno zadataci su zahtjevniji a komunikacije je malo.*
b) *Mjerenja u kojima se dijeli 49 zadatka su manje krupnozrnata u odnosu na mjerenja sa 7 zadataka, u mjernjima sa 49 zadataka zadatci su dosta manji i puno je više komunikacije između procesa.*
c) *Mjerenja u kojima se dijeli 343 zadatka su sitnozrnata, zadatci su najjednostavniji u odnsou na ostale primjere i ima puno više komunikacije.*

Sva tri mjerenja su imala vrlo slična ubrzanja i učinkovitost, ali je mjerenje s 343 u mom slučaju imalo malo bolju učinkovitost u odnosu na ostala mjerenja dok je mjerenje s 49 zadataka imalo jako slično ubrzanje u odnosu na mjerenje s 343 zadatka te je mjerenje sa 7 zadataka bilo najsporije i najmanje učinkovito.