HP-SEE

# Uvod u OpenMP primjer paralelizacije

www.hp-see.eu

**Mihajlo Savic**

**Elektrotehnicki fakultet Banja Luka**

HP-SEE

High-Performance Computing Infrastructure
for South East Europe's Research Communities

# Izvori informacija

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- Prezentacija je zasnovana na materijalima sa sljedećih stranica:
  - https://computing.llnl.gov/tutorials/openMP/
  - www.slac.stanford.edu/comp/unix/farm/openmp.html
  - www.openmp.org
- Prezentacija koristi primjere iz prethodnih predavanja vezanih za MPI programiranje

# Paralelno procesiranje

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- Vrste paralelnih mašina
- Višeprocesorske mašine
- Klasteri
- Dijeljena memorija
  - **OpenMP**
- Komunikacija porukama

# OpenMP

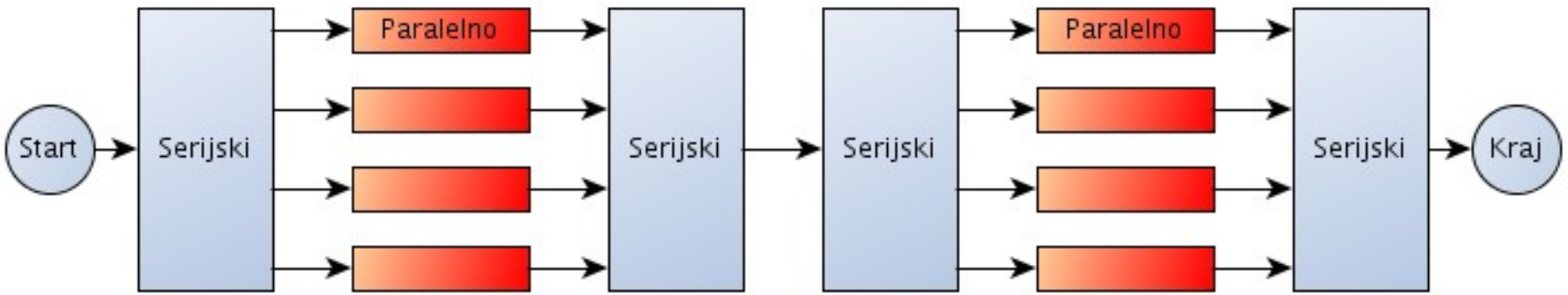
HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- OpenMP - Open Multi-Processing
  - Pojednostavljuje rad sa nitima
  - API - Application Programming Interface
  - Samo za mašine sa dijeljenom memorijom (bez komunikacije porukama)
- Ako koristimo OpenMP na čvorovima a MPI između čvorova – hibridni pristup

# OpenMP

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- Programiramo standardni serijski program
- U određenom trenutku dijelimo izvršavanje na više niti
- OpenMP upravlja nitima, poslovima koje obavljaju, sinhronizacijom, ...
- Za C/C++ koristimo direktive kompajleru #pragma i vitičaste zagrade za blok { } na koji se odnosi



# Prvi OpenMP program

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#include <omp.h>
#include <stdio.h>

int main (int argc, char **argv) {
  #pragma omp parallel
  {
    printf ("Moj prvi OpenMP program!\n");
  }
}
```

# Kompajliranje i pokretanje

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- Snimimo fajl pod nazivom omp1.c
- Kompajliranje:
  gcc -fopenmp omp1.c -o omp1
- Pokretanje:
  ./omp1
  Moj prvi OpenMP program! - osam puta
- Program će biti pokrenut u jednom procesu, ali će dio unutar #pragma { } bloka biti izvršen u osam paralelnih niti (ako je CPU sa 8 jezgara)

# Upravljanje brojem niti

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- Broj niti se najlakše kontroliše kroz promjenjivu OMP_NUM_THREADS
- Ako želimo da koristimo tačno 4 niti:
  export OMP_NUM_THREADS=4
  ./omp1
  Moj prvi MPI program! - 4 puta
- Možemo i u istoj liniji
  OMP_NUM_THREADS=16 ./omp1
  Moj prvi MPI program! - 16 puta
- Alternativno upotreba omp_set_num_threads()

# Broj niti i koja nit sam ja?

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#include <omp.h>
#include <stdio.h>

int main (int argc, char **argv) {
  int cvor, ukupno;
  #pragma omp parallel private(cvor)
  {
     cvor = omp_get_thread_num();
     ukupno = omp_get_num_threads();
     printf ("Ukupno pokrenutih niti: %d. Moj broj: %d\n", ukupno, cvor);
  }
}
```

# Izlaz programa

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
gcc -fopenmp omp2.c -o omp2
./omp2
Ukupno pokrenutih niti: 8. Moj broj: 6
Ukupno pokrenutih niti: 8. Moj broj: 1
Ukupno pokrenutih niti: 8. Moj broj: 7
Ukupno pokrenutih niti: 8. Moj broj: 3
Ukupno pokrenutih niti: 8. Moj broj: 0
Ukupno pokrenutih niti: 8. Moj broj: 5
Ukupno pokrenutih niti: 8. Moj broj: 4
Ukupno pokrenutih niti: 8. Moj broj: 2
```

# Paralelizacija i komunikacija

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- Za razliku od MPI-ja, ovdje nema slanja i primanja poruka
- Sve se radi preko dijeljene memorije
- Promjenjive mogu biti zajedničke (shared) ili privatne (private)
- OpenMP olakšava rad sa paralelizacijom for petlji, itd.
  - Automatski ili statički ili dinamički dijeli petlju

# Primjer rada sa matricom

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#define N 8
#include <stdio.h>

int main (int argc, char **argv) {
  double matrica[N][N], suma=0.0;  int i,j;
  for(i=0;i<N;i++) {
    for(j=0;j<N;j++) { matrica[i][j]=i*N+j;
      printf("%12.2f ",matrica[i][j]); }
    printf("\n"); }

    for(i=0;i<N;i++)
      for(j=0;j<N;j++)
        suma+=matrica[i][j];
  printf("Suma elemenata matrice je: %12.2f\n",suma);
}
```

# Primjer rada sa matricom

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#define N 8
#include <stdio.h>
#include <omp.h>
int main (int argc, char **argv) {
  double matrica[N][N], suma=0.0;  int i,j;
  for(i=0;i<N;i++) {
    for(j=0;j<N;j++) { matrica[i][j]=i*N+j;
      printf("%12.2f ",matrica[i][j]); }
    printf("\n"); }
  #pragma omp parallel for reduction(+:suma) private(j)
    for(i=0;i<N;i++)
      for(j=0;j<N;j++)
        suma+=matrica[i][j];
  printf("Suma elemenata matrice je: %12.2f\n",suma);
}
```

# omp_pi.c

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <omp.h>

int main (argc, argv)
     int argc;
     char *argv[];
{
  long double pi_30_decimala = 3.14159265358979323846264338327 9; // referentni pi
  long int i, intervala,broj,ukupno; // pomocne varijable
  long double pi, h, suma, x; // pomocne varijable
  if (argc!=2) { // trebaju nam tacno dva unosa u komandnoj liniji
    printf("Uputstvo: %s <broj_intervala>\nPrimjer: %s 1000\n",argv[0],argv[0]);
    return 1;
  }
  intervala=strtoll(argv[1],NULL,10); // koliko intervala za racunanje
```

# omp_pi.c

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#pragma omp parallel private(broj)
{
  broj = omp_get_thread_num();  ukupno = omp_get_num_threads();
  if (broj==0) printf("Intervala: %ld Niti: %ld\n",intervala,ukupno);
}
  if (intervala > 0) { // ako je vise od 0 racunamo
      suma = 0.0; h = 1.0 / (long double) intervala;
      #pragma omp parallel for reduction(+:suma)
      for (i = 1; i <= intervala; i ++)
          suma += (4.0 / (1.0 + pow(h * ((long double) i - 0.5),2)));
      pi = h * suma;
  }
  if (omp_get_thread_num()==0)
      printf ("pi: %.30Lf greska: %.30Lf\n", pi, (pi - pi_30_decimala));
  return 0;
}
```

# Pokretanje i performanse

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- `gcc -fopenmp omp_pi.c -o omp_pi`
- `./omp_pi 1000`

```
Intervala: 1000 Niti: 8
pi: 3.14159273692312657005019926881 3 greska:
0.000000083333333454052235800269
```

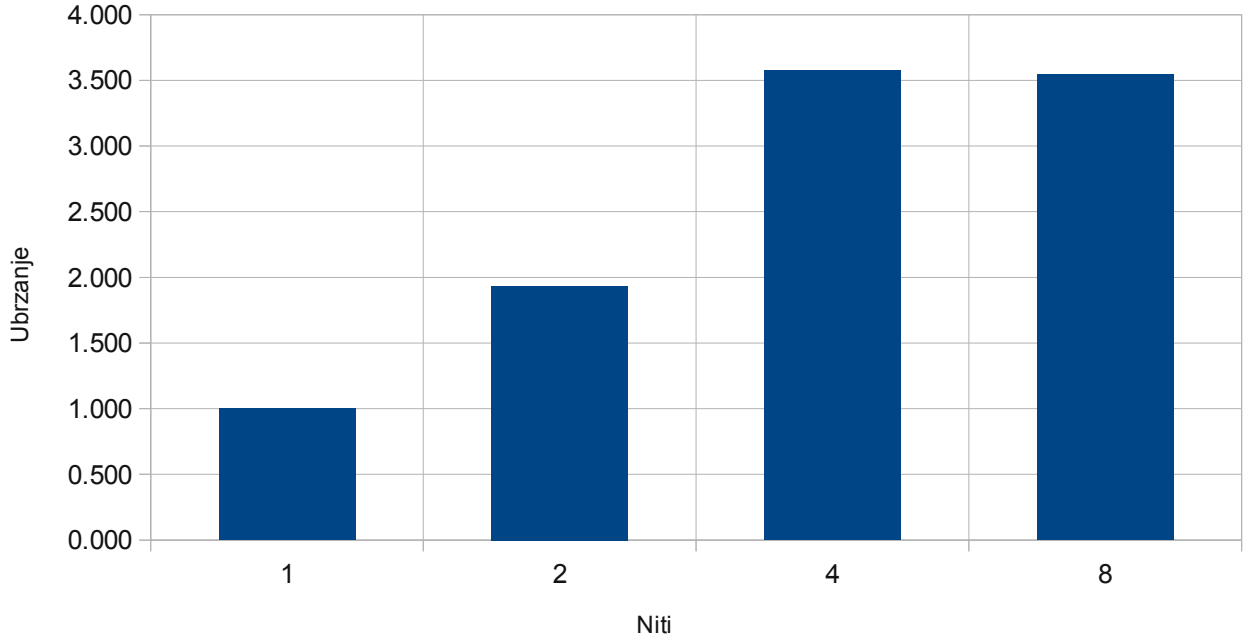
- `time ./omp_pi 1000000000`

```
Intervala: 1000000000 Niti: 8

pi: 3.14159265358979333652468535209 6 greska:
0.000000000000000220526721883552


real    0m2.210s
user    0m17.350s
sys     0m0.000s
```

# Performanse

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

| Niti | Real [s] | User [s] | Ubrzanje |
|---|---|---|---|
| 1 | 7.811 | 7.800 | 1.000 |
| 2 | 4.039 | 8.070 | 1.934 |
| 4 | 2.185 | 8.720 | 3.575 |
| 8 | 2.205 | 17.440 | 3.542 |



# Performanse

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

- export OMP_NUM_THREADS=32
- time ./omp_pi 10000000000

  Intervala: 10000000000 Niti: 32

  pi: 3.14159265358979323826521572955 greska: 0.00000000000000118828558104411

  real 0m21.755s

  user 2m44.540s

  sys 0m0.010s