Elektronski fakultet
Katedra za računarstvo

1 6. 2020.

PARALELNI SISTEMI

1. Projektovati trostepeni protočni sistem za izračunavanje elemenata niza H na osnovu izraza:
$h_i=((a_i+b_i)*d_i / c_i)/((g_i*(f_i+h_{i-1})+e_i) / m_i)$, i=1,2,..., n; $h_0=0$. Svaki PE u sistemu može da obavlja osnovne aritmetičke operacije i naredbu NOP. Na raspolaganju su procesni elementi sa i bez lokalne memorije.

a) Odrediti format mikronaredbe opisanog sistema;

b) Napisati mikroprogram, kao i sadržaj lokalnih memorija koji mu odgovara, za izračunavanje prva tri elementa niza H. Obratiti pažnju na optimalnost rešenja.

2. Koristeći CUDA tehnologiju, napisati program koji primenjuje konvolucioni filter za izoštravanje slike nad matricama koje predstavljaju piksele i koji su dostupni u matricama R_{NxM}, G_{NxM} i $B_{N,M,}$ dok se konvoluciona matrica nalazi u promenjivoj $K_{3x3.}$ Rezultujuće matrice $RR_{NxM,}$ GR_{NxM} i BR_{NxM} se dobijaju množenjem odgovarajuće matrice matricom K na seledeći način (matrice GR i BR se dobijaju identičnim obrascem):

$$\forall i, i \in \{0..N-2\}, \forall j, j \in \{0..M-2\},$$

$$RR_{(i+1),(j+1)} = \sum_{m=0}^{2}\sum_{n=0}^{2} R_{(i+m),(j+n)} * K_{m,n} / \sum_{k=0}^{2}\sum_{l=0}^{2} K_{k,l}$$

Elementi u vrstama sa indeksima 0 i M-1, kao i kolonama 0 i N-1 u rezultujućoj matrici ostaju nepromenjeni.
Veličine vektora N i M unosi korisnik. Maksimalno redukovati broj pristupa globalnoj memoriji. Obratiti pažnju na efikasnost paralelizacije.

Primer: $RR_{7,8} = (R_{6,7} * K_{0,0} + R_{6,8} * K_{0,1} + R_{6,9} * K_{0,2} + R_{7,7} * K_{1,0} + R_{7,8} * K_{1,1} + R_{7,9} * K_{1,2} + R_{8,7} * K_{2,0} + R_{8,8} * K_{2,1} + R_{8,9} * K_{2,2}) / (K_{0,0} + K_{0,1} + K_{0,2} + K_{1,0} + K_{1,1} + K_{1,2} + K_{2,0} + K_{2,1} + K_{2,2})$

3. Napisati MPI program koji realizuje množenje matrice A_{mxn} i matrice B_{nxk}, čime se dobija rezultujuća matrica C. Matrica A i matrica B se inicijalizuju u master procesu. Matrica A je podeljena u blokove po vrstama (m je deljivo sa p) i to tako da proces P_i dobija vrste sa indeksima l, $l \bmod p = i$ $(0 \le i \le p-1)$ tj. vrste sa indeksima i, $i+p$, $i+2p$, ..., $i+m-p$. Master proces distribuira odgovarajuće blokove matrice A i celu matricu B svim procesima. Predvideti da se slanje blokova matrice A svakom procesu obavlja odjednom. Svaki proces obavlja odgovarajuća izračunavanja i učestvuje u generisanju rezultata koji se prikazuje u procesu koji sadrži minimum svih vrednosti matrice C. Zadatak rešiti korišćenjem grupnih operacija.

4. a) Napisati MPI program kojim se elementi u matrici C procesa sa rankom i($i \neq 0$) formiraju slanjem/primanjem podataka iz matrice A_{nxn} (n-parno) procesa sa rankom 0. Proces 0 šalje po deo matrice A ostalim procesima. Proces 1 dobija A(i,j) for i=0,...,n/2-1 i j=n/2,...,n-1, proces 2 i=n/2,...,n-1, and j=0,...,n/2-1 i proces 3 A(i,j) for i=n/2,...,n-1 and j=n/2,...,n-1. Primanje podataka se za svaki proces odvija na isti način prikazan za n=4 na slici za slučaj primanja u proces 1.

$$A=\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \rightarrow C=\begin{bmatrix} 0 & 0 & 0 & a_{00} \\ 0 & 0 & a_{01} & a_{10} \\ 0 & a_{11} & a_{00} & a_{01} \\ a_{10} & a_{11} & 0 & 0 \end{bmatrix}$$

Ostali elementi matrice C su jednaki nuli. Rešenje realizovati tako da P0 sa svakim procesom komunicira samo jednim slanjem/primanjem podataka.

b) Napisati OpenMP kod koji sadrži sledeću petlju:

```
v = start ;
sum =0;
for ( int i = 0; i < N ; i++)
{
   sum = sum + f ( v ) ;
   v = v + step ;
}
```

i proučiti da li moguće izvršiti njenu paralelizaciju. Ako nije izvršiti njenu transformaciju tako da paralelizacija bude moguća. Vrednosti za promenljive *start* i *step*, kao i kod koji relizuje funkciju *f* su poznati pre petlje. Nakon petlje treba prikazati vrednosti za promenljive *v* i *sum*, generisanim u okviru petlje. Testiranjem sekvencijalnog i paralelnog rešenja za proizvoljno N i proizvoljan broj niti, pokazati korektnost paralelizovanog koda.