Arhitectura sistemelor de calcul paralel - examen

15 iunie 2021

Durata examenului este de două ore. Tratați următoarele subiecte și transmiteți soluția dumneavoastră (coduri sursă C MPI) prin e-mail, la adresele lucian@solid.fizica.unibuc.ro, lucian.ion@g.unibuc.ro în intervalul 12:00 - 12:30.
Mesajele electronice trimise după ora 12:30 nu vor fi luate în considerare.

1. Scrieți un program C MPI care să poată fi rulat cu *nproc* procese (minimum $nproc = 4$) și să asigure următoarea funcționalitate:

   (a) Procesul cu rangul 0 alocă dinamic blocurile *data, rdata* pentru a stoca $N = 100$ valori double. Inițializează blocul *data* cu o partiție uniformă cu N valori a intervalului mărginit de $a = 0$ și $b = 100$;

   (b) Toate procesele (inclusiv procesul 0!) alocă dinamic blocurile *local_data, local_rdata* pentru a stoca $\frac{N}{nproc}$ valori double;

   (c) Procesul cu rangul 0 distribuie valorile din *data* către toate procesele (*nproc* subintervale egale, fiecare cu $N/nproc$ valori), valori care vor fi stocate în blocul *local_data*. Partiția inițială va fi: procesul cu rangul 0 tratează primul subinterval, procesul cu rangul 1 pe al doilea, etc.;

   (d) Toate procesele calculează pentru fiecare valoare x din *local_data* expresia:

   $$\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(x-45)^2}{2\sigma^2}\right) \tag{1}$$

   cu $\sigma = 10.5$, rezultatul fiind stocat succesiv în blocul *local_rdata*;

   (e) Procesul 0 colectează **ordonat** valorile din *local_rdata* de la toate procesele în blocul *rdata*, apoi scrie rezultatul într-un fișier text *data.dat*, pe două coloane, în formatul *"%.7lf\t%.7lf\n"*. Prima coloană conține valorile din *data*, iar a doua valorile corespunzătoare din *rdata*.

   (f) Funcții MPI sugerate: *MPI_Init(), MPI_Comm_size(), MPI_Comm_rank(), MPI_Scatter(), MPI_Gather(), MPI_Finalize()* **(4p)**.

2. Algoritmul QAWC este o procedură adaptativă de integrare numerică în sens de valoare principală Cauchy a funcțiilor de tipul

   $$\int_a^b \frac{f(x)}{x-c}dx = \lim_{\epsilon\to 0}\left(\int_a^{c-\epsilon}\frac{f(x)}{x-c}dx + \int_{c+\epsilon}^b \frac{f(x)}{x-c}dx\right) \tag{2}$$

   cu o singularitate a integrandului în punctul $c \in (a, b)$. Biblioteca de funcții numerice GSL implementează acest algoritm în funcția
   *int gsl_integration_qawc (gsl function * f , double a , double b , double c , double epsabs , double epsrel , size t limit , gsl integration workspace * workspace , double * result , double * abserr).*
   Programul secvențial *qawc_serial.c*, atașat, calculează numeric integrala de mai sus pentru funcția:

   $$f(x) = e^{-0.1\cdot(x-5)^2}\cdot\sqrt{x} \tag{3}$$

   pe intervalul $[1, 10]$ cu $c = 4.35$. Vi se cere să-l paralelizați, astfel încât să asigure următoarea funcționalitate:

   (a) Vor fi lansate în execuție $nproc = 4$ procese, procesul cu rangul 0 (procesul master) definind valorile $a = 1$, $b = 10$, $c = 4.35$ și distribuind inițial o partiție uniformă a intervalului de integrare (a, b) (patru subintervale egale, fiecare de lungime $\frac{b-a}{nproc}$);

   (b) O soluție posibilă ar fi ca fiecare proces să definească un șir *double local_lims[2];* în care să recepționeze limitele de integrare inferioară și superioară de la procesul cu rang 0 (evident, acesta își va defini local limitele de integrare);

(c) Partiția inițială va fi: procesul cu rangul 0 tratează primul subinterval, procesul cu rangul 1 pe al doilea, etc.;

(d) Toate procesele vor defini suplimentar variabilele *double local_result, local_error;*, care vor stoca, respectiv, rezultatul și eroarea estimată pentru calculul local;

(e) Urmează partea secvențială, în care va fi aplicat algoritmul de integrare exact la fel ca în programul secvențial, însă valorile calculate vor fi stocate în *local_result* și *local_error*;

(f) Valorile locale vor fi acumulate de procesul cu rangul 0 în variabilele *result* și *error* (o soluție elegantă constă în utilizarea funcției *MPI_Reduce()*);

(g) Procesul cu rangul 0 afișează rezultatul final și estimarea marginii superioare a erorii;

(h) Funcții MPI sugerate: *MPI_Init(), MPI_Comm_size(), MPI_Comm_rank(), MPI_Send(), MPI_Recv(), MPI_Reduce(), MPI_Finalize()* **(5p)**

Se acordă 1p din oficiu. Funcțiile MPI indicate mai sus sunt doar sugestii de lucru. Orice soluție corectă, care asigură funcționalitatea cerută, va fi luată în considerare. Întrucât sunt utlizate implementări ale algoritmului QAWC incluse în biblioteca GSL, compilarea se face cu:

mpicc -o nume_program fisier_sursa.c -lgslcblas -lgsl -lm

Dacă sistemul dumneavoastră de operare este Linux Ubuntu, compilați cu:

mpicc -o nume_program fisier_sursa.c -lgsl -lgslcblas -lm

Succes!