Multiprocesorski sistemi

Uputstvo za izradu i predaju domaćih zadataka

Uvod

Domaći zadaci na predmetu Multiprocesorski sistemi se rade **samostalno** ili **u paru**. Domaći zadaci se predaju u roku o kome predmetni asistent studente blagovremeno obaveštava putem odgovarajuće liste elektronske pošte predmeta. Svaki student predaje domaće zadatke zasebno u svoj repozitorijum sa **odgovarajućim izveštajem**. Cilj domaćih zadataka je da studentima na praktičan način približi rad sa paralelnim programskim okruženjima i konceptima koji se predaju na nastavi. Na predmetu postoji tri domaća zadataka koji ukupno nose 30 poena i jedna laboratorijska vežba za ocenu koja nosi 5 poena. Domaći zadaci se mogu nezavisno predavati, a brane se u terminu koji odredi predmetni asistent. Domaći zadaci i laboratorijska vežba se zadaju iz oblasti rada sa OpenMP tehnologijom (10 poena), MPI biblioteke (10 poena), koherencije keš memorija (5 poena) i CUDA programskog modela (10 poena).

Opšta pravila

U cilju lakšeg, uniformnog prevođenja i pregledanja, domaći zadaci će biti testirani (prevođeni i izvršavani) na računaru **`rtidev5.etf.rs`**. Testiranje je delimično automatizovano, tako da će poene dobiti i biti pregledani samo oni zadaci koji se uspešno prevedu.

Svi studenti koji su upisali tekuću školsku godinu i izabrali predmet imaju nalog na ovom računaru. Nalog je istog oblika kao na studentskim servisima, a inicijalna lozinka je saopštena u terminu laboratorijskih vežbi. Studenti koji ne znaju nalog i lozinku se mogu obratiti predmetnom asistentu lično ili putem elektronske pošte isključivo sa svojih studentskih naloga kako bi potvrdili svoj identitet.

Tekstom svakog domaćeg zadatka je navedeno kako program treba testirati i sa kojim skupom ulaznih podataka. Radi korektnog procesiranja ulaza, potrebno je pre bilo kakvog unosa pozvati funkciju **`fflush(stdout)`**. Ukoliko drugačije nije navedeno, ne ispisivati sadržaj struktura podataka (nizova, matrica) na ekran. Svaka poruka koju program ispisuje treba da ima znak za kraj reda, **`\n`** na kraju. Programi nakon svog izvršavanja ne treba da čekaju na bilo kakvu akciju korisnika, već da odmah završe svoje izvršavanje. Nije potrebno da korisnik pritisne bilo koji taster ili slično (bez poruka poput "Pritisnite ENTER za kraj...").

Domaći zadaci se predaju postavljanjem u odgovarajuće SVN skladište (repozitorijum). Svim studentima je napravljeno SVN skladište na serveru **`rtidev5.etf.rs`**, sa sledećim pristupnim URL:

svn+ssh://NALOG@rtidev5.etf.rs/svn_mps/NALOG

gde je NALOG isti kao za pristup **`rtidev5.etf.rs`** računaru. Više o radu sa SVN sistemom se može videti u dokumentu koji se nalazi na sajtu predmeta. Potrebno je u skladište postaviti izvorne kodove svih programa do zadatog roka, imenovane po sledećem formatu:

`dzNzX.[c|cpp|cu]`

gde je **`N`** redni broj domaćeg zadatka, a **`X`** redni broj zadatka unutar tog domaćeg zadatka (na primer, **`dz1z2.c`**, **`dz2z4.cpp`**). Kodni raspored (encoding) datoteka sa programskim kodom mora biti ili US-ASCII ili UTF-8 with BOM (65001, with signature) ili UTF-8 without BOM (65001, without signature). **Vrlo važno je da datoteke sa izvornim kodom imaju imena tačno prema zadatom formatu i da budu u osnovnom direktorijumu skladišta. Nije dozvoljeno praviti bilo kakve posebne direktorijume.**

Ako programi koriste neki zajednički, deljeni kod, sav takav kod mora biti u jednom zajedničkom zaglavlju (**`.h`** datoteci). Prevodilac na prevođenje dobija samo datoteku sa oznakom zadatka da prevede i poveže. Ako se nešto bude nalazilo u nekoj zasebnoj **`.c`** ili **`.cpp`** datoteci prevodilac to neće prevesti. **Ukoliko se sekvencijalni kod, čija se paralelizacija vrši, nalazi u više datoteka, rešenje smesiti u jednu datoteku, imenovanu po gornjoj konvenciji.**

Podešavanje okruženja za rad sa OpenMP

OpenMP je dostupan na Windows i Linux operativnim sistemima uz odgovarajući GCC prevodilac. Prilikom prevođenja, potrebno je navesti opciju **`-fopenmp`**, a prevođenje i povezivanje programa se može izvršiti pomoću sledeće komandne linije:

Programski jezik C: **`gcc -fopenmp -lm -o dz1z1.exe dz1z1.c`**

Programski jezik C++: **`g++ -fopenmp -lm -o dz1z1.exe dz1z1.c`**

Na Windows operativnom sistemu, GCC prevodilac je dostupan u okviru Cygwin okruženja (http://www.cygwin.com/) koje simulira Linux komandno okruženje bash. Alternativno, može se koristiti GCC implementacija u okviru MinGW-w64 projekta ili Linux Subsystem for Windows.

Podešavanje okruženja za rad sa MPI okruženjem

Za rad sa MPI okruženjem na operativnom sistemu Linux dostupna je za korišćenje OpenMPI implementacija na računaru **`rtidev5.etf.rs`**. Primer prevođenja MPI programa pod Linux operativnim sistemom na računaru **`rtidev5.etf.rs`** je zadat sledećom naredbom:

Programski jezik C: **`mpicc -lm -o dz3z1.exe dz3z1.c`**
Programski jezik C++: **`mpic++ -lm -o dz3z1.exe dz3z1.cpp`**

Za rad sa MPI okruženjem na operativnom sistemu Windows, mogu se pronaći uputstva na internetu.

Podešavanje okruženja za rad sa CUDA okruženjem

Detaljna uputstva za instaliranje, podešavanje i prvo izvršavanje CUDA programa se mogu naći na adresi http://developer.nvidia.com/nvidia-gpu-computing-documentation pod nazivom CUDA Getting Started Guide (Windows) ili CUDA Getting Started Guide (Linux) u zavisnosti koji operativni sistem se koristi. Po datim uputstvima podesiti okruženje za razvoj i kontrolisano izvršavanje (debugging) CUDA programa na lokalnom računaru, ukoliko računar poseduje NVIDIA grafičku karticu. Alternativno, koristiti CUDA Toolkit (**`nvcc`**) na računaru **`rtidev5.etf.rs`**. Primer prevođenja CUDA programa pod Linux operativnim sistemom na računaru **`rtidev5.etf.rs`** je zadat sledećom naredbom:

Programski jezik C: **`nvcc -lm -o dz6z1.exe dz6z1.cu`**

Instalacija CUDA Toolkit (**`nvcc`**) se nalazi u direktorijumu **`/usr/local/cuda/`**, dok se primeri CUDA programa iz se mogu naći u direktorijumu **`/usr/local/cuda/samples/`**.

Korišćenje simulatora za koherenciju keš memorije

Sa sajta predmeta preuzeti arhivu http://mups.etf.rs/lab/MPS_Lab3_Cache_coherence_simulator.zip koja sadrži portabilnu verziju Mozilla Firefox pregledača koji podržava rad sa Vivio plugin-om. Pokrenuti pregledač i uneti adresu http://mups.etf.rs/simulatori/vivio/. Na datoj adresi su dostupni simulatori za nekoliko protokola za koherenciju keš memorije. U slučaju da Vivio simulator nije dostupan na lokalnom računaru, ispratiti uputstva za podešavanje Vivio okruženja na lokalnom računaru i podesiti lokalni računar prema njima. Odgovori na pitanja iz domaćeg zadatka u vezi koherencije keš memorije se daju u tekstualnom fajlu.