Multiprocesorski sistemi

Laboratorijska vežba 1

Pthreads – međusobno isključivanje

Uvod

Cilj vežbe je da studentima približi **`Pthreads`** niti i da ih obuči da koriste brave (**`pthread_mutex`**) za međusobno isključenje.

Podešavanje okruženja

Preuzeti http://mups.etf.rs/vezbe/pthreads/pthreads-win32.zip. Prema uputstvima u priloženom fajlu podesiti okruženje za razvoj i izvršavanje programa sa **`Pthreads`** nitima.

Zadaci

Ukoliko tekstom zadatka nije drugačije naglašeno, smatrati da je broj niti u programu (uključujući i glavnu nit, koja treba da kreira ostale, korisničke niti) određen konstantom **`NUM_OF_THREADS`**. Minimalna vrednost ove konstante za koju programi treba ispravno da rade data je u uglastim zagradama na kraju svakog zadatka.

Napomena: vrednost broja π se može statistički odrediti na više načina uz generator pseudoslučajnih brojeva uniformne raspodele. Jedan od načina je generisanjem vrednosti (tačaka u ravni) sa koordinatama u intervalu [0,1]. Vrednost broja π se tada može odrediti na osnovu odnosa broja tačaka koje se nalaze u delu kruga poluprečnika 1 sa centrom u koordinatnom početku i broja tačaka koje pripadaju kvadratu stranice 1 koji obuhvata sve generisane tačke.

1. Sastaviti program koji određuje vrednost broja π. Glavna nit treba da obavesti ostale niti o broju tačaka koje treba da obrade (unosi se sa glavnog ulaza), prikupi podatke od ostalih niti i ne treba da učestvuje u računanju. Niti koje računaju ne smeju međusobno da se sinhronizuju i ne dele zajedničke podatke. Za sinhronizaciju koristiti samo **`pthread_join()`** [2].
2. Sastaviti program koji određuje vrednost broja π. Glavna nit treba da obavesti ostale niti o broju tačaka koje treba da obrade (unosi se sa glavnog ulaza), prikupi podatke od ostalih niti i treba da učestvuje u računanju. Sve niti dele samo jednu celobrojnu promenljivu i komuniciraju preko nje. Obezbediti ispravnost programa međusobnim isključivanjem pomoću brava (**`pthread_mutex`**) [1].
3. Sastaviti program koji će korisničke niti programa podeliti u dve grupe, prema parnosti identifikatora. Svaka nit pri kreiranju dobija od glavne niti jedinstveni identifikator, počev od 1. Glavna nit unosi ukupan broj korisničkih niti sa glavnog ulaza. Niti treba da izračunaju sumu kvadrata svojih identifikatora u grupi. Glavna nit treba da ispiše rezultat obe grupe i ne učestvuje u računanju. Niti iz različitih grupa međusobno ne komuniciraju, a unutar grupe dele najviše jednu zajedničku promenljivu.
4. Sastaviti program koji kvadrira sve elemente niza celih brojeva. Glavna nit treba da učita veličinu i elemente niza, da svakoj od preostalih niti prosledi početni i krajnji indeks dela niza koji treba da kvadrira i da sama učestvuje u kvadriranju. Po završenom kvadriranju, glavna nit ispisuje rezultujući niz. Raspodela posla treba da bude što je moguće ravnopravnija [1].
5. Sastaviti program koji pronalazi maksimalni i minimalni element u 2D matrici realnih brojeva. Glavna nit treba da učita dimenzije i elemente matrice, rasporedi posao ostalim nitima, učestvuje u poslu i na kraju ispiše rezultat. Krajnji rezultat treba da ostane u dve globalne realne promenljive **`min`** i **`max`**, koje će ujedno biti i jedine promenljive za komunikaciju između niti. Raspodelu posla izvršiti tako što će svakoj niti biti dodeljen odgovarajući broj vrsta matrice, tako da posao bude što bolje raspoređen. Zadatak rešiti tako da se niti nikada ne blokiraju (koristiti **`pthread_mutex_trylock`**) [1]. Uputstvo: koristiti posebne brave za promenljive **`min`** i **`max`**.

VAŽNO: Ukoliko u bilo kom zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku i da nastavi da izgrađuje svoje rešenje temeljima uvedene pretpostavke.