UMEÅ UNIVERSITET **18 november 2021**

Mdu

Systemnära programmering 5DV088 - 7,5hp

Namn och email:
Gustaf Söderlund - et14gsd@cs.umu.se

Innehåll

1 Changes

Det här är den första inlämningen av rapport.

2 Resonemang kring trådsäkerhet

För att uppnå trådsäkerhet i systemet börjar systemet med att initialisera ett mutex lås och en condition variabel. När antalet trådar är specifierade och programmet ska börja använda trådarna används funktionen *pthread_create* som tar en specifierad tråd, vilken trådfunktion som ska köras och ett indata som ska användas i trådfunktionen.

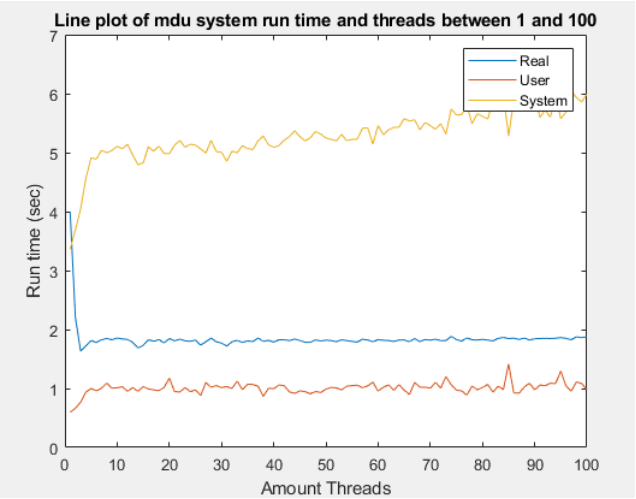
Innuti trådfunktionen låser en tråd mutex låset med hjälp av funktionen *pthread_mutex_lock* och blir låst fram tills det blir upplåst igen. När en tråd har låst mutex kan endast en tråd jobba med en specifik "uppgift". Detta hjälper till med att undvika race condition. Upplåsningen görs med hjälp av funktionen *pthread_mutex_unlock*.

Condition variabeln som används i programmet har tre syften. Vänta på en händelse, signalera att en händelse har inträffat och broadcasta. Funktionen *pthread_cond_wait* används då en tråd måste vänta på att en händelse ska inträffa. När en annan tråd orsakar att en händelse har inträffat skickas en signal till de trådar som väntar med hjälp av funktionen *pthread_cond_signal*, dvs signalera att det finns jobb att göra. När alla trådar väntar vilket indikerar på att trådfunktionen är färdig används funktionen *pthread_cond_broadcast* som väcker alla trådar innan de går ut ur trådfunktionen.

Med hjälp av funktionen *pthread_join* väntar huvudtråden in de trådar som arbetar i den konstruerade trådfunktionen. Till slut förstörs mutex låset och condition variabeln med hjälp av funktionerna *pthread_mutex_destroy* och *pthread_mutex_destroy*.

3 Analys över prestanda

För att analysera programmets prestanda skapades ett shellscript som kör programmet med olika mängder trådar från 1 till 100 trådar. Shellscriptet använder kommandot *time* som ger System, user och real tid för varje körning. Datat som givs av shellscriptet sparas i ett textdokument som i sintur används i ett matlab program som tar fram en plottad graf för körtiderna jämtemot antalet trådar. I Figur 1 åskådliggörs den plottade grafen som konstruerats i programmet matlab tillsammans med ett shellscript.



Figur 1: Körning av program med 1 till 100 trådar. På y-axeln är tiden i sekunder och på x-axeln är antalet trådar som använts under en körning.

Utifrån grafen kan man dra slutsatsen att fler trådar gör inte programmet snabbare. I grafen från Figur 1 ser man en blå linje, Real, som representerar tiden från då programmet börjar och tills det slutar. Man märker en markant lutning på Real-tiden i början. Vid en tråd tar progammet ca fyra sekunder att köra, men efter tre trådar kör programmet mellan 1,6 till 1,9 sekunder. Tiden blir inte mindre ju fler trådar man använder efter tre trådar, den ökar lite grann. Detta påvisar att 100 trådar inte är markant bättre än tre trådar i det här fallet. I grafen från Figur 1 ser man också att CPU-tiden (system och User) ökar exponentiellt och därefter är ökningen lite långsammare allteftersom. Även fast man inte får en bättre hastighet på programmet ökas användningen av CPU på grunda av fler trådar i systemet.

4 Diskussion och reflektion

Jag personligen upplever att uppgiften var lite mer avancerad än de tidigare uppgifterna som gjorts under kursens gång, men jag känner att jag lärt mig massor efter att ha utfört den. Jag hade lite problem med hur jag skulle hantera trådar, jag fick problemet att en tråd beräknade hela summan och de andra trådarna gjorde ingenting. Något som jag lyckats lösa. Faktiskt väldigt intressant att få lära sig hur man skapar och använder shellscript. Jag kan se att det är något som är väldigt användbart i framtida program för att optimera sitt program.