Gebze Teknik Üniversitesi
Bilgisayar Mühendisliği

CSE 344 - 2018 BAHAR

FİNAL PROJESİ RAPORU

SİNAN ELVEREN
111044074

1 Proje Kapsamında Yapılanlar

Multithread programala ile client-sunucu modeli yapı kurularak bunların haberleşmesi ve senkronize sekilde çalışararak üretilen yeni threadler ile istenilen işlemleri gerçekleştirip sonuclarını geri almak.

1.1 Yapılanlar

1. **Server Tarafı**
   Her provider için birer “thread“ oluşturuldu ve “condition variable” uyandırlması beklendi. Uyanış clienttan istek geldikceçe gerçekleşecek ve istek olmadıgında “condition wait” ile bekletiliyor.
   “Client” ile “Server” arasında ”socket ” olusturuldu ve ana “thread” 'imiz client'tan surekli dinleme yapmaktadır, eger socket'ten veri okunursa “clienttın” göndermiş olduğu verilerle başarılı şekilde en uygun providerların bir arrayi sıralı sekilde depolanıyor.
   `findProviderForCalc`
   (ancak ilerki aşamada sadece arrayin ilk elemanı yani ilk provider için işlem yapmaktadır)
   `createJobThread`
   fonksiyonu clienttan gelen her istek dogrultusunda basarılı seilde yeni thread oluşturuyor. Gerekli veriler `Job struct ında tutularak bu thread fonksiyona gönderiliyor.`
   `ProviderJob`
   `hesaplamalar için kullanılan thread function`
   `Hesaplama sonucları calcInfo structında global olarak tutuluyor. Mutex kullancaktım fakat kullanmadım. Mutexle korunması daha saglıklı ancak programımda eksiklikler oldugunu dusunuyorum, bu durumda saglıklı calısıyor.`
   `Tum providerlar providerFlag[max_provider] indexleri ile birlikte bu arrayde tutuyluyor ve cond variable ve mutex ile check edilip erişim saglanıyor.`
   `pthread_cond_broadcast ile providerWork da bulunan tum uyuyan thread leri uyandırıyorum ve uygun olan thread providerFlag degişkeni ve providerFlag[index] ile dogru bir şekilde göreve hazır hale getirlilyor.`
   `İlgili provider verilen gorevi yaparak işlem sonucunu bulup bitirdikten sonra Client a iletmek üzere calcInfo adlı struc a kaydediyor.`
   `Server thread tarafından olusturlan yeni thread bu talimatları verdikten sonra providerın işinin bitişinin ardından başarılı şekilde bu calcInfo daki bilgileri write() ile thread e gondermektedir.`
   `Bu sırada ana threadimiz(server thread) yeni bir thread olusturup tekrar socket dinlemeye gitmişti.`
   `2Sonraki adımların gerçekleşmesi bu döngü ile devam edebilmektedir.`

   Sinyal handler tam olarak dogru olmayabilir ama implement edilmiştir.
   Senkronizasyon çoklu client çalışmasında sorun cıkaracaktır, çünkü conditinvariablelar daha efektif sekilde kullanmam gerekiyordu. Zamanım yetmediğinden dolayı en azından düzgün birşekilde çalışır hale getirmeye çalıştım. Valgrind ile testte oluşan memory leak pthread fonksiyonları yuzunden oldugunu dusnuyorum. Cancelable thread kullanmam gerekiyordu.