İşletim Sistemleri

Proje -2-

December 21, 2020

Proje Başlangıç:21.12.2020

Ara Rapor:28.12.2020

Proje Bitiş:04.01.2021

1 Ödev

1.1 Part A- Thread Senkronizasyonu

Ana thread tarafından N adet işçi thread oluşturacak bir program kodlayınız. Program K adet düğüm sayısına sahip paylaşılabilir bir arama ağacı içermelidir (global değişken olarak tanımlanmalıdır). Bu ağaç tüm threadler arasında paylaşılacaktır. Ağaç pointerlar yardımıyla implement edilecektir. Dizi kullanamazsınız. Ağaç, girdi dosyaları içerisindeki şu ana kadar olan en büyük K değerlerini tutacaktır. Her bir worker thread yalnızca bir dosya ile işlem yapabilir(her bir satırda 1 integer değer olacak). N adet input dosyası olacaktır. Her bir worker thread aynı anda dosyadan yalnızca bir integer değeri okuyabilir. Bu kez fscanf ile okuma işlemi yapabilirsiniz(daha kolay). İnput dosyası içerisinden değerler okunduktan sonra, worker threadler eğer şartlar sağlanıyorsa değeri ağaca ekleyecektir değilse integer değer bırakılacak ve dosyadaki diğer integer değer alınacak. Tüm N adet worker threadler bu şekilde eş zamanlı olarak çalışacaklardır. Tüm worker threadler bitirdiği zaman N dosya içinde bulunan en büyük K adet integer değeri ağaçta göreceğiz. Ardından main thread bu değerleri sıralı azalan bir şekilde çıktı dosyasına yazacaktır. Ağacı kendiniz oluşturacaksınız. Herhangi bir kütüphane veya hazır kod kullanamazsınız.
Programı part1 K N infile1 ... infileN outfile şeklinde çalıştırılacaktır.
Eğer threadleri senkronize etmeniz gerekirse POSIX mutex ve durum değişkenleri kullanabilirsiniz.

1.2 Part B- Process Senkronizasyonu

Aynı programı tekrar yazacaksınız. Fakat input dosyalarını işlemek için worker processler kullanmalısınız. Aynı zamanda parent ve child processler arasında bilgi paylaşımı için shared memory kullanacaksınız. Bu demek oluyorki shared memory'i kullanmak için veri yapısı implement etmelisiniz. Paylaşılan bellek yeterince büyük olmalı. Yapı binary arama ağacı veya bir heap veya basit bir dizi olabilir. Size kalmış. Fakat yapı ne olursa olsun, girdi dosyaları içerisindeki şu ana kadar olan en büyük K değerlerini tutacaktır. Bu demek oluyor ki,paylaşılan veri yapsının içeriği input dosyasının child process tarafından işlenirken dinamik olarak değişmesi gerektiğidir. İnteger değerleri shared memory dışında herhangi bir yerde saklayamazsınız.(her bir child process için 1 dizi gibi.)
Programı part2 K N infile1 ... infileN outfile
Processleri senkronize etmek için POSIX semaphores kullanılacaktır.

1.3 Part C- Deneysel Çalışmalar

Her iki program içinde eğer threadleri ve processleri senkronize etmezsek ne olur? Aynı anda yanlış sonuçlar elde eder miyiz?
Çeşitli parametreler kullanarak ölçüm ve kıyaslamalar yapınız. Örneğin;N, K, input dosyalarında bulunan integer sayısı gibi değerler sabit tutulup veya değiştirilerek sonuçlar elde edilebilir. Sonuçlar çizdirilebilir. Deneysel deneyimlerinizi ve sonuçlarınızı rapor haline getirip PDF formatına çeviriniz. Rapor içerisinde tablolar, grafikler ve sonuçlar olmalıdır.

2 Rapor Teslimi

Teslim dosyanız içerisinde yazdığınız rapor.pdf, Öğrenci numaralarınızın ve adınızın bulunduğu README.txt dosyası, Makefile, part1.c,part2.c dosyaları olmalıdır.

3 İpucu ve Açıklamalar

- Kademeli ve sabırla çalışın.
- Ödevi son güne bırakmayın.
- **"man shm_overview"** komutu ile linux shell'de POSIX shared memory kullanımı hakkında bilgi alabilirsiniz.
- **"man sem_overview"** komutu ile linux shell'de POSIX semaphores kullanımı hakkında bilgi alabilirsiniz.
- **"man pthreads"** komutu ile linux shell'de POSIX threads ve aynı zamanda mutex ve durum değişkenleri kullanımı hakkında bilgi alabilirsiniz.
- Max K değeri 10000, min 100.
- Max N değeri 10, min 1.
- input dosyaları çok fazla sayıda integer değer içerebilir.