İşletim Sistemleri

Proje 1

Proje Başlangıç: 23.11.2020

Ara Rapor:3 0.11.2020

Proje Bitiş: 7.12.2020

Assignment

Part 1. Ana process'in yanı sıra N adet child process oluşturabilecek bir program tasarlayınız. **N** adet child process ile içerisinde birçok integer sayı bulunan N adet dosya içerisindeki **k.** en büyük sayıyı buldurma işlemi yapılacaktır. N adet girdi dosyası aynı anda N işlem tarafından işlenecektir. Her bir girdi dosyası çok büyük sayıda pozitif integer değerler içerebilir. Output dosyasına yazılacak olan sonuçlar büyükten küçüğe doğru sıralanmış olacaktır. K 1 ile 1000 arasında bir sayı olabilir. N değeri 1 ve 5 arasında bir sayı olabilir. Input ve output dosyaları ASCII text dosyaları olacaktır. Program aşağıdaki şekilde çağrılacaktır:

findtopk <k> <N> <infile1> ...<infileN> <outfile>

Her bir child process bir adet input dosyasını okuyacak ve işleyecektir. İşlenen dosyası intermediate (ara) dosyaya yazacaktır. Tüm child işlemler bittiği zaman ana process bu intermediate dosyaları okuyup işleyecektir.

Normalde dosyaları okumak ve yazmak için fscanf ve fprintf kullanabilirdiniz. Fakat bu projede low-level I/O komutları olan read() ve write() kullanılacaktır. Bu sayede low-level I/O işlemlerinde pratik yapabilirsiniz. Dosyaları açmak ve kapatmak için open() ve close() fonksiyonlarını kullanabilirsiniz. Her seferinde yalnızca bir girdi dosyasında bir (veya çok az) karakter okuyabilirsiniz. İnput dosyaları içerisinde boşluk karakterleri ile (space,tab,newline) ile ayrılmış sayılar içermektedir. Outpur dosyasında her bir satırda yalnızca bir integer değer olmalıdır. Tüm işlemler bittiğinde processler sonlandığında program, intermediate dosyaları silmelidir.

Part 2. Aynı projeyi, child processlerdeki bilgileri parent processlere aktarırken POSIX message queues kullanarak yapınız. Intermediate dosyalar kullanılmayacaktır. Bu durumda programın adı findtopk_mqueue olacaktır.

Part 3. Aynı projeyi POSIX thread'leri kullanarak yapınız. Her bir girdi dosyası için ayrı bir thread oluşturulacaktır. Programın adı findtopk_thread olacaktır. Thread global değişkenleri bir thread'den diğer thread'e bilgileri aktarmak için kullanılacaktır. Bu durumda intermediate dosyalara ve message queue'lere ihtiyaç duyulmamaktadır.

Part 4. Experimentation. Yazmış olduğunuz programları çeşitli k, N ve dosya boyutu(bir dosyadaki integer numara sayısı) için çalıştırarak çalışma sürelerini ölçünüz. Sonuçları tablolar ve plot şeklinde raporlayınız. Bir dosyadaki integer sayıların adetini ve k değerini sabitleyiniz. Örneğin k 1000 olabilir. Bir dosya içerisindeki integer sayısı 1.000.000 olabilir. Bu değerlere göre N sayısını 1,2 ve 3 olarak değiştirerek programların çalışma sürelerini raporlayınız. Tablolar(matris şeklinde olabilir N=1,2ve 3 için 3 farklı yönteme ait çalışma süreleri) ve plot

değerler çizdirilecektir. Deney sonunda detaylı bir şekilde şekillerin tabloların ve yöntemlerin açıklandığı bir rapor yazılacaktır. Tablo, şekil isimleri düzenli ve açıklayıcı olmalıdır.

Proje Teslim.

Tüm dosyalar tek bir dosya altında toplanmalıdır. Bu dosyanın ismi grupta bulunan herhangi bir öğrencinin numarası olabilir. README.txt dosyası içerisinde tüm grup üyelerinin isimleri ve numaraları yazılmalıdır. Dosyanın içeriği; README.txt, Makefile, findtopk.c, fintopk_mqueue.c, fintopk_thread ve report.pdf olmalıdır. Make yazdığımız zaman tüm programlarınız compile edilebilmeli ve ilgili exe'ler üretilmelidir. En son dosyayı tar ve gzip ediniz. Örneğin Öğrenci numarası 123456789 olsun. 123456789 ismindee bir directory oluşturup buraya tüm dosyaları koyun. Ardından tar cvf 123456789.tar 123456789 ve gzip 123456789.tar yapınız. En son elinizde 123456789.tar.gz dosyası olacaktır. Teslim ederken bu dosyayı yükleyiniz.

- Geç teslimler kesinlikle kabul edilmeyecektir.(Bahane ne olursa olsun).
- Geç teslimler otomatik 0 alacaktır.

İpuçları ve Açıklamalar

- **Kademeli olarak çalışın.**
- **Her gün az bile olsa çalışın.**
- **Son günlere ödevi bırakmayın.**