BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
İŞLETİM SİSTEMLERİ
36. GRUP PROJE ÖDEVİ

Öğrenci Adı Soyadı	***Numarası***	***E-posta***	***Sınıf***
Ümmühan AKYILDIZ	*G221210501*	*G221210501@ogr.sakarya.edu.tr*	*2A*
Nazlı Nur ESMERAY	*B191210074*	*B191210074@ogr.sakarya.edu.tr*	*1C*
Esra EREN	*G131210306*	*G131210306@ogr.sakarya.edu.tr*	*2A*
GitHub Bağlantısı	*https://github.com/ummuhn/IsletimSistemleri*		

PROJENİN TANIMI:

Bu projede CPU sıralama yöntemlerini anlatan bir program tasarlanmıştır. FCFS, Priority ve Round Robin sıralama yöntemi kullanılarak verilen işlemin yapılması sağlanmıştır.

NEDEN CPU SIRALAMA İHTİYACI DOĞMUŞTUR?

Tek çekirdekli sistemlerde birim zamanda sadece bir işlem çalışabilir. Diğer işlemlerin çalışabilmesi için işlemci çekirdeklerinin boşalması ve tekrar zamanlanabilir hale gelmesi gerekir. Çoklu programlamanın amacı ise işlemci kullanımını en üst düzeye çıkartmaktır.

Basit bilgisayar sistemlerinde bir işlemin başlayabilmesi için diğer işlemin bitmesi gerekir. Örnek olarak işlemciye gelen bir I/O işlemi aslında o dakikadan itibaren bir bir başka cihazın işlem biriminde çalışıyor olsa da henüz o cihazdan cevap gelmeyip işlem tamamlanmadığından cevap gelene kadar CPU boşa çalışmış olur. Fakat çoklu programlama ile birlikte zamanı daha verimli kullanmaya başlarız. Birkaç işlem aynı anda bellekte barınabilir ve bu sayede bir işlem beklerken işletim sistemi CPU'yu bu işlemden uzaklaştırıp CPU'ya yeni bir işlem verebilir. Böylece işlem gücü boşa harcanmamış olur.

Bellekte(RAM) çalışmaya hazır halde bekleyen işlemlerden birini seçerek işlemciyi ona ayırır.

CPU Scheduler kararlarını işlem;

1. *Çalışma(running) durumundan bekleme(waiting) durumuna geçerken,*
2. *Çalışma(running) durumundan hazır(ready) durumuna geçişte*
3. *Bekleme(waiting) durumundan hazır(ready) durumuna geçişte*
4. *Sonlandığında(terminates) verir.*

- *1 ve 4 durumlarında yapılan zamanlama nonpreemptive(kesmeyen)*
- *Diğer durumlar ise preemptive(kesen)*
    - *Tüm modern işletim sistemleri preemptive zamanlama algoritmalarını kullanırlar.*

CPU SIRALAMA ALGORİTMALARI (CPU Scheduling Algorithms)

1. ***İLK GELEN İŞLEM ÖNCE YAPILIR "FCFS" (First-Come, First-Served Scheduling)***

CPU'nun, işlemleri gelme sırasına göre işleme aldığı zamanlama algoritmasıdır. FCFS'da ilk gelen işlem ilk yapılır, o bittikten sonra sıradaki işlem CPU'ya alınır ve bu düzen böyle devam eder.

- *İşlem önceliği, işlemlerin sıraya girme sıralarına göre belirlenir.*

2. ***ÖNCELİKLİ İŞ (Priority Scheduling)***

Her bir işleme öncelik sayısı (tamsayı) atanır. CPU en öncelikli olan işleme tahsis edilir.

- *En yüksek öncelik; en küçük tam sayıya aittir., (kabul bu yöndedir)*

Preemptive*(Kesilebilen işlem) ise işlem devam etse dahi o an daha öncelikli bir işlem geldiğinde yarıda kesilip öncelikli işlem yapılır.*

Nonpreemptive *(Kesilemeyen işlem)'lerde ise işlem tamamlandıktan sonra en yüksek öncelik kimde ise CPU'yu o işlem kullanır.*

Bu algoritmanın yanında getirdiği bir problem vardır. Bu da Starvation problemidir(açlık). Düşük öncelikli bir işlem üzerine ondan daha öncelikli işlemlerin çokça gelmesi durumunda az öncelikli işlemimiz asla CPU'yu kullanamaz. Bu sorunu gidermek için bir çözüm de vardır.

Aging *yöntemi (yaşlandırma) önüne aldığı her işlemde bu önceliği az olan işlemimizin önceliğini yavaş yavaş artırırız. Böylece bu önceliksiz işlem de sistem kaynaklarından yararlanabilir.*

3. ÇEVRİMSEL SIRALAMA (Round Robin Scheduling) "RR"

İşlemlerin belirlenen bir süreye göre sıra sıra işleme sokulmasıdır. İşlemlerin CPU'da kalabileceği maksimum süreye time quantum denir. İşlemler time quantum'a göre sıra sıra işleme girer ve çıkarlar. Sırası gelen işlem ready queue'den alınıp CPU'da işleme konur time quantum süresi bitmesi ardından ready queue'nın sonuna koyulur, ready queue'da hazır bekleyen işlem de CPU'ya verilir. Bir işlemin time quantum'dan önce tamamlanması halinde ise bir sonraki işleme geçilir.

Time quantum'un artması ve azalmasıyla ortaya çıkan bazı problemler var. Bunlar;

Çok yüksek time quantum belirlersek örneğin;

Time quantum'a 1 yıl dersek; bu bir yılda sadece bir işlem çalışacak demek oluyor yani başka bir proses çalışmayacak. Tabi bu durumda işlem bir yıldan önce biteceği için. İşlem bittikten sonra öbür işlem gelecek ... , Bir noktadan sonra aslında iş FCFS'a dönmüş oluyor.

"*Çok yüksek time quantum belirlersek iş FCFS'a döner."*

Time quantum'u çok düşük belirlememiz halinde de CPU'daki context switchlerin maliyeti artamaya başlıyor. Çünkü time quantum'u küçük belirlememiz çok sık process değişikliği yapmamıza sebep oluyor.

"Çok küçük time quantum belirlemek yaşanacak context switch sayısını artırır ve buna bağlı olarak işlem maliyetini artırmış oluruz."

4. SONUÇ:

Ödev dosyasında verilen .txt dosyasındaki veriler FJFS, Priority Scheduling ve Round Robin algoritamaları kullanılarak CPU sıralama işlemleri gerçekleştirilmiştir.

KAYNAK:

Tezcan, H., ., (2019, 23 Ocak), CPU Scheduling Konu Anlatımı,
https://www.hasantezcan.dev/blog/cpu-scheduling.html