

Planlama:Çok Düzeyli Geri Bildirim Kuyruğu

Bu bölümde, Çok Düzeyli Geri Bildirim Kuyruğu (MLFQ) olarak bilinen programlamaya yönelik en iyi bilinen yaklaşımlardan birini geliştirme sorununu ele alacağız. Çok Düzeyli Geri Bildirim Kuyruğu (MLFQ) planlayıcısı ilk olarak Corbato ve arkadaşları tarafından tanımlanmıştır. 1962'de [C+62] Uyumlu Zaman Paylaşım Sistemi (CTSS) olarak bilinen bir sistemde ve bu çalışma, daha sonra Multics üzerinde yapılan çalışmalarla birlikte, ACM'nin Corbato'ya en büyük ödülü olan Turing Ödülü'nü vermesine yol açtı. Zamanlayıcı daha sonra, bazı modern sistemlerde karşılaşılabilecek uygulamalara göre yıllar boyunca geliştirildi.

MLFQ'nun ele almaya çalıştığı temel sorun iki yönlüdür. İlk olarak, bir önceki notta gördüğümüz gibi, önce daha kısa işler çalıştırılarak yapılan geri dönüş süresini optimize etmek istiyor; ne yazık ki, işletim sistemi genellikle bir işin ne kadar süreceğini bilmez, tam olarak SJF (veya STCF) gibi algoritmaların gerektirdiği bilgi. İkinci olarak, MLFQ, bir sistemin etkileşimli kullanıcılara (yani oturan ve ekrana bakan, bir işlemin bitmesini bekleyen kullanıcılar) karşı duyarlı olmasını sağlamak ve böylece yanıt süresini en aza indirmek ister; ne yazık ki, Round Robin gibi algoritmalar yanıt süresini azaltır, ancak geri dönüş süresi açısından korkunçtur. Bu nedenle, sorunumuz: genel olarak bir süreç hakkında hiçbir şey bilmediğimize göre, bu hedeflere ulaşmak için nasıl bir programlayıcı oluşturabiliriz? Zamanlayıcı, sistem çalışırken yürüttüğü işlerin özelliklerini nasıl öğrenebilir ve böylece daha iyi çizelgeleme kararları alabilir?

Dönüm Noktası:

KUSURSUZ BİLGİ OLMADAN NASIL PLANLANIR?

Hem etkileşimli işler için yanıt süresini en aza indiren hem de iş uzunluğu hakkında önceden bilgi sahibi olmadan geri dönüş süresini en aza indiren bir programlayıcıyı nasıl tasarlayabiliriz?

İpucu: GEÇMİŞTEN ÇIKARIM

Çok seviyeli geri bildirim kuyruğu, geleceği tahmin etmek için geçmişten öğrenen bir sistemin mükemmel bir örneğidir. Bu tür yaklaşımlar, işletim sistemlerinde (ve donanım dalı tahmin edicileri ve önbelleğe alma algoritmaları dahil olmak üzere Bilgisayar Bilimindeki diğer birçok yerde) yaygındır. Bu tür yaklaşımlar, işlerin davranış aşamaları olduğunda ve dolayısıyla öngörülebilir olduğunda işe yarar; Tabii ki, bu tür teknikler konusunda dikkatli olunmalıdır, çünkü bunlar kolayca yanlış olabilir ve bir sistemi hiçbir bilgi olmadan sahip olduklarından daha kötü kararlar almaya yönlendirebilir.

8.1 MLFQ: Temel Kurallar

Böyle bir zamanlayıcı oluşturmak için, bu bölümde çok seviyeli bir geri besleme kuyruğunun arkasındaki temel algoritmaları açıklayacağız; uygulanan birçok MLFQ'nun özellikleri farklı olsa da [E95], çoğu yaklaşım benzerdir.

İşlemimizde, MLFQ'nun her biri farklı bir öncelik düzeyine atanan bir dizi farklı kuyruğu vardır. Herhangi bir zamanda, çalışmaya hazır olan bir iş tek bir kuyruktadır. MLFQ, belirli bir zamanda hangi işin çalıştırılacağına karar vermek için öncelikleri kullanır: daha yüksek önceliğe sahip bir iş (örn.daha yüksek kuyruk) çalıştırmak için seçilir.

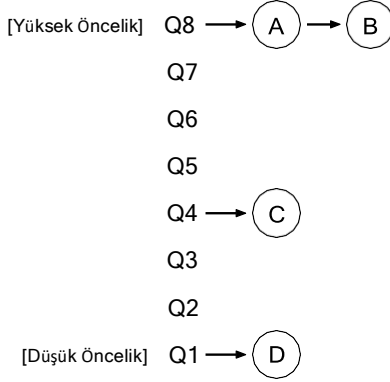
Tabii ki, birden fazla iş belirli bir kuyrukte olabilir ve dolayısıyla aynı önceliğe sahip olabilir. Bu durumda, bu işler arasında yalnızca sıralı çizelgeleme kullanacağız. Thus, we arrive at the first two basic rules for MLFQ:

- **Kural 1:** Eğer $\text{Priority}(A) > \text{Priority}(B)$, A çalışır (B çalışmaz).
- **Kural 2:** Eğer $\text{Priority}(A) = \text{Priority}(B)$, A ve B RR'da çalışır.
- **Priovity= Öncelik**

Bu nedenle, MLFQ planlamasının anahtarı, programlayıcının öncelikleri nasıl belirlediğinde yatmaktadır. MLFQ, her işe sabit bir öncelik vermek yerine, bir işin önceliğini gözlemlenen davranışına göre değiştirir. Örneğin, klavyeden girdi beklerken bir iş sürekli olarak CPU'dan vazgeçerse, etkileşimli bir süreç bu şekilde davranabileceğinden MLFQ önceliğini yüksek tutacaktır. Bunun yerine, bir iş CPU'yu uzun süre yoğun bir şekilde kullanıyorsa, MLFQ önceliğini azaltacaktır. Bu şekilde MLFQ, çalışırken süreçler hakkında bilgi edinmeye çalışacak ve böylece gelecekteki davranışını tahmin etmek için işin geçmişini kullanacaktır.

Belirli bir anda kuyrukların nasıl görünebileceğinin bir resmini ortaya koysaydık, aşağıdakine benzer bir şey görebilirdik (Şekil 8.1). Şekilde, iki iş (A ve B) en yüksek öncelik seviyesinde, C işi ortada ve iş D en düşük önceliğe sahiptir. MLFQ'nun nasıl çalıştığına dair mevcut bilgimiz göz önüne alındığında, sistemdeki en yüksek öncelikli işler oldukları için programlayıcı zaman dilimlerini A ve B arasında değiştirir; C ve D'nin asla kaçamayacağı kötü işler - bir rezalet!

Tabii ki, sadece bazı sıraların statik anlık görüntüsünü göstermek size MLFQ'nun nasıl çalıştığı hakkında gerçekten bir fikir vermiyor. İhtiyacımız olan, iş önceliğinin zaman içinde nasıl değiştiğini anlamaktır. Ve bu, sadece bu kitaptan bir bölümü ilk kez okuyanları şaşırtacak şekilde, tam da bundan sonra yapacağımız şey.



Şekil 8.1: MLFQ Örneği

Tabii ki, sadece bazı sıraların statik anlık görüntüsünü göstermek size MLFQ'nun nasıl çalıştığı hakkında gerçekten bir fikir vermiyor. İhtiyacımız olan, iş önceliğinin zaman içinde nasıl değiştiğini anlamaktır. Ve bu, sadece bu kitaptan bir bölümü ilk kez okuyanları şaşırtacak şekilde, tam da bundan sonra yapacağımız şey.

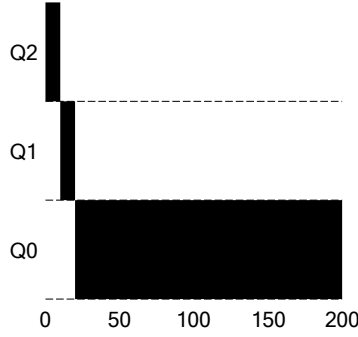
8.2 Deneme #1: Öncelik Nasıl Değiştirilir

Şimdi MLFQ'nun bir işin öncelik seviyesini (ve dolayısıyla hangi kuyrukta olduğunu) bir işin ömrü boyunca nasıl değiştireceğine karar vermeliyiz. Bunu yapmak için iş yükümüzü aklımızda tutmalıyız: kısa süren (ve sıklıkla CPU'yu bırakabilen) etkileşimli işler ve çok fazla CPU süresi gerektiren ancak daha uzun süren bazı "CPU'ya bağlı" işler karışımı. Yanıt süresinin önemli olmadığı durumlarda. İşte bir öncelik ayarlama algoritmasına yönelik ilk girişimimiz:

- **Kural 3:** Bir iş sisteme girdiğinde en yüksek önceliğe (en üst sıraya) yerleştirilir.
- **Kural 4a:** Bir iş çalışırken tüm zaman dilimini kullanırsa, önceliği azalır (yani, bir sıra aşağı iner).
- **Kural 4b:** Bir iş, zaman dilimi dolmadan önce CPU'dan vazgeçerse, aynı öncelik seviyesinde kalır.

Örnek 1: Uzun Süren Tek Bir İş

Bazı örneklere bakalım. İlk olarak, sistemde uzun süredir devam eden bir iş olduğunda ne olduğuna bakacağız. Şekil 8.2, üç kuyruklu bir zamanlayıcıda bu işe zaman içinde ne olduğunu göstermektedir.



Şekil 8.2: Zaman içinde Uzun süredir devam eden iş

Örnekte de görebileceğiniz gibi iş en yüksek önceliğe (Q2) giriyor. 10 ms'lik tek bir zaman diliminden sonra, zamanlayıcı işin önceliğini bir azaltır ve böylece iş Q1'dedir. Bir zaman dilimi için Q1'de çalıştıktan sonra iş, sistemdeki en düşük önceliğe (Q0) indirilir ve orada kalır. Oldukça basit, değil mi?

Örnek 2: Birlikte etkileşimli yeni bir iş gelmesi

Şimdi daha karmaşık bir örneğe bakalım ve umarım MLFQ'nun SJF'ye nasıl yaklaşmaya çalıştığını görelim. Bu örnekte iki iş vardır: Uzun süreli CPU yoğun bir iş olan A ve kısa süreli etkileşimli bir iş olan B. A'nın bir süredir koştuğunu ve ardından B'nin geldiğini varsayalım. Ne olacak? MLFQ, B için SJF'ye yaklaşacak mı?

Şekil 8.3 bu senaryonun sonuçlarını göstermektedir. A (siyahla gösterilmiştir) en düşük öncelikli kuyruktaki çalışmaktadır (uzun süre çalışan CPU yoğun işlerde olduğu gibi); B (gri renkle gösterilmiştir) T = 100 zamanında gelir ve bu nedenle en yüksek kuyruğa eklenir; çalışma süresi kısa olduğundan (yalnızca 20 ms), B iki zaman diliminde alt kuyruğa ulaşmadan önce tamamlar; sonra A çalışmaya devam eder (düşük öncelikte).

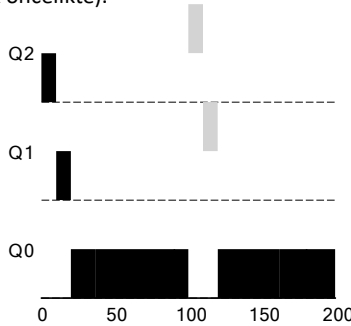
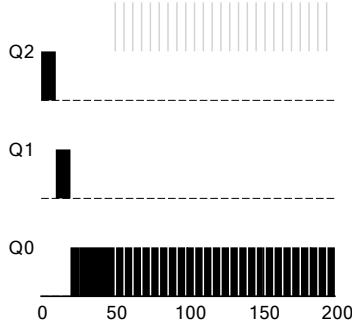


Figure 8.3: Birlikte etkileşimli yeni bir iş gelmesi



Şekil 8.4: Karma G/Ç yoğun ve CPU yoğun İş Yükü

Şekil 8.3 bu senaryonun sonuçlarını göstermektedir. A (siyahla gösterilmiştir) en düşük öncelikli kuyruktaki çalışmaktadır (uzun süre çalışan CPU yoğun işlerde olduğu gibi); B (gri renkle gösterilmiştir) $T = 100$ zamanında gelir ve bu nedenle en yüksek kuyruğa eklenir; çalışma süresi kısa olduğundan (yalnızca 20 ms), B iki zaman diliminde alt kuyruğa ulaşmadan önce tamamlar; sonra A çalışmaya devam eder (düşük öncelikte).

Bu örnekten, umarım algoritmanın ana hedeflerinden birini anlayabilirsiniz: bir işin kısa bir iş mi yoksa uzun soluklu bir iş mi olacağını bilmediği için, önce bunun kısa bir iş olabileceğini varsayar, böylece iş yüksek öncelikli. Aslında kısa bir iş ise, hızlı çalışır ve tamamlanır; kısa bir iş değilse, yavaş yavaş kuyruklarda aşağı inecek ve böylece kısa sürede uzun soluklu, daha parti benzeri bir süreç olduğunu kanıtlayacaktır. Bu şekilde MLFQ, SJF'ye yaklaşır.

Örnek 3: Peki ya G/Ç Nedir?

Şimdi biraz G/Ç içeren bir örneğe bakalım. Yukarıda Kural 4b'de belirtildiği gibi, bir süreç, zaman dilimini kullanmadan önce işlemciden vazgeçerse, onu aynı öncelik seviyesinde tutarız. Bu kuralın amacı basittir: örneğin etkileşimli bir iş çok fazla G/Ç yapıyorsa (klavyeden veya fareden kullanıcı girdisini bekleyerek), zaman dilimi tamamlanmadan önce CPU'dan vazgeçecektir; böyle bir durumda işi cezalandırmak ve böylece aynı seviyede tutmak istemiyoruz.

Şekil 8.4, bunun nasıl çalıştığının bir örneğini, uzun süre çalışan bir toplu iş A (siyahla gösterilmiştir) ile CPU için yarışan bir G/Ç gerçekleştirmeden önce CPU'ya yalnızca 1 ms ihtiyaç duyan etkileşimli bir iş B (gri ile gösterilmiştir) ile gösterir.). MLFQ yaklaşımı, B'yi en yüksek öncelikte tutar çünkü B, CPU'yu serbest bırakmaya devam eder; B etkileşimli bir işse, MLFQ etkileşimli işleri hızla yürütme hedefine daha da ulaşır.

Mevcut MLFQ'muzla İlgili Sorunlar

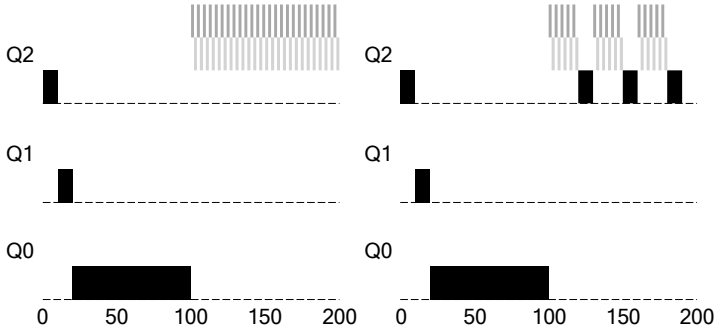
Böylece temel bir MLFQ'muz var. CPU'yu uzun süren işler arasında adil bir şekilde paylaştırarak ve kısa veya G/Ç yoğun etkileşimli işlerin hızlı bir şekilde çalışmasına izin vererek oldukça iyi bir iş çıkarıyor gibi görünüyor. Şimdiye kadar geliştirdiğimiz yaklaşım ne yazık ki ciddi eksiklikler içeriyor. Herhangi birini düşünebilir misin?

SCHEDULING:

THE MULTI-LEVEL FEEDBACK QUEUE

6

(Bu, durakladığımız ve olabildiğince dolambaçlı düşündüğünüz yerdir)



Şekil 8.5: (Solda) ve (Sağda) Öncelikli Artırma Olmadan

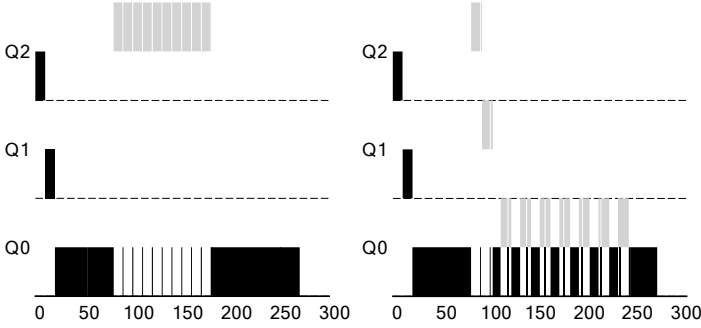
Birincisi, açlık sorunu vardır: sistemde "çok fazla" etkileşimli iş varsa, bunlar birleşerek tüm CPU zamanını tüketir ve bu nedenle uzun süreli işler hiçbir zaman CPU zamanı almazlar (aç kalırlar). Bu senaryoda bile bu işlerde biraz ilerleme kaydetmek istiyoruz.

İkincisi, akıllı bir kullanıcı programlayıcıyı oynatmak için programını yeniden yazabilir. Programlayıcıda oyun oynamak, genellikle programlayıcıyı size kaynaktan adil payınıza düşenden daha fazlasını vermesi için kandırmak için sinsice bir şeyler yapma fikrine atıfta bulunur. Tanımladığımız algoritma şu saldırıya karşı hassastır: zaman dilimi sona ermeden önce (umursamadığınız bir dosyaya) bir G/Ç işlemi gerçekleştirin ve böylece CPU'dan vazgeçin; bunu yapmak, aynı kuyrukta kalmanıza ve böylece daha yüksek bir CPU süresi yüzdesi kazanmanıza olanak tanır. Doğru yapıldığında (örneğin, CPU'dan vazgeçmeden önce bir zaman diliminin %99'unu çalıştırarak), bir iş neredeyse CPU'yu tekeline alabilir.

Son olarak, bir program davranışını zaman içinde değiştirebilir; CPU'ya bağlı olan şey bir etkileşim aşamasına geçebilir. Mevcut yaklaşımımızla, böyle bir iş şanssız olur ve sistemdeki diğer etkileşimli işler gibi ele alınmaz..

İpucu: PLANLAMA SALDIRIYA KARŞI GÜVENLİ OLMALIDIR

İster işletim sisteminin kendi içinde (burada tartışıldığı gibi) ister daha geniş bir bağlamda (örneğin, dağıtılmış bir depolama sisteminin I/O istek işleme [Y+18]) içinde olsun, bir zamanlama politikasının bir güvenlik olmadığını düşünebilirsiniz. endişe, ancak giderek artan birçok durumda, tam olarak budur. Dünyanın dört bir yanından kullanıcıların CPU'ları, bellekleri, ağları ve depolama sistemlerini paylaştığı modern veri merkezini düşünün; ilke tasarımı ve uygulamasında özen gösterilmeden, tek bir kullanıcı başkalarına zarar verebilir ve kendisi için avantaj elde edebilir. Bu nedenle, zamanlama politikası, bir sistem güvenliğinin



Şekil 8.6: (Sol) ve (Sağ) Oynama Toleransı Olmadan

8.3 Deneme #2: Öncelik Artışı

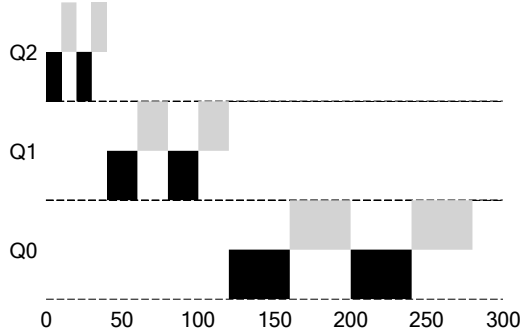
Kuralları değiştirmeye çalışalım ve açlık sorununu önleyip önleyemeyeceğimize bakalım. CPU'ya bağlı işlerin biraz ilerleme kaydetmesini garanti etmek için ne yapabiliriz (çok fazla olmasa bile?).

- Buradaki basit fikir, tüm işlerin önceliğini periyodik olarak artırmaktır. sistemde. Bunu başarmanın pek çok yolu var ama basit bir şey yapalım: hepsini en üstteki kuyruğa atalım; dolayısıyla, yeni bir kural:
- **Kural 5:** Belirli bir S süresinden sonra, sistemdeki tüm işleri en üstteki kuyruğa taşıyın.

Yeni kuralımız aynı anda iki sorunu çözüyor. İlk olarak, süreçlerin aç kalmaması garanti edilir: en üst sırada oturan bir iş, CPU'yu diğer yüksek öncelikli işlerle sıralı bir şekilde paylaşacak ve böylece sonunda hizmet alacaktır. İkinci olarak, CPU'ya bağlı bir iş etkileşimli hale gelirse, öncelik yükseltmesini aldıktan sonra zamanlayıcı onu uygun şekilde ele alır.

Bir örnek görelim. Bu senaryoda, iki kısa süreli etkileşimli işle CPU için rekabet ederken uzun süreli bir işin davranışını gösteriyoruz. Şekil 8.5'te (sayfa 6) iki grafik gösterilmektedir. Solda, öncelik artışı yoktur ve bu nedenle, iki kısa iş geldiğinde uzun süren iş aç kalır; sağda, her 50 ms'de bir öncelik artırma var (bu muhtemelen çok küçük bir değerdir, ancak burada örnek için kullanılmıştır) ve bu nedenle en azından uzun soluklu işin bir miktar ilerleme kaydedeceğini garanti ediyoruz. her 50 ms'de bir en yüksek öncelik ve böylece periyodik olarak çalışmaya başlama.

Elbette, S zaman periyodunun eklenmesi bariz bir soruya yol açar: S neye ayarlanmalıdır? Saygın bir sistem araştırmacısı [O11] olan John Ousterhout, sistemlerdeki bu tür değerleri voodoo sabitleri olarak adlandırdı, çünkü onları doğru bir şekilde ayarlamak için bir çeşit kara büyü gerektiriyor gibiydiler. Ne yazık ki, S'de bu tat var. Çok yükseğe ayarlanırsa, uzun süreli işler aç kalabilir; çok düşüktür ve etkileşimli işler CPU'dan uygun bir pay alamayabilir.



Şekil 8.7: Daha Düşük Öncelik, Daha Uzun Kuanta

8.4 Deneme #3: Daha iyi Hesaplama

Şimdi çözmemiz gereken bir sorunuz daha var: programlayıcımızın oynamasını nasıl önleyebiliriz? Buradaki gerçek suçlu, tahmin etmiş olabileceğiniz gibi, zaman dilimi sona ermeden önce CPU'dan vazgeçerek bir işin önceliğini korumasına izin veren Kural 4a ve 4b'dir. Yani ne yapmalıyız?

- Buradaki çözüm, her seferinde CPU zamanını daha iyi hesaplamaktır. MLFQ seviyesi. Bir sürecin belirli bir seviyede ne kadar zaman dilimini kullandığını unutmak yerine, programlayıcı takip etmelidir; Bir işlem payını kullandıktan sonra bir sonraki öncelik sırasına indirilir. Zaman dilimini tek bir uzun patlamada mı yoksa çok sayıda küçük dilimde mi kullandığı önemli değil. Böylece, Kural 4a ve 4b'yi aşağıdaki tek kurala göre yeniden yazıyoruz:
- **Kural 4:** Bir iş, belirli bir düzeyde kendisine ayrılan zamanı kullandığında (CPU'dan kaç kez vazgeçtiğine bakılmaksızın), önceliği azaltılır (yani, bir sıra aşağı iner).
- Bir örneğe bakalım. Şekil 8.6 (sayfa 7), bir iş yükü zamanlayıcıyı eski Kural 4a ve 4b (solda) ve ayrıca yeni oyun önleme Kuralı 4 ile oynamaya çalıştığında ne olduğunu gösterir. G/Ç, bir zaman dilimi sona ermeden hemen önce ve dolayısıyla CPU zamanına hakim olur. Bu tür korumalar uygulandığında, işlemin G/Ç davranışından bağımsız olarak, yavaş yavaş sıralarda ilerler ve bu nedenle CPU'dan haksız bir pay alamaz.

8.5 MLFQ Ayarı ve Diğer Sorunlar

MLFQ planlamasıyla ilgili birkaç sorun daha ortaya çıkıyor. Büyük bir soru, böyle bir zamanlayıcının nasıl parametrelendirileceğidir. Örneğin, kaç kuyruk olmalıdır? Kuyruk başına zaman dilimi ne kadar büyük olmalıdır? Açlıktan kaçınmak ve davranış değişikliklerini hesaba katmak için öncelik ne sıklıkla artırılmalıdır? Bu soruların kolay yanıtları yoktur ve bu nedenle, yalnızca iş yükleri ve ardından programlayıcının

SCHEDULING:

THE MULTI-LEVEL FEEDBACK QUEUE

10

ayarlanmasıyla ilgili bazı deneyimler tatmin edici bir dengeye yol açacaktır.

**İpucu: VOO-DOO SABİTLERİNDEN KAÇININ
(OUSTERHOUT YASASI)**

Voo-doo sabitlerinden kaçınmak, mümkün olduğunda iyi bir fikirdir. Ne yazık ki, yukarıdaki örnekte olduğu gibi, genellikle zordur. Sistemin iyi bir değer öğrenmesi sağlanmaya çalışılabilir, ancak bu da kolay değildir. Sık görülen sonuç: deneyimli bir yöneticinin bir şeyler düzgün çalışmadığında ince ayar yapabileceği varsayılan parametre değerleriyle dolu bir yapılandırma dosyası. Tahmin edebileceğiniz gibi, bunlar genellikle değiştirilmeden bırakılır ve bu nedenle, varsayılanların sahada iyi çalıştığını ummak zorunda kalırız. Bu ipucu size eski OS profesörümüz John Ousterhout tarafından getirildi ve bu nedenle ona Ousterhout Yasası diyoruz.

Örneğin, çoğu MLFQ varyantı, farklı kuyruklarda değişen zaman dilimi uzunluğuna izin verir. Yüksek öncelikli sıralara genellikle kısa zaman dilimleri verilir; sonuçta etkileşimli işlerden oluşurlar ve bu nedenle aralarında hızlı bir şekilde geçiş yapmak mantıklıdır (örneğin, 10 veya daha az milisaniye). Düşük öncelikli kuyruklar ise aksine, CPU'ya bağlı uzun süre devam eden işler içerir; bu nedenle, daha uzun zaman dilimleri iyi çalışır (örneğin, 100 saniye). Şekil 8.7 (sayfa 8), iki işin en yüksek kuyrukta 20 ms (10 ms zaman dilimiyle), 40 ms ortada (20 ms zaman dilimi) ve 40 ms kuyrukta çalıştığı bir örneği gösterir. en düşük zaman dilimi.

Zaman Paylaşımlı planlama sınıfı veya TS olan Solaris MLFQ uygulamasının yapılandırılması özellikle kolaydır; bir sürecin önceliğinin kullanım ömrü boyunca tam olarak nasıl değiştirildiğini, her bir zaman diliminin ne kadar sürdüğünü ve bir işin önceliğinin ne sıklıkta artırılacağını [AD00] belirleyen bir dizi tablo sağlar; bir yönetici, programlayıcının farklı şekillerde davranmasını sağlamak için bu tabloyla uğraşabilir. Tablo için varsayılan değerler, 20 milisaniyeden (en yüksek öncelik) birkaç yüz milisaniyeye (en düşük) yavaş yavaş artan zaman dilimi uzunlukları ve her 1 saniyede bir artan önceliklerle 60 sıradır.

Diğer MLFQ planlayıcıları bir tablo veya bu bölümde açıklanan kesin kuralları kullanmaz; bunun yerine öncelikleri matematiksel formüller kullanarak ayarlarlar. Örneğin, FreeBSD zamanlayıcı (sürüm 4.3), bir işin mevcut öncelik seviyesini hesaplamak için, işlemin ne kadar CPU kullandığını [LM+89] temel alarak bir formül kullanır; ayrıca kullanım zamanla azalır ve istenen öncelik artışı burada açıklanandan farklı bir şekilde sağlanır. Bu tür bozunma-kullanım algoritmalarına ve özelliklerine [E95] ilişkin mükemmel bir genel bakış için Epema'nın makalesine bakın.

Son olarak, birçok programlayıcının karşılaşılabileceğiniz birkaç özelliği daha vardır. Örneğin, bazı planlayıcılar en yüksek öncelik düzeylerini işletim sistemi çalışması için ayırır; bu nedenle, tipik kullanıcı işleri hiçbir zaman sistemdeki en yüksek öncelik düzeylerini elde edemez. Bazı sistemler, önceliklerin belirlenmesine yardımcı olmak için bazı kullanıcı tavsiyelerine de izin verir; örneğin nice komut satırı yardımcı programını kullanarak bir işin önceliğini (biraz) artırabilir veya azaltabilir ve böylece herhangi bir zamanda çalışma şansını artırabilir veya azaltabilirsiniz. Daha fazlası için man sayfasına bakın.

İpucu: MÜMKÜN OLDUĞUNDA TAVSİYE KULLANIN

İşletim sistemi, sistemin her bir işlemi için neyin en iyi olduğunu nadiren bildiğinden, kullanıcıların veya yöneticilerin işletim sistemine bazı ipuçları vermesine olanak tanıyan arabirimler sağlamak genellikle yararlıdır. İşletim sisteminin buna dikkat etmesi gerekmediğinden, daha iyi bir karar vermek için tavsiyeyi dikkate alabileceğinden, genellikle bu tür ipuçlarına tavsiye diyoruz. Bu tür ipuçları, programlayıcı (ör. nice ile), bellek yöneticisi (ör. madvise) ve dosya sistemi (ör. bilgilendirilmiş önceden getirme ve önbellege alma [P+95]) dahil olmak üzere işletim sisteminin birçok bölümünde yararlıdır.

8.5 MLFQ: Özet

Çok Düzeyli Geri Bildirim Kuyruğu (MLFQ) olarak bilinen bir zamanlama yaklaşımı tanımladık. Umarız artık neden böyle adlandırıldığını anlamışsınızdır: birden çok sıra düzeyi vardır ve belirli bir işin önceliğini belirlemek için geri bildirim kullanır. Tarih onun kılavuzudur: işlerin zaman içinde nasıl davrandığına dikkat edin ve onlara buna göre davranın.

- Bölüm boyunca yayılmış rafine edilmiş MLFQ kuralları seti, izleme zevkiniz için burada yeniden oluşturulmuştur:
- **Kural 1:** Eğer $\text{Priority}(A) > \text{Priority}(B)$, A çalışır (B Çalışmaz).
- **Kural 2:** Eğer $\text{Priority}(A) = \text{Priority}(B)$, A ve B, verilen kuyruğun zaman dilimi (kuantum uzunluğu) kullanılarak sıralı bir şekilde çalışır.
- **Kural 3:** Bir iş sisteme girdiğinde en üstte yer alır. öncelik (en üst sıra).
- **Kural 4:** Bir iş, belirli bir düzeyde kendisine ayrılan zamanı kullandığında (CPU'dan kaç kez vazgeçtiğine bakılmaksızın), önceliği azaltılır (yani, bir sıra aşağı iner).
- **Kural 5:** Belirli bir S süresinden sonra, sistemdeki tüm işleri taşıyın en üst sıraya.

MLFQ şu nedenle ilgi çekicidir: Bir işin doğası hakkında önceden bilgi talep etmek yerine, bir işin yürütülmesini gözlemler ve ona göre öncelik verir. Bu şekilde, her iki dünyanın da en iyisini elde etmeyi başarır: kısa süreli etkileşimli işler için mükemmel genel performans (SJF/STCF'ye benzer) sağlayabilir ve adildir ve uzun süreli CPU yoğun iş yükleri için ilerleme sağlar. Bu nedenle, BSD UNIX türevleri [LM+89, B86], Solaris [M06] ve Windows NT ve sonraki Windows işletim sistemleri [CS97] dahil birçok sistem, temel zamanlayıcı olarak bir MLFQ biçimi kullanır.

References

- [AD00] "Multilevel Feedback Queue Scheduling in Solaris" by Andrea Arpaci-Dusseau. Available: <http://www.ostep.org/Citations/notes-solaris.pdf>. A great short set of notes by one of the authors on the details of the Solaris scheduler. OK, we are probably biased in this description, but the notes are pretty darn good.
- [B86] "The Design of the UNIX Operating System" by M.J. Bach. Prentice-Hall, 1986. One of the classic old books on how a real UNIX operating system is built; a definite must-read for kernel hackers.
- [C+62] "An Experimental Time-Sharing System" by F. J. Corbato, M. M. Daggett, R. C. Daley. IFIPS 1962. A bit hard to read, but the source of many of the first ideas in multi-level feedback scheduling. Much of this later went into Multics, which one could argue was the most influential operating system of all time.
- [CS97] "Inside Windows NT" by Helen Custer and David A. Solomon. Microsoft Press, 1997. The NT book, if you want to learn about something other than UNIX. Of course, why would you? OK, we're kidding; you might actually work for Microsoft some day you know.
- [E95] "An Analysis of Decay-Usage Scheduling in Multiprocessors" by D.H.J. Epema. SIG- METRICS '95. A nice paper on the state of the art of scheduling back in the mid 1990s, including a good overview of the basic approach behind decay-usage schedulers.
- [LM+89] "The Design and Implementation of the 4.3BSD UNIX Operating System" by S.J. Leffler, M.K. McKusick, M.J. Karels, J.S. Quarterman. Addison-Wesley, 1989. Another OS classic, written by four of the main people behind BSD. The later versions of this book, while more up to date, don't quite match the beauty of this one.
- [M06] "Solaris Internals: Solaris 10 and OpenSolaris Kernel Architecture" by Richard McDougall. Prentice-Hall, 2006. A good book about Solaris and how it works.
- [O11] "John Ousterhout's Home Page" by John Ousterhout. www.stanford.edu/~ouster/. The home page of the famous Professor Ousterhout. The two co-authors of this book had the pleasure of taking graduate operating systems from Ousterhout while in graduate school; indeed, this is where the two co-authors got to know each other, eventually leading to marriage, kids, and even this book. Thus, you really can blame Ousterhout for this entire mess you're in.
- [P+95] "Informed Prefetching and Caching" by R.H. Patterson, G.A. Gibson, E. Ginting, D. Stodolsky, J. Zelenka. SOSP '95, Copper Mountain, Colorado, October 1995. A fun paper about some very cool ideas in file systems, including how applications can give the OS advice about what files it is accessing and how it plans to access them.
- [Y+18] "Principled Schedulability Analysis for Distributed Storage Systems using Thread Architecture Models" by Suli Yang, Jing Liu, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau. OSDI '18, San Diego, California. A recent work of our group that demonstrates the difficulty of scheduling I/O requests within modern distributed storage systems such as Hive/HDFS, Cassandra, MongoDB, and Riak. Without care, a single user might be able to monopolize system resources.

Ödev (Similasyon)

Bu program, mlfq.py, bu bölümde sunulan MLFQ zamanlayıcısının nasıl davrandığını görmenizi sağlar. Ayrıntılar için README sayfasına bakın.

Questions

1. Yalnızca iki iş ve iki sıra ile rastgele oluşturulmuş birkaç sorunu çalıştırın; her biri için MLFQ yürütme izini hesaplayın. Her işin süresini sınırlayarak ve G/Ç'leri kapatarak hayatınızı kolaylaştırın.
2. Bölümdeki örneklerin her birini yeniden oluşturmak için programlayıcıyı nasıl çalıştırırsınız?
3. Zamanlayıcı parametrelerini, döngüsel bir zamanlayıcı gibi davranacak şekilde nasıl yapılandırırsınız?
4. İki iş ve zamanlayıcı parametresiyle bir iş yükü oluşturun, böylece bir iş, zamanlayıcıyı oynatmak ve belirli bir zaman aralığında CPU'nun %99'unu elde etmek için eski Kural 4a ve 4b'den (-S bayrağıyla etkinleştirildi) yararlanır.
5. En yüksek kuyruğunda kuantum uzunluğu 10 ms olan bir sistem verildiğinde, tek bir uzun süreli (ve potansiyel olarak-) aç iş, CPU'nun en az %5'ini alıyor mu?
6. Planlamada ortaya çıkan bir soru, G/Ç'yi henüz bitirmiş bir işin kuyruğun hangi ucuna ekleneceğidir; -I bayrağı, bu zamanlama simülatörü için bu davranışı değiştirir. Bazı iş yükleriyle oynayın ve bu bayrağın etkisini görüp göremediğinizi görün.

Cevaplar

1. Soru

Yalnızca iki iş ve iki sıra ile rastgele oluşturulmuş birkaç sorunu çalıştırın; her biri için MLFQ yürütme izini hesaplayın. Her işin süresini sınırlayarak ve G/Ç'leri kapatarak hayatınızı kolaylaştırın.

İlk olarak mlfq.py dosyası visual studio code programında çalıştırılıp kod bloğundaki “PARSE ARGUMENTS” kısmında değerler, seed=0 numQueues=2 numJobs=2 maxlen=20 maxio=0 olarak değiştirilip çalıştırılmıştır.

```

38 parser.add_option('-s', '--seed', help='the random seed',
39                  default=0, action='store', type='int', dest='seed')
40 parser.add_option('-n', '--numQueues',
41                  help='number of queues in MLFQ (if not using -Q)',
42                  default=2, action='store', type='int', dest='numQueues')
43 parser.add_option('-q', '--quantum', help='length of time slice (if not using -Q)',
44                  default=10, action='store', type='int', dest='quantum')
45 parser.add_option('-a', '--allotment', help='length of allotment (if not using -A)',
46                  default=1, action='store', type='int', dest='allotment')
47 parser.add_option('-Q', '--quantumList',
48                  help='length of time slice per queue level, specified as ' + \
49                  'x,y,z,... where x is the quantum length for the highest ' + \
50                  'priority queue, y the next highest, and so forth',
51                  default='', action='store', type='string', dest='quantumList')
52 parser.add_option('-A', '--allotmentList',
53                  help='length of time allotment per queue level, specified as ' + \
54                  'x,y,z,... where x is the # of time slices for the highest ' + \
55                  'priority queue, y the next highest, and so forth',
56                  default='', action='store', type='string', dest='allotmentList')
57 parser.add_option('-j', '--numJobs', default=2, help='number of jobs in the system',
58                  action='store', type='int', dest='numJobs')
59 parser.add_option('-m', '--maxlen', default=20, help='max run-time of a job ' +
60                  '(if randomly generating)', action='store', type='int',
61                  dest='maxlen')
62 parser.add_option('-M', '--maxio', default=2,
63                  help='max I/O frequency of a job (if randomly generating)',
64                  action='store', type='int', dest='maxio')
65 parser.add_option('-B', '--boost', default=0,
66                  help='how often to boost the priority of all jobs back to ' +
67                  'high priority', action='store', type='int', dest='boost')
68 parser.add_option('-i', '--ioTime', default=5,
69                  help='how long an I/O should last (fixed constant)',
70                  action='store', type='int', dest='ioTime')
71 parser.add_option('-S', '--stay', default=False,
72                  help='reset and stay at same priority level when issuing I/O',
73                  action='store_true', dest='stay')

```

16

Ardından terminalden " \$ python mlfq.py -s 0 -n 2 -j 2 -m 20 -M 0 -c " komutu ile çalıştırılıp istenilen sonuç elde edilmiştir.

2. Soru

Bölümdeki örneklerin her birini yeniden oluşturmak için programlayıcıyı nasıl çalıştırırsınız?

İkinci soruda da birinci sorudaki "PARSE AGRUMENTS" kısmında değişiklikler yapılmıştır aşağıda her başlık için terminal üzerindeki giriş değerleri ve kodun çıktısı verilmiştir.

Örnek 1: Uzun süren tek bir iş

Kod çıktısı metin olarak şu şekildedir.

Here is the list of inputs:

```

OPTIONS jobs 1
OPTIONS queues 3
OPTIONS allotments for queue 2 is 1
OPTIONS quantum length for queue 2 is 10
OPTIONS allotments for queue 1 is 1
OPTIONS quantum length for queue 1 is 10
OPTIONS allotments for queue 0 is 1
OPTIONS quantum length for queue 0 is 10
OPTIONS boost 0
OPTIONS ioTime 5
OPTIONS stayAfterIO False
OPTIONS iobump False

```

For each job, three defining characteristics are given:

startTime : at what time does the job enter the system
 runTime : the total CPU time needed by the job to finish
 ioFreq : every ioFreq time units, the job issues an I/O
 (the I/O takes ioTime units to complete)

Job List:

Job 0: startTime 0 - runTime 200 - ioFreq 0

Execution Trace:

```

[ time 0 ] JOB BEGINS by JOB 0
[ time 0 ] Run JOB 0 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 199 (of 200) ]
[ time 1 ] Run JOB 0 at PRIORITY 2 [ TICKS 8 ALLOT 1 TIME 198 (of 200) ]
[ time 2 ] Run JOB 0 at PRIORITY 2 [ TICKS 7 ALLOT 1 TIME 197 (of 200) ]
[ time 3 ] Run JOB 0 at PRIORITY 2 [ TICKS 6 ALLOT 1 TIME 196 (of 200) ]
[ time 4 ] Run JOB 0 at PRIORITY 2 [ TICKS 5 ALLOT 1 TIME 195 (of 200) ]
[ time 5 ] Run JOB 0 at PRIORITY 2 [ TICKS 4 ALLOT 1 TIME 194 (of 200) ]
[ time 6 ] Run JOB 0 at PRIORITY 2 [ TICKS 3 ALLOT 1 TIME 193 (of 200) ]
[ time 7 ] Run JOB 0 at PRIORITY 2 [ TICKS 2 ALLOT 1 TIME 192 (of 200) ]
[ time 8 ] Run JOB 0 at PRIORITY 2 [ TICKS 1 ALLOT 1 TIME 191 (of 200) ]
[ time 9 ] Run JOB 0 at PRIORITY 2 [ TICKS 0 ALLOT 1 TIME 190 (of 200) ]
[ time 10 ] Run JOB 0 at PRIORITY 1 [ TICKS 9 ALLOT 1 TIME 189 (of 200) ]
[ time 11 ] Run JOB 0 at PRIORITY 1 [ TICKS 8 ALLOT 1 TIME 188 (of 200) ]
[ time 12 ] Run JOB 0 at PRIORITY 1 [ TICKS 7 ALLOT 1 TIME 187 (of 200) ]
[ time 13 ] Run JOB 0 at PRIORITY 1 [ TICKS 6 ALLOT 1 TIME 186 (of 200) ]
[ time 14 ] Run JOB 0 at PRIORITY 1 [ TICKS 5 ALLOT 1 TIME 185 (of 200) ]
[ time 15 ] Run JOB 0 at PRIORITY 1 [ TICKS 4 ALLOT 1 TIME 184 (of 200) ]
[ time 16 ] Run JOB 0 at PRIORITY 1 [ TICKS 3 ALLOT 1 TIME 183 (of 200) ]

```

SCHEDULING:

THE MULTI-LEVEL FEEDBACK QUEUE

18

```
[ time 17 ] Run JOB 0 at PRIORITY 1 [ TICKS 2 ALLOT 1 TIME 182 (of 200) ]
[ time 18 ] Run JOB 0 at PRIORITY 1 [ TICKS 1 ALLOT 1 TIME 181 (of 200) ]
[ time 19 ] Run JOB 0 at PRIORITY 1 [ TICKS 0 ALLOT 1 TIME 180 (of 200) ]
[ time 20 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 179 (of 200) ]
[ time 21 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 178 (of 200) ]
[ time 22 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 177 (of 200) ]
[ time 23 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 176 (of 200) ]
[ time 24 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 175 (of 200) ]
[ time 25 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 174 (of 200) ]
[ time 26 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 173 (of 200) ]
[ time 27 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 172 (of 200) ]
[ time 28 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 171 (of 200) ]
[ time 29 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 170 (of 200) ]
[ time 30 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 169 (of 200) ]
[ time 31 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 168 (of 200) ]
[ time 32 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 167 (of 200) ]
[ time 33 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 166 (of 200) ]
[ time 34 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 165 (of 200) ]
[ time 35 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 164 (of 200) ]
[ time 36 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 163 (of 200) ]
[ time 37 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 162 (of 200) ]
[ time 38 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 161 (of 200) ]
[ time 39 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 160 (of 200) ]
[ time 40 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 159 (of 200) ]
[ time 41 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 158 (of 200) ]
[ time 42 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 157 (of 200) ]
[ time 43 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 156 (of 200) ]
[ time 44 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 155 (of 200) ]
[ time 45 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 154 (of 200) ]
[ time 46 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 153 (of 200) ]
[ time 47 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 152 (of 200) ]
[ time 48 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 151 (of 200) ]
[ time 49 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 150 (of 200) ]
[ time 50 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 149 (of 200) ]
[ time 51 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 148 (of 200) ]
[ time 52 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 147 (of 200) ]
[ time 53 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 146 (of 200) ]
[ time 54 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 145 (of 200) ]
[ time 55 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 144 (of 200) ]
[ time 56 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 143 (of 200) ]
[ time 57 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 142 (of 200) ]
[ time 58 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 141 (of 200) ]
[ time 59 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 140 (of 200) ]
[ time 60 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 139 (of 200) ]
[ time 61 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 138 (of 200) ]
[ time 62 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 137 (of 200) ]
[ time 63 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 136 (of 200) ]
[ time 64 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 135 (of 200) ]
[ time 65 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 134 (of 200) ]
[ time 66 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 133 (of 200) ]
[ time 67 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 132 (of 200) ]
```

SCHEDULING:

THE MULTI-LEVEL FEEDBACK QUEUE

19

[time 68] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 131 (of 200)]
 [time 69] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 130 (of 200)]
 [time 70] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 129 (of 200)]
 [time 71] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 128 (of 200)]
 [time 72] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 127 (of 200)]
 [time 73] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 126 (of 200)]
 [time 74] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 125 (of 200)]
 [time 75] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 124 (of 200)]
 [time 76] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 123 (of 200)]
 [time 77] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 122 (of 200)]
 [time 78] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 121 (of 200)]
 [time 79] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 120 (of 200)]
 [time 80] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 119 (of 200)]
 [time 81] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 118 (of 200)]
 [time 82] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 117 (of 200)]
 [time 83] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 116 (of 200)]
 [time 84] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 115 (of 200)]
 [time 85] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 114 (of 200)]
 [time 86] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 113 (of 200)]
 [time 87] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 112 (of 200)]
 [time 88] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 111 (of 200)]
 [time 89] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 110 (of 200)]
 [time 90] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 109 (of 200)]
 [time 91] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 108 (of 200)]
 [time 92] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 107 (of 200)]
 [time 93] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 106 (of 200)]
 [time 94] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 105 (of 200)]
 [time 95] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 104 (of 200)]
 [time 96] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 103 (of 200)]
 [time 97] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 102 (of 200)]
 [time 98] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 101 (of 200)]
 [time 99] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 100 (of 200)]
 [time 100] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 99 (of 200)]
 [time 101] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 98 (of 200)]
 [time 102] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 97 (of 200)]
 [time 103] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 96 (of 200)]
 [time 104] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 95 (of 200)]
 [time 105] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 94 (of 200)]
 [time 106] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 93 (of 200)]
 [time 107] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 92 (of 200)]
 [time 108] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 91 (of 200)]
 [time 109] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 90 (of 200)]
 [time 110] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 89 (of 200)]
 [time 111] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 88 (of 200)]
 [time 112] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 87 (of 200)]
 [time 113] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 86 (of 200)]
 [time 114] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 85 (of 200)]
 [time 115] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 84 (of 200)]
 [time 116] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 83 (of 200)]
 [time 117] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 82 (of 200)]
 [time 118] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 81 (of 200)]

SCHEDULING:

THE MULTI-LEVEL FEEDBACK QUEUE

20

[time 119] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 80 (of 200)]
 [time 120] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 79 (of 200)]
 [time 121] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 78 (of 200)]
 [time 122] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 77 (of 200)]
 [time 123] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 76 (of 200)]
 [time 124] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 75 (of 200)]
 [time 125] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 74 (of 200)]
 [time 126] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 73 (of 200)]
 [time 127] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 72 (of 200)]
 [time 128] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 71 (of 200)]
 [time 129] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 70 (of 200)]
 [time 130] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 69 (of 200)]
 [time 131] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 68 (of 200)]
 [time 132] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 67 (of 200)]
 [time 133] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 66 (of 200)]
 [time 134] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 65 (of 200)]
 [time 135] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 64 (of 200)]
 [time 136] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 63 (of 200)]
 [time 137] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 62 (of 200)]
 [time 138] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 61 (of 200)]
 [time 139] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 60 (of 200)]
 [time 140] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 59 (of 200)]
 [time 141] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 58 (of 200)]
 [time 142] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 57 (of 200)]
 [time 143] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 56 (of 200)]
 [time 144] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 55 (of 200)]
 [time 145] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 54 (of 200)]
 [time 146] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 53 (of 200)]
 [time 147] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 52 (of 200)]
 [time 148] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 51 (of 200)]
 [time 149] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 50 (of 200)]
 [time 150] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 49 (of 200)]
 [time 151] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 48 (of 200)]
 [time 152] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 47 (of 200)]
 [time 153] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 46 (of 200)]
 [time 154] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 45 (of 200)]
 [time 155] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 44 (of 200)]
 [time 156] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 43 (of 200)]
 [time 157] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 42 (of 200)]
 [time 158] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 41 (of 200)]
 [time 159] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 40 (of 200)]
 [time 160] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 39 (of 200)]
 [time 161] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 38 (of 200)]
 [time 162] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 37 (of 200)]
 [time 163] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 36 (of 200)]
 [time 164] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 35 (of 200)]
 [time 165] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 34 (of 200)]
 [time 166] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 33 (of 200)]
 [time 167] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 32 (of 200)]
 [time 168] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 31 (of 200)]
 [time 169] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 30 (of 200)]

SCHEDULING:

THE MULTI-LEVEL FEEDBACK QUEUE

21

```
[ time 170 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 29 (of 200) ]
[ time 171 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 28 (of 200) ]
[ time 172 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 27 (of 200) ]
[ time 173 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 26 (of 200) ]
[ time 174 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 25 (of 200) ]
[ time 175 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 24 (of 200) ]
[ time 176 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 23 (of 200) ]
[ time 177 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 22 (of 200) ]
[ time 178 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 21 (of 200) ]
[ time 179 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 20 (of 200) ]
[ time 180 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 19 (of 200) ]
[ time 181 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 18 (of 200) ]
[ time 182 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 17 (of 200) ]
[ time 183 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 16 (of 200) ]
[ time 184 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 15 (of 200) ]
[ time 185 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 14 (of 200) ]
[ time 186 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 13 (of 200) ]
[ time 187 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 12 (of 200) ]
[ time 188 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 11 (of 200) ]
[ time 189 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 10 (of 200) ]
[ time 190 ] Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 200) ]
[ time 191 ] Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 8 (of 200) ]
[ time 192 ] Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 7 (of 200) ]
[ time 193 ] Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 6 (of 200) ]
[ time 194 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 200) ]
[ time 195 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 200) ]
[ time 196 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 200) ]
[ time 197 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 200) ]
[ time 198 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 200) ]
[ time 199 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 200) ]
[ time 200 ] FINISHED JOB 0
```

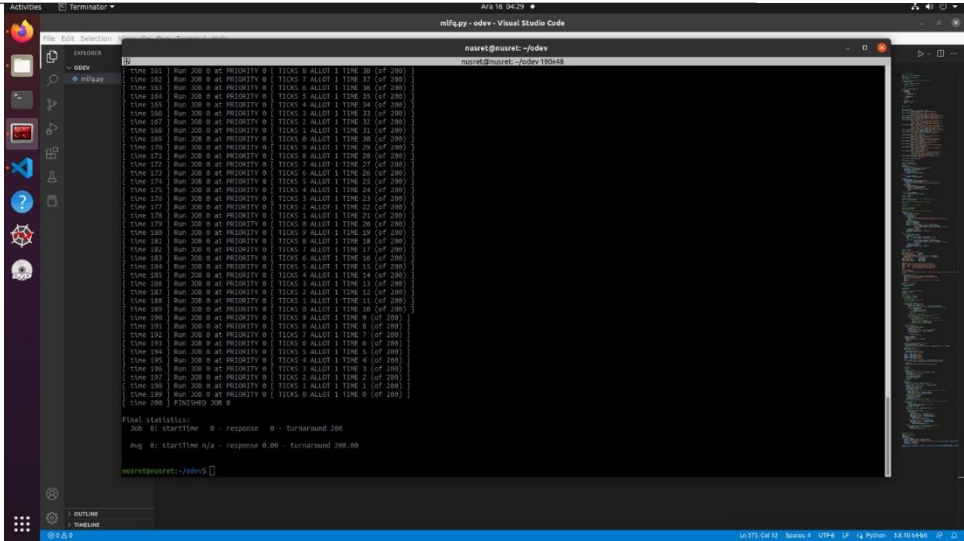
Final statistics:

Job 0: startTime 0 - response 0 - turnaround 200

Avg 0: startTime n/a - response 0.00 - turnaround 200.00

SCHEDULING: THE MULTI-LEVEL FEEDBACK QUEUE

22



```

time 161 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 36 (of 200)
time 162 Run 200 0 at PROCSERV 0 Ticks 7 ALLOTT 1 TIME 37 (of 200)
time 163 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 38 (of 200)
time 164 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 39 (of 200)
time 165 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 40 (of 200)
time 166 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 41 (of 200)
time 167 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 42 (of 200)
time 168 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 43 (of 200)
time 169 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 44 (of 200)
time 170 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 45 (of 200)
time 171 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 46 (of 200)
time 172 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 47 (of 200)
time 173 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 48 (of 200)
time 174 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 49 (of 200)
time 175 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 50 (of 200)
time 176 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 51 (of 200)
time 177 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 52 (of 200)
time 178 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 53 (of 200)
time 179 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 54 (of 200)
time 180 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 55 (of 200)
time 181 Run 200 0 at PROCSERV 0 Ticks 7 ALLOTT 1 TIME 56 (of 200)
time 182 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 57 (of 200)
time 183 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 58 (of 200)
time 184 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 59 (of 200)
time 185 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 60 (of 200)
time 186 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 61 (of 200)
time 187 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 62 (of 200)
time 188 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 63 (of 200)
time 189 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 64 (of 200)
time 190 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 65 (of 200)
time 191 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 66 (of 200)
time 192 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 67 (of 200)
time 193 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 68 (of 200)
time 194 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 69 (of 200)
time 195 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 70 (of 200)
time 196 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 71 (of 200)
time 197 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 72 (of 200)
time 198 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 73 (of 200)
time 199 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 74 (of 200)
time 200 FINISHED JOB 0

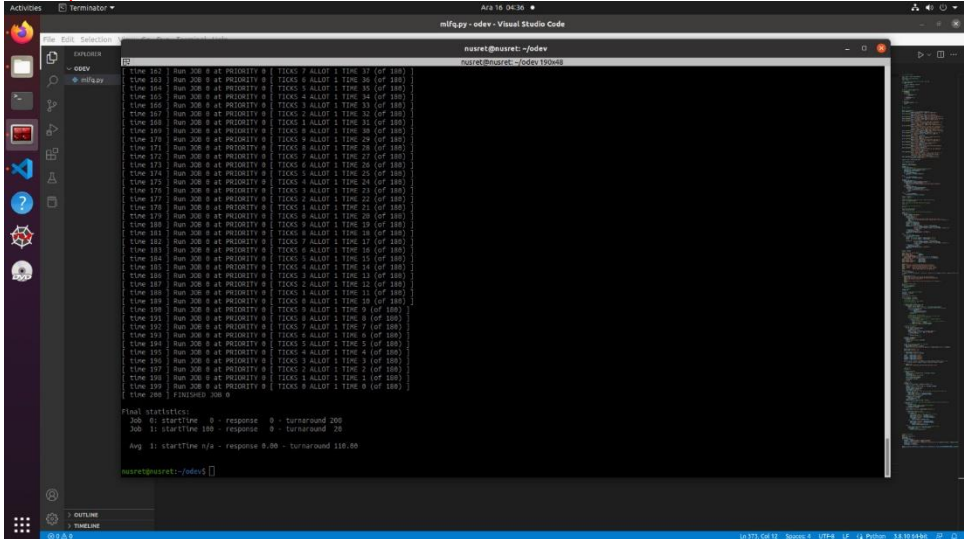
Final statistics:
JOB 0: startTime 0 - response 0 - turnaround 200
Avg 0: startTime N/A - response 0.00 - turnaround 200.00
msret@msret:~/mlfq$

```

Bu ve bundan sonraki örneklerde çıktının hepsi metin olarak yazdırılmamış çalıştırma kodu ve kran görüntülerine yer verilmiştir.

Örnek 2 : Birlikte etkileşimli yeni bir iş gelmesi

Terminalde “\$ python mlfq.py -n 3 -q 10 -l 0,180,0:100,20,0 -c” çalıştırılıp şu çıktı alınmıştır.



```

time 161 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 36 (of 180)
time 162 Run 200 0 at PROCSERV 0 Ticks 7 ALLOTT 1 TIME 36 (of 180)
time 163 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 37 (of 180)
time 164 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 38 (of 180)
time 165 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 39 (of 180)
time 166 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 40 (of 180)
time 167 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 41 (of 180)
time 168 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 42 (of 180)
time 169 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 43 (of 180)
time 170 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 44 (of 180)
time 171 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 45 (of 180)
time 172 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 46 (of 180)
time 173 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 47 (of 180)
time 174 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 48 (of 180)
time 175 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 49 (of 180)
time 176 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 50 (of 180)
time 177 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 51 (of 180)
time 178 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 52 (of 180)
time 179 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 53 (of 180)
time 180 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 54 (of 180)
time 181 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 55 (of 180)
time 182 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 56 (of 180)
time 183 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 57 (of 180)
time 184 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 58 (of 180)
time 185 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 59 (of 180)
time 186 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 60 (of 180)
time 187 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 61 (of 180)
time 188 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 62 (of 180)
time 189 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 63 (of 180)
time 190 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 64 (of 180)
time 191 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 65 (of 180)
time 192 Run 200 0 at PROCSERV 0 Ticks 6 ALLOTT 1 TIME 66 (of 180)
time 193 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 67 (of 180)
time 194 Run 200 0 at PROCSERV 0 Ticks 4 ALLOTT 1 TIME 68 (of 180)
time 195 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 69 (of 180)
time 196 Run 200 0 at PROCSERV 0 Ticks 2 ALLOTT 1 TIME 70 (of 180)
time 197 Run 200 0 at PROCSERV 0 Ticks 3 ALLOTT 1 TIME 71 (of 180)
time 198 Run 200 0 at PROCSERV 0 Ticks 0 ALLOTT 1 TIME 72 (of 180)
time 199 Run 200 0 at PROCSERV 0 Ticks 5 ALLOTT 1 TIME 73 (of 180)
time 200 FINISHED JOB 0

Final statistics:
JOB 0: startTime 0 - response 0 - turnaround 200
JOB 1: startTime 180 - response 0 - turnaround 20
Avg 1: startTime N/A - response 0.00 - turnaround 110.00
msret@msret:~/mlfq$

```

Şekil 8.4'deki Karma G/Ç yoğun ve CPU yoğun İş Yüğü

Terminalde “\$ python mlfq.py -n 3 -q 10 -l 0,175,0:50,25,1 -i 5 -S -c” çalıştırılıp şu çıktı

alınmıştır.

```

Line 173 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 22 (of 175) ]
Line 174 Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 21 (of 175) ]
Line 175 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 28 (of 175) ]
Line 176 IO_DONE by JOB 1
Line 177 IO_START by JOB 1
Line 178 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 19 (of 175) ]
Line 179 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 17 (of 175) ]
Line 180 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 16 (of 175) ]
Line 181 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 15 (of 175) ]
Line 182 IO_DONE by JOB 0
Line 183 IO_START by JOB 2 [ TICKS 9 ALLOT 1 TIME 2 (of 25) ]
Line 184 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 14 (of 175) ]
Line 185 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 13 (of 175) ]
Line 186 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 12 (of 175) ]
Line 187 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 11 (of 175) ]
Line 188 IO_DONE by JOB 1
Line 189 Run JOB 0 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 10 (of 175) ]
Line 190 IO_START by JOB 1
Line 191 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 0 (of 175) ]
Line 192 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 0 (of 175) ]
Line 193 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 0 (of 175) ]
Line 194 IO_DONE by JOB 1
Line 195 Run JOB 2 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 0 (of 25) ]
Line 196 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 0 (of 175) ]
Line 197 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 0 (of 175) ]
Line 198 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 0 (of 175) ]
Line 199 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 0 (of 175) ]
Line 200 FINISHED JOB 0

Final statistics:
JOB 0: startTime = 0 - response = 0 - turnaround 200
JOB 1: startTime = 0 - response = 0 - turnaround 145
Avg 1: startTime = 0 - response 0.00 - turnaround 172.50

msfqr@msfqr:~$

```

Örnek 3: Peki ya G/Ç Nedir?

Terminalde “\$ python mlfq.py -n 3 -q 10 -l 0,120,0:100,50,1:100,50,1 -i 1 -S -c” ve “\$ python mlfq.py -n 3 -q 10 -l 0,120,0:100,50,1:100,50,1 -i 1 -S -B 50 -c” çalıştırılıp şu çıktı alınmıştır.

“\$ python mlfq.py -n 3 -q 10 -l 0,120,0:100,50,1:100,50,1 -i 1 -S -c” kodunun çıktısı:

```

Line 195 Run JOB 2 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 2 (of 50) ]
Line 196 IO_START by JOB 2
Line 197 IO_DONE by JOB 1
Line 198 Run JOB 2 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 2 (of 50) ]
Line 199 IO_START by JOB 2
Line 200 Run JOB 2 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 0 (of 50) ]
Line 201 IO_DONE by JOB 1
Line 202 Run JOB 2 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 0 (of 50) ]
Line 203 FINISHED JOB 2
Line 204 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 19 (of 120) ]
Line 205 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 17 (of 120) ]
Line 206 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 16 (of 120) ]
Line 207 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 15 (of 120) ]
Line 208 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 14 (of 120) ]
Line 209 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 13 (of 120) ]
Line 210 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 12 (of 120) ]
Line 211 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 11 (of 120) ]
Line 212 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 10 (of 120) ]
Line 213 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 9 (of 120) ]
Line 214 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 8 (of 120) ]
Line 215 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 7 (of 120) ]
Line 216 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 6 (of 120) ]
Line 217 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 5 (of 120) ]
Line 218 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 4 (of 120) ]
Line 219 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 3 (of 120) ]
Line 220 FINISHED JOB 0

Final statistics:
JOB 0: startTime = 0 - response = 0 - turnaround 220
JOB 1: startTime = 0 - response = 0 - turnaround 99
JOB 2: startTime = 0 - response = 1 - turnaround 100
Avg 1: startTime = 0 - response 0.33 - turnaround 139.67

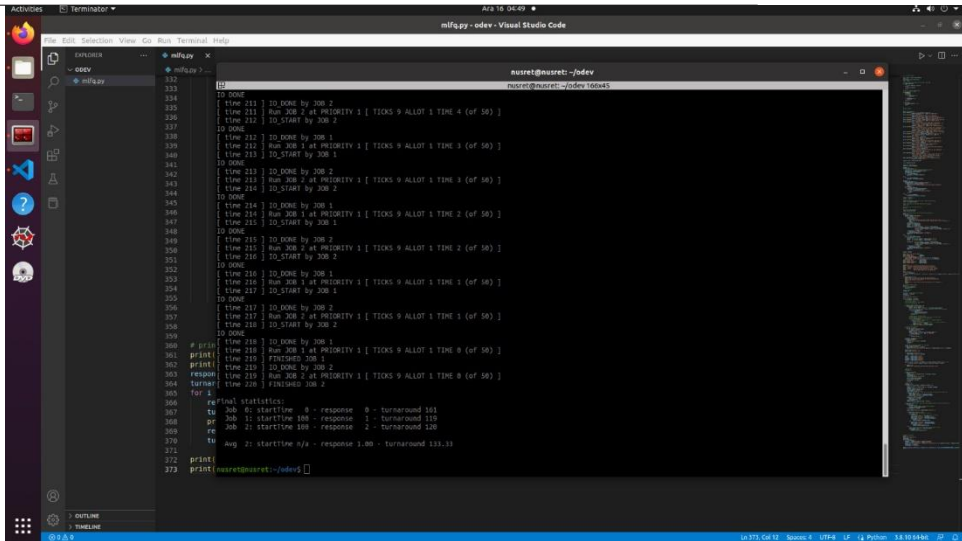
msfqr@msfqr:~$

```

“\$ python mlfq.py -n 3 -q 10 -l 0,120,0:100,50,1:100,50,1 -i 1 -S -B 50 -c” kodunun çıktısı:

SCHEDULING: THE MULTI-LEVEL FEEDBACK QUEUE

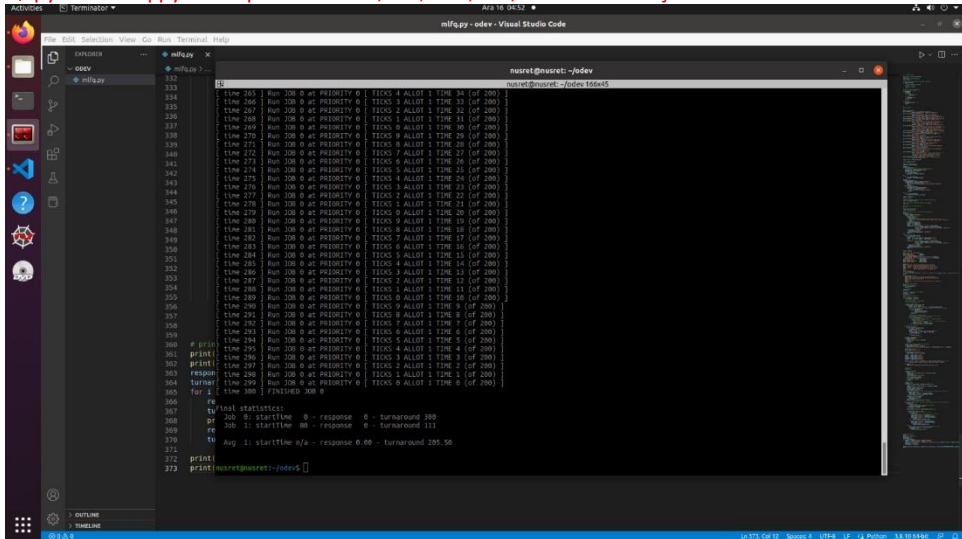
24



Şekil 8.6: (Sol) ve (Sağ) Oynama Toleransı Olmadan

Terminalde “\$ python mlfq.py -n 3 -q 10 -i 1 -S -l 0,200,0:80,100,9 -c” ve “\$ python mlfq.py -n 3 -q 10 -i 1 -l 0,200,0:80,100,9 -c” çalıştırılıp şu çıktı alınmıştır.

“\$ python mlfq.py -n 3 -q 10 -i 1 -S -l 0,200,0:80,100,9 -c” kodunun çıktısı



“\$ python mlfg.py -n 3 -q 10 -i 1 -l 0,200,0:80,100,9 -c” kodunun çıktısı

SCHEDULING: THE MULTI-LEVEL FEEDBACK QUEUE

25

```

nuser@nuser:~/dev
mlfq.py -dev -Visual Studio Code

time 277 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 22 (of 100) ]
time 278 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 23 (of 100) ]
time 279 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 24 (of 100) ]
time 280 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 25 (of 100) ]
time 281 ID_START by JOB 1.
ID DONE
time 282 ID_DONE by JOB 1
time 283 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 18 (of 100) ]
time 284 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 17 (of 100) ]
time 285 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 15 (of 100) ]
time 286 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 14 (of 100) ]
time 287 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 13 (of 100) ]
time 288 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 12 (of 100) ]
time 289 Run JOB 1 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 11 (of 100) ]
time 290 Run JOB 1 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 10 (of 100) ]
time 291 ID_START by JOB 1.
ID DONE
time 291 ID_DONE by JOB 1
time 292 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 100) ]
time 293 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 8 (of 100) ]
time 294 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 7 (of 100) ]
time 295 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 6 (of 100) ]
time 296 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 100) ]
time 297 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 100) ]
time 298 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 100) ]
time 299 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 100) ]
time 300 Run JOB 1 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 100) ]
time 301 ID_START by JOB 1.
ID DONE
time 301 ID_DONE by JOB 1
time 302 Run JOB 1 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 100) ]
time 303 FINISHED JOB 1
Final statistics:
Job 1: startTime 0 - response 0 - turnaround 272
Job 2: startTime 88 - response 8 - turnaround 223
Avg 1: startTime n/a - response 8.88 - turnaround 247.58

nuser@nuser:~/dev
271
272 print('\n Avg %2d: startTime n/a - response %.2f - turnaround %.2f % (1, float(responsesum/numJobs), float(turnaroundsum/numJobs))
273 print('\n')

```

Örnek 4 şekil 8.7 daha düşük öncelik daha uzun kuanta

Terminalde “\$ python mlfq.py -n 3 -a 2 -Q 10,20,40 -I 0,200,0:0,200,0 -c” çalıştırılıp şu çıktı alınmıştır.

```

nuser@nuser:~/dev
mlfq.py -dev -Visual Studio Code

time 368 Run JOB 0 at PRIORITY 0 [ TICKS 13 ALLOT 1 TIME 13 (of 200) ]
time 369 Run JOB 0 at PRIORITY 0 [ TICKS 12 ALLOT 1 TIME 12 (of 200) ]
time 370 Run JOB 0 at PRIORITY 0 [ TICKS 11 ALLOT 1 TIME 11 (of 200) ]
time 371 Run JOB 0 at PRIORITY 0 [ TICKS 10 ALLOT 1 TIME 10 (of 200) ]
time 372 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 200) ]
time 373 Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 8 (of 200) ]
time 374 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 7 (of 200) ]
time 375 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 6 (of 200) ]
time 376 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 200) ]
time 377 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 200) ]
time 378 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 200) ]
time 379 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 200) ]
time 380 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 200) ]
time 381 FINISHED JOB 0
time 382 Run JOB 1 at PRIORITY 0 [ TICKS 19 ALLOT 1 TIME 19 (of 200) ]
time 383 Run JOB 1 at PRIORITY 0 [ TICKS 18 ALLOT 1 TIME 18 (of 200) ]
time 384 Run JOB 1 at PRIORITY 0 [ TICKS 17 ALLOT 1 TIME 17 (of 200) ]
time 385 Run JOB 1 at PRIORITY 0 [ TICKS 16 ALLOT 1 TIME 16 (of 200) ]
time 386 Run JOB 1 at PRIORITY 0 [ TICKS 15 ALLOT 1 TIME 15 (of 200) ]
time 387 Run JOB 1 at PRIORITY 0 [ TICKS 14 ALLOT 1 TIME 14 (of 200) ]
time 388 Run JOB 1 at PRIORITY 0 [ TICKS 13 ALLOT 1 TIME 13 (of 200) ]
time 389 Run JOB 1 at PRIORITY 0 [ TICKS 12 ALLOT 1 TIME 12 (of 200) ]
time 390 Run JOB 1 at PRIORITY 0 [ TICKS 11 ALLOT 1 TIME 11 (of 200) ]
time 391 Run JOB 1 at PRIORITY 0 [ TICKS 10 ALLOT 1 TIME 10 (of 200) ]
time 392 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 200) ]
time 393 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 8 (of 200) ]
time 394 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 7 (of 200) ]
time 395 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 6 (of 200) ]
time 396 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 200) ]
time 397 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 200) ]
time 398 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 200) ]
time 399 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 200) ]
time 400 Run JOB 1 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 200) ]
time 401 FINISHED JOB 1
time 402 Run JOB 2 at PRIORITY 0 [ TICKS 19 ALLOT 1 TIME 19 (of 200) ]
time 403 Run JOB 2 at PRIORITY 0 [ TICKS 18 ALLOT 1 TIME 18 (of 200) ]
time 404 Run JOB 2 at PRIORITY 0 [ TICKS 17 ALLOT 1 TIME 17 (of 200) ]
time 405 Run JOB 2 at PRIORITY 0 [ TICKS 16 ALLOT 1 TIME 16 (of 200) ]
time 406 Run JOB 2 at PRIORITY 0 [ TICKS 15 ALLOT 1 TIME 15 (of 200) ]
time 407 Run JOB 2 at PRIORITY 0 [ TICKS 14 ALLOT 1 TIME 14 (of 200) ]
time 408 Run JOB 2 at PRIORITY 0 [ TICKS 13 ALLOT 1 TIME 13 (of 200) ]
time 409 Run JOB 2 at PRIORITY 0 [ TICKS 12 ALLOT 1 TIME 12 (of 200) ]
time 410 Run JOB 2 at PRIORITY 0 [ TICKS 11 ALLOT 1 TIME 11 (of 200) ]
time 411 Run JOB 2 at PRIORITY 0 [ TICKS 10 ALLOT 1 TIME 10 (of 200) ]
time 412 Run JOB 2 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 200) ]
time 413 Run JOB 2 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 8 (of 200) ]
time 414 Run JOB 2 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 7 (of 200) ]
time 415 Run JOB 2 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 6 (of 200) ]
time 416 Run JOB 2 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 200) ]
time 417 Run JOB 2 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 200) ]
time 418 Run JOB 2 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 200) ]
time 419 Run JOB 2 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 200) ]
time 420 Run JOB 2 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 200) ]
time 421 FINISHED JOB 2
Final statistics:
Job 0: startTime 0 - response 0 - turnaround 388
Job 1: startTime 0 - response 18 - turnaround 408
Avg 1: startTime n/a - response 5.00 - turnaround 346.66

nuser@nuser:~/dev
271
272 print('\n Avg %2d: startTime n/a - response %.2f - turnaround %.2f % (1, float(responsesum/numJobs), float(turnaroundsum/numJobs))
273 print('\n')

```

3. Soru
Zamanlayıcı parametrelerini, döngüsel bir zamanlayıcı gibi davranacak şekilde nasıl yapılandırırsınız?

Zamanlayıcı parametrelerini, döngüsel bir zamanlayıcı gibi davranması için zaman diliminin, maksimum iş uzunluğunun iş sayısına bölümünden küçük yada eşit olmalıdır.

Zaman dilimi \leq (maksimum iş uzunluğu) / (iş sayısı)

4. Soru

İki iş ve zamanlayıcı parametresiyle bir iş yükü oluşturun, böylece bir iş, zamanlayıcıyı oynatmak ve belirli bir zaman aralığında CPU'nun %99'unu elde etmek için eski Kural 4a ve 4b'den (-S bayrağıyla etkinleştirildi) yararlanır.

Terminalde “\$ python mlfq.py -n 3 -q 10 -l 0,50,0:0,50,9 -i 1 -S -c” çalıştırılıp şu çıktı alınmıştır.

Here is the list of inputs:

OPTIONS jobs 2

OPTIONS queues 3

OPTIONS allotments for queue 2 is 1

OPTIONS quantum length for queue 2 is 10

OPTIONS allotments for queue 1 is 1

OPTIONS quantum length for queue 1 is 10

OPTIONS allotments for queue 0 is 1

OPTIONS quantum length for queue 0 is 10

OPTIONS boost 0

OPTIONS ioTime 1

OPTIONS stayAfterIO True

OPTIONS iobump False

For each job, three defining characteristics are given:

startTime : at what time does the job enter the system

runTime : the total CPU time needed by the job to finish

ioFreq : every ioFreq time units, the job issues an I/O
(the I/O takes ioTime units to complete)

Job List:

Job 0: startTime 0 - runTime 50 - ioFreq 0

Job 1: startTime 0 - runTime 50 - ioFreq 9

Execution Trace:

[time 0] JOB BEGINS by JOB 0

[time 0] JOB BEGINS by JOB 1

[time 0] Run JOB 0 at PRIORITY 2 [TICKS 9 ALLOT 1 TIME 49 (of 50)]

[time 1] Run JOB 0 at PRIORITY 2 [TICKS 8 ALLOT 1 TIME 48 (of 50)]

[time 2] Run JOB 0 at PRIORITY 2 [TICKS 7 ALLOT 1 TIME 47 (of 50)]

[time 3] Run JOB 0 at PRIORITY 2 [TICKS 6 ALLOT 1 TIME 46 (of 50)]

[time 4] Run JOB 0 at PRIORITY 2 [TICKS 5 ALLOT 1 TIME 45 (of 50)]

[time 5] Run JOB 0 at PRIORITY 2 [TICKS 4 ALLOT 1 TIME 44 (of 50)]

[time 6] Run JOB 0 at PRIORITY 2 [TICKS 3 ALLOT 1 TIME 43 (of 50)]

[time 7] Run JOB 0 at PRIORITY 2 [TICKS 2 ALLOT 1 TIME 42 (of 50)]

```

[ time 8 ] Run JOB 0 at PRIORITY 2 [ TICKS 1 ALLOT 1 TIME 41 (of 50) ]
[ time 9 ] Run JOB 0 at PRIORITY 2 [ TICKS 0 ALLOT 1 TIME 40 (of 50) ]
[ time 10 ] Run JOB 1 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 49 (of 50) ]
[ time 11 ] Run JOB 1 at PRIORITY 2 [ TICKS 8 ALLOT 1 TIME 48 (of 50) ]
[ time 12 ] Run JOB 1 at PRIORITY 2 [ TICKS 7 ALLOT 1 TIME 47 (of 50) ]
[ time 13 ] Run JOB 1 at PRIORITY 2 [ TICKS 6 ALLOT 1 TIME 46 (of 50) ]
[ time 14 ] Run JOB 1 at PRIORITY 2 [ TICKS 5 ALLOT 1 TIME 45 (of 50) ]
[ time 15 ] Run JOB 1 at PRIORITY 2 [ TICKS 4 ALLOT 1 TIME 44 (of 50) ]
[ time 16 ] Run JOB 1 at PRIORITY 2 [ TICKS 3 ALLOT 1 TIME 43 (of 50) ]
[ time 17 ] Run JOB 1 at PRIORITY 2 [ TICKS 2 ALLOT 1 TIME 42 (of 50) ]
[ time 18 ] Run JOB 1 at PRIORITY 2 [ TICKS 1 ALLOT 1 TIME 41 (of 50) ]
[ time 19 ] IO_START by JOB 1
IO DONE
[ time 19 ] Run JOB 0 at PRIORITY 1 [ TICKS 9 ALLOT 1 TIME 39 (of 50) ]
[ time 20 ] IO_DONE by JOB 1
[ time 20 ] Run JOB 1 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 40 (of 50) ]
[ time 21 ] Run JOB 1 at PRIORITY 2 [ TICKS 8 ALLOT 1 TIME 39 (of 50) ]
[ time 22 ] Run JOB 1 at PRIORITY 2 [ TICKS 7 ALLOT 1 TIME 38 (of 50) ]
[ time 23 ] Run JOB 1 at PRIORITY 2 [ TICKS 6 ALLOT 1 TIME 37 (of 50) ]
[ time 24 ] Run JOB 1 at PRIORITY 2 [ TICKS 5 ALLOT 1 TIME 36 (of 50) ]
[ time 25 ] Run JOB 1 at PRIORITY 2 [ TICKS 4 ALLOT 1 TIME 35 (of 50) ]
[ time 26 ] Run JOB 1 at PRIORITY 2 [ TICKS 3 ALLOT 1 TIME 34 (of 50) ]
[ time 27 ] Run JOB 1 at PRIORITY 2 [ TICKS 2 ALLOT 1 TIME 33 (of 50) ]
[ time 28 ] Run JOB 1 at PRIORITY 2 [ TICKS 1 ALLOT 1 TIME 32 (of 50) ]
[ time 29 ] IO_START by JOB 1
IO DONE
[ time 29 ] Run JOB 0 at PRIORITY 1 [ TICKS 8 ALLOT 1 TIME 38 (of 50) ]
[ time 30 ] IO_DONE by JOB 1
[ time 30 ] Run JOB 1 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 31 (of 50) ]
[ time 31 ] Run JOB 1 at PRIORITY 2 [ TICKS 8 ALLOT 1 TIME 30 (of 50) ]
[ time 32 ] Run JOB 1 at PRIORITY 2 [ TICKS 7 ALLOT 1 TIME 29 (of 50) ]
[ time 33 ] Run JOB 1 at PRIORITY 2 [ TICKS 6 ALLOT 1 TIME 28 (of 50) ]
[ time 34 ] Run JOB 1 at PRIORITY 2 [ TICKS 5 ALLOT 1 TIME 27 (of 50) ]
[ time 35 ] Run JOB 1 at PRIORITY 2 [ TICKS 4 ALLOT 1 TIME 26 (of 50) ]
[ time 36 ] Run JOB 1 at PRIORITY 2 [ TICKS 3 ALLOT 1 TIME 25 (of 50) ]
[ time 37 ] Run JOB 1 at PRIORITY 2 [ TICKS 2 ALLOT 1 TIME 24 (of 50) ]
[ time 38 ] Run JOB 1 at PRIORITY 2 [ TICKS 1 ALLOT 1 TIME 23 (of 50) ]
[ time 39 ] IO_START by JOB 1
IO DONE
[ time 39 ] Run JOB 0 at PRIORITY 1 [ TICKS 7 ALLOT 1 TIME 37 (of 50) ]
[ time 40 ] IO_DONE by JOB 1
[ time 40 ] Run JOB 1 at PRIORITY 2 [ TICKS 9 ALLOT 1 TIME 22 (of 50) ]
[ time 41 ] Run JOB 1 at PRIORITY 2 [ TICKS 8 ALLOT 1 TIME 21 (of 50) ]
[ time 42 ] Run JOB 1 at PRIORITY 2 [ TICKS 7 ALLOT 1 TIME 20 (of 50) ]
[ time 43 ] Run JOB 1 at PRIORITY 2 [ TICKS 6 ALLOT 1 TIME 19 (of 50) ]
[ time 44 ] Run JOB 1 at PRIORITY 2 [ TICKS 5 ALLOT 1 TIME 18 (of 50) ]
[ time 45 ] Run JOB 1 at PRIORITY 2 [ TICKS 4 ALLOT 1 TIME 17 (of 50) ]
[ time 46 ] Run JOB 1 at PRIORITY 2 [ TICKS 3 ALLOT 1 TIME 16 (of 50) ]
[ time 47 ] Run JOB 1 at PRIORITY 2 [ TICKS 2 ALLOT 1 TIME 15 (of 50) ]
[ time 48 ] Run JOB 1 at PRIORITY 2 [ TICKS 1 ALLOT 1 TIME 14 (of 50) ]
[ time 49 ] IO_START by JOB 1

```

IO DONE

[time 49] Run JOB 0 at PRIORITY 1 [TICKS 6 ALLOT 1 TIME 36 (of 50)]

[time 50] IO_DONE by JOB 1

[time 50] Run JOB 1 at PRIORITY 2 [TICKS 9 ALLOT 1 TIME 13 (of 50)]

[time 51] Run JOB 1 at PRIORITY 2 [TICKS 8 ALLOT 1 TIME 12 (of 50)]

[time 52] Run JOB 1 at PRIORITY 2 [TICKS 7 ALLOT 1 TIME 11 (of 50)]

[time 53] Run JOB 1 at PRIORITY 2 [TICKS 6 ALLOT 1 TIME 10 (of 50)]

[time 54] Run JOB 1 at PRIORITY 2 [TICKS 5 ALLOT 1 TIME 9 (of 50)]

[time 55] Run JOB 1 at PRIORITY 2 [TICKS 4 ALLOT 1 TIME 8 (of 50)]

[time 56] Run JOB 1 at PRIORITY 2 [TICKS 3 ALLOT 1 TIME 7 (of 50)]

[time 57] Run JOB 1 at PRIORITY 2 [TICKS 2 ALLOT 1 TIME 6 (of 50)]

[time 58] Run JOB 1 at PRIORITY 2 [TICKS 1 ALLOT 1 TIME 5 (of 50)]

[time 59] IO_START by JOB 1

IO DONE

[time 59] Run JOB 0 at PRIORITY 1 [TICKS 5 ALLOT 1 TIME 35 (of 50)]

[time 60] IO_DONE by JOB 1

[time 60] Run JOB 1 at PRIORITY 2 [TICKS 9 ALLOT 1 TIME 4 (of 50)]

[time 61] Run JOB 1 at PRIORITY 2 [TICKS 8 ALLOT 1 TIME 3 (of 50)]

[time 62] Run JOB 1 at PRIORITY 2 [TICKS 7 ALLOT 1 TIME 2 (of 50)]

[time 63] Run JOB 1 at PRIORITY 2 [TICKS 6 ALLOT 1 TIME 1 (of 50)]

[time 64] Run JOB 1 at PRIORITY 2 [TICKS 5 ALLOT 1 TIME 0 (of 50)]

[time 65] FINISHED JOB 1

[time 65] Run JOB 0 at PRIORITY 1 [TICKS 4 ALLOT 1 TIME 34 (of 50)]

[time 66] Run JOB 0 at PRIORITY 1 [TICKS 3 ALLOT 1 TIME 33 (of 50)]

[time 67] Run JOB 0 at PRIORITY 1 [TICKS 2 ALLOT 1 TIME 32 (of 50)]

[time 68] Run JOB 0 at PRIORITY 1 [TICKS 1 ALLOT 1 TIME 31 (of 50)]

[time 69] Run JOB 0 at PRIORITY 1 [TICKS 0 ALLOT 1 TIME 30 (of 50)]

[time 70] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 29 (of 50)]

[time 71] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 28 (of 50)]

[time 72] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 27 (of 50)]

[time 73] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 26 (of 50)]

[time 74] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 25 (of 50)]

[time 75] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 24 (of 50)]

[time 76] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 23 (of 50)]

[time 77] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 22 (of 50)]

[time 78] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 21 (of 50)]

[time 79] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 20 (of 50)]

[time 80] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 19 (of 50)]

[time 81] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 18 (of 50)]

[time 82] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 17 (of 50)]

[time 83] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 16 (of 50)]

[time 84] Run JOB 0 at PRIORITY 0 [TICKS 5 ALLOT 1 TIME 15 (of 50)]

[time 85] Run JOB 0 at PRIORITY 0 [TICKS 4 ALLOT 1 TIME 14 (of 50)]

[time 86] Run JOB 0 at PRIORITY 0 [TICKS 3 ALLOT 1 TIME 13 (of 50)]

[time 87] Run JOB 0 at PRIORITY 0 [TICKS 2 ALLOT 1 TIME 12 (of 50)]

[time 88] Run JOB 0 at PRIORITY 0 [TICKS 1 ALLOT 1 TIME 11 (of 50)]

[time 89] Run JOB 0 at PRIORITY 0 [TICKS 0 ALLOT 1 TIME 10 (of 50)]

[time 90] Run JOB 0 at PRIORITY 0 [TICKS 9 ALLOT 1 TIME 9 (of 50)]

[time 91] Run JOB 0 at PRIORITY 0 [TICKS 8 ALLOT 1 TIME 8 (of 50)]

[time 92] Run JOB 0 at PRIORITY 0 [TICKS 7 ALLOT 1 TIME 7 (of 50)]

[time 93] Run JOB 0 at PRIORITY 0 [TICKS 6 ALLOT 1 TIME 6 (of 50)]

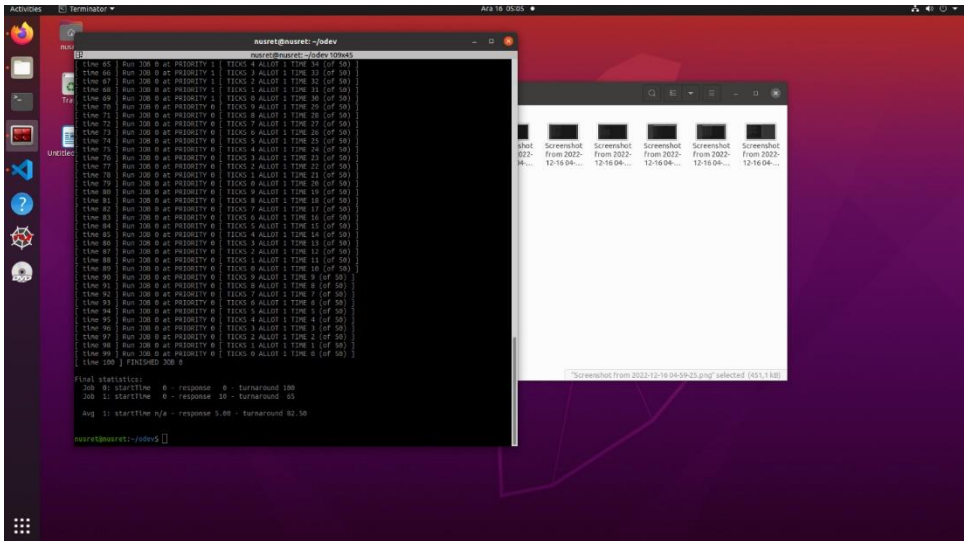
```
[ time 94 ] Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 50) ]
[ time 95 ] Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 50) ]
[ time 96 ] Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 50) ]
[ time 97 ] Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 50) ]
[ time 98 ] Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 50) ]
[ time 99 ] Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
[ time 100 ] FINISHED JOB 0
```

Final statistics:

Job 0: startTime 0 - response 0 - turnaround 100

Job 1: startTime 0 - response 10 - turnaround 65

Avg 1: startTime n/a - response 5.00 - turnaround 82.50



```
Line 65 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 50) ]
Line 66 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 50) ]
Line 67 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 50) ]
Line 68 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 50) ]
Line 69 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 50) ]
Line 70 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 71 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 72 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 73 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 74 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 75 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 76 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 77 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 78 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 79 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 80 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 81 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 82 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 83 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 84 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 85 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 86 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 87 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 88 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 89 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 90 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 91 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 92 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 93 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 94 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 95 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 96 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 97 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 98 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 99 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
Line 100 FINISHED JOB 0

Final statistics:
Job 0: startTime 0 - response 0 - turnaround 100
Job 1: startTime 0 - response 10 - turnaround 65
Avg 1: startTime n/a - response 5.00 - turnaround 82.50

nureti@nureti:~$
```

5. Soru

En yüksek kuyruğunda kuantum uzunluğu 10 ms olan bir sistem verildiğinde, tek bir uzun süreli (ve potansiyel olarak-) aç iş, CPU'nun en az %5'ini alıyor mu?

Terminalde “\$ python mlfq.py -n 3 -q 10 -l 0,200,0:0,200,1:0,200,1 -i 1 -S -B 200 -c” çalıştırılıp şu çıktı alınmıştır.

SCHEDULING: THE MULTI-LEVEL FEEDBACK QUEUE

31

```

Activities Terminator
Aix 16 06:12
nrsrsl@nrsrsl:~/joke5
nrsrsl@nrsrsl:~/joke5$ ./index22sh55
time=62 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 50) ]
time=63 10, DONE by JOB 1
time=63 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 9 (of 50) ]
time=64 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 6 (of 50) ]
time=65 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 6 (of 50) ]
time=66 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 5 (of 50) ]
time=67 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 4 (of 50) ]
time=68 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 3 (of 50) ]
time=69 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 2 (of 50) ]
time=70 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 1 (of 50) ]
time=71 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 0 (of 50) ]
time=72 FINISHED JOB 0
time=72 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 27 (of 50) ]
time=73 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 26 (of 50) ]
time=74 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 25 (of 50) ]
time=75 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 24 (of 50) ]
time=76 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 23 (of 50) ]
time=77 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 22 (of 50) ]
time=78 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 21 (of 50) ]
time=79 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 20 (of 50) ]
time=80 Run JOB 1 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 19 (of 50) ]
time=81 Run JOB 1 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 18 (of 50) ]
time=82 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 17 (of 50) ]
time=83 10, START by JOB 1
time=84 10, DONE by JOB 1
time=84 TIME
time=84 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 16 (of 50) ]
time=85 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 15 (of 50) ]
time=86 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 14 (of 50) ]
time=87 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 13 (of 50) ]
time=88 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 12 (of 50) ]
time=89 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 11 (of 50) ]
time=90 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOT 1 TIME 10 (of 50) ]
time=91 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOT 1 TIME 9 (of 50) ]
time=92 Run JOB 1 at PRIORITY 0 [ TICKS 1 ALLOT 1 TIME 8 (of 50) ]
time=93 Run JOB 1 at PRIORITY 0 [ TICKS 0 ALLOT 1 TIME 7 (of 50) ]
time=94 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 6 (of 50) ]
time=95 10, START by JOB 1
time=96 10, DONE by JOB 1
time=96 TIME
time=96 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOT 1 TIME 5 (of 50) ]
time=97 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOT 1 TIME 4 (of 50) ]
time=98 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOT 1 TIME 3 (of 50) ]
time=99 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOT 1 TIME 2 (of 50) ]
time=100 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOT 1 TIME 1 (of 50) ]
time=101 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOT 1 TIME 0 (of 50) ]
time=102 FINISHED JOB 1
Final statistics:
job 0: starttime 0 - response 0 - turnaround 72
job 1: starttime 0 - response 10 - turnaround 182
avg 1: starttime n/a - response 5.00 - turnaround 87.00
nrsrsl@nrsrsl:~/joke5$

```

```
$ python mlfq.py -n 2 -q 10 -l 0,50,0:0,50,11 -i 1 -S -l -c"
```

```
Activities Terminator
Ara 16 06:12
nsurel@nsurel:~/joker-22b0c$
nsurel@nsurel:~/joker-22b0c$
time
time 00 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOTT 1 TIME 18 (of 50) ]
time 01 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOTT 1 TIME 19 (of 50) ]
time 02 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOTT 1 TIME 18 (of 50) ]
time 03 Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOTT 1 TIME 19 (of 50) ]
time 04 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOTT 1 TIME 17 (of 50) ]
time 05 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOTT 1 TIME 16 (of 50) ]
time 06 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOTT 1 TIME 15 (of 50) ]
time 07 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOTT 1 TIME 14 (of 50) ]
time 08 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOTT 1 TIME 13 (of 50) ]
time 09 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOTT 1 TIME 12 (of 50) ]
time 10 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOTT 1 TIME 11 (of 50) ]
time 11 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOTT 1 TIME 10 (of 50) ]
time 12 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOTT 1 TIME 17 (of 50) ]
time 13 ID_START by JOB 1
time 14 ID_DONE
time 15 Run JOB 0 at PRIORITY 0 [ TICKS 9 ALLOTT 1 TIME 9 (of 50) ]
time 16 ID_DONE by JOB 1
time 17 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOTT 1 TIME 16 (of 50) ]
time 18 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOTT 1 TIME 15 (of 50) ]
time 19 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOTT 1 TIME 14 (of 50) ]
time 20 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOTT 1 TIME 13 (of 50) ]
time 21 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOTT 1 TIME 12 (of 50) ]
time 22 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOTT 1 TIME 11 (of 50) ]
time 23 Run JOB 1 at PRIORITY 0 [ TICKS 3 ALLOTT 1 TIME 10 (of 50) ]
time 24 Run JOB 1 at PRIORITY 0 [ TICKS 2 ALLOTT 1 TIME 9 (of 50) ]
time 25 Run JOB 1 at PRIORITY 0 [ TICKS 1 ALLOTT 1 TIME 8 (of 50) ]
time 26 Run JOB 1 at PRIORITY 0 [ TICKS 0 ALLOTT 1 TIME 7 (of 50) ]
time 27 Run JOB 0 at PRIORITY 0 [ TICKS 8 ALLOTT 1 TIME 6 (of 50) ]
time 28 Run JOB 0 at PRIORITY 0 [ TICKS 7 ALLOTT 1 TIME 5 (of 50) ]
time 29 Run JOB 0 at PRIORITY 0 [ TICKS 6 ALLOTT 1 TIME 4 (of 50) ]
time 30 Run JOB 0 at PRIORITY 0 [ TICKS 5 ALLOTT 1 TIME 3 (of 50) ]
time 31 Run JOB 0 at PRIORITY 0 [ TICKS 4 ALLOTT 1 TIME 2 (of 50) ]
time 32 Run JOB 0 at PRIORITY 0 [ TICKS 3 ALLOTT 1 TIME 1 (of 50) ]
time 33 Run JOB 0 at PRIORITY 0 [ TICKS 2 ALLOTT 1 TIME 2 (of 50) ]
time 34 Run JOB 0 at PRIORITY 0 [ TICKS 1 ALLOTT 1 TIME 3 (of 50) ]
time 35 Run JOB 0 at PRIORITY 0 [ TICKS 0 ALLOTT 1 TIME 4 (of 50) ]
time 36 FINISHED JOB 0
time 37 ID_START by JOB 1
time 38 ID_DONE
time 39 ID_DONE by JOB 1
time 40 ID_DONE
time 41 Run JOB 1 at PRIORITY 0 [ TICKS 9 ALLOTT 1 TIME 5 (of 50) ]
time 42 Run JOB 1 at PRIORITY 0 [ TICKS 8 ALLOTT 1 TIME 4 (of 50) ]
time 43 Run JOB 1 at PRIORITY 0 [ TICKS 7 ALLOTT 1 TIME 3 (of 50) ]
time 44 Run JOB 1 at PRIORITY 0 [ TICKS 6 ALLOTT 1 TIME 2 (of 50) ]
time 45 Run JOB 1 at PRIORITY 0 [ TICKS 5 ALLOTT 1 TIME 1 (of 50) ]
time 46 Run JOB 1 at PRIORITY 0 [ TICKS 4 ALLOTT 1 TIME 0 (of 50) ]
time 47 FINISHED JOB 1
time 48 FINISHED JOB 1
final statistics:
Job 0: starttime 0 - response 0 - turnaround 93
Job 1: starttime 0 - response 10 - turnaround 50
Avg 1: starttime n/a - response 5.00 - turnaround 97.00
nsurel@nsurel:~/joker5$
```