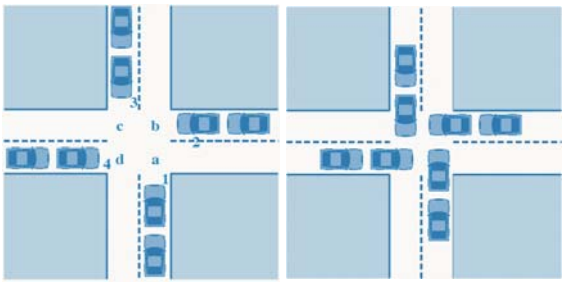
5 ÖLÜMCÜL KİLİTLENME

Ölümcül Kilitlenme

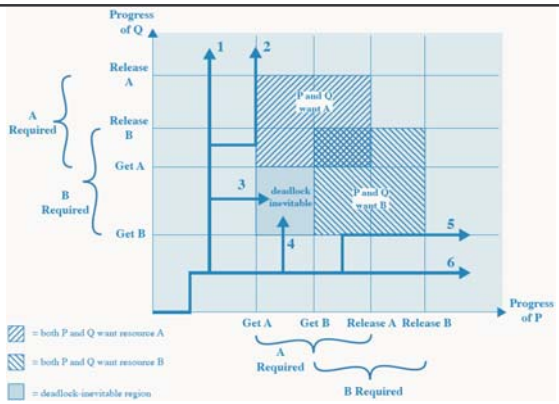
- Sistem kaynaklarını ortak olarak kullanan veya birbiri ile haberleşen bir grup prosesin kalıcı olarak bloke olması durumu : *ölümcül kilitlenme*
- Birden fazla proses olması durumunda proseslerin bir kaynağı ellerinde tutmaları ve bir başka kaynağı istemeleri durumunda ölümcül kilitlenme olası.
- Etkin bir çözüm yok

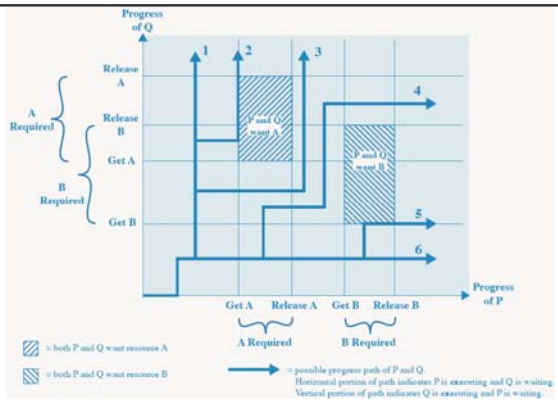


a) Olası ölümcül kilitlenme b) Ölümcül kilitlenme

Ölümcül Kilitlenme Örneği - 1

Process P		Process Q	
Step	Action	Step	Action
p_0	Request (D)	q_0	Request (T)
p_1	Lock (D)	q_1	Lock (T)
p_2	Request (T)	q_2	Request (D)
p_3	Lock (T)	q_3	Lock (D)
p_4	Perform function	q_4	Perform function
p_5	Unlock (D)	q_5	Unlock (T)
p_6	Unlock (T)	q_6	Unlock (D)





Ölümcül Kilitlenme Örneği - 2

- 200K sekizlilik bir bellek bölgesi proseslere atanabilir durumda ve aşağıdaki istekler oluşuyor:
- Her iki proses de ilk isteklerini aldıktan sonra ikinci isteklerine devam ederlerse kilitlenme oluşur.

P1 Prosesi	**P2 Prosesi**
80K sekizli iste	70K sekizli iste
...	...
60K sekizli iste	80K sekizli iste

Ölümcül Kilitlenme Örneği - 3

- Mesaj alma komutu bloke olan türden ise ölümcül kilitlenme olabilir.

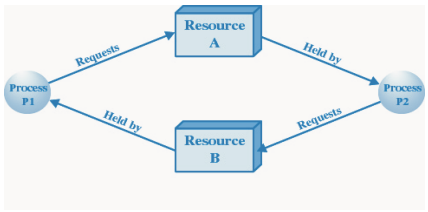
P1 Prosesi	**P2 Prosesi**
Al (P2);	Al (P1);
...	...
Gönder (P2,M1);	Gönder (P1,M2);

Ölümcül Kilitlenme Olması İçin Gereken Koşullar

- Karşılıklı dışlama
- Proseslerin ellerine geçirdikleri kaynakları diğer istedikleri kaynakları da ele geçirene kadar bırakmamaları
- Proseslerin ellerinde tuttukları kaynaklar işletim sistemi tarafından zorla geri alınamıyorsa (“pre-emption” yok)

Ölümcül Kilitlenme Olması İçin Gereken Koşullar

- Çevrel bekleme durumu oluşuyorsa



Kaynak Atama Grafı

- Yönlü bir graf
- Kaynakların ve proseslerin durumunu gösterir
- Prosesler P_i ve kaynaklar K_j olsun.
  - $R_3 \rightarrow P_5$: Bir adet R_3 kaynağının P_5 prosesine atanmış olduğunu gösterir: ***atama kenarı***
  - $P_2 \rightarrow R_1$: P_2 prosesi bir adet R_1 kaynağı için istekte bulunmuştur: ***istek kenarı***

Kaynak Atama Grafı

Örnek:

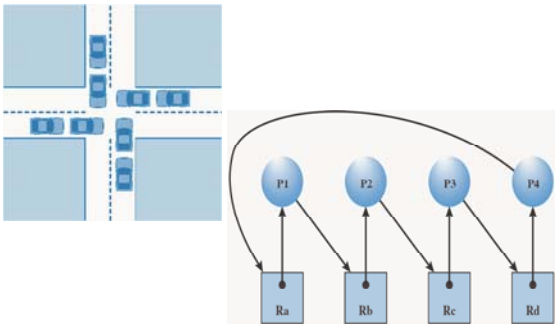
P, R ve E kümeleri olsun.

- $P=\{P_1,P_2,P_3\}$
- $R=\{R_1,R_2,R_3,R_4\}$
- $E=\{P_1\rightarrow R_1,\ P_2\rightarrow R_3,\ R_1\rightarrow P_2,\ R_2\rightarrow P_2,\ \ R_2\rightarrow P_1,\ R_3\rightarrow P_3\}$

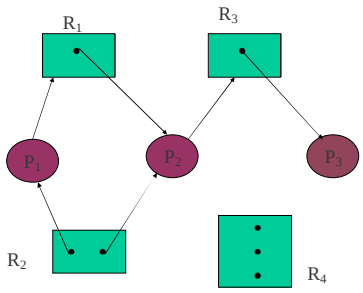
Kaynaklar:
- 1 adet R_1
- 2 adet R_2
- 1 adet R_3
- 3 adet R_4

Proses Durumları:
- P_1: bir adet R_2 elinde tutuyor, bir adet R_1 istiyor
- P_2: birer adet R_1 ve R_2 elinde tutuyor, bir adet R_3 istiyor
- P_3: bir adet R_3 elinde tutuyor

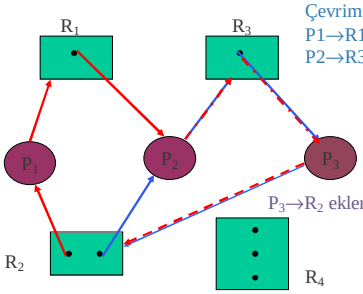
Örnek



Kaynak Atama Grafı



Kaynak Atama Grafı

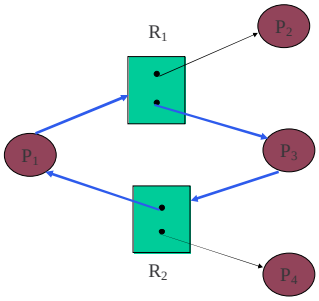


Çevrimler:
P1→R1→P2→R3→P3→R2→P1
P2→R3→P3→R2→P2

$P_3 \rightarrow R_2$ eklensin.

P_1, P_2 ve P_3 Ölümcül kilitlenme durumundalar!

Kaynak Atama Grafı



Ölümcül kilitlenme durumu yok.
NEDEN?

Ölümcül Kilitlenme Durumunda Kullanılan Yaklaşımlar

- Sistemin hiçbir zaman ölümcül kilitlenme durumuna girmemesini sağlamak
- Sistemin, ölümcül kilitlenme durumuna girdikten sonra bu durumdan kurtulmasını sağlamak
- Problemi gözardı edip sistemde ölümcül kilitlenme olmayacağını varsaymak

Ölümcül Kilitlenme Durumunda Kullanılan Yaklaşımlar

- Sistemin hiçbir zaman ölümcül kilitlenme durumuna girmemesini sağlamak
  - ölümcül kilitlenmeyi önlemek
  - ölümcül kilitlenmeden kaçınmak

Ölümcül Kilitlenmeyi Önleme

- Ölümcül kilitlenmenin oluşmasının mümkün olmadığı bir sistem tasarlanması
  - Dört gerekli koşuldan en az birinin geçerli olmaması

Ölümcül Kilitlenmeyi Önleme

- **Karşılıklı Dışlama:**
  - Paylaşılan kaynaklar olması durumunda bu koşulun engellenmesi mümkün değil.
- **Tut ve Bekle:**
  - Kaynak isteyen prosesin elinde başka k aynak tutmuyor olmasının sağlanması
  - Proseslerin tüm kaynak isteklerini baştan belirtmeleri ve tüm istekleri birden karşılanana kadar proseslerin bekletilmesi yaklaşımı ile bu koşul engellenebilir.

Ölümcül Kilitlenmeyi Önleme

- Etkin değil:
  - Proses tüm istekleri karşılanana kadar çok bekleyebilir, halbuki bir kısım kaynağını ele geçirerek işinin bir bölümünü bitirebilir.
  - Bir prosese atanan kaynakların bir kısmı bir süre kullanılmadan boş bekleyebilir.
- Proses tüm kaynak ihtiyaçlarını baştan bilemeyebilir.

Ölümcül Kilitlenmeyi Önleme

- Diğer yöntem: Bir prosesin yeni bir kaynak isteğinde bulunduğunda, kaynak ataması yapılmadan önce elindeki tüm kaynakları bırakmasının beklenmesi:
  - kaynak kullanımı etkin değil
  - tutarlılık sorunu olabilir
- Her iki yöntemde de açlık (starvation) olası.

Ölümcül Kilitlenmeyi Önleme

Örnek:

Teyp sürücüsündeki verileri okuyup diskte bir dosyaya yazan, sonra diskteki dosyadaki bilgileri sıralayıp, yazıcıya bastıran proses.

- Her iki yöntemin farkı ne?
- Her iki yöntemin sorunları ne?

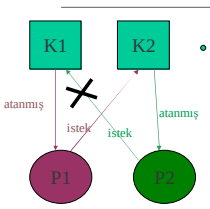
Ölümcül Kilitlenmeyi Önleme

- **Pre-Emption Olmaması Koşulu :**
  - Elinde bazı kaynakları tutan bir prosesin yeni bir isteği karşılanamıyorsa elindekileri de bırakması ve gerekiyorsa yeni kaynakla birlikte tekrar istemesi ile bu koşul engellenebilir.
  - Bir proses bir başka proses tarafından elinde tutulan bir kaynak isterse, ikinci prosesin kaynaklarını bırakması işletim sistemi tarafından sağlanabilir. (*Bu yöntem proseslerin eşit öncelikli olmadıkları durumda uygulanabilir.*)
  - Bu yaklaşım durumları saklanabilen ve yeniden yüklenebilen kaynaklar için uygun.

Ölümcül Kilitlenmeyi Önleme

- **Çevrel Bekleme Koşulu:**
  - Kaynak tipleri arasında lineer bir sıralama belirleyerek önlenebilir.
    - Örneğin elinde R tipi kaynaklar tutan bir proses sadece lineer sıralamada R'yi izleyen kaynakları alabilir.
  - Bu yöntem de prosesleri yavaşlatmasının ve kaynakları gereksiz yere boş bekletmesinin olası olması nedeniyle etkin değil

Ölümcül Kilitlenmeyi Önleme



- K2 lineer sırada K1'den sonra.
- K2'yi elinde tutan P2, K1 için istekte bulunamaz.

Ölümcül Kilitlenmeden Kaçınma

- Yeni bir kaynak isteği geldiğinde, bu isteğin karşılanmasının bir kilitlenmeye neden olup olamayacağı dinamik olarak belirlenir.
- Proseslerin gelecekteki kaynak istekleri (ne zaman ve hangi sırayla) ve kaynakları geri verme anları hakkında önceden bilgi gerektirir.

Ölümcül Kilitlenmeden Kaçınma

- Temel olarak iki yöntem var:
  - Eğer istekleri bir ölümcül kilitlenmeye yol açabilecekse prosesi başlatma
  - Eğer istek bir ölümcül kilitlenmeye yol açabilecekse prosesin ek kaynak isteklerini karşılama.

Prosesin Başlamasının Önlenmesi

- n adet proses ve m adet farklı tip kaynak olsun.
- Sistemdeki tüm kaynaklar:
  Kaynak=(R1, R2, ..., Rm)
- Sistemdeki boş kaynaklar:
  Boş=(V1, V2, ..., Vm)

Prosesin Başlamasının Önlenmesi

- Proseslerin her kaynak için maksimum istek matrisi:

$$\text{İstek} = \begin{bmatrix} C_{11} & C_{12} & \ldots & C_{1m} \\ C_{21} & C_{22} & \ldots & C_{2m} \\ & \ldots\ldots\ldots & & \\ C_{n1} & C_{n2} & \ldots & C_{nm} \end{bmatrix}$$

Prosesin Başlamasının Önlenmesi

- Proseslere şu anda her kaynaktan atanmış olanların gösterildiği matris:

$$\text{Atanmış} = \begin{bmatrix} A11 & A12 & \dots & A1m \\ A21 & A22 & \dots & A2m \\ & \dots\dots\dots & & \\ An1 & An2 & \dots & Anm \end{bmatrix}$$

Prosesin Başlamasının Önlenmesi

- Tüm kaynaklar ya boştur ya da kullanılmaktadır.
- Prosesler sistemde bulunandan daha fazla kaynak isteğinde bulunamaz.
- Prosese, ilk başta belirttiğinden fazla kaynak ataması yapılmaz.

Prosesin Başlamasının Önlenmesi

- P_{n+1} prosesini ancak ve ancak aşağıdaki koşul gerçeklendiğinde başlat:

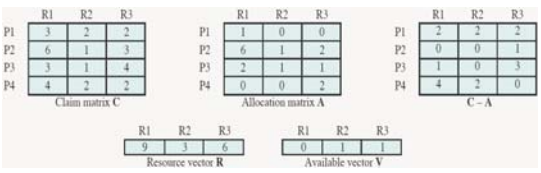
$$R_i \geq C_{(n+1)i} + \sum_{k=1}^{n} C_{ki} \quad \text{, tüm i'ler için}$$

- Optimal değil. Tüm proseslerin maksimum kaynak gereksinimlerini birden isteyeceğini varsayar.

Kaynak Atamasının Engellenmesi

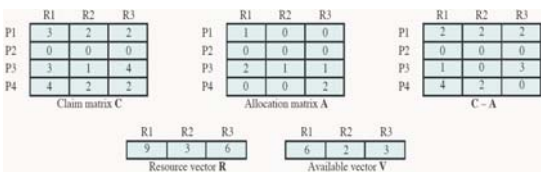
- **Banker algoritması**
- Sistemde sabit sayıda proses ve kaynak var.
- ***Sistemin durumu*:** Kaynakların proseslere o anki ataması ⇒ durum ***Kaynak*** ve ***Boş*** vektörlerinden ve ***Atanmış*** ve ***İstek*** matrislerinden oluşur.
- ***Emin durum*:** Ölümcül kilitlenmeye yol açmayacak en az bir atama sekansı olduğu durum (yani tüm proseslerin sonlanabileceği durum)
- ***Emin olmayan durum***

Emin Durumun Belirlenmesi Başlangıç Durumu

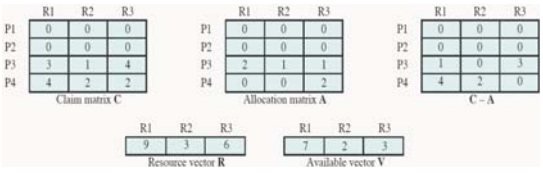


Bu emin bir durum mu?

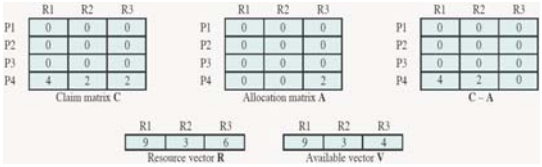
Emin Durumun Belirlenmesi P2 Sonlanır



Emin Durumun Belirlenmesi P1 Sonlanır



Emin Durumun Belirlenmesi P3 Sonlanır

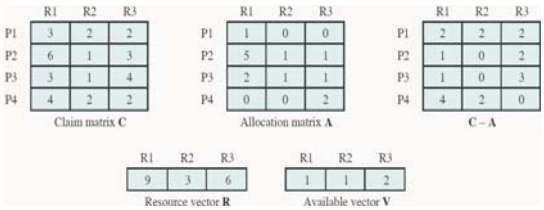


En son olarak da P4 sonlanır ⇒ Emin Durum √

Ölümcül Kilitlenmeden Kaçınma

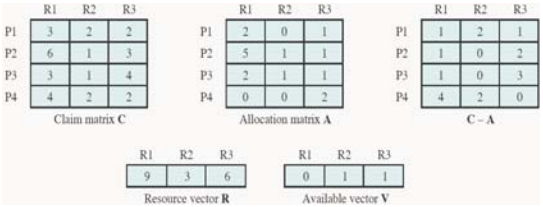
- Bir proses bir grup kaynak için istekte bulunduğunda işletim sistemi,
  - Kaynakların atandığını varsayarak sistem durumunu günceller.
  - Yeni durum emin bir durum mu diye kontrol eder.
  - Emin ise atamayı gerçekleştirir.
  - Emin değilse atamayı yapmaz ve istekte bulunan prosesi isteklerin karşılanması uygun olana kadar bloke eder.

Emin Olmayan Durumun Belirlenmesi: Başlangıç Durumu



P2 bir adet R1 ve bir adet de R3 kaynağı için daha istekte bulunursa ne olur?

Emin Olmayan Durumun Belirlenmesi



P1 bir adet R1 ve bir adet de R3 kaynağı için daha istekte bulunursa ne olur?

Emin Olmayan Durumun Belirlenmesi

- Emin bir durum değil. Bu nedenle P1'in yeni istekleri karşılanmaz ve P1 bloke edilir.
- Oluşan durum ölümcül kilitlenme durumu değildir sadece ölümcül kilitlenme oluşması potansiyelini taşımaktadır.

Ölümcül Kilitlenmeden Kaçınma

- Kullanımındaki kısıtlamalar:
  - Her prosesin maksimum kaynak ihtiyacı önceden belirtilmeli.
  - Prosesler birbirlerinden bağımsız olmalı. Koşma sıralarının önemi olmamalı.
  - Kaynak sayısı sabit olmalı.
  - Elinde kaynak tutan proses kaynakları bırakmadan sonlanmamalı.

Ölümcül Kilitlenmeyi Sezme

- Ölümcül kilitlenmeyi önleme yöntemleri kadar kısıtlayıcı değil.
- Tüm kaynak istekleri karşılanır. İşletim sistemi periyodik olarak sistemde çevrel bekleme durumu oluşup oluşmadığını kontrol eder.
- Kontrol sıklığı ölümcül kilitlenme oluşması sıklığına bağlıdır: Her yeni kaynak isteğinde bile yapılabilir.

Ölümcül Kilitlenmeyi Sezme

- Ölümcül kilitlenmenin sezilmesi için kullanılan algoritmada *Atanmış* matrisi ve *Boş* vektörü kullanılıyor. Q istek matrisi tanımlanıyor. (Burada q_{ij}: i. prosesinin j tipi kaynaktan kaç tane istediği.)
- Algoritma kilitlenmemiş prosesleri belirleyip işaretler.
- Başlangıçta tüm prosesler işaretsizdir. Aşağıdaki adımlar gerçekleştirilir:

Ölümcül Kilitlenmeyi Sezme

- **Adım 1:** Atanmış matrisinde bir satırının tamamı sıfır olan prosesleri işaretle.
- **Adım 2:** *Boş* vektörüne karşılık düşecek bir W geçici vektörü oluştur.
- **Adım 3:** Q matrisinin i. satırındaki tüm değerlerin W vektöründeki değerlerden küçük ya da eşit olduğu bir i belirle (P_i henüz işaretlenmemiş olmalı).

  $Q_{ik} \le W_k$, $1 \le k \le m$

Ölümcül Kilitlenmeyi Sezme

- **Adım 4:** Böyle bir satır bulunamıyorsa algoritmayı sonlandır.
- **Adım 5:** Böyle bir satır bulunduysa i prosesini işaretle ve *Atanmış* matrisinde karşılık düşen satırı W vektörüne ekle.

  $W_k = W_k + A_{ik}$, $1 \le k \le m$
- **Adım 6:** Adım 3'e dön.
- Algoritma sonlandığında işaretsiz proses varsa ⇒ Ölümcül kilitlenme var.

Ölümcül Kilitlenmeyi Sezme

- Algoritma sonlandığında işaretsiz proses varsa ⇒ Ölümcül kilitlenme var.
- İşaretsiz prosesler kilitlenmiş.
- Temel yaklaşım, mevcut kaynaklarla istekleri karşılanabilecek bir proses bulmak, kaynakları atamak ve o proses koşup sonlandıktan sonra bir başka prosesin aranmasıdır.
- Algoritma sadece o an ölümcül kilitlenme olup olmadığını sezer.

Ölümcül Kilitlenmeyi Sezme

	R1	R2	R3	R4	R5
P1	0	1	0	0	1
P2	0	0	1	0	1
P3	0	0	0	0	1
P4	1	0	1	0	1

Request matrix **Q**

	R1	R2	R3	R4	R5
P1	1	0	1	1	0
P2	1	1	0	0	0
P3	0	0	0	1	0
P4	0	0	0	0	0

Allocation matrix **A**

R1	R2	R3	R4	R5
2	1	1	2	1

Resource vector

R1	R2	R3	R4	R5
0	0	0	0	1

Available vector

1) P4'ü işaretle.
2) W = (0 0 0 0 1)
3) P3'ün isteği W'dan küçük veya eşit ⇒ P3 işaretle.
W = W + (0 0 0 1 0) = (0 0 0 1 1)
4) İşaretsiz hiçbir proses için Q'daki ilgili satır W'dan küçük ya da eşit değil ⇒ Algoritmayı sonlandır.

Sonuçta P1 ve P2 isaretsiz ⇒ kilitlenmişler

Ölümcül Kilitlenme Sezildikten Sonra Yapılabilecekler

- Tüm kilitlenmiş prosesleri sonlandır.
- Tüm kilitlenmiş proseslerin eski bir kontrol noktasına kadar kopyasını al ve tüm prosesleri bu noktadan yeniden başlat.
  - aynı ölümcül kilitlenme yeniden oluşabilir
- Kilitlenme ortadan kalkana kadar sırayla kilitlenmiş prosesleri sonlandır.
- Kilitlenme ortadan kalkana kadar, sırayla atanmış kaynakları geri al.

Kilitlenmiş Prosesler İçin Seçim Kriterleri

- O ana kadar en az işlemci zamanı kullanmış olan
- O ana kadar en az sayıda çıktı satırı oluşturmuş olan
- Beklenen çalışma süresi en uzun olan
- O ana kadar en az kaynak atanmış olan
- En düşük öncelikli olan

Makarnacı Düşünürler Problemi