İşletim Sistemlerine Giriş

Ölümcül Kilitlenme (Deadlock)

Ölümcül Kilitlenme (Deadlock)

Bilgisayar sistemleri, bir anda sadece tek bir kullanıcı tarafından kullanılabilecek kaynaklar ile doludur.

*Yazıcı, disket sürücüleri, kesiciler, cd yazıcı, cd okuyucu,...

Aynı anda iki süreç yazıcıdan çıktı almaya çalışırsa ve alabilirse oluşan çıktı anlamsız verilerden oluşur. Aynı anda iki süreç aynı dosya ya yazar ise oluşan dosya bozulur ya da dosya sistemi bozulur.

Her işletim sistemi, belirli kaynaklara aynı anda sadece tek bir sürecin erişmesini sağlayan erişim mekanizmalarına sahiptir.

Ölümcül Kilitlenme (Deadlock)

Süreçler aynı anda birden fazla paylaşılan kaynağı kullanabilir.

Örneğin, A ve B süreçleri tarayıcıdan aldıkları veriyi CD ye kayıt etmek istesinler. A süreci tarayıcının kullanım hakkını elde etsin, B süreci de CD nin yazım hakkını elde etsin.

A süreci yazıcıya yazmak istediğinde B'nin kullanımı nedeniyle A sürecine yetki verilmez.

Aynı şekilde B sürecide tarayıcıya erişmeye çalıştığında o da A sürecinin tarayıcı yetkisine sahip olması nedeniyle tarayıcıya erişemez.

Ölümcül Kilitlenme (Deadlock)

Bu durumda her iki süreçte bloklanırlar ve hiçbir zaman bu durumdan kurtulamazlar. Gerçekleşen bu duruma ***ölümcül kilitlenme (deadlock)*** denilir.

Sistem kaynaklarını ortak olarak kullanan veya birbirleri ile haberleşen bir grup sürecin kalıcı olarak bloklanması durumudur.

Genelde süreçlerin ellerindeki kaynakları bırakmadan, başka süreçlere ait olan kaynakları istedikleri durumlarda gerçekleşir.

Ölümcül Kilitlenme (Deadlock)

Bilgisayarda bulunan kaynaklar aşağıdaki şekilde kategorileştirilebilir.

-Ele geçirilebilir (preemptable)
-Ele geçirilemez (non-preemptable)

Ele geçirilebilir kaynak, bir süreç bu kaynağa sahip ise elinden alındığında problemler oluşturmuyan kaynaktır.(bellek)

Ele geçirilemeyen kaynak, süreç bu kaynağa sahiptir ve elinden alınamaz. Örneğin, bir süreç cd yazıcıyı kullanıyorsa cd yazıcının elinden alınması oluşan cd nin bozulmasına yol açar. Ölümcül kilitlenme bu kaynaklarda oluşur.

Ölümcül Kilitlenme (Deadlock)

Bir kaynağı kullanmak için aşağıdaki adımlar izlenir:

1. Kaynağı iste
2. Kaynağı kullan
3. Kaynağı serbest bırak

Genelde her kaynağa bir semafor verilir. Kaynak kullanmadan önce semafora ait olan down() metodu çalıştırılır, kaynak kullanılır, kaynak ile işlem bittikten sonra up() metodu çalıştırılır.

Ölümcül Kilitlenme (Deadlock)

ders06_001.c

```
semafor kaynak_1 = 1;
semafor kaynak_2 = 1;

void surec_A(){
      down(&kaynak_1);
      kaynak_1_i_kullan();
      up(&kaynak_1);
}

void surec_B(){
      down(&kaynak_1);
      down(&kaynak_2);
      kaynak_1_i_kullan();
      kaynak_2_i_kullan();
      up(&kaynak_2);
      up(&kaynak_1);
}
```

Ölümcül Kilitlenme (Deadlock)

ders06_002.c

```
semafor kaynak_1 = 1;
semafor kaynak_2 = 1;

void surec_A(){
      down(&kaynak_1);
      down(&kaynak_2);
      kaynaklari_kullan();
      up(&kaynak_2);
      up(&kaynak_1);
}

void surec_B(){
      down(&kaynak_1);
      down(&kaynak_2);
      kaynaklari_kullan();
      up(&kaynak_2);
      up(&kaynak_1);
}
```

ölümcül kilitlenme olmaz

ders06_003.c

```
semafor kaynak_1 = 1;
semafor kaynak_2 = 1;

void surec_A(){
      down(&kaynak_1);
      down(&kaynak_2);

      kaynaklari_kullan();

      up(&kaynak_2);
      up(&kaynak_1);
}

void surec_B(){
      down(&kaynak_2);
      down(&kaynak_1);

      kaynaklari_kullan();

      up(&kaynak_1);
      up(&kaynak_2);
```

ölümcül kilitlenme olasılığı var

Aynı kaynakların farklı sıralarda istenilmesinde ölümcül kilitlenme olma olasılığı yüksektir.

Ölümcül Kilitlenme (Deadlock)

Örneğin sistemimizde, 200 K bellek bölgesi süreçler istediklerinde bu süreçlerin kullanımına verilsin. Bellek kullanan başka süreçlerin olmadığını düşünürsek, aşağıdaki durumda ölümcül kilitlenme olma olasılığı vardır.

1. Süreç	2.Süreç
----------	----------
iste(80K)	iste(70K)
...	...
...	...
iste(60K)	iste(80K)

Ölümcül Kilitlenme (Deadlock)

Süreçler, mesaj alıp vererek haberleşiyor olsunlar. Süreç beklediği mesaj gelene kadar bloklansın. Aşağıdaki durumda ölümcül kilitlenme olma olasılığı vardır.

1. Süreç	2.Süreç
receive(Süreç2)	receive(Süreç1)
...	...
...	...
send(Süreç2,mesaj)	send(Süreç1,mesaj)

Ölümcül Kilitlenmeye (Deadlock) Neden Olan Durumlar

*** Karşılıklı Dışlama(Mutual Exclusion)**

Her kaynağı aynı anda bir süreç ya kullanıyordur ya da atanmıştır.

*** Sahiplenme ve Bekleme Koşulu**

Elindeki kaynağı bırakmadan başka kaynak ister ve bu kaynağı elde edemediğinde beklemeye geçen süreçler.

*** Geri Alınamaz Koşulu**

Bir sürece daha önceden verilen kaynaklar zorla elinden alınamaz.

***Çevrimsel Bekleme Koşulu**

İki veya daha fazla süreç zincir şeklinde kendinden önceki sürece ait olan kaynağı bekler.

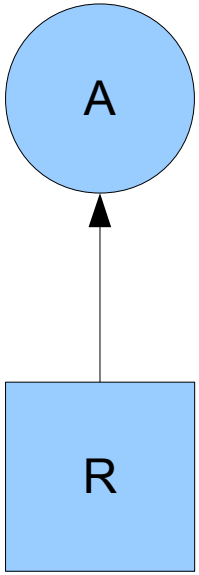
Ölümcül Kilitlenmenin Modellenmesi

Süreçleri daireler şeklinde, kaynakları kareler şeklinde gösterelim.

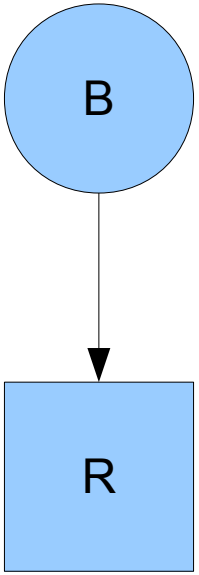
Bir süreçten kaynağa doğru giden ok, sürecin kaynak için bloklandığını gösterir.

Kaynaktan sürece doğru gösterilen ok, sürecin kaynağa sahip olduğunu gösterir.

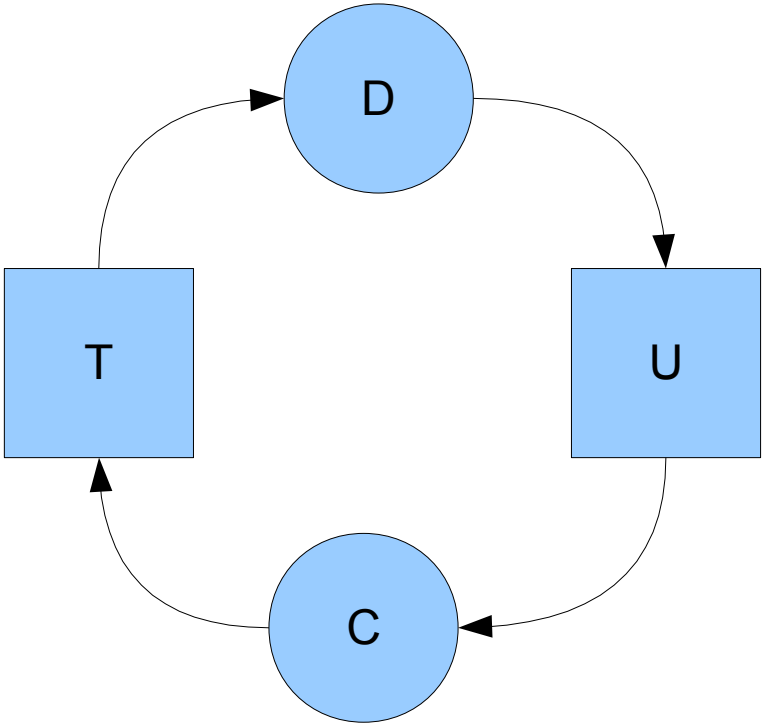
Ölümcül Kilitlenmenin Modellenmesi



A süreci R kaynağına sahiptir



B süreci R kaynağı için bekliyor



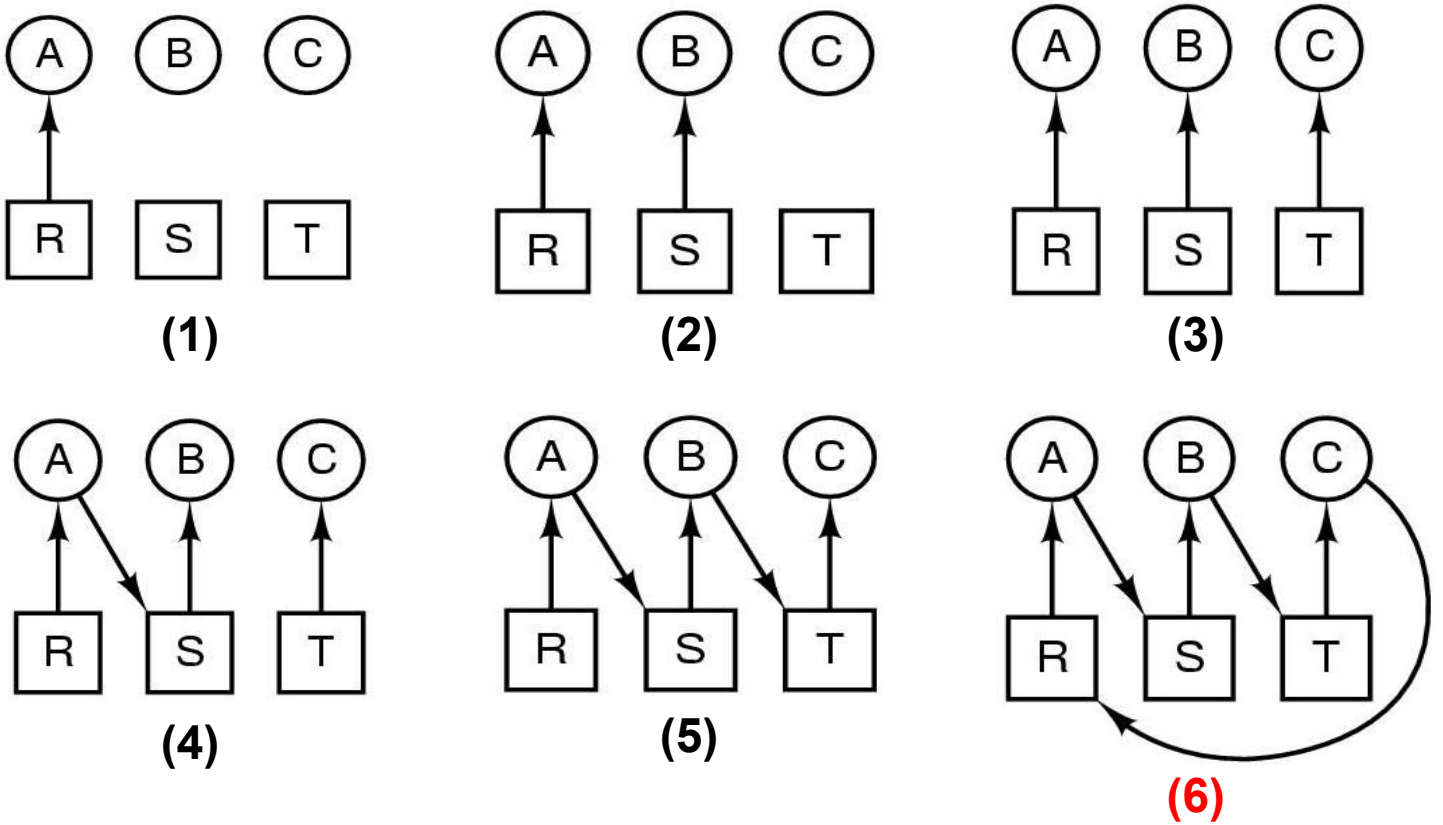
Ölümcül Kilitlenme

A süreci	B süreci	C süreci
R ister	S ister	T ister
S ister	T ister	R ister
R bırakır	S bırakır	T bırakır
S bırakır	T bırakır	R bırakır

Süreçlerin Çalışması

1. A, R yi ister
2. B, S i ister
3. C, T yi ister
4. A, S i ister
5. B, T yi ister
6. C, R yi ister

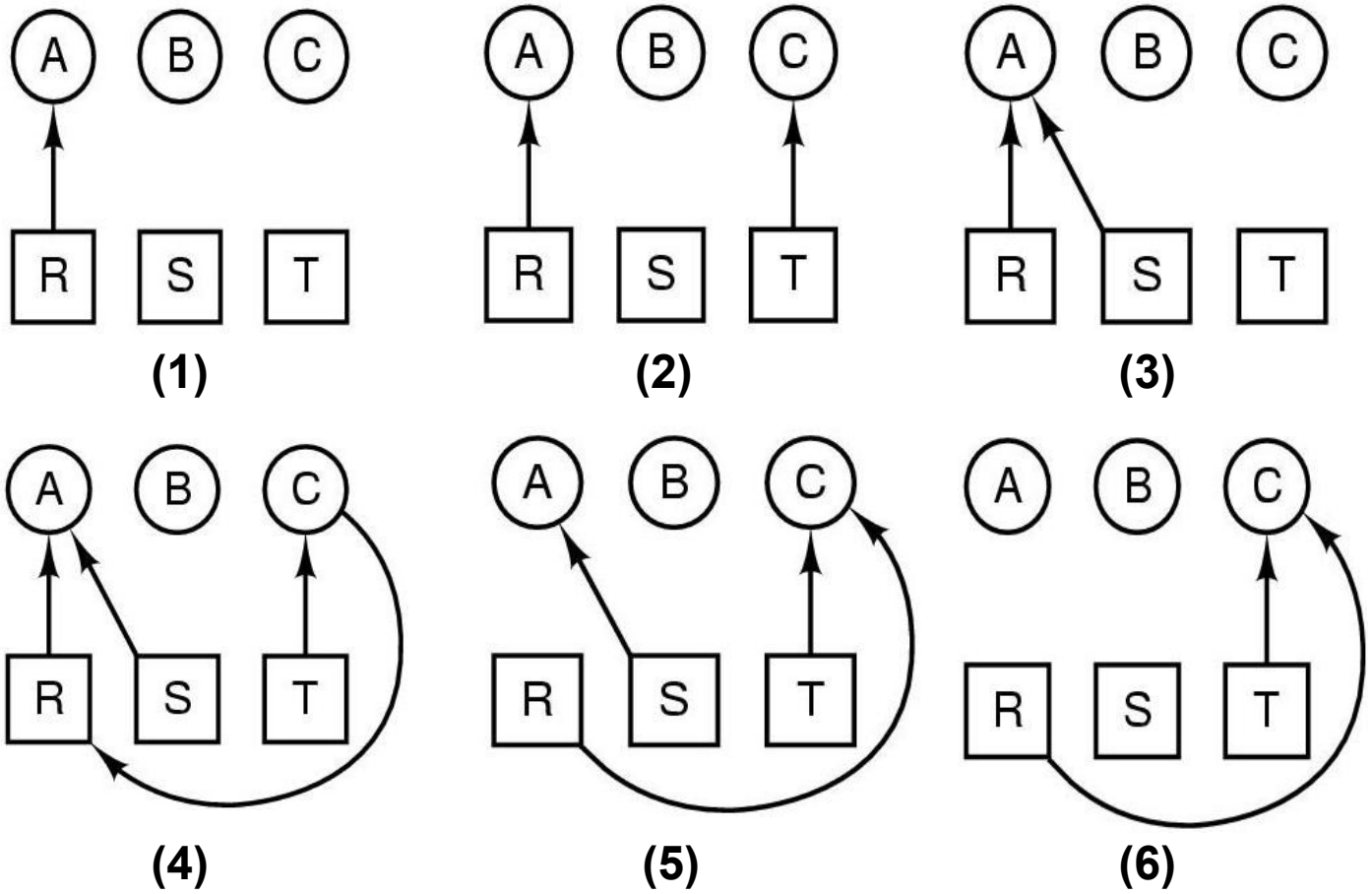
(ölümcül kilitlenme)



A süreci	B süreci	C süreci
R ister	S ister	T ister
S ister	T ister	R ister
R bırakır	S bırakır	T bırakır
S bırakır	T bırakır	R bırakır

Süreçlerin Çalışması

1. A, R yi ister
2. C, T yi ister
3. A, S i ister
4. C, R yi ister
5. A, R yi bırakır
6. A, S i bırakır



Devekuşu Algoritması (Ostrich Algorithm)

Başınızı kuma gömerek, bir şey yokmuş gibi davranmak.

Matematikçiler tamamen kabul edilmez bulurlar ve ölümcül kilitlenmelerin ne pahasına olursa olsun önlenmesini söylerler.

Mühendisler ise problemin ne kadar sıklıkla olması beklendiğini ve sitemin diğer nedenlerden ne kadar sıklıkla bozulduğunu sorgularlar.

Eğer ölümcül kilitlenmeler yılda birkez oluyorsa fakat sistemdeki donanım bozulmaları, derleyici hataları ve işletim sistemi hataları haftada birkez bozuluyorsa, birçok mühendis ölümcül kilitlenmelerin çözülmesininin getireceği performans kaybına, performansın iyi olmasını tercih eder.
Birçok işletim sistemi yok sayar.

Ölümcül Kilitlenme Tespiti ve Onarma

1.Her tip kaynaktan sistemde bir adet olduğu durumda ölümcül kilitlenme tespiti

Sistemde bir yazıcı, bir cd yazıcı, bir disket sürücü... olsun.

Sistem için kaynak çizgesi(graph) oluşturulur.

Bu çizge bir veya daha fazla döngü(cycle) içeriyorsa, bir ölümcül kilitlenme vardır.

Döngünün parçası olan süreçler ölümcül kilitlenmiştir.

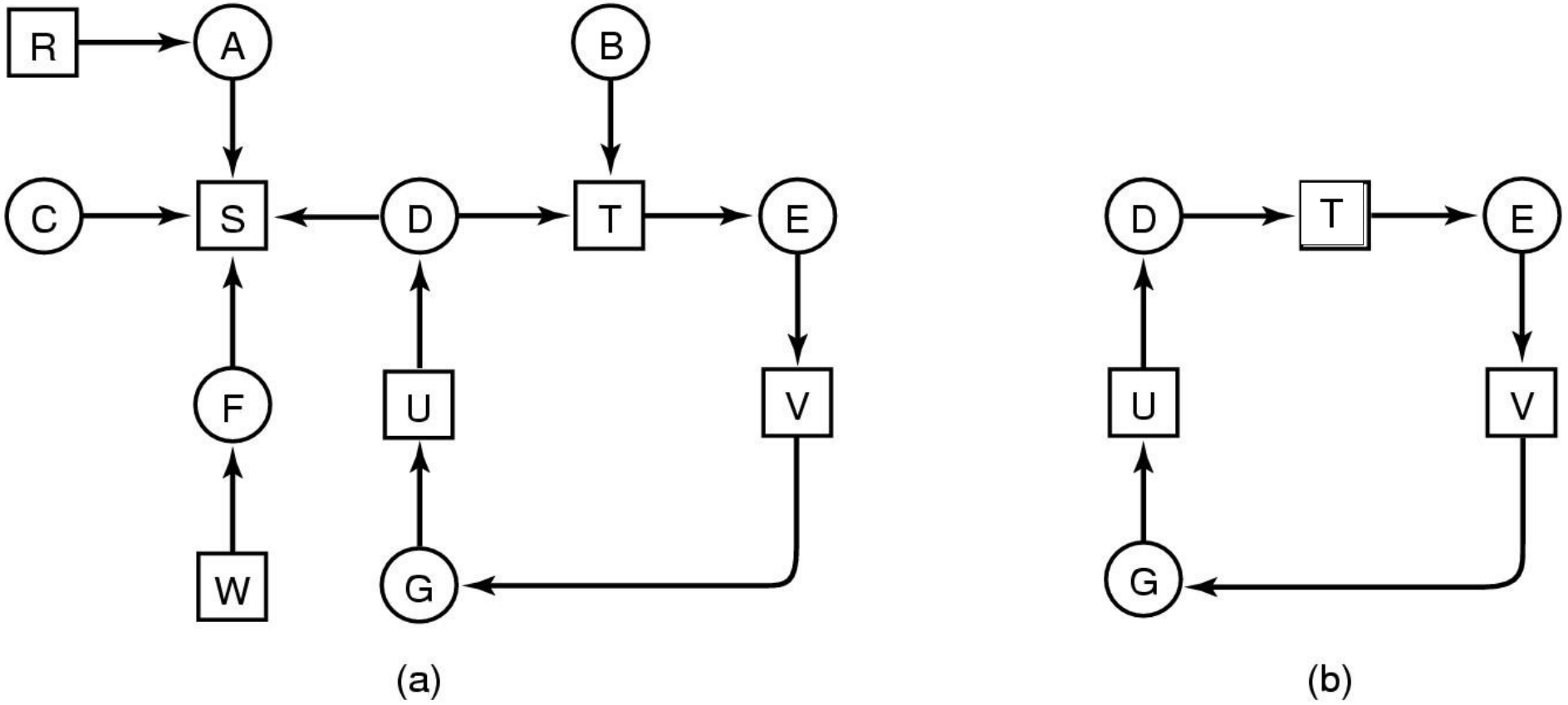
Ölümcül Kilitlenme Tespiti ve Onarma

Örneğin; 7 süreç ve 6 adet kaynak olsun. A,B,C,D,E,F,G süreçler; R,S,T,U,V,W kaynakları göstersin. Mevcut sistemdeki sahip olunan kaynak durumu ve istenilen kaynak durumu şu şekilde olsun:

1. A süreci R yi tutar ve S i ister.
2. B süreci T yi ister.
3. C süreci S i ister.
4. D süreci U yu tutar ve S ile T yi ister.
5. E süreci T yi tutar ve V yi ister.
6. F süreci W yi tutar ve S i ister.
7. G süreci V yi tutar ve U yu ister.

SORU:Sistem ölümcül kilitli midir ? Öyleyse hangi süreçler ?

Ölümcül Kilitlenme Tespiti ve Onarma



Yönlü çizge(graph) lerde döngü(cycle) leri tespit eden bir çok algoritma bulunmaktadır.

Ölümcül Kilitlenme Tespiti ve Onarma

Algoritma bir döngü bulunca çıkar. L düğümlerin listesidir.

1. Çizgedeki her N düğümü için, N'i başlangıç düğümü alarak 2-6 adımlarını yap.

2. L yi boş liste olarak başlat. Tüm kenarları işaretlenmemiş yap.

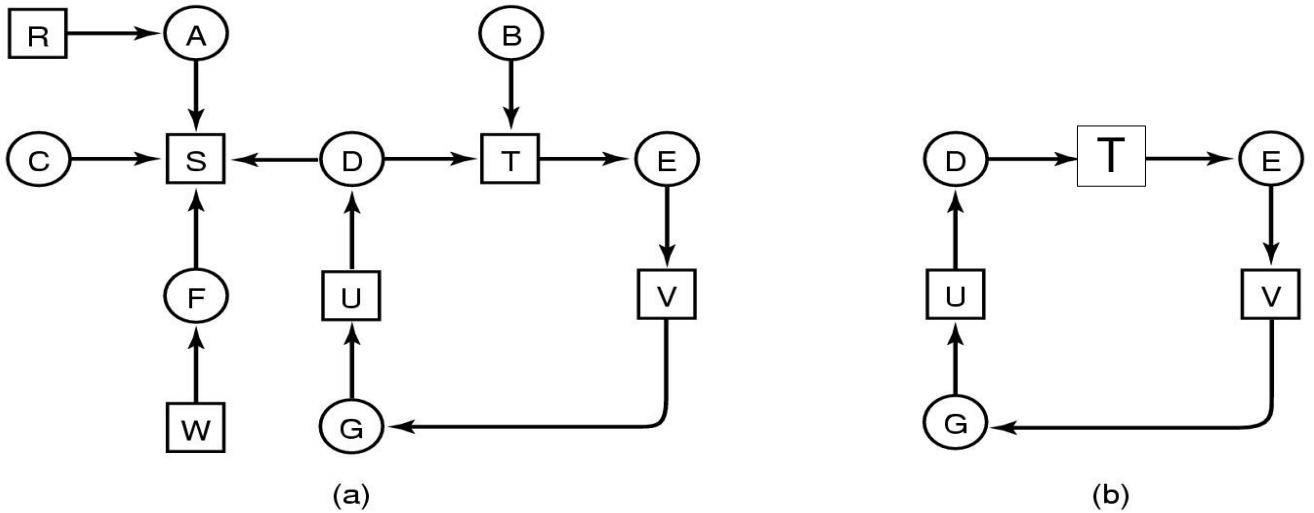
3. Şu anki düğümü L ye ekle. L de iki kez olup olmadığını kontrol et. İkinci kez bulunursa çizgede döngü vardır ve algoritma sonlanır.

4. Verilen düğümden başka düğümlere işaretlenmemiş giden kenar var mı? Varsa 5. adıma yoksa 6. adıma git.

5. İşaretli olmayan bir kenar seç ve işaretle. Bu kenarı izleyerek yeni düğüme git. Adım 3 e git.

6. Bir ölü noktaya ulaştık. Bu düğümü çıkar ve bir önceki düğüme git. Önceki düğümü aktif düğüm yap ve adım 3 e git. Eğer bu düğüm başlangıç düğümü ise döngü yoktur.

Ölümcül Kilitlenme Tespiti ve Onarma



a) L=[ ] b) L=[R] c)L=[R,A] d)L=[R,A,S] burası daha fazla ilerlemez. R için biter.

Algoritmayı A için tekrar başlatırız. ...
Algoritmayı B için başlatırız. B 'den D ye kadar gelince L=[B,T,E,V,G,U,D] olur. Bu noktada eğer S i seçersek ölü nokta olur ve D ye geri geliriz. T yi seçelim. L=[B,T,E,V,G,U,D,T] olur. Bu noktada döngü tespit edilir ve algoritma durur.

İşletim Sistemlerine Giriş

Ölümcül Kilitlenme (Deadlock)