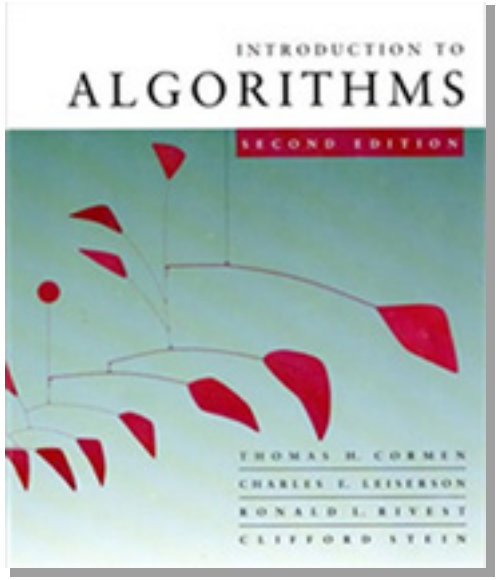
Algoritmalara Giriş

6.046J/18.401J

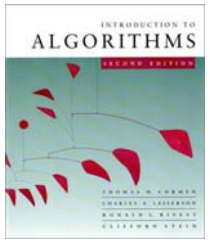


DERS 7

Kıyım Fonksiyonu I (Hashing I)

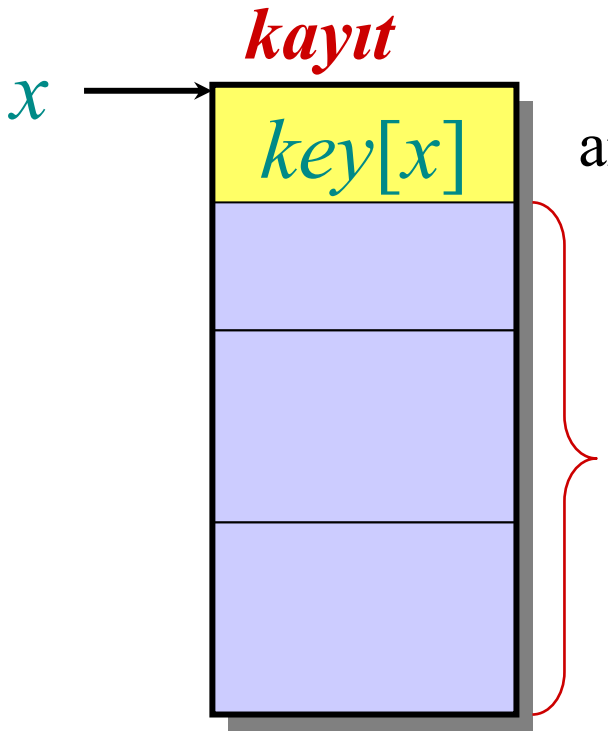
- Doğrudan erişim tabloları
- Çarpışmaları ilmekleme ile çözmek
- Kıyım fonksiyonu seçimi
- Açık adresleme

Prof. Charles E. Leiserson



Sembol-tablosu problemi

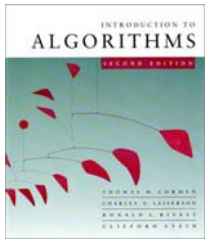
Sembol tablosu S 'nin içinde n ***kayıt*** var:



S 'deki işlemler:

- ARAYA SOKMA(S, x)
- SİLME(S, x)
- ARAMA(S, k)

Veri yapısı S nasıl organize edilmelidir?



Doğrudan erişim tablosu

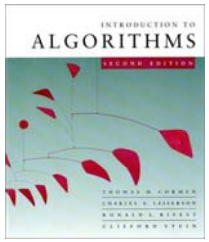
FİKİR: Anahtarların $U \subseteq \{0, 1, \ldots, m-1\}$ setinden seçildiğini ve birbirlerinden farklı olduklarını varsayın. $T[0 \,..\, m-1]$ dizilimini oluşturun:

$$T[k] = \begin{cases} x & \text{eğer } x \in K \text{ ve } key[x] = k \text{ ise,} \\ 0 & \text{diğer durumlarda.} \end{cases}$$

Burada işlemler $\Theta(1)$ zamanı alır.

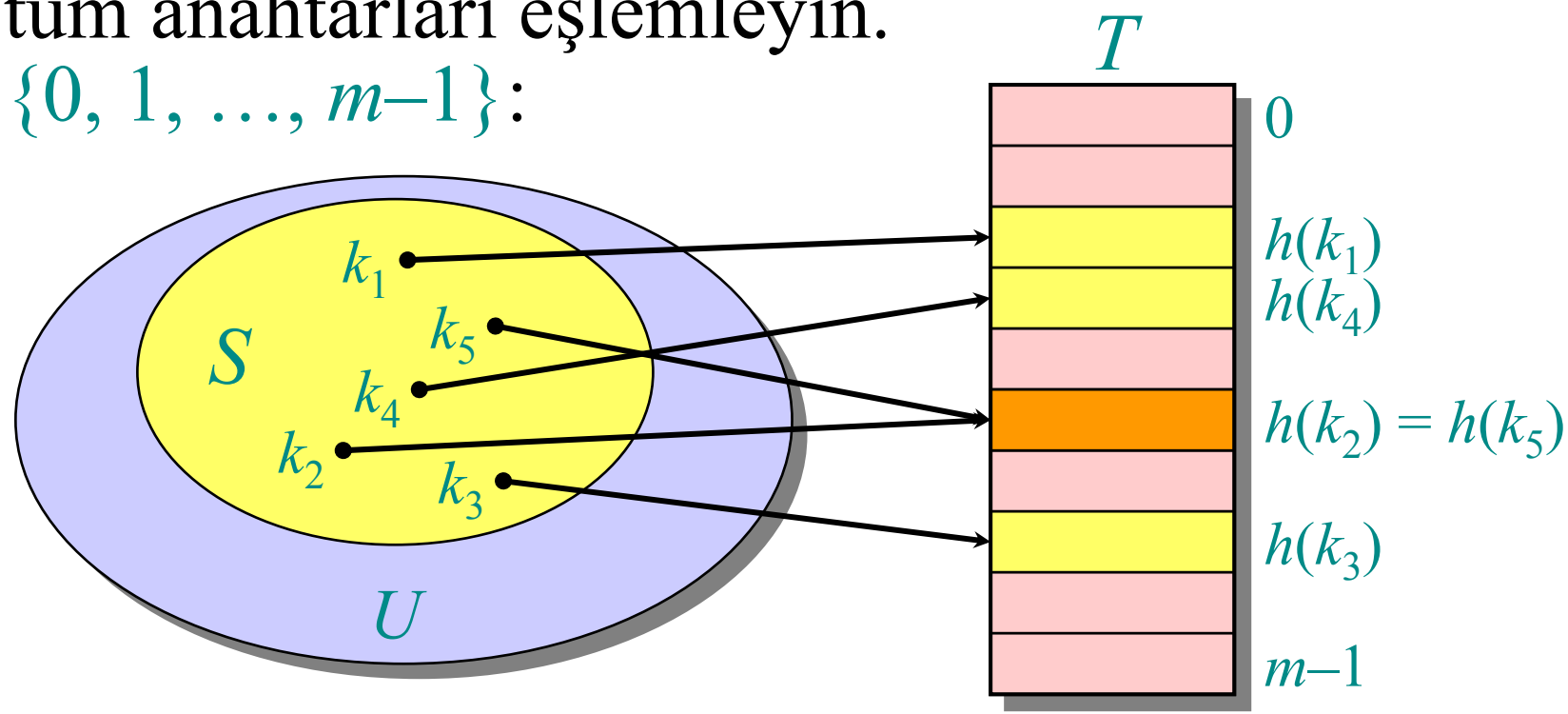
Problem: Anahtarların değer kümesi büyük olabilir:

- 64-bit sayılar (18,446,744,073,709,551,616 farklı anahtarları temsil eder),
- (daha da fazla) karakter dizgisini içerebilir.

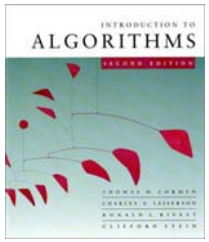


Kıyım fonksiyonları

Çözüm: ***Kıyım fonksiyonu*** h ile U evrenindeki tüm anahtarları eşlemleyin. $\{0, 1, \ldots, m–1\}$:

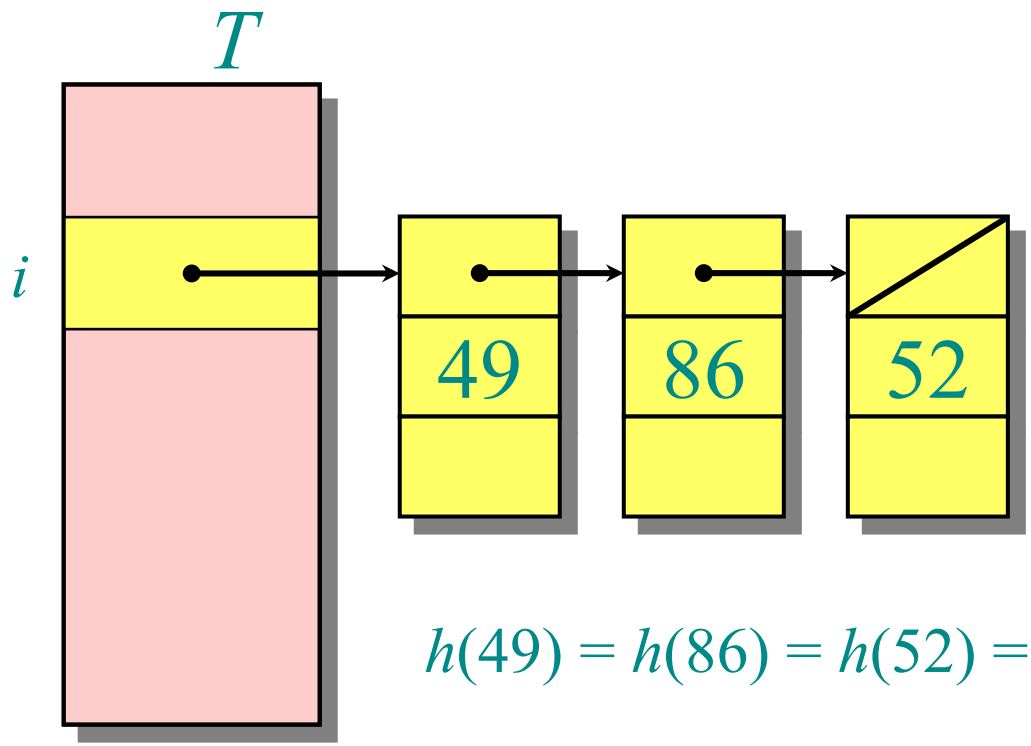


Araya yerleştirilecek kayıt T ' deki dolu bir yuvaya eşlemlendiğinde, bir ***çarpışma*** oluşur.



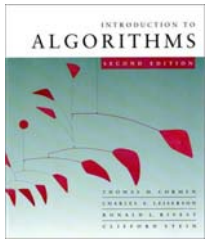
Çarpışmaları ilmeklemeyle çözme

- Aynı yuvadaki kayıtları bir listeyle ilişkilendirin.



En kötü durum:

- Tüm anahtarlar aynı yuvaya kıyımlanır.
- Erişim süresi = $\Theta(n)$ eğer $|S| = n$ ise



İlmeklemede ortalama durum çözümlemesi

Basit tekbiçimli kıyımlama için şu varsayımı yaparız:

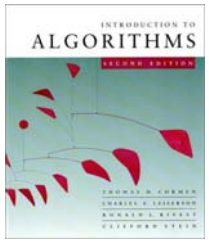
- Her anahtar $k \in S$, T tablosunun her yuvasına diğer anahtarların nereye kıyımlandığından bağımsız olarak kıyımlanır.

n bu tablodaki anahtarların sayısı ve
m de yuvaların sayısı olsun.

T 'nin ***yük oranını*** tanımlarken;

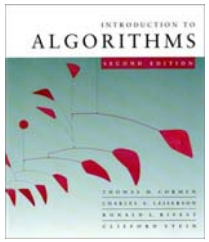
$$\alpha = n/m$$

= yuva başına ortalama anahtar sayısıdır.



Arama maliyeti

Belirli bir anahtar kaydı için ***başarısız*** bir aramadaki beklenen süre $= \Theta(1 + \alpha)$' dır.



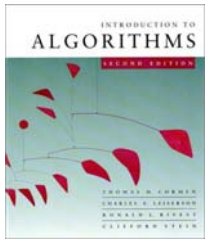
Arama maliyeti

Belirli bir anahtar kaydı için ***başarısız*** bir aramadaki beklenen süre

$= \Theta(1 + \alpha)$.

listeyi arama

kıyım fonksiyonu uygulama ve yuvaya erişim



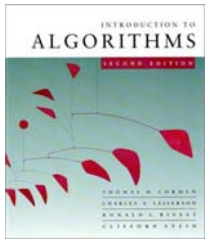
Arama maliyeti

Belirli bir anahtar kaydı için ***başarısız*** bir aramadaki beklenen süre $= \Theta(1 + \alpha)$.

listeyi arama

kıyım fonksiyonu uygulama ve yuvaya erişim

Beklenen arama süresi $= \Theta(1)$, eğer $\alpha = O(1)$ ise, veya eğer $n = O(m)$ ise...



Arama maliyeti

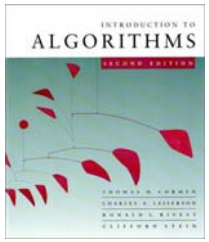
Belirli bir anahtar kaydı için ***başarısız*** bir aramadaki beklenen süre
$= \Theta(1 + \alpha)$.

listeyi arama

kıyım fonksiyonu uygulama ve yuvaya erişim

Beklenen arama süresi $= \Theta(1)$, eğer $\alpha = O(1)$, veya eğer $n = O(m)$ ise..

Başarılı bir arama da aynı asimptotik sınıra sahiptir, ama çok sıkı bir argüman biraz daha karmaşıktır. (Kitaba bakınız.)

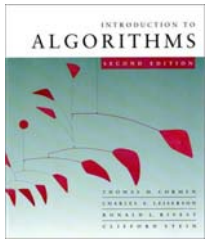


Bir kıyım fonksiyonu seçmek

Basit tekbiçimli kıyımlamanın varsayımını garanti etmek zordur, ama eksikliklerinden kaçınılabildiği sürece pratikte iyi çalışan bazı ortak teknikler vardır.

İstenilenler:

- İyi bir kıyım fonksiyonu, anahtarları tablonun yuvalarına tekbiçimli dağıtabilmelidir.
- Anahtar dağılımındaki düzenlilik bu tekbiçimliliği etkilememelidir.



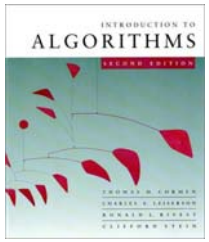
Bölme metodu

Tüm anahtarların tam sayı olduğunu kabul edin ve şöyle tanımlayın: $h(k) = k \bmod m$ (ölçke).

Sakınca: m' yi küçük bir d böleni olacak şekilde seçmeyin. Anahtarlardan çoğu ölçke (modulo) d ile çakışırsa, bu durum tekbiçimliliği olumsuz etkiler.

Uç sakınca: Eğer $m = 2^r$ ise, kıyım fonksiyonu k' nın bütün bitlerine bağımlı bile olmaz:

- Eğer $k = 10110001110110\,10_2$ ve $= 6$ ise, $h(k) = 011010_2$ dır. $h(k)$



Bölme metodu (devam)

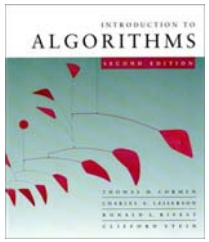
$$h(k) = k \bmod m.$$

m'yi , 2 veya 10' un bir kuvveti olmayacak şekilde ve bilgisayar dünyasında yaygın kullanılmayan asal sayılar arasından seçin.

Rahatsızlık:

- Bazen, tablo boyutunu asal oluşturmak uygun değildir.

Buna rağmen bu metot yaygındır ama bir sonraki metot daha üstündür.



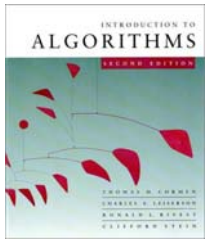
Çarpma metodu

Tüm anahtarların tamsayı $m = 2^r$, ve bilgisayar sözcüklerinin de w-bit olduğunu kabul edin.

$h(k) = (A \cdot k \bmod 2^w)$ rsh $(w - r)$ 'yi tanımlayın,

burada rsh "bit bazında sağa kayma" işlemcisi ve A da $2^{w-1} < A < 2^w$ aralığında tek tamsayı olsun.

- A 'yı 2^{w-1} veya 2^w 'ye çok yakın seçmeyin.
- Ölçke (modulo) 2^w ile çarpma bölmeye oranla daha hızlıdır.
- rsh (bit bazında sağa kaydırma) operatörü hızlıdır.



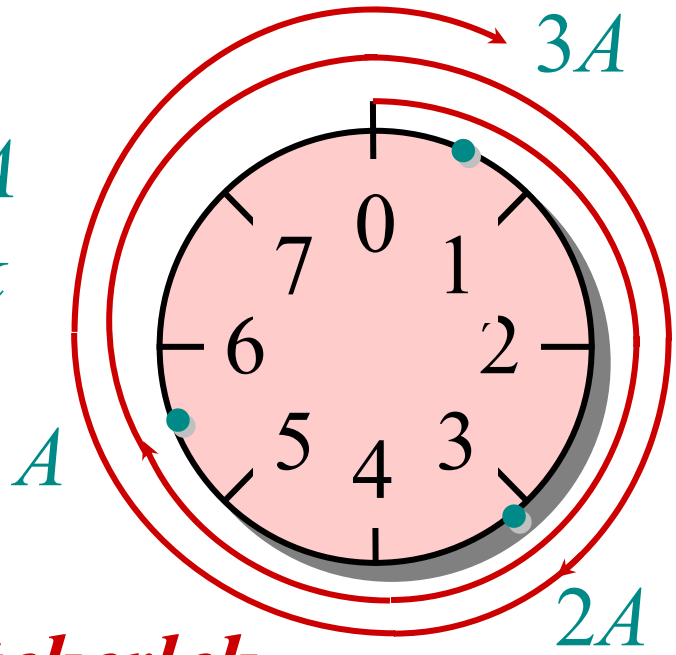
Çarpma metodu örneği

$$h(k) = (A \cdot k \bmod 2^w) \text{ rsh } (w - r)$$

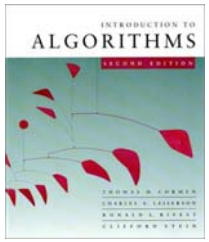
$m = 8 = 2^3$ olsun ve bilgisayarımızda da $w = 7$-bit sözcükler olsun:

$$\begin{array}{rr} & 1\,0\,1\,1\,0\,0\,1 = A \\ \times & 1\,1\,0\,1\,0\,1\,1 = k \\ \hline 1\,0\,0\,1\,0\,1\,0\,0\,1\,1\,0\,0\,1\,1 & \end{array}$$

$h(k)$



Modüler tekerlek

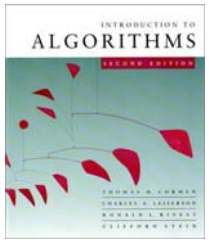


Açık adresleme ile çarpışmaları çözmek

Kıyım tablosunun dışında depo alanı kullanılmaz.

- Araya yerleştirme boş bir yuva bulunana kadar tabloyu sistematik biçimde sondalar.
- Kıyım fonksiyonu hem anahtara hem de sonda sayısına bağımlıdır:

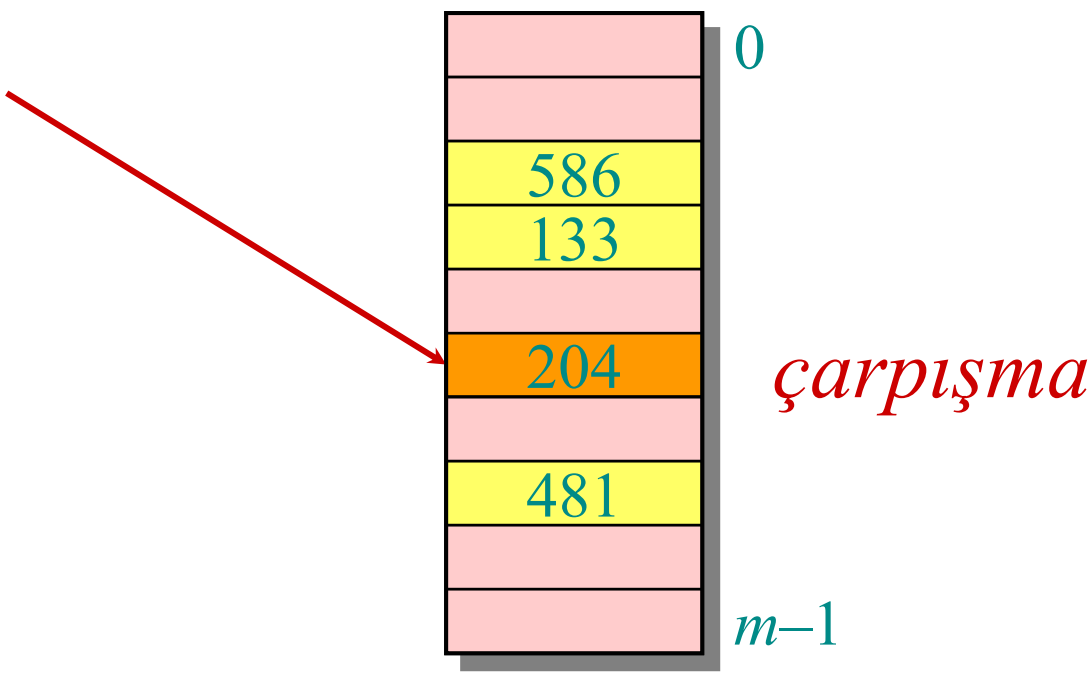
  $h : U \times \{0, 1, \ldots, m-1\} \rightarrow \{0, 1, \ldots, m-1\}$.
- Sonda dizisi $\langle h(k,0), h(k,1), \ldots, h(k,m-1)\rangle$# $\{0, 1, \ldots, m-1\}$' in bir permütasyonu olmalıdır.
- Tablo dolabilir ve silme işlemi zordur, (ama imkansız değildir).

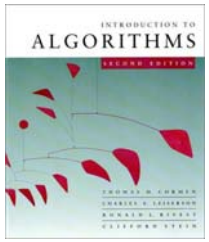


Açık adresleme için örnek

Anahtarı $k = 496$ araya yerleştirin:

0. Sonda $h(496,0)$

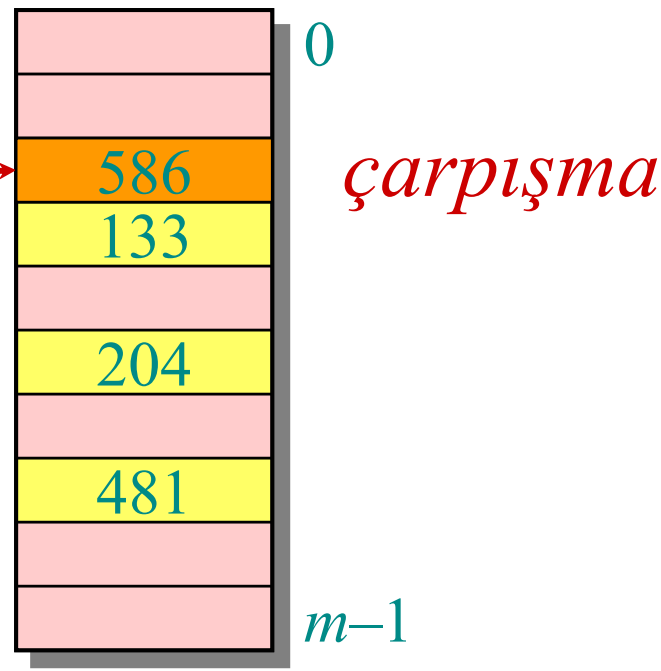


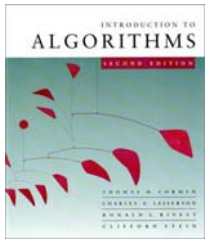


Açık adresleme için örnek

Anahtarı $k = 496$ araya yerleştirin:

0. Sonda $h(496,0)$
1. Sonda $h(496,1)$

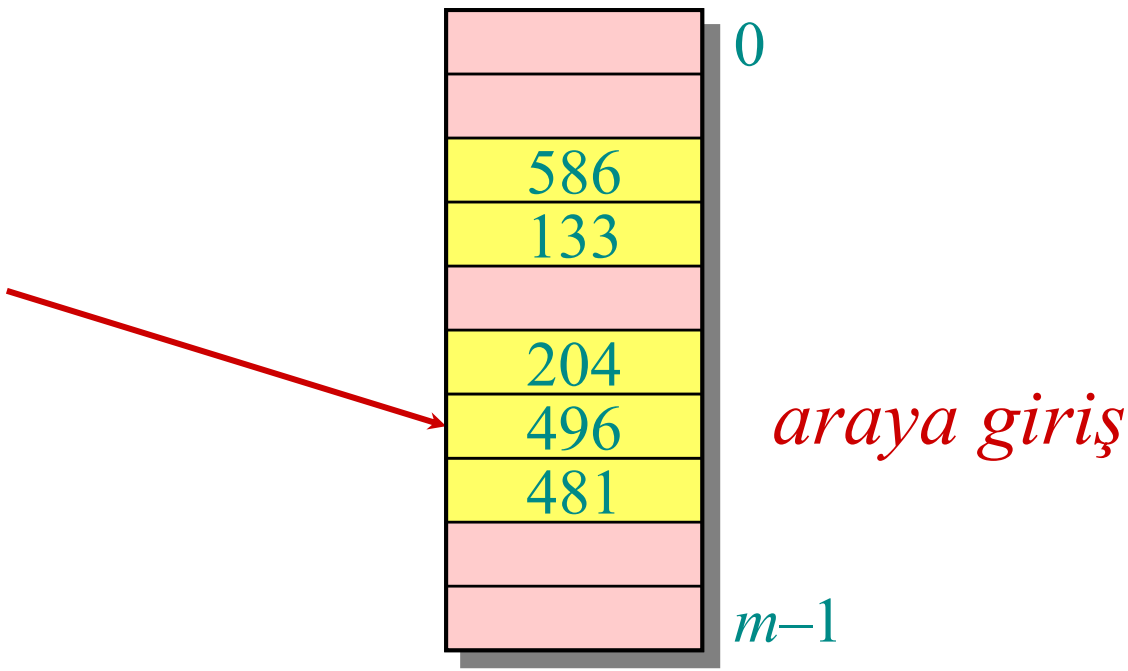


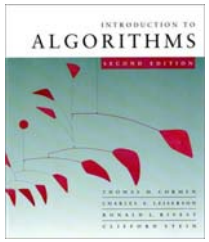


Açık adresleme için örnek

Anahtarı $k = 496$ araya yerleştirin:

0. Sonda $h(496,0)$
1. Sonda $h(496,1)$
2. Sonda $h(496,2)$

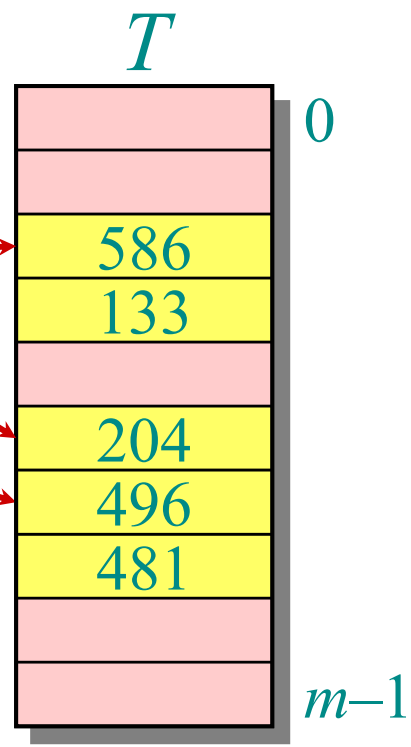




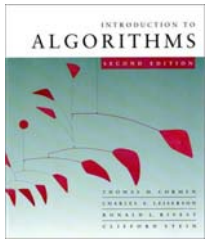
Açık adresleme için örnek

$k = 496$: Anahtarı arama

0. Sonda $h(496,0)$
1. Sonda $h(496,1)$
2. Sonda $h(496,2)$



Arama da aynı sonda dizisini kullanır; anahtarı bulursa başarıyla, boş bir yuvayla karşılaşırsa başarısızlıkla sona erer.



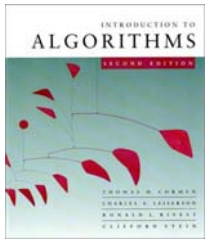
Sonda stratejileri

Doğrusal sondalama:

$h'(k)$, gibi basit bir kıyım fonksiyonu verildiğinde, doğrusal sondalama şu kıyım fonksiyonunu kullanır:

$$h(k,i) = (h'(k) + i) \bmod m.$$

Bu metot basit olmakla birlikte ***asal gruplandırma*** sıkıntı yaratır; dolu yuvalar uzun sıralar oluşturur ve ortalama arama süresi artar. Ayrıca, dolu yuvaların sıra uzunluğu giderek artar.



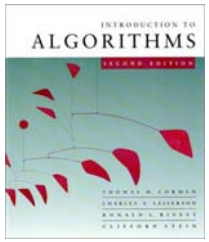
Sonda stratejileri

Çifte kıyımlama

$h_1(k)$ ve $h_2(k)$,gibi iki basit kıyım fonksiyonu varsa, çifte kıyımlama şu kıyım fonksiyonunu kullanır:

$$h(k,i) = (h_1(k) + i \cdot h_2(k)) \bmod m.$$

Bu metot genelde mükemmel sonuçlar verir, ama $h_2(k)$, m 'e göre asal olmalıdır. Bunun bir yolu m 'yi, 2 'nin bir kuvveti yapmak ve $h_2(k)$ 'yı sadece tek sayılar üretecek şekilde tasarlamaktır.

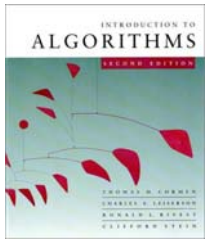


Açık adresleme çözümlemesi

Tekbiçimli kıyımlama varsayımı yaparız:

- Her anahtar, kendi sonda dizisinin herhangi bir $m!$ permütasyonuna sahip olabilir.

Teorem. Yük oranı $\alpha = n/m < 1$ olan açık-adresli bir kıyımlama tablosu verildiğinde, başarısız bir aramadaki sondaların beklenen sayısı en çok $1/(1-\alpha)$' dır.

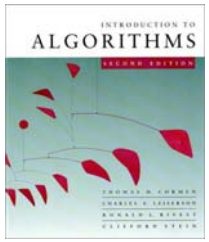


Teoremin kanıtlanması

Kanıt.

- En az bir sondalama mutlaka gereklidir.
- n/m olasılığıyla, ilk sonda dolu bir yuvaya gider ve ikinci sondalama gerekli olur.
- $(n–1)/(m–1)$ olasılığıyla, ikinci sonda dolu bir yuvaya gider ve üçüncü sondalama gerekli olur.
- $(n–2)/(m–2)$ olasılığıyla, üçüncü sonda da dolu bir yuvaya gider, v.b.

Gözlemle: $\frac{n-i}{m-i} < \frac{n}{m} = \alpha$; $i = 1, 2, \ldots, n$ için...



Kanıtlama (devam)

Bu nedenle, beklenen sonda sayısı:

$$1+\frac{n}{m}\left(1+\frac{n-1}{m-1}\left(1+\frac{n-2}{m-2}\left(\cdots\left(1+\frac{1}{m-n+1}\right)\cdots\right)\right)\right)$$

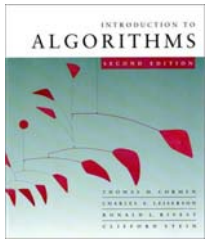
$$\le 1+\alpha(1+\alpha(1+\alpha(\cdots(1+\alpha)\cdots)))$$

$$\le 1+\alpha+\alpha^2+\alpha^3+\cdots$$

$$=\sum_{i=0}^{\infty}\alpha^i$$

$$=\frac{1}{1-\alpha}\,.\ \square$$

Kitapta daha kesin bir kanıtlama ve başarılı aramaların bir analizi de var.



Teoremin açılımları

- Eğer α bir sabitse, açık adresli bir kıyım tablosuna erişim sabit zaman alır.
- Eğer tablo yarı doluysa, beklenen sonda sayısı $1/(1–0.5) = 2$' dir.
- Eğer tablo % 90 doluysa, beklenen sonda sayısı $1/(1–0.9) = 10$' dur.