

File Organization

physical file = "... .txt"

logical file = code

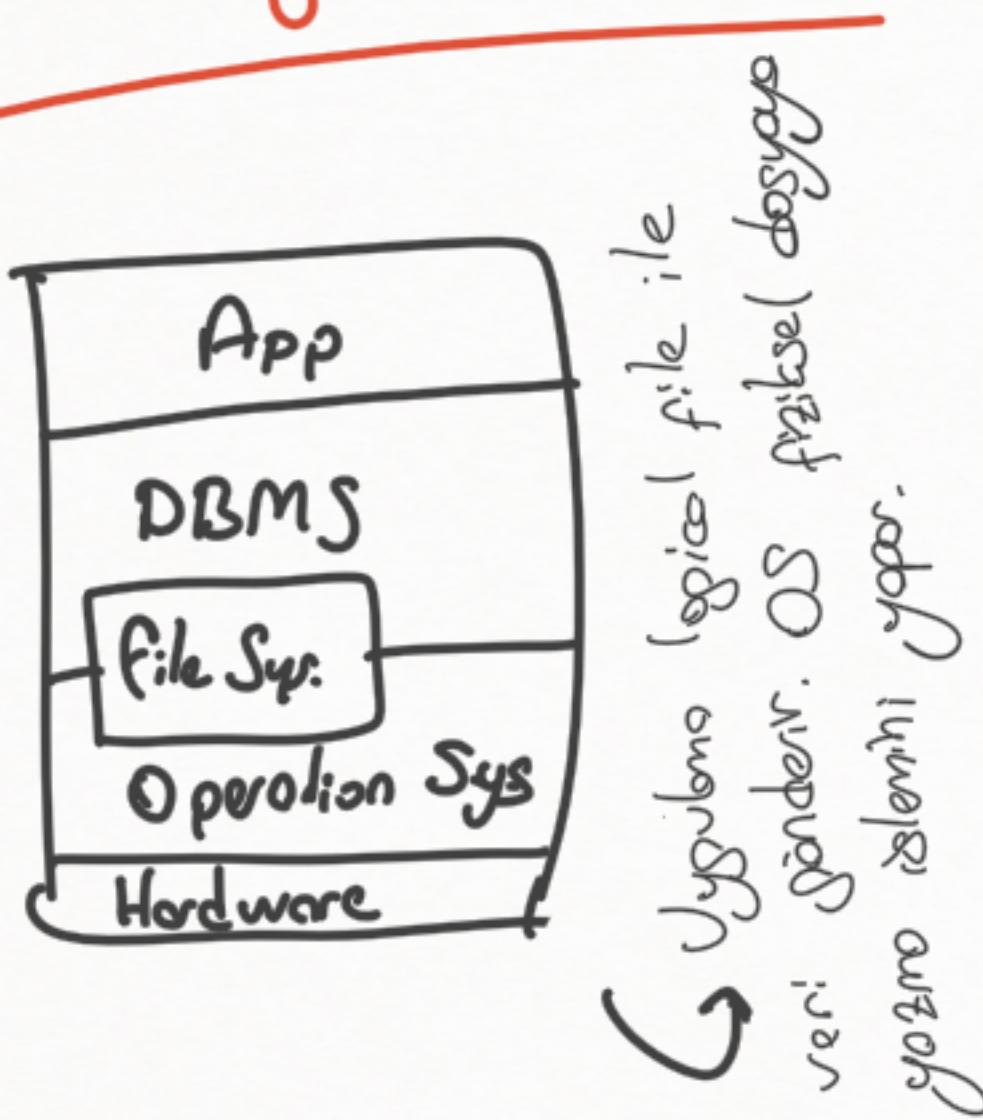
Track Capacity = Trackbeli sektor sayısı * sektor kapasitesi

Cylinder Capacity = Silinderdeki track sayısı * Track cap.

Drive Capacity = Silinder sayısı \times S. kapasitesi

Silindr sayısı = Yüzeydeki

file Organization



File Structures.

- Storage
 - Organization
 - Access
 - > Processing

Amos; Bağlantılı obr
verileri gruplara ve veri
alma / işleme yapısını hızlandırmak
Yani "hız".

(RAM) (Hard - Disk)
⇒ Primary & Secondary

- fast (+) / Pers:
 - Small (-) / "
 - Volatile (-) / "

✓

gerici
elektrot kavurkende)
gider.

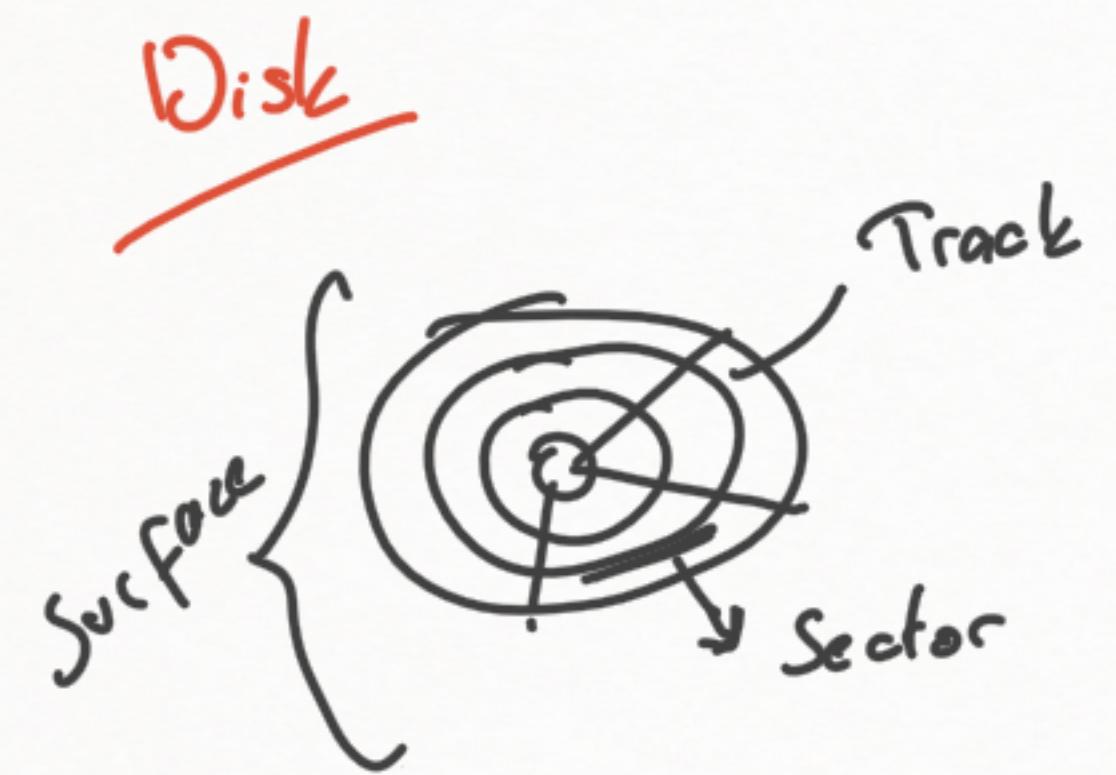
→ Data manipulated in RAM.
Stored in Disks / tapes ...

$\rightarrow \text{H}_2$ forti yatkınlık $4 \cdot 10^6$ dir.
(4 milyon)

/ # myfile.txt (physical file)
file *days (logical file) *

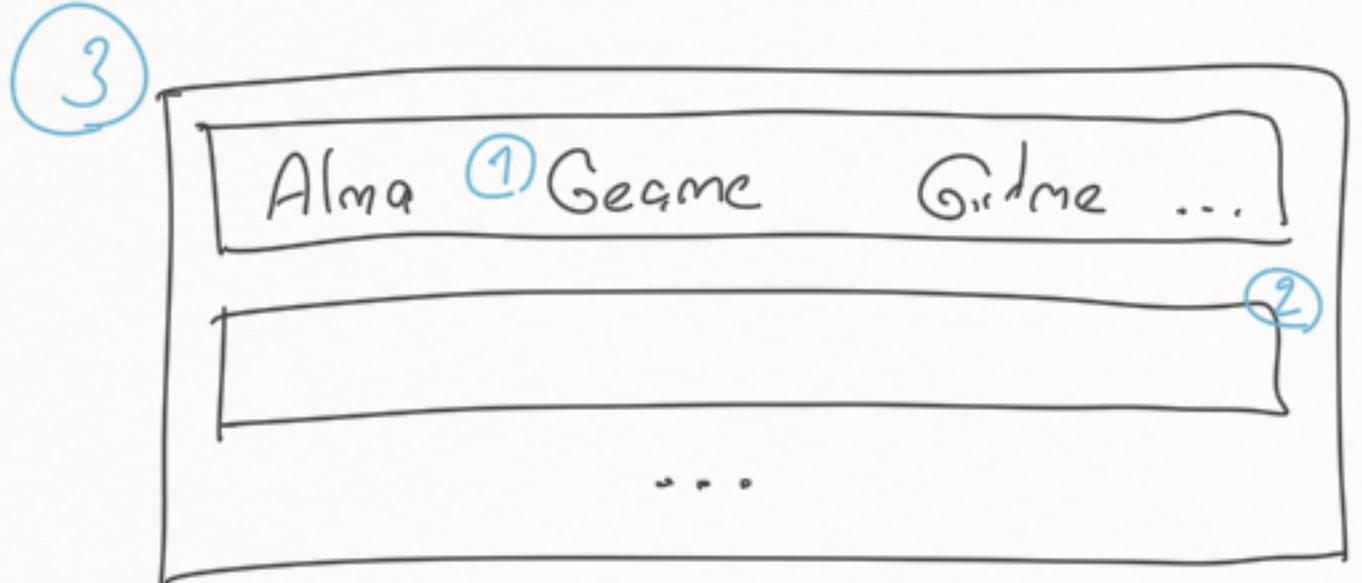
Galiso Presib;

- Veriler SS (Secondary Storage) 'den
MM'ye (main memory) alınır. Bu
sayede işlem hızlı gerçekleştir.



• En ufak okunumabilir
unité Sector'dür.

file organization
Collision - final



Gazi Uni - Sequential and Direct file (2).

2.)

Sequential Search

- ① Field
- ② Record
- ③ File

- Ortalama $n/2$ probe gereklidir
- Kayıtlar sıralıysa dört az prob
- n boyunda denklemi vardır

Binary Search

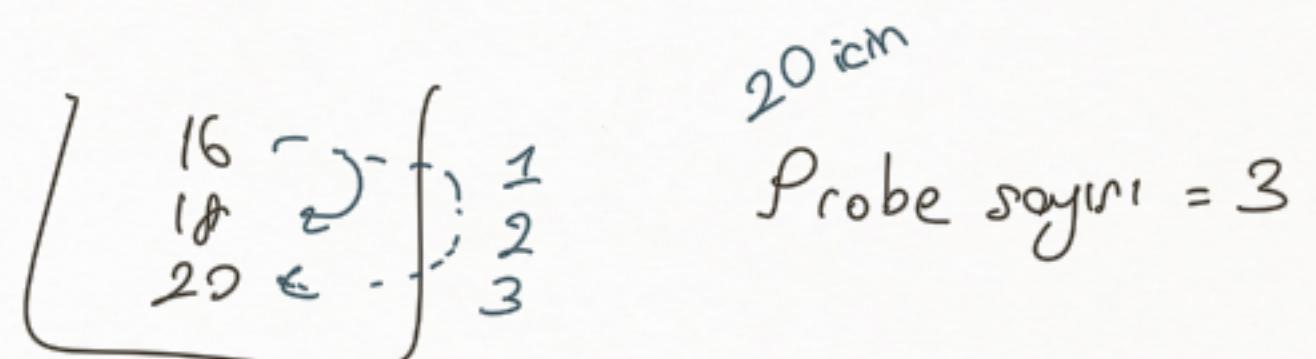
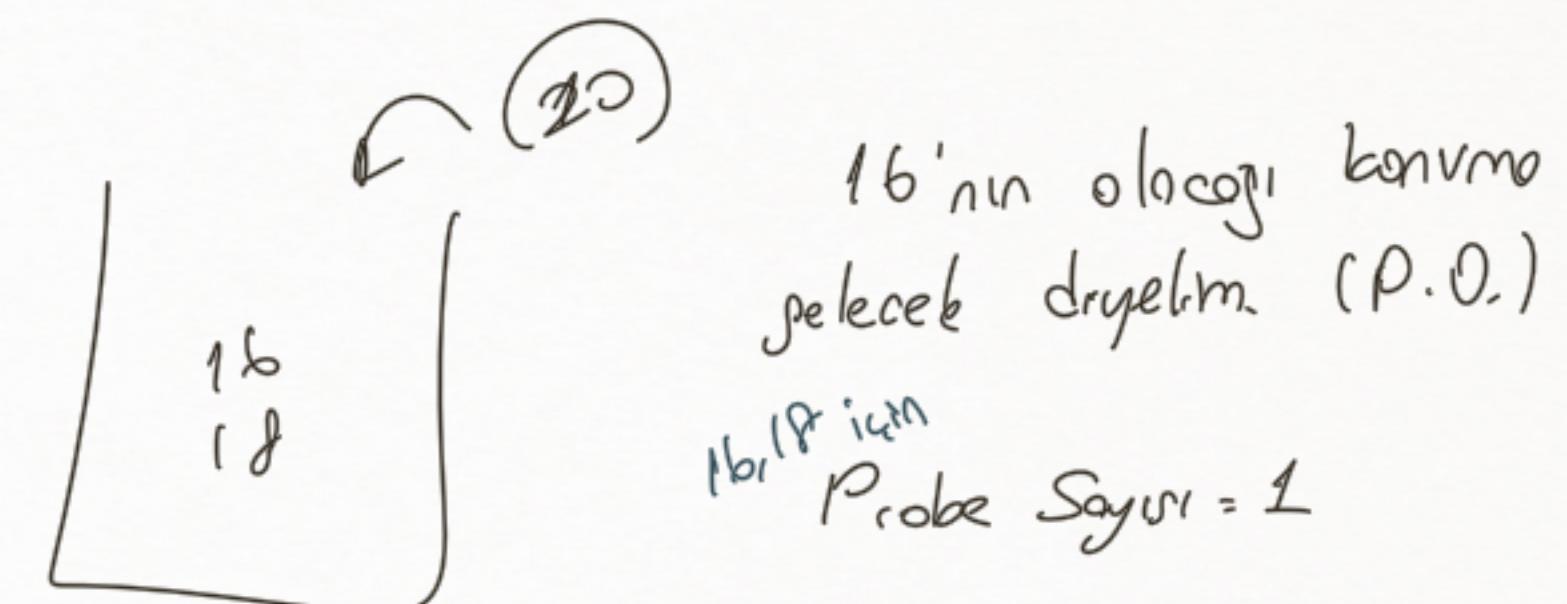
- Aranan ortadan aranmaya basar
- İlk defasında kayıtların yarısı elenir.
- $\log_2 n$ boyunda denklemi yazılır.

* Packing Factor:
$$\frac{\# \text{ record stored}}{\# \text{ storage location}}$$

* Collision : Çakışma

* Probe: Uyarıyi kaydede kadar geçen adım sayısı

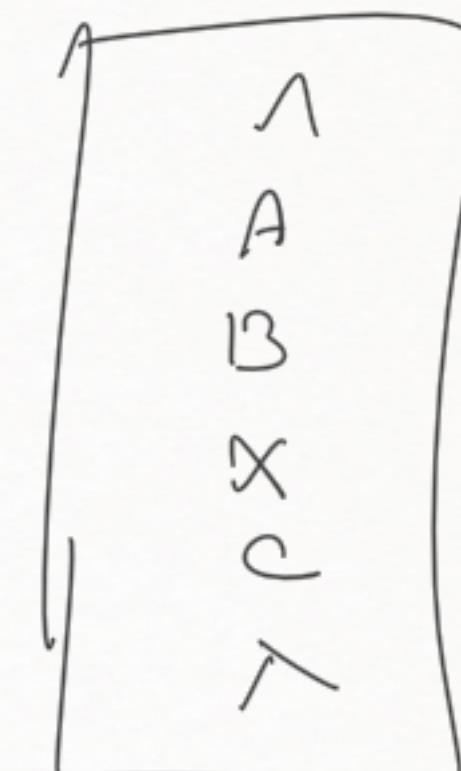
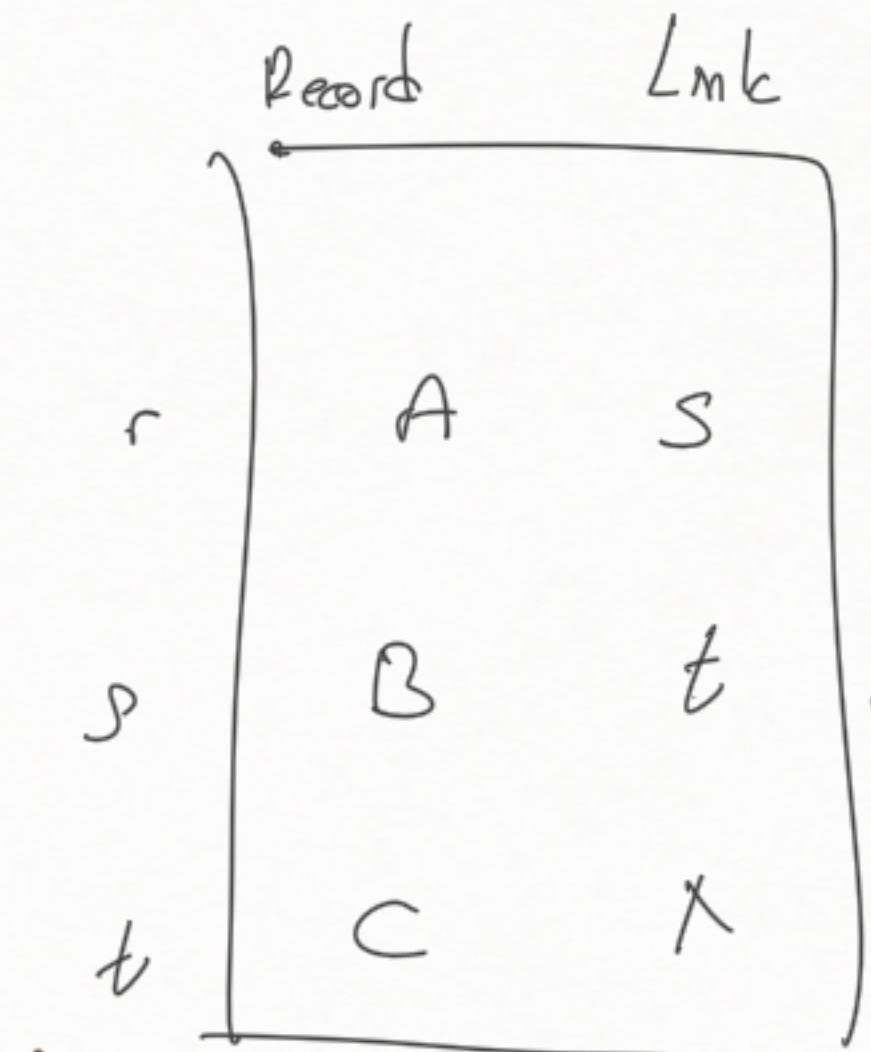
~~YNS
Ene
Sor~~



Collision Resolution with Link

→ Aynı adresi sahip
birler arası bağlantı
oluşturulur.

- Bağlantılar içi farzeden
yer更换ir.



Bağlantılar

* Coalesced: Birlesik

Coalesced Hashing

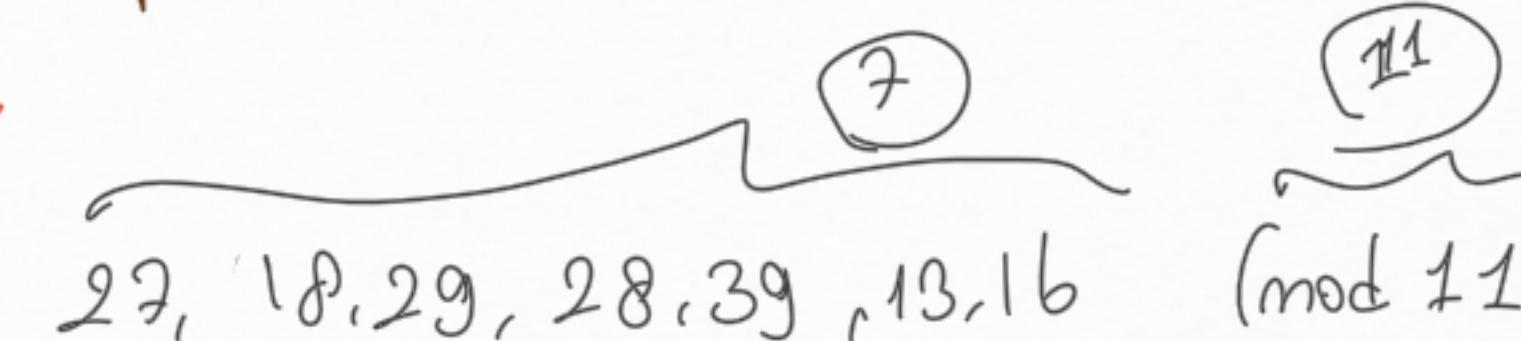
→ Synonym (es anlamlı) zincir
elemanları arasında bağlantı yopılır.

→ listenin sonundaki eleman
silinen yerine gelir.

③ 18 silinirse; sonun yerine 29
gelir.

fast insertion
standart coalesced
hashing

LISCT



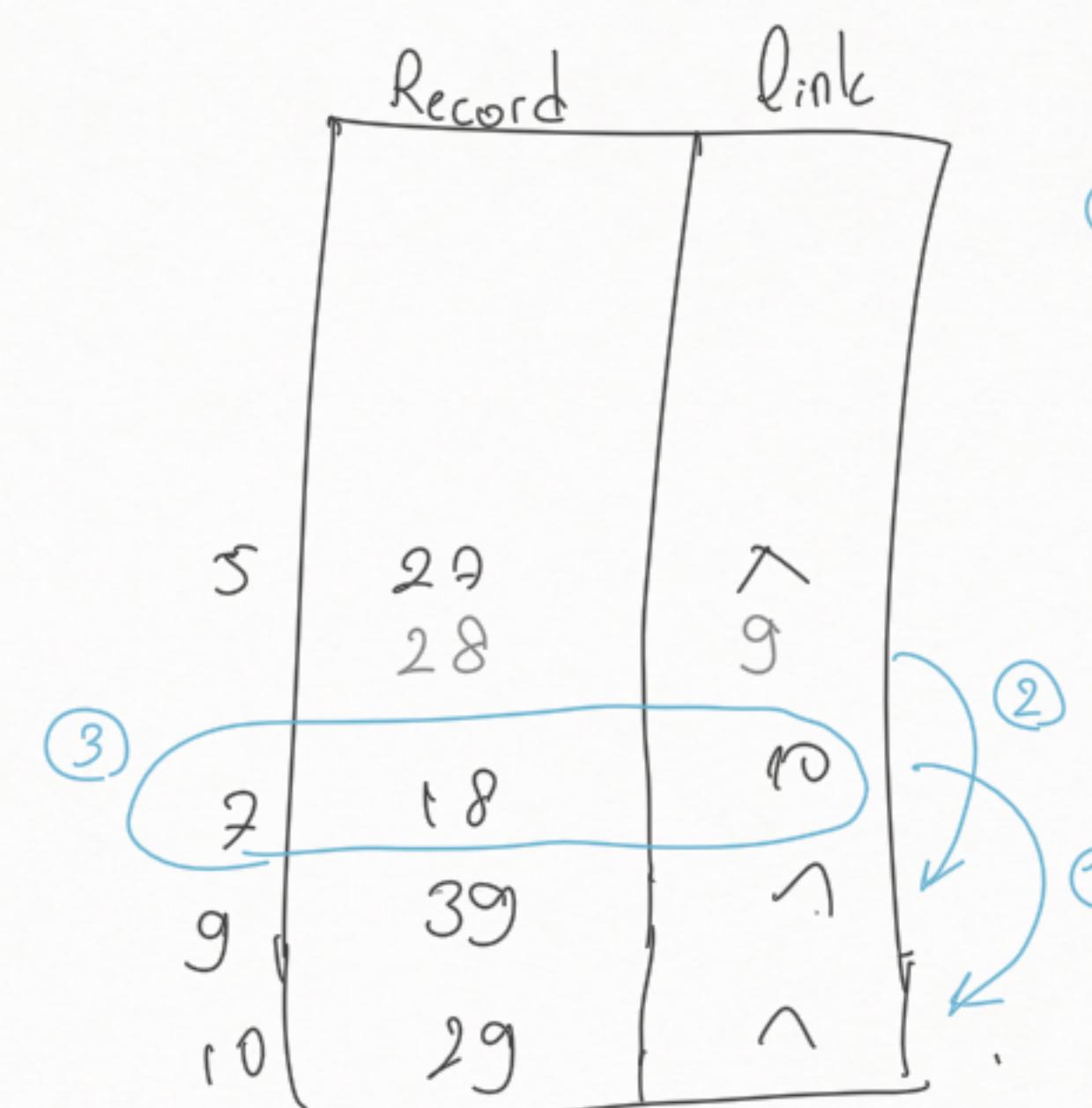
$$\text{Packing Factor} = \frac{7}{11}$$

• Overflow olmaz.

$$\begin{aligned} ① & 29 \equiv 7 \pmod{11} \\ 18 & \equiv 7 \pmod{11} \end{aligned}$$

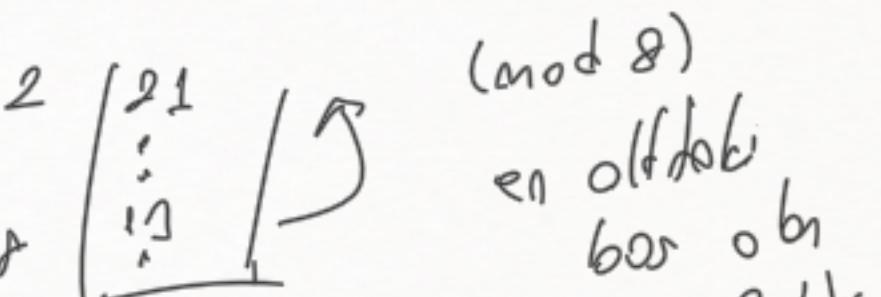
ilk 18 eklenip, 18'ye 29
için link verilir. 29 en alto yozilir.

(eğer alto yer yoksa, boş en alto
yozilir.)



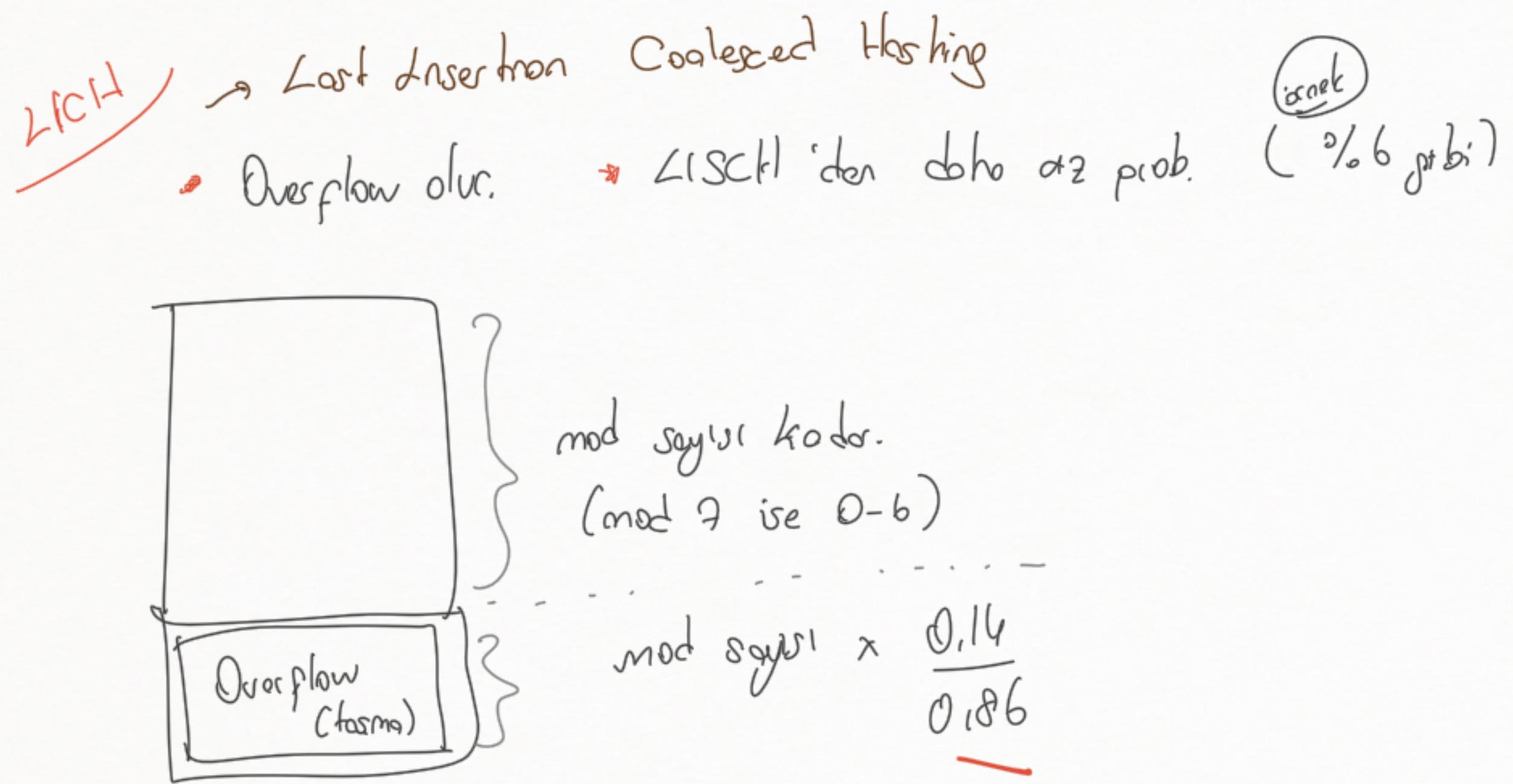
* P.F., carlesme yapılırsa, hiz ardr. (Silme karisik.)

* Çekirdek bulutları boyutları başta olusa ardr.



Digerleri de bkr.

en olusturul
bos ol
yer 2'dr.



- LISCHE Early Insertion ...
- REISCH Random early ...
- RISCH Random Late ...
- BLISCH Bidirectional ...
- BEISCH

!
 L: Late (en sonra)
 E: Early (en basina sonra)
 S: Standard (Overflow yok)
 R: Random (?)
 B: Bidirectional
 (Bir olun bire witten)

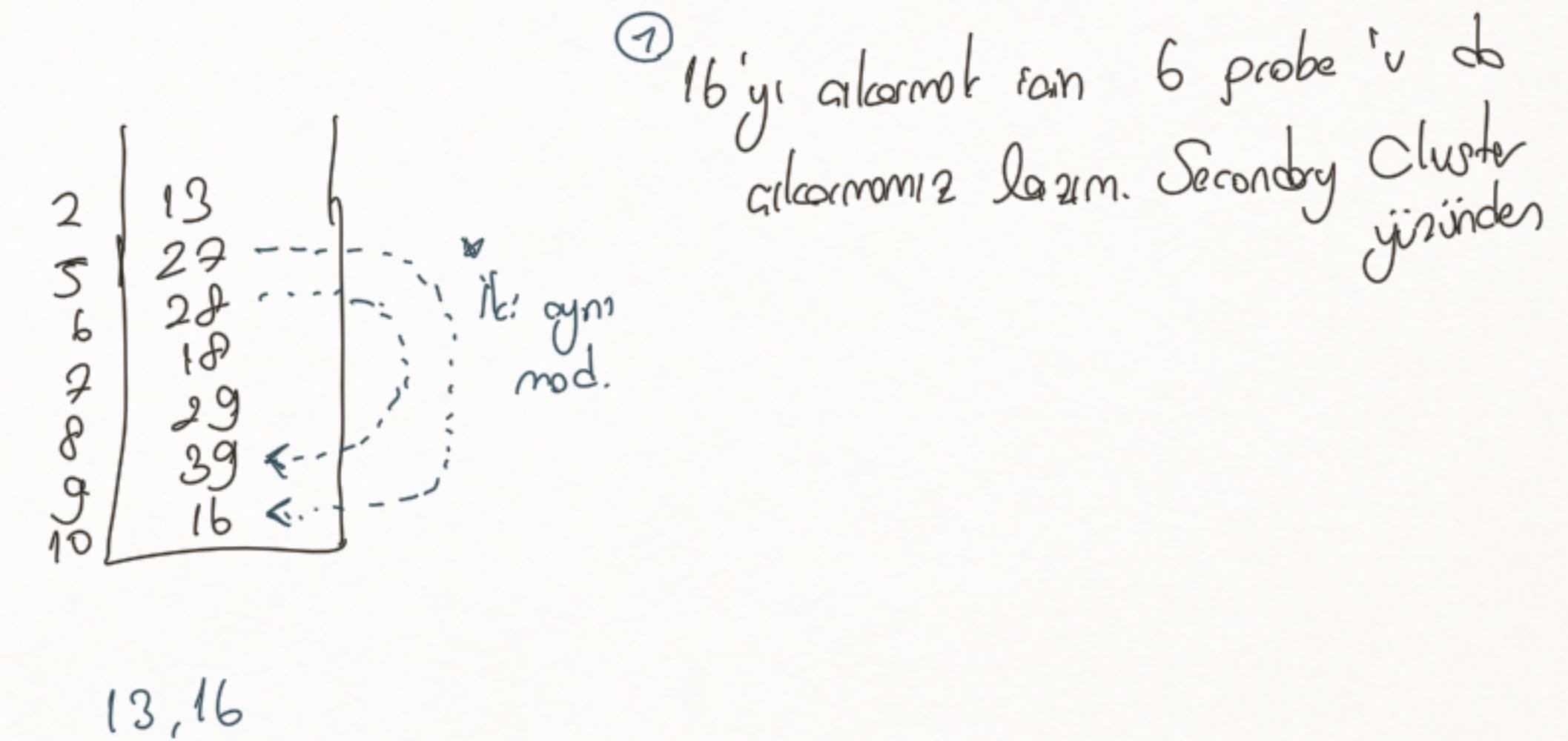
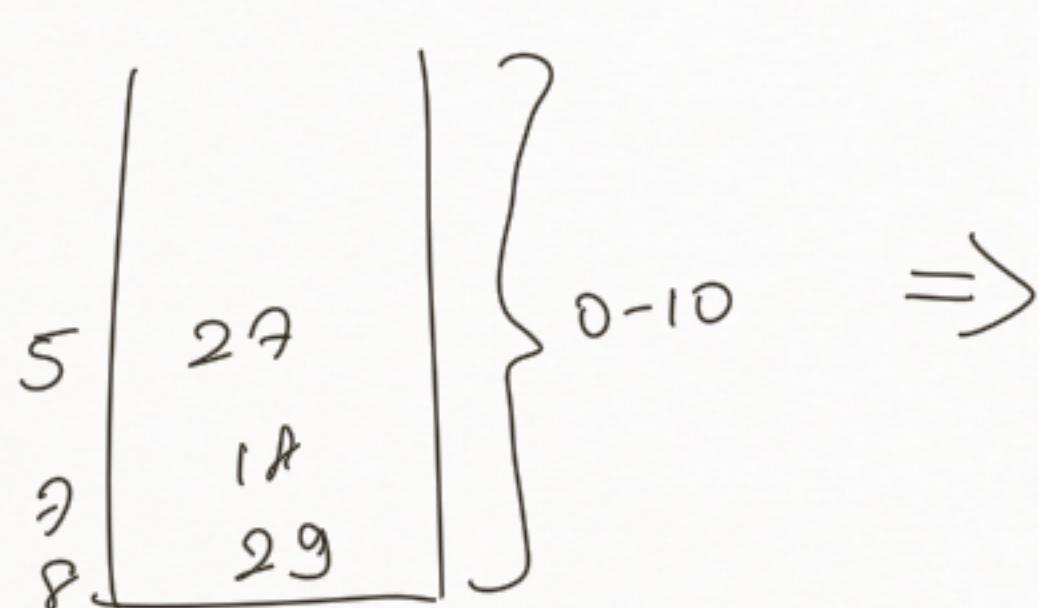
(3 Sonu)

Progressive Overflow

* En dardık olanı

- ① Secondary Clustering: 2 veya daha fazla bayit aynı probe adresi arasında dayanır.
- Primary Clustering: Aşk fazla bayit aynı home adres'e sahipse.
- Silmede yerine tombstone eklenir.

Ex) Keys = 27, 18, 29, 28, 39, 13 ve 16 (mod 11)



27, 18, 29

28, 39

* Retrieval = bilgi arkanı

→ Tek tek indirimiz için probe sayısı çok fazladır. Ort: 2.3 probe

* Practical = kullanışlı

→ * Bilgi alırmak çok zordur. Secondary Strope'a erişim kolisi

olduğu için P.O. * kullanılsı değildir.

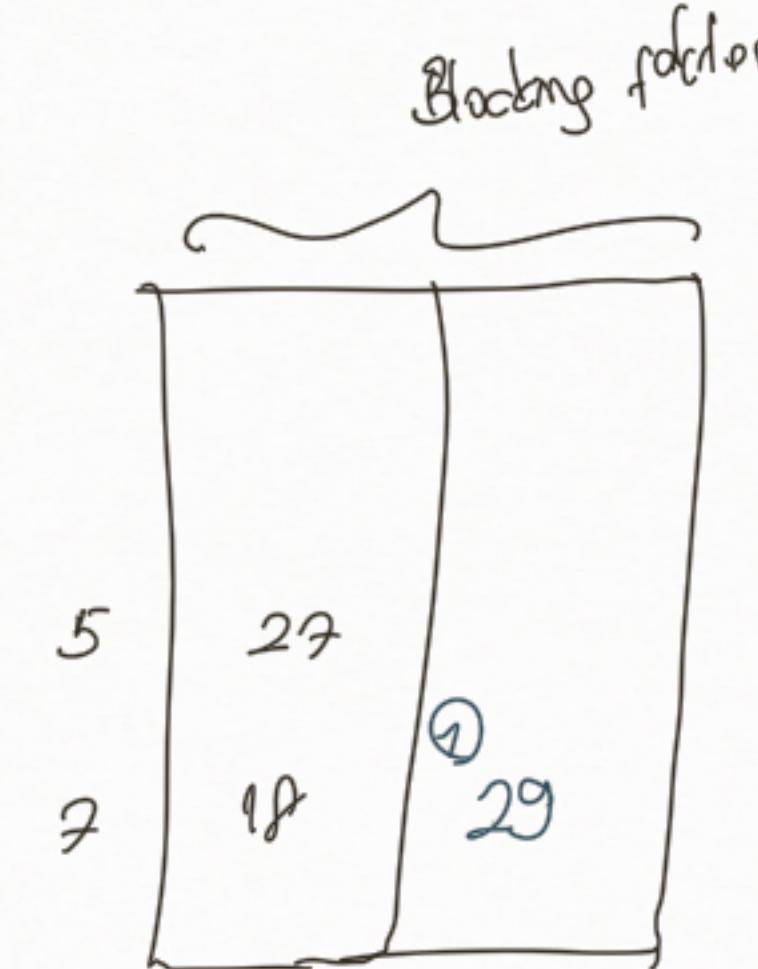
(Coalesced Collision icon)
Ort: 1.6

Bucket

Keys: 27, 18, 29, 28, 39, 13 ve 16

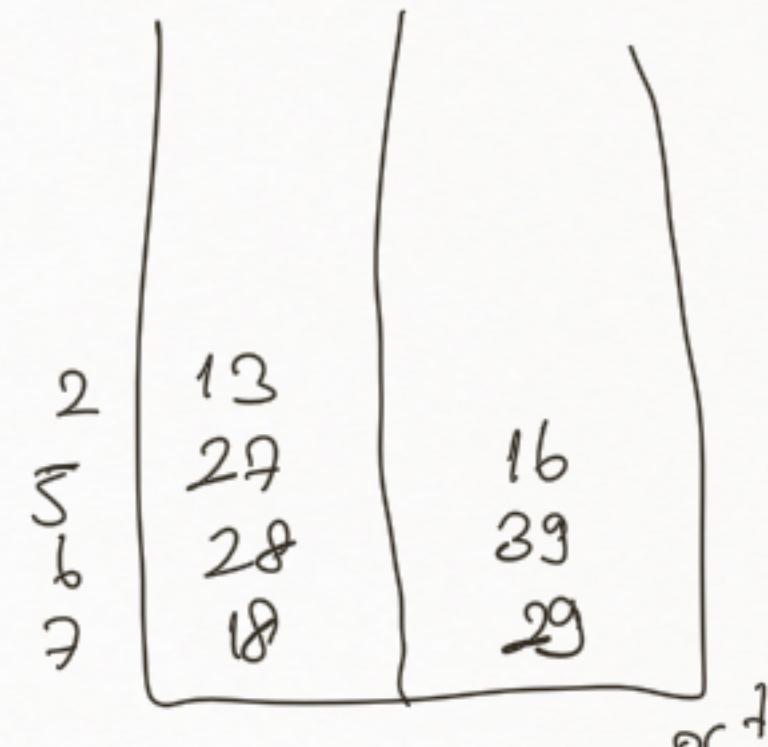
$$\text{Hash(key)} = \text{key mod } 11$$

$$\text{Blocking Factor} = 2$$



- ① Blocking factor 2 olursa iki 1 yerde 2 var yaşılabılır.

\Rightarrow



Probe = 1

* Collision olmadığı için

* Oran = Quotient

27, 18, 29

④ Blocking factor

$$\text{Packing Factor} = 7 / 11 \times 2 = 32\%$$

* Progressive Overflow olasıdır
7/11
= 64%

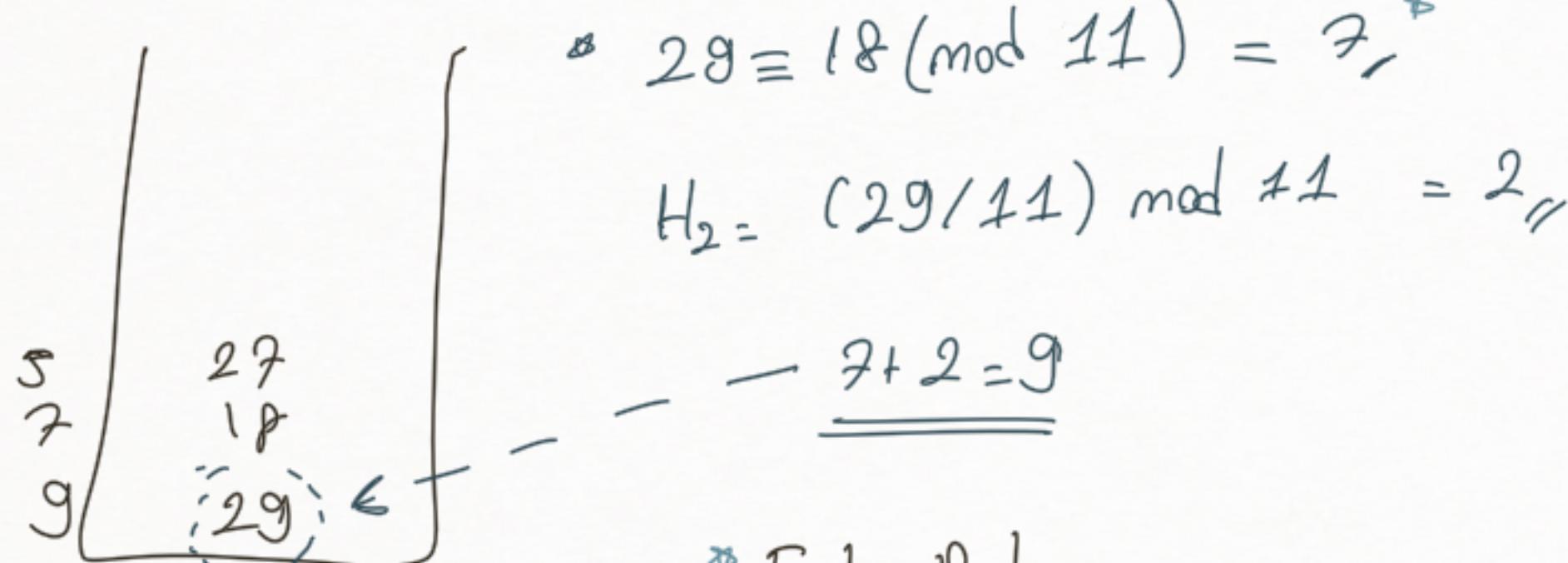
Linear
+ Quotient

\rightarrow P. O. ile aynı tek farklı hashing
var.

$$\rightarrow H_1(\text{key}) = \text{key mod } P$$

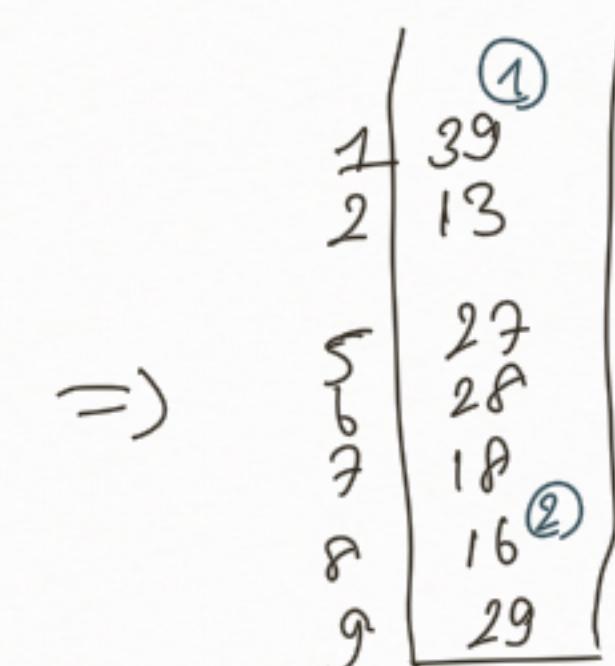
$$\rightarrow H_2 = \text{Quotient} (\text{key}/11) \text{ mod } 11 \text{ (artı)} \quad \text{aynı ornek}$$

Ex



* Extra Probe

$$\# \text{ Probes} = 2 + 1 + 1 = 4$$



$$① 39 \equiv 28 \pmod{11} = 6$$

$$H_2 = (39/11) \text{ mod } 11 = 3$$

$$6+3=9 \quad \text{(yine dolu devam)}$$

$$9+3=12 = 1 \pmod{11}$$

* Extra Probe

$$\# = 1 + 3 + 1 + 1 = 6$$

Total = 13 probe

$$13/7 = 1.8 \text{ ad.}$$

4 probe

$$② 16 \equiv 27 \pmod{11} = 5$$

$$H_2 = 1$$

$$5+1 = \text{dolu}$$

$$6+1 = \text{dolu}$$

$$7+1 = 8$$

Brent's Method

- Lineer Quotient gibi ama dinamik metoddur.
- Sadece elemen eklenmede kullanılır.

• 27, 18, 29, 28, 39, 13, 16

• $H_1(\text{key}) = \text{key} \bmod 11$

• $H_2 = \text{Quotient}(\text{key} / 11) \bmod 11$

| | |
|---|----|
| 5 | 27 |
| 7 | 18 |
| 9 | 29 |

$$* 28 \equiv 18 \pmod{11} = 7$$

$$H_2 = (29 / 11) \bmod 11 = 2$$

$$7+2=9 \text{ yeni konum.}$$

$$\# \text{ probes} = 1+1+2=4$$

27, 18, 29 *
(5) (7) (2)

Sonuç

$$\boxed{\text{Total Probe} = 4+4+4=12}$$

$$\text{Orf. probe} = \underline{12 / 7 = 1.7}$$

| | |
|---|----|
| 2 | 13 |
| 5 | 29 |
| 6 | 39 |
| 7 | 18 |
| 8 | 28 |
| 9 | 29 |

28, 39, 13
(6) (6) (2)

Dinamik method : hər keçid etibaribirlər.

Statik Method : Yerlestirilen hər keçid etibariləməz.

Bos gelir.

$$* 39 = 28 \pmod{11} = 6 \quad \checkmark$$

$$H_2 = 39 / 11 = 3 \quad H_2 = 28 / 11 = 2$$

$$6+3=9 \text{ dolu} \quad 6+2=8 \text{ (bos)}$$

$$\# \text{ Probes} = 2+1+1 = 4$$

Bos geldigi ian 28'i basirdik.

Bos Geldi!

| | |
|---|----|
| 0 | 27 |
| 2 | 13 |
| 5 | 16 |
| 6 | 39 |
| 7 | 18 |
| 8 | 28 |
| 9 | 29 |

$$* 16 = 27 \pmod{11} = 5 \quad \checkmark$$

$$H_2 = 16 / 11 = 1 \quad H_2 = 27 / 11 = 2$$

$$5+1=6 \text{ dolu}$$

$$6+1=7 \text{ dolu}$$

$$7+1=8 \text{ dolu}$$

$$8+2=11=0 \text{ bos}$$

$$\# \text{ Probes} = 4$$

Binary Tree

- Sadece ekleme yöntemidir, veri olmamış gibi dir.
- Brond's method yapısına benzer, daha gelişmiş.

- 29, 18, 29, 28, 39, 13, 16, 41, 17, 19
- $H_1(\text{key}) = \text{key} \bmod 11$
- $H_2 = \text{Quotient} (\text{key}/11) \bmod 11$



| | |
|---|----|
| 5 | 27 |
| 7 | 18 |
| 9 | 29 |
| | |

27, 18, 29
(5) (7) (7)

$$* 29 \equiv 18 \pmod{11} \equiv 2$$

$$H_2 = 28/11 = 2$$

29 → 7 (18)
g (bos)
(7+2)

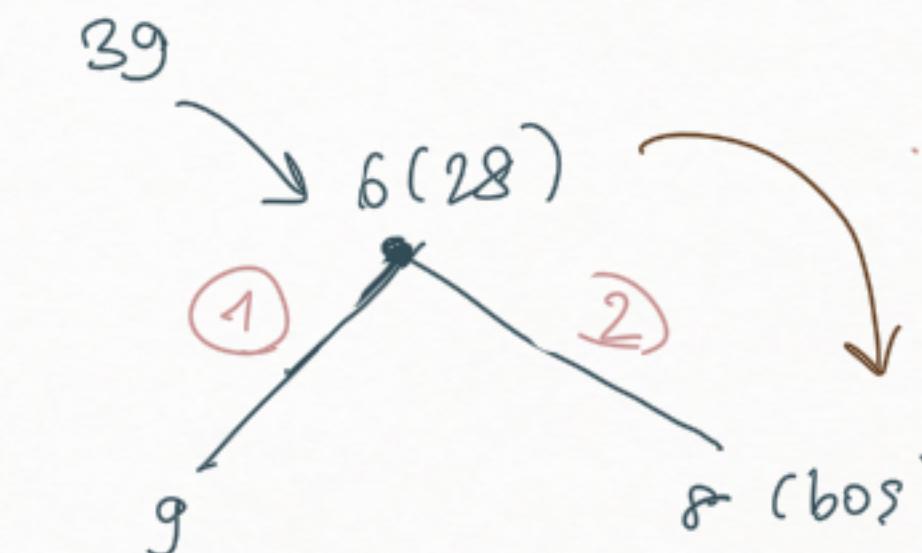
| | |
|---|----|
| 2 | 13 |
| 5 | 27 |
| 6 | 39 |
| 7 | 18 |
| 8 | 28 |
| 9 | 28 |
| | |

28, 39, 13
(6) (6) (2)

$$* 28 \equiv 39 \pmod{11} = 6$$

$$\textcircled{1} H_2 = 39/11 = 3$$

$$\textcircled{2} H_2 = 28/11 = 2$$



→ 28 8'e tasınır,
39 onun yerine gelir.

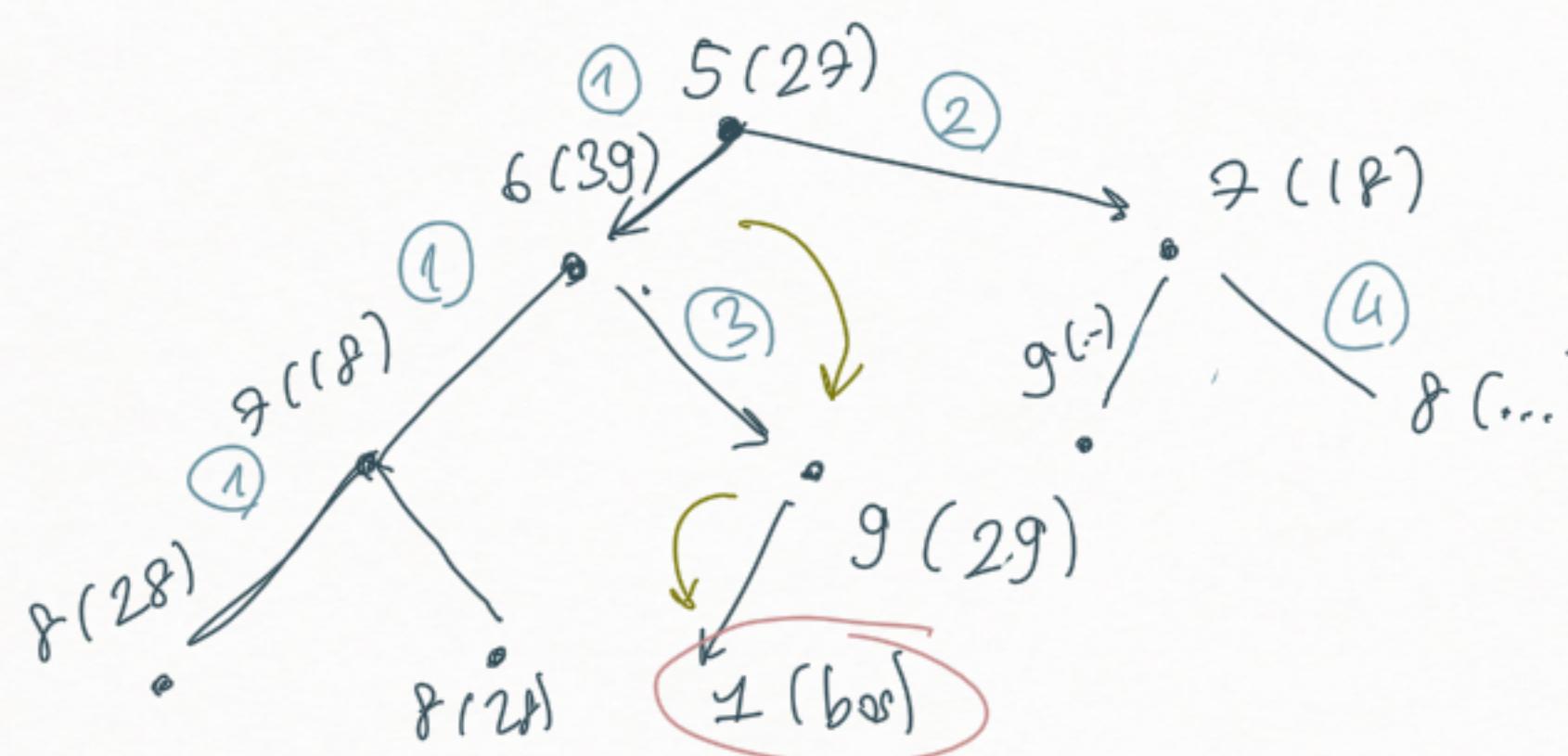
! Sağa geçişlerde
tasınma olur.

| | |
|---|----|
| 1 | 39 |
| 2 | 13 |
| 5 | 27 |
| 6 | 16 |
| 7 | 18 |
| 8 | 28 |
| 9 | 29 |
| | |

16 (5)

$$* 16 \equiv 27 \pmod{11} = 5$$

$$\textcircled{1} H_2 = 16/11 = 1 \quad \textcircled{2} H_2 = 27/11 = 2$$



→ Solu geçikçe referans
değismez. Sağda geçerse yeni düğümü
referans alırız. (H_2 ian)

- ! 16 → 6'ya geçti (Sol geçis) $\textcircled{1}$ Sağ ian referans düğüm:
 - 39 → 9'a geldi (Sag geçis) $\textcircled{2}$ → Referans = 27
 - 39 → 1'e geldi (Sol geçis) $\textcircled{3}$ → Referans = 39
 - $\textcircled{4}$ → Referans = 18
- 1 sağdan döşendi. 29 → 1 olurdu.

Direct file Organization
linkisiz

- Daha düşük performers, az yer.
- Progressive Overflow, Linear Quotient, Brent's method

linkli

- Computed Chaining ile link adresi kayit edilmez.
- Probe chain 'de adres yerine offset kullanılır.
- Birkaç bit ile adres bilgisine ulaşılır. (Pseudo Link)

① Bölüm, oran

c.c
ct

| | |
|----|---------------------------|
| 5 | 27 |
| 6 | 28 |
| 7 | 18 |
| 8 | 29 |
| 10 | 39 |
| | 28, 39, 13 (6) (6) (2) |

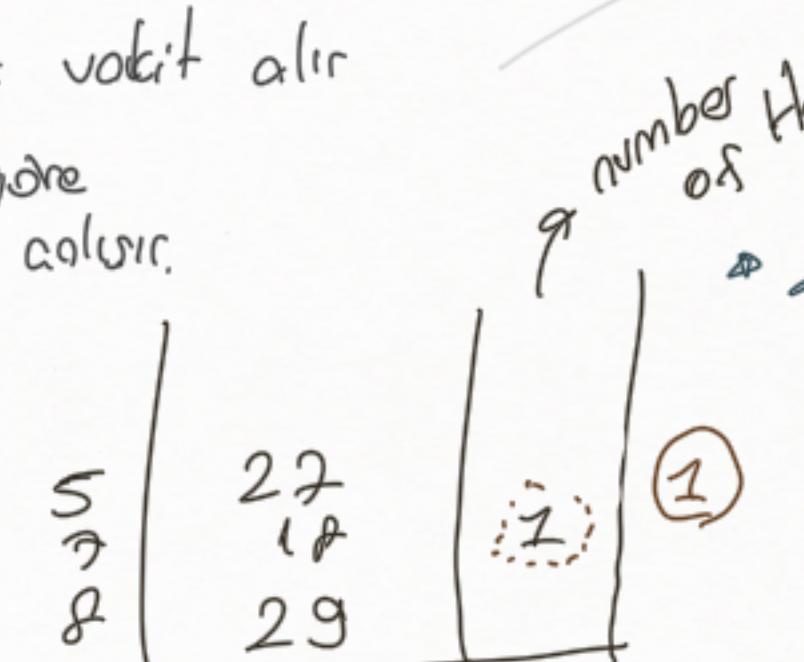
27, 18, 29, 28, 39, 13, 16, 38, 53

Hash key 11

$$;(\text{key}) = \text{Quotient} \quad (\text{key} / 11) \bmod 11$$

Eklenme - Silme çok uygulatılır

* Diğer linklere göre
daha hızlı olur.



$$* H_2 = 28 / 11 = 2^0 \quad \left\{ \begin{array}{l} 2 \text{ tane} \\ H_2 \text{ dolu} \end{array} \right.$$

$6 + 2 = 8$ dolu
 $8 + 2 = 10$ boş

27 18 29
(5) (7) (7)

diger sayfa

4. Sayf
gazi in
Computed Chaining

→ Coalesced Chain olmasından etkilenen kaydın home
adresine yazılımış kayıtlar yeni bir adrese taşınır. Sonuç olarak;
Tüm kayıtlar aynı home adresine sahiptir.

①
→ Probe Sayısı = Probe Chain Dalandığı Pozisyonu

→ Bir sonraki adres, etkilenen veya obranın
değil, o adresinki kayda göre bulunur.

$$\begin{aligned} & \rightarrow 29 \equiv 18 \pmod{11} = 7^0 \quad \left\{ \begin{array}{l} 1 \text{ kere} \\ H_2 \text{ kullanıldı.} \end{array} \right. \quad (1 \text{ yazıyoruz}) \\ & H_2 = 18 / 11 = 1^0 \quad \left\{ \begin{array}{l} 1 \text{ kere} \\ H_2 \text{ kullanıldı.} \end{array} \right. \quad (18 yonca) \\ & 7 + 1 = 8 \text{ boş} \end{aligned}$$

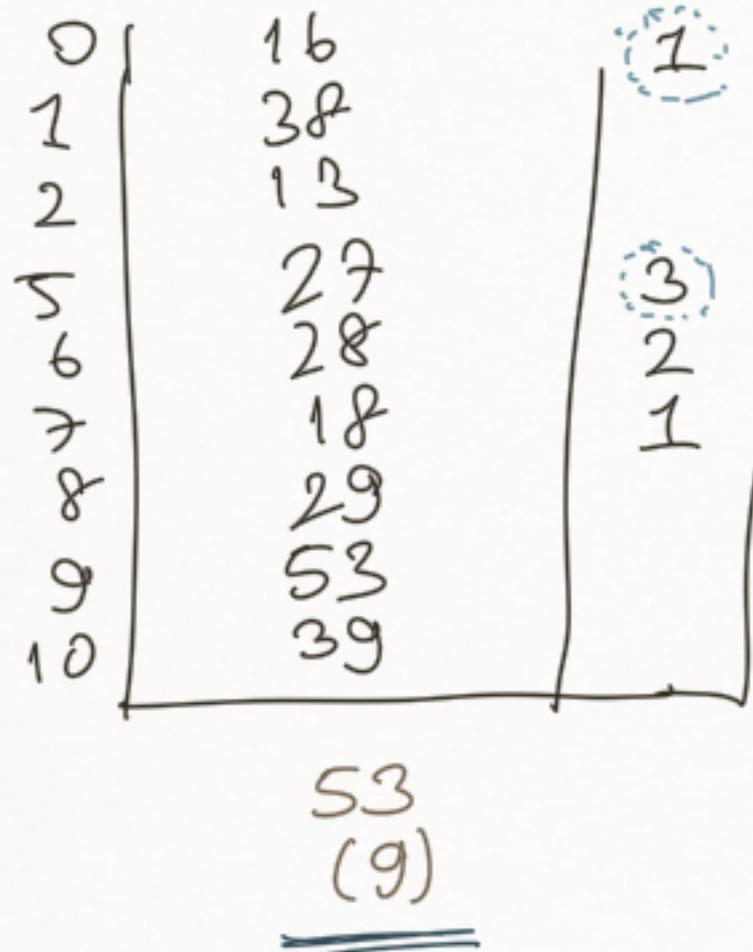
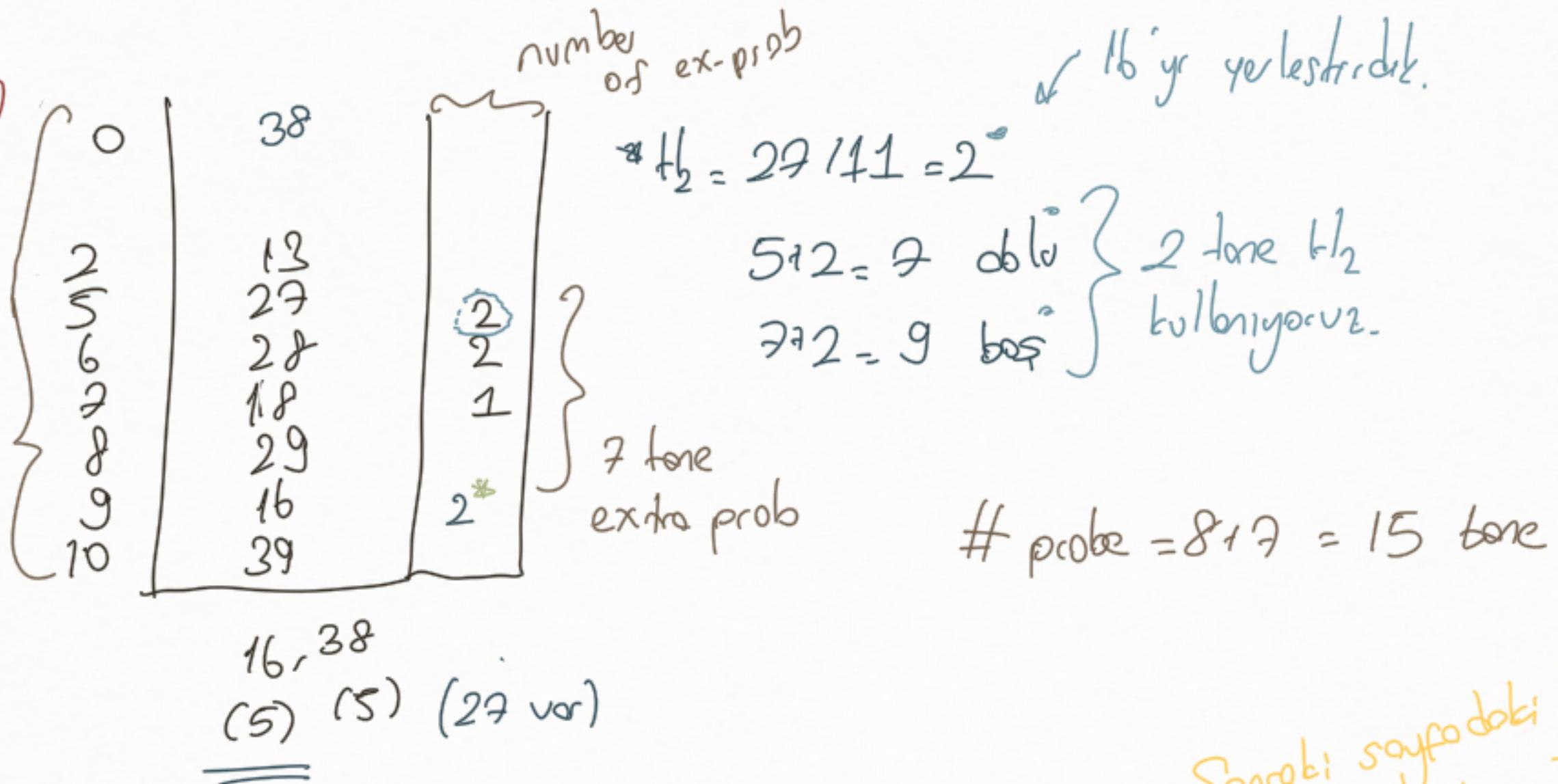
→ Ayrıca 1 extra probe sayısıdır.

$$④ \# \text{ probe} = 3 + 1 = 4$$

Compound
chains

Devoni

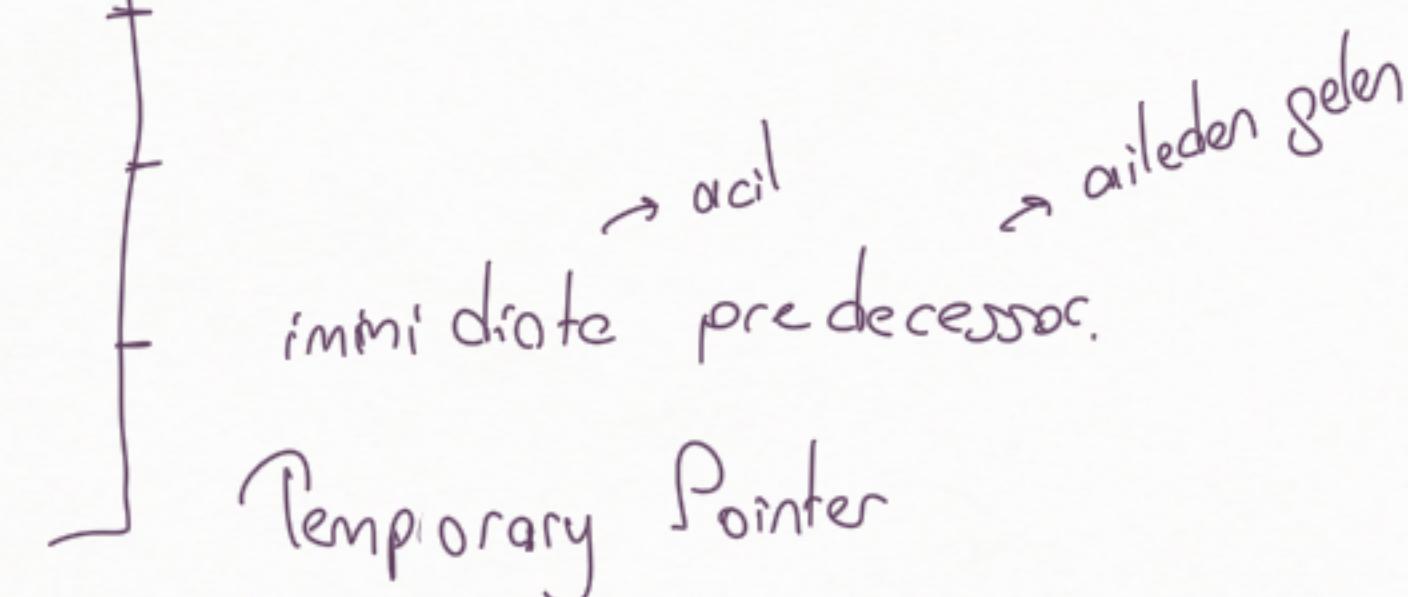
8 tone
prob



Seçim Önceliği

Home Address

öncelik
azaltır.



(53 ion bu oldu.)

Sonraki sayfadaki
ünlem ile ilişkili



Günku:
oldugundan eski bir 2mcr var.
2mcrin ucuna (16'ya) gideriz.

27 1 2

27 ion H_2 bulunur (Önceden 2:di)

sayısi 3 oldu.

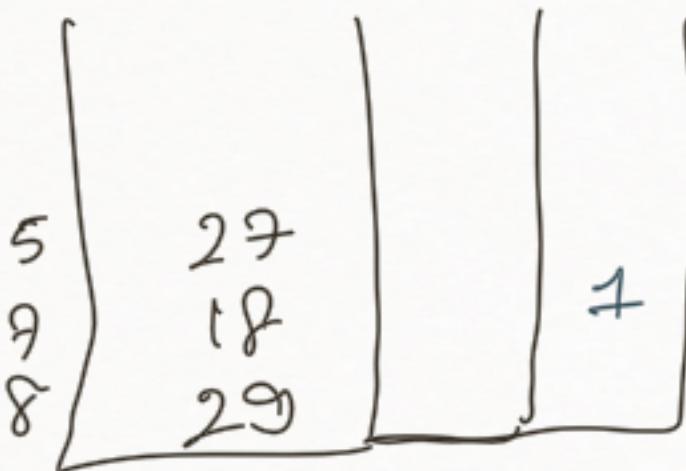
16 ion H_2 bulunur sayısi 1 oldu. (Önceden 2:di)

✓ 38 ion

* 27'de 2 kez H_2 yapıyoruz ($5+2+2$) 9
9'da 16 var.
 $H_2 = 16/11 = 1$
 $\begin{cases} 9+1=10 \text{ dolu} \\ 10+1=11=0 \text{ bos} \end{cases}$
* 2 tone H_2 yapıldı.

Multiple chains

→ Fazladan g hashingimiz olmaz olur.



$27, 18, 29$
 $(5, 7)$ $(7, 0)$ $(7, 1)$

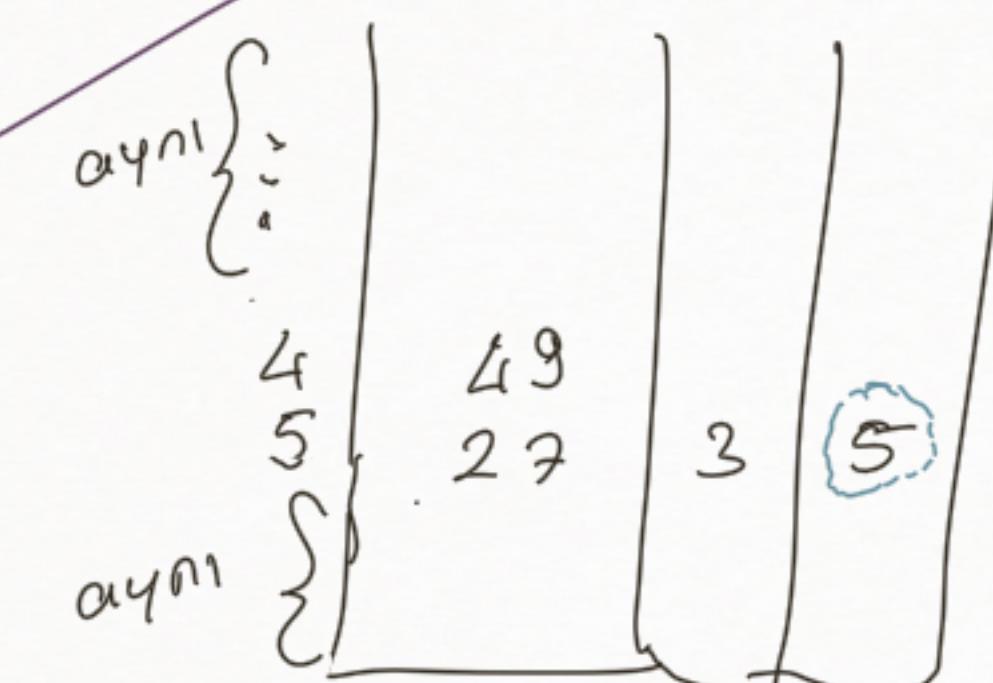
$$18 \equiv 29 \pmod{11} = 7$$

$$H_2 = 18/11 = 1$$

$$7+1=8 \cdot \left(\begin{array}{l} \text{ek} \\ \text{probe sayısı 1} \end{array} \right)$$

! $29 \equiv 1 \pmod{2}$ olduğundan 1
boşesine yazıyoruz. (Home'a bookmarks'ınız)

Son hâline
49 eklersek



ayrınlı

! $49 \equiv 1 \pmod{11}$ ve 27 iin 7 no'lu

sütun boş olduğu için, diğer verilere ellemiyoruz.

- - - - - $49 \equiv 27 \equiv 5 \pmod{11}$ -. . . Eğer dolu olsaydı:

$$H_2 = 27/11 \pmod{11} = 2$$

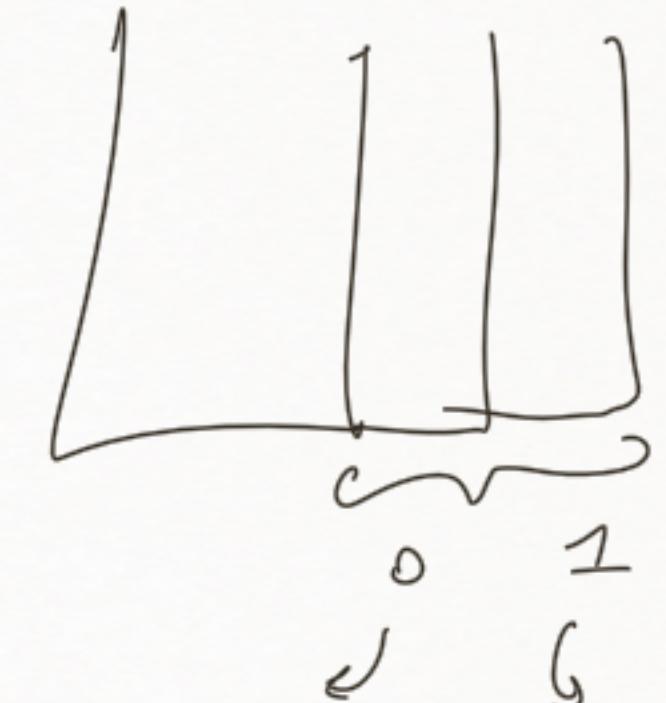
$$5+2=7 \text{ (dolu)} \quad 7+2=9 \text{ (dolu)}$$

$$8+2=11=0 \text{ (dolu)} \quad 0+2=2 \text{ (dolu)}$$

$2+2=4$ (boş) * Toplam 5 kere H_2 uyguladık.

* $49 \equiv 1 \pmod{2}$ olduğundan sağa yazılmır.

- 29, 18, 29, 28, 39, 13, 16, 38 ve 53
- Hash(key) = key mod 11
- i(key) = Quotient (key / 11) mod 11
- g(key) = key mod 2 ← yani 0'dır

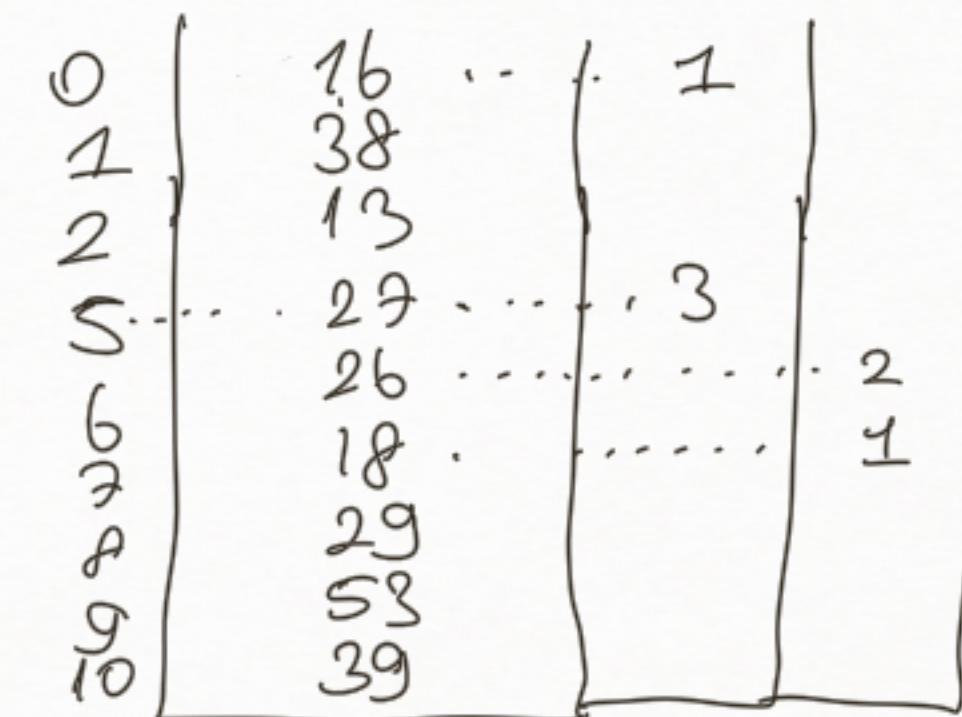


Gift
Sayıbr
Tek
Sayıbr.

• 16 ye H_2 uyguladık
 $0+16/11=4$ bu da
 38'dir.

$38 \equiv 0 \pmod{2}$
 Olduğundan 16'inin 1'i
 solda yazılır.

Son hâli



0 1
 (Çift) (Tek)

• 27 ye H_2 uyguladık.

$5+(27/11)\times 3=11$
 $=0$ 'da 16 vardır.

$16 \equiv 0 \pmod{2}$
 solda yazılır.

• 26 ye H_2 uyguladık
 $6+(26/11)\times 2=10$

10'da 39 vardır.

$39 \equiv 1 \pmod{2}$
 sağa yazılır.

arka
sayfaeki
ünlüm

~~Yarışma~~

Move on to:

Computed Chains

Binary Tree

Ex Storage: LSCH
Computed Chains

Brent's Method

Genel Özeti

Coalesced Hashing:

DCWC:

Progressive Overflow:

Linear Quotient:

Brent's Method:

Computed Chaining:

Small amount of space,
excellent performance

Requires some spaces for
pseudo link field, perf. below
that of DCWC

Usage

① Bol

ample space is available

ample space is available and
insertion time is not crit.

NEVER

Space is premium, retrieval performance
is not crit.

file is limited, temp space is
available for tree, performance
is important.

Some extra file space is
available and performance is important.
insertion time is not critical.

Perfect hashing

$$\rightarrow \text{p.hash.}(\text{key}) \rightarrow (h_0(\text{key}) + g[h_1(\text{key})] + g[h_2(\text{key})]) \bmod N$$

\rightarrow Potansiyel Adres ile anahtar soyisi aynı ise minimal perfect hashing function dens.
 $\left[\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \right] \rightarrow \left[\begin{array}{c} 0,1,2,3 \text{ da dolu.} \\ 4 \end{array} \right]$

Değerlendirme

\rightarrow Toplam backtracking sayısı her zamanı süresini çok etkilemektedir.

\rightarrow Gerçek zamanlı uygulamalarda minimal perfect hashing ian. Eşecen süre öneşenmiş.

\rightarrow Algoritması kolaydır.

Değerlendirme (b)

\rightarrow Aynı 2 değer olmaz.

\rightarrow 45 elemandan fazla olan listeleri alt gruplara böleriz.

\rightarrow Her liste için minimal perfect hashing function olmöglichilir.

Cichelli's Algorithm

$$① h_0 = v_{2n} lük$$

$h_1 = \text{ilk karakter}$

$h_2 = \text{son karakter}$

$g = T(x) \quad \} \text{ tablodan gelir.}$

$$\text{Ex/ } e = 0 \quad a = 3 \quad (mod 4)$$

$\{ \text{key}$

$$\text{p.hash}(e|ma) = h_0(e|ma) + T(e) + T'a$$

$$= 4 + 0 + 3 = 7$$

$$= 7 \equiv 3 \pmod{4}$$

$$\boxed{\text{p.hash} = 3}$$

Overview Storage & Indexing (Chapter 8)

6

auxiliary = yardımcı

cluster = kümeli

heap = yığın

entries = girdi

implied = lütfen etmek / kast etmek

typically = sıklıkla

orthogonal = dikdört / birbirine dik

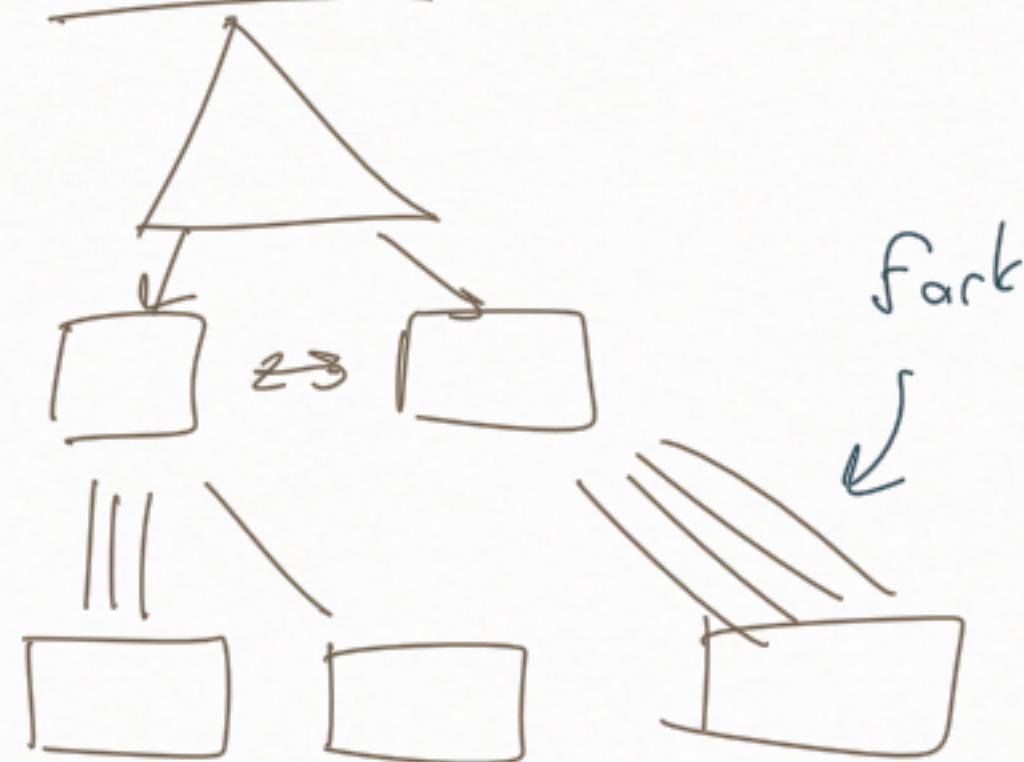
assumption = varsayılmak

estimate = tahmin etmek

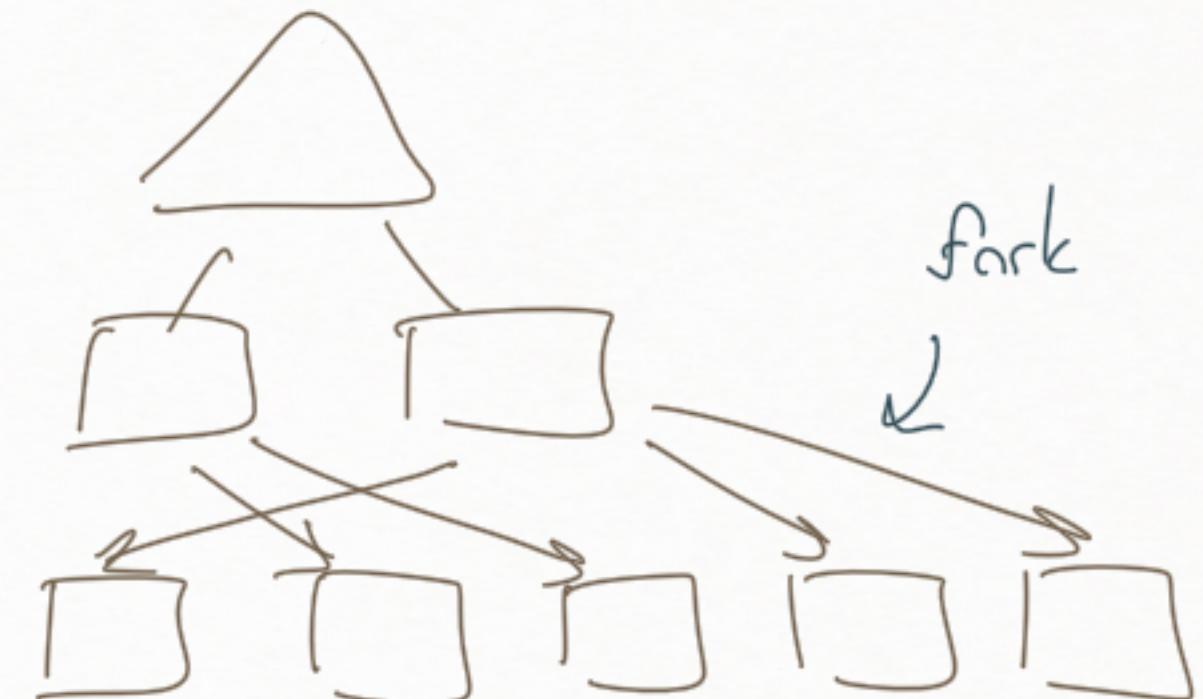
queries = sorular

involving = kapsama / ilgilendiren

clustered



Unclustered



Ex Clustered

→ Select E.dno from Emp E where E.age > 40
GROUP BY E.dno

→ SELECT E.dno from Emp E WHERE

E.hobby = Stamps

(Clustering on E.hobby)

* (age < 30)

Summary

→ Hash-based indexing saadece eşitlik araması için iyidir.

→ Sorted files ve Tree-based indexing * mesafe soruları için çok iyidir.

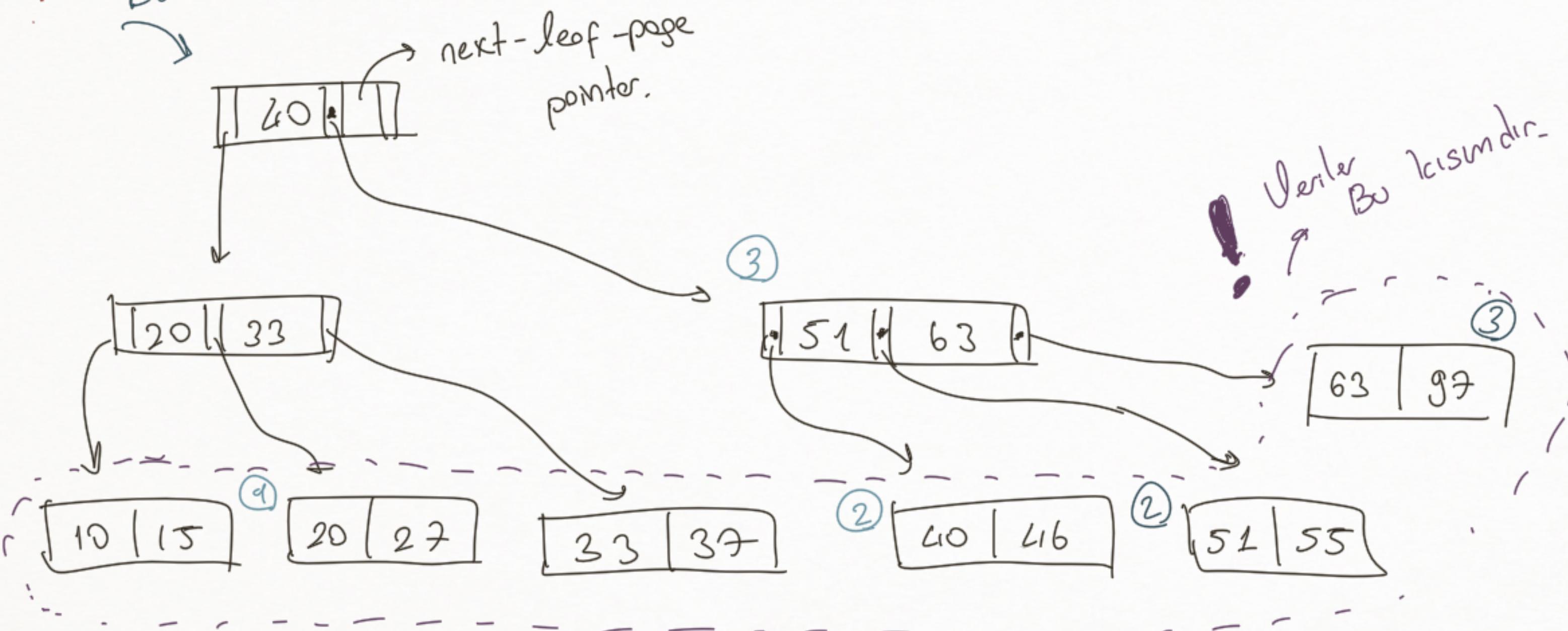
(Dosyalar nadiren uygunlukta sıralı olur sokoniyorsa
B+ Tree daha iyidir.)

C7

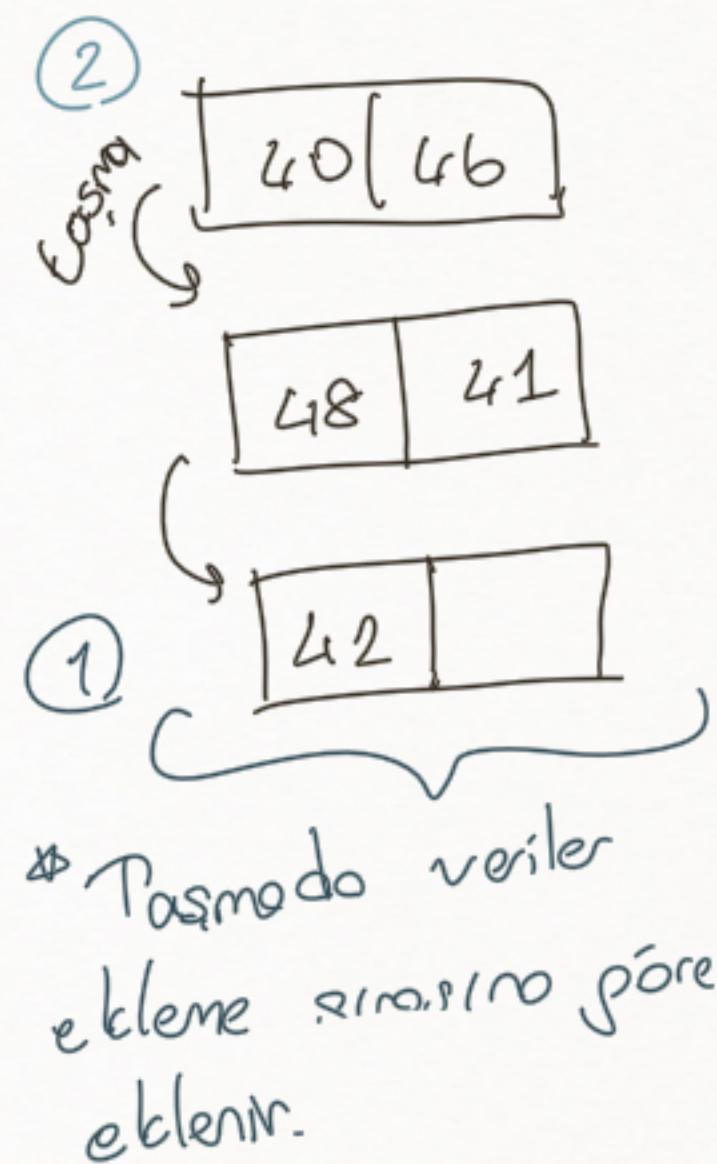
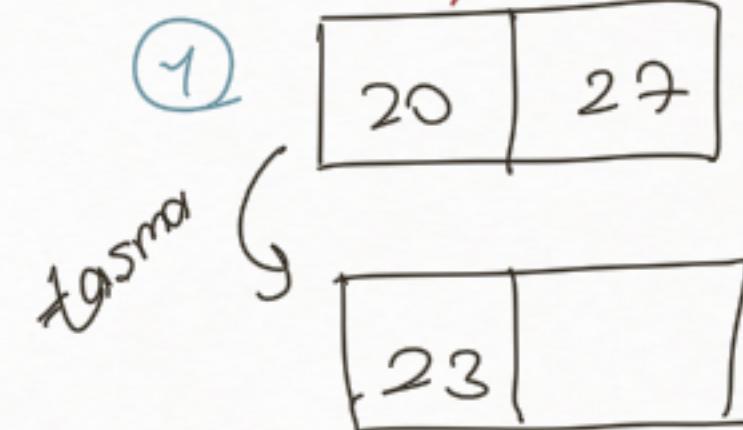
SAM

Kök

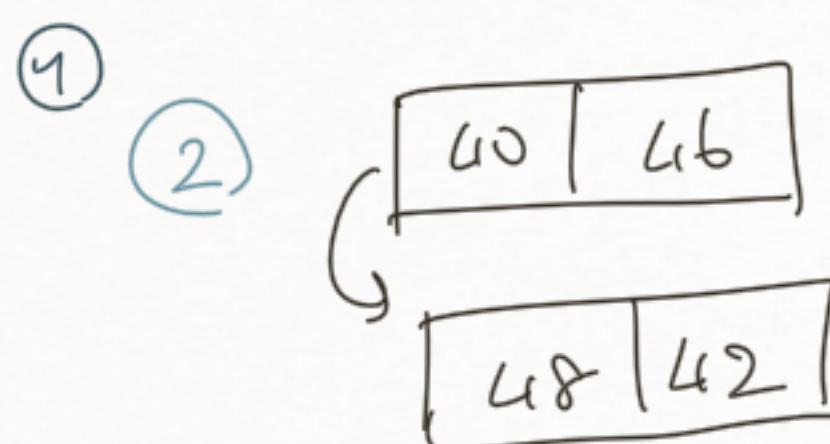
→ Her dğum 2 vert altı. Kökde tek var ols.



① 23, 48, 41, 42 silersek;

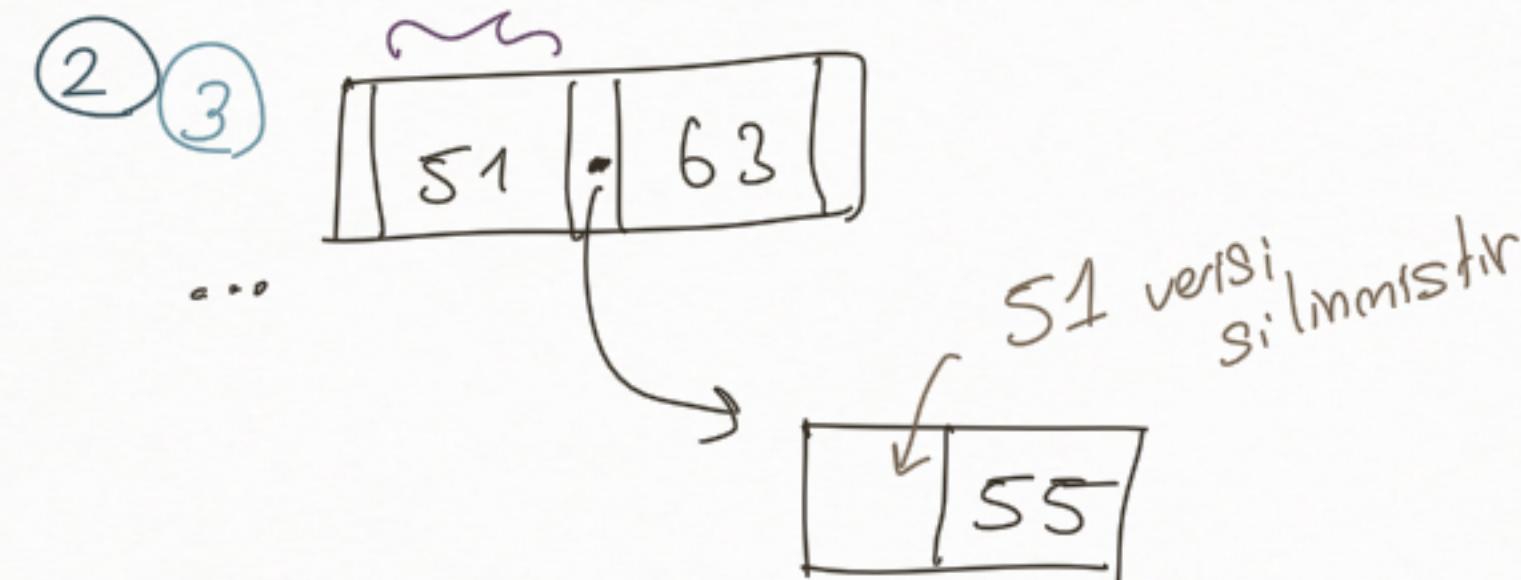


② ① ② ③ 42, 51, 97 silersek;

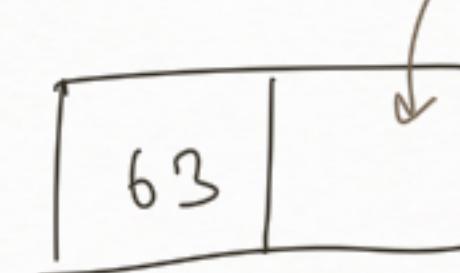


Silindi

! Veri olmadığı için silinmez.



93 versi silinmiştir.



B^x tree

→ LSAM statikts - Bu dinamiktir.

① → Her düğüm (kök hariç) en az ve en çok 2'ye anahtar içerişir.

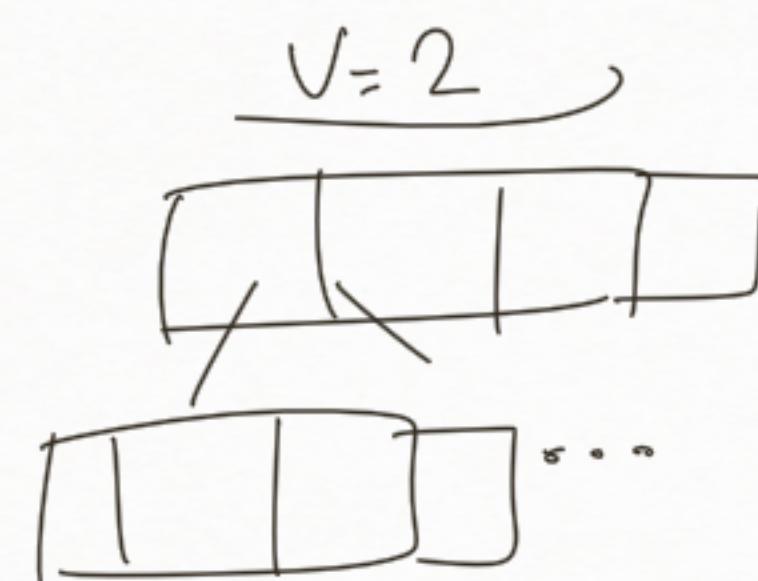
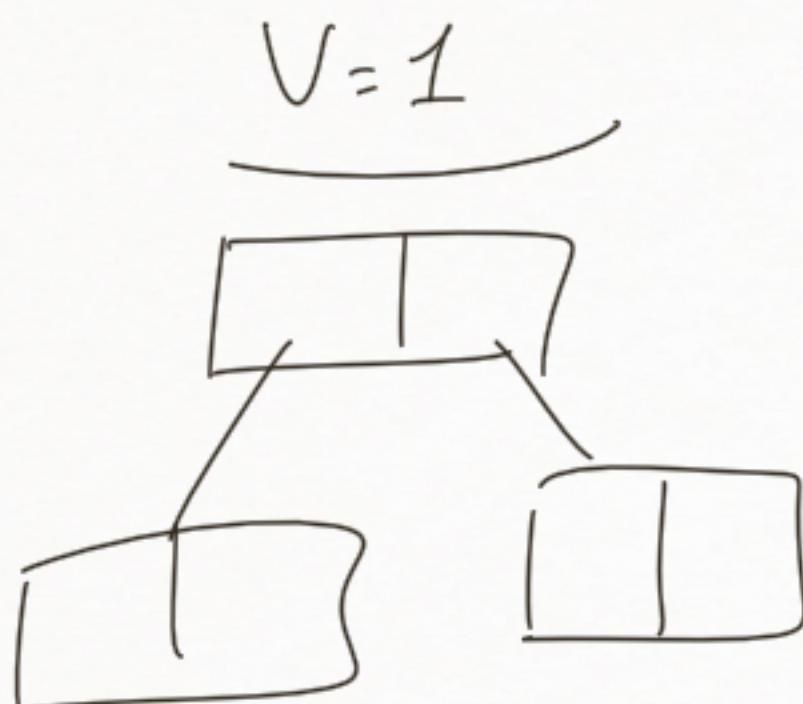
② → Kök en az 2 çocuğu sahip olmalıdır, yaprak olumsuz.

③ → Bütün yapraklar aynı seviyededir.

④ → K anahtarlı is düğüm, $K+1$ çocuk içerişir.

! En önemli kriter Yapılmalıdır.

① V: Ağacı düzeni



deomi fotobide

