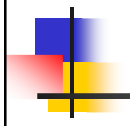


BM 307 Dosya Organizasyonu (File Organization)

Hazırlayan: M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Konular

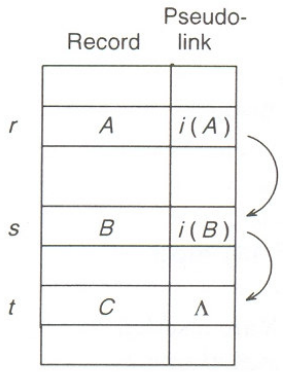
- Direct File Organization
  - Computed Chaining
  - Comparison of Collision Resolution Methods
  - Perfect Hashing
  - Cichelli's Algorithm

Direct File Organization
Computed Chaining

- Link kullanarak çakışma çözümü yapan metodlar (colaesced hashing) ve link kullanmadan çözüm yapan metodlar (progressive overflow, linear quotient, Brent's method, binary tree)
- Link kullanan metodlar ek yer gerektirmekte ancak daha iyi performansa sahiptir
- Link kullanmayan metodlar ek yer gerektirmemekte ancak daha kötü performansa sahiptir
- Computed chaining matodunda link adresleri kayıt edilmezler hesaplanarak bulunurlar
- Probe chain içerisindeki elemanların tam adresi yerine ofset adresleri saklanır
- Birkaç bit kullanılarak adres bilgisine ulaşılabilir

Direct File Organization
Computed Chaining

- Coalesced chain oluşmaması için eklenen kaydın home adresine yazılmış kayıtlar yeni bir adrese taşınır
- Böylece bir probe chain içindeki tüm kayıtlar aynı home adresine sahitir
- Bir kayıt için probe sayısı o kaydın probe chain içindeki pozisyonuna eşittir
- Computed chain bir sonraki adresi hesaplarken eklenen veya okunacak kaydı değil o adresteki kaydı kullanır



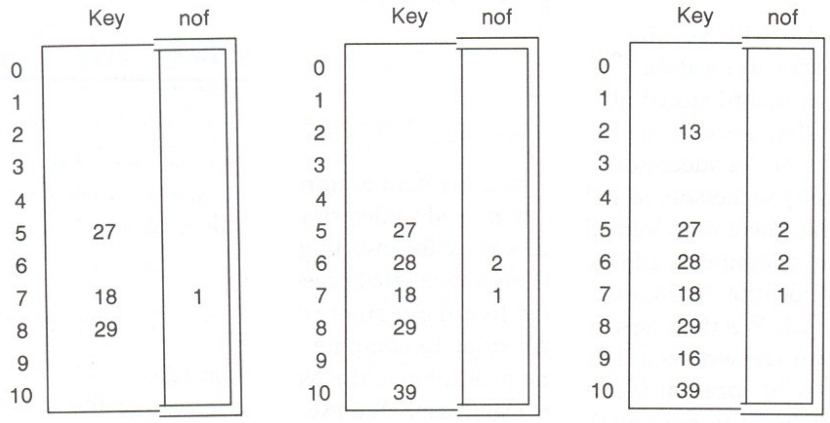
Direct File Organization

Computed Chaining

Örnek

- 27, 18, 29, 28, 39, 13, 16, ve 38, 53
- Hash(key) = key **mod** 11
- i(key) = Quotient (Key / 11) mod 11

nof (number of offsets, pseudolink value)



Direct File Organization

Computed Chaining

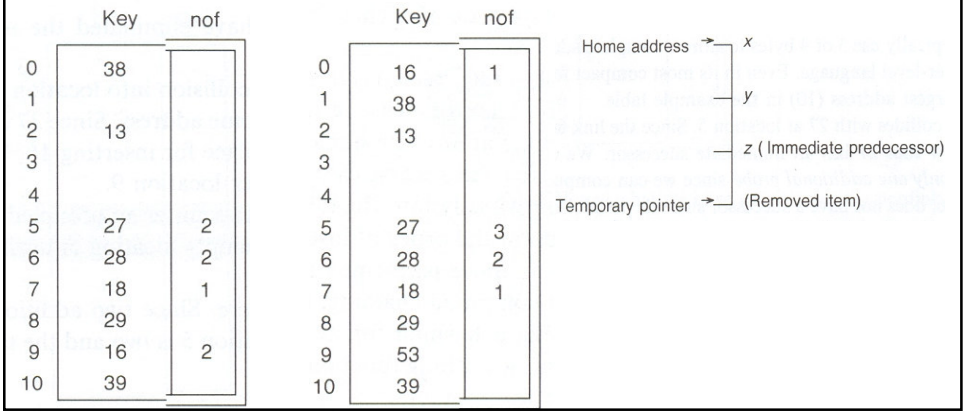
Değerlendirme

- İlk 7 kayıt için ortalama probe sayısı 1.4 tür.
- Bu sayı binary tree için 1.7 ve LISCH için 1.4 olmuştur.
- LISCH ile aynı performansa sahiptir ancak pseudolink için gereken alan LISCH deki link için kullanılan alandan daha azdır.
- Pseudolink için 2 bit gerekmektedir. Pointer'la tam adres gösterimi için 3 veya 4 byte gerekmektedir.
- En büyük adres 10 olsa bile en fazla 4 bit kulanılması yeterlidir.

Direct File Organization
Computed Chaining

Örnek (devam)

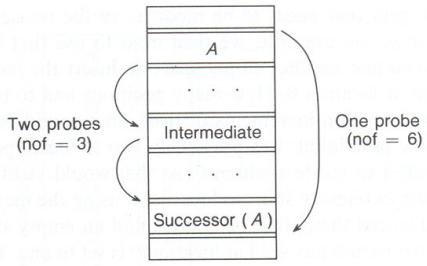
- 38 ve 53 eklendiğinde son durum aşağıdaki gibi olur.
- Hash(key) = key **mod** 11
- i(key) = Quotient (Key / 11) mod 11



Direct File Organization
Computed Chaining

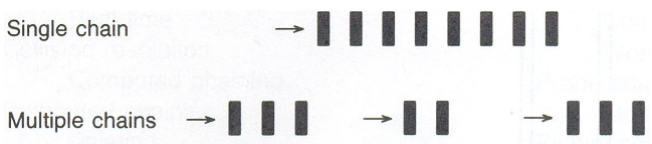
Değerlendirme

- Link kullanmayan metodlara göre daha iyi performansa sahiptir
- Bir kaydın retrieval probe sayısı probe chain'deki sırasına eşittir
- Worst case retrieval performansı herhangi bir home adresteki çakışma sayısıyla belirlenir
- Worst case retrieval performansı daha düzgün dağılımlı hash fonksiyonu seçimiyle iyileştirilebilir
- Silme işlemi silinen elemandan sonraki tüm kayıtların yeniden yerleştirilmesini gerektirir
- Computed chain metodu diğer metodlara göre silme ve ekleme için daha fazla süre gerektirir ancak retrieval için daha az süre gerektirir



Direct File Organization
Computed Chaining – Multiple Chains

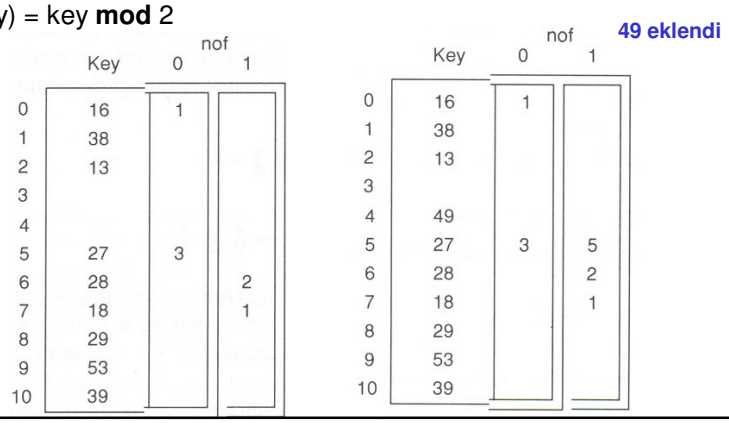
- Divide and Conquer stratejisini kullanır
- Bir probe chain birden fazla parçaya bölünür ve bir kayıt için sadece ilgili olduğu gruba bakılır
- Bir kayıt alanına birden fazla pseudolink oluşturulur
- Standart hash fonksiyonu ve artırma fonksiyonunun yanısıra birde probe chain seçimi için kullanılan g(key) fonksiyonu vardır
- g(key) = 0, 1, . . . , R-1 olabilir. R alt grup sayısıdır
- g(key) = key **mod** R



Direct File Organization
Computed Chaining – Multiple Chains

Örnek

- 27, 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



Direct File Organization

Comparison of Collision Resolution Methods

Karşılaştırma

TABLE 3.3 COMPARISON OF MEAN NUMBER OF PROBES FOR SUCCESSFUL LOOKUP (n = 997; α = packing factor)

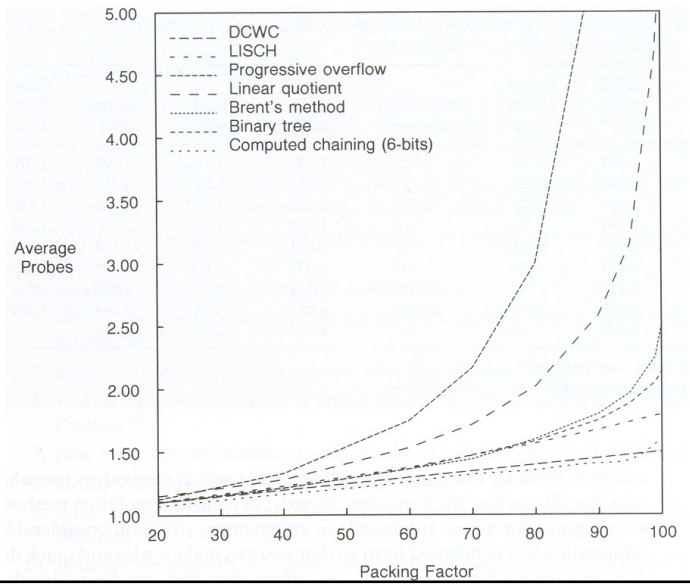
α (percent)	DCWC (10-bit link)*	LISCH (10-bit link)	Progressive overflow†	Linear quotient	Brent's method	Binary tree	Computed chaining (6-bit link)	Computed chaining (2-bit link)
20	1.100	1.106	1.125	1.15	1.102	1.102	1.070	1.070
40	1.200	1.232	1.333	1.277	1.217	1.217	1.168	1.214
60	1.300	1.379	1.750	1.527	1.367	1.364	1.264	1.381
70	1.350	—	2.167	1.720	1.444	—	1.323	1.528
80	1.400	1.566	3.000	2.011	1.599	1.579	1.356	1.715
90	1.450	1.674	5.500	2.558	1.802	1.751	1.408	2.062
95	1.475	1.734	10.500	3.153	1.972	1.880	1.433	2.414
99	1.495	1.783	50.500	4.651	2.242	2.049	1.601	3.330
100	1.500	—	—	6.522	2.494	2.134	—	—

*Theoretical results; mean probes = $1 + \alpha/2$.
†Theoretical results; mean probes = $(1 - \alpha/2) / (1 - \alpha)$

Direct File Organization

Comparison of Collision Resolution Methods

Karşılaştırma



Direct File Organization

Comparison of Collision Resolution Methods

Karşılaştırma

TABLE 3.4 SEARCH, RELOCATION, AND STORAGE COMPARISONS

Criteria	LISCH	Progressive overflow	Linear quotient	Brent's method	Binary tree	Computed chaining
Successful search						
Best case	1	1	1	1	1	1
Worst case	n	n	n	n	n	n
Move an item	No	No	No	Yes	Yes	Yes
Extra storage	Yes	No	No	No	No	Yes

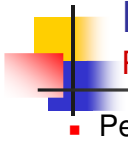
Direct File Organization

Comparison of Collision Resolution Methods

Karşılaştırma

TABLE 3.5 ADVANTAGES, DISADVANTAGES, AND WHEN TO USE VARIOUS COLLISION RESOLUTION METHODS

Method	Advantages	Disadvantages	When to use
Coalesced hashing	Excellent performance	Requires space for link field	When ample space is available
[DCWC]	Superior performance	Requires space for link field	When ample space is available and insertion time is not critical
Progressive overflow	No additional space; simple algorithm	Very poor performance	Almost NEVER
Linear quotient	No additional space; reasonable performance	Performance below that of other methods	When space is at a premium; retrieval performance is not critical
Brent's method	No additional space; good performance	Performance below that of linked methods	When space is at a premium, even space for a binary tree
Binary tree	No additional space; very good performance	Performance below that of linked methods	When file space is limited but temporary space is available for tree; performance is important
Computed chaining	Small amount of additional space; excellent performance	Requires some space for pseudolink field; performance below that of DCWC	When some extra file space is available and performance is important; insertion time is not critical



Direct File Organization
Perfect Hashing

- Perfect hashing bir key değerini unique bir adresle eşleştirir.

  hash (key) -> unique address.

- Az sayıdaki kayda sahip dosyalarla kullanılır. (Örn.: Programlama dillerindeki reserved words, en çok kullanılan kelimeler v.b.).
- Eğer potansiyel adres sayısı ile anahtar sayısı aynı ise bu fonksiyona minimum perfect hashing fonkisyonu denir.

  p.hash(key) -> (h_0(key) + g[h_1(key)] + g[h_2(key)]) **mod** N

- Worst case computational comlexity değeri O(r^6), r toplam kayıt sayısıdır

Direct File Organization
Perfect Hashing – Cichelli's Algorithm

- Aşağıdaki fonksiyonlar kullanılır

  h_0 -> length(key)
  h_1 -> first_karakter(key)
  h_2 -> last_karakter(key)
  g -> T(x)

- T herbir x karakteri için hazırlanan tablodaki değeri gösterir
- Ençok zaman alan kısım T değerinin hesaplanmasıdır

Direct File Organization
Perfect Hashing – Cichelli's Algorithm

TABLE 3.6 VALUES ASSOCIATED WITH THE CHARACTERS OF THE PASCAL RESERVED WORDS

A=11	B=15	C=1	D=0	E=0	F=15
G=3	H=15	I=13	J=0	K=0	L=15
M=15	N=13	O=0	P=15	Q=0	R=14
S=6	T=6	U=14	V=10	W=6	X=0
Y=13	Z=0				

$$\begin{aligned} \text{p.hash(}\textbf{begin}\text{)} &= 5 + T(h_1(\text{key})) + T(h_2(\text{key})) \\ &= 5 + T(\text{b}) + T(\text{n}) \\ &= 5 + 15 + 13 \\ &= 33 \end{aligned}$$

Direct File Organization
Perfect Hashing – Cichelli's Algorithm

- Tablonun oluştutrulması için önce bütün anahtarlar ilk ve son karakterlerinin kullanım sıklığı belirlenir
- Herbir anahtara ilk ve son karakterlerinin kullanım sıklıklarının toplam değeri atanır
- Anahtarlar atanan değerlere göre büyükten küçüğe sıralanır
- Backtracking metodu kullanılarak herbir başlangıç ve bitiş karakterine değer atanır

Direct File Organization

Perfect Hashing – Cichelli's Algorithm

Örnek

- cat, ant, dog, gnat, chimp, rat, toad
- Bir karaktere en fazla 4 değeri atansın

İlk ve son karakterlerin sıklığı:

a = 1 c = 2 d = 2 g = 2 p = 1 r = 1 t = 5

cat	7	toad	7	toad	7
ant	6	gnat	7	gnat	7
dog	4	cat	7	dog	4
gnat	7	rat	6	cat	7
chimp	3	ant	6	rat	6
rat	6	dog	4	ant	6
toad	7	chimp	3	chimp	3

İlk ve son karakterleri daha önce geçenler en son geçenden sonraya alınır

Direct File Organization

Perfect Hashing – Cichelli's Algorithm

Örnek (devam)

Bütün anahtarlar sırayla değerlendirilir. Aynı olanlar için back tracking yapılarak ilk ve son karakterlerine yeniden değer atanır.

$t = 0 \quad d = 0$

p.hash(toad) = 4

$t = 0 \quad d = 0 \quad g = 1$

p.hash(toad) = 4
p.hash(gnat) = 5

$t = 0 \quad d = 0 \quad g = 2$

p.hash(toad) = 4
p.hash(gnat) = 6

$t = 0 \quad d = 0 \quad g = 2$

p.hash(toad) = 4
p.hash(gnat) = 6
p.hash(dog) = 5

$t = 0 \quad d = 0 \quad g = 2 \quad c = 0$

p.hash(toad) = 4
p.hash(gnat) = 6
p.hash(dog) = 5
p.hash(cat) = 3

$t = 0 \quad d = 0 \quad g = 2 \quad c = 0 \quad r = 4$

p.hash(toad) = 4
p.hash(gnat) = 6
p.hash(dog) = 5
p.hash(cat) = 3
p.hash(rat) = 7

$t = 0 \quad d = 0 \quad g = 3 \quad c = 0 \quad r = 2$

p.hash(toad) = 4
p.hash(gnat) = 7
p.hash(dog) = 6
p.hash(cat) = 3
p.hash(rat) = 5

Direct File Organization

Perfect Hashing – Cichelli's Algorithm

Örnek (devam)

$$t = 0 \quad d = 0 \quad g = 4 \quad c = 0 \quad r = 2 \quad a = 3$$

p.hash(toad) = 4
p.hash(gnat) = 8
p.hash(dog) = 7
p.hash(cat) = 3
p.hash(rat) = 5
p.hash(ant) = 6

Son çözüm aşağıdaki gibidir

$$t = 0 \quad d = 0 \quad g = 4 \quad c = 0 \quad r = 2 \quad a = 3 \quad p = 4$$

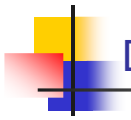
p.hash(toad) = 4
p.hash(gnat) = 8
p.hash(dog) = 7
p.hash(cat) = 3
p.hash(rat) = 5
p.hash(ant) = 6
p.hash(chimp) = 9

Direct File Organization

Perfect Hashing – Cichelli's Algorithm

Değerlendirme

- Toplam backtracking sayısı hesaplama süresini çok etkilemektedir
- Özellikle gerçek zamanlı uygulamalarda minimal perfect function oluşturmak için geçen süre önemsenmez
- Dezavantajları:
  - Aynı uzunlukta ve aynı ilk ve son karaktere sahip iki anahtar olamaz
  - 45 elemandan fazla olan listeleri alt gruplara bölmek gerekebilir
  - Her liste için minimal perfect hashing function bulunamayabilir
- Algoritmanın programlanması kolaydır



Direct File Organization

Haftalık Ödev

C#.NET programlama dilindeki reserved word'ler için Cichelli's Algorithm kullanarak perfect hash fonksiyonu oluşturunuz. Program görsel arayüze sahip olacak ve girilen reserved word için üretilen adres değerini ekrana yazacak. Ayrıca ayrılmış kelimelerin oluşturacağınız dosya içindeki sıralanışları ekranda görüntülecenek.