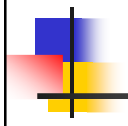


BM 307 Dosya Organizasyonu (File Organization)

Hazırlayan: M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Konular

- Bits of Information
  - Binary Özellikler
  - Superimposed Coding
  - Signature Formation
  - Değerlendirme

Bits of Information
Binary Özellikler

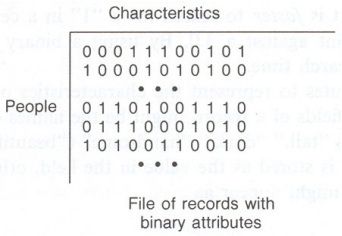
- Bir binary özellik iki değer alabilir (kapalı veya açık; var veya yok gibi)
- Bir kişiye ait bilgiler binary olarak aşağıdaki gibi gösterilebilir

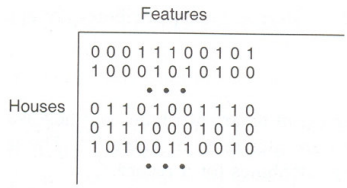
tall	dark	handsome	smart	witty	clever	strong	graceful	persuasive	confident	wise
0	0	0	0	0	0	0	0	0	0	0

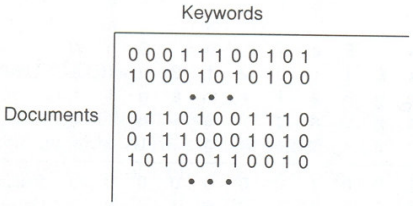
tall	dark	handsome	smart	witty	clever	strong	graceful	persuasive	confident	wise
1	1	1	1	1	1	1	1	1	1	1

Bits of Information
Binary Özellikler

- İstenen bir özelliğe bağlı sorgulama ilgili sütun üzerinde yapılır







Bits of Information
Binary Özellikler

- Örnek: Bileşimlerin 16-bit dizileriyle gösterimi

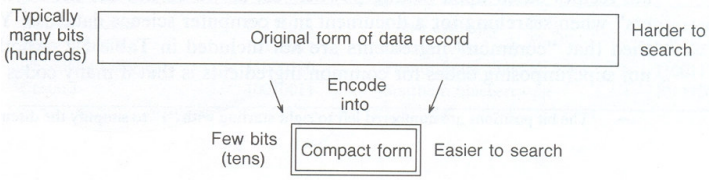
TABLE 5.1 DESSERT RECIPES REPRESENTED WITH BINARY ATTRIBUTES

	Ingredients																
Desserts	Baking powder	Blueberries	Brown sugar	Butterscotch	Chocolate	Cinnamon	Eggs	Flour	Cream	Lemon juice	Milk	Nutmeg	Salad	Sugar	Vanilla	Water	Special ingredients
Banana sundaes	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	Bananas
Berry crumble	0	1	1	1	0	1	0	1	1	1	0	0	1	0	0	0	Oats
Chocolate toffee bars	1	0	1	1	1	0	1	1	0	0	0	0	1	0	1	0	Nuts
Custard	0	0	0	0	0	0	1	0	0	0	1	1	1	1	1	0	
Fresh apple pie	0	0	0	1	0	1	0	1	0	0	0	1	1	1	0	0	Apples
Glazed pound cake	1	0	1	0	0	0	1	1	0	0	1	0	1	1	1	0	Graham crackers
Peanut-fudge pudding cake	1	0	1	0	1	0	0	1	1	0	1	0	1	1	1	1	Peanut butter, vegetable oil
Southern blueberry pie	0	1	0	0	0	1	0	0	0	1	0	0	1	1	0	1	Cornmeal, cornstarch

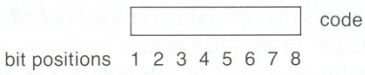
Source: Betty Crocker's Microwave Cookbook, Random House, New York, Copyright © 1981. Adapted with permission.

Bits of Information
Superimposed Coding

- Bilgi daha kolay ve hızlı işlenebilmesi için kodlanır
- Bilginin kodlanması için kullanılan yapıya **superimposed coding** denir



- Önceki örnekteki bileşimleri 8 bit kullanarak ifade edebiliriz



$$ {}_mC_k = \frac{m!}{(m-k)!\,k!} $$

- m toplam bit sayısını (8), ve k ise kullanılacak 1 sayısını ifade eder
- k = 1 için toplam farklı kod sayısı 8, k = 2 için 28 olur

Bits of Information
Superimposed Coding

- Bileşimlere atanan kodlar
- Genellikle tüm kodlarda kullanılan özellikler superimposed kod içerisinde ifade edilmezler. Böylece retrieval süresi azaltılmış olur

TABLE 5.2 INGREDIENT CODES

Apples	00100100	Ice cream	01000001
Bananas	00100010	Lemon juice	00100001
Blueberries	10000001	Nutmeg	00010001
Brown sugar	01000010	Nuts	01000100
Chocolate	10000100	Oats	00010010
Cinnamon	00011000	Peanut butter	01001000
Cornmeal	00101000	Vanilla	10000010
Graham crackers	00010100		

- Sonuç değerler bileşiklerin kodlarının OR işlemiyle bulunur.

Brown sugar	01000010
Chocolate	10000100
Ice cream	01000001
Vanilla	10000010
Peanut butter	01001000
Peanut-fudge pudding cake	11001111

Bits of Information
Superimposed Coding

- Bir bileşimin bileşenlerini bulmak için bileşen kodlarında 1 olan pozisyonlara bakılır (AND işlemi yapılır)
- Bir bileşenin bulunduğu bileşim kodlarını bulmak içinde bileşimde 1 olan pozisyonlara bakılır (AND işlemi yapılır)
- Örnek olarak Chocolate bulunduran bileşimlere bakılırsa, Chocolate Toffee Bars, Glazed Pound Cake ve Peanut-Fudge Pudding Cake olduğu görülür
- Orijinal kodlar ile Chocolate bulunduran bileşimlere bakıldığında Glazed Pound Cake olmadığı görülür
- Superimposed coding sonucunda kullanılan bit sayısına ve kodlamaya bağlı olarak bilgi kaybı (false drop) olabilir. Ancak retriveal süresi azaltılır
- Tüm bileşimler için atanan kodlar

TABLE 5.3 SUPERIMPOSED RECIPE CODES

Banana sundaes	01111011	Fresh apple pie	00111101
Berry crumble	11111011	Glazed pound cake	11010110
Chocolate toffee bars	11000110	Peanut-fudge pudding cake	11001111
Custard	10010011	Southern blueberry pie	10111001

Bits of Information
Superimposed Coding

Değerlendirme

- Superimposed coding yüzlerce bitlik bilgiyi daha kısa şekilde ifade eder
- Bilginin elde edilmesi için gereken süre kısaltılır
- Superimposed kodlar fazladan yer kaplar. False drop olanları bulmak için orijinal kodların saklanması gerekmektedir.
- False drop sayısı kullanılan bit sayısı artırılarak azaltılabilir.
- k değeri artırılarak false drop sayısı azaltılamaz aksine artar çünkü kod içerisindeki 1 sayısı artar

Bits of Information
Text Searching

- Naive text searching algoritması aranan string (pattern) ile aranılacak string (string) arasında baştan sona kadar bir bir karşılaştırma yapar

```
string   Problem solving is a common paradigm of computer science

pattern  computer
```

- Worst case computational complexity O(mn) olur. m aranan ve n ise aranılacak string uzunluğudur
- Örnekte toplam 50 karşılaştırma yapılmıştır

```
Problem solving is a common paradigm of computer science.     Number of comparisons

computer                                                            1
 computer                                                           2
  computer                                                          3
   . . .
          computer                                                  22-24
           computer                                                 25
            computer                                                26
             . . .
                                  computer                          43-50

[The _ represents the pattern symbol being compared.]
```

Bits of Information
Text Searching

- Boyer-Moore daha gelişmiş bir arama algoritması geliştirmişlerdir
- Arama işlemine baştan değil sondan başlanır
- Pattern sonundaki karakterle string içindeki karakter aynı değilse, string içindeki karakterin pattern içindeki (varsa) ensağ pozisyonuna kadar pattern kaydırılır
- Pattern içindeki herhangi bir karakterle string içindeki karakterin karşılaştırmasında aynı karakter olmazsa kaydırma işlemi stringdeki karşılaştırılan karakter için yapılır
- Önceki örnekteki arama toplam 14 karşılaştırma yapılarak bulunur

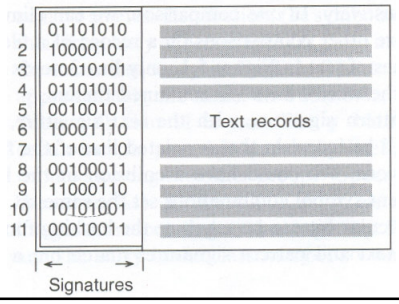
```
                                                        Number of
Problem solving is a common paradigm of computer science.     comparisons

computer                                                          1
   computer                                                       2
      computer                                                    3
         computer                                                 4
            computer                                              5
               computer                                           6
                  computer                                    7-14

   [The _ represents the pattern symbol being compared.]
```

Bits of Information
Signature Formation

- Text search işleminde tüm string üzerinde arama yapmak yerine, olma ihtimali olan kısım üzerinde arama yapmak, arama hızını artırır
- Text içindeki segmentler (satırlar veya paragraflar) için birer imza (signature) oluşturulur
- Önce signature bilgisine göre aranan string'in olup olmadığı belirlenir daha sonra diğer algoritmalarla (Örn: Boyer-Moore) string üzerinde arama yapılır
- Record signature veya text signature bir blok metnin içeriğinin kodlanmasıdır



Bits of Information
Signature Formation

- k-adet gruplanmış yanyana sembolün hash fonksiyonuyla m boyutundaki signature içerisinde ilgili pozisyona aktarılmasıdır

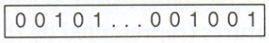
string Problem solving is a common paradigm of computer science

if $k = 2$ then all 2-symbol pairs are hashed, that is,

h(Pr), h(ro), h(ob), h(bl), h(le), h(em), h(m▒), . . . , h(e.)

where h is the hashing function and ▒ represents a blank symbol.

- k değerinden daha küçük stringler aranamayacağı için k genellikle 2 olarak alınır
- Örnek bir kodlama



- Örneğin **hash** kelimesinin signature değeri **10010100** olursa önceki tablodan sadece 3.satıra bakılmalıdır
- İlgili signature değerinin uygun olması aranan string'in olduğunu göstermez
- İlgili signature değerinin **uygun olmaması** aranan string'in **kesinlikle olmadığını** gösterir

Bits of Information
Signature Formation

- **Örnek**
  - k = 2, m = 64 bit ve 128 bit olarak alınsın.
  - Kayıt boyutunun 80 byte olduğu varsayılsın.
  - Signature işleminden sonra kayıt boyutu 88 byte veya 96 byte olacaktır.
  - Sembollerin kullanılma sıklığına göre gruplandırılması iki şekilde alınmıştır. Birisinde 8 grup diğerinde 11 grup oluşturulmuştur.
  - Her grup içindeki sembollerin kullanılma sıklığı toplamı birbirine eşit veya yaklaşık olacaktır.
  - Bu örnekte İngiliz harfleri için Knuth tarafından önerilen değerler alınmıştır.
  - Diğer uygulamalarda seçilen metin içindeki kullanım sıklıkları alınabilir
  - Grup sayısı (n) değeri, k=2 için n^2 <= m olacak şekilde seçilir. m signature length değeridir. m=64 için n değeri 8 seçilebilir ve m=128 için n değeri 11 seçilebilir

Bits of Information
Signature Formation

- **Örnek (Devam)**
  - y_1 ve y_2 sembol çifti için hash fonksiyonu

$$h(y_1 y_2) = \text{number\_of\_classes} * T(y_1) + T(y_2)$$

8 grup

TABLE 5.4a DISTRIBUTION OF TEXT SYMBOLS FOR 8-BYTE SIGNATURES

Class	Symbols
0	▒ (blank)$_{[18.6]*}$
1	E$_{\{10.3\}\dagger}$ B$_{\{1.3\}[11.6]}$ 6 : & ' " ?
2	T$_{\{8.0\}}$ X$_{\{0.1\}}$ Z$_{\{0.1\}}$ W$_{\{1.8\}}$ G$_{\{1.5\}[11.5]}$ 5 ; / * <
3	A$_{\{6.4\}}$ F$_{\{2.1\}}$ Y$_{\{1.6\}}$ P$_{\{1.5\}[11.6]}$ 4 ,) ! > ^
4	O$_{\{6.3\}}$ L$_{\{3.2\}}$ C$_{\{2.2\}[11.7]}$ 3 . (@ [_
5	I$_{\{5.7\}}$ K$_{\{0.5\}}$ D$_{\{3.2\}}$ M$_{\{2.0\}}$ J$_{\{0.1\}}$ Q$_{\{0.1\}[11.6]}$ 2 9 #] |
6	N$_{\{5.7\}}$ V$_{\{0.8\}}$ S$_{\{5.1\}[11.6]}$ 1 8 – $ {
7	H$_{\{4.7\}}$ U$_{\{2.3\}}$ R$_{\{4.8\}[11.8]}$ 0 7 + % }

*Total percentages for letters in that group.
†Percentage of occurrence for that letter.

11 grup

TABLE 5.4b DISTRIBUTION OF TEXT SYMBOLS FOR 16-BYTE SIGNATURES

Class	Symbols
0	▒ (blank) $_{[18.6]}$
1	E$_{[10.3]}$ 1 ! + *
2	T$_{[8.0]}$ 2 ? < @
3	A$_{\{6.4\}}$ G$_{\{1.5\}[7.9]}$ 3 . > /
4	O$_{\{6.3\}}$ Y$_{\{1.6\}[7.9]}$ 4 & (
5	I$_{\{5.7\}}$ F$_{\{2.1\}}$ Q$_{\{0.1\}[7.9]}$ 5 ; =)
6	N$_{\{5.7\}}$ M$_{\{2.0\}}$ J$_{\{0.1\}[7.8]}$ 6 : – {
7	S$_{\{5.1\}}$ U$_{\{2.3\}}$ K$_{\{0.5\}[7.9]}$ 7 — # }
8	R$_{\{4.8\}}$ C$_{\{2.2\}}$ V$_{\{0.8\}}$ X$_{\{0.1\}[7.9]}$ 8 ' ^ [
9	H$_{\{4.7\}}$ W$_{\{1.8\}}$ B$_{\{1.3\}}$ Z$_{\{0.1\}[7.9]}$ 9 %]
10	D$_{\{3.2\}}$ L$_{\{3.2\}}$ P$_{\{1.5\}[7.9]}$ 0 " $ 1

Bits of Information
Signature Formation

- **Örnek (Devam)**
  - 64 bit signature için ve 128 bit signature için hash fonksiyonu

$$h(y_1 y_2) = 8 * T(y_1) + T(y_2)$$

$$h(y_1 y_2) = 11 * T(y_1) + T(y_2)$$

  - Hash fonksiyonundan alınan değerin bulunduğu pozisyona 1 değeri atanır
  - **computer** kelimesi için 8 byte ve 16 byte signature değerleri

Hash function	Bit position set: 8-byte signature	Bit position set: 16-byte signature
h(co)	36	92
h(om)	37	50
h(mp)	43	76
h(pu)	31	117
h(ut)	58	79
h(te)	17	23
h(er)	15	19

h(co) = 8 * T(c) + T(o)
= 8* 4 + 4
= 36

h(co) = 11 * T(c) + T(o)
= 11* 8 + 4
= 92

Bits of Information

Değerlendirme

- Signature uzunluğuna göre text file ve program file üzerindeki aramaların performans değerleri aşağıdaki şekilde görülmektedir
- Text file, program file üzerinde aramadan daha fazla satır aramayı gerektirmektedir
- 16 bit signature kullanılması durumunda arama yapılan satır sayısı önemli ölçüde düşmektedir
- Bu aramada **computer** kelimesi için substring olduğu kelimelerin satırlarınada bakılır (Örn: minicomputer)
- Substring şeklinde arama yapılmaması için aranan string'in başına ve sonuna delimiter (Örn:boşluk) konulabilir

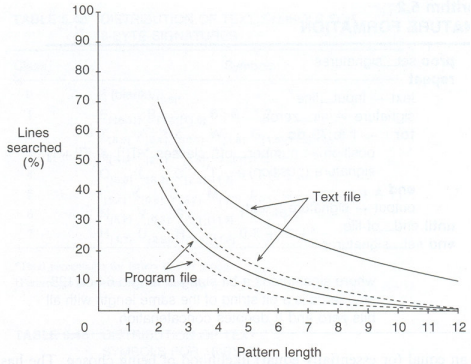


Figure 5.2 Lines searched vs. pattern length ——— for 8-byte signatures and – – – – for 16-byte signatures.

Bits of Information

Haftalık Ödev

Seçeceğiniz 320 kelimelik bir metin için her 80 karakter bilgiyi bir grup olarak alarak signature oluşturunuz. Signature değerleri k = 2, n = 8 ve m = 64 için bulunacaktır. Harflerin kullanılma sıklığını ve grup sayısını gösteren tabloyu, kullandığınız metin için oluşturunuz. Tabloda 8 gruba ait karakterleri belirleyiniz. C#.NET programlama diliyle bir arayüz hazırlayarak arama işlemini gerçekleştiriniz.