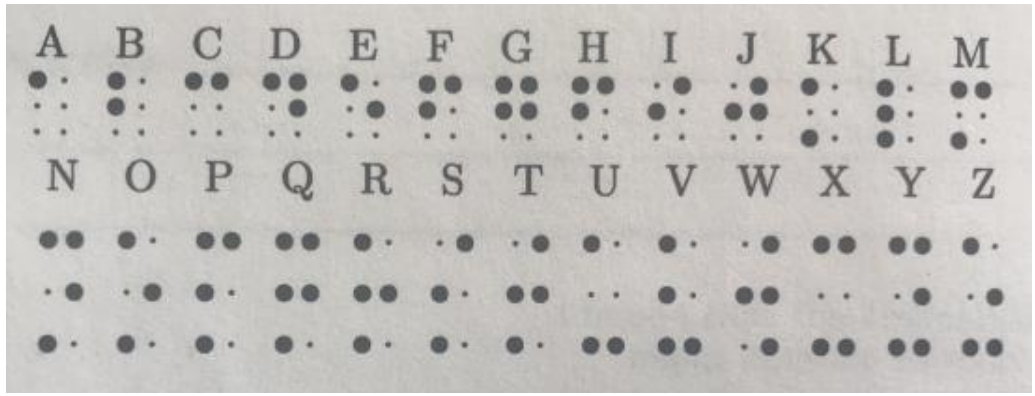
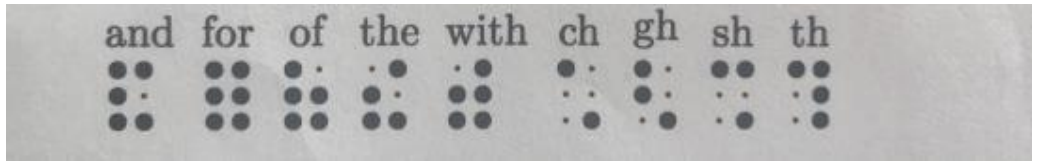
VERİ SIKIŞTIRMA

Temel Teknikler -3

Prof.Dr. Banu DİRİ

- Sezgisel Sıkıştırma (Intuitive Compression)
  - Braille Kodlama
  - Morse Kodlama





A	.-	N	-.	1	.----	Period	.-.-.-
B	-...	O	---	2	..---	Comma	--..--
C	-.-.	P	.--.	3	...--	Colon	---...
Ch	----	Q	--.-	4	....-	Question mark	..--..
D	-..	R	.-.	5	.....	Apostrophe	.----.
E	.	S	...	6	-....	Hyphen	-....-
F	..-.	T	-	7	--...	Dash	-..-.
G	--.	U	..-	8	---..	Parentheses	-.--.-
H	....	V	...-	9	----.	Quotation marks	.-..-.
I	..	W	.--	0	-----		
J	.---	X	-..-				
K	-.-	Y	-.--				
L	.-..	Z	--..				
M	--						

Braille kodlama
- Literary
- Nemeth
- CBC
- Music

Mors kodlama
- Her nokta 1 birim, her çizgi 3 birim
- Nokta ve çizgiler arası 1 birim
- Karakterler arası 3, kelimeler arası 6 birim
- Hata yapıldığında son kelimenin silinmesi için 8 nokta

- Geri Dönüşümü Olmayan Sıkıştırma (Irreversible Text Compression)

  Birden fazla boşluğun silinmesi, küçük/büyük harfe çevirme

- Ad Hoc Sıkıştırma

  Geri dönüşümü olan basit, kişiye özel çözümler

  Sparse String Sıkıştırma

Sparse String Sıkıştırma

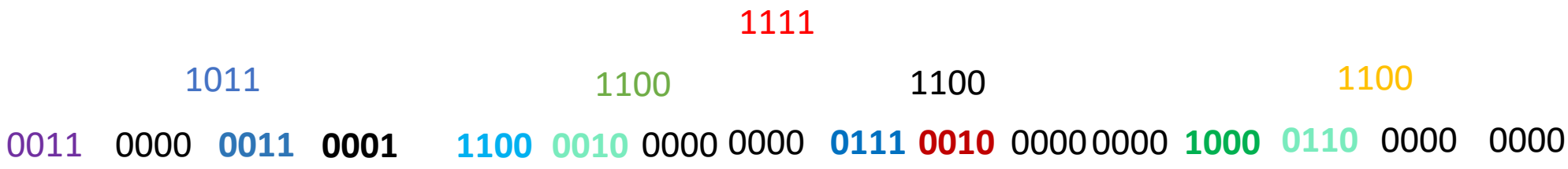
Bu__dersin__adı___veri_sıkıştırma___ve_pazartesi_günü__yapılıyor → boşluk yerine 1 kodlanıyor

L1=0011|0000|0011|0001|1100|0010|0000|0000|0111|0010|0000|0000|1000|0110|0000|0000
L2=1011|1100|1100|1100
L3=1111

L1 ve L2 deki sıfır olan gruplar silinerek L1, L2 ve L3 stringleri ve arkasından boşluksuz olarak text gönderilir.

L1=0011|**0011**|**0001**|**1100**|0010|**0111**|**0010**|**1000**|0110
L2=1011|1100|1100|1100
L3=1111

L1 L2 L3 budersinadıverisıkıştırmavepazartesigünüyapılıyor



- Baudot Kodlama

Letters	Code	Figures	Letters	Code	Figures
A	10000	1	Q	10111	/
B	00110	8	R	00111	-
C	10110	9	S	00101	SP
D	11110	0	T	10101	na
E	01000	2	U	10100	4
F	01110	na	V	11101	'
G	01010	7	W	01101	?
H	11010	+	X	01001	,
I	01100	na	Y	00100	3
J	10010	6	Z	11001	:
K	10011	(	LS	00001	LS
L	11011	=	FS	00010	FS
M	01011	)	CR	11000	CR
N	01111	na	LF	10001	LF
O	11100	5	ER	00011	ER
P	11111	%	na	00000	na

LS, Letter Shift; FS, Figure Shift.
CR, Carriage Return; LF, Line Feed.
ER, Error; na, Not Assigned; SP, Space.

- Önek Sıkıştırma (Front Compression)

a	a
aardvark	1ardvark
aback	1back
abaft	3ft
abandon	3ndon
abandoning	3ndoning
abasement	3sement
abandonment	3ndonment
abash	3sh
abated	3ted
abate	3te
abbot	2bot
abbey	3ey
abbreviating	3reviating
abbreviate	9e
abbreviation	9ion

Run Length Encoding Compression

- Eğer bir input stream içerisinde bir «d» verisi «n» defa arka arkaya geliyorsa bunu bir seferde «nd» çifti ile gösterebiliriz.
- «n» değerine «run» denir.
- Hem text hem de görüntü sıkıştırmada kullanılır.
- Kayıplı ve kayıpsız versiyonları vardır.
- Run Lenght Encoding veya RLE olarak adlandırılır.

Uncompressed

aaaaabbbbbbbbbbbbccccddddddddddeeeeeeeeeee

Compressed

5a12b4c9d10e

Uncompressed

aabccdeefghijjjklmnopqrrstttuvvwwxyyyz

Compressed *Negative compression*

2a1b2c1d2e1f1g1i3j1k1l1m1n1o1p1q2r1s3t1u2v2w1x3y1z

Uncompressed

2. all is too well

Compressed

2. a2l is t2o we2l

1- Sıkıştırma var mı?
2- Decompressor 2'leri anlayacak mı?

Nasıl Çözülür ?

Uncompressed

2. all is too well

Özel escape kod kullanalım

Compressed

2. a@2l is t@2o we@2l

1- Orijinal veriden daha uzundur
2- İki karakter yerine 3 karakter kullanılmıştır
3- Tekrar faktörünü 3 veya daha fazla karakter için kullanmak gerekir
4- Tekrar sayısı 3'den az ise karakter aynen yazılır

Temel problemler

- Doğal dilde yazılmış olan metinlerde arka arkaya tekrar eden karakter nadirdir (boşluklar hariç)
- Tekrar sayısına 1 byte ayrılmış olup, 255 ile sınırlıdır. Tekrar sayısı 3 tekrar karakterden sonra kullanılacak ise 258 adet aynı karakter 3 byte ile kodlanır
- Kullanılan özel escape kodu metin içerisinde geçebilir (farklı karakter seçilmelidir)
- Exe türü dosyalarda tüm ASCII karakterler dosya içerisinde yer alabilir. Bu durumda farklı bir yol izlenmelidir.†

†

MNP5 (Microcom Network Protocol) uygulanan yöntem kullanılır

Input stream'de 3 karakter arka arkaya tekrar ediyorsa, output stream 3 kopyada yazılır. Dördüncü byte ise run değeri yazılır.

3 kopya için (aaa) → aaa0 → 3 byte yerine 4 byte (genişleme)
4 kopya için (aaaa) → aaa1 → 4 byte yerine 4 byte (sıkıştırma yok)
5 kopya için (aaaaa) → aaa2 → 5 byte yerine 4 byte (sıkıştırma var)

N : karakter sayısı M : tekrar eden grup sayısı L : gruplardaki ortalama elemen sayısı

RLE compression performance = [(N - (M*L) + M*3] / N

MNP5 compression performance = [(N - (M*L) + M*4] / N

Örnek

N=1000, M=10, L=5

RLE compression performance = [(N - (M*L) + M*3] / N = (1000-(50)+30)/1000 = 0,98 (%2 sıkıştı)

MNP5 compression performance = [(N - (M*L) + M*4] / N = (1000-(50)+40)/1000 = 0,99 (%1 sıkıştı)

Görüntü Sıkıştırma

Dijital bir görüntü, pixel adı verilen noktalardan oluşur.
Her pixel (1 bit B&W, birkaç bit gray scale, 8 bit color table, 24 bit true color) oluşur

4 bit 16 renk
7 bit 128 renk
8 bit 256 renk

RGB paletleri aynı değere sahiptir.
True Color (red green blue her kanal için 8 bit)

- Pixel'ler, bitmap içerisinde scan line'lara yerleşir.
- İlk bitmap pixel'i resmin sol üst köşesindeki nokta, son bitmap pixel'i de resmin sağ alt köşesindeki noktadır.
- RLE ile bir resim sıkıştırılacak ise nereden başlanacağı önemlidir.
- Görüntü içerisinde rastgele seçilmiş bir pixel'in komşularının kendisiyle aynı renkte olma ihtimali zayıftır.

Compressor bitmap'i **satır satır**, **sütun sütun** veya **zigzag** olarak tarar.

Örnek: Elimizde siyah&beyaz bir resim olsun.

- Compressor ilk okunacak pixel'in beyaz olduğunu kabul eder.
- İlk pixel beyaz değilse output stream 0 adet beyaz olarak yerleşir.

- Resim 17 beyaz, onu izleyen 1 siyah ve 55 beyaz pixel'den oluşmuş ise, output stream'e 17 1 55 diye yerleştirilir.

- Output stream'in başına bitmap'in çözünürlük değeri yazılır.

- Sıkıştırılmış bir stream'in uzunluğu şeklin karmaşıklığına bağlıdır. Şekilde ne kadar fazla detay varsa, sıkıştırma oranı da o kadar kötüdür.

- Uniform bir alanda RLE için sıkıştırma oranı (şeklin çevre uzunluğunun yarısı/alan içerisindeki toplam pixel sayıdır)

- Her scanline için bir giriş bir çıkış noktası olduğundan 2*scan line değeri uniform şeklin çevre uzunluğunu verir.

Palette

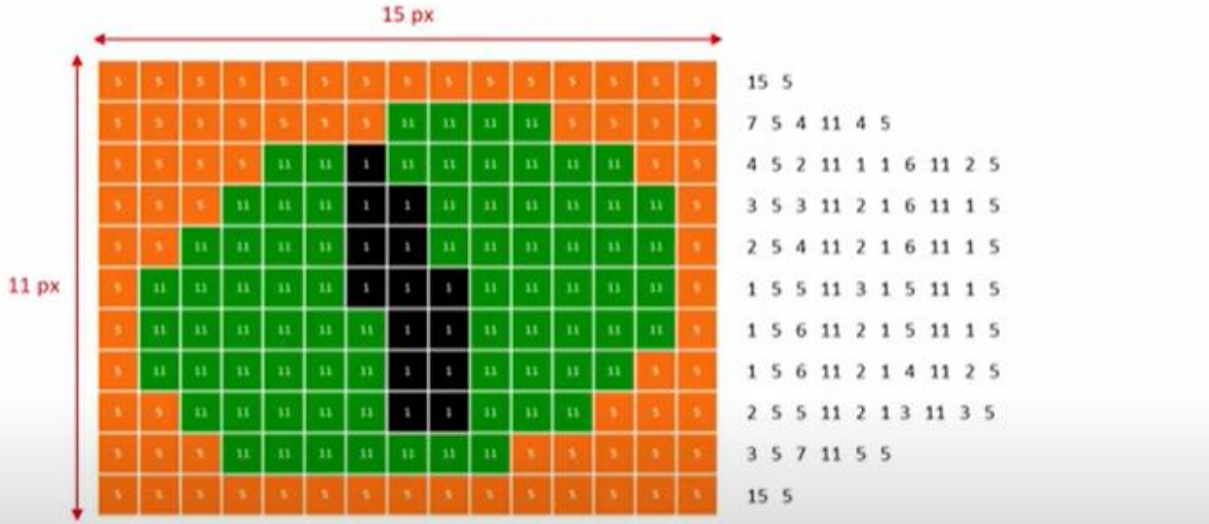
1	000000
2	0000FF
3	A2D9FB
4	87CEFA
5	F86C0F
6	74B9D8
7	00FF00
8	C9E9FD
9	FFFF00
10	5684FC
11	008C00
12	C3D895
13	FFFFFF
14	586C67
15	0C4317
16	9ED7CA

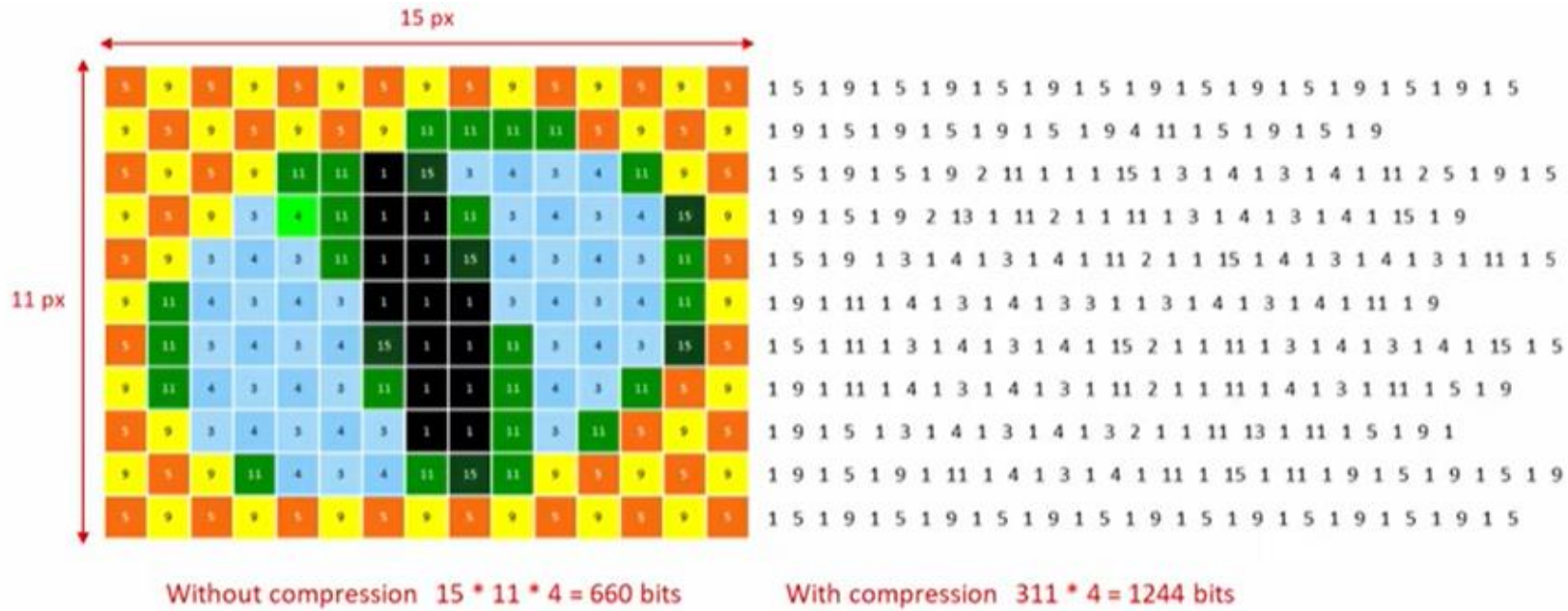
Without compression 15 * 11 * 4 = 660 bits

15 5
7 5 4 11 4 5
4 5 2 11 1 1 1 15 4 7 1 11 2 5
3 5 2 7 1 11 2 1 1 11 4 7 1 15 1 5
2 5 3 7 1 11 2 1 1 15 4 7 1 11 1 5
1 5 1 11 4 7 3 1 4 7 1 11 1 5
1 5 1 11 4 7 1 15 2 1 1 11 3 7 1 15 1 5
1 5 1 11 4 7 1 11 2 1 1 11 2 7 1 11 2 5
2 5 5 7 2 1 1 11 1 7 1 1 3 5
3 5 1 11 3 7 1 11 1 15 1 11 5 5
15 5

Without compression 15 * 11 * 4 = 660 bits

With compression 134 * 4 = 536 bits


15 px
11 px
15 5
7 5 4 11 4 5
4 5 2 11 1 1 6 11 2 5
3 5 3 11 2 1 6 11 1 5
2 5 4 11 2 1 6 11 1 5
1 5 5 11 3 1 5 11 1 5
1 5 6 11 2 1 5 11 1 5
1 5 6 11 2 1 4 11 2 5
2 5 5 11 2 1 3 11 3 5
3 5 7 11 5 5
15 5
Without compression 15 * 11 * 4 = 660 bits
With compression 86 * 4 = 344 bits


15 px
11 px
1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5
1 9 1 5 1 9 1 5 1 9 1 5 1 9 4 11 1 5 1 9 1 5 1 9
1 5 1 9 1 5 1 9 2 11 1 1 1 15 1 3 1 4 1 3 1 4 1 11 2 5 1 9 1 5
1 9 1 5 1 9 2 13 1 11 2 1 1 11 1 3 1 4 1 3 1 4 1 15 1 9
1 5 1 9 1 3 1 4 1 3 1 4 1 11 2 1 1 15 1 4 1 3 1 4 1 3 1 11 1 5
1 9 1 11 1 4 1 3 1 4 1 3 3 1 1 3 1 4 1 3 1 4 1 11 1 9
1 5 1 11 1 3 1 4 1 3 1 4 1 15 2 1 1 11 1 3 1 4 1 3 1 4 1 15 1 5
1 9 1 11 1 4 1 3 1 4 1 3 1 11 2 1 1 11 1 4 1 3 1 11 1 5 1 9
1 9 1 5 1 3 1 4 1 3 1 4 1 3 2 1 1 11 13 1 11 1 5 1 9 1
1 9 1 5 1 9 1 11 1 4 1 3 1 4 1 11 1 15 1 11 1 9 1 5 1 9 1 5 1 9
1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5 1 9 1 5
Without compression 15 * 11 * 4 = 660 bits
With compression 311 * 4 = 1244 bits

RLE Farklı Versiyonları

Örnek :
RLE algoritmasını gray scale resimlerin sıkıştırılmasında kullanalım (run değeri, pixel renk kodu).

8 bitlik gray scale olan bir bitmap

Uncompressed 12 12 12 12 12 12 12 12 12 35 76 112 67 87 87 87 5 5 5 5 5 5 1 ……

Compressed **9** 12 35 76 112 67 **3** 87 **6** 5 1 …

Problem Nedir ?

Grayscale değeri ile tekrar sayısı nasıl ayırt edilecek ?

I

Görüntü 128 (0-127) grays cale ile sınırlı ise, her byte'ın soldaki biti gray scale değeri mi ? Tekrar sayısı mı ? diye kullanılır.

Eğer gray scale değeri 256 (0-255) ise bu değer bir azaltılarak kullanılır (255 renk değeri 254 olarak alınır). 255 sayısı toplam değerden önce yazılarak, arkasından gelen iki byte'ın tekrar sayısı ve pixel değeri olacağı bilgisi gönderilir

Uncompressed 12 12 12 12 12 12 12 12 12 35 76 112 67 87 87 87 5 5 5 5 5 5 1 ……

Compressed **255** **9** 12 35 76 112 67 **255** **3** 87 **255** **6** 5 1 …

II

Her bir byte'ın ne olduğunu gösteren bir bit kullanılır. Bu extra bitler 8'lik gruplar halinde tutulur. Output stream'e tekrar ve renk değerleri gönderilmeden önce bu bilgiye ait flag değerlerini tutan 1 byte'lık bu değer gönderilir. Her 8 byte için fazladan 1 byte gönderilir.

Uncompressed 12 12 12 12 12 12 12 12 12 35 76 112 67 87 87 87 5 5 5 5 5 5 1 ……

Compressed **10000010** **9** 12 35 76 112 67 **3** 87 **100….** **6** 5 1 …

III

Renkli bir resimde her pixel 3 byte ile gösterilir. Red, Green ve Blue.

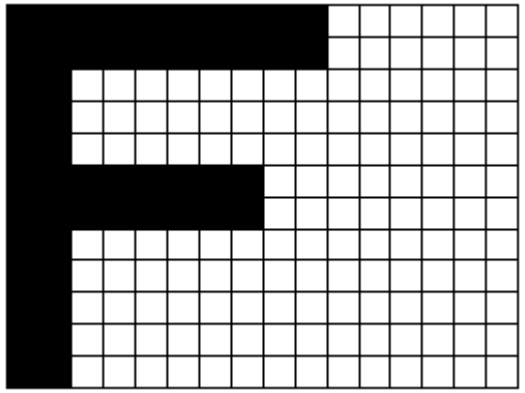
4 pixel değerimiz

(171 85 34) (172 85 35) (172 85 30) (173 85 33)

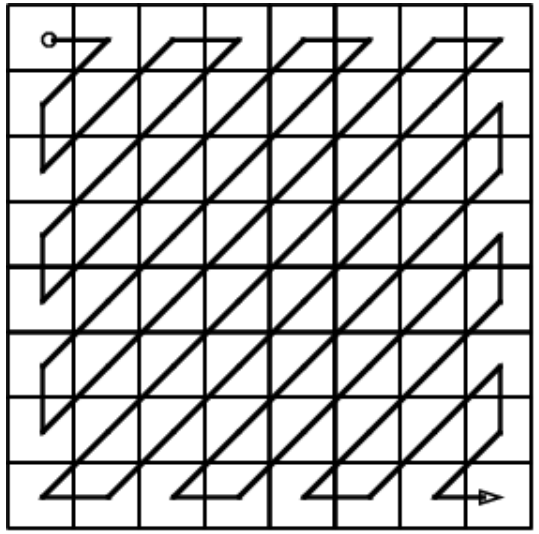
Her palet için ayrı ayrı RLE uygulanır

(171 172 172 173)
(85 85 85 85)
(34 35 30 33)

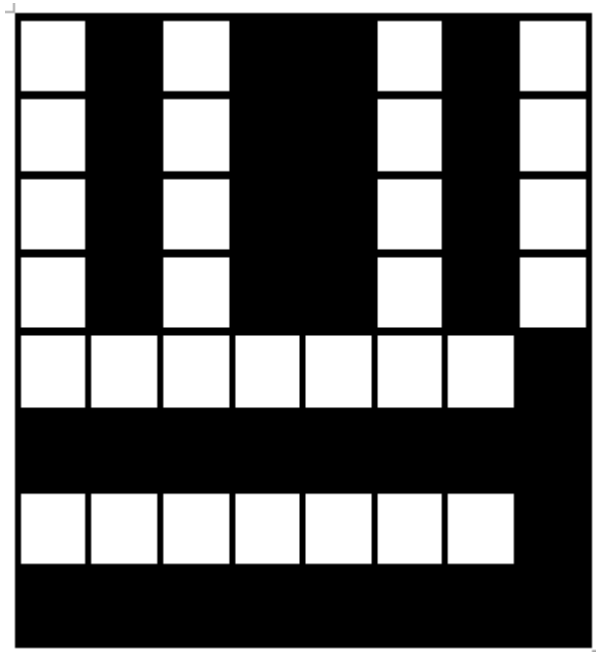
RLE için resim dosyaları satır-satır, sütun-sütun veya zig-zag olarak okunur.



Satır-satır mı / sütun-sütun mu okumalıyız ?



zig-zag okuma



Row-row

1 1 1 2 1 1 1 <eoln> → 8x4 = 32
7 1 <eoln> → 3x2 = 6
0 8 <eoln> → 3x2 = 6

44 bytes

Column-column

5 1 1 1 <eoln> → 5x3 = 15
0 4 1 1 1 1 <eoln> → 7x4 = 28
4 4 <eoln> → 3x1 = 3

44 bytes

RLE- Kayıplı Görüntü Sıkıştırma

- Görüntü sıkıştırmada daha iyi sıkıştırma oranları elde etmek istiyorsak bazı ihmallerde bulunmamız gerekir.
- Kayıplar kullanıcıya göre bazı uygulama alanlarında kabul edilebilir.
- Kısa olan «run» değerleri ihmal edilebilir.
- Kullanıcıya ihmal edilebilecek en büyük «run» uzunluğu sorulur. Kullanıcı üç demiş ise anlamı 1, 2 ve 3 pixel uzunluğundaki pixel'in değeri yerine, en yakın komşu pixel'in değerini kabul etmektir.

Özet

- RLE kayıpsız bir sıkıştırma yöntemidir
- Aynı değere sahip uzun verilerde en yüksek başarıyı verir
- Kodlanması kolaydır
- Çeşitli varyasyonları vardır
- Hızlı çalışır
- Karışık veri üzerinde negatif sıkıştırma yapar