Huffman Encoding

by gunaytemur

Veri Sıkıştırmaya Giriş

- Elde edilen sonuç bakımından en temelde iki tür sıkıştırma yöntemi kategorisi vardır:
  - Kayıplı Sıkıştırma (Lossy Compression)
  - Kayıpsız Sıkıştırma (Lossless Compression)

Kayıplı Sıkıştırma

- Sıkıştırma sonucu elde edilen veriden önceki verinin aynısını elde etmek mümkün değildir.
- Sıkışan verinin bazı bölümleri “kaybolur”.
- Örnekler:
  - MP3: Ham ses verisinden insan kulağının duyamayacağı frekanslar çıkarılır.
  - JPEG: “Göze batmayacak” bozulmalar yapılır.

Kayıpsız Sıkıştırma

- Sıkıştırma sonucu elde edilen veriden önceki verinin aynısını elde etmek mümkündür.
- Örnekler:
  - ZIP, RAR, vs.: Sıkıştırılan dosyaları açtığınızda orijinal dosyaları elde edersiniz. Hiçbir şey "kaybolmaz".
  - **Huffman Encoding**
  - **LZ 778**

Niçin Dosya Sıkıştırmaya İhtiyaç Duyulmuştur?

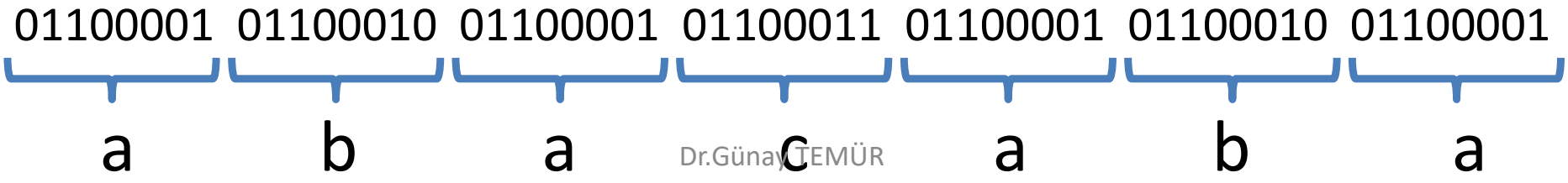
- Minimum alana maksimum veri sığdırmak
- Daha hızlı aktarım sağlamak, erişim süresini azaltmak
- Sıra düzensel verileri daha hızlı işleyebilmek için.

Huffman Kodlamasının Mantığı

- Sıkıştırılacak dosyanın içinde bazı byte değerleri hiç geçmiyor olabilir. Mesela içinde hiç “x” harfi olmayan bir metin dosyası olabilir.
- Belli byte değerleri diğerlerine göre daha sık geçebilir. Mesela “aaaabbbaaaab” gibi bir metinde “a”, “b”den daha sık geçmektedir.
- Dosyada **bulunan** ve **sık geçen** byte değerlerini daha kısa bir bit düzeniyle ifade etme mantığıdır.

Kodlama Başlıyor

- Örneğin metin.txt adlı bir dosyayı sıkıştıralım.
- Bu dosyanın içeriği “abacaba” şeklinde olsun.
- Dosyanın boyutu 7 byte, yani (7 x 8 =) 56 bittir.
- a’nın değeri 97, yani bit düzeni 01100001’dir.
- b’nin değeri 98,yani bit düzeni 01100010’dir.
- c’nin değeri 99, yani bit düzeni 01100011’dir.
- O halde metin.txt dosyasının bit düzeni:



1. Adım: Sayım

- Dosyada bulunan her byte değerinin dosyada kaç kere geçtiği tespit edilir.
- [metin.txt]: “abacaba”
- a: 4 adet
- b: 2 adet
- c: 1 adet

2. Adım: Kümeleme

- Elimizdeki byte değeri türlerinden en seyrek geçen iki tanesi seçilir, bu ikisi birleştirilir ve bu birleşik değer diğerlerinin arasına geri konur.
- Bu işlem, tek bir değer çeşidi kalana kadar devam eder.

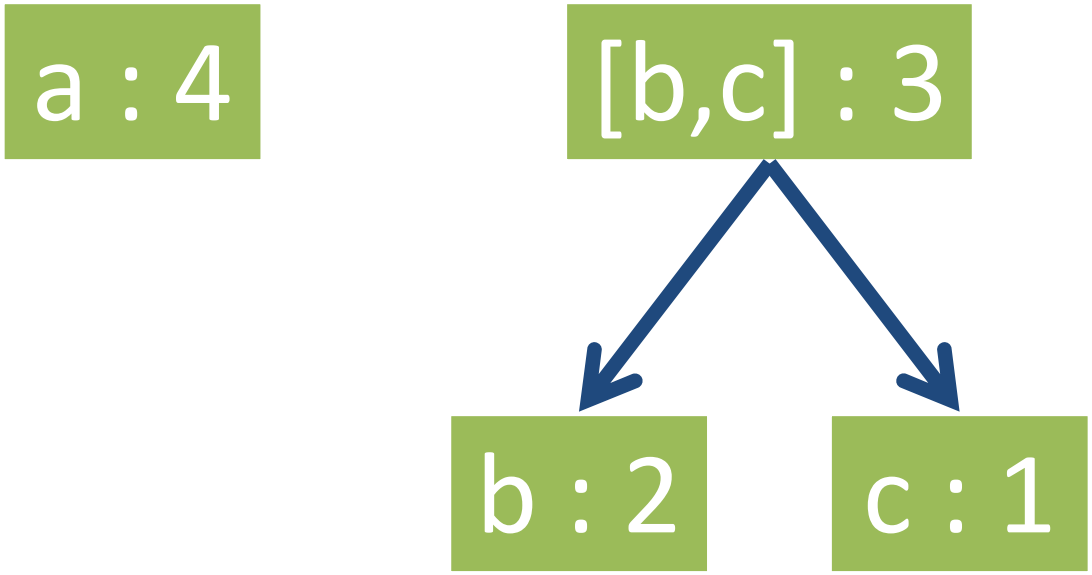
a : 4

b : 2

c : 1

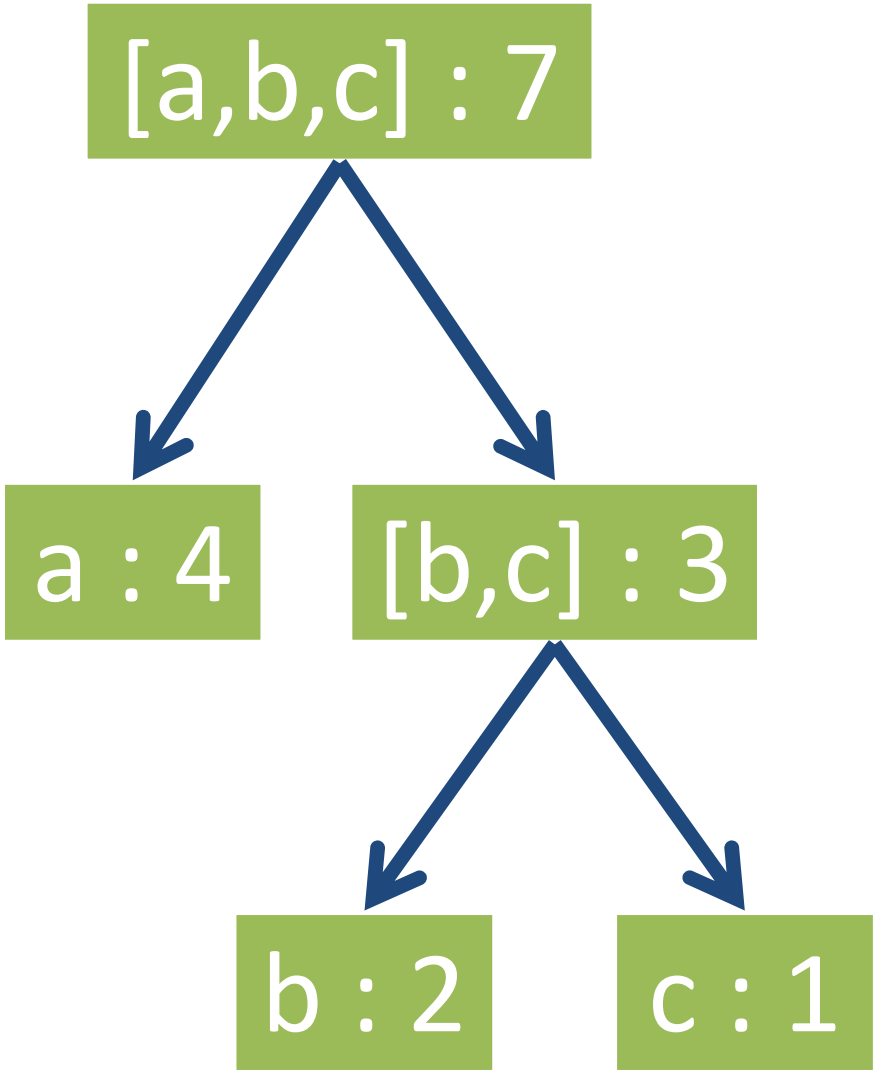
Örnek Kümeleme İşlemi (1. Tur)

- En seyrek geçen iki karakter olan b (2) ve c (1) çekilir ve birleştirilir. Birleşim a'nın yanına geri atılır.



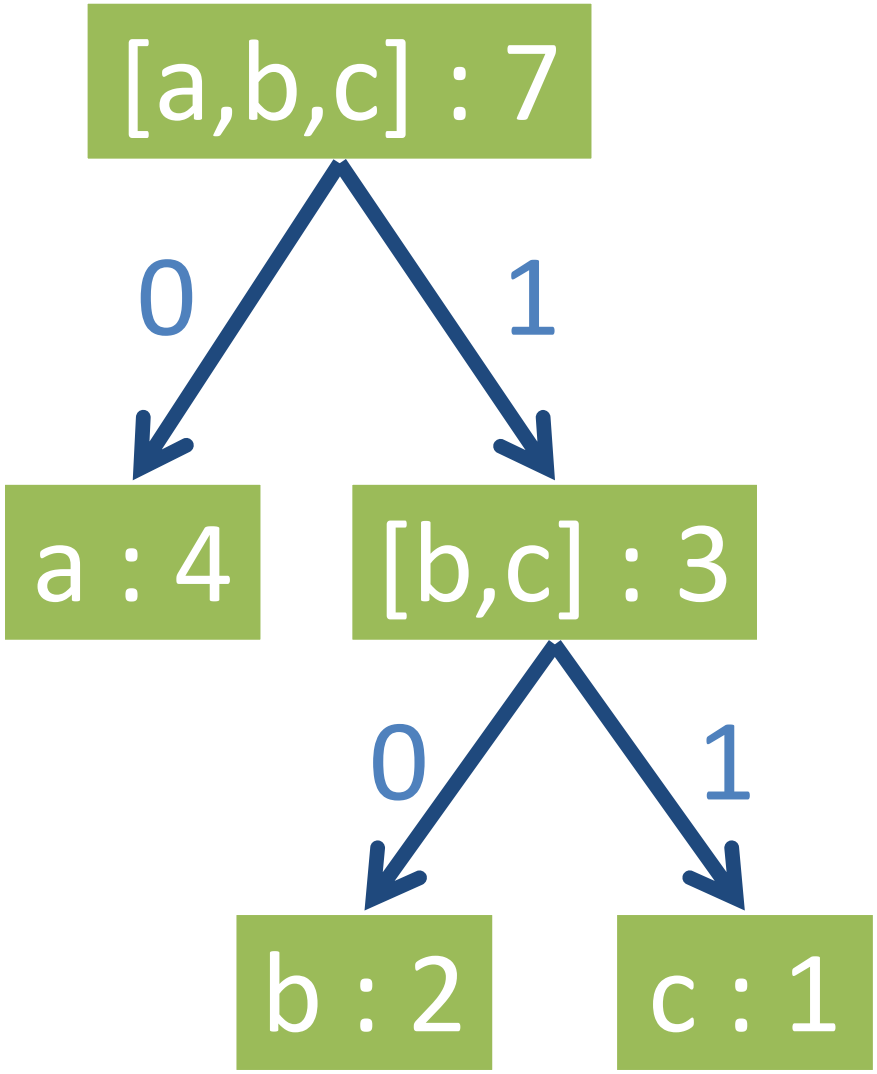
Örnek Kümeleme İşlemi (2. Tur)

- En seyrek geçen iki karakter olan a (4) ve [b,c] (3) çekilir ve birleştirilir. Yeni birleşim geri atılır.
- Bu işlemin sonunda tek bir değer çeşidi ([a,b,c] : 7) kaldığından kümeleme aşaması sonlanır.



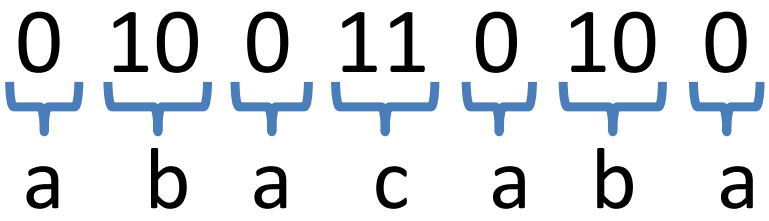
3. Adım: Kodlama

- Kümeleme işlemi sonucunda elde edilen ağaç yapısının sola giden oklarına “0”, sağa giden oklarına ise “1” denir.
- Her bir karakterin bit düzeyindeki yeni ifadesi, en tepeden o karaktere giden yoldaki 0 ve 1’lerden oluşur.
- Yani a: 0, b: 10, c: 11



Sıkıştırma İşleminin Sonucu

- Elde edilen yeni bit düzeyindeki ifadelere göre (a: 0, b: 10, c: 11) dosya içeriği tekrar yazılır:



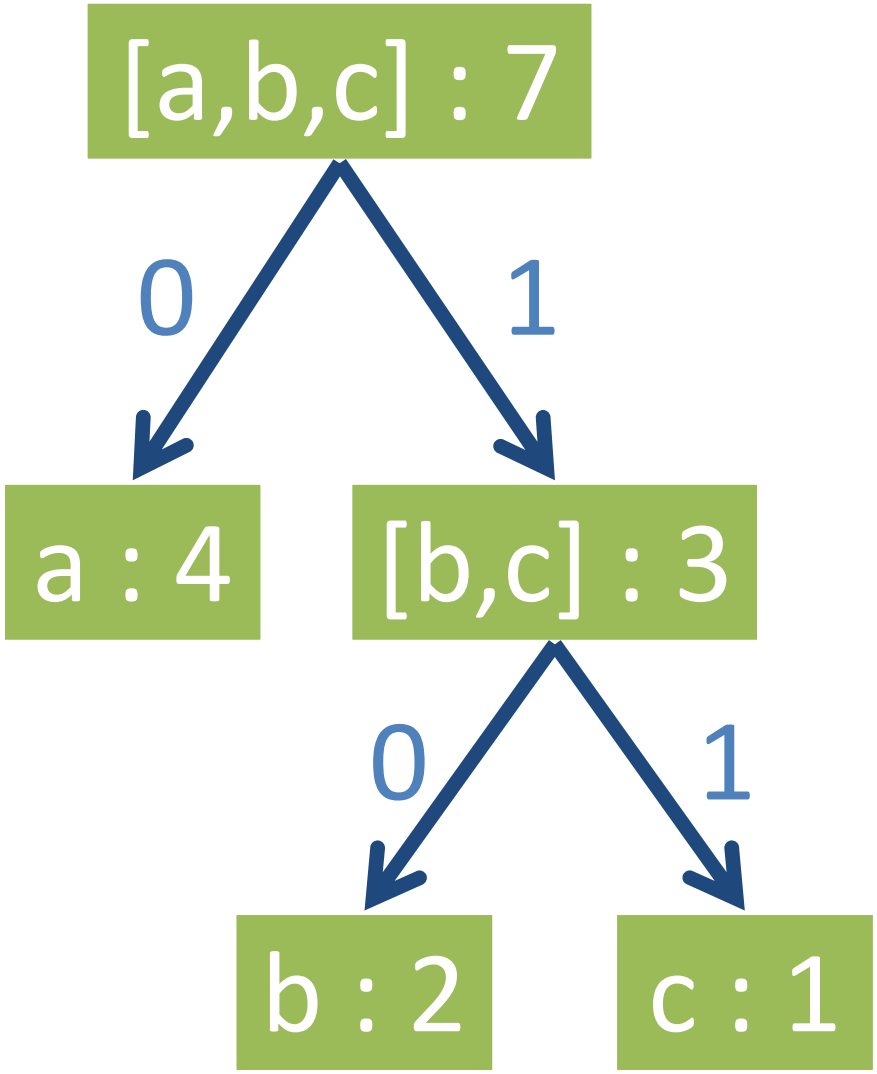
- Dosyamızın ilk içeriği 56 bit (7 byte) iken, sıkıştırıldıktan sonraki boyutu gördüğünüz gibi 10 bit (2 byte) oldu.
- Yaklaşık olarak %71 oranında sıkıştırma gerçekleşti.

Sıkıştırılmış Dosyanın Açılması

- Elimizdeki sıkıştırılmış ve bit düzeyindeki veri:

  0100110100

- Bu veriyi açacak olan anahtar ise yanda verilen ağaç yapısı.
- Tek yapmamız gereken elimizdeki bit verisini ağaç üzerinde takip edip bir sona geldiğimizde vardığımız karakteri basıp tekrar ağacın başına dönmek.

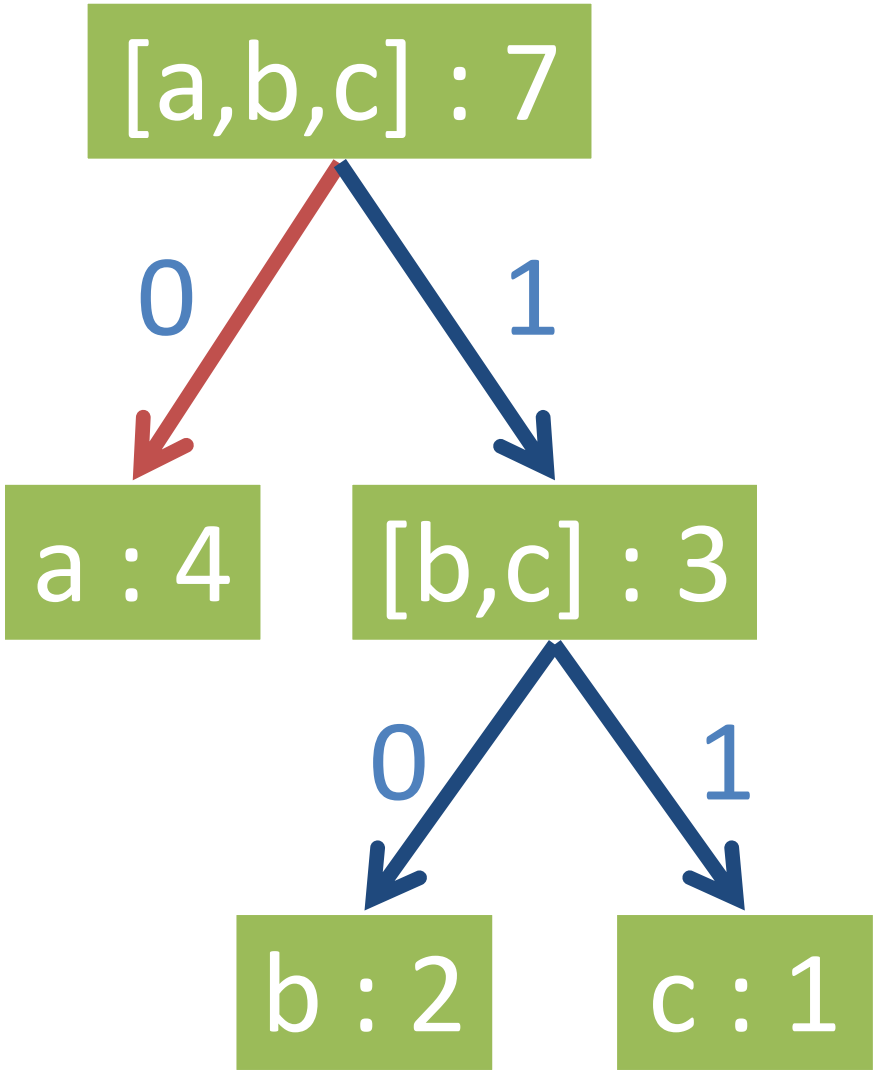


Sıkıştırılmış Dosyanın Açılması

0100110100

a

- İlk bit olan “0”ı okuduk ve ağaç üzerinde “a” karakterine vardık.
- O halde “a” karakterini basıp ağaçta en başa dönüyoruz.

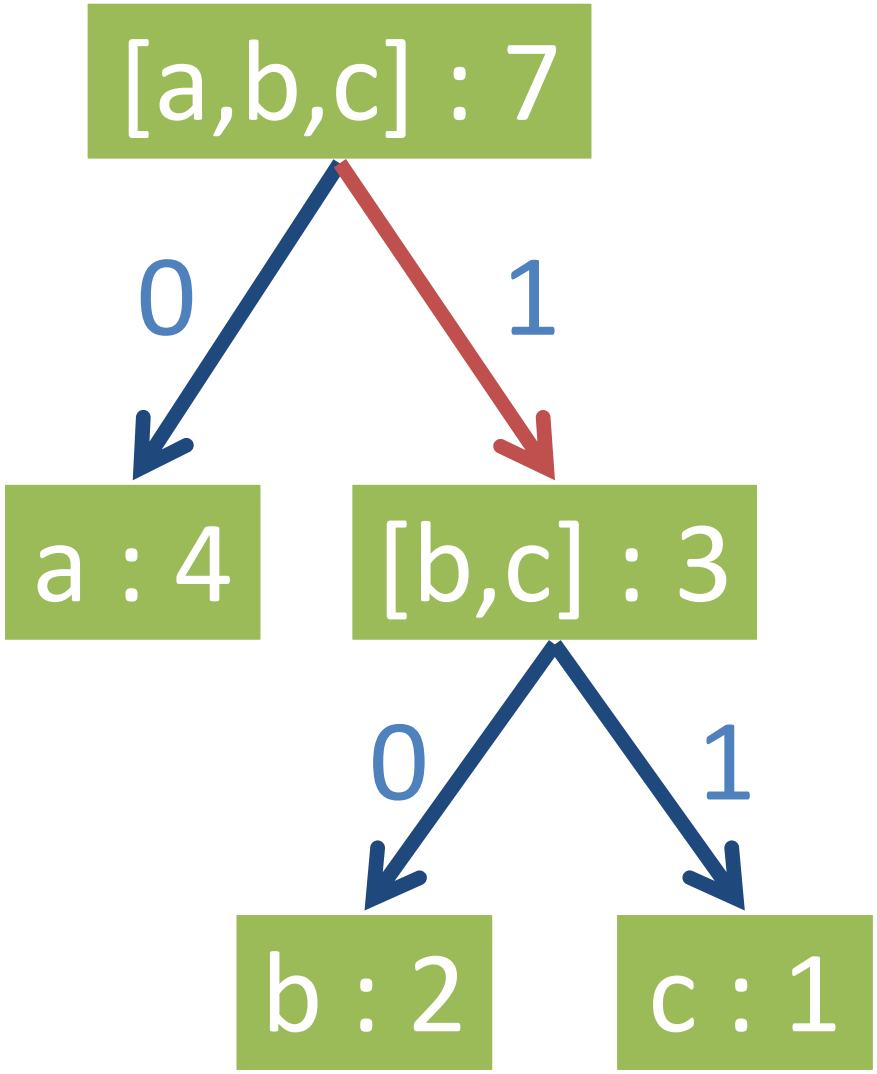


Sıkıştırılmış Dosyanın Açılması

0100110100

a

- Bir sonraki bit olan “1”i okuduk ve ağaç üzerinde herhangi bir sona varamadık.
- O halde bulunduğumuz yerden devam ediyoruz.

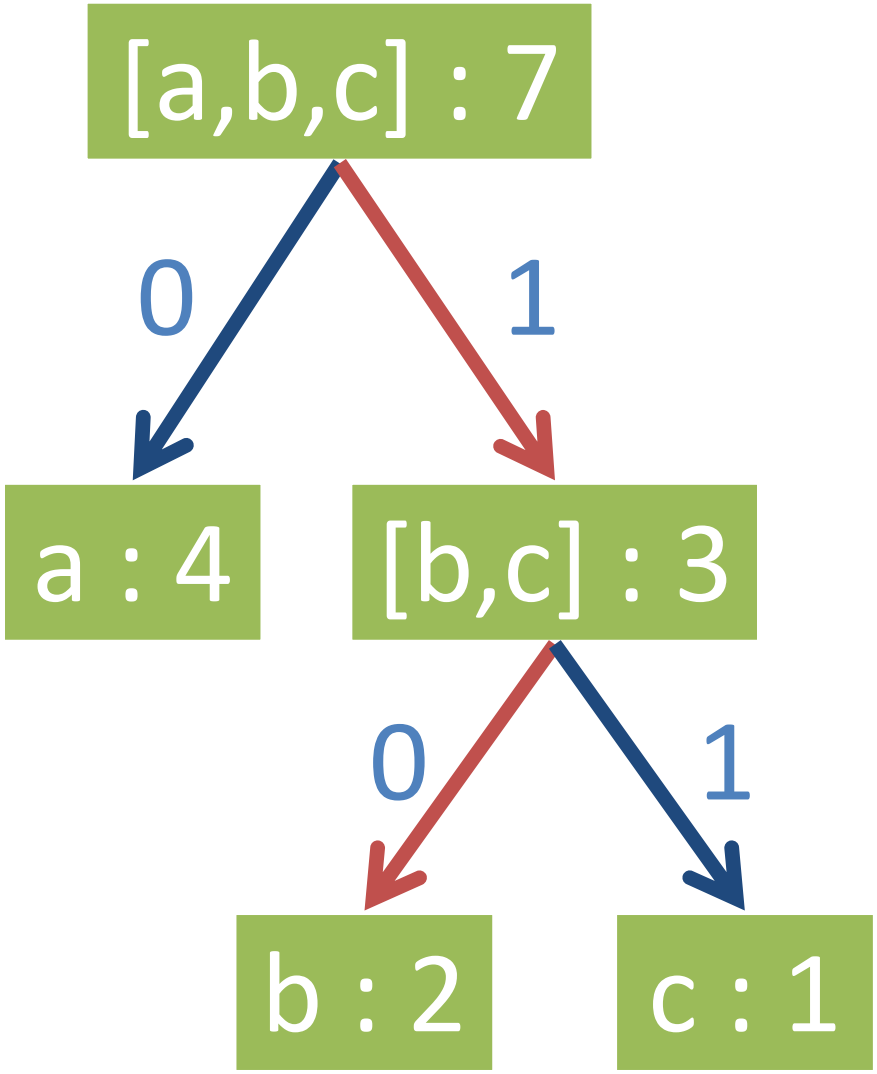


Sıkıştırılmış Dosyanın Açılması

01**0**0110100

a**b**

- Bir sonraki bit olan “0”ı okuduk ve ağaç üzerinde “b” karakterine vardık.
- O halde “b” karakterini basıp ağaçta en başa dönüyoruz.

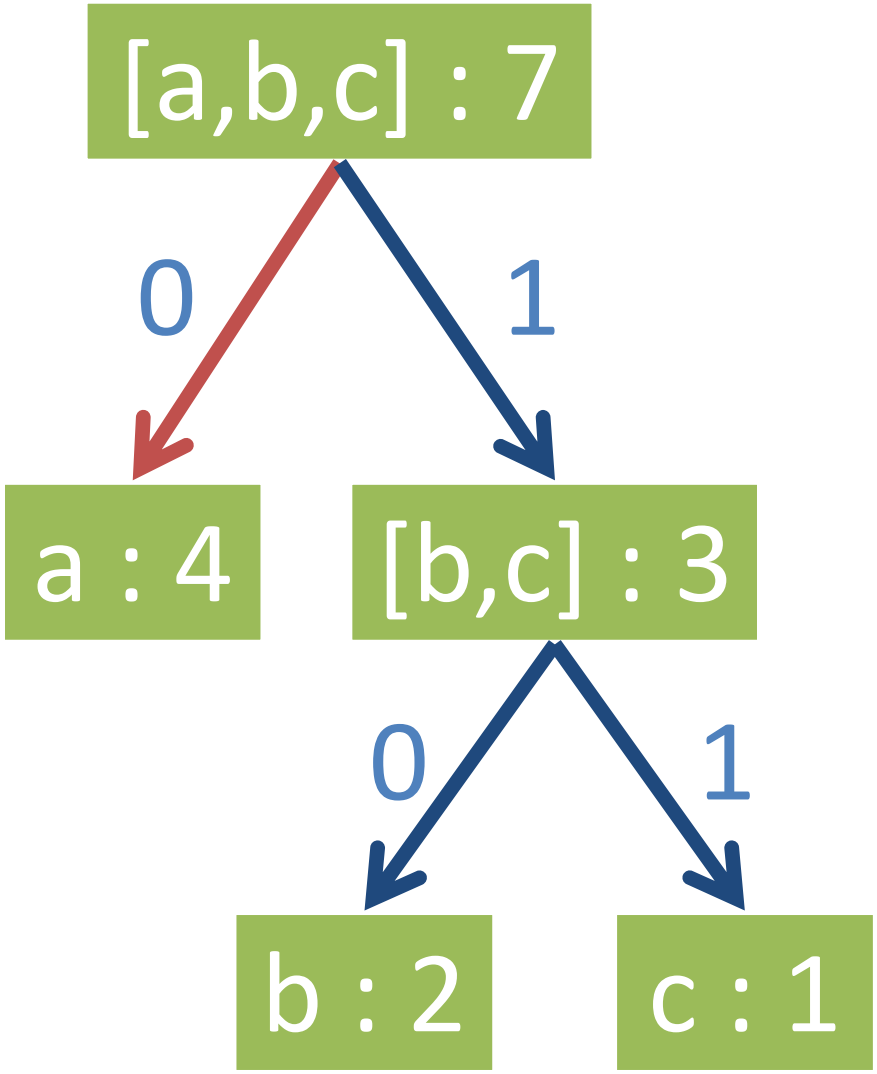


Sıkıştırılmış Dosyanın Açılması

0100110100

aba

- Bir sonraki bit olan “0”ı okuduk ve ağaç üzerinde “a” karakterine vardık.
- O halde “a” karakterini basıp ağaçta en başa dönüyoruz.

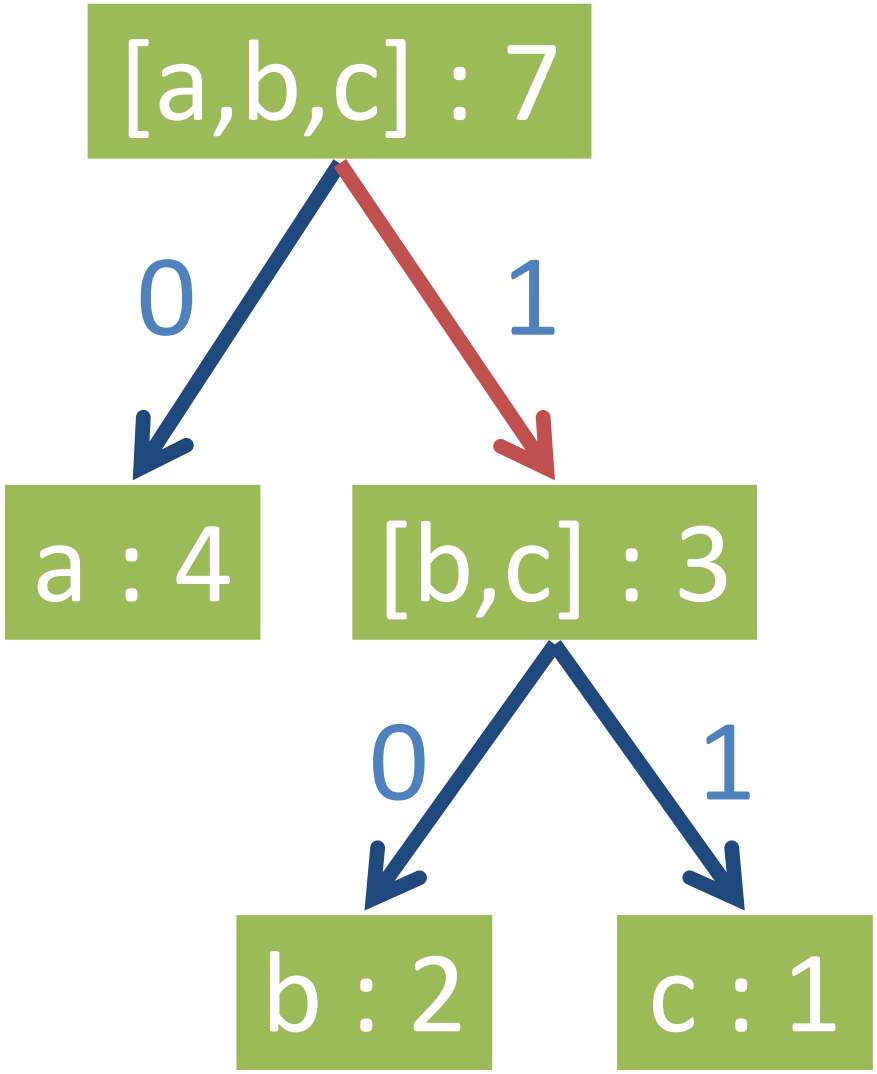


Sıkıştırılmış Dosyanın Açılması

0100110100

aba

- Bir sonraki bit olan “1”i okuduk ve ağaç üzerinde herhangi bir sona varamadık.
- O halde bulunduğumuz yerden devam ediyoruz.

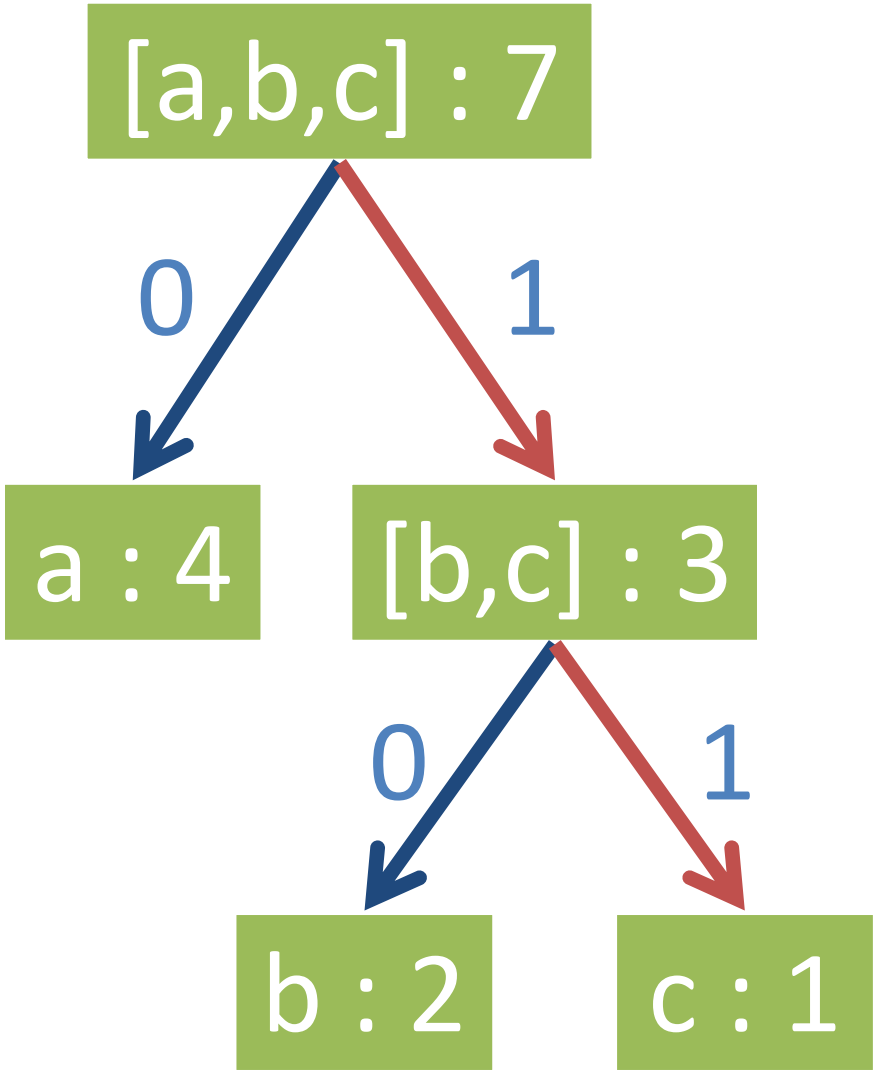


Sıkıştırılmış Dosyanın Açılması

0100110100

abac

- Bir sonraki bit olan “1”i okuduk ve ağaç üzerinde “c” karakterine vardık.
- O halde “c” karakterini basıp ağaçta en başa dönüyoruz.

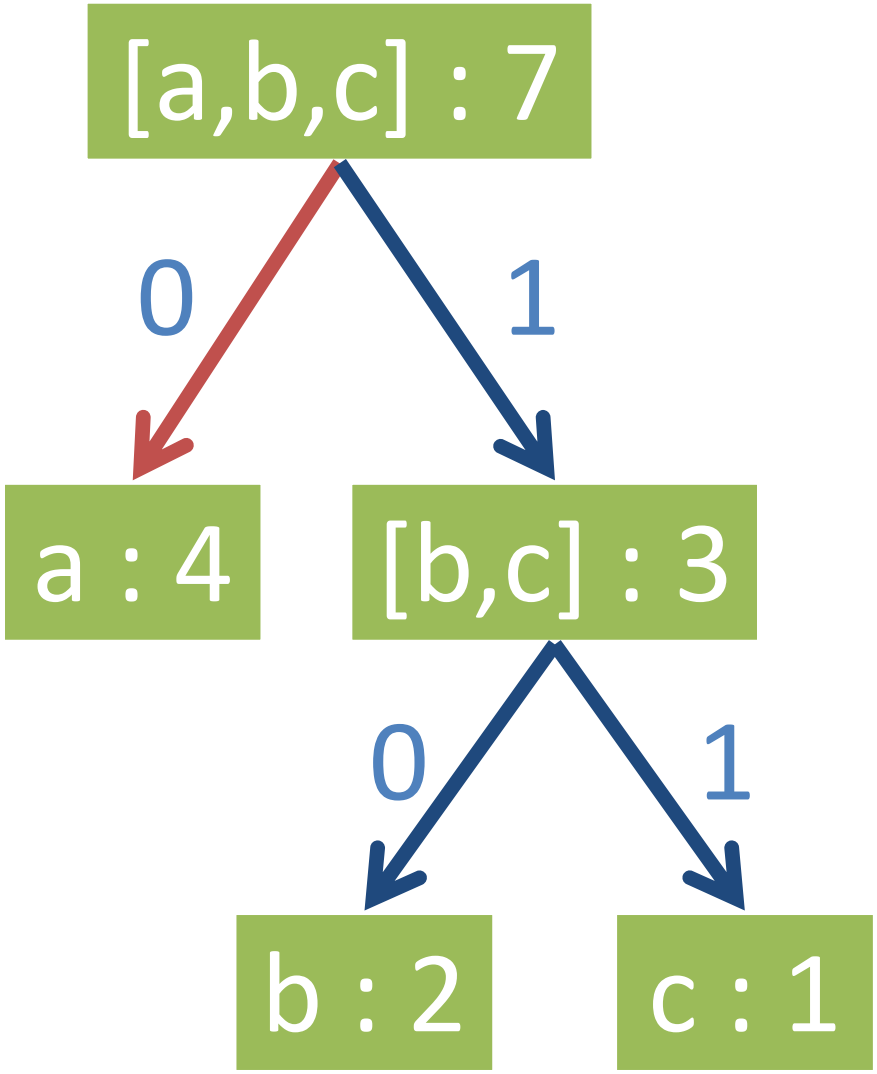


Sıkıştırılmış Dosyanın Açılması

0100110100

abaca

- Bir sonraki bit olan “0”ı okuduk ve ağaç üzerinde “a” karakterine vardık.
- O halde “a” karakterini basıp ağaçta en başa dönüyoruz.

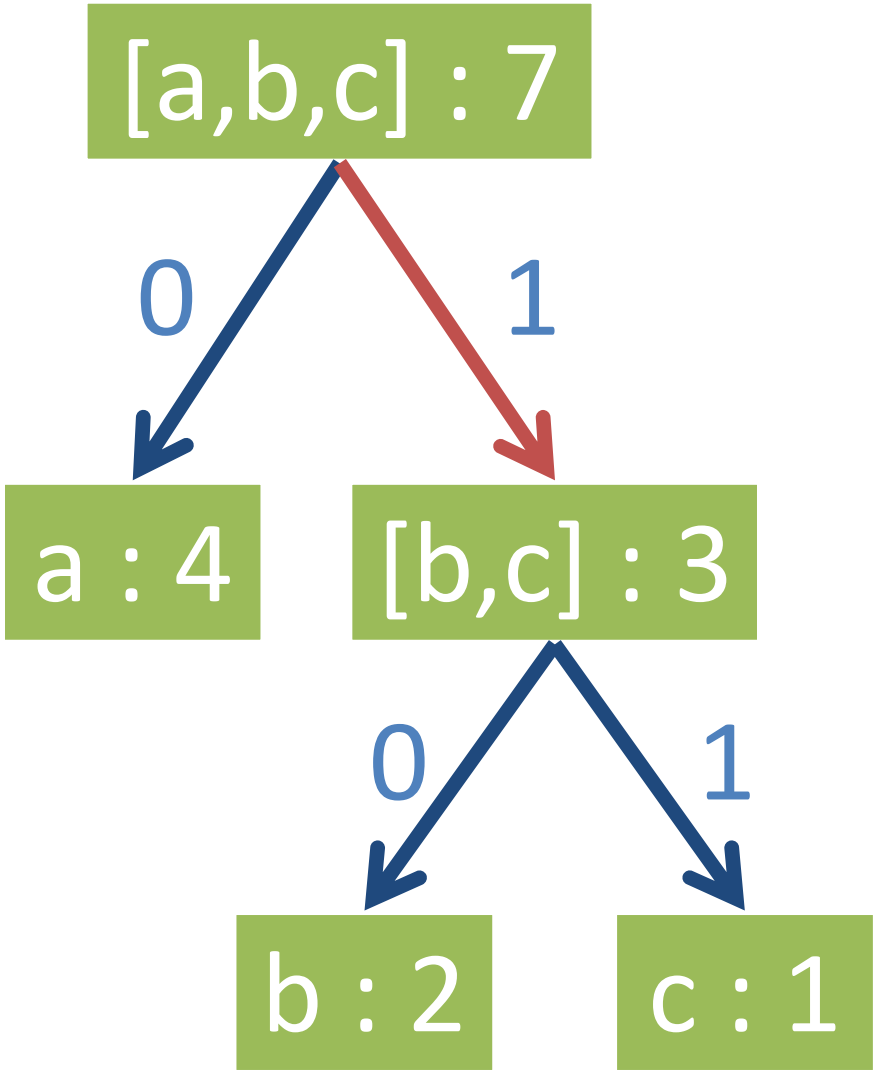


Sıkıştırılmış Dosyanın Açılması

0100110100

abaca

- Bir sonraki bit olan “1”i okuduk ve ağaç üzerinde herhangi bir sona varamadık.
- O halde bulunduğumuz yerden devam ediyoruz.

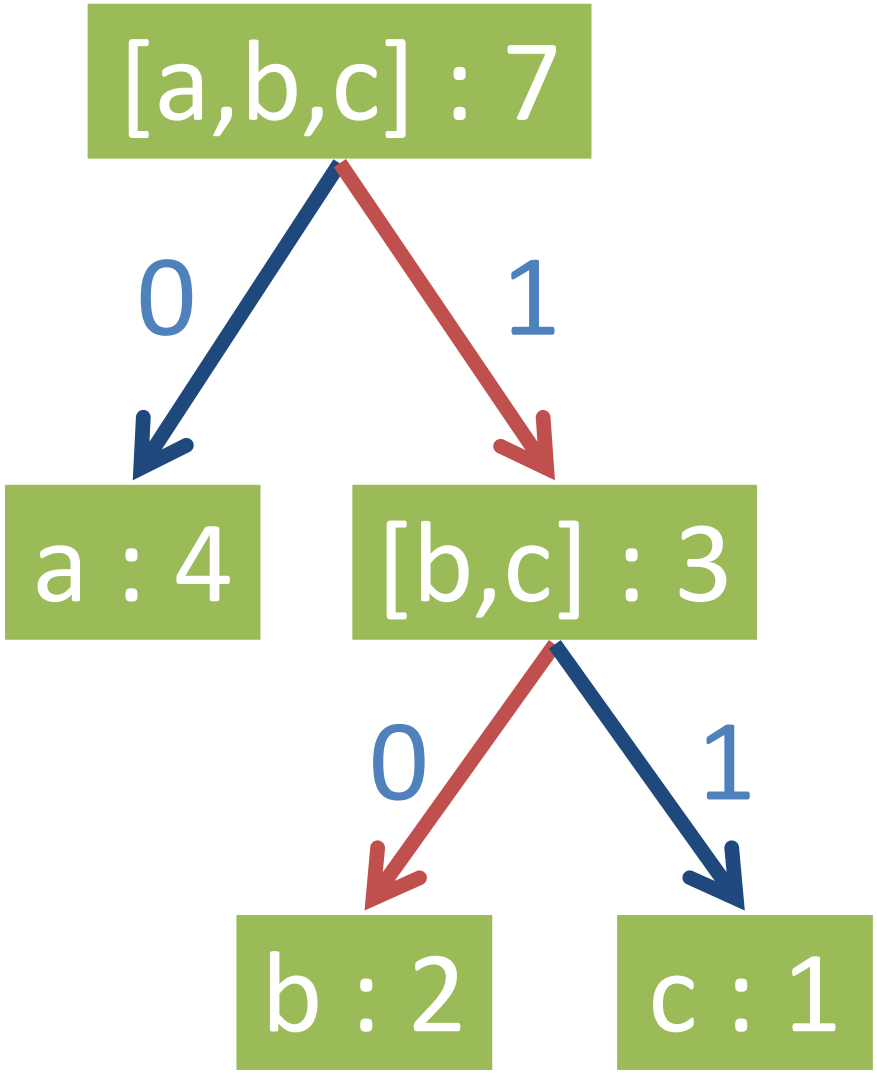


Sıkıştırılmış Dosyanın Açılması

0100110100

abacab

- Bir sonraki bit olan “0”ı okuduk ve ağaç üzerinde “b” karakterine vardık.
- O halde “b” karakterini basıp ağaçta en başa dönüyoruz.



Sıkıştırılmış Dosyanın Açılması

0100110100

abacaba

- Son bit olan “0”ı okuduk ve ağaç üzerinde “a” karakterine vardık.
- O halde “a” karakterini basıp ağaçta en başa dönüyoruz.
- Bit dizisi sona erdiği için kod açma işlemini sona erdiriyoruz.