İSTANBUL TİCARET ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
BİLGİSAYAR SİSTEMLERİ LABORATUVARI

Veri Sıkıştırma Yöntemleri ve Huffman Kodlama ile Veri Sıkıştırma

1. Deney Amacı

Veri sıkıştırma sadece bilgisayar bilimlerinde uygulama alanı olmayıp değişik disiplinler içinde, kullanılan haberleşme kanalı üzerindeki veri iletim miktarını arttırmak için kullanılmıştır. Bu deneyde bilgisayar ve haberleşme dünyasında sıklıkla kullanılan veri sıkıştırma yöntemleri ve tipleri tanıtılıp, Huffman kodlama kullanan bir sıkıştırma uygulaması yapılacaktır.

2. Veri Sıkıştırma Yöntemleri

Veri Sıkıştırma tanım olarak, bir bilginin orijinal halinden daha az yer kaplayacak şekilde kodlanması olarak tanımlanabilir. Bir iletim kanalından iletilecekse genelde kodlama, diğer durumlarda veri sıkıştırma olarak adlandırlır.

Veri sıkıştırılmada temel prensip, sıkıştırılacak dosya içinde gereksiz (unnecessary) ya da tekrar eden kısımların (redundancy) bulunmasıdır. Bu tür kısımların olmadığı bir dosyayı sıkıştırmak bilinen yöntemlerle mümkün olmayacaktır.

Veri Sıkıştırma çeşitli kaynaklara göre farklı kategorize edilmesine rağmen genel olarak kayıplı ve kayıpsız olmasına göre iki grupta incelenebilir. Kayıplı ve kayıpsız sıkıştırma yöntemlerinin ne demek olduğu deney föyünün ilerleyen sayfalarında açıklanmıştır. Farklı şekilde uygulama alanlarına (resim, ses, video vb) göre de kategorize edilebilir. [1] e sıkıştırmada kullanılan yöntemlerin benzerliğine göre kategorilerde inceleniştir.

Kayıplı sıkıştırmaların bir kısmı [4]

Jpeg, MPEG-2 (DVD), MPEG-4 (DivX, Xvid..) , H.265, MPEG-1 VCD, H.264 Blu-ray, HD DVD, MP3, Vorbis, JPEG 2000

Kayıpsız sıkıştırmaların bir kısmı [4]

Lempel-Ziv (LZ) , Lempel-Ziv–Renau LZR (ZIP), DEFLATE (PKZIP, Gzip, PNG), LZW (Lempel–Ziv–Welch) GIF için, document compression standard DjVu, Aritmetik Kodlama WMA 9 Lossless, FLAC (free lossless audıo codec), ALAC Apple, DVD-Audio, Dolby TrueHD, JPEG 2000

Kullanılan Yöntemlere ve Uygulama Alanlarına Göre Kategorileri[1]

- İstatiksel Yöntemle
  - Huffman Coding
  - Facsimile Compression
  - Arithmetic Coding
  - Adaptive Arithmetic Coding
- Sözlük Kullanan Yöntemler

    - LZ77 (Sliding Window) Lempel-Ziv
    - LZSS
    - LZ78
    - LZW Lempel–Ziv–Welch
- Resim Sıkıştırma
    - Progressive Image Compression
    - JPEG
    - JPEG-LS
- Wavelet (dalgacık) yöntemleri
    - Averaging and Differencing
    - The Haar Transform
    - Subband Transform
    - Filter Banks
    - DWT
    - The Daubechies Wavelets
    - SPIHT
- Video Sıkıştırma
    - MPEG-2
- Ses Sıkıştırma
    - MPEG-1 Audio

Veri sıkıştırma yöntemlerini tanıtmaya başlamadan önce bu alanda sıklıkla kullanılan bazı terimler ve anlamları üzerinde durulacaktır.

Kayıplı Kayıpsız Sıkıştırma, Bazı sıkıştırma yöntemleri kayıplıdır. Bu şekilde bazı bilgiler kaybedilerek daha iyi sıkıştırma elde edilir. Bu tür yöntemlerde sıkıştırılan veri, tekrardan açıldığında orijinal veri ile aynısı elde edilmez. Bu şekilde bir sıkıştırma yöntemi ancak resim, video ve ses verileri üzerinde uygulandığında anlam ifade etmektedir. Eğer kayıp az ise kullanıcı tarafından fark edilmeyecektir. Buna karşılık bilgisayar dosyalarındaki 1 bitlik bir bilgi kaybı dahi o dosyayı kullanılamaz hale getirebilir. Bu tür dosyaların sıkıştırılmasının o yüzden kayıpsız olması gereklidir. Sıkıştırılmış veriler açıldığında, orijinal hale dönüldüğünde bu tür yöntemlere kayıpsız sıkıştırma yöntemleri denir. Metin dosyaları sıkıştırılırken iki konuya dikkat edilmelidir. 1 sıkıştırılacak veri, bir programlama dilinin kaynak koduna ait ise içindeki boşluklar zaten derleyici tarafından ihmal edileceği için, bu kısımlar sıkıştırmada kullanılabilirler. 2 bir kelime işlem programının çıktısı metin belgesi olarak kaydedilmek istenirse, font bilgisi gibi bilgiler ihmal edilebilir.

Sıkıştırıcı kodlayıcı, giriş olarak tekrarın çok olduğu (redundency) veri alınıp düşük tekrarlı (low redundency) dosya oluşturan koddur. Kodlama anlamı çok genel olmasına rağmen burada veri sıkıştırma olarak kullanılacaktır.

Adaptif olmayan sıkıştırıcı, bu yöntem verileri sıkıştırırken kullandığı tablo parametrelerini ya da metodunu sıkıştırılacak veriye göre değiştirmezler. Bazı sıkıştırma algoritmaları ham veriyi inceleyip çalışmasını ona göre değiştiren yöntemlere ***adaptif sıkıştırıcı yöntemler*** denir. Huffman kodlama bu tip bir yöntemdir. Bazı sıkıştırma algoritmaları iki fazlıdır. İlk fazda sıkıştırılacak veri hakkında istatistiksel bilgi toplanır, diğer fazında bu verilere elde edilen parametre ve kodlara bağlı olarak sıkıştırma gerçekleştirilir. Bu ***yöntemlere yarı Adaptif yöntemler*** olarak ifade edilir.

Simetrik sıkıştırma yöntemlerinde, sıkıştırma ve sıkışmış verinin açılması aynı temel algoritmayı <zıt> yönlerde çalıştıran yöntemlerdir.

Sıkıştırma performansı için değişik büyüklükler kullanılır. Bu faktörler sırası ile anlatılacaktır:

1. En sık kullanılanı sıkıştırma oranı (compression ratio) dur denklem 1 deki gibi ifade edilir.

$$Compression\ Ratio = \frac{Size\ of\ output\ file}{Size\ of\ input\ file} \quad (1)$$

örneğin 0.6 değerinin anlamı sıkıştırmadan sonra veri orijinal verinin %60 ı kadar yer kaplıyor olacaktır. Aynı şekilde 1 den büyük değerler sıkıştırılan dosya büyüklüğünün orijinal veriden daha fazla olacağı anlamındad olup ***negatif sıkıştırma*** olarak adlandırılır. Sıkıştırma oranı ***bpb*** (bit per bit) birimle de ifade edilebilir. Resim sıkıştırmada ise benzer bir birim kullanılır ***bpp*** (bits per pixel). Metin dosyaları için bu ifade ***bpc*** olur (bits per character: Bir karakteri sıkıştırmak için ortalama gerekli bit miktarı).

Sıkıştırma oranı ile iki terimden de söz edilmesi gereklidir. Bunlardan ilki ***bitrate*** bpb ve bpc için genel bir terimdir. Bundan dolayı veri sıkıştırmadaki temel hedef girilen herhangi bir veriyi düşük bitrate'lerde ifade etmektedir. ***Bit budget*** sıkıştırılmış bir dosyada bir bitin görevini ifade eder.

2. Sıkıştıra oranının tersi ise Sıkıştırma Faktörü (***compression Factor)***

$$Compression\ Factor = \frac{Size\ of\ input\ file}{Size\ of\ output\ file} \quad (2)$$

Bu durumda 1 den büyük değerler sıkıştırmayı, küçük değerler genişlemeye işaret edecektir.

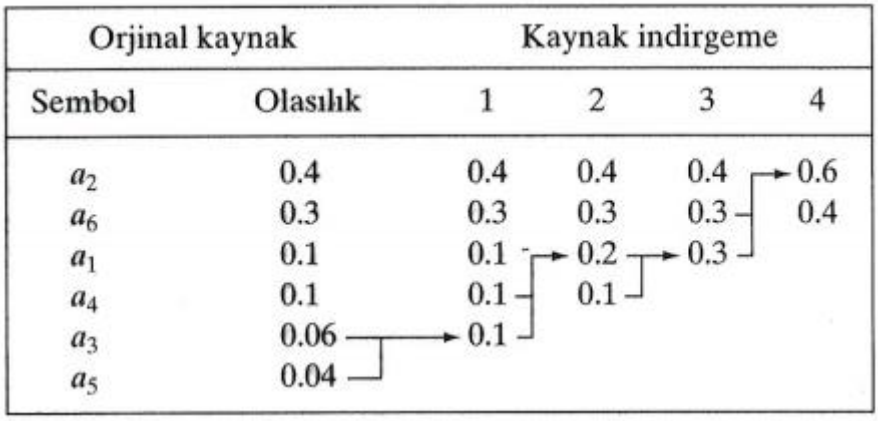
3. 100x(1-compression ratio) da anlamlı bir ölçüm performans göstergesidir. 60 değeri çıkış dosyasının orijinal dosyanın %40 ı kadar yer kapladığı anlamındadır. Yada sıkıştırma ile %60 lık tasarruf edilmiştir.
4. Resim sıkıştırmada bpp sıkılıkla kullanılmaktadır. Bu Bir pikseli sıkıştırmak için ortalama gerekli olan bit miktarını vermektedir.

Olasılık modeli, istatistiksel veri sıkıştırma metotlarında önemli bir kavramdır. Bazen sıkıştırma algoritması iki kısımdan oluşmaktadır **probability model** ve sıkıştırmanın kendisini ifade etmektedir.

Entropi 1948 yılında Claude Shannon Tarafından Bell Laboratuarında Informasyon Teorisi oluşturulmuştur. Veri sıkıştırmayı anlamak için bilinmesi gereken en önemli enformasyon teorisi kavramı entropidir. Olasığı P olan bir ***a*** sembolünün entropisi $-P\log_2 P$ olarak ifade edilir. Örneğin olasılığı 0.5 olan a sembolünün entropisi 0.5 çıkacaktır. **a1** den **an**'e kadar tüm sembollerin Entropisi ise, P1 den Pn e o sembollerin olasılıkları olacak şekilde ifade edilirse $\sum_n -P_i \log_2 P_i$ eşitliği ile bulunur. Çoğunlukla bir sistemdeki rastgelelik ve düzensizlik olarak tanımlanan bu istatistikten teknolojide birçok alanda yararlanılır.

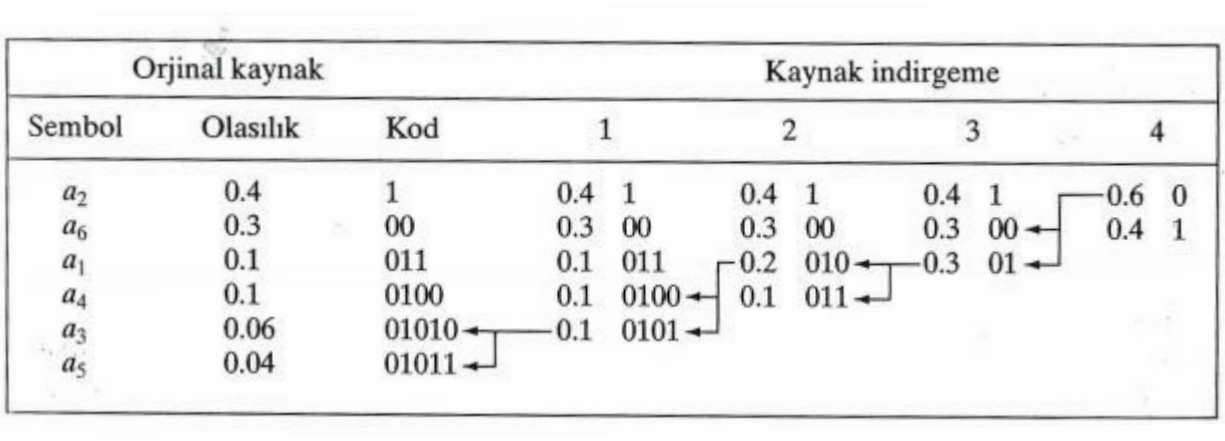
3. Huffman Kodlama

Bilgi kaynağını içindeki sembolleri kodlarken Huffman kodlama kaynak başına en küçük sayıdaki kod sembolü üretecektir [3]. Huffman kodlamanın ilk aşamasında var olan sembollerin olasılıkları bulunarak sıralanır. Bu semboller arasında en düşük olasılığa sahip olan ikisi kodlanmak üzere yeni bir sembolde birleştirilir. Sonradan oluşan indirgenmiş kolar tekrardan sıralanır ve en düşük olasılığa sahip ikisi toplanır. Bu işlem toplanan olasılıklar sonucu 1 olana kadar devam edilir. Şekil 1 de örnek verilen 6 sembol için olasılıkları en büyük olanı en üstte olacak şekilde en solda sıralanmıştır. En düşük olasılığa sahip a5 ve a3 sembolleri olasılıkları toplanıp indirgenmiş birleşik sembol elde edilir.



Şekil 1. Huffman kaynak sembol indirgenmesi

İkinci aşama bunların kodlanması olacaktır. En küçük kaynaktan başlayarak asıl sembol kaynaklarına doğru yerleştirme yapılır. İkili sbir kodlama için kullanılacak en az uzunluk elbette ki 0 ve 1 olacaktır. Şekil 2 de bu kodlamanın nasıl uygulandığı gözükmektedir. Şekilde gözüktüğü gibi bu iki sembole Şekil 2 nin en sağında atanmıştır. 0.6 aslında bir önceki adımda ki iki olasılığın toplamıydı, 0.6 yı kodlamak için kullandığımız 0 şimdi onu oluşturan iki sembol için de ön kod olarak kullanılacak şekilde, altta kalan sembollere yeniden 0 ve 1 sembolleri verilir. Bu her bir indirgenmiş sembol için devam ederek en sonunda orijinal sembollere kadar devam edilir. Burada dikkat edilecek husus dallanmada kodlama yaparken, 0 kodunu büyük olan tarafa vermişsek diğer dallarda da aynı prensibe dikkat etmeliyiz.



Şekil 2. Huffman code atama prosedürü

4. Huffman Kod Çözümlenmesi

Kod doğru bir şekilde üretildikten sonra kodun çözülmesi bir tablo bakma sayesinde hatasız bir şekilde gerçekleştirilecektir. ***Block Code*** dur çünkü, her kaynak kod belirli bir kod serisine eşleştirilmiştir. Kodlar soldan sağa doğru yaklaşımla çözülmelidir. Şekil 2 de elde edilen Huffman Kodları ile aşağıdaki kodlanmış veri dikkate alındığında

010100111100

Kod çözülmüş hali aşağıdaki gibi olacaktır

a3a1a2a2a6

Referanslar

[1] David Salomon "A Guide To Data Compression Methods" 2001 Springer

[2] Entropi http://tr.wikipedia.org/wiki/Entropi

[3] Rafael C. Gonzalez, Richard E. Woods Digital Image Processing 2008 (third Edition) Pearson International

[4] Data Compression http://en.wikipedia.org/wiki/Data_compression