Assignment 07: Compression

Brock Davis

User Interface

0: Convert standard images to .bdif images

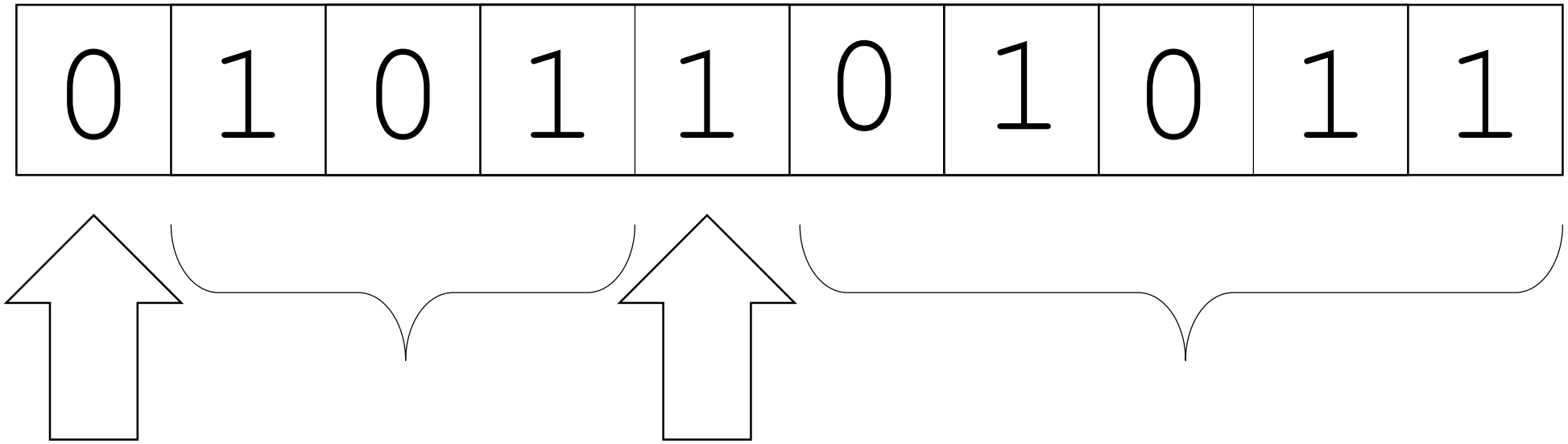
1: Convert .bdif images to standard images

> [user input]

- Each option leads to an input image and what to save the image as

Compression: Overview

- Image is flattened to 1D array of bits (B or W)
- Compressions follows general form:



Compression: Set up

- First, a BitWriter is created and the image is flattened
- The run lengths are then calculated

```
first=flat[0]
current=first
count=0
run_lengths=[]
for num in flat:
    if num==current:
        count+=1
    else:
       run_lengths.append(count)
       current=num
       count=1
if count:
    run_lengths.append(count)
```

Compression: Statistics

- To determine the number of bits to be used, number of bits required to store the 1st standard deviation of run lengths. Likewise, the number of bits to be used while the flag is true is calculated using the 2nd standard deviation

```
false_pow=1
while
(np.sum(run_lengths<(2**false_pow))<.6827*len(run_le
ngths)):
    false_pow+=1
    if false_pow==7:
        break
    false_thresh=2**false_pow
```

Compression: Header

- Height: 16 bits
- Width: 16 bits
- Depth: 8 bits
- First color: 1 bit
- Flag=false bits: 3 bits
- Flag=true bits: 4 bits

Compression: Run Length Encoding

```
if num>=true_thresh+false_thresh:
    while num>=true_thresh+false_thresh:
        f.write(1,1)
        f.write(true_thresh-1,true_pow)
        f.write(0,1)
        f.write(0,false_pow)
        num-=(true_thresh+false_thresh-1)
if num<false_thresh:
    f.write(0,1)
    f.write(num,false_pow)
elif num<true_thresh+false_thresh:
    f.write(1,1)
    f.write(num-false_thresh,true_pow)
```

Decompression

- Frist, the header is read and stored to determine the shape of the image and how to interpret the following bits
- A 1D array of pixels is created. Based on the number of pixels in a row represented by the number encoded

Compression: Code

```
nums=[]
    while len(nums)<h*w*d:
      flag=f.read(1)
      if flag:
          num=f.read(true_pow)+false_thresh
      else:
          num=f.read(false_pow)
      if current:
          nums+=list(np.ones((num)))
      else:
          nums+=list(np.zeros((num)))
      current=not current
```

Results

XKCD #976

- Monochrome Bitmap: 30 KB
- .bdif: 14 KB
- .png: 11 KB