Data compression

- You've used it
  - .zip for your lab exercises
  - sound formats, .au, .mp3 etc
  - graphics .jpg etc
  - etc.
- Many Meg stored in Fewer Meg
  so original data can be recovered
- How is it done?
  - We will look at general principles

Based on Repetition

- If there is no pattern of any kind in your data, equivalently
- If your data is entirely random
- Then there is no hope of compressing it.
  - Read about Claude Shannon and information entropy.

A compression algorithm

- will typically search through a file (uncompressed text, uncompressed graphics, uncompressed music, etc)
  - to find frequently occurring patterns
- It will then devise an encoding scheme to represent each frequently occurring pattern with codewords
  - we hope the codewords are more compact than the original pattern

Example

- T'amo spaghettini! T'amo linguini! T'amo tortellini! Ah, t'amo, t'amo, t'amo!
- Why not "t" for "T'amo" leading to
- t spaghettini! t linguini! t tortellini! Ah, t'amo, t'amo, t'amo!
  - maybe you can do better???
  - how about decompression?

Decompression

- t spaghettini! t linguini! t tortellini! Ah, t'amo, t'amo, t'amo!
  - using "T'amo" for "t"
- T'amo spagheT'amoT'amoini! T'amo linguini! T'amo T'amoorT'amoellini! Ah, T'amo'amo, T'amo'amo, T'amo'amo!
- Oops!

w or x or k!

- would be better
  - since none of these is in the Italian alphabet

Compression

- T'amo spaghettini! T'amo linguini! T'amo tortellini! Ah, t'amo, t'amo, t'amo
- using "w" for "T'amo"
- w spaghettini! w linguini! w tortellini! Ah, t'amo, t'amo, t'amo!

Compression

- T'amo spaghettini! T'amo linguini! T'amo tortellini! Ah, t'amo, t'amo, t'amo!
- using "w" for "'amo"
- Tw spaghettini! Tw linguini! Tw tortellini! Ah, tw, tw, tw!

Which is better?

- using "w" for "T'amo"
- w spaghettini! w linguini! w tortellini! Ah, t'amo, t'amo, t'amo!
- using "w" for "'amo"
- Tw spaghettini! Tw linguini! Tw tortellini! Ah, tw, tw, tw!

Different algorithms

- will compress differently
  - What's best for one file
- may not be best for another Figuring out the absolute best
  - is computationally prohibitive
- Seek a simple algorithm with decent compression and unique decoding

Side note

- In the interests of full disclosure
- Some compression schemes are lossy
- This means that you cannot recover all the information that was in the original from the compressed file
- Maybe ok for rock music or emailed photos
  - but we are only looking at lossless schemes

Basic principles of Lossless compression

- Fact
  - any compressible data file has some degree of repetition
- Use shorter encoding for repeated fragments

Fixed length Code

- ASCII
- American Standard Code for Information Interchange
  - each keyboard character is 7 bits
    - despite a byte having 8 bits
- ‘a’ is 1100001, ‘b’ is 1100010, ‘c’ is 1100011, ‘d’ is 1100100, ..., ‘x’ is 1111000, ‘y’ is 1111001, ‘z’ is 1111010

Unicode

- is similar uses 16 bits
- What do you know?
- How do you find out more?

Advantages of fixed-length codes

- always know where one character ends and another starts
- file size is easily related to number of characters
- easier code for transformations like,
  - rotation of image
  - transposition of music
- processing, in general

Disadvantages of fixed-length codes

- every item, no matter how frequently it occurs, uses the same “bandwidth”
- That’s why Morse code, in a time when transmission was expensive, uses short sequences for common letters
  - Eg. e = dot, t = dash

Is Morse Code binary?

- We all know ...---... is “SOS”
- But it could also be
- “EEETTTEEE”
- or “EIOIE”
- or “VGI”

Truth is

- Morse code is not binary
- In addition to . and - there is also the gap (silence)
- Morse code operators would separate individual characters with a silent gap
- SOS is not ...---... It is ...<gap>---<gap>...

Prefix codes

- A prefix code has the property that no codeword is a prefix of any other codeword
- Morse is not a prefix code
  - .. (I) is a prefix of ... (S)
  - So when you read the third dot
    - you don't know if
    - you're reading an S, or
    - you've finished reading an E and you're starting on the next character

Which is a prefix code?

Word	Code1	Code2	code3
Me	OI	OOO	OIO
You	IO	OI	IO
jane	I	OOI	OI
tarzan	O	I	III

Code2 is a prefix code?

Word	Code1	Code2	code3
Me	OI	OOO	OIO
You	IO	OI	IO
jane	I	OOI	OI
tarzan	O	I	III

Huffman Codes

- binary tree
- going to left
  - is a O
- going to right
  - is a1
- Decode
  - OOOOIOOII
  - OIOOIOOOI
  - IIO

Huffman Trees

- characters occur only at leaves
- Notice the importance of being a prefix code
  - to guarantee this
- To decode using a Huffman tree

```
repeat {
  go to root;
  while (not leaf) {
    eat a char;
    if it is 0
      go left;
    else go right;
  }
output leaf label;
}
```

Build a Huffman Tree

- for Hamlet's soliloquy
  - converted to lowercase
  - punctuation removed file
  - hamletsoliloquy.txt
- count the letters

SP 241	a 89	b 17	c 23	d 43	e 146	f 36	g 14	h 79
i 57	j0	k 10	l 44	m 32	n 70	o 99	p 25	q2
r 71	s 89	t 120	u 41	v8	w 29	x0	y 18	z2

Remove

- any letters that are used 0 time
  - no need to encode them

SP 241	a 89	b 17	c 23	d 43	e 146	f 36	g 14	h 79
i 57		k 10	l 44	m 32	n 70	o 99	p 25	q2
r 71	s 89	t 120	u 41	v8	w 29		y 18	z2

Treat the structure as a heap

- Repeatedly remove two lightest
- Combine
- insert the combination

SP 241	a 89	b 17	c 23	d 43	e 146	f 36	g 14	h 79
i 57		k 10	l 44	m 32	n 70	o 99	p 25	q2
r 71	s 89	t 120	u 41	v8	w 29		y 18	z2

Step 1

- Repeatedly remove two lightest
- Combine
- insert the combination

SP 241	a 89	b 17	c 23	d 43	e 146	f 36	g 14	h 79
i 57		k 10	l 44	m 32	n 70	o 99	p 25	qz4
r 71	s 89	t 120	u 41	v8	w 29		y 18	