Genome 540 Discussion

Conor Camplisson

January 10th, 2023

Genome Sciences
UNIVERSITY OF WASHINGTON

Outline

- Related topics
  - Information content
  - Compression
  - Burrows-Wheeler transform
- Homework #1 questions

Outline

- Related topics
  - Information content
  - Compression
  - Burrows-Wheeler transform
- Homework #1 questions

Information Content

file	size	compressed
dog1.txt	732	54
dog2.txt	199	173

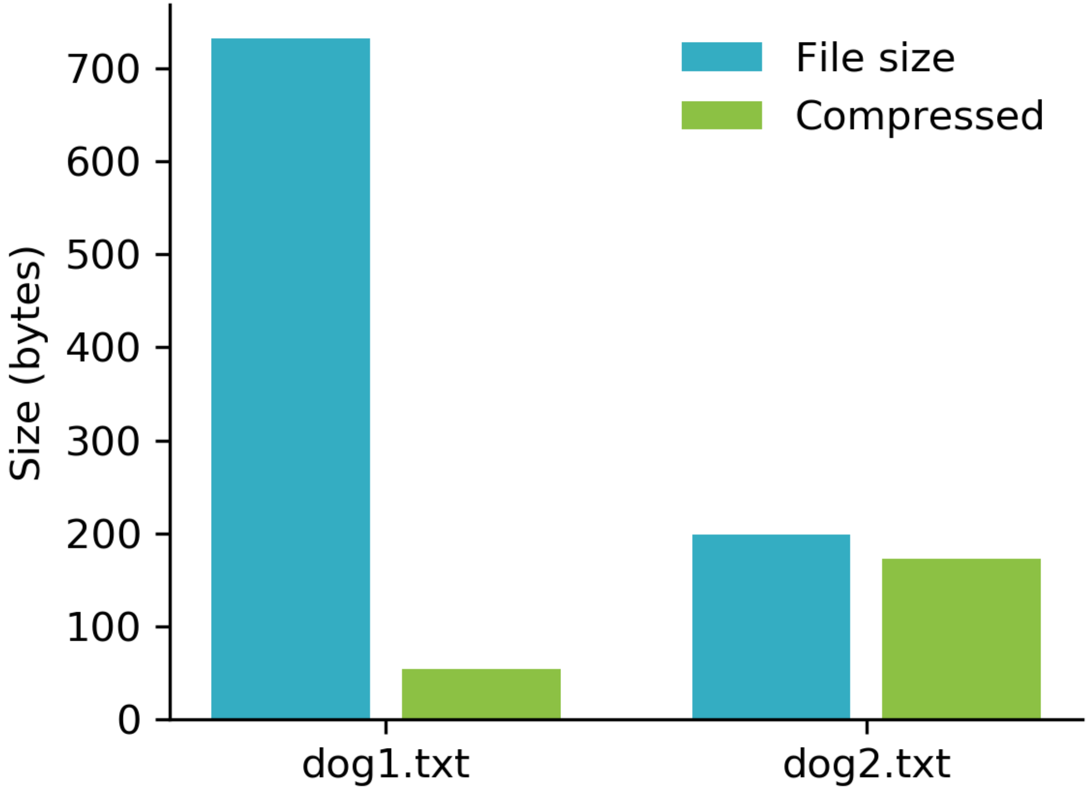
```
In [61]: print_contents(file_1)
```

```
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG DOG
```

```
In [62]: print_contents(file_2)
```

```
Lassie is a fictional character created by Eric Knight.

She is a female Rough Collie dog, and is featured in a short story
that was later expanded to a full-length novel called Lassie Come-Home.
```



Information Theory

sending information over a noisy channel

Information Theory

sending information over a noisy channel

Harry Nyquist

Certain Topics in Telegraph Transmission Theory

H. NYQUIST, MEMBER, A. I. E. E.

Classic Paper

Bell Laboratories

Communication in the Presence of Noise*

CLAUDE E. SHANNON†, MEMBER, IRE

***Summary*—A method is developed for representing any communication system geometrically. Messages and the corresponding signals are points in two "function spaces," and the modulation process is a mapping of one space into the other. Using this representation, a number of results in communication theory are deduced concerning expansion and compression of bandwidth and the threshold effect. Formulas are found for the maximum rate of transmission of binary digits over a system when the signal is perturbed by various types of noise. Some of the properties of "ideal" systems which transmit at this maximum rate are discussed. The equivalent number of binary digits per second for certain information sources is calculated.**

* Decimal classification: 621.38. Original manuscript received by the Institute, July 23, 1940. Presented, 1948 IRE National Convention, New York, N. Y., March 24, 1948; and IRE New York Section, New York, N. Y., November 12, 1947.

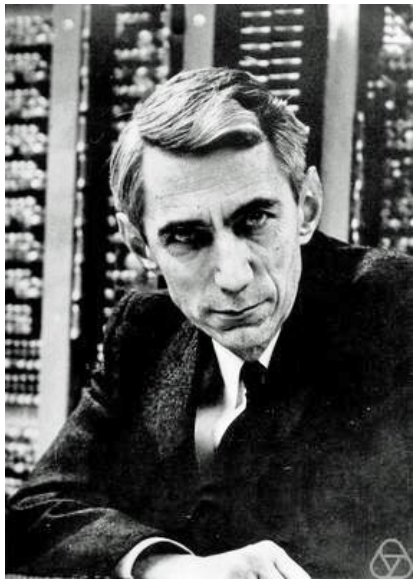
† Bell Telephone Laboratories, Murray Hill, N. J.

I. Introduction

A GENERAL COMMUNICATIONS system is shown schematically in Fig. 1. It consists essentially of five elements.

1. *An information source.* The source selects one message from a set of possible messages to be transmitted to the receiving terminal. The message may be of various types; for example, a sequence of letters or numbers, as in telegraphy or teletype, or a continuous function of time $f(t)$, as in radio or telephony.

2. *The transmitter.* This operates on the message in some way and produces a signal suitable for transmission to the receiving point over the channel. In teleph-



Claude Shannon

Data compression

- Compression – pressing something into a smaller space, increasing its density
- Data compression – storing information (or a close approximation) with a higher information density, occupying less space
- Compression algorithms
  - Lossy vs. lossless
  - Tradeoffs, best algorithm depends on situation
- Applications to big data: images, audio, videos, machine learning, genetics

https://en.wikipedia.org/wiki/Data_compression

Data compression - repetition

- Degree of compression depends on the raw data and algorithm used
  - In general, repetitive data is more compressible

```
raw:          AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
compressed:   30A
```

```
raw:          AAAAACCCCCCGGGGGTTTTTAAAAAAAAA
compressed:   5A6C5G5T9A
```

```
raw:          ACGTGCTAGTACGTCTATGTGCAGTACAGT
compressed:   1A1C1G1T1G1C1T1A1G1T1A1C1G1T1C1T1A1T1G1T1G1C1A1G1T1A1C1A1G1T
```

Burrows-wheeler transform

- Results in same-character runs (repetitive)
- Reversible (useful in lossless compression)
- Efficient suffix-array based implementation

Transformation				
1. Input	2. All rotations	3. Sort into lexical order	4. Take the last column	5. Output
^BANANA|	^BANANA| |^BANANA A|^BANAN NA|^BANA ANA|^BAN NANA|^BA ANANA|^B BANANA|^	**A**NANA|^B **AN**A|^BAN **A**|^BANAN **B**ANANA|^ **N**ANA|^BA **NA**|^BANA ^BANANA| |^BANANA	ANANA|^**B** ANA|^BA**N** A|^BANA**N** BANANA|^ NANA|^B**A** NA|^BAN**A** ^BANANA| |^BANAN**A**	BNN^AA|A

Applications

- Sequence alignment
  - BWA = Burrows-Wheeler Alignment tool
  - Bowtie uses Burrows-Wheeler indexing
- Compression of genomic databases
- Sequence prediction
- Image compression

https://en.wikipedia.org/wiki/Burrows%E2%80%93Wheeler_transform

Outline

- Related topics
  - Information content
  - Compression
  - Burrows-Wheeler transform

- Homework #1 questions

HW1 input FASTA files

- Assume each file has exactly one header line
  - The header line begins with “>”

- Non-alphabetic characters
  - Exclude the header line
  - Exclude white space (e.g. spaces)
  - Include only digits from seq position numbers

- Convert nucleotide sequences to upper case

IUPAC Nucleotide ambiguity code

IUPAC Code	Meaning	Complement
A	A	T
C	C	G
G	G	C
T/U	T	A
M	A or C	K
R	A or G	Y
W	A or T	W
S	C or G	S
Y	C or T	R
K	G or T	M
V	A or C or G	B
H	A or C or T	D
D	A or G or T	H
B	C or G or T	V
N	G or A or T or C	N

- Useful in general
- Suffix array approach
  - Exact matches only
- In HW1
  - Shouldn't come up

https://droog.gs.washington.edu/parc/images/iupac.html

Structs & pointers

- For HW1 can use integers or explicit pointers
- Struct members can be values or pointers

```
typedef struct
{
    int iLen;
    char *pcName;
} Info;
```

The above structure “Info” contains two members, integer variable (iLen) and a pointer to the character (pcName).

- More info and examples

https://aticleworld.com/pointers-as-member-of-structure-in-c/

“Number of match strings”

HW1 Example Template

```
The longest match length: 122
Number of match strings: 1

Match string:
GTCGGGTAAATTCCGTCCCGCTTGAATGGTGTAACCATCTCTTGACTGTCTCGGCTATAGACTCG
GTGAAATCCAGGTACGGGTGAAGACACCCGTTAGGCGCAACGGGACGGAAAGACCCC
Description: This sequence comes from [look up entry in .gbff
annotation file using the position information below]

Fasta: CP001872.fna
Position: 338240
Strand: forward

Fasta: CP001872.fna
Position: 82469
Strand: forward
```

- If there are several different perfectly repeated subsequences of the same maximum length, find all of them.
- If a longest subsequence is present multiple times in either sequence, please report *all locations* of the subsequence.

Edge cases

```
>test genome 1
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

>test genome 2
TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT

>test genome 3
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

- If run on 1 & 2
  - Program runs as usual
    - searches both strands!
- If run on 1 & 3
  - No match!

Don’t need to handle no-match edge case for HW1

Good to think about edge cases while programming!

Idea: tracking match length during sort (?)

Idea:

To speed up the algorithm by computing the longest matching string during the sort (since you would be checking the nearest neighbors during the sort anyways)

Why it likely slows down, rather than speeds up – low-level implementation details:

“Yes, you'd save some time at the end step (passing through the sorted list), but at the cost of extra time per comparison that almost certainly would outweigh it. As I mentioned in the lecture, cache misses (memory accesses outside of cache) are actually the main time cost in the suffix array method. In your suggested method you need to save information regarding the best match lengths seen thus far for each suffix which takes a fair amount of memory (8N bytes or so, if N is the sequence length), and accessing that will cause a lot of cache misses, potentially one for each comparison. That's potentially NlogN extra cache misses (in addition to the ones for accessing the sequence). You would save some cache misses at the final step (passing through the sorted list), because you'd no longer have to retrieve the sequences to know how long their match to the neighbor is, since you've saved that. But that's only N cache misses. So you're potentially adding NlogN cache misses to save potentially N cache misses, a net loss.” -Phil

. . .