Information

What is Information?

Depending on who you ask this question, the answer you receive can vary in accuracy and formality.

Thus, it's productive to introduce a formal definition which can be referred to when discussing various properties of information

Computer Scientist:

"Binary, 1's and 0's!"

Physicist:

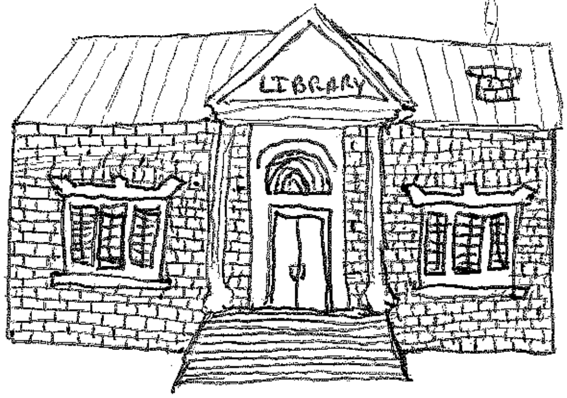
"The state of a particle!"

Your Younger Sibling:

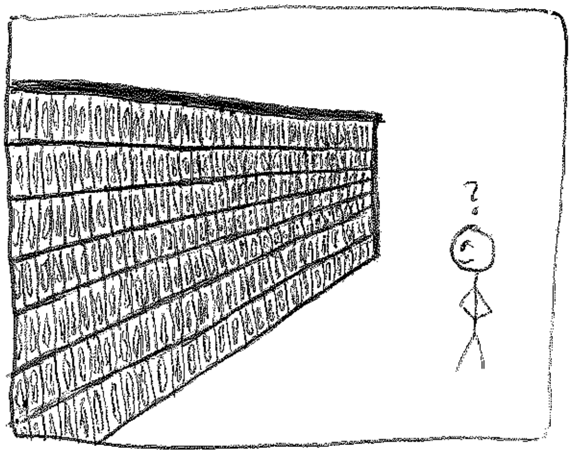
"I think it has to do with numbers and stuff?"

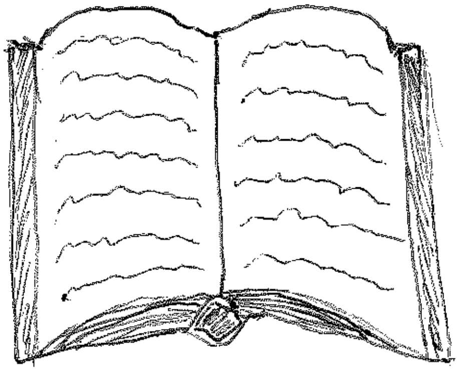
The formal definition can be understood intuitively by starting with a simple example.

Let's say one day you decide to venture to your local library

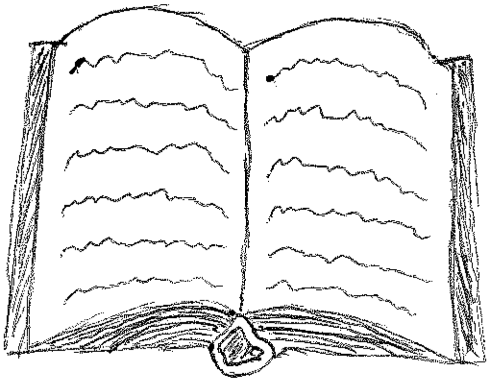


You are feeling adventurous, So you go to the latin Section and select two books at random. You open both books to their initial page, and see what is written.



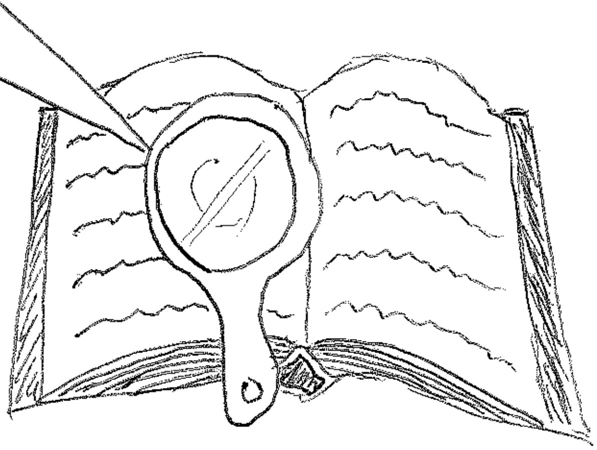


The first book only has only 2 words, which are repeated in an alternating pattern for the entire page. You don't Understand the words, but you figure there must not be anything important.

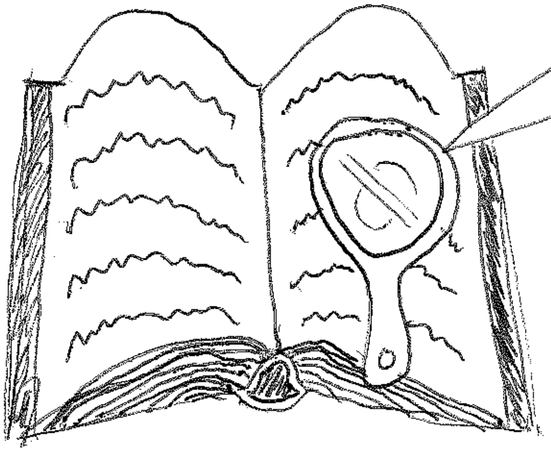


The second book has more than 2 words. Sometimes words repeat, but their is no real pattern to their appearance. You conclude that this book must be more important than the first.

Using this example, we define the quantity of information an object holds as the smallest representation that can be used to completely reconstruct the object.



In the first book you read, the sentences were composed of 2 words repeated. If you wanted to describe to someone else what was on the page, all you would have to do is write down both of the words, and tell them these two words were repeated until they filled the page. Since it is so easy to describe this page with a small representation, this book can be said to have a low quantity of information.



On the other hand, when evaluating the second book, the task of distilling the page summary to your friend suddenly becomes much harder. If none of the words are the same, you would essentially have to write down the entire page in order to represent the information. Thus, this book can be said to have a high quantity of information

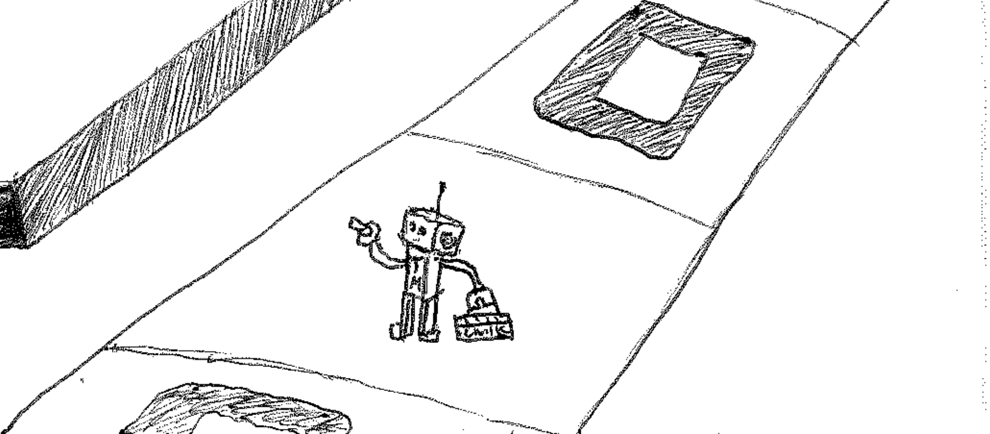
The issue here is that Latin is not the "Universal Language" of information. Early pioneers in information theory found out that having a language with more than 2 alphabet symbols will not increase the information that can be represented by the language. For this reason, instead of looking at Latin books, we now look at binary strings.

Unum ⇒ 01010101011011100111010101101101

We use binary strings because we're computer scientists, and it also happens to be the case that binary strings can be used to represent any other object.

Each Latin word in our selected book can be converted into a binary representation using the ASCII code. This binary representation will be longer than the Latin words in the book, but that's a sacrifice we're willing to make in order to construct our general definition.

To construct our definition, we take advantage of a Turing Machine. It is the Turing Machine's job to take a ~~non~~-important / arbitrary binary string as input, perform a computation, and output the binary representation of the desired object.



With all the necessary pieces in place, the formal definition is as follows:

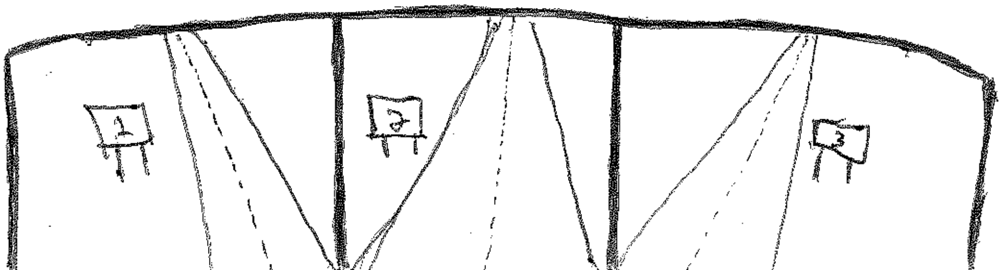
Let x be a binary string. The <u>minimal description</u> of x, written d(x), is the shortest string <M,w> where TM M on input w halts with x on its tape. If several such strings exist, select the lexicographically first among them. The <u>descriptive complexity</u> of x, written K(x), is:

$$K(x) = |d(x)|$$

This formal definition is very powerful in that it allows for objects represented by binary strings to be quantified in terms of how much information they possess.

Now, we can explore more fundamental properties of information by using the definition as a base. We will branch off into 3 areas,

- Properties of descriptive Complexity
- Optimality of descriptive Complexity
- Incompressible Strings and Randomness



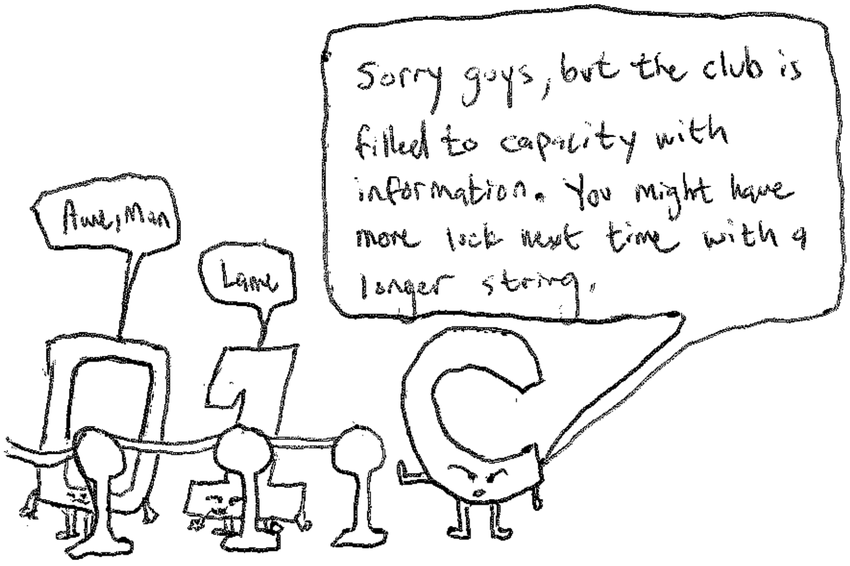
For our first area, we will explore some properties behind descriptive complexity.

These properties will be stated, but for the sake of space and sanity, the proofs will not be explored. This is reasonable though because the proofs are rather trivial. The important aspects of the properties appear in what they can be applied to.

#1 The first property of descriptive complexiby states that the descriptive complexity of any string is at most a fixed constant more than its length. This constant is universal, therefore it does not depend on the string.

So what does this property imply?

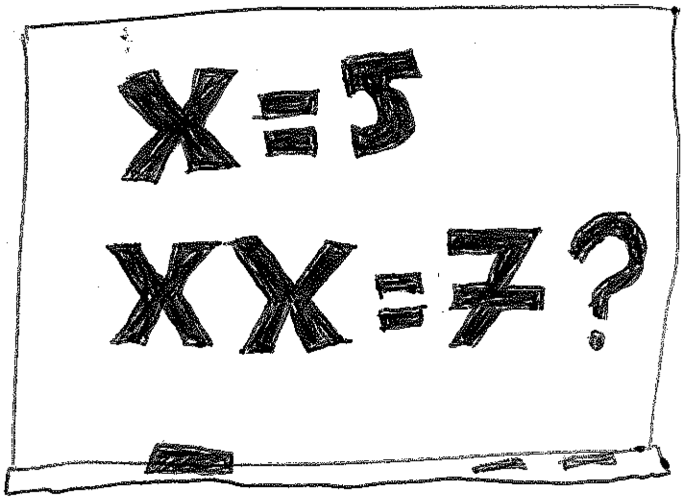
It simply means that the amount of information contained in some string can't be much more than the size of the actual string.


Awe, Man
Lame
Sorry guys, but the club is filled to capacity with information. You might have more luck next time with a longer string.

The next property we describe is closely related to the first. It states:

#2 The descriptive complexity of a string XX is at most a fixed constant more than the descriptive complexity of x

This means that the extra information created by copying the string X and concatinating them together will only be greater than the original information in X by a constant Value



The final property is similar to the second property, but leads to a few surprising conclusions

#3 The descriptive complexity of a string xy is at most a fixed constant more than twice the descriptive x plus the descriptive complexity of y.

This is strange at first, but more intuitive when you really think deeply about it. The concatination of two different binary strings X and y has a higher upperbound on the quantity of information it can contain in comparison to the concatination of X and X, In the next area, we will briefly explore the optimality of descriptive complexity.

This principle helps to illustrate that adding more "power" in the form of a better <u>description language</u> does not give significantly more information than the Turing Machine model which was previously defined.

- A description language is any computable function $p: \Sigma^* \to \Sigma^*$
  - The minimal description of x with respect to p, $(d_p(x))$ is the first string s where $p(s)=x$
  - $K_p(x) = |d_p(x)|$

I claim that: $\forall x [K(x) \leq K_p(x) + c]$

To prove this claim, we simply do the following

Take any description language p and consider the following Turing Machine M.

M = " On input w:
1. output $p(w)$"

Then $\langle M \rangle d_p(x)$ is a description of x whose length is at most a fixed constant greater than $K_p(x)$. This constant is the length of $\langle M \rangle$.

For our final area, we will discuss some aspects of Incompressiable Strings and Randomness

We start out be asking the question

"Do some strings lack short descriptions?"

The answer, as hinted at in the properties of descriptive complexity is:

There do exist strings that can't be described any more concisely than writing them out explicitly

Yes ☑

No □

More formally, we:

Let x be a string. Say that x is <u>c-compressible</u> if

$$K(x) \leq |x| - c$$

If x is not c-compressible, we say that x is <u>incompressible by c</u>

If x is incompressible by 1, we say that x is <u>incompressible</u>

Now that we have a more formal definition of what it means to be incompressible, we will explore two interesting properties of incompressibility.

The first interesting property is that incompressible strings of every length exist.

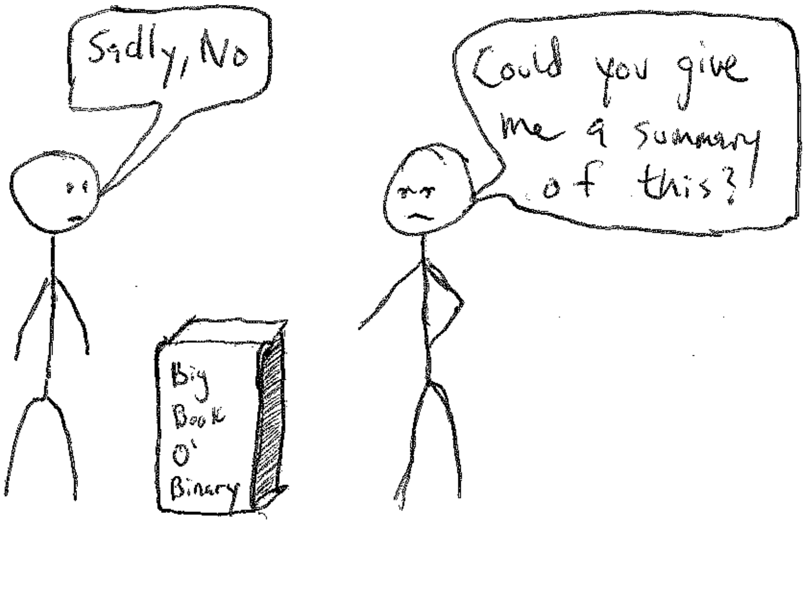
The underlying idea is that the number of strings of length n is greater than the number of descriptions of length less than n. Since each description describes at most one string, some string of length n is not described by any description of length less than n.

Proof:

The number of binary strings of length n is 2^n. Each description is a binary string, so the number of descriptions of length less than n is at most $2^n - 1$. (Sum of the number of strings of each length up to n-1)

The number of short descriptions is less than the number of strings of length n. Therefore, at least one string of length n is incompressible

This could end up being problematic if the size of n is large, and it happens to be that the binary string you are trying to describe is incompressible



The second interesting property draws from previously defined properties of descriptive complexity, and states that there is no way of obtaining long incompressible strings, and that there is no way to determine if a string is incompressible,

This property comes from the fact that

- The K measure of complexity is not computable
- No algorithm can decide in general whether strings are incompressible
- No infinite subset of them is turing recognizable

The closest we can get is ~~to~~ describe certain strings that are nearly incompressible, We claim:

For some constant b, for every string x, the minimal description d(x) of x is incompressible by b.

Proof:

Consider the following TM M:

```
M = "On input $\langle R, y \rangle$ where R is a TM and y is a string:
    1. Run R on y and reject if its output is not of
       the form $\langle S, z \rangle$
    2. Run S on z and halt with its output on the tape."
```

Let b be $|\langle M \rangle| + 1$. We show b satisfies the theorem, Suppose to the contrary that d(x) is b-compressible for some string x. Then

$$|d(d(x))| \le |d(x)| - b$$

But then $\langle M \rangle d(d(x))$ is a description of x whose length is at most

$$|\langle M \rangle| + |d(d(x))| \le (b-1) + (|d(x)| - b) = |d(x)| - 1$$

This description of x is shorter than d(x), contradicting the latter's minimality.

We now move away from the dense proof and go to a more high level view of the concept. The astounding reality is that there is guarenteed to be an incompressible string of length n, but even if you had the string you wouldn't be able to prove the string you have is actually incompressible.

This is bad news for college students everywhere, because now they will know it is impossible to determine if their final exam material is actually incompressible, or if they are just doing extra work.

Fin