Foundations of Computer Science

00 - The Bit

TABLE OF CONTENTS

- What is a bit?
- Why do we need a binary system?
- What is the meaning of a bit?
- Conclusion

WHAT IS A BIT?

All data you store in a computer, eventually, is stored as a sequence of 0s and 1s.

This is where the idea of a **bit** comes in.

In its most simple description,

A bit is a slot where you can put either a 0 or a 1

It does not matter where

- a piece of paper
- an address within a computer's memory

Anytime we can store either a 0 or a 1, we have a bit.

A bit therefore is a **place-holder for binary digits**.

We can also call this a **binary-slot**.

WHY DO WE NEED A BINARY SYSTEM?

When we learn about **computer science** we're often told that everything within computers is memorized with bits, that is, with sequences of 0 and 1.

Modern computer architectures work with **bytes**.

$$1 \text{ byte } = 8 \text{ bits}$$

Computers store everything using 0 and 1.

Q: Why is that the case?

The basic idea is that before we can even start to work with a computer, we must be able to **represent different things** within the computer memory.

When we grow up we learn that in language such as english we represent different things through different combinations of letters.

We start with an alphabet

ABCDEFGHIJKLMNOPQRSTUVWXYZ

and we create combination of these symbols, which we call **words**, **phrases**, and **paragraphs**

HELLO WORLD

We then assign a **meaning** to these words, phrases and paragraphs in order to represent concepts taken from the world.

TREE →

In the language of computers, the same idea applies.

We need a **notational system** comprised of **different symbols** in order to represent concepts and ideas, and of **rules** for manipulating those symbols.

What's different in computers is that, instead of using the english alphabet, we use a **binary alphabet** comprised only of two different symbols

$$0 \ , \ 1$$

This is the basic idea behind the **binary system**

It is a "**system**", because it's comprised of different symbols and of rules for manipulating those symbols

---

It is "**binary**", because it contains only two symbols, the 0 and the 1.

Note that there's nothing special about the symbols 0 and 1 themselves, just the fact that they look different

$$0 \neq 1$$

(I will come back to this point in the future)

WHAT IS THE MEANING OF A BIT?

Let us now begin to understand how we can use such binary slots to represent concepts and information about the real world.

1 BIT

We begin by asking

How many things can we represent with just a single bit?

By definition, a single bit can contain only two values, which are 0 and 1. This means that with a single bit we can represent only two things

$$\begin{aligned} 0 &\longrightarrow \text{ a thing} \\ 1 &\longrightarrow \text{ another thing} \end{aligned}$$

Notice here I did not specify exactly what we were representing. For example, if we want to express emotions, we can define the following mapping

$$\begin{aligned} 0 &\longrightarrow \text{ sad} \\ 1 &\longrightarrow \text{ happy} \end{aligned}$$

If instead we want to represent the weather, we can use the following mapping

$$
\begin{aligned}
0 &\longrightarrow \ \text{sunny} \\
1 &\longrightarrow \ \text{not sunny}
\end{aligned}
$$

What to do if we want to make our representation more descriptive of the real world? The idea is simple:

To express more things, you need more bits

2 BITS

Once again, we ask ourselves

How many things can we represent with two bits?

With two bits the possibilities expand, because now we have to slots to fill, and each one can be filled either with a 0 or with a 1.

With two bits, we find the following possibilities

$$00$$

$$01$$

$$10$$

$$11$$

Using the previous examples, we're able to capture two more emotional states, and two more weathers.

$$\begin{array}{lllllll}
00 & \longrightarrow & \text{happy} & \qquad & 00 & \longrightarrow & \text{sunny} \\
01 & \longrightarrow & \text{fear} & \qquad & 01 & \longrightarrow & \text{stormy} \\
10 & \longrightarrow & \text{sadness} & \qquad & 10 & \longrightarrow & \text{cloudy} \\
11 & \longrightarrow & \text{surprise} & \qquad & 11 & \longrightarrow & \text{rainy}
\end{array}$$

3 BITS

How many things can we represent with three bits?

With three bits we can express up to eight different things.

000

001

010

011

100

101

110

111

Notice how, once again, going from two bits to three bits, we have doubled our expressive power.

4 BITS

How many things can we represent with four bits?

With four bits we're able to express up to sixteen different things. Once again, exactly doubled from the three bit case.

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

We can keep going and we can use five bits, and then six bits, and then seven bits and so on. Each time, we will double the number of things we can represent.

WHERE'S THE MEANING?

In all the previous examples, we did not define a single, specific meaning to the sequences of bits.

000

001

010

011

100

101

110

111

This is intended, as

bits, by themselves, do not have a specific a priori meaning

It is us, as human beings, as engineers, as computer scientists, as programmers, that assign meanings to those bits, depending on the domain we're interacting with.

Humans give meanings to bits

In this lack of meaning lies the power of abstraction inherent in digital computers and computer science.

By not defining a specific meaning to the bits, we're able to capture anything that can be captured by a digital computation

CONCLUSION

In future videos I will talk about how we can give meaning to these bits in order to capture:

1. The english alphabet

   ABCDEFGHIJKLMNOPQRSTUVWXYZ

2. Indo-Arabaic numerals

   0123456789

3. Colors

   RED *GREEN* *BLUE*