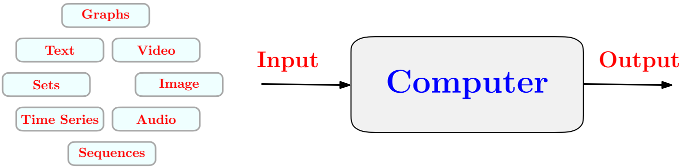
Computation, Encoding and Languages

- Computational Problems, Strings and Data Encoding
- Binary Encoding
- Language
- Versions of Computational Problems
- Decision Problems as Language Recognition
- Models of Computation – CPU + Memory

Imdad ullah Khan

Binary Encoding

Computation: Processing information by applying a finite set of rules



How do we represent/encode input and output?

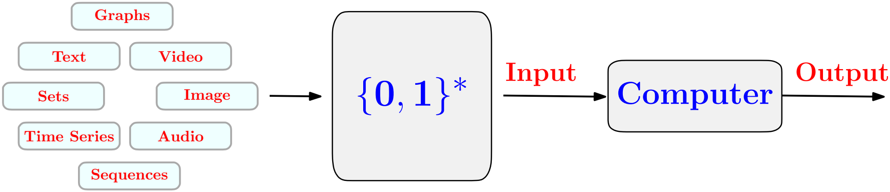
An encoding/representation scheme for a set of objects O is a one-to-one function $E : O \mapsto \{0, 1\}^*$

Encoding should be one-to-one for decoding

$$D : range(E) \mapsto O \quad s.t \quad D(E(x)) = x \quad \forall x \in O$$

Binary Encoding of Data

We give simple binary representation for common types of data

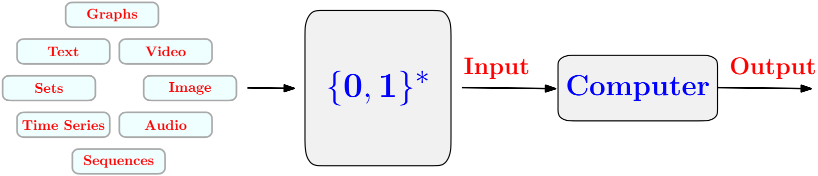


Can we represent "everything" with $\{0,1\}^*$?

"everything" ?

Binary Encoding of Data

Can we represent $\mathbb{N}$ with $\{0,1\}^*$?



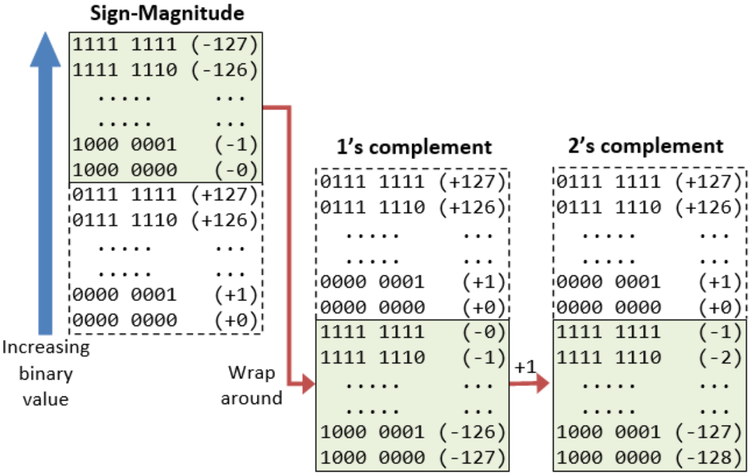
Does Σ matter?

Decimal	Binary	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

Does $|\Sigma|$ matter?

Decimal	Binary	Unary
0	0	ϵ
1	1	1
2	10	11
3	11	111
4	100	1111
5	101	11111
6	110	111111
7	111	1111111
8	1000	11111111
9	1001	111111111
10	1010	1111111111
11	1011	11111111111
12	1100	111111111111
13	1101	1111111111111
14	1110	11111111111111
15	1111	111111111111111
16	10000	1111111111111111

Can we represent $\mathbb{Z}$ with $\{0,1\}^*$?



Can we represent $\mathbb{N} \times \mathbb{N}$ with $\{0,1\}^*$?

Why? 2-d numbers, rational numbers, cell/pixel address of matrix/image

We have $E : \mathbb{N} \mapsto \{0,1\}^*$, we need encoding of pairs of (natural) numbers

Can we encode $x, y \in \mathbb{N} \times \mathbb{N}$ as $E(x)E(y)$? $\triangleright$ concatenation

Fixed Length Code

Fixed number of bits for each object (symbox)

- e.g. ASCII (7 bits) and Unicode (UTF-8, UTF-16)
- ASCII can represent $2^7 = 128$ symbols

Variable Length Code

Variable number of bits for each object

- Can use fewer bits for more frequent symbols
- e.g. Huffman code
- Difficult to find, needs compression scheme

Characters	a	b	c	d
Fixed-Length Code	00	01	10	11
Variable Length Code 1	0	10	110	111
Variable Length Code 2	0	1	01	10

- Let the string be **b a a d a b**
- Fixed Code: 01 00 00 11 00 01 $\rightarrow$ 12 bits
- Variable Code 1: 10 0 0 111 0 10 $\rightarrow$ 10 bits
- Variable Code 2: 1 0 0 10 0 1 $\rightarrow$ 7 bits

- Variable Code 2 compresses a lot
- Codes must be **uniquely** decodable
- Variable Code 2: 1001001 can be decoded as dabac or bacad or ...

Prefix free encoding : When no code is a prefix of another

If a code is prefix free, then it is uniquely decodable

Characters	a	b	c	d
Fixed-Length Code	00	01	10	11
Variable Length Code 1	0	10	110	111
Variable Length Code 2	0	1	01	10

Variable Length Code 2 is not prefix free

▷ Code for '*a*' (0) is a prefix of code for '*c*' (01)

Binary Encoding of Common Data Type

We give simple binary representation for common types of data

Can we represent $\mathbb{N} \times \mathbb{N}$ with $\{0,1\}^*$?

Why? 2-d numbers, rational numbers, cell/pixel address of matrix/image

We have $E : \mathbb{N} \mapsto \{0,1\}^*$, we need encoding of pairs of (natural) numbers

Can we encode $x, y \in \mathbb{N} \times \mathbb{N}$ as $E(x)E(y)$? ▷ concatenation

Theorem: If $E : O \mapsto \{0,1\}^*$ is prefix-free, then we can use it to encode $O \times O$ by concatenation ▷ also encode longer lists of objects in O

If $E : O \mapsto \{0,1\}^*$ is prefix free, then $E' : O \times O \mapsto \{0,1\}^*$ defined as $E'(xy) = E(x)E(y)$ is one-to-one

Is our earlier mapping $E : \mathbb{N} \mapsto \{0,1\}^*$ (decimal2Binary) prefix-free?

Theorem: Every encoding can be converted to a prefix-free one

We construct a prefix-free encoding $E : \{0,1\}^* \mapsto \{0,1\}^*$

Define $E_1 : \{0,1\}^* \mapsto \{0,1,\#\}^*$ as

for $x_1 \dots x_k \in \{0,1\}^*$ $\quad E_1(x_1 \dots x_k) = x_1 \dots x_k\#$

▷ $E_1(10101) = 10101\#, \quad E_1(011) = 011\#, \quad E_1(0111) = 0111\#$

Clearly, E_1 is prefix-free

Define $e_2 : \{0,1,\#\} \mapsto \{0,1\}^2$ as $\quad e_2(0) = 01, \quad e_2(1) = 10, \quad e_2(\#) = 11$

Clearly, e_2 is prefix-free

Define $E_2 : \{0,1,\#\}^* \mapsto \{0,1\}^*$ as

for $x_1 \dots x_m \in \{0,1,\#\}^*$ $\quad E_2(x_1 \dots x_m) = e_2(x_1) \dots e_2(x_m)$

▷ $E_1(101\#) = 10011011, \quad E_2(10\#) = 100111, \quad E_1(0\#1) = 011110$

Then $E : \{0,1\}^* \mapsto \{0,1\}^* = E_2 \circ E_1$ is prefix-free

Is our earlier mapping $E : \mathbb{N} \mapsto \{0,1\}^*$ (decimal2Binary) prefix-free?

Theorem: Every encoding can be converted to a prefix-free one

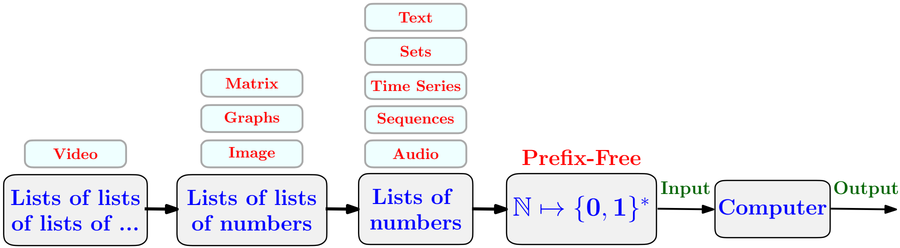
Let $E : \{0,1\}^* \mapsto \{0,1\}^*$ be the prefix-free code we constructed previously

Denote by $\textsc{d2b} : \mathbb{N} \mapsto \{0,1\}^*$ the standard decimal to binary encoding

$E' : \mathbb{N} \mapsto \{0,1\}^* = E \circ \textsc{d2b}$ is a prefix-free encoding

We give simple binary representation for common types of data

Can we represent “everything” with $\{0,1\}^*$?



Can we represent $\mathbb{R}$ with $\{0,1\}^*$?

Recall Cantor's **DIAGNOLIZATION** proof to show

The set $\mathbb{I}$ of real numbers between 0 and 1 is not countable

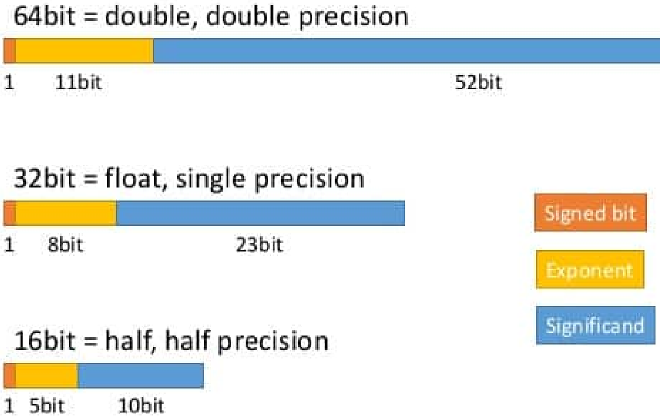
▷ See CS 210 slides and Textbook [Barak Theorem 2.5]

$\mathbb{R}$ cannot be represented with $\{0,1\}^*$

Binary Encoding of Common Data Type

Can we represent $\mathbb{R}$ with $\{0,1\}^*$?

Use single/double-precision floating point as approximate representation



IEEE 754 Floating Point Standards