HUFFMAN Coding And Data Compactation

Kowshic Roy (S201705001)
Mahdi Hasnat Siyam (S201705003)

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

July 14, 2021

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

Disk Shortage Problem

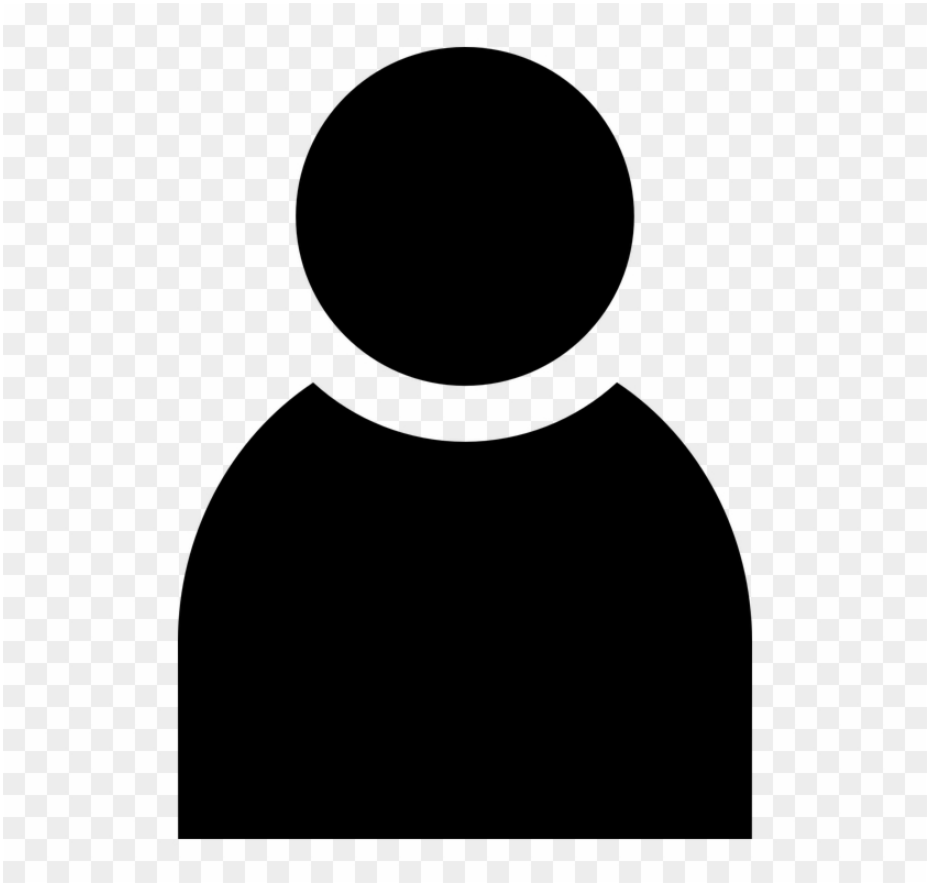
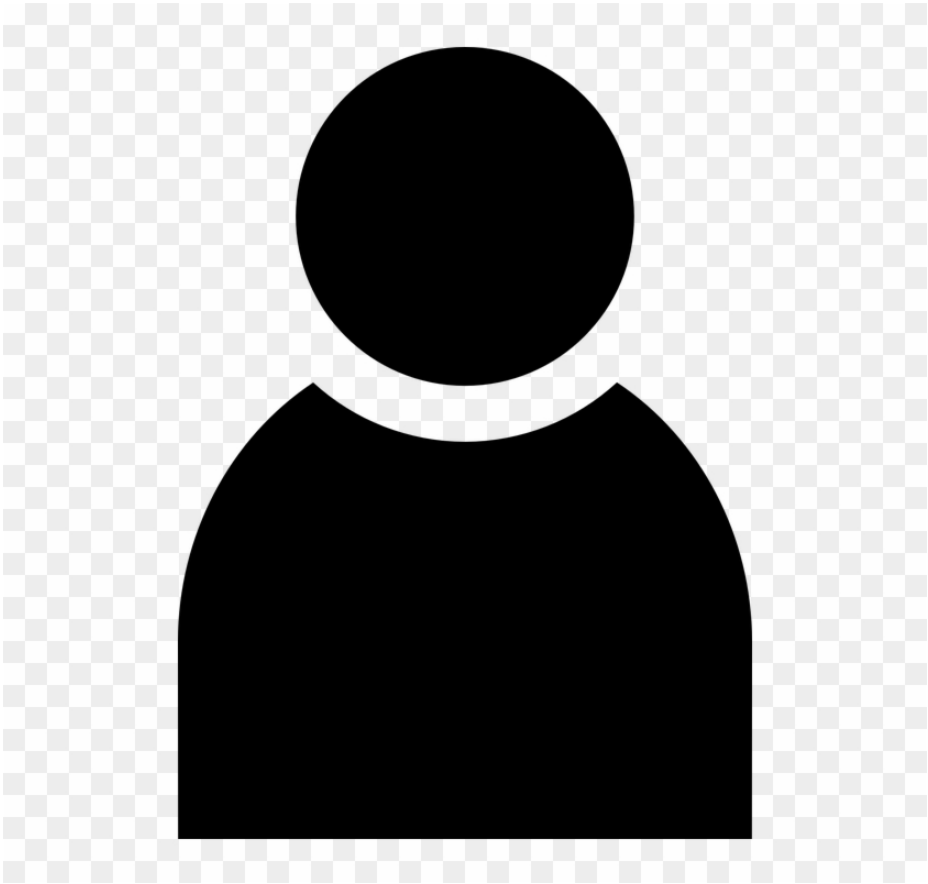


Figure: Robin, An NLP Enthusiast.

Disk Shortage Problem

Disk Shortage Problem



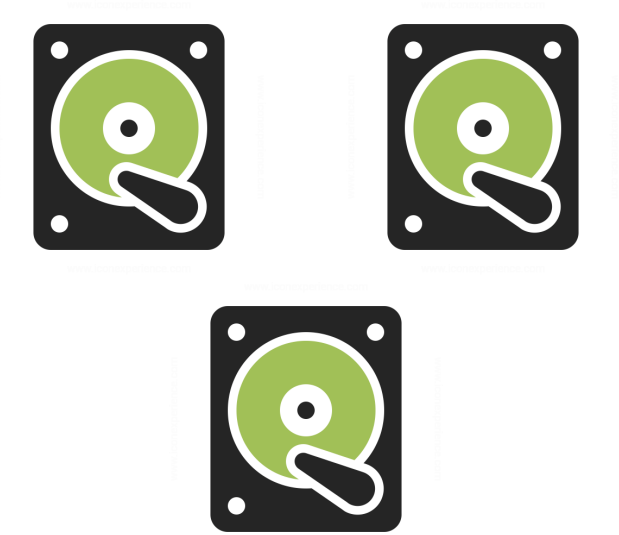
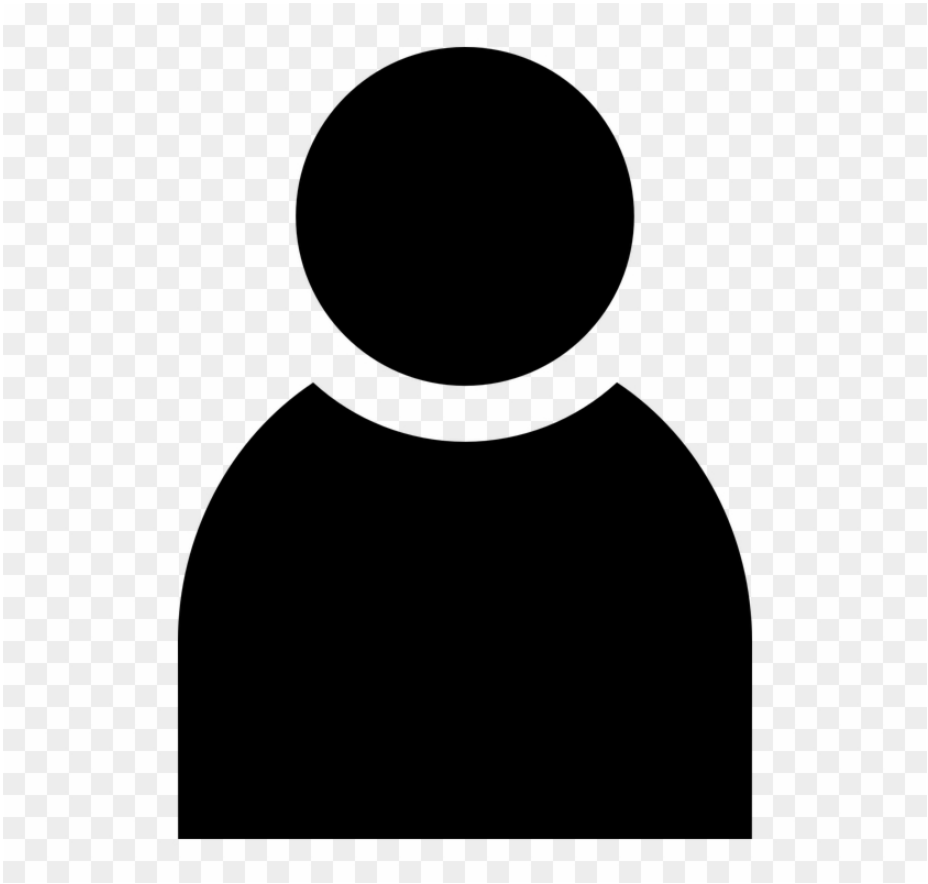


Figure: Robin, An NLP Enthusiast.

Disk Shortage Problem



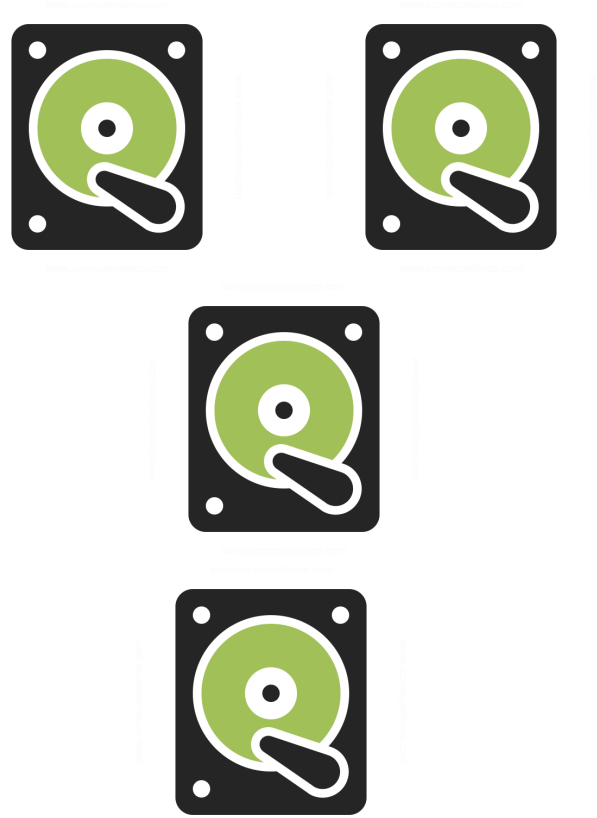
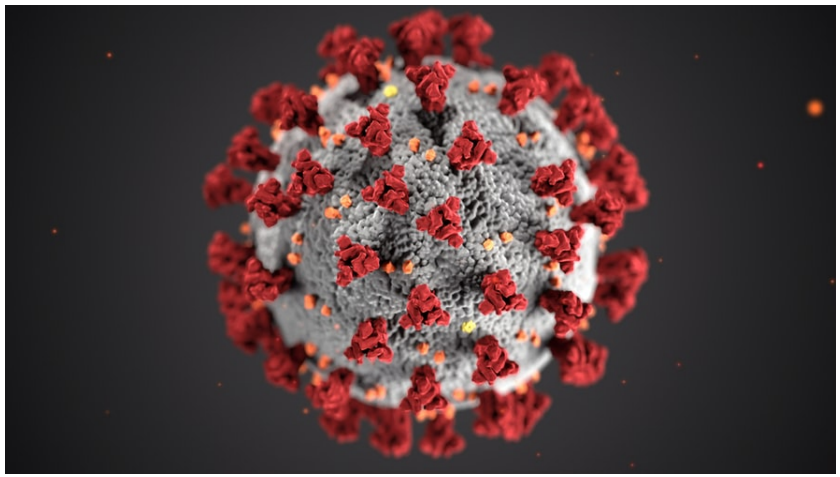


Figure: Robin, An NLP Enthusiast.

Disk Shortage Problem

Corona Pandemic

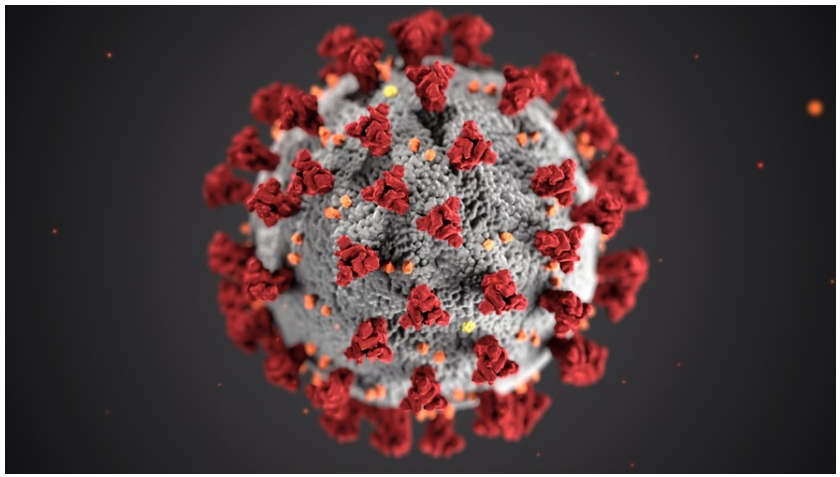
CORONA!!!



Disk Shortage Problem

Corona Pandemic

CORONA!!!



Beginning of Our Journey

Some Questions

- Can we store more data in the same disk space?

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding
A	0100 0001

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding
A	0100 0001
B	0100 0010

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding
A	0100 0001
B	0100 0010
C	0100 0011

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding
A	0100 0001
B	0100 0010
C	0100 0011
D	0100 0100

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding
A	0100 0001
B	0100 0010
C	0100 0011
D	0100 0100

So on...

How Data is Stored

Standard Encoding - ASCII

ASCII

Character	Encoding
A	0100 0001
B	0100 0010
C	0100 0011
D	0100 0100

So on...

Fixed length code

How Encoding Works

ASCII

How Encoding Works

ASCII

Main Text

JAVA

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001 0100 0001

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001 0100 0001

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001 0100 0001 0101 0110

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001 0100 0001 0101 0110

How Encoding Works

ASCII

Main Text

JAVA

Encoded Bits

0100 1001 0100 0001 0101 0110 0100 0001

How Decoding Works

ASCII

Encoded Bits

0100 10010100 00010101 01100100 0001

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001010101100100000 1

Decoded Text

How Decoding Works

ASCII

Encoded Bits

0100 1001|010000010101011001000001

Decoded Text

J

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001|0101011001000001

Decoded Text

J

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001|0101011001000001

Decoded Text

J A

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001|01010110|01000001

Decoded Text

J A

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001|01010110|01000001

Decoded Text

J A V

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001|01010110|01000001

Decoded Text

J A V

How Decoding Works

ASCII

Encoded Bits

0100 1001|01000001|01010110|01000001

Decoded Text

J A V A

Our Example String

AAABRACADABRAAA

AAABRACADABRAAA

Character	Frequency	Encoding
A	9	01000001
B	2	01000010
C	1	01000011
D	1	01000100
R	2	01010010

Total = 15

Our Example String

AAABRACADABRAAA

AAABRACADABRAAA

Character	Frequency	Encoding
A	9	01000001
B	2	01000010
C	1	01000011
D	1	01000100
R	2	01010010

Total = 15

Bits Needed

$15 \times 8 = 120$

Can we do better?

Observation - 1

Why always taking 8 bit?

Can we do better?

Observation - 1

Why always taking 8 bit?

New Encoding Scheme

unique character $= 5$

Can we do better?

Observation - 1

Why always taking 8 bit?

New Encoding Scheme

unique character $= 5$

$ceil(log_2(5)) = 3$

bits is enough to represent them uniquely.

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100
B	2	011

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100
B	2	011
C	1	010

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100
B	2	011
C	1	010
D	1	001

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100
B	2	011
C	1	010
D	1	001
R	2	000

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100
B	2	011
C	1	010
D	1	001
R	2	000

Total = 15

Our Own Encoding

Our first ever encoding table

Character	Frequency	Encoding
A	9	100
B	2	011
C	1	010
D	1	001
R	2	000

Total $= 15$

bits needed

$15 \times 3 = 45$

Our own encoding

Finding the hidden cost

- Do 45 bits are enough ?

Our own encoding

Finding the hidden cost

- Do 45 bits are enough ?
- Don't we have to save the table ?

Our own encoding

Finding the hidden cost

- Do 45 bits are enough ?
- Don't we have to save the table ?
- Bits needed for the encoding table can be safely ignored.

Takeaway - 1

Intelligent encoding requires less bits.

ASCII	Mod. F. L.
120	45

Table: Bits needed in different encoding

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

Can we do more better?

Character	Frequency	Encoding
A	9	100
B	2	0 11
C	1	0 10
D	1	00 1
R	2	00 0

Total = 15

Intuition - 1

Aren't leading zeroes redundant?

Variable length encoding

Character	Frequency	Encoding
A	9	100
B	2	11
C	1	10
D	1	1
R	2	0

Total = 15

Variable length encoding

Character	Frequency	Encoding
A	9	100
B	2	11
C	1	10
D	1	1
R	2	0

Total = 15

Trust me!

Variable length encoding

Character	Frequency	Encoding
A	9	100
B	2	11
C	1	10
D	1	1
R	2	0

Total = 15

Trust me!

bits needed

$3 \times 9 + 2 \times 2 + 2 \times 1 + 1 \times 1 + 1 \times 2 = 36$

Can we do better ?

Observation - 2

Why are we not using the frequency of the characters ?

Can we do better ?

Observation - 2

Why are we not using the frequency of the characters ?

Intuition - 2

Characters with highest frequency should have less number of bits

Frequency based variable length encoding

Character	Frequency	Encoding

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0
B	2	1

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0
B	2	1
R	2	10

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0
B	2	1
R	2	10
C	1	11

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0
B	2	1
R	2	10
C	1	11
D	1	100

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0
B	2	1
R	2	10
C	1	11
D	1	100

Total = 15

Frequency based variable length encoding

Character	Frequency	Encoding
A	9	0
B	2	1
R	2	10
C	1	11
D	1	100

Total = 15

Trust me!

bits needed

$1 \times 9 + 1 \times 2 + 2 \times 1 + 3 \times 1 + 2 \times 2 = 20$

Takeaway - 2

Takeaway - 2

Frequency based encoding is a good technique in reducing bits count.

ASCII	Mod. F.L.	V. L.(Rand.)	V. L.(Frequency)
120	45	36	20

Table: Bits needed in different encoding

But...

Is our encoding right?

Is our approach right ?

But...

Is our encoding right?

Is our approach right ?

Can it be decoded correctly?

But...

Is our encoding right?

Encoded bits

10

But...

Is our encoding right?

Encoded bits

10

Character	Encoding
A	0
B	1
R	10
C	11
D	100

But...

Is our encoding right?

Encoded bits

10

Decoded Text - 1

BA

Character	Encoding
A	0
B	1
R	10
C	11
D	100

But...

Is our encoding right?

Encoded bits

10

Decoded Text - 1

BA

Decoded Text - 2

R

Character	Encoding
A	0
B	1
R	10
C	11
D	100

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

What Went Wrong?

Is our encoding right?

Encoded bits

10

Decoded Text - 1

BA

Decoded Text - 2

R

Character	Encoding
A	0
B	1
R	10
C	11
D	100

1 is a prefix of 10

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

Prefix Properties

- No whole code word is prefix of any other code word

Prefix Properties

- No whole code word is prefix of any other code word

Character	Code
A	0
J	11
V	10

This encoding satisfy prefix property.

Prefix Properties

- No whole code word is prefix of any other code word

Character	Code
A	0
J	11
V	10

This encoding satisfy prefix property.

Character	Code
A	0
J	1
V	01

"0" is prefix of "01"
This encoding does not satisfy prefix property.

Table of Contents

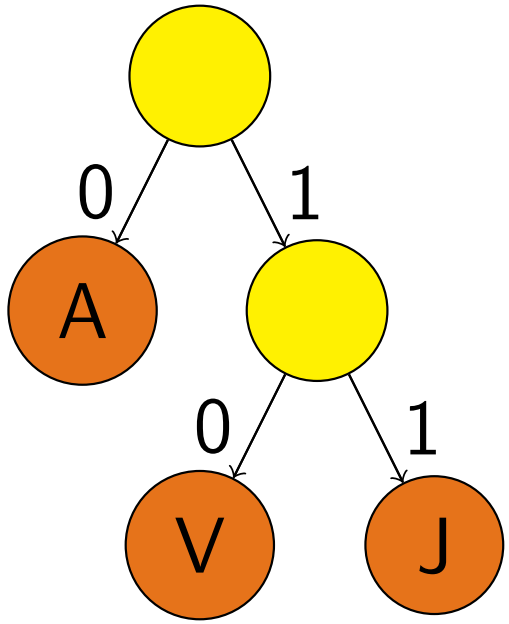
1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

Representation of encoding as Trie

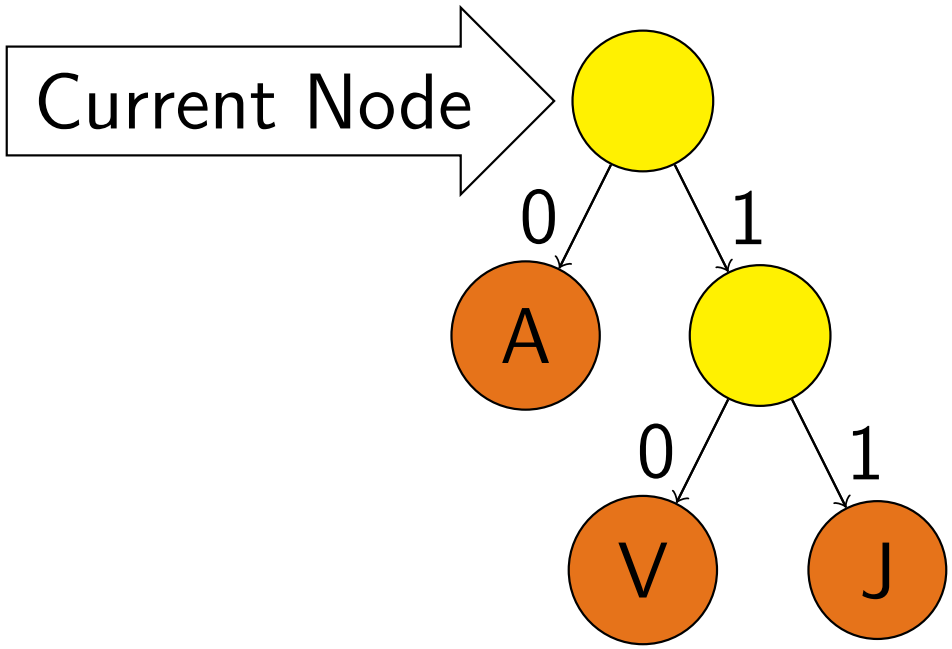
Character	Code
A	0
J	11
V	10

Representation of encoding as Trie

Character	Code
A	0
J	11
V	10

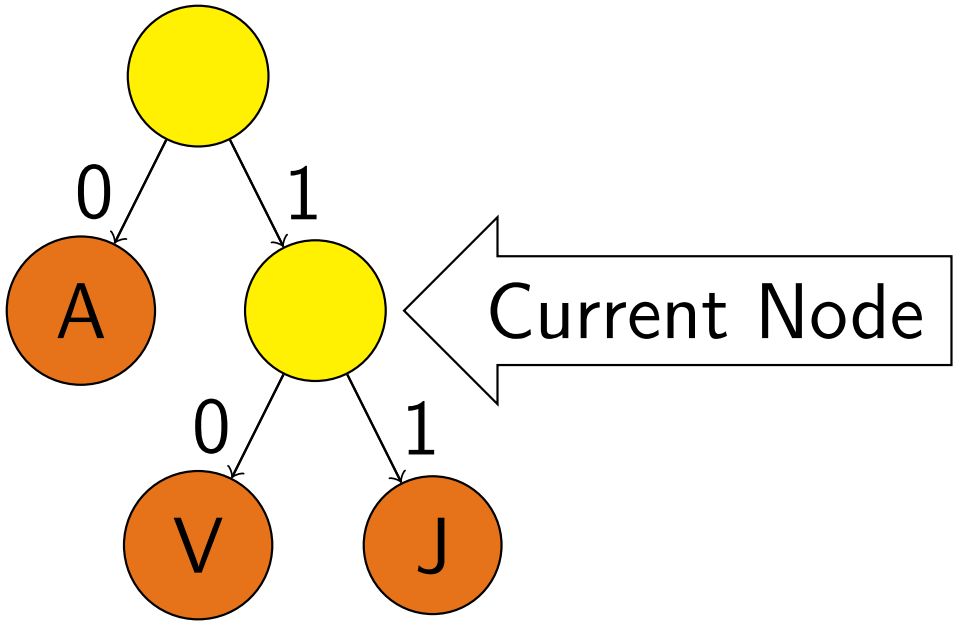


Decoding from trie



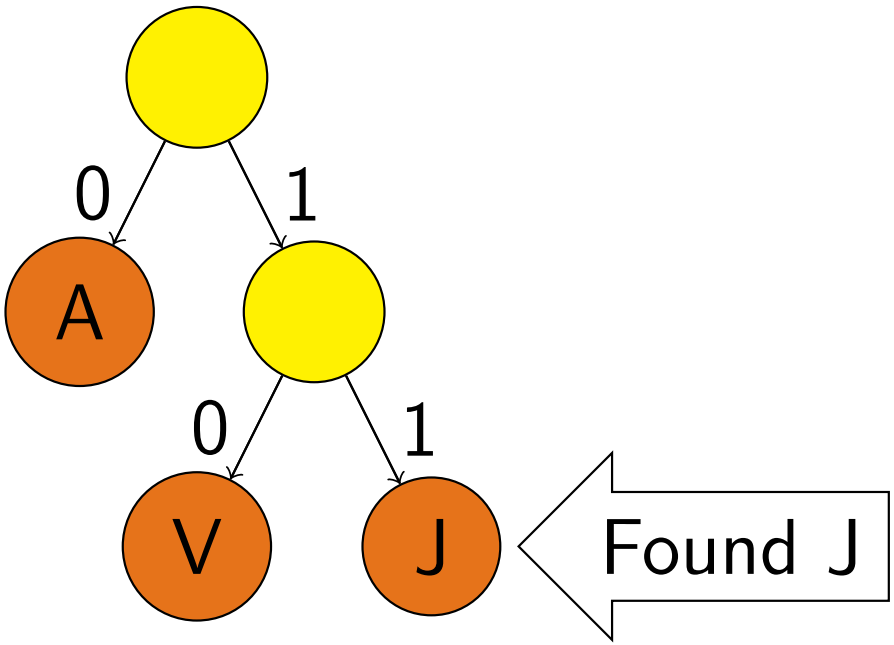
Encoded Bits : “|110100”
Decoded Text : “”

Decoding from trie



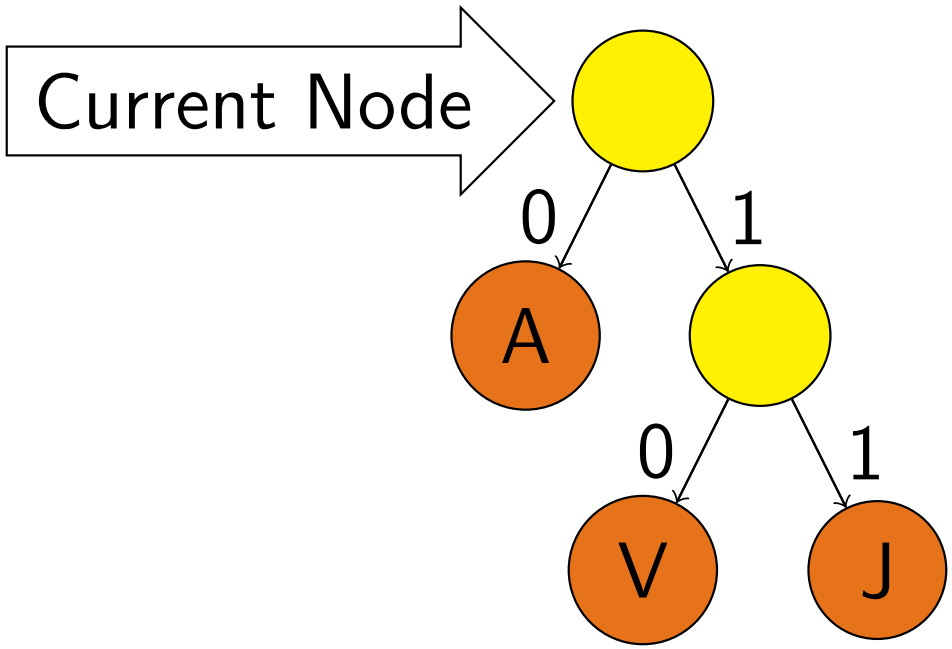
Encoded Bits : “1|10100”
Decoded Text : “”

Decoding from trie



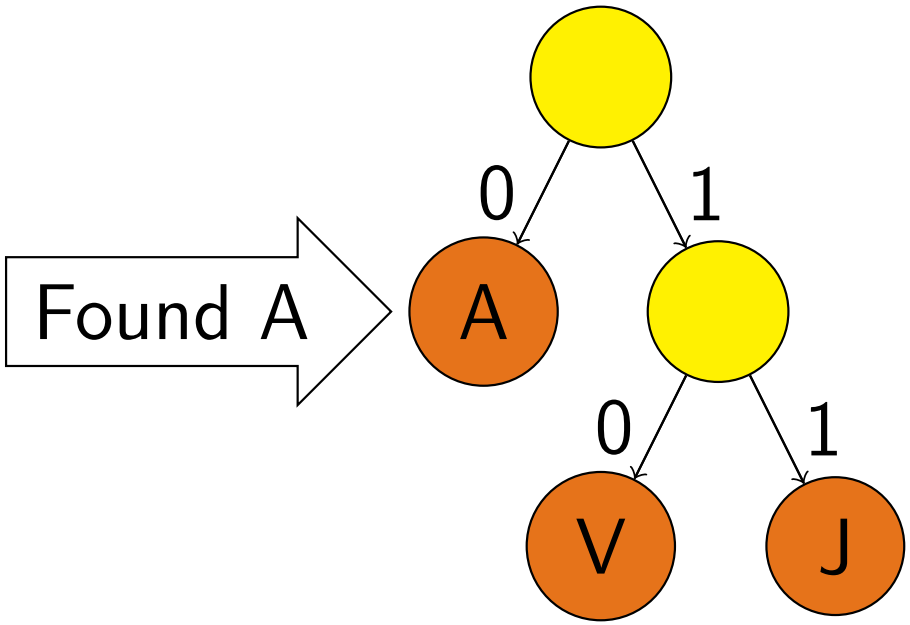
Encoded Bits : “11|0100”
Decoded Text : “”

Decoding from trie



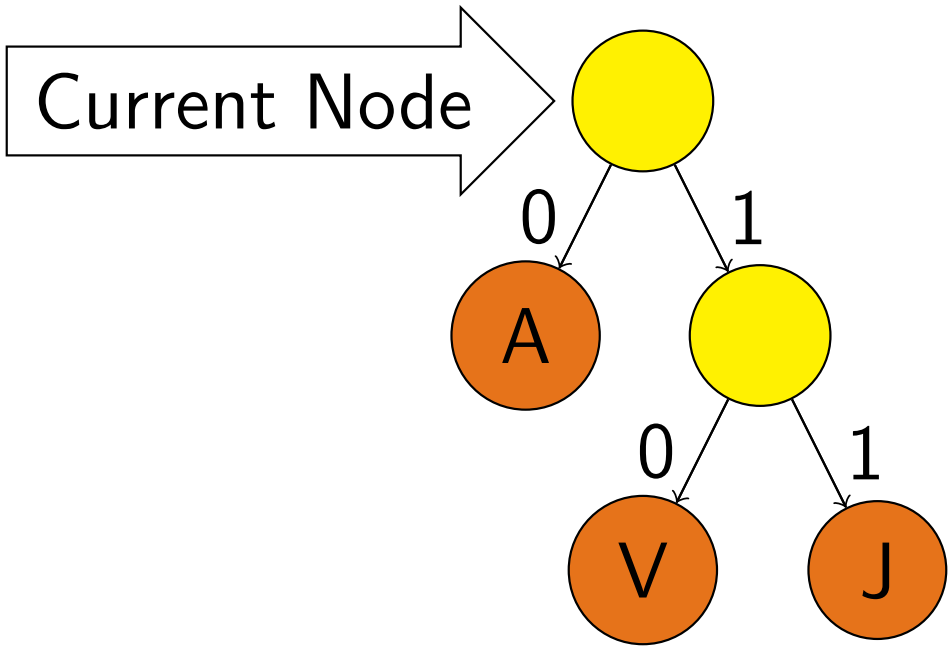
Encoded Bits : “11|0100”
Decoded Text : “J”

Decoding from trie



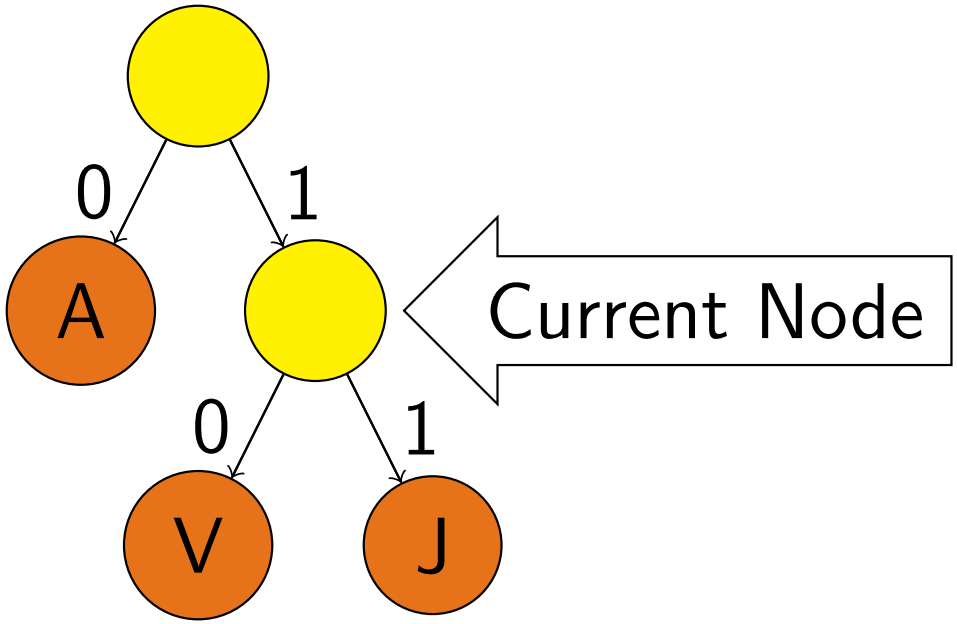
Encoded Bits : “110|100”
Decoded Text : “J”

Decoding from trie



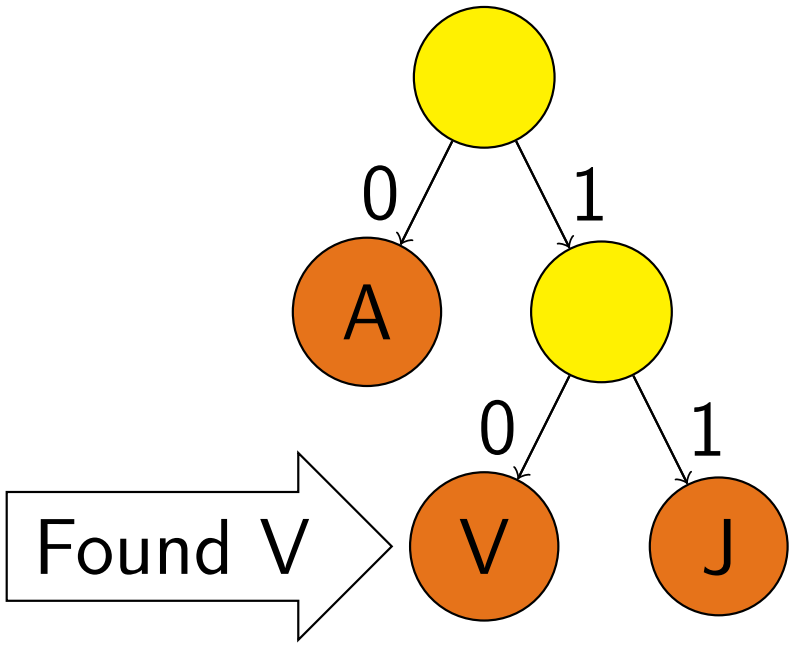
Encoded Bits : “110|100”
Decoded Text : “JA”

Decoding from trie



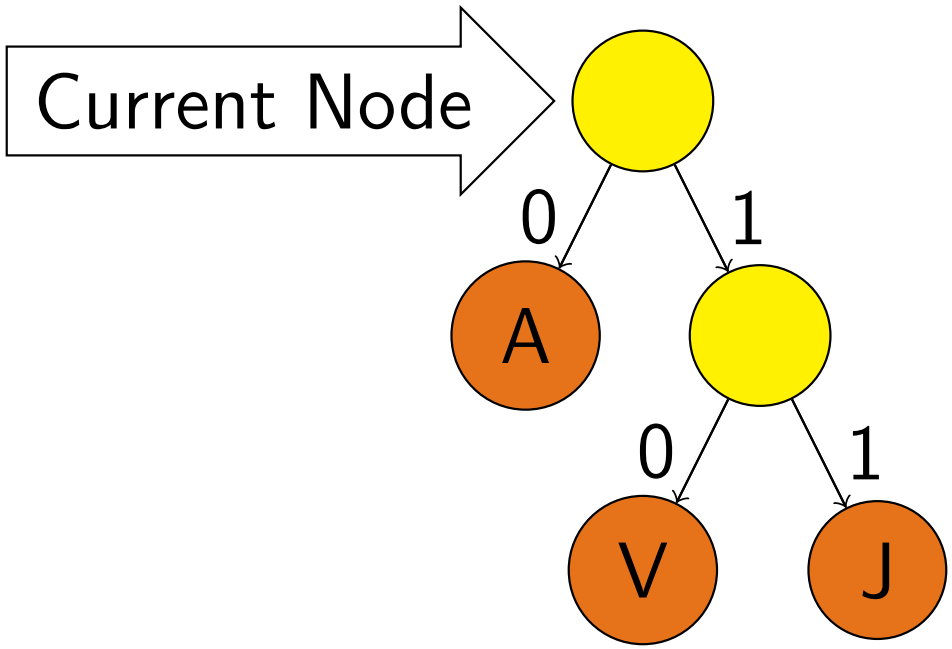
Encoded Bits : “1101|00”
Decoded Text : “JA”

Decoding from trie



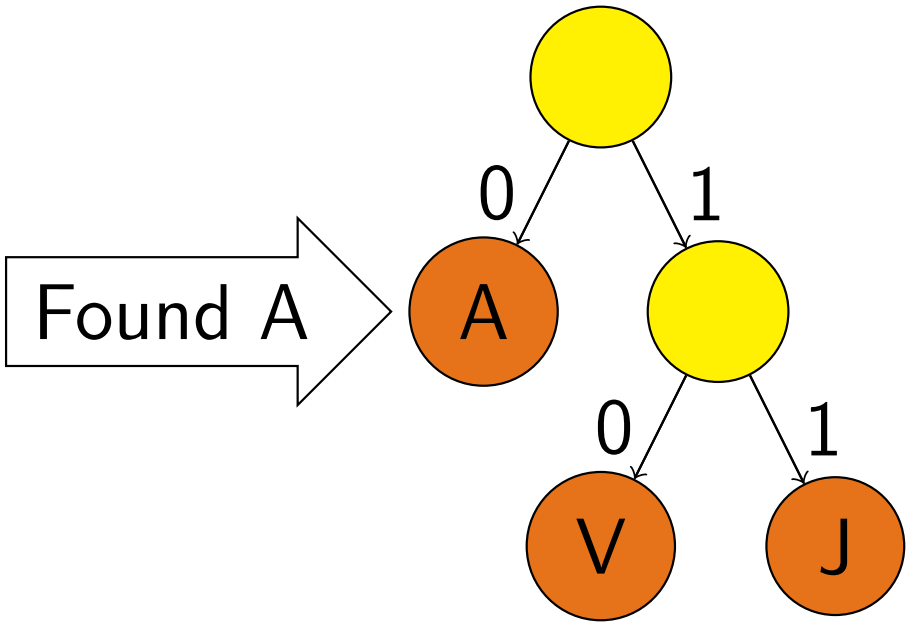
Encoded Bits : “11010|0”
Decoded Text : “JA”

Decoding from trie



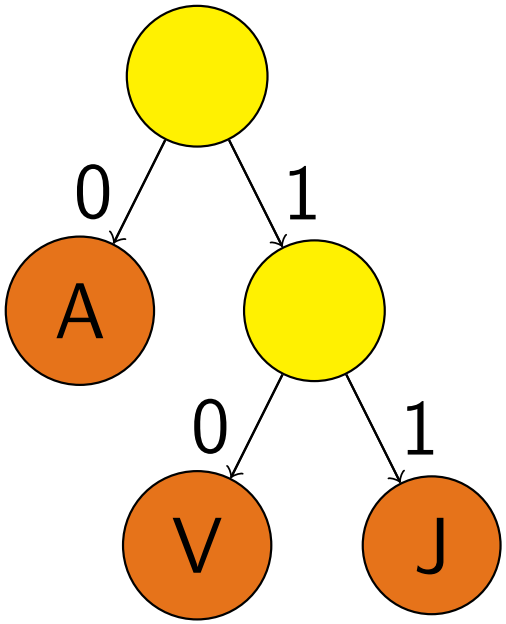
Encoded Bits : “11010|0”
Decoded Text : “JAV”

Decoding from trie



Encoded Bits : “110100|”
Decoded Text : “JAVA”

Decoding from trie



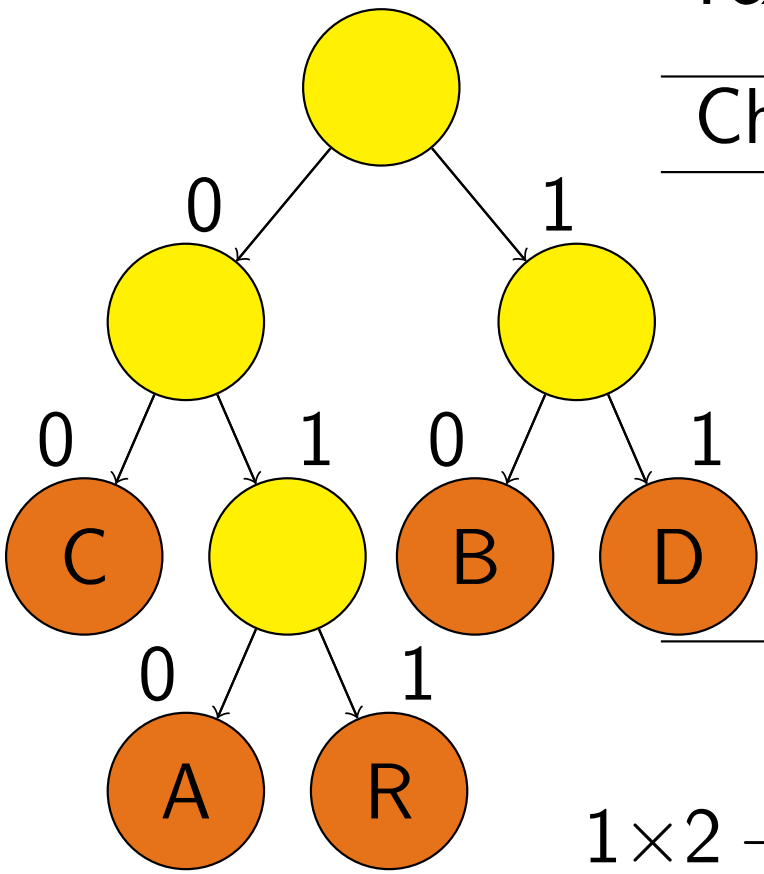
Encoded Bits : "110100"
Decoded Text : "JAVA"

Complexity of decoding = O(length of the encoded bits)

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

Example

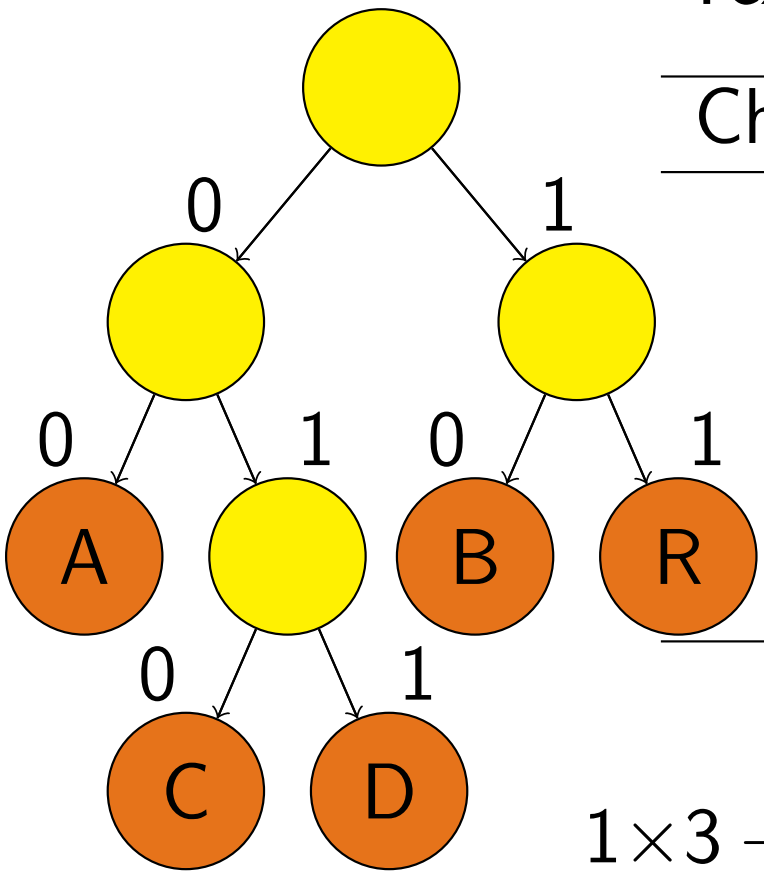


Text = "AAABRACADABRAAA"

Character	Encoding	Frequency
A	010	9
B	10	2
C	00	1
D	11	1
R	011	2

$$\text{Total bits} = 9\times3 + 2\times2 + 1\times2 + 1\times2 + 2\times3 = 41 \text{ bits}$$

Example



Text = "AAABRACADABRAAA"

Character	Encoding	Frequency
A	00	9
B	11	2
C	010	1
D	011	1
R	11	2

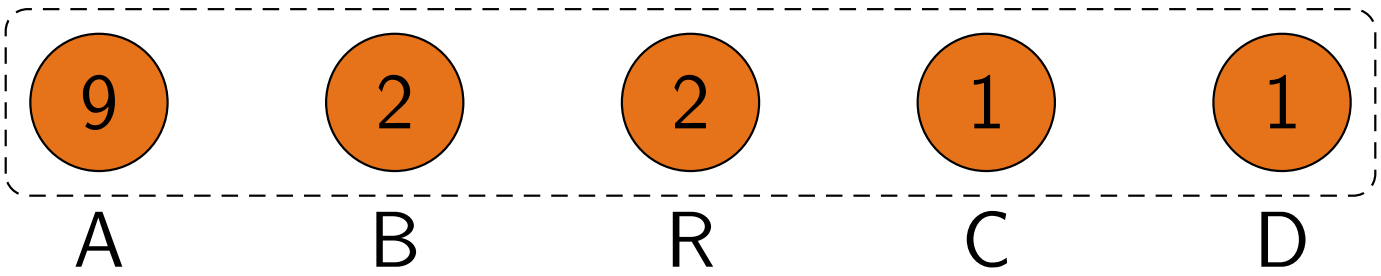
$$\text{Total bits} = 9\times2 + 2\times2 + 1\times3 + 1\times3 + 2\times2 = 32 \text{ bits}$$

Table of Contents

1. An Imaginary Scenerio
2. Fixed Length Code
3. Variable Length Code
4. Reason Why Variable Encoding scheme failed
5. Properties of correct variable length encoding
6. Encoding as Trie
7. Optimal Compression
8. Construction of Huffman trie

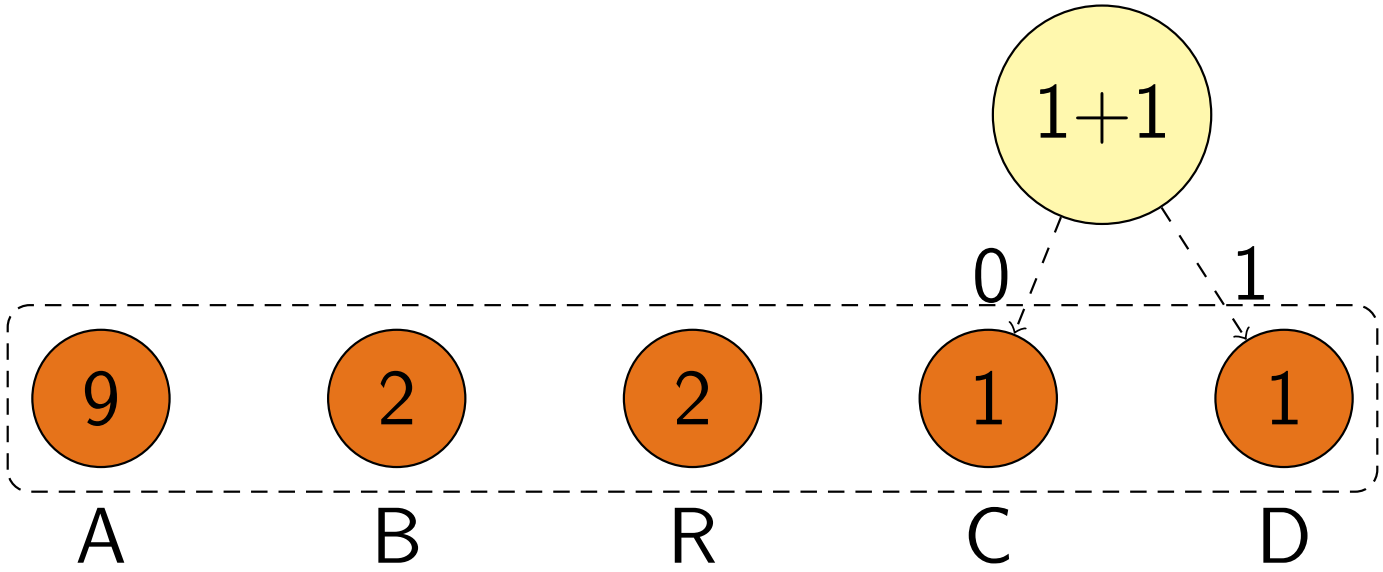
Construction of Huffman trie

AAABRACADABRAAA



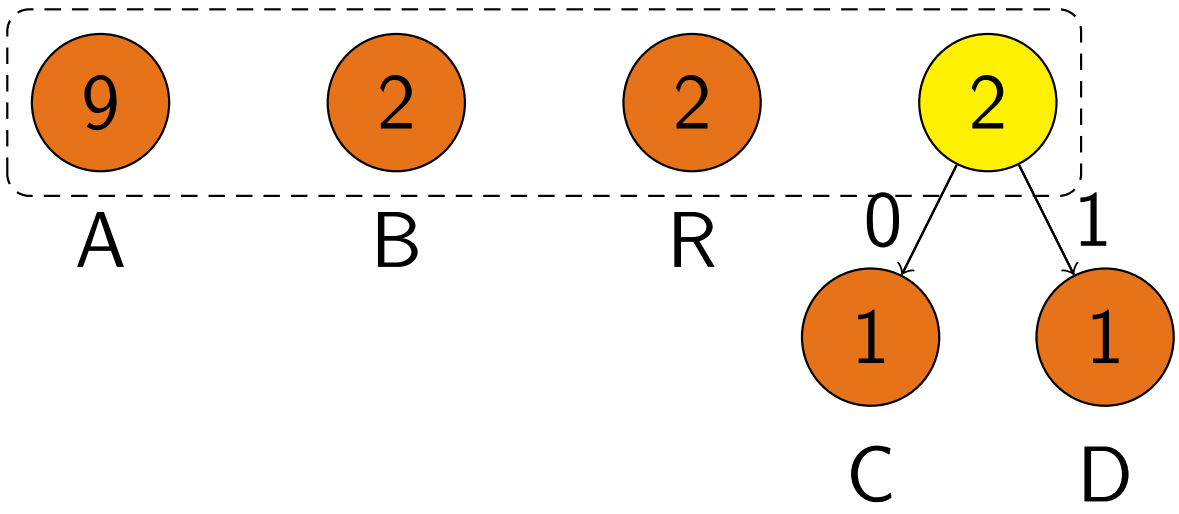
Construction of Huffman trie

AAABRACADABRAAA



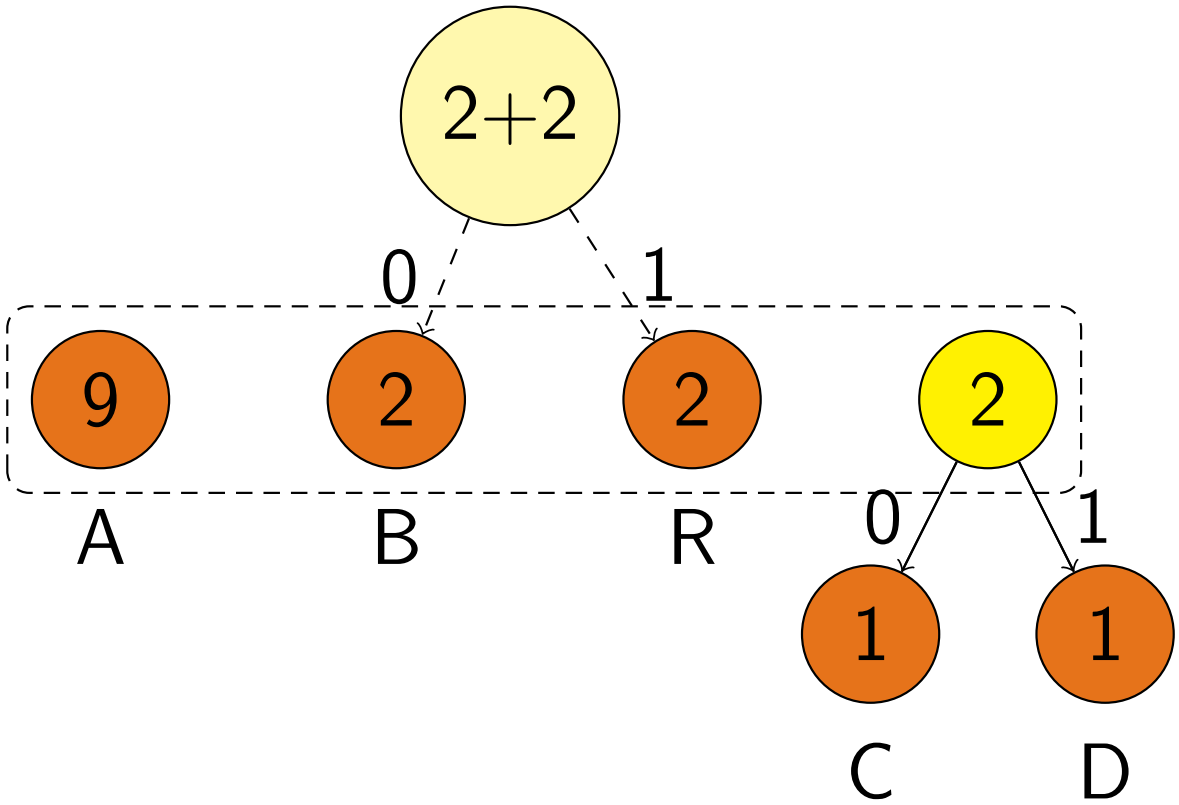
Construction of Huffman trie

AAABRACADABRAAA



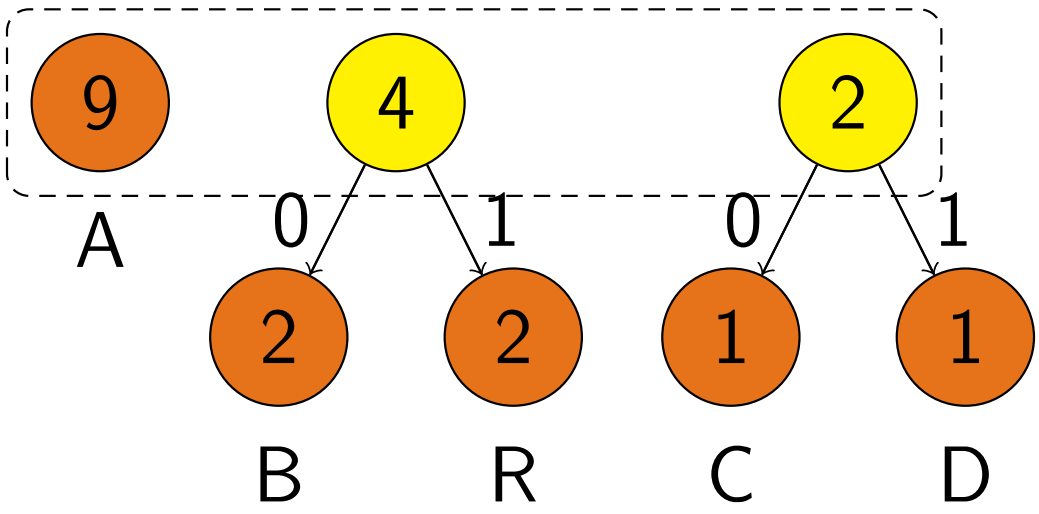
Construction of Huffman trie

AAABRACADABRAAA



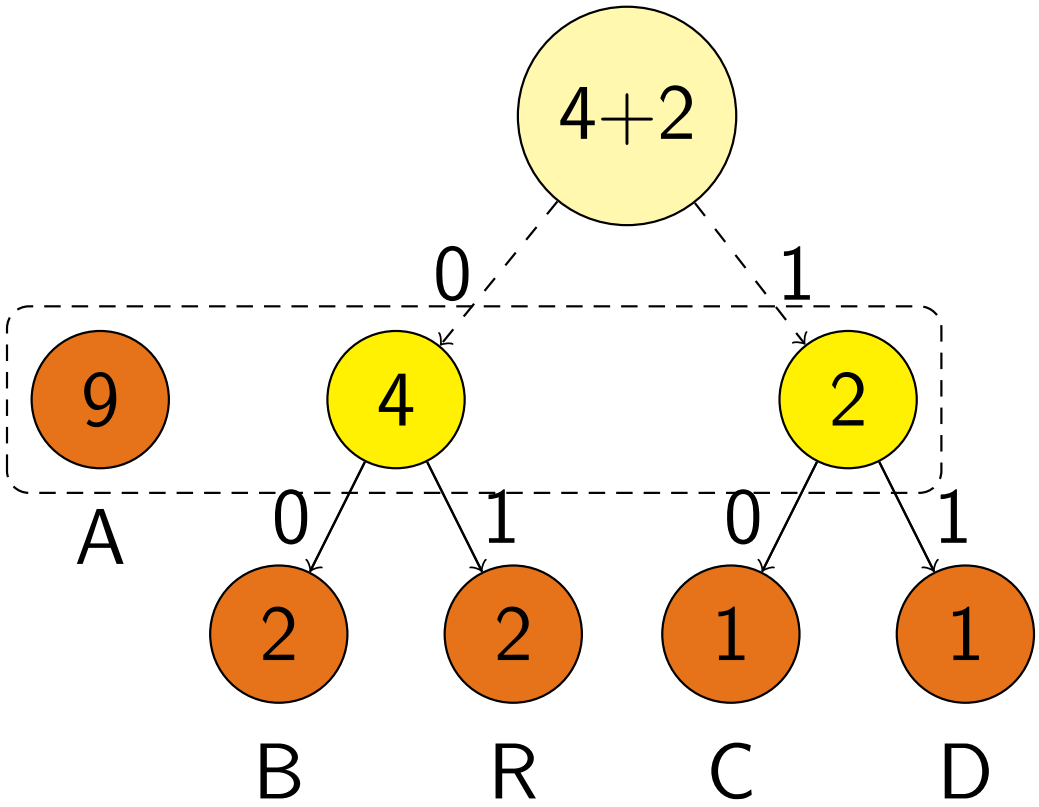
Construction of Huffman trie

AAABRACADABRAAA



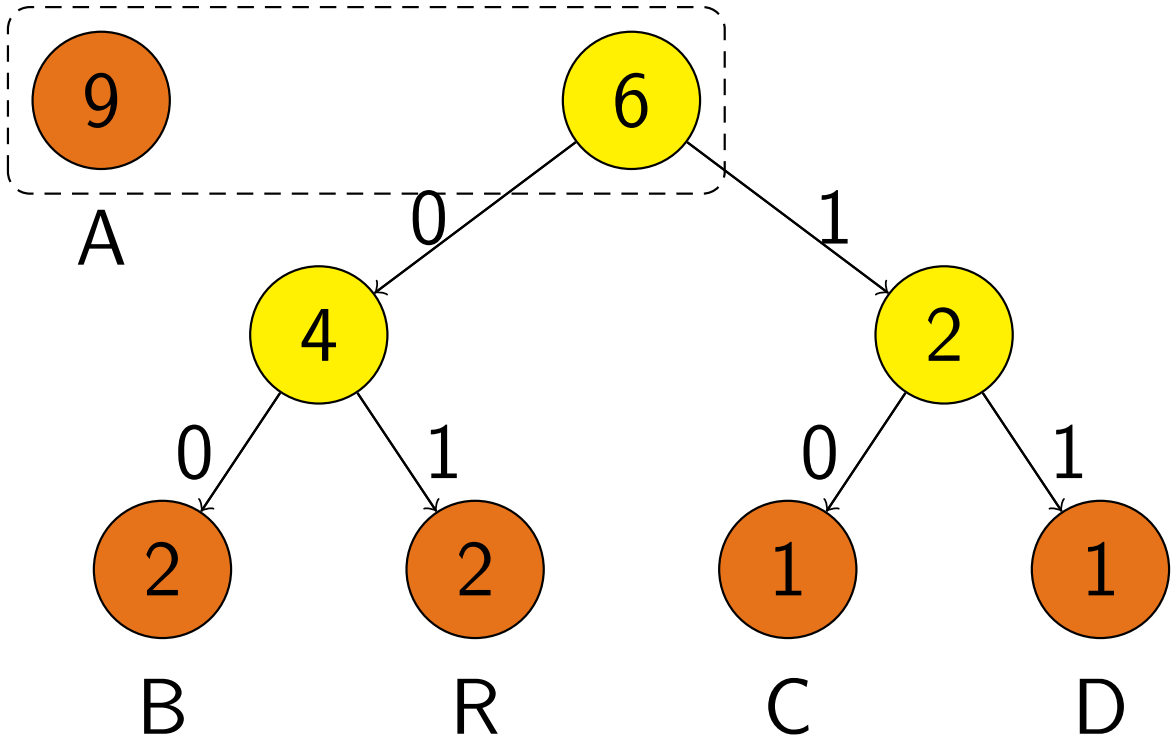
Construction of Huffman trie

AAABRACADABRAAA



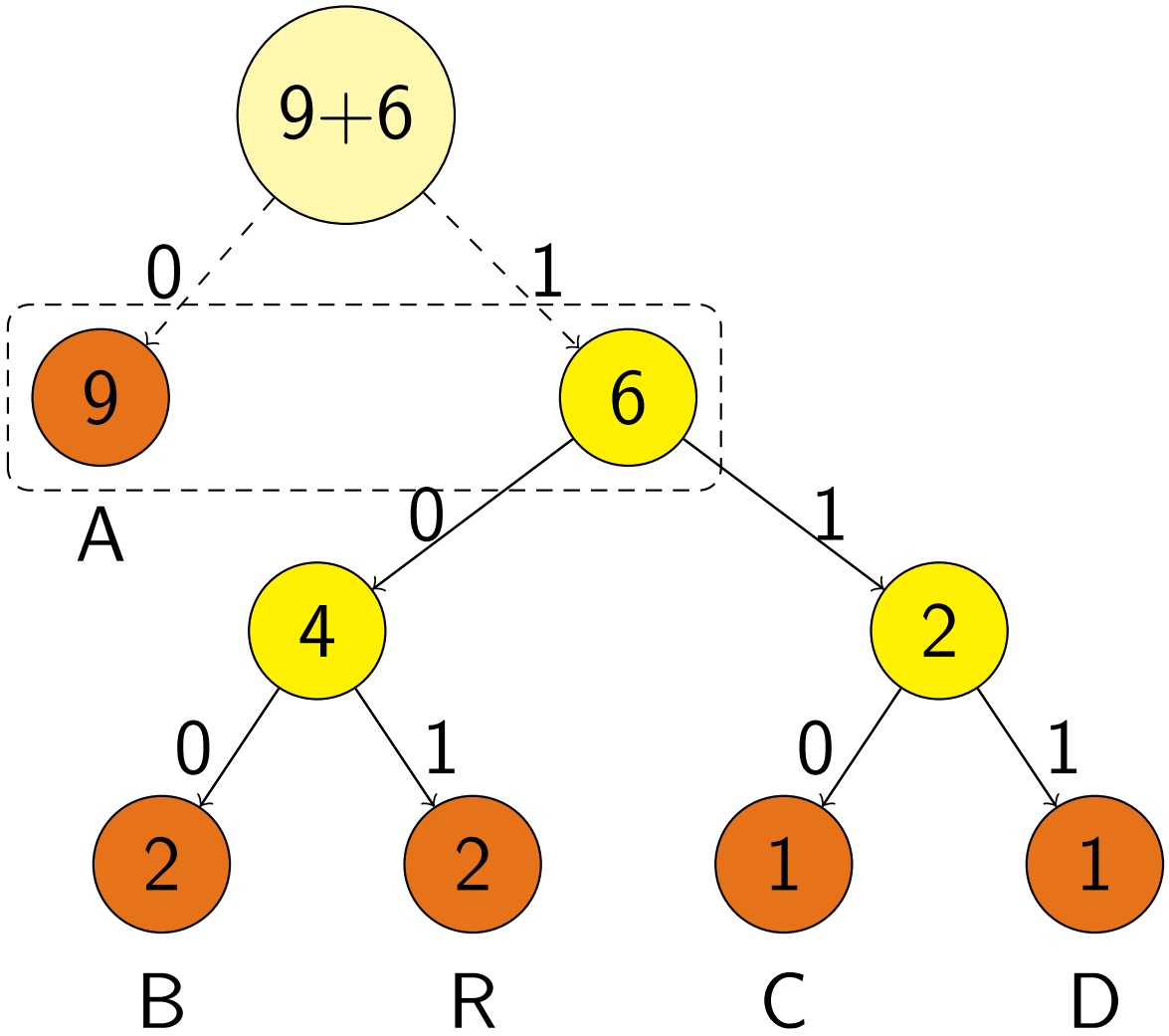
Construction of Huffman trie

AAABRACADABRAAA



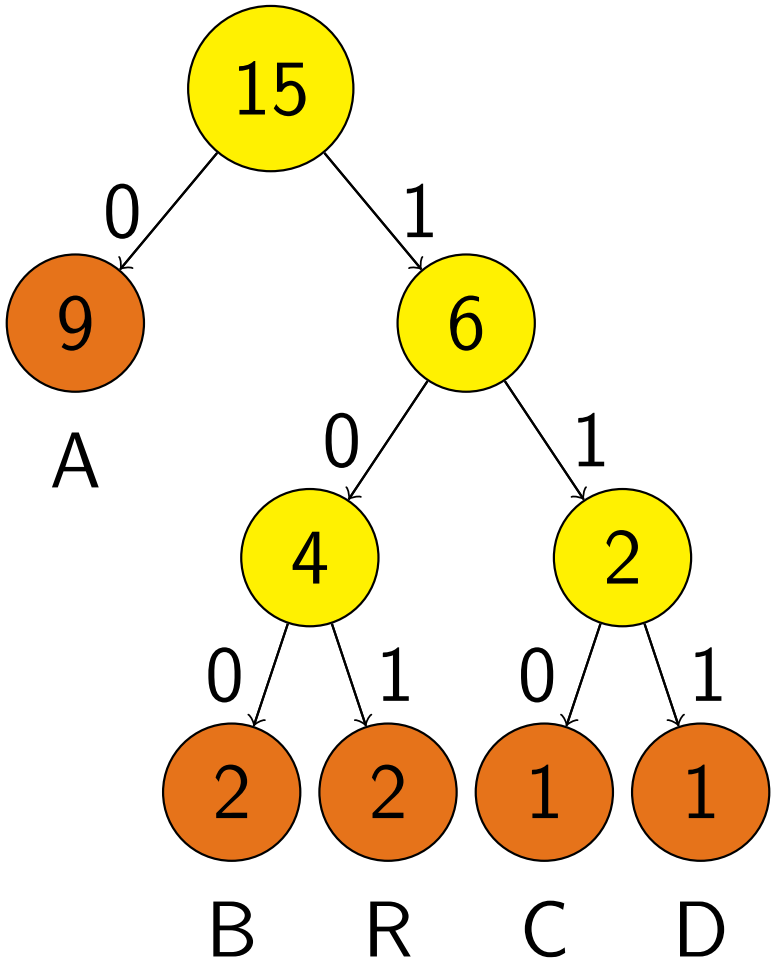
Construction of Huffman trie

AAABRACADABRAAA



Construction of Huffman trie

AAABRACADABRAAA



Character	Encoding	Frequency
A	0	9
B	100	2
C	110	1
D	111	1
R	101	2

$$\text{Total bits} = 9\times1 + 2 \times 3 + 1\times3 + 1 \times 3 + 2 \times 3 = 27 \text{ bits}$$

Results

Final Results

Huffman encoding crushes others!

ASCII	Mod. F. L.	Trie (Rand.)	Huffman Trie
120	45	41	27

Table: Bits needed in different encoding

Thank You