

## Note: LZW decoding

# COMP9319 Web Data Compression and Search

LZW,  
Adaptive Huffman,  
(live lecture)

- There is one special case that the LZW decoding pseudocode presented is unable to handle.
- This is your exercise to find out in what situation that happens, and how to deal with it.
- I'll go through this at the live lecture.

1                   2

## Assignment Project Exam Help

LZW Notes  
(Handling the special case)  
<https://powcoder.com>

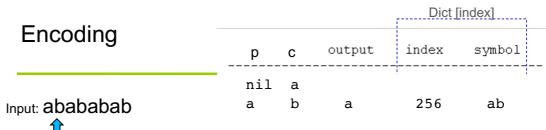


```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

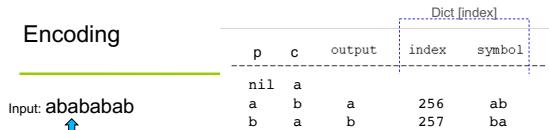
Add WeChat powcoder

3

4



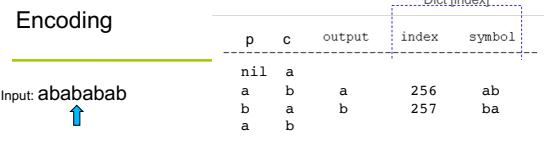
```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```



```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

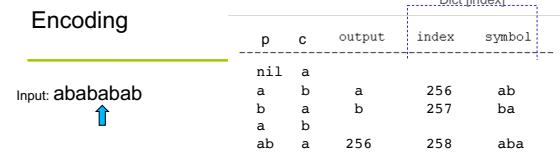
5

6



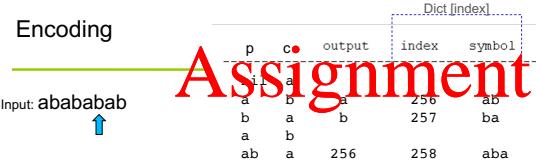
```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

7



```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

8



```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

9



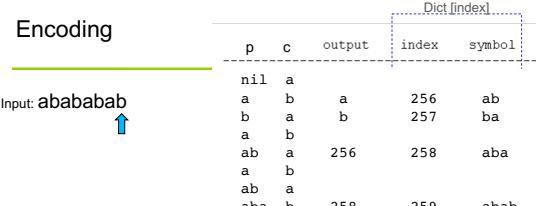
```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

10

# Assignment Project Exam Help

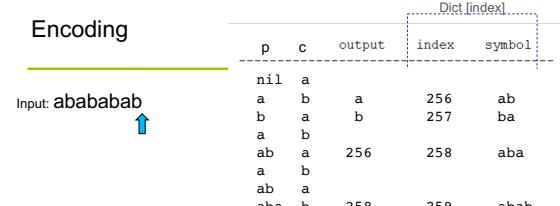
<https://powcoder.com>

Add WeChat powcoder



```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

11



```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
        p = c;
```

12

### Encoding

Input: abababab

| Dict [index] |     |        |       |        |
|--------------|-----|--------|-------|--------|
| p            | c   | output | index | symbol |
| nil          | a   |        |       |        |
| a            | b   | a      | 256   | ab     |
| b            | a   | b      | 257   | ba     |
| a            | b   |        |       |        |
| ab           | a   | 256    | 258   | aba    |
| a            | b   |        |       |        |
| ab           | a   |        |       |        |
| aba          | b   | 258    | 259   | abab   |
| b            | EOF | b      |       |        |

```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
    p = c;
```

13

### Decoding

Input: ab<256><258>b

| Dict [index] |   |        |       |        |
|--------------|---|--------|-------|--------|
| p            | c | output | index | symbol |
| a            | a |        |       |        |

```
read(c); // c is likely to be > 8 bits
output c;
p = c;
while read(c):
    output Dict[c];
    add p + Dict[c][0] to Dict;
    p = Dict[c];
```

14

### Decoding

Input: ab<256><258>b

| Dict [index] |   |        |       |        |
|--------------|---|--------|-------|--------|
| p            | c | output | index | symbol |
| a            | a |        |       |        |
| a            | b |        |       |        |

```
read(c); // c is likely to be > 8 bits
output c;
p = c;
while read(c):
    output Dict[c];
    add p + Dict[c][0] to Dict;
    p = Dict[c];
```

https://powcoder.com

### Decoding

Input: ab<256><258>b

| Dict [index] |   |        |       |        |
|--------------|---|--------|-------|--------|
| p            | c | output | index | symbol |
| a            | b |        |       |        |
| b            | b |        |       |        |

```
read(c); // c is likely to be > 8 bits
output c;
p = c;
while read(c):
    output Dict[c];
    add p + Dict[c][0] to Dict;
    p = Dict[c];
```

16

Add WeChat powcoder

15

### Decoding

Input: ab<256><258>b

| Dict [index] |   |        |       |        |
|--------------|---|--------|-------|--------|
| p            | c | output | index | symbol |
| a            | b |        |       |        |
| b            | b |        |       |        |

```
read(c); // c is likely to be > 8 bits
output c;
p = c;
while read(c):
    output Dict[c];
    add p + Dict[c][0] to Dict;
    p = Dict[c];
```

<256> <258> WHAT???

17

### Encoding

Input: abababab

| Dict [index] |     |        |       |        |
|--------------|-----|--------|-------|--------|
| p            | c   | output | index | symbol |
| nil          | a   |        |       |        |
| a            | b   | a      | 256   | ab     |
| b            | a   | b      | 257   | ba     |
| a            | b   |        |       |        |
| ab           | a   | 256    | 258   | aba    |
| a            | b   |        |       |        |
| ab           | a   |        |       |        |
| aba          | b   | 258    | 259   | abab   |
| b            | EOF | b      |       |        |

```
p = nil; // p for prev char
while read(c):
    if pc ∈ Dict:
        p = pc;
    else:
        add pc to Dict;
        output code(p);
    p = c;
```

18

## Decoding

| Input: ab<256><258>b |       |        |              |        |
|----------------------|-------|--------|--------------|--------|
| p                    | c     | output | Dict [index] | symbol |
| a                    | a     | a      | 256          | ab     |
| b                    | <256> | ab     | 257          | ba     |
| <256> <258>          | aba   | 258    | 259          | aba    |

## Decoding

| Input: ab<256><258>b |       |        |              |        |
|----------------------|-------|--------|--------------|--------|
| p                    | c     | output | Dict [index] | symbol |
| a                    | a     | a      | 256          | ab     |
| b                    | <256> | ab     | 257          | ba     |
| <256> <258>          | aba   | 258    | 259          | aba    |
| <258>                | b     | b      |              | bab    |

19

20

## Adaptive Huffman (FGK)

# Assignment Project Exam Help

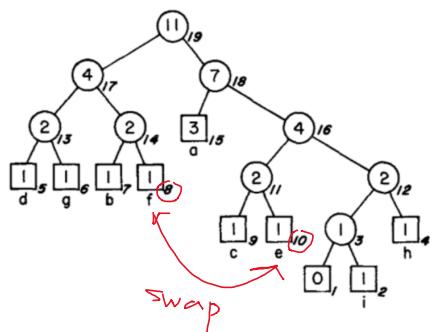
## Adaptive Huffman (FGK)



22

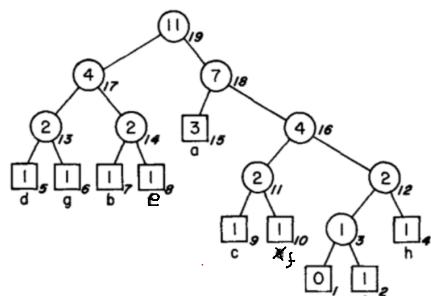
23

## Adaptive Huffman (FGK)



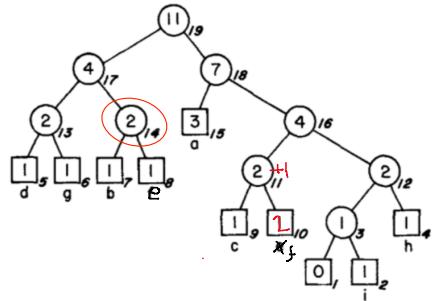
24

## Adaptive Huffman (FGK)



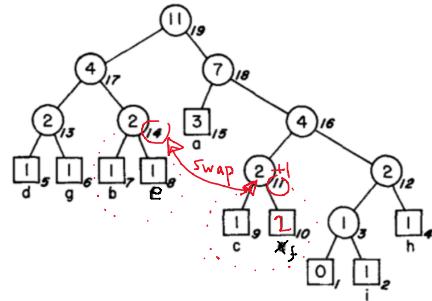
25

## Adaptive Huffman (FGK)



26

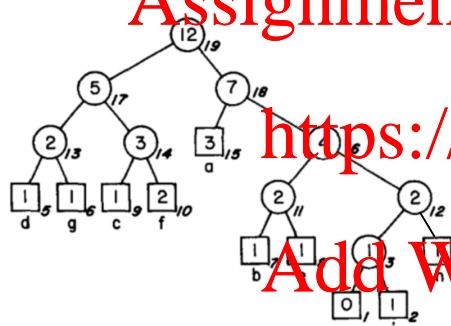
## Adaptive Huffman (FGK)



27

Adaptive Huffman (FGK): when  
f is inserted

## Assignment Project Exam Help

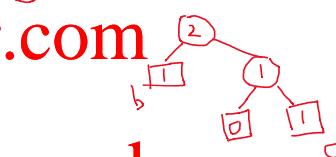


28

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001,  
b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string bcaabb.



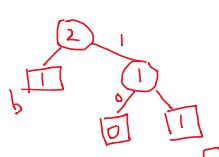
01100010 001100011

20

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001,  
b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string bcaabb.



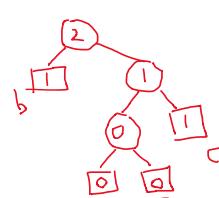
01100010 001100011 1001100001

21

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001,  
b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string bcaabb.



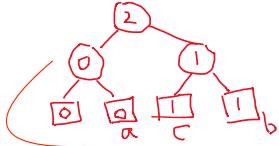
01100010 001100011 1001100001

22

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



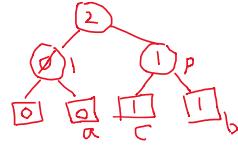
01100010 001100011 1001100001

23

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



01100010 001100011 1001100001

24

## Adaptive Huffman (Ex2, Q2)

# Assignment Project Exam Help

The initial coding before any transmission is: a=01100010, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



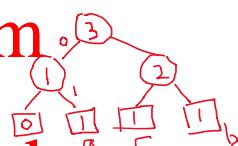
01100010 001100011 1001100001

25

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



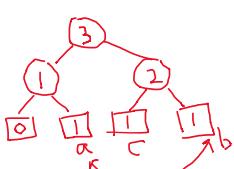
01100010 001100011 1001100001 01

26

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



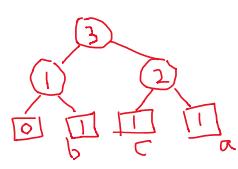
01100010 001100011 1001100001 01

27

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



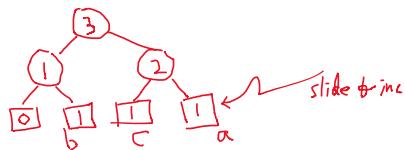
01100010 001100011 1001100001 01

28

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



01100010 001100011 1001100001 01

29

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



01100010 001100011 1001100001 01

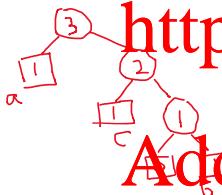
30

## Adaptive Huffman (Ex2, Q2)

# Assignment Project Exam Help

The initial coding before any transmission is: a=01100010, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



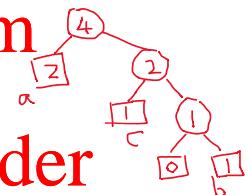
01100010 001100011 1001100001 01

31

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



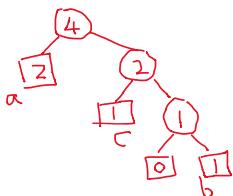
01100010 001100011 1001100001 01

32

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



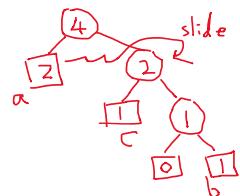
01100010 001100011 1001100001 01 0

33

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



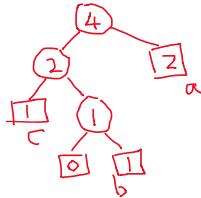
01100010 001100011 1001100001 01 0

34

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



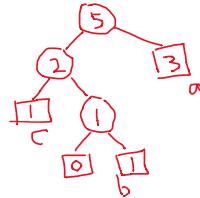
35

01100010 001100011 1001100001 01 0

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



36

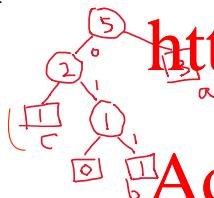
01100010 001100011 1001100001 01 0

## Adaptive Huffman (Ex2, Q2)

# Assignment Project Exam Help

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



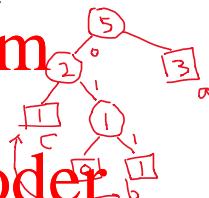
37

01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



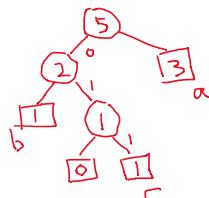
38

01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



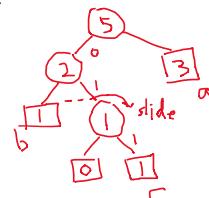
39

01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



40

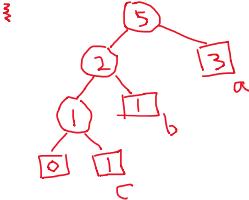
01100010 001100011 1001100001 01 0 011

**https://powcoder.com**  
**Add WeChat powcoder**

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.

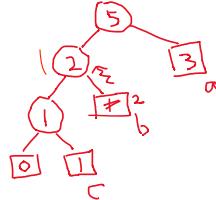


41      01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.

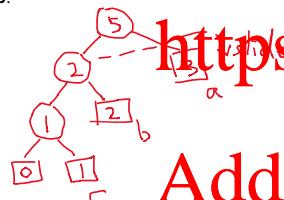


42      01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100011, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.

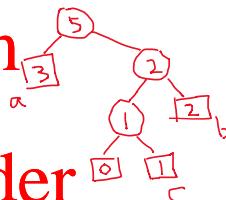


43      01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.

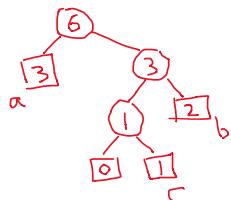


44      01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.

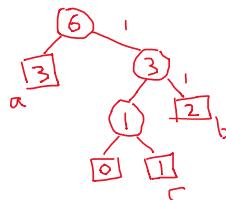


45      01100010 001100011 1001100001 01 0 011

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



46      01100010 001100011 1001100001 01 0 011

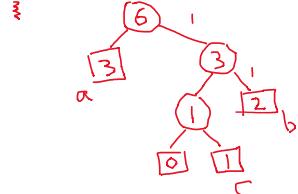
<https://powcoder.com>

Add WeChat powcoder

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



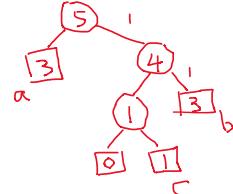
01100010 001100011 1001100001, 01 0 011 11

47

## Adaptive Huffman (Ex2, Q2)

The initial coding before any transmission is: a=01100001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



01100010 001100011 1001100001 01 0 011 11

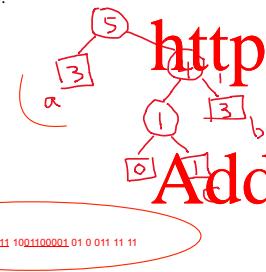
48

## Adaptive Huffman (Ex2, Q2)

# Assignment Project Exam Help

The initial coding before any transmission is: a=110001, b=01100010, c=01100011.

Derive the encoded bitstream produced by the encoder for the string **bcaaabb**.



01100010 001100011 1001100001, 01 0 011 11 11

49

<https://powcoder.com>

Add WeChat powcoder