COMP 261 Test 4

27 May 2023

Instructions

- Time allowed: **50 minutes** .
- Answer **all** the questions. There are 50 marks in total.
- Write your answers in the booklets and remember to include your Student ID.
- If you are a remote student, then write the answers on paper and submit a photo to the relevant submission system entry.
- If you think some question is unclear, ask for clarification.
- This test contributes 25% of your final grade
- You may use paper translation dictionaries, and non-programmable calculators.
- You may write notes and working on this paper, but make sure your answers are clear.

Questions	**Marks**
1. Data Compression	[26]
2. String Search	[16]
3. Fast Fourier Transform	[8]

1. Data Compression **(26 marks)**

*****Huffman Coding*****

(a) **(8 marks)** The following text is to be encoded using Huffman coding, based on the letter frequencies in the text itself:

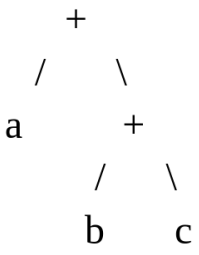
`to_be_or_not_to_be`

- Construct the relevant binary tree,
- Show the code it assigns to each character.

Note 1: If the two nodes have the same frequency in the priority queue, pick the node with the smaller character alphabetically. Specifically, "_" comes **before all the lowercase letters.*

Note 2: When building a parent node, use the child node with the smaller frequency as the **left child.*

**Note 3: You can use a "+" to represent the non-leaf nodes when drawing a Huffman tree, like:*



Draw your Huffman tree below:

The code for each character is:
t:
o:
_:
b:
e:
r:
n:

(b) **(2 marks)** Now encode the string using your above codebook and show the encoded binary bit stream:

(c) **(1 mark)** What is the compression rate compared with the result of a fixed-length coding method where 3 bits are used for representing one symbol?
Note: If your encoded sequence has 40 bits, but the fixed-length encoding give you 50bits, the compression rate is 40/50 = 80%

(d) **(2 marks)** If we use the **above huffman tree** to encode the following sequence: `net_born_rent`
You will have a longer sequence than fixed-length coding. Please explain why using the previous Huffman coding tree can not result in a good compression rate for the new sequence?

EXTRA ANSWER BOX IF NEEDED (PLEASE INDICATE THE QUESTION IDs):

Lempel-Ziv Coding

(e) **(3 marks)** Suppose that the following tuple sequence is an encoding result by the Lempel-Ziv encoding method:

```
[0, 0, 'b'][0, 0, 'e'][1, 1, 't'][0, 0, '_'][5, 3, '_'][9, 4, '']
```

**Note: The final '' in the last tuple means an empty character*

Which one of the following four strings is the correct decoding result for the previous tuple sequence: ☐

(A) beet_beet_bee

(B) bet_bet_bet_t

(C) beet_bee_beet

(D) beet_beet_bee

(f) **(4 marks)** For the following string:

```
a_cat_catches_it
```

Which option shows the correct encoding result using the Lempel-Ziv method? ☐

Note: Here, we assume the length of the sliding search window behind the current character to be processed is 16 and the size of the "lookahead" window for the substring/pattern to search is 16.

(A) `[0,0,'a'][0,0,'_'][0,0,'c'][3,1,'t'][0,0,'t'][4,4,'h'][0,0,'h'][0,0,'e'] [0,0,'s'][12,1,'i'][0,0,'i'][11,1,'']`

(B) `[0,0,'a'][0,0,'_'][0,0,'c'][3,1,'t'][4,4,'c'][0,0,'h'][0,0,'e'][0,0,'s'] [12,1,'i'][11,1,'']`

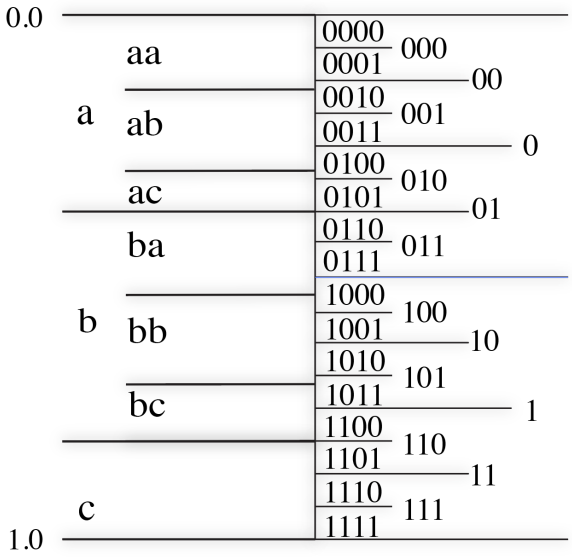
(C) `[1,1,'a'][1,1,'_'][1,1,'c'][3,1,'t'][4,4,'c'][1,1,'h'][1,1,'e'][1,1,'s'] [12,1,'i'][11,1,'']`

(D) `[0,0,'a'][0,0,'_'][0,0,'c'][3,1,'t'][4,1,'_'][4,3,'c'][0,0,'h'][0,0,'e'] [0,0,'s'][12,1,'i'][11,1,'']`

EXTRA SPACE TO WORK OUT THE QUESTIONS (WILL NOT BE MARKED)

Arithmetic Coding

(g) **(6 marks)** Suppose that we have an alphabet of {a, b, c}, with a probability distribution of {0.375, 0.4375, 0.1875}. Then we can have the following partitioning scheme:



Please choose the correct answer for each of the following questions:

(1) For the string "aa", what is the binary code sequence after encoding it using the arithmetic coding algorithm? [ ]

(A) 000 (B) 0000 (C) 001 (D) 0

(2) For the string "ac", what is the binary code sequence after encoding it using the arithmetic coding algorithm? [ ]

(A) 010 (B) 0101 (C) 0100 (D) 00

(3) When doing **on-the-fly** encoding for transmitting "aa", when read in the first 'a', what will be transmitted? [ ]

(A) 1 (B) a (C) nothing (D) 0

(4) When doing **on-the-fly** encoding for transmitting "ba", when read in the first 'b', what will be transmitted? [ ]

(A) b (B) nothing (C) 1 (D) 0

2. String Search **(16 marks)**

(a) **(6 marks)** Show the partial match table **M** generated for the KMP algorithm for the following search string **S**:
The string S: ”ananxany”

S	a	n	a	n	x	a	n	y
M	-1	0						

(b) **(10 marks)** Show the steps the KMP algorithm using the above partial match table of S will take when searching through the following text T until the match is found. You only need to show the **start position of each match attempt** and **the part that can been matched** before the fail position.

Pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
T	a	n	a	n	t	a	n	a	n	x	a	n	a	n	x	a	n	y

**Note: Please use the following format to show where the match attempts are for each step. One example has been given.*
Note: **You may not need to fill in all the following empty steps given in the box. Please just end if a result can be returned from the KMP algorithm.*

Match Attempt 1
Start Position in T: **T[0]** Partially Matched Characters: **”anan”**
Return a value? (If yes, please write down the value. Otherwise, write ”No”): **No**

Match Attempt 2:
Start Position in T: Partially Matched Characters:
Return a value? (If yes, please write down the value. Otherwise, write ”No”):

Match Attempt 3:
Start Position in T: Partially Matched Characters:
Return a value? (If yes, please write down the value. Otherwise, write ”No”):

Match Attempt 4:
Start Position in T: Partially Matched Characters:
Return a value? (If yes, please write down the value. Otherwise, write ”No”):

Match Attempt 5:
Start Position in T: Partially Matched Characters:
Return a value? (If yes, please write down the value. Otherwise, write ”No”):

Match Attempt 6:
Start Position in T: Partially Matched Characters:
Return a value? (If yes, please write down the value. Otherwise, write ”No”):

Match Attempt 7:
Start Position in T: Partially Matched Characters:
Return a value? (If yes, please write down the value. Otherwise, write ”No”):

3. Fast Fourier Transform **(8 marks)**

(a) **(4 marks)** Suppose we are using FFT to evaluate the following polynomial $P(x)$:
$P(x) = x^7 + 6x^6 + 3x^5 + 9x^4 + x^3 + 2x^2 + 7$
What are the two lower-degree polynomials (P_{even} and P_{odd}) we should evaluate during the subsequent recursion step following the splitting of the polynomial P(x): ☐
Hint: P_{odd} is **NOT** directly obtained by combining all the odd-degree individual terms.

(A) $P_{even}(x^2) = 6x^6 + 9x^4 + 2x^2$, $P_{odd}(x^2) = x^7 + 3x^5 + x^3 + 7$
(B) $P_{even}(x^2) = 6x^6 + 9x^4 + 2x^2$, $P_{odd}(x^2) = x^6 + 3x^4 + x^2$
(C) $P_{even}(x^2) = 6x^6 + 9x^4 + 2x^2 + 7$, $P_{odd}(x^2) = x^7 + 3x^5 + x^3$
(D) $P_{even}(x^2) = 6x^6 + 9x^4 + 2x^2 + 7$, $P_{odd}(x^2) = x^6 + 3x^4 + x^2$

(b) **(4 marks)** When using FFT algorithm for polynomial multiplication, what are the **fourth roots of unity** that can be used as the four points to evaluate the values of an input polynomial of **degree 3**? Please choose from the following options: ☐

(A) $1, -1, 1+i, 1-i$
(B) $1, -1, i, 0$
(C) $1, -1, i, -i$
(D) $i, i+1, i+2, i+3$

**********END OF TEST 4 **********

* * * * * * * * * * * * * * *