1. Linux compression commands:

Here is the compression commands demoed in the class. Please find a linux terminal to practice the commands using different files.

Following links are recommended: <https://explainshell.com/> and https://tldr.sh/

7za a tensorflow-master

tar -zcf tensor.tar.gz tensorflow-master

tar -jcf tensor.tar.bz2 tensorflow-master

tar -Zcf tensor.tar.Z tensorflow-master

7za x tensorflow-master.7z

tar -Zxf tensor.tar.Z

tar -jxf tensor.tar.bz2

tar -zxf tensor.tar.gz

**(Already done in class and you seen us do it)**

**Question 2:**

***One method of reducing bandwidth use is to compress the data being transmitted. Let A = {a/20, b/15, c/5, d/15, e/45} be the alphabet and its frequency distribution. Compute the optimal coding for each character. What is the average number of bits/symbol of the codes?***

Combine the two lowest frequencies:

* + Combine 'a' and 'c': (a+c)/25
  + Combine 'b' and 'd': (b+d)/30
  + Combine the resulting nodes with 'e': ((a+c)/25 + (b+d)/30 + e)/75

Assign binary codes:

* + 'a': 00
  + 'c': 01
  + 'b': 10
  + 'd': 11
  + 'e': 1

Calculate the average number of bits per symbol:

* + Average = (2 \* 20/100 + 2 \* 5/100 + 2 \* 15/100 + 2 \* 15/100 + 1 \* 45/100) bits/symbol = (40/100 + 10/100 + 30/100 + 30/100 + 45/100) bits/symbol = 155/100 bits/symbol = 1.55 bits/symbol

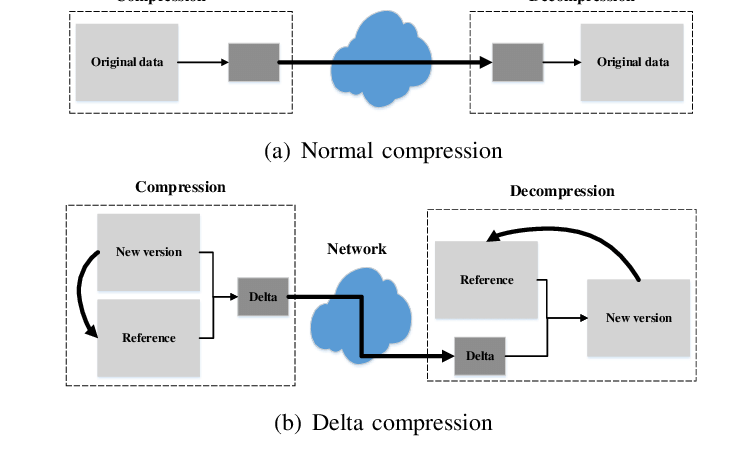
So, the average number of bits per symbol of the codes is 1.55 bits/symbol.

**Question 3:**

**Please draw the information exchange flow chart according to the diagram for delta compression.**

Chart

Description automatically generated with medium confidence



**Question 4:**

**One method of reducing bandwidth use is to compress the data being transmitted. Use the LZW algorithm to compress the string: BABAABAAA. Note that Uppercase A has ASCII value 65 in decimal. Draw diagrams to aid your explanation if appropriate.**

* + **Process the first character 'B':**
    - Add 'B' to the current input string.
    - 'B' is not in the dictionary, so output the code for the empty string (0).
    - Add 'B' to the dictionary with code 1.
    - Reset the current input to 'B'.
    - Output: 0 (code for empty string).
  + **Process the next character 'A':**
    - Add 'A' to the current input string ('BA').
    - 'BA' is not in the dictionary, so output the code for 'B' (1).
    - Add 'BA' to the dictionary with code 2.
    - Reset the current input to 'A'.
    - Output: 1 (code for 'B').
  + **Process the next character 'B':**
    - Add 'B' to the current input string ('AB').
    - 'AB' is not in the dictionary, so output the code for 'A' (65).
    - Add 'AB' to the dictionary with code 3.
    - Reset the current input to 'B'.
    - Output: 65 (ASCII value of 'A').
  + **Process the next character 'A':**
    - Add 'A' to the current input string ('BA').
    - 'BA' is in the dictionary with code 2.
    - Reset the current input to 'A'.
    - Output: 2 (code for 'BA').
  + **Process the next character 'A':**
    - Add 'A' to the current input string ('AA').
    - 'AA' is not in the dictionary, so output the code for 'A' (65).
    - Add 'AA' to the dictionary with code 4.
    - Reset the current input to 'A'.
    - Output: 65 (ASCII value of 'A').
  + **Process the next character 'B':**
    - Add 'B' to the current input string ('AB').
    - 'AB' is in the dictionary with code 3.
    - Reset the current input to 'B'.
    - Output: 3 (code for 'AB').
  + **Process the next character 'A':**
    - Add 'A' to the current input string ('BA').
    - 'BA' is in the dictionary with code 2.
    - Reset the current input to 'A'.
    - Output: 2 (code for 'BA').
  + **Process the final character 'A':**
    - Add 'A' to the current input string ('AA').
    - 'AA' is in the dictionary with code 4.
    - Reset the current input to 'A'.
    - Output: 4 (code for 'AA').

**Output:**

The compressed output sequence is: 0 1 65 2 65 3 2 4.

OR ELSE THIS WAY

|  |  |  |  |
| --- | --- | --- | --- |
| **Representing** | **Next Character** | **ASCII Code** | **Dictionary Entry** |
| B | A | - | - |
| BA | B | 66 | 3: “BA” |
| A | B | - | - |
| AB | A | 65 | 4: “AB” |
| B | A | - | - |
| BA | A | 3 | - |
| AA | A | 66 | 5: “AA” |

**BABAABAAA = 66 65 3 66**