The University of New South Wales

COMP9319 Web Data Compression and Search
Final Exam 20T2 (SAMPLE ONLY)

Tuesday 25 August 2020

1. Time Allowed: 3 hours
2. To be completed between:
   9:00am-9:00pm Tue 25 August 2020 (AEST)
3. **Carefully read the notes below before you start the exam.**

By submitting the exam answers via WebCMS or give, you declare that **all of the work submitted for this exam is your own work, completed without assistance from anyone else**.
Please refer to the Student Conduct web site for details.

Notes

1. General Instructions:

- Total number of questions: 5
- Total marks available: 50
- Answer **all** questions.
- Questions are not worth equal marks.
- Marks are shown on each question.
- Questions may be answered in any order.
- During the 3-hr Exam, you must **not**:
  - access any of your own files
  - access any web pages except the Lecture slides in this course at WebCMS3 and the C or C++ reference.
- During the entire 12-hr period from 9:00am Tue 25 August 2020 (AEST), you must **not**:
  - communicate with other students in *any* way
- Your answers must be submitted using `give` or via WebCMS
- If you have any clarification questions between AEST 9:00am-5:00pm 25th August, 2020 (we may be unavailable outside this period), please email cs9319@cse.unsw.edu.au via an UNSW email account.
- You are expected to write your program using simple C or C++ constructs, functions and libraries, but you may include and use any C or C++ libraries available in CSE linux machines.

There are two types of same questions below: programming and written questions. For each of the programming questions, you are required to put your C or C++ code in one `.cc` file. A sample template `.cc` file for each of these questions is provided. For each of the written questions, you are required to type or draw your answers in a PDF file. You may use any software to type or draw your answers (e.g., Microsoft Word) as far as the final submission is in PDF format.

If you believe that insufficient information has been provided to answer a given question, then you should write any assumptions (as comments in your code, or in the PDF) that you think are necessary to complete the question and continue work from there.

For the programming question, you can only receive full marks for a correctly-working program that compiles with no warnings by simply running "`make`" on a CSE linux machine. If your program is close to correctly-working, you may receive partial-marks. We do not measure speed nor runtime memory usage of your program, but it should complete and terminate in a reasonable time.

2. Submission:

You can submit your exam solution either using:

i) WebCMS: Login to Course Web Site > Exam > Final Exam > Final Exam Paper > Make Submission > upload required files > [Submit]

Or:

ii) The give command: `give cs9319 exam makefile q1.cc q2.cc q3.cc q4.pdf q5.pdf`

Required Files: `makefile q1.cc q2.cc q3.cc q4.pdf q5.pdf`

Deadline: Tuesday 25 August 2020 at 9:00pm (AEST)

No late submissions will be accepted.

3. Downloads:

Downloads: exam.tgz or exam.zip

Both `.tgz` and `.zip` files contain the same material.
Each archive contains a sample `makefile`; an answer template file for each programming question and a sample test file for each programming question.

4. How to Start:

- read these notes carefully and completely
- download one of the archive files above
- unpack the downloaded file
- for programming questions, modify and put your code in the corresponding `.cc` file
- modify the supplied makefile accordingly so that the make command will generate a correct executable program from your `.cc` file
- for written questions, type and/or draw your answers using an appropriate software that can save them as PDF
- login to `wagner` or a CSE linux machine, and test your makefile and programs
- the sample test files provided are provided for requirement clarifications, you are expected to make more tests yourself to check for your program correctness
- submit all the Required Files via WebCMS3 (or give) as described above

End of Notes

Exam Questions

Question 1 (8 marks)

Write an encoder (called `q1`) the static Arithmetic Coding algorithm as presented in the lecture week 1 in C or C++. The encoder takes a filename as a commandline argument. The input file may contain at least 1 and at most 10 characters with ASCII value between 0 and 126 inclusively. Assume that the symbols will divide the number line [0, 1) according to their lexicographical order, the encoder will output the low and the high (separated by a space) of the final result to the standard output, as shown in the example below. No penalty to your solution if more than necessary number of decimal digits are printed.

```
%wagner> cat test1.txt
BILL GATES%wagner>
%wagner> ./q1 test1.txt
0.2572167752 0.2572167756
%wagner>
```

Instructions:

- Put all your code in the provided `q1.cc`. Modify the makefile accordingly.

Question 2 (10 marks)

Suppose T is a text file that contains at least 1 and at most 5120 characters with ASCII value between 0 and 126 inclusively, and it is always ended with a newline `\n` (the only newline in the file). Write a BWT decoder (called `q2`) that reverses the BWT encoded T back to the original T. q2 accepts two commandline arguments: the name of a BWT encoded file; the output filename. The following is a simple test of q2 using the provided sample test:

```
wagner % ./q2 test2.bwt myoutput.txt
wagner % diff myoutput.txt test2.txt
wagner %
```

Instructions:

- Put all your code in the provided `q2.cc`. Modify the makefile accordingly.

Question 3 (12 marks)

Suppose T is a text file that contains at least 1 and at most 5120 characters with ASCII value between 0 and 126 inclusively, and it is always ended with a newline `\n` (the only newline in the file). Write a BWT backward search program (called `q3`) that performs backward search on the BWT encoded T given with a search term, and outputs the number of matches (including zero) to the standard output. q3 accepts two commandline arguments: a search term (for simplicity, you may assume that it is a combination of alphabets and digits); the name of a BWT encoded file. The following is a simple test of q3 using the provided sample test:

```
wagner % ./q3 test3.bwt ATATA
68
wagner %
```

Instructions:

- Put all your code in the provided `q2.cc`. Modify the makefile accordingly.

Question 4 (8 marks)

Given the text string below:

```
jejunojejunostomy
```

a. What is its entropy?

b. Draw a Huffman tree based on the letters and their corresponding distributions for the above text string (Do not need to draw trees for the intermediate steps).

c. Provide the resulting Huffman code for each letter.

d. What is the average number of bits needed for each letter, using your Huffman code? How does it compare to the entropy ? (i.e., equal/larger/small and why)

Instructions:

- Save your answer and/or drawing as a file called `q4.pdf` in PDF format.

Question 5 (12 marks)

Using the linear-time SA construction algorithm discussed in the class, derive the Burrows Wheeler transform BWT(S) of character sequence S:

```
database$
```

Show all the steps involved. After that, derive the Move-to-Front transform of the BWT(S), assuming we use the 255 ASCII symbols as the symbol table. Note: If you need an ASCII table, use the "`ascii`" command available on CSE linux machines.

Instructions:

- Save your answer and/or drawing as a file called `q5.pdf` in PDF format.

End of Exam