CS 345: Homework 3 (Due: Oct 19, 2015)

Rules. Individual homeworks; see Handout 1 (aka Syllabus). Hardcopies may be submitted no later than start of class the day they are due; electronic copies by 23:59:59 the same day.

Problem 1. (12 points) **Query-based corpus size estimation**

This is a follow up of the w_1, w_2 problem of Homework 2. In that problem you had to statistically and probabilistically estimate with three queries the corpus size of Google. In this problem you are asked to devise a single query that will generate as a response in Bing and Google as many documents as the Corpus size of the corresponding search engine. In class we tried `a` that worked fine for Google but not for Bing. Perform the same query in Google and Bing to obtain an approximate Corpus size for both engines. Make sure you get some number over 20G and less than 40G that is also consistent with Problem 1 of HW2. Provide screenshots to justify your answer.

```
Query performed |
                |
-------------------------------------------------------------------
1.Bing results  |
                |
-------------------------------------------------------------------
2.Google results|
                |
-------------------------------------------------------------------
```

Problem 2. (12 points) **(Heaps' Law)**
Is the state of Google's dictionary (lexicon) in 1998 (use paper L1 and Subject 2 for reference) consistent with Heaps' Law i.e. how many words does the lexicon maintain and how many unique words does Heaps' Law predict about the 150GB or so of the document collection?

Problem 3. (14 points) **(Elias etc)**

(a) Provide v-codes for 100 and 400 in hexadecimal.
(b) You are given an inverted list for a term t that looks like

```
(10,1)(10,5)(10,11)(10,15)(10,17)(10,20)(12,10)(12,18)(12,30)(12,40)(3,5)(13,11)(13,25)(14,80)(14,81)
```

v-byte encode this inverted list by providing the following information. (Note that v-byte encoding is also discussed on page 150 of the textbook. However, the example of the textbook incorrectly applies gap encoding to a count field shown here in bold-face, contradicting the algorithm stated there or in the notes: *1,2,1,6,1,***2***, 6, 11, etc.*)
(b1) Give the flattened (no parenthesis) form of the to be v-byte encoded list just before the v-byte encoding is applied when the list is flattened but still in decimal notation, and
(b2) after the v-byte encoding has been applied and the list given in hexadecimal notation.

Problem 4. (28 POINTS) **Hash Table design circa 1998**
You have 512MiB of main memory of which 200MiB are being used by the operating system plus related programs such as those manipulating tables A and T below. You are asked to organize the remaining space to support a hash table along the lines of the paper where words are stored in a contiguous table which is an array A of characters delimited by a null character `\0`. A hash table T will store a `wordID` along with a reference (pointer or index) p to A. That way a `wordID` will be associated with a specific `word`.

The average length of a `word` is given as 9 characters stored in ASCII. A `wordID` and p can only be in multiples of one byte (i.e. 1B, 2B, 3B but and thus 21bits won't be an option) for efficiency. Organize T and A for maximum efficiency.

In an efficient implementation answer the following question by filling the data below for $n, m, \ldots$.

- (a) How many words n can the scheme support,
- (b) How big would the hash table size m (number of entries) be,
- (c) How many bits for a `wordID`,
- (d) How many bits for pointer p
- (e) How much space (bytes) will you scheme use for T,
- (f) How much space (bytes) will you scheme use for A,
- (g) What is the total space $A + T$ used by your scheme?

Justify your answers and choices. Round to nearest million for n, m, T, A but make sure you don't exceed the amount of available memory (i.e. $A + T$ should be accommodated easily by the available memory). Make sure that you fill the following table with data.

```
   n        =
   m        =
   wordID   =
   p        =
   T        =
   A        =
 A+T        =
```

Date Posted: 10/1/2015