Compression of the dictionary and posting lists
Summary of class discussion – Part 1

Remarks on Zipf's law (covered in Section 5.1.2 of *Introduction to Information Retrieval*):

General law: f_i = frequency of the i^{th} most frequent item $\propto i^{-\theta} f_1$

for some constant θ. (symbol $\propto$ denotes "is proportional" or "grows as") For our application, items are terms that appear in the documents of a collection. This gives rise to the textbook's notation cf_i standing for "content frequency". One study gives θ of 1.5-2.0 for this application. The law is observed to hold for other applications with varying values of θ. The text *Introduction to Information Retrieval* focuses on $\theta = 1$. (The text also uses a general constant c rather than cf_1.)

Taking logs, we have a linear relationship between $\log(f_i)$ and $\log(i)$:

$$\log(f_i) \propto \log(f_1) - \theta \log(i)$$

f_i could refer to either the fraction of the total number of occurrences or an actual count of occurrences. If f_i is the actual count of occurrences, *M* is the number of distinct terms and *T* is the total count of occurrences of all items, then

$$f_i \propto \frac{T}{\sum_{j=1}^{M} j^{-\theta}} i^{-\theta} .$$

($\sum_{j=1}^{M} j^{-\theta}$ is a well-known mathematical quantity: the order θ harmonic number of M.)

Heap's Law:
The material covered in class is identical to Section 5.1.1 of *Introduction to Information Retrieval.*

Dictionary compression:
The dictionary compression we considered in class is covered in Section 5.2 of *Introduction to Information Retrieval*.

We can do a very rough estimate of the size of a modern Google dictionary from the size of the dictionary of the early Google in 1998. To apply Heap's law, we need the number of tokens in each collection. We don't have this information, but we'll substitute the size in bytes of the collection and the index and just look at the growth rate. 1998 Google had 147.8GB of documents and a 53.5GB index using lossy compression. The documents contained 14×10^6 unique terms. In 2010, Google reported that its new index structure was 100PB. So growth has been from roughly 100GB to 100PB, or a factor of one million. Then Heap's Law gives:

$$M_{1998} = K*(T_{1998})^{\beta} \text{ and } M_{2010} = K*(T_{2010})^{\beta} \approx K*(T_{1998} *10^{6})^{\beta}$$

assuming that K and β have not changed in 22 years. Empirically, β is about 0.5, giving

$$M_{2010} \approx 10^{3} * K*(T_{1998})^{\beta} = 10^{3} * M_{1998} = 10^{3} * (14 * 10^{6})$$

Thus we estimate a dictionary of 14 billion terms.

Using this estimate of dictionary size and using these values:

- 1 byte per character
- 10 characters *on average* per term
- 5-byte pointers into the character string of terms
- 8-byte pointers to the postings lists

compressing the dictionary using one long string of terms and pointers into the string requires approximately $(14\times10^{9}) \times (5 + 10 + 8) =$ 322 GB. Compare this to the $(14 \times10^{9}) \times (20 +8) =$ 392 GB of an array of term entries with 20 bytes allocated per term or to the $(14 \times10^{9}) \times (30 + 8) =$ 532 GB of an array with 30 bytes allocated per term.