CMPSCI 345 — Homework 11

Map Reduce

For the following exercises, map and reduce functions can be written informally as textual description or pseudo-code for full credit. You may use the simple map-reduce programs in the course slides as a reference for the level of detail required.

> **Extra Credit (15 points)** *For problems 1(a), 2(a), and 2(b) below, implement the map-reduce programs in Java or Python, create sample data, and execute the programs. You should submit your source code, the sample input files, and include a transcript of the output produced. An easy way to implement a MapReduce program in Python is to use the* `mrjob` *package. If you prefer Java, you may install Hadoop. In either case, you need only execute your program locally to complete the extra credit. Note: this extra credit can be turned in after the deadline of the main assignment; see Moodle.*

1. (30 points) Consider a collection of Web documents identified by their URL. The input is a collection of key-value pairs (url, doc) where url is the URL and doc is the text content of the document.

   (a) Write map and reduce functions to compute an *inverted index*. An inverted index lists, for each distinct word appearing in any document, the *distinct* URLs of each document that contains at least one appearance of the word.

   (b) Decide if the following two map reduce programs return the same output and explain why they do or why they do not.

   - **Map1**: given input (url, doc) parse doc and divide it into separate words. Then for each word w in doc, emit $(0, length(w))$ where $length(w)$ is the number of characters in the word.
   - **Reduce1**: given input $(0, (l_1, \ldots l_n))$ return the maximum of $l_1 \ldots l_n$
   - **Map2**: given input (url, doc) parse doc dividing it into separate words $w_1 \ldots w_m$. Compute the maximum length a of the words in doc and emit $(0, a)$.
   - **Reduce2**: given input $(0, (a_1, \ldots a_k))$ return the maximum of $a_1 \ldots a_k$

   (c) Decide if the following two map reduce programs return the same output and explain why they do or why they do not.

   - **Map3**: given input (url, doc) parse doc and divide it into separate words. Then for each word w in doc, emit $(0, length(w))$ where $length(w)$ is the number of characters in the word.
   - **Reduce3**: given input $(0, (l_1, \ldots l_n))$ return the average of $l_1 \ldots l_n$
   - **Map4**: given input (url, doc) parse doc dividing it into separate words $w_1 \ldots w_m$. Compute the average length a of the words and emit $(0, a)$.

- **Reduce4**: given input $(0, (a_1, \ldots a_k))$ return the average of $a_1 \ldots a_k$

2. (20 points) Consider input consisting of records of the following form:

(line-number, longitude, latitude, continent, month, day, year, temp)

(a) Write map and reduce functions to compute the distinct years for which there are more than 100 temperature observations.

(b) Write map and reduce functions to compute the average temperature for each continent in each year, as long as there are more than 50 temperature observations for that continent and year.