15-712 Advanced and Distributed Operating Systems (Spring 2012) Homework 1

Due: Feb. 10th, 2012, 11:59 PM

Markov Sentence Generator using MapReduce

A new semester has begun. Lots of homework and project reports are waiting for you, not to mention papers you want to finish.

Harry Bovik got similar problems, but now he has a clever plan—using a random sentence generator. Instead of making himself to work on papers, he is going to make computers to work on papers for him. Unfortunately, he is really busy with preparing ACH SIGBOVIK 2012.

Could you help him by writing a random sentence generator?

In particular, you will implement a Markov chain sentence generator on Hadoop.

Markov chain sentence generators construct a sentence incrementally using word frequency information derived from some corpus text. Beginning with a seed word, a generator keeps appending a new random word based on the previous word(s), while the more frequently the word pair has been seen in the corpus, the more likely the word pair appears in the generated sentence. Look for the examples of Markov chain sentence generators on the Web. For example,

- `http://allanwirth.com/javascript/markov.html`
- `http://www.jwz.org/dadadodo/dadadodo.cgi`

Rules:

- You must write your own code; however, you can collaborate with other classmates for discussing high-level ideas.
- You must use OpenCloud, a Hadoop cluster at CMU. You can request an account:
  `https://yogi.pdl.cmu.edu/www.pdl/Registration/opencloud-user-reg.html`
  Feel free to contact the course staff if you have difficulty in obtaining access to a Hadoop cluster.
- The Markov chain your program builds and uses must be of at least order 1 (i.e., assigning a state for each word in the corpus).
- You may choose your corpus freely, while it must contain at least 1,000 plain English words.
- Try your sentence generator with at least 10 seed words. Your program should be able to process the seed words together, not one-by-one. Try to obtain some nontrivial (not too short) sentences.

Please include the following in your submission:

- The source code and Makefile of your program
- Instructions about building and executing your program.
- The corpus file you used and its source (e.g., “Wikipedia article: Largest organisms”)
- Sample seed words
- Raw Hadoop output

- Generated sentences for the chosen corpus and seed words (fine to have sentences making no sense)

The grading criteria are as follows:

- Correctness. The submitted code should implement a random sentence generator correctly as described in the rules above.
- Efficiency. The program should take advantage of the parallelism provided by Hadoop.
- Easiness to grade. By following the instructions you have supplied, the grader (TA) should be able to run your program and verify the output without difficulties.

It may be helpful to think about the following questions while doing the assignment:

(a) What parts of your program are easy to write and what else are difficult to parallelize in Hadoop? (Hint: MapReduce is said to be useful for "embarrassingly parallel" workloads.)

(b) How would be the performance characteristic of your program? How would your program perform for more dynamic workloads (i.e., rapidly changing seed words and/or corpus text)?

(c) Does your program explicitly take care of possible Hadoop node failure? (If not, who would handle?) How will the failure be handled?

Start early! Hadoop clusters may be overloaded by the due date. Also, please be careful not to exhaust computing resources, as you are sharing the Hadoop cluster.