15-440 Project 3: Map-Reduce

Read The Web

Vital Stats

Partners: Yes

Due Date: December 2nd

Handin procedure: Check into your repository

Introduction

Can computers learn to read? We think so. "Read the Web" is a research project that attempts to create a computer system that learns over time to read the web. Since January 2010, a computer system called NELL (Never-Ending Language Learner) has been running continuously, attempting to perform two tasks each day:

- First, it attempts to "read," or extract facts from text found in hundreds of millions of web pages (e.g., `playsInstrument(George_Harrison, guitar)`).
- Second, it attempts to improve its reading competence, so that tomorrow it can extract more facts from the web, more accurately.

At present, NELL has accumulated a knowledge base of 461,939 beliefs that it has read from various web pages. It is not perfect, but NELL is learning.

(http://rtw.ml.cmu.edu/rtw/)

The Question

We'd like you to use *Hadoop* to create your own much simpler version of NELL. We have provided some data files on the hadoop cluster for you to run your algorithm on. Three input files are located at HDFS:/15-440/webinput/ To read the files, just set your job's input path to be "/15-440/webinput/", preferably by using a command-line argument. Output your results to a directory of your own.

Here is an outline of the algorithm you will be implementing:

1. Start with a list of seed words.

2. For each sentence in the data set, if it contains a seed word, create a series of template 4-grams around the seed word. For example, if the seed word is "dog" and the sentence is "The red dog likes to eat cookies.", then you could generate the following:
   The red #SEED# likes
   red #SEED# likes to
   #SEED# likes to eat

3. You will then rank these template sentences based on the number of occurrences and take the top 10 of them.

4. Then, for each sentence in the data set, if it matches one of the top 10 templates, you save the word that matched in place of #SEED#. For example, if your template is "red #SEED# likes to", and your sentence is "Every red goat likes to dance.", then you would save the word "goat".

5. You then count how often each of the words you saved appeared and output the top 10 (but make sure you don't output any of the seed words). Also output the counts of each word.

How you implement this algorithm in hadoop is up to you. Good luck!

The Deliverables

- Your source code
- Your makefile
- File describing command line arguments your code takes
- The relevant portion of the raw output of your program run with a set of seeds of your choosing as well as these seed word sets:
  - cat, dog
  - desk, table
  - milk, water, soda

  Here is the sample output of our version when run on the set: cat, dog:

  most 4052
  best 1475
  first 926
  way 700
  few 525
  things 433
  main 396
  following 390
  largest 382
  many 372

  Note: Your output may not be the same! This is **OKAY**!
  Also note: The internet is weird. The output may be weird. This is also okay!

Hints

- You will find the hadoop API very useful: http://hadoop.apache.org/common/docs/r0.20.2/api/
  - Take a look at the FileSystem class which may come in handy
  - You may also need to use the java.util.regex package
- Before you start coding, figure out a solid algorithm and break it into hadoop map-reduce phases (hint: it won't be just 1 phase)
- Start early!