Processing large-scale datasets

The goal of this practical session is to experiment with various tools (XML SAX, HDFS, Hadoop MapReduce, Spark) to perform computations over a large amount of data. Specifically, we will construct an *inverted index* from Wikipedia. To have a simpler example dataset, we will use the dump of Simple English Wikipedia, a version of Wikipedia made in a simpler language than the regular English Wikipedia, and that is far smaller. But of course, the techniques used can be deployed for (much) larger datasets, provided sufficient computing resources.

This lab session comes in two variants, depending on the capacities of your machine: if you are able to install a Virtualbox x86 64bit virtual machine (which should be possible on all laptops except those running an ARM-based CPU) and if you have enough disk space (a few dozen gigabytes) and RAM (at least 4 GB) to run it, follow the instructions in the first part; otherwise, follow the instructions in the second part (which does roughly the same thing, but without the MapReduce part, and in a fully local environment).

Variant 1: Using a Hadoop VM

1. Retrieve the VirtualBox virtual machine (also available here if necessary) that contains all required software, and install and launch it on your machine (if you have enough RAM on your laptop, it is worth allocating more than the default 2 GB to the VM). This is a Ubuntu Linux installation, the main user is named `hadoop` with password `hadoop` and can launch commands as other users or as `root` with `sudo`. It is possible to use the VM graphically, or to connect to it by SSH by launching a SSH connection to localhost, port 2222.
2. The Simple English Wikipedia dump, also available online, is already on the VM in the `data` subdirectory. Keep it compressed. You can examine its content with the `bzless` command.
3. Using Python's `xml.sax` standard modle, write a program that reads the Wikipedia file as is (using the `bz2` module to uncompress it on the fly) and parse it as an XML file using the SAX API. The idea of SAX is to handle parsing events (start and end of an element, character data, etc.) as they occur in a linear pass over the document, without storing the whole content in memory. See this tutorial on SAX. Your program should output one line per Wikipedia article, with first the title of the article, then a tabulation, then the content of the article – of course, this means that all newlines occurring in article contents need to be replaced by regular spaces. Do not process articles that are not in the main Wikipedia namespace (i.e., those whose names contain a colon). You can also do some data cleaning to get rid (when easy) of templates, complex links, etc.
4. Put the resulting TSV file onto HDFS; the `hdfs dfs -`*`command`* command can be used to manipulate data on the distributed storage; see the documentation of available commands.
5. Write the inverted index construction as a MapReduce program, formed of mapper and reducer scripts written in Python. As seen in class, the text needs to be tokenized; stop words need to be removed (a stop words file is provided in the `data/` directory); the tokens need to be stemmed (using the pystemmer module); finally, the tf-idf score of each token within a document needs to be computed.
6. Try the mapper and reducer locally on a sample of the data – when they work, launch the MapReduce job using Hadoop. Inspect the result.
7. Now program the construction of an inverted index using Spark instead of MapReduce. To execute a Python Spark program, launch: `spark-submit monprogramme.py`. You can also experiment with Spark in a dedicated Spark shell by running `pyspark`. A Spark context `sc` is obtained with: `sc=SparkContext()`. Refer to the documentation of Spark for the methods available on a Spark context or to an RDD.

Variant 2: Using your local machine

1. Install the `pystemmer` and `pyspark` Python modules on your local machine (e.g., using `pip`).
2. Download the latest version of the Simple English Wikipedia dump (you need the main file, containing **Articles, templates, media/file descriptions, and primary meta-pages**). Keep it compressed. On Linux, you can examine its content while keeping it compressed with the `bzless` command.
3. Using Python's `xml.sax` standard modle, write a program that reads the Wikipedia file as is (using the `bz2` module to uncompress it on the fly) and parse it as an XML file using the SAX API. The idea of SAX is to handle parsing events (start and end of an element, character data, etc.) as they occur in a linear pass over the document, without storing the whole content in memory. See this tutorial on SAX. Your program should output one line per Wikipedia article, with first the title of the article, then a tabulation, then the content of the article – of course, this means that all newlines occurring in article contents need to be replaced by regular spaces. Do not process articles that are not in the main Wikipedia namespace (i.e., those whose names contain a colon). You can also do some data cleaning to get rid (when easy) of templates, complex links, etc.
4. Write the inverted index construction as a MapReduce program, formed of mapper and reducer scripts written in Python. As seen in class, the text needs to be tokenized; stop words need to be removed (a stop words file is provided on Moodle); the tokens need to be stemmed (using the pystemmer module); finally, the tf-idf score of each token within a document needs to be computed.
5. Try the mapper and reducer locally on a sample of the data – unfortunately, as you probably don't have a Hadoop system installed, you cannot try the full MapReduce program.
6. Now program the construction of an inverted index using Spark instead of MapReduce. To execute a Python Spark program, launch: `spark-submit monprogramme.py`. You can also experiment with Spark in a dedicated Spark shell by running `pyspark`. A Spark context `sc` is obtained with: `sc=SparkContext()`. Refer to the documentation of Spark for the methods available on a Spark context or to an RDD.

Modifié le: mardi 5 décembre 2023, 11:11