Dask Processing and Analytics for Large Datasets

Alina Lazar
Department of CSIS
Youngstown State University
Youngstown, OH
alazar@ysu.edu

***Abstract*—This paper describes the assignment titled "Dask Analytics" that is used for the assessment of student learning as part of a graduate data science and data mining course. However, the assignment could be easily adapted for upper division undergraduate courses. For this assignment students are required to read, process and answer queries using a large dataset that does not fit in the RAM memory of a commodity laptop. Using the Python framework Dask, which extends a small set of Pandas's operations, students can become familiar with parallel and distributed processing. In addition, the assignment teaches students about the basics operations implemented in Dask in a very interesting and applied way, as well as operations and algorithms that are harder to parallelize.**

I. Motivation for the Assignment

Recently, the Pandas library has become one of the most popular and favourite data science tools for the Python programming language to perform exploratory data analysis. Not only that, but Pandas is usually used for data preprocessing and transformation, operations required before using any algorithms part of the Scikit-Learn machine learning library [1]. Pandas and Scikit-Learn work great for tabular datasets that fit in memory (e.g. the size of the dataset is less than 1 GB), with no concern about the performance. However, for datasets between 1 GB and 100 GB that do not completely fit in the RAM of a commodity computer other solutions are needed. Usually, students do not have access to parallel or distributed computing. This type of problem provides a good appropriate challenge to stimulate students' learning. One possible approach to this problem is to divide the large dataset into chunks and load the chunks into multiple Pandas DataFrames. The idea is to load smaller chunks of data into memory, one at the time, process or analyze it, store intermediate results to disk, and aggregate the results in the end. Dividing the data into chunks, coordinating the writing and re-reading of intermediate results to and from disk are tedious and error prone tasks. However, existing good quality, high-level libraries, such as Dask [2] or Ray [3] not only implement the tasks described above, but their APIs are very similar to the Pandas API. These libraries make possible for students to quickly grasp these concepts and to write short scripts for processing large files.

For this assignment we are using Dask, which is a light weighted framework that works well on a single machine by using all the cores to process larger-than-memory datasets. Dask also scales up resiliently and elastically on clusters with hundreds of nodes for solving even larger problems. Dask focuses on parallel analytics, providing Dask-specific modules to be used in place of Numpy Arrays or Pandas Dataframes to facilitate parallel execution. The dask.dataframe module implements a blocked parallel DataFrame object that mimics a large subset of the Pandas DataFrame. To perform any operation on a Dask DataFrame, many Pandas operations on the smaller Pandas DataFrames are executed.

Under the Anconda platform, Dask can be installed with a single conda install command, under a separate environment. Otherwise Dask can also be installed using the pip command.

II. Assignment

The goal of this assignment is to use Dask DataFrames to read, preprocess, aggregate and summarize a large given dataset that does not fit in the RAM of a commodity laptop.

The dataset we used for this assignment comes from the Stack Overflow website [4], one of the most popular question and answer sites. Over the years the website has slowly evolved into a large free repository of knowledge. Currently, the site receives around 8000 questions per day, and includes over 16 million questions, 24 million answers and 66 million comments all available to download in a data dump collection. The total size of the dataset in compressed format is just below 45 GB, with the most important file (Posts.xml) containing mainly the all the questions and answers at around 14 GB.

For the first part of the assignment students are asked to read the Posts.xml file, upload it in a Dask DataFrame and answer the following questions:

- How many rows and columns are in the dataset?
- How many questions are in the dataset?
- How many answers are in the dataset?
- What is the average length of the title and body for all the questions in the dataset?
- What day of the week has the most questions submitted on average?
- How many closed versus open questions are there?
- Find how many unanswered questions are in the dataset.

For the second part of the assignment, students are required to create a smaller subsample (100 MB) of the dataset and save it in CSV or Apache Parquet format. These files are going to be used in subsequent assignments as input for machine learning tasks such as classification and clustering.

Most of this assignment can be solved using Dask DataFrame operations such as: value counts, row-wise se-

Fig. 1. Part of the Dask dashboard shows the processing progress and how the problem is divided in smaller chunks.

lections, group by aggregations and date time resampling. Debugging and profiling code that runs in parallel can be challenging, but Dask has a diagnostic visual dashboard, partially showed in figure 1, that provides insights on performance and progress. This dashboard is very useful for the students. Subsets extracted from the Posts file of Stack Overflow have been used before [5] to build classification models for predicting closed questions. Another supervised problem is to predict a question's tags [6].

This challenging assignment introduces students to parallel and distributed computing in an easy, unintimidating way. Students should be familiar with Pandas DataFrames and Scikit-Learn by the time in the semester when they have to solve this assignment. The skills acquired while solving this assignment will be useful for solving other course assignments and other student's projects.

REFERENCES

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[2] M. Rocklin, "Dask: Parallel computation with blocked algorithms and task scheduling," in *Proceedings of the 14th Python in Science Conference*, no. 130-136, 2015.

[3] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan *et al.*, "Ray: A distributed framework for emerging {AI} applications," in *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, 2018, pp. 561–577.

[4] A. Ahmad, "A survey on mining stack overflow: question and answering (Q&A) community," *Data Technologies and Applications*, vol. 52, no. 2, pp. 190–247, Jan. 2018.

[5] G. E. Lezina and A. M. Kuznetsov, "Predict closed questions on StackOverflow."

[6] C. Stanley and M. D. Byrne, "Predicting tags for stackoverflow posts," in *Proceedings of ICCM*, 2013.