BigData Project

[BFS with MapReduce and Wikipedia Crawler]

Ekimov Victor
University of Trento
MAT.174212
ekimov.victor@gmail.com

Nyaika David
University of Trento
MAT.174258
davie2086@gmail.com

ABSTRACT

Actual paper is a **"Big Data"** course project at **University of Trento, Italy**. First part is devoted to the transformation of a well-known serial **Breadth-first search** algorithm into a parallel version and their head-to-head comparison. Second part is targeted at building a **web crawler** with a single purpose: data-mine the entire wikipedia onto a single laptop using parallel computation and NoSQL database. Paper contains references to the source code freely available on **github** accounts as well as instructions on how to run them and what technologies have been used.

1. PART A: BFS WITH MAPREDUCE

In this section we will start from the general introduction into graph data structure and inner implementation, followed by exploring the breadth-first search, one of the most well-known algorithm for graph processing. Serial version of the algorithm is provided by **"Algorithms, 4th Edition" by Robert Sedgewick and Kevin Wayne** [1], while parallel version will be backed up by our custom implementation on **GitHub: BFS-with-MapReduce**. Section includes the benchmark and algorithms comparison to answer the question: does the parallel performs better?

1.1 Introduction to Graphs

Pairwise connections carry an important role in representing many real world relations and abstract structures. Whether a social network, maps, web site cross links or software dependency libraries, having a sound model to put object together in a meaningful way helps discovering new facts about all of them at once, rather than individual piece at a time. Mathematics defines a graph theory for study such phenomena.

Definition 1. A *graph* is a set of *vertices* and a collection of *edges* that each connect a pair of vertices. [1]

A graph does not necessarily imply that any vertex is reachable from any other vertex, thus leaving a room for situations where graph becomes a set of connected components. In such way, two vertices might have nothing in common.

Definition 2. A *graph* is *connected* if there is a part from every vertex to every other vertex. [1]

The way two vertices are connected by an edge may fall into categories of:

- **self-loop** vertex is directly connected to itself
- **sing edge** vertex is directly connected to another vertex by exactly one edge
- **parallel edges** vertex is directly connected to another vertex by more than one edge

Definition 3. A *path* in a *graph* is a sequence of vertices connected by edges. [1]

Edges may be of two forms:

- **directed** specifying the direction from one vertex to another
- **undirected** where no direction is implied (equivalent to bi-directional edge)

1.2 Serial Breadth-first search

1.3 Parallel Breadth-first search

1.4 Implementation of BFS with MapReduce

1.5 Performance comparison

1.6 Conclusion

2. PART B: TECHNOLOGICAL

The technological section of the project will focus on crawling large amounts of web pages from wikipedia as a datasource.

HTML pages will be downloaded in parallel using spark. The downloaded data will be aggregated using the reducer and later on filtering will be applied to transform the data into standard UTF-8 encoded strings before storing them into MongoDB data store as the data repository.

Java 7, while leveraging the **crawler4J API (v3.5)**, shall be used to implement the crawler. **Apache Spark 1.3.1** will be used for parallel downloading and **MongoDB 3.0.3** for data storing.

2.1 Conclusions

Outcome and github to the code here!

3. REFERENCES

[1] R. Sedgewick and K. Wayne. Algorithms, 4th edition. 4:518–566, March 2011.