BigData Systems Project

TeamName: MatrixTeam

Members:

<Mohamad> <Kaser>, <Kaser@students.uni-marburg.de>
<Parmida> <Talebi>, < Talebip@students.uni-marburg.de >
<Anusha Nepal> <Upadhaya>, <Nepalupa@students.uni-marburg.de>

GENERAL IDEA

Our task is to compare two columns together to find IND, this is a brief explanation of our solution for this task.

DEPENDENCY MINER

- First we stored all data in a List<List<Set<String>>> called “alldata”. In this list, every Set is a column in a table, so basically, we have a list of all tables.
- We create indexes for every column that we have and we store them in a list(called “taskkeys”), so we can use these indexes when we want to pass the task to the worker. Also we create a map called “taskresults” and store every task key and its result in this map(at first all the results are null).
- We have a function called “assignTasktoWorker”, in this function, we are using the indexes (task keys) to make a new task. So, we store two columns in a list called “taskcolumns” which contains dependent and reference columns, and then we pass it to the dependency worker as its task. Also, we add this worker and its task to the busy worker Hashmap.
- If the worker replied with the true value as the result, it means that we found an IND.
- In the last step, we check if we did all of the tasks(by checking the “taskresults” map that we have), so we are done with the discovery.

DEPENDENCY WORKER

- Here the task message contains: “taskcolumns” , “tasktables”.
- The “taskcolumns” is a list<Set<String>> which contains the dependent column and reference column. The “tasktables” is the keys of the tables and columns that we are comparing(these keys are just for tracking reason)
- We stored the reference column and the dependent column in two lists, so we can check the IND in the next step.
- Now by using “parallelstream.allMatch” we check the IND for these columns and tell the result to the miner. parallelStream() method allows the comparison to be performed concurrently on multiple threads, potentially improving the performance for large sets of columns.