ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

National Symposium On Emerging Trends In Computing & Informatics, NSETCI 2016, 12th July 2016, Rajagiri School of Engineering & Technology, Cochin, India.

A Progressive Approach for Duplicate Detection with Map Reduce

***Manju V J and Chinchu Krishna S.**
Department of Information Technology, RSET, India.

Manuscript Info

Key words:
MapReduce,
Duplicate Detection,
Progressiveness.

Abstract

Duplicate detection is a crucial step for data quality and data integration. Cloud infrastructure is a popular paradigm that enables efficient parallel processing of data and computational intensive tasks such as duplicate detection on a very large datasets. Different cloud computing applications make use of a programming model called MapReduce that supports parallel execution of data-intensive computing tasks in cluster environments with up to thousands of nodes. Most of the duplicate pairs in the duplicate detection process can be identified as early as possible by using a method called Progressive Sorted Neighbourhood. To reduce the typically high execution times, this paper investigate how progressive sorted neighborhood for data intensive duplicate detection can be realized in a cloud infrastructure using MapReduce. This paper mainly focuses on the use of MapReduce programming model aiming at a highly efficient duplicate detection implementation for a very large datasets.

Introduction

Data is one of the most important resources in a company. It represents the whole business knowledge about products, customers, suppliers and transactions and forms the main source of information. Since data changes in the course of daily business, faults occur and information invalidate. Steve Sarsfield emphasizes the growing of a company and its data volume as a major reason for quality issues. The insertion of new data items always bears the risk of producing errors and existing data items might become obsolete in the ever changing business environment. Redundant data management and the integration of different data sources may as well result in faulty data pools. Ignoring these errors forces a company to base its strategic decisions on incorrect information. The consequences are avoidable costs and competitive disadvantages.

Duplicate Detection, as an integral part of data cleansing, focuses on finding different representations of the same real world entity by comparing multiple records. A duplicate detection process is very costly due to extremely large data sets and compute-intensive record comparisons. At the same time, it may be very important to run duplicate detection within a limited amount of time. Many studies have evaluated several state-of-the-art duplicate detection methods. A conservative duplicate detection approach collects all duplicates internally and emits them in the end. On the contrary, incremental method already report intermediate results at execution time. Hence, they deliver first duplicates right from the start. In this paper we focus on progressive approach that try to report most matches as early as possible. To achieve this, progressive duplicate detection approach (Papenbrock et al (2015)) estimate the similarity of the comparison candidates to compare most promising record pairs first. Progressive Sorted Neighborhood method is a progressive duplicate detection that uses a blocking technique necessary to reduce the

number of entity comparison while maintaining the match quality. This is achieved by semantically partitioning the input data into blocks of similar records and restricting entity resolution to entities of the same block. It sorts all entities using an appropriate blocking key and only compares entities within a predefined distance window w. The window is iteratively varied, starting with a small window of size two that quickly finds the most promising records, until it reaches a maximum window size.

MapReduce (MR) is a popular programming model for parallel processing on cloud infrastructures with up to thousands of nodes . The availability of MR distributions such as Hadoop makes it attractive to investigate its use for the efficient parallelization of data-intensive tasks. In this paper we investigate the use of MapReduce for the parallel execution of Progressive Sorted Neighborhood blocking and entity resolution. By combining the use of blocking and parallel processing, a highly efficient duplicate detection implementation for very large datasets can be achieved.

Related Works:-

Duplicate detection , also known as entity resolution, record linkage, merge/ purge, deduplication, reference reconciliation and object identification, describes the process of finding same real world entities in a collection of records(Elmagarmid et al(2007)). The performance bottleneck for duplicate detection is typically the expensive attribute comparison with similarity measures between record pairs . The standard (naive) approach to find matches in n input entities is to apply matching techniques on the Cartesian product of input entities. This can constitute a complexity of $O(n^2)$. To avoid these prohibitively expensive comparisons of all pairs of records, a common technique is to carefully partition the records into smaller subsets and search for duplicates only within these partitions. Sorted neighborhood (SN) is one of the most popular blocking approaches that can reduce the complexity to O(n.w). Kolb et al(2011) described about the Sorted Neighborhood approach for entity matching which can be implemented using MapReduce programming model. Comparing with the previous approaches for parallel entity resolution, the authors do not investigate how progressive approach can be realized with MapReduce. The work proposed here investigate Progressive approach of SNM with MapReduce that can increase the efficiency of detecting duplicates by changing the order in which records are selected for comparison so that most promising records are matched earlier than less promising pairs. Learning-based approaches are another method for entity resolution that show high effectiveness at the expense of poor efficiency. To reduce the typically high execution times, Hanna Köpcke et al (2011) investigated how learning- based entity resolution can be realized in a cloud infrastructure using MapReduce. But these approache for entity resolution results in a quadratic complexity of $O(n^2)$ results in intolerable execution times for large datasets. Compared to this, the proposed method uses blocking technique that can reduce the complexity by reducing the number of record comparisons.

Proposed Work:-

The proposed work investigate the use of MapReduce for the parallel execution of Progressive Sorted Neighborhood method of duplicate detection. By combining the use of blocking and parallel processing , a highly efficient duplicate detection implementation for very large datasets can be achieved. This work mainly focuses on the problem of duplicate detection within one source. The input data source D={ri} consist of a finite set of records ri . The goal is to find all pairs of records $R=\{(ri , rj)|\ ri , rj \epsilon D \}$ that are regarded as duplicates.

The overall duplicate detection method consist of a blocking strategy and a matching strategy. Blocking strategy semantically divides a data source D into blocks bi , with $D = \cup bi$. The partitioning into blocks is usually done with the help of blocking or sorting keys based on the record's attribute values. Blocking keys can be selected as a single attribute or a combination of several attributes. The matching strategy finds pairs of matching records of the same block. It involves pairwise similarity computation of records to calculate the degree of similarity. By implementing blocking within the map function and by implementing matching within the reduce function, the overall duplicate detection method can be realized with MapReduce .

Progressive Sorted Neighborhood is a duplicate detection method that identifies most duplicate pairs early in the detection process. Progressive Sorted Neighborhood Method(PSNM) is a method using windowing technique that can significantly increase the efficiency of finding duplicates if the execution time is limited. Here the order for the comparisons of records are executed in a way to match most promising record pairs earlier than less promising record pairs (Whang et al (2011)). This enables the method to report most duplicate pair matches earlier. In Progressive Sorted neighbourhood, a sorting key K is determined for each of n records. Typically, the prefixes of a few attributes of each record are concatenated to form a sorting key. The map phase first determines the sorting key for each record. The MapReduce framework groups records with the same blocking key to blocks. A block may contain different

keys but all values with the same key are in the same block. The blocks are then redistributed such that all entities with the same blocking key are sent to the same reducer. The reduce step then matches the records within one block. The matching step involves the use of window w which is moved over the records and only records within the window are compared. The window is iteratively varied, starting with a small window of size two that quickly finds the most promising records, until it reaches a maximum window size. The reduce outputs can finally be merged to achieve the overall match result. Thus, the map function identifies the blocking key for each input entity independently. The map output is then distributed to multiple reducers that implement the sliding window approach for each reduce partition.

The figure below illustrate an example for n=9 records, a-I, of an input data source D using 3 mappers and 2 reducers. First, the data source is divided into 3 partitions and assigns one partition to each mappers. Then, the individual mappers read their local data in parallel and determine a blocking or sorting key value K for each of the input entities. For example, record c has blocking key value 3. Afterwards all entities are dynamically redistributed by a partition function such that all entities with the same blocking key value are sent to the same reducer (node). In the example of Figure, blocking key values 1 and 2 are assigned to the first reducer whereas key 3 is assigned to the second reducer. Then the records in each block are sorted by their blocking keys. The reduce step then matches the records by using window of size w which is moved over the records. Starting with the maximum window of size 2 , the window w slides over the records until it reaches the maximum window size. In the example given below , the maximum window size is 3. By using window w=2, it makes 5 pair comparisons and using w=3 it makes 4 pair comparisons. In this way, promising close neighbours are selected first and less promising far-away neighbours later on, thereby finding most matches as early as possible. The reduce output is then merged to get the final match results.

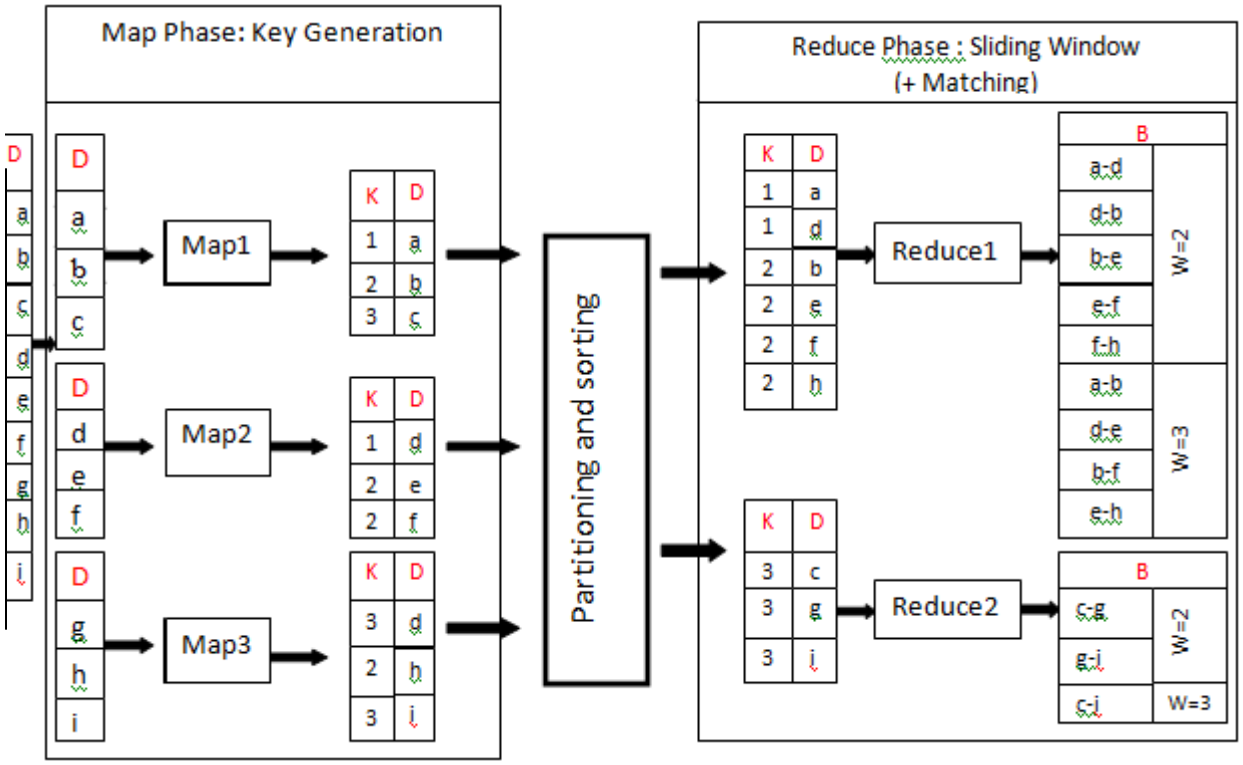

Fig: Example execution of Progressive sorted neighborhood with maximum window size w=3

Conclusion:-

Duplicate Detection also known as Entity Resolution, Entity matching, reference reconciliation or record linkage and is a critically important task for data cleaning and data integration. One can think of it, as the task of finding entities matching to the same entity in the real world. These entities can belong to a single source of data, or distributed data-sources. The standard (naive) approach to find matches in n input entities is to apply matching techniques on the Cartesian product of input entities. This can constitute a complexity of O(n2). Sorted neigh-

borhood (SN) is one of the most popular blocking approaches that can reduce the complexity to O(n.w). Progressive Sorted Neighborhood Method (PSNM) is more efficient than SN method that can significantly increase the efficiency of finding duplicates if the execution time is limited. It identifies most duplicate pairs early in the detection process. Instead of reducing the overall time needed to finish the entire process, progressive approaches try to reduce the average time after which a duplicate is found. MapReduce has become a popular programming model for efficiently processing data and computationally intensive application tasks. It supports parallel data-intensive computing in cluster environments with up to thousands of nodes. Duplicate Detection (also known as Entity resolution) is such a data-intensive and performance-critical task that can likely benefit from MapReduce programming model. The focus of this paper is to use MapReduce Programming Model for combining parallel processing and progressive Sorted Neighborhood Method to achieve a highly efficient duplicate detection implementation for very large dataset. By using Progressive approach of SNM, the efficiency for reporting duplicates can be increased compared to using simple SN method for reporting duplicates. In this work, the map function identifies the blocking key for each input entity independently. The map output is then distributed to multiple reducers that implement the sliding window approach for each reduce partition.

References:-

1. **T. Papenbrock, A. Heise, and F. Naumann.(2015)** Progressive Duplicate Detection, IEEE Trans. Knowl. Data Eng., vol. 27, no. 5, pp. 1316-1329, May.
2. **S. E. Whang, D. Marmaros, and H. Garcia-Molina (2012)**, Pay-as-you-go entity resolution, IEEE Trans. Knowl. Data Eng., vol. 25, no. 5, pp. 1111-1124, May.
3. **Kolb, L., Thor, A., Rahm, E (2011),** Parallel sorted neighborhood blocking with MapReduce., In: BTW, pp. 45-64.
4. **K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios(2007)**, Duplicate record detection: A survey, IEEE Trans. Knowl. Data Eng., vol. 19, no. 1, pp. 1-16, Jan.
5. **Kolb, Lars, Hanna Köpcke, Andreas Thor, Erhard Rahm (2011).** Learning-based entity resolution with MapReduce. Proceedings of the third international workshop on Cloud data management. ACM.