ISSN:2231-0711

A Survey On Efficient Service Recommendation On Large Data Clusters

Anumol Johnson
Computer Science and Engineering
Sahrdaya College of Engineering and Technology
Kodakara, India

Asst. Prof. **Divya R**
Computer Science and Engineering
Sahrdaya College of Engineering and Technology
Kodakara, India

Abstract- Recommendation system is an information filtering technique, which provides users with information, which he/she may be interested in. It helps in addressing the information overload problem by retrieving the information desired by the user based on his or similar users' preferences and interests. Service recommender systems proves to be a valuable tool for providing appropriate recommendations to users. Most of existing service recommender systems present the same ratings and rankings of services to different users without considering diverse users preferences, and therefore fails to meet users personalized requirements. A Keyword-Aware Service Recommendation method, named KASR has been prposed to address the above challenges. It aims at presenting a personalized service recommendation list and recommending the most appropriate services to the users effectively. Specifically, keywords are used to indicate users preferences, and a user-based Collaborative Filtering algorithm is adopted to generate appropriate recommendations. To improve its scalability and efficiency in big data environment, KASR is implemented on Hadoop, a widely-adopted distributed computing platform using the MapReduce parallel processing paradigm.

Keywords- Recommendation System, Hadoop, MapReduce, Filtering Algorithms

I. INTRODUCTION

The increase in the number of services over the internet has inundated service users with many choices. For instance, Netflix.com has over 17,000 movies in its selection, and Amazon. com has over 410,000 titles in its Kindle store alone. In order to reduce the number of choices users can decide on, recommendation systems are necessary. Recommendation systems are attracting lots of attention because they provide users with prior knowledge of candidate choices to deal with information overload on the Web.

Similar to most big data applications, the big data tendancy also poses heavy impacts on service recommender systems. With the growing number of alternative services, effectively recommending services that users preferred has become an important research issue. Service recommender systems have been shown as valuable tools to help users deal with services overload and provide appropriate recommendations to them. Examples of such practical applications include CDs, books, webpages and various other products now use recommender systems. Over the last decade, there has been much research done both in industry and academia on developing new approaches for service recommender systems.

Collaborative filtering (CF) is one of the widely used service recommendation techniques that bases its recommendations on the ratings or behavior of other users in the system . Intuitively, it assumes that, if users agree about the quality or relevance of some service items, then they will likely agree about other service items as well. Existing memory-based CF techniques accomplish this by computing the similarity between users or service items using nonfunctional attribute values obtained at service invocation. However using nonfunctional attribute values of invoked services alone gives inaccurate similarity measure. This is because, the invoked services are typically based on different user personalized preferences on those nonfunctional attributes.

Hadoop-MapReduce has become a powerful Computation Model addresses to these problems. Hadoop HDFS became more popular amongst all the Big Data tools as it is open source with exible scalability, less total cost of ownership and allows data stores of any form without the need to have data types or schemas defined. Hadoop MapReduce is a programming model and software framework for writing applications that rapidly process vast amounts of data in parallel on large clusters of compute nodes. Map reduce is a software frame work introduced by Google in 2004 to support distributed computing on large data sets on clusters of computers. The original MapReduce implementation by Google, as well as its open-source counterpart, Hadoop, is aimed for parallelizing computing in large clusters of commodity machines. MapReduce model advantage is the easy scaling of data processing over multiple computing nodes.

II. LITERATURE SURVEY

A "recommender system" is a fully functional software system that applies at least one implementation to make recommendations. In addition, recommender systems feature several other components, such as a user interface, a corpus of recommendation candidates, and an operator that owns/runs the system. Some recommender systems also use two or more recommendation approaches: CiteULike, a service for discovering and managing scholarly references, lets their users choose between two approaches.

The First recommender system was developed by Goldberg, Nichols, OkiTerry in 1992. Tapestry was an electronic messaging system that allowed users to either rate messages ("good" or "bad") Recommender system as defined by M. Deshpande and G. Karypis: A personalized information filltering technology used to either predict whether a particular user will like a particular item (prediction problem) or to identify a set of N items that will be of interest to a certain User. Recommender systems form or work from a specific type of information filtering system technique that attempts to recommend information items (movies, TV program/show/episode, video on demand, music, books, news, images, web pages, scientific literature etc.) or social elements (e.g. people, events or groups) that are likely to be of interest to the user. Typically, a recommender system compares a user profile to some reference characteristics, and seeks to predict the ’rating’ or ’preference’ that a user would give to an item they had not yet considered. These characteristics may be from the information item (the content-based approach) or the user’s social environment (the collaborative filtering). The recommender system apply data mining techniques and prediction algorithms to predict users interest on information ,product and services user .

Recommender systems apply techniques and methodologies from another neighboring areas such as Human computer interaction (HCI) or Information Retrieval(IR). However, most of these systems bear in their core an algorithm that can be understand as a particular instance of a data mining (DM) technique. The process of data mining consists of 3 steps carried out in succession: Data Preprocessing, Data Analysis and Result Interpretation. Examples of recommender system areamazon.com,Reel.com,eBay,Levis,Moviefinder.com. Recommender systems typically produce a list of recommendations in one of two ways through collaborative or content-based filtering. Collaborative filtering approaches building a model from a user’s past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. These approaches are often combined.

Collaborative filtering (CF) is one of the widely used service recommendation techniques that bases its recommendations on the ratings or behavior of other users in the system . Intuitively, it assumes that, if users agree about the quality or relevance of some service items, then they will likely agree about other service items as well. Existing memory-based CF techniques accomplish this by computing the similarity between users or service items using nonfunctional attribute values obtained at service invocation. However using nonfunctional attribute values of invoked services alone gives inaccurate similarity measure. This is because, the invoked services are typically based on different user personalized preferences on those nonfunctional attributes. The nonfunctional attribute values observed by users during service invocation may not necessarily represent their satisfaction for that service. For this reason, disregarding the personalized preferences of users in similarity computation creates a gap between users nonfunctional attribute value and their satisfaction. Users personalized preferences ensures that the nonfunctional attribute closely aligns with their satisfaction, bridging that gap and resulting in similarity values that accurately depicts the similar relationship between two users. Intuitively, if a nonfunctional attribute value used in similarity computation fails to satisfy a user’s personalized preference it in turn produces similarity results that are inaccurate. Thus, to accurately recommend services, which are personalized to users, it is necessary for recommendation systems to incorporate users personalized preferences on nonfunctional attributes when recommending services to an active user.

One growing area of research in the area of recommender systems is mobile recommender systems. With the increasing ubiquity of internet-accessing smart phones, it is now possible to offer personalized, context-sensitive recommendations. This is a particularly difficult area of research as mobile data is more complex than recommender systems often have to deal with (it is heterogeneous, noisy, requires spatial and temporal auto-correlation, and has validation and generality problems). Additionally, mobile recommender systems suffer from a transplantation problem - recommendations may not apply in all regions (for instance, it would be unwise to recommend a recipe in an area where all of the ingredients may not be available). One example of a mobile recommender system is one that offers potentially profitable driving routes for taxi drivers in a city. This system takes as input data in the form of GPS traces of the routes that taxi drivers took while working, which include location (latitude and longitude), time stamps, and operational status (with or without passengers). It then recommends a list of pickup points along a route that will lead to optimal occupancy times and profits. This type of system is obviously location-dependent, and as it must operate on a handheld or embedded device, the computation and energy requirements must remain low.

Mobile recommendation systems have also been successfully built using the Web of Data as a source for structured information. A good example of such system is SMARTMUSEUM The system uses semantic modelling, information retrieval and machine learning techniques in order to recommend contents matching user’s interest, even when the evidence of user’s interests is initially vague and based on heterogeneous information.[

III. METHOD

A. TF-IDF

TF-IDF, short for Term Frequency-Inverse Document Frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general. Variations of the tf-idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document’s relevance given a user query. tf-idf can be successfully used for stopwords filtering in various subject fields including text summarization and classification.

Term frequency: Suppose we have a set of English text documents and wish to determine which document is most relevant to the query "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document and sum them all together; the number of times a term occurs in a document is called its term frequency. Inverse document frequency: However, because the term "the" is so common, this will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less common words "brown" and "cow". Hence an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely.

B. MapReduce

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key.

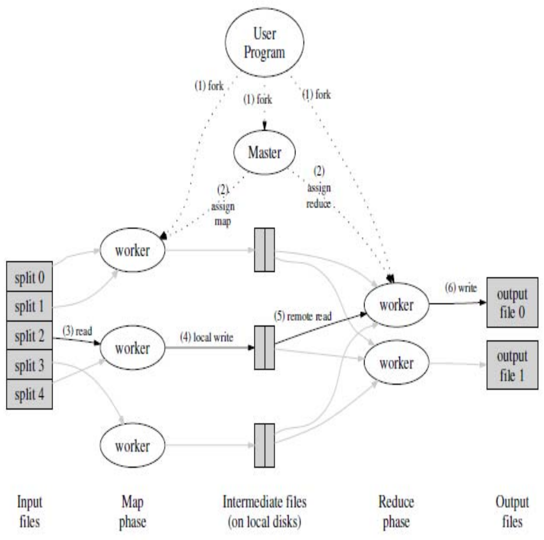

Figure 3.1: Execution Overview

Figure 3.1 shows the overall flow of a MapReduce operation in our implementation. When the user program calls the MapReduce function, the following sequence of actions occurs (the numbered labels in Figure 3.1 correspond to the numbers in the list below): The MapReduce library in the user program first splits the input files into M pieces of typically 16 megabytes to 64 megabytes (MB) per piece (controllable by the user via an optional parameter). It then starts up many copies of the program on a cluster of machines. One of the copies of the program is special - the master. The rest are workers that are assigned work by the master. There are M map tasks and R reduce tasks to assign. The master picks idle workers and assigns each one a map task or a reduce task. A worker who is assigned a map task reads the contents of the corresponding input split. It parses key/value pairs out of the input data and passes each pair to the user-defined Map function. The intermediate key/value pairs produced by the Map function are buffered in memory. Periodically, the buffered pairs are written to local disk, partitioned into R regions by the partitioning function. The locations of these buffered pairs on the local disk are passed back to the master, who is responsible for forwarding these locations to the reduce workers. When a reduce worker is notified by the master about these locations, it uses remote procedure alls to read the buffered data from the local disks of the map workers. When a reduce worker has read all intermediate data, it sorts it by the intermediate keys so that all occurrences of the same key are grouped together.

The sorting is needed because typically many different keys map to the same reduce task. If the amount of intermediate data is too large to t in memory, an external sort is used. The reduce worker iterates over the sorted intermediate data and for each unique intermediate key encountered, it passes the key and the corresponding set of intermediate values to the user’s Reduce function. The output of the Reduce function is appended to a final output file for this reduce partition. When all map tasks and reduce tasks have been completed, the master wakes up the user program. At this point, the MapReduce call in the user program returns back to the user code. After successful completion, the output of the mapreduce execution is available in the R output files (one per reduce task, with file names as specified by the user). Typically, users do not need to combine these R output files into one file - they often pass these files as input to another MapReduce call, or use them from another distributed application that is able to deal with input that is partitioned into multiple files.

IV. RESULT

Recommender systems made a significant progress over the last decade when numerous content based, collaborative and hybrid methods were proposed and several “industrial-strength” systems have been developed. However, despite all these advances, the current generation of recommender systems surveyed in this paper still requires further improvements to make recommendation methods more effective in a broader range of applications. In this paper, we reviewed various limitations of the current recommendation methods and discussed possible extensions that can provide better recommendation capabilities. This paper presented the various techniques and algorithm to build the recommender system.

ACKNOWLEDGMENT

I express my deepest thanks to “Miss. Divya R” the mentor of the project for guiding and correcting various documents of mine with attention and care. She has taken the pain to go through the project and make necessary correction as and when needed. I also extend my heartfelt thanks to my family and well wishers.

REFERENCES

[1] Witten I. H. and Frank I. Data Mining, Morgan Kaufman Publishers, San Francisco, 2000.

[2] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, John Riedl, GroupLens: an open architecture for collaborative filtering of netnews, Proceedings of the 1994 ACM conference on Computer supported cooperative work, p.175-186, October 22-26, 1994, Chapel Hill, North Carolina, United States

[3] John S. Breese, David Heckerman and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence, pages 43-52, July 1998 [4] Deshpande, M., and Karypis, G. Item-based top-<i>n</i> recommendation algorithms. ACM Trans. Inf. Syst. 22, 1 (2004), 143-177.

[4] Deshpande, M., and Karypis, G. Item-based top-<i>n</i> recommendation algorithms. ACM Trans. Inf. Syst. 22, 1 (2004), 143-177.

[5] Breese, J., Heckerman, D., and Kadie, C., Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence, page 4352, 1998.

[6] Mukta kohar and Chhavi Rana, "Survey Paper on Recommendation System", International Journal of Computer Science and Information Technologies, Vol. 3 (2) , 2012,3460-3462

[7] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," IEEE Internet Computing, vol. 7, no. 1, pp. 76-80, Jan. 2003.

[8] Kenneth K. Fletcher and Xiaoqing (Frank) Liu Department of Compute Science Missouri University of Science and Technology Rolla, USA, "A Collaborative Filtering Method for Personalized Preference-based Service Recommendatio " , 2015 IEEE International Conference on Web Services.

[9] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Google, Inc.

[10] FengXu, "Service Recommendation with Case-based Reasoning", Proceedings 0[2015 IEEE 12th International Conference on Networking, Sensing and Control