블록체인과 합의 알고리즘

구민준, Giang-Truong Nguyen

디지털콘텐츠학회논문지 **Journal of Digital Contents Society Vol. 18, No. 8, pp. 1593-1601, Dec 2017**
(발표자 연락처: Giang-Truong Nguyen)

[ABSTRACT]

In a big data platform, association rule mining applications could bring some benefits. For instance, in a agricultural big data platform, the association rule mining application could recommend specific products for farmers to grow, which could increase income. The key process of the association rule mining is the frequent itemsets mining, which finds sets of products accompanying together frequently. Former researches about this issue, e.g. Apriori, are not satisfying enough because huge possible sets can cause memory to be overloaded. In order to deal with it, SON algorithm has been proposed, which divides the considered set into many smaller ones and handles them sequently. But in a single machine, SON algorithm cause heavy time consuming. In this paper, we present a method to find association rules in our Hadoop based big data platform, by parallelling SON algorithm. The entire process of association rule mining including pre-processing, SON algorithm based frequent itemset mining, and association rule finding is implemented on Hadoop based big data platform. Through the experiment with real dataset, it is conformed that the proposed method outperforms a brute force method.

[KEYWORDS] Big Data Platform, Association Rule Mining, Frequent Itemsets, SON Algorithm

1. 서 론

As being a farmer, choosing which products should be grown in a season is a very crucial question, because it is a factor for manipulating their income and productivity. Usually, each farmer would choose a main product which they think it would be most suitable for them. However, in order to get most benefits, they should also choose other types to grow with their chosen ones [1]. This is a big question because among many products, finding the suitable one requires a lot of conditions. Beside the best but too much time-consuming and experience-required scientific researches methods in Biology [1][2], one of the solutions is finding what other more-experience farmers often grow together. Following the footstep of those seniors, from some already chosen products, farmers could choose other suitable ones to combine for their season. This solution could be conducted by a well-known method called association rules mining. Mining the association rules has received attentions from many researchers for a long time. The original work of this technique is from a work of many retailers like Walmart, Amazon: giving a collection of transactions and their corresponding purchased items, mining the association rules finds items (consequent) which customers could possibly grab after taking (some) one(s) (antecedent). In order to implement this mining, the prior work should be handled is finding frequent itemsets [3]: trying to get all the set of items whose rate of appearance over the total number of transaction (a.k.a. their support) is larger than a given threshold. Therefore, coming back to the agricultural solution as mentioned above: considering a given collection of farms as transactions; and their corresponding grown products as items, mining the association rules could be used to suggest the farmers to choose which products should be grown with some given ones, which partly supports those people to improve their productivity and income. The problem of finding frequent itemsets could be solved by many algorithms up to now, but most of them is used for only the single-machine environment only. The first and original method called Brute-Force algorithm (BFA), which lists all the possible set of items; then finds their number of appearance in each transaction, is too “naive” because it uses too much time and memory. A-Priori algorithm [4] follows the idea of the BFA, however it finds the frequent itemsets in each levels of their increasing length, then in every level; it prunes the unsatisfied sets to reduce the number of computations for the next phase. Unfortunately, this algorithm could have some problems if the number of transactions and number of items are too large, causing the single-machine to be not able to handle loading all the data from hard dish to the main memory. Avoiding this problem, an improvement made by Savarese et al. called SON algorithm [5] has been proposed, which divides the collection of transactions into non-overlapping smaller ones to avoid the problem of overloading memory which a single machine could face. In this paper, we propose a method for mining the association rules, whose prior work is finding the frequent itemsets based on the idea of SON algorithm dividing the big collection of transactions for some machines, to process on them. However, our method is not conducted in a single machine, but in our implemented Hadoop [12] based distributed big data platform [8][9][10]. The overall process could be described in Fig. 1. First of all, from the collected agriculture data about many farms in Korea, by a pre-processing phase, the list of farms with their corresponding grown products would be gotten. Then by those so-called “transactions”, the frequent itemsets would be found. These two above phases are made with the support of MapReduce function [11]. Finally, on those results, another algorithm based on a former research is utilized to find the association rules about the grown agricultural products. The rest of this paper is organized as follow: section 2 overviews the background supporting for mining association rules; which includes A-Priori algorithm, SON algorithm and association rules mining method, section 3 describes our big data platform for storing and processing the Korean agricultural data. Section 4 mentions about our method to mine the association rules from the agriculture big data. Our evaluation is mentioned in section 5 and finally; section 6 is our conclusion.

2. 본 론

Given a set of items = { , …, } and the collection of transactions listing all the item which each user purchased at the same time, finding the frequent itemsets is trying to get all the set of items, whose rate of appearance over the total number of

transaction (their support) is larger than a given threshold. As being mentioned before, the most basic and naive method BFA lists all the possible set of items, then finds their supports in each of transaction; which will be a real catastrophe because the number of transactions and items are too big. A-Priori and SON algorithm are proposed to reduce the number of computation needed to be taken. 2-1 A-Priori algorithm Being quite similar to the basic algorithm, A-Priori algorithm also generates all the possible sets of items. However, it solves the problem with an enhancement: it prunes many unsatisfied sets for reducing the number of possible generated ones. A-Priori algorithm is described in Algorithm 1. As seen in the algorithm, when the number of transactions and possible items are too large, this algorithm could make the main memory to be overloaded. Considering a case when there are m items, so there could be possible transactions, causing the computation complexity is O( ). Consequently, an alternative algorithm should be proposed in order to reduce the data loaded into the main memory

2-2 SON algorithm SON is a kind of "divide to conquer" algorithm [6]: it first splits the overall collection of transactions into smaller non-overlapping parts. Afterwards, 2 passing of transactions collection are employed to solve the problem. In the first passing, a "local" frequent itemsets finding algorithm will be executed on each small part with a minimum supply threshold being much smaller than the overall one. The output of this phase is all of each smaller part's frequent itemset, which then will be aggravated to become the global frequent itemset candidates. After that, in order to get their number of appearances, a second pass will be executed on each of smaller part again. Finally, the total number of appearances of each global frequent candidate will be gotten and compared with the overall minimum threshold to get the final results. SON algorithm is described in Algorithm 2. 2-3 Mining the association rules After getting the frequent itemsets, association rules could be gotten. Following the article [4], this finding could be conducted by making a loop for every found frequent itemset, then generating every possible subset of them and calculating a so-called confidence. Consider a frequent itemset l whose an instance of subset is (a), then the confidence will be calculated by: If the confidence of any given rules is larger than a given threshold called minimum confidence, this rule will be regarded as an association rule. The problem of this method is it considers all the possible subsets of the gotten frequent itemset, which is not appropriate. In order to improve this method, the authors from [4] also suggested 2 methods for finding those rules. The first one is generating the subset of each frequent itemset by their length increasing in each level: if the minimum confidence condition is satisfied by a given subset, it would be added to the output list and its own generated subset with length is 1-lower than its will be added to the next level for checking. This work is done when the generated subset list is blank. The second method is quite different: initially, each frequent itemset will generate their subsets with length of 1; then if any of those subsets satisfies the condition, all of the possible subsets of the considered frequent itemset which contains this size-1 one will be added to the output list without considering. Afterwards, the next level would come with the length increases and without the subsets satisfies in the previous level. This implementation will be terminated until the length of generated subset reach the length of the considered itemset. The latter method seems to be more effective than the former one; however, it depends on the condition of the data: if the association rules with short length is more than the ones with long length, the former will be more suitable; and vice versa, the latter will be better to be applied if the rules with long length are more than the short ones. Employing the first method and considering a case: giving an itemset I = {ABCDE}, and in the third level, the subset {AB} will be considered at least twice, while it is needed to be considered only once. This problem should be solved more carefully.

3. 결 론

In this paper, we paralleled SON algorithm for finding frequent itemset on distributed environment, then the association rules mining is conducted. We implemented 2 phases of MapReduce to deal with the problem of large dataset, which a single machine could not handle well. In the future, we will focus on optimizing the algorithm by using Spark [7]. Moreover, we will also focus on other

conditions like the income, growing area to make the recommendation better.

REFERENCES:

[1] S. R. Lee, H. Berry, O. Temam, and M. Lipasti, “Performance improvement of WDM channels using inline dispersion management in transmission links with OPC placed at various position,” The Journal of Korea Navigation Institute, Vol. 14, No. 5, pp. 668-676, Oct. 2010.

[2] S. R. Lee, H. Berry, O. Temam, and M. Lipasti, “Performance improvement of WDM channels using inline dispersion management in transmission links with OPC placed at various position,” The Journal of Korea Navigation Institute, Vol. 14, No. 5, pp. 668-676, Oct. 2010.

[3] J. G. Proakis, Digital Communications, 4th ed. New York, NY: McGraw-Hill, 1993.

[4] J. L. Hennessy and D. A. Patterson, Instruction-level parallelism and its exploitation, in Computer Architecture: A Quantitative Approach, 4th ed. San Francisco, CA: Morgan Kaufmann Pub., ch. 2, pp. 66-153, 2007

[5] A. Hashmi, H. Berry, O. Temam, and M. Lipasti, “Automatic abstraction and fault tolerance in cortical microarchitectures,” in Proceeding of the 38th Annual International Symposium on Computer Architecture, New York: NY, pp. 1-10, 2011.

[6] B. Alavi, “Distance measurement error modeling for time- of-arrival based indoor geolocation”, Ph.D. dissertation, Worcester Polytechnic Institute, Worcester, MA, 2006.

[7] Y. Z. Ben, D. K. John, and Anthony, Tapestry: An infrastructure for fault-tolerant wide-area location and routing, University of California, Berkeley: CA, Technical Report CSD-01-1141, 2001.

[8] Malardalen Real-Time Research Center. The worst-case execution time (WCET) analysis project [Internet]. Available: http://www.mrtc.mdh.se/projects/wcet/.

[9] H. Nowakowska, M. Jasinski, P. S. Debicki and J. Mizeraczyk (2011, October). Numerical analysis and optimization of power coupling efficiency in waveguide-based microwave plasma source. IEEE Transactions on Plasma Science [Online]. 39(10), pp. 1935-1942. Available:http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6012536.

[10] 2018060001, 구민준, S. Arya, D. M. Mount, “Approximate Nearest Neighbor Queries in Fixed Dimensions”, Open Journal, No. 1, May. 2018.

[11] 2018060002, 변구훈, Mohammad Alfraheed, “An Approach for Features Matching Between Bilateral Images of Streo Vision System Applied for Automated Heterogeneous Platoon”, Open Journal, No. 2, May. 2018.

[12] 2018060003, 엄형근, J. P. Lee, “New Convergence Engineering Approach utilizing Automata and Arduino Technology”, Open Journal, No. 3, May. 2018.

CONTRIBUTORS:

[1] 2, 변구훈, 구글

[2] 3, 엄형근, 아마존

[3] 4, 차민준, 삼성 연구원