ISSN No:-2456-2165

Implementation of Anti-Crawler System Based on Spark

Yisong Wang
School of Computer Science and Technology
Shandong University of Technology
Zibo City, Shandong Province, China

Dongmei Zhang
School of Computer Science and Technology
Shandong University of Technology
Zibo City, Shandong Province, China

Abstract:- With the advent of the data age, the extraction and utilization of data information has become a huge challenge. The crawler algorithm is designed to obtain website information in batches. However, the use of some malicious crawlers has interfered with the normal business and operation of the website, such as website ticket grabbing behavior and so on. So anti-reptiles was proposed as a new research topic. From the initial front-end anti-crawler, an anti-crawler system based on big data emerged, which greatly improved the efficiency of anti-crawler. The purpose of this topic is to develop an anti-crawler system. After conducting certain research on anti-crawler strategies and technologies, it is determined that the system functions include data classification, data landing, data processing, data access, and ip sensitive representation. The goal is to meet the anti-crawler needs of ticketing websites, ensure normal business operations, and improve user satisfaction. The system adopts technologies such as spark, redis, kafka, nginx + lua, and uses idea as a development tool. After the development of the system is completed, it has undergone functional and performance tests. Its functions are simple and convenient, with good accuracy, and good scalability, which can meet development needs.

Keywords:- *anti-reptile; hadoop; spark; redis; kafka; nginx+lua.*

I. INTRODUCTION

During holidays such as the Spring Festival, there are a lot of trips, and it is often difficult to find a ticket. In particular, the use of malicious crawlers has interfered with the normal business and operation of the website. Therefore, anti-reptiles is proposed as a relatively new research topic. This subject is proposed based on this background. The purpose of the subject is to develop an anti-crawler system. The design and implementation of the system, based on the current relatively new anti-crawler technology based on big data, according to the anti-crawler strategy, designed a complete anti-crawler process for it, and conducted actual tests. The technologies involved are scala, shell, DW data warehouse, Spark, Kafka, Redis, Mysql and Hbase, etc.

II. DEMAND ANALYSIS

A. General Introduction of the System

This system is an anti-crawler system based on spark. It analyzes user behavior through logs collected from the front end, and then judges the user's tendency to crawl or not, processes user access, restricts access, and identifies sensitive signs.

The main function points implemented by the system include data cleaning, data cleaning, data landing, and data processing. Data cleaning is to filter the collected log information and unnecessary log information, and remove css, js, jsp and other redundant pages other than key pages. Data landing is the landing of the data to the storage engine, the required content is displayed on the front end, and the data processing is the ip sensitive identification of the content according to the threshold content.

B. Feasibility Analysis

For a complete visit, every time a user clicks and modifies the ticket website, the content will trigger the js embedded log, and a log will be generated in the background server. The log contains the request, request method, specified content code, The content of the request body, part of the information in the request header httpReferer, local address, specified access tool, specified time format content, service address, content, number of active connections, and two different direct lines of flightSelectDirect.html and flightSelectFro.html are accessed through the user in http. Connect or round-trip web pages format the data. For the round-trip direct content, it is an abstraction of the upper level of the data. After the data is cleaned up from the content of js, css, and html, the content of the program is adjusted for the number of active connections and the amount of ip data. Categorize, and then save the score of the ip tag into redis in the classified content, so that the ip data is retained.

C. Solution Introduction

The system functions are divided into: data cleaning, data classification, data processing and landing. When each user visits the ticketing website, some log files of css, js, and html will be generated. By filtering out the more log files, the log files of the user's query are found for processing. For one-way and round-trip data query The content is classified, the actual access status of each different user is counted, and then the actual behavior of the user is analyzed to determine whether the user is crawling. It can improve the user experience, reduce the waste of system resources, enhance the competitiveness of the platform, and achieve real-time user traffic diagnosis content.

III. SYSTEM OUTLINE DESIGN

A. System Overall Architecture Design

The overall structure is divided into data collection, data cleaning, data processing, and data landing. The anti-crawler platform collects the buried point information of the front-end log to reach the nginx server, and then the LUA script sends the full amount of log information to the kafka distributed message queue. The Spark computing engine

consumes the full amount of log information by combining with kafka, cleans up unnecessary data according to the cleanup rules saved in mysql, encapsulates the case class, and classifies different topics through the flag bit of the encapsulated log information, and encapsulates The classified logs are entered into different Kafka topics according to the classification basis. These are different topics such as one-way and round-trip in Kafka. According to the nature of each piece of data, the Spark computing engine combines Kafka to consume the Topic content after the classification in Kafka is successful. After the event is stored and aggregated through transformation inside the Spark computing engine, the sink is landed on the data export and storage engine, and the backend server reads the storage engine and displays it in real time on the front end.

B. System Logic Architecture Design

The core module of the logical architecture is the content maintenance of the ip score, and the function of this module is to maintain the ip score to identify the blacklist ip. Ip score logical architecture design, ip visits key pages, if the number of ip visits and the visit time are less than the visit interval or the total visit time is greater than the time period of visits, the ip will be scored for sensitive ip, when the ip score reaches the threshold When the ip becomes the content of the ip blacklist, the background uses the content of the ip to restrict access to the ip, and the content of the ip is accessed every time. If the conditions of the illegal access are met, it will be The corresponding score content in each ip is added. When each content in the ip reaches the limit of the specified score, the ip will be turned into a blacklist, and access to this ip will be restricted to a certain extent so that the front-end is not used to restrict user access.

C. Data Classification Architecture Design

The following is to make statistics on key pages (query, reservation), and by filtering the statistics of key pages, you can easily identify the crawler information, because ordinary users will not visit these key pages in a large amount in a short period of time. According to the key page rules of the database, it is possible to know which IPs have a particularly high number of visits to key pages, because crawlers crawl for special pages, so that crawlers can be identified. The content of the round trip is an abstraction of the upper level of the data. After the data is cleaned up from the content of js, css, and html, the program content is classified according to the number of active connections and the amount of ip data, and then the classified content is classified The score of the ip tag is stored in redis so that the ip data is retained.

D. IP Sensitive Identification and Threshold Architecture Design

For the score identification of sensitive ip, we have a certain expression in it, and set different thresholds. This is to adapt to the iteration of the version. For each log generation, different thresholds are checked. If the ip score reaches 90 After the division, if the threshold does not appear to modify the content, the ip will be changed to the blacklisted ip and the access will be restricted in the background. In this way, our maintenance of the threshold of ip scores is a healthy state. At the same time, adjusting the proportion of each score reasonably can make the anti-crawler version iterate appropriately, making it more adaptable to the content of the business, and improving the user's favorability. Architecture It is also relatively reasonable.

E. Big data platform environment deployment

- Make sure that jdk1.8 is installed on the three machines, the SSH configuration is password-free, each machine is set with a host name and a successful host mapping is established, and the time is synchronized.
- Install Hadoop fully distributed cluster.
- Install Kafka cluster.

IV. DETAILED DESIGN AND IMPLEMENTATION

A. Command Module to Start

To start the big data components required by the system, zookeeper, kafka, nginx, redis, etc., first start the zookeeper coordination service, which is the core adjustment component of kafka, and then start the kafka distributed message queue, kafka is responsible for entering the source side of the data Content, the data in kafka is realized by nginx combined with lua, and finally the storage engine redis is started.

Fig. 1: Zookeepe startup renderings

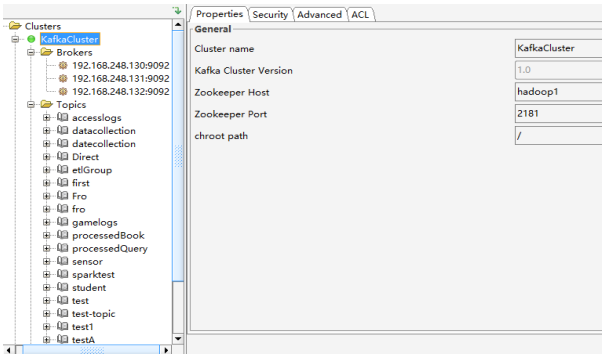

Fig. 2: Kafka startup renderings

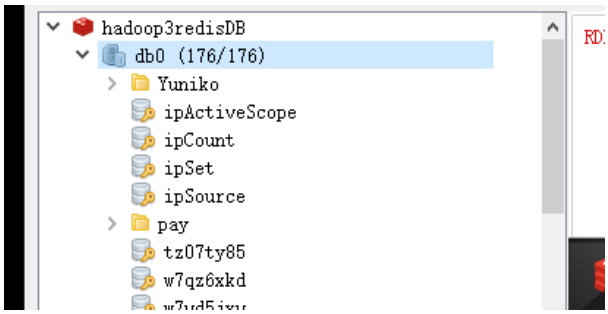

Fig. 3: Redis startup renderings

B. *Initial Data Source Module*

The initial data source is the beginning of the data, which is equivalent to the content of the beginning of data processing. This is the data situation of the log. Data cleaning is to clean up the data, cleaning up css, js and other redundant files that do not need to be analyzed. The initial data such as Shown in Figure 4.

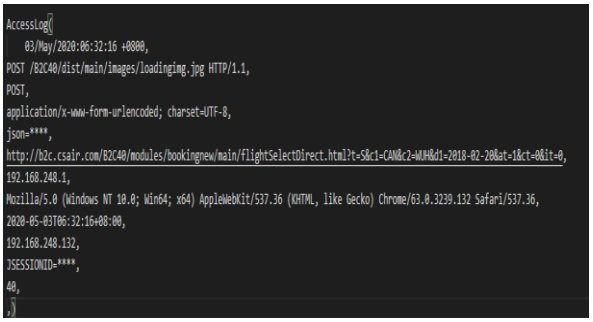

Fig. 4: Initial data source startup renderings

C. *Data Classification Module*

The data classification module classifies the http field in the log. The starting data source is to mix the content together, including one-way and round-trip content, and some data cleaning content. In the data cleaning phase, the redundant part has been removed, and then the data is written to different data terminals through the one-way and round-trip in the field identification.

D. *Data Processing and Landing Module*

Flow diagnosis is a flow diagnosis for all data content. This is a process without data cleaning. Flow diagnosis mainly reflects the real-time access status of the website. This time, the spark batch processing streaming engine is used to achieve this.

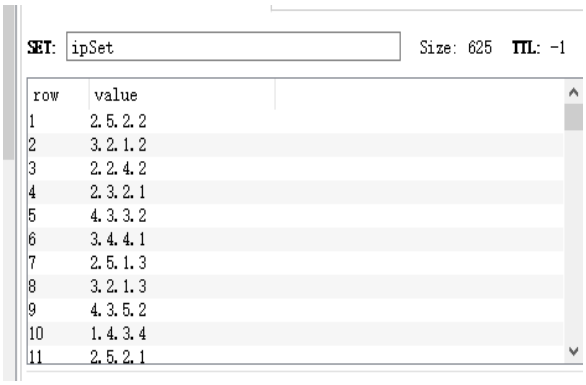

Fig. 5: The total IP history of the traffic diagnosis module

For the maintenance of the IP content of the key page, the initial content is in redis, and then the data that reaches the threshold is put into mysql through the data polling verification in redis. This process requires that the scores in redis are always accumulated, and his scores are modified by the threshold of the score reached. In this way, different ips have their own scores. When the content reaches a certain threshold, it will be put into Mysql and handed over. Back-end display. The data effect of ip maintained in Redis is shown in Figure 6.

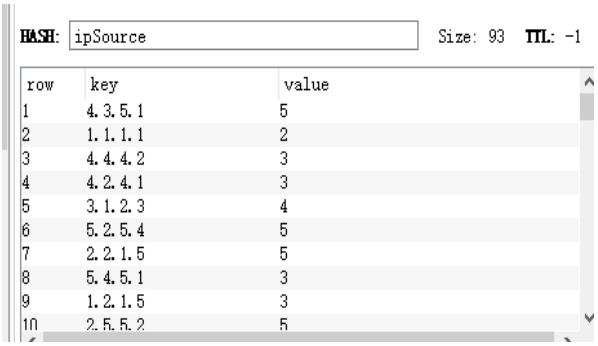

Fig. 6: Redis startup renderings

V. TESTING AND OPTIMIZATION

A. *Data Cleaning Test*

Modify the cleanup rules, and the data cleaned by data printing is in compliance with the cleanup rules. Figure 7 shows the situation where the cleanup rules are modified to the data before css, js, and jsp.

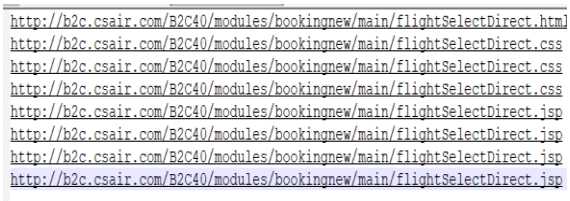

Fig. 7: The content of the source data before modification

The data situation after modifying the cleaning rules is shown in Figure 8.

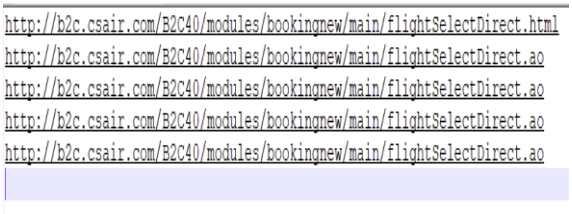

Fig. 8: The content of the modified source data

Test conclusion: Through the content of different cleaning rules, the content of the files left in the log file is different, and the data can be effectively processed.

B. *Data Score Test*

Test conclusion: Reasonable modification of the threshold can effectively screen out ips that violate the rules to a certain level. The amount of ip data in the blacklist increases significantly, and the test is successful.

C. *Technology Iterative Optimization*

Technical optimization and related optimization and optimization of business lines in the project include:

- Optimization of saving of kakfa offset.
- The number of partitions and replica data of Kafka are reasonably set and optimized.
- The setting of Spark calculation chain and the optimization of reasonable application operators.
- Reasonably use lua scripts to write data to the kafka distributed message queue.

D. Iterative Optimization of Business

For frequency: the control of the threshold, and the control time of the total of the user's access time and the score of the total number of times total——time (special users wait a long time after the query is completed before querying the next time, and only record the time within 1min. And the weight is relatively low), the thresholds of ip scores at different times are shown in Table 5-1.

For times: times total-count For the number of visits to the same ip, this is the second determining factor of our weight. Each crawler has a specific point in common. For multiple visits to an ip, this still exists. The score thresholds for different times are shown in Table 5-2.

For the interval of visits: There are different scoring modes for different intervals of a visit.

E. Optimization and Judgment of Initial Data Source

The average online time is 60 min => 5 logs are generated in 1min min => 2K per log. The amount of data in one day is 11 million * 60 * 2K => 1.320 million K => 1.22T. Average log data volume => 1.22T/18*60*60 => 20m/S Normal 1w/S. Peak log data volume => 50m/s (noon/night). The number of peak data in the log => 25600 records/s. The problem of deduplication of massive data is not ideal using bitmap in Redis. HyperLogLog is implemented in batches according to the situation of the data landing, and the choice of different data landing is based on the actual situation of the storage engine. Choose the window opening time according to the actual situation of the data. Real-time data landing when batch storage.

VI. CONCLUSION

This article mainly discusses the analysis, design and implementation of the anti-crawler system. The application of crawler technology has brought serious problems to the operation of certain websites, interfered with the normal business of the website, and harmed the rights and interests of users. Anti-reptile technology came into being. In response to this problem, this topic chooses anti-crawler technology based on big data, investigates and analyzes website anti-crawler requirements, combines technical elements, and outlines the design of the subject system, designs system modules and usage procedures, and analyzes each module Carried out the design of data structure and algorithm, and completed the construction of the system.

For the realization of the subject, choose a big data distributed environment and technology. The system has the possibility of diagnosing the existence of crawlers in the website IP, real-time traffic diagnosis, and the function and role of providing the data interface required by the back-end.

Tests show that the system is easy to operate and stable. Able to better complete data processing and corresponding landing tasks, meet system requirements, and achieve development goals. The test found that the system still has problems/deficiencies such as excessive data volume and performance bottlenecks. In the future learning, it will be further optimized and improved.

REFERENCES

[1.] (America) Tom White, Wang Hai, East China, Liu Yu, Lu Yuehai translation. The definitive guide to Hadoop, Tsinghua University Press, 2017
[2.] Neha Narkhede, Gwen Shapira, Todd Palino. The authoritative guide to Kafka. People's Posts and Telecommunications Publishing House, 2018
[3.] (America) Matei Zaharia, (America) Bill Chambers. Spark: The Definitive Guide, OReilly, 2018
[4.] (America) Holden Karau, (America) Andy Konwinski, (America) Patrick Wendell, (Canada) Matei Zaharia. Translation by Wang Daoyuan. Spark fast big data analysis. People's Posts and Telecommunications Press, 2015
[5.] Ning Haiyuan, Zhou Zhenxing, Peng Lixun, etc. High-performance MySQL (Third Edition) [M]. Beijing: Publishing House of Electronics Industry, 2013.
[6.] Qian Wenpin. The Deep Adventure of Redis: Core Principles and Application Practice [M]. Beijing: Electronic Industry Press, 2019.
[7.] (America) Tom White|Translators: Wang Hai, East China, Liu Yu, Lu Yuehai. Hadoop Authority Guide: Big Data Storage and Analysis (Fourth Edition) [M]. Beijing: Tsinghua University Press, 2017.
[8.] (America) Flavio Junqueira, Benjamin Reed|Translator: Xie Chao. ZooKeeper: Detailed Explanation of Distributed Process Collaboration Technology [M]. Beijing: Mechanical Engineering Press, 2016.
[9.] Lin Yiqun. In-depth analysis of Hadoop HDFS[M]. Beijing: Mechanical Industry Press, 2017.
[10.] Zhu Jie. Detailed explanation of big data architecture: from data acquisition to deep learning. Beijing: Electronics Industry Press, 2016.
[11.] (America) Edward Capriolo, (America) Dean Wampler, (America) Jason Rutherglen. HIVE Programming Guide. People's Posts and Telecommunications Press, 2013
[12.] (America) Tom White, Wang Hai, East China, Liu Yu, translated by Lu Yuehai. The authoritative guide to Hadoop, Tsinghua University Press, 2017
[13.] Xu Shiwei, Lv Guihua. GO language programming. People's Posts and Telecommunications Press, 2012
[14.] Neha Narkhede, Gwen Shapira, Todd Palino. The authoritative guide to Kafka. People's Posts and Telecommunications Publishing House, 2018
[15.] Fu Lei, Zhang Yijun. Redis development and operation and maintenance. Machinery Industry Press, 2017.