2019 the 4th IEEE International Conference on Big Data Analytics

An Approach for Dynamic Scheduling of Data Analysis Algorithms

Jun Gui[1,2,3,4,5], Zeyu Zheng[1,2,3], Yuan Gao[1,2,3], Zhaobo Qin[5]
[1]Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China
[2]Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110016, China
[3]Key Laboratory of Network Control System, Chinese Academy of Sciences, Shenyang 110016, China
[4]University of Chinese Academy of Sciences, Beijing 100049, China
[5]School of Information Science and Engineering, Shenyang University of Technology, Shenyang 110870, China
e-mail: guijun@sia.cn

***Abstract*—Dynamic scheduling of a set of algorithms is a key problem for data analysis platform. In this paper, we propose an approach to efficiently execute and monitor algorithms. Our approach classifies all algorithms into timing tasks, real-time tasks，and equal interval times tasks and configures them separately. An intelligent strategy performs configuration checking for algorithms before scheduling them dynamically. The execution of each algorithm is monitored and controlled according to configuration and feedbacked operating information. Based on this approach, we develop an intelligent data analysis platform with more than 100 algorithms. By stable running for months, our approach is proved to be accurate and effective, and can be applied in many platforms.**

Keywords-data analysis; Scheduling Strategy; Monitoring algorithm

I. INTRODUCTION

In the past decade, the amount of data storage in the world has grown exponentially. The era of big data had come [1]. Thanks to the popularity of digital devices [2] and the development of statistical data[3], scientists in many fields have collected a large amount of data through experiments and observations. They tried to solve the problem through data analysis. For example, Physicists use k-means clustering algorithm and Bayesian to further analyze the conductive phenomena at complex oxide interfaces [4]. Meteorologists use data to create models of extreme weather for easy understanding and prediction. Chemists have successfully screened high-output polymers using combinatorial chemistry and data mining tools in catalyst and polymer development [5]. Biologists systematically collect gene expression data sets for data analysis and consistent signal evaluation to identify key factors in biological processes [6]. Sociologists use data mining methods to explore topics that media topics are viewed as social issues by online audiences [7]. These examples fully demonstrate that advances in technology are closely related to the development of data science.

Data science is developing very rapidly in the present, but there are also many problems that need to be solved. Big data will increase the risk of disclosure of personal privacy information. At this stage, security technology cannot meet the security of personal information under big data.[8] The ever-increasing amount of data has placed increasing demands on the technology of storage, acquisition and transmission [9]. At the same time, it also faces one of the most important issues--data analysis, especially for the fast processing of different kinds of dynamic data is crucial [10].

Due to the variety of data analysis algorithms and different usage scenarios, most of the execution of the algorithm is still artificial control, lacking an automatic control scheduling method [11]. And there is a lack of means for monitoring and recording the overall implementation of the algorithm. Especially for the current big data processing and other scenarios, involving a large number of algorithms to perform scheduling monitoring problems [12]. Due to the variety of data, the variety and number of algorithms, relying on artificial or semi-automatic methods for data analysis and mining algorithm control scheduling, affecting overall efficiency and increased labor consumption.

Therefore, a scheduling algorithm is proposed to process real-time multi-class dynamic data. By classifying the pre-scheduling algorithm, the algorithm configuration table is pre-positioned into the database, and the monitoring program continuously scans the algorithm state table to increase the speed of data processing. The mutual exclusion judgment mechanism ensures that the system can be processed stably when the data load is large.

The structure of this paper is as follows. Section II discusses about the principle of scheduling method work. Section III Introduce the algorithm configuration table, which is applied at the beginning of the algorithm. Section IV discusses about the specific workflow of the scheduling method. Section VI Introduce an algorithm state table is introduced, which is applied from the beginning to the end of the program. Section VI discusses about the working principle of the three kind of core algorithms applied in the scheduling method. Section VII Algorithm running results and analysis. Finally, Section VIII presents conclusion.

978-1-7281-1282-4/19/$31.00 ©2019 IEEE

II. SCHEDULING METHOD FRAMEWORK

This method will first classify the pre-scheduling algorithm into the following categories: Real-time algorithm that is always executed in the background; Timing algorithm that is executed once a day at a fixed time; Interval algorithm that is executed once every interval, which can be set to full minutes and hours. Store the algorithm configuration table storing the above classification algorithm and other configuration information into the database. Flexible setting of different algorithm category identification and execution time identification by changing the algorithm configuration table. Different types of algorithms are used for the corresponding scheduling and monitoring framework packaging. When the scheduling algorithm is enabled, the classification algorithm matches and executes the corresponding scheduling algorithm and monitoring mode. Provide mutual exclusion mechanisms for different types of algorithms. Ensure that each algorithm does not repeat execution during execution, thereby avoiding clutter in algorithm execution. After the execution is completed, the execution status and algorithm information are stored in the algorithm state table of the database so that it can be called at any time. The working process is shown in Figure 1.

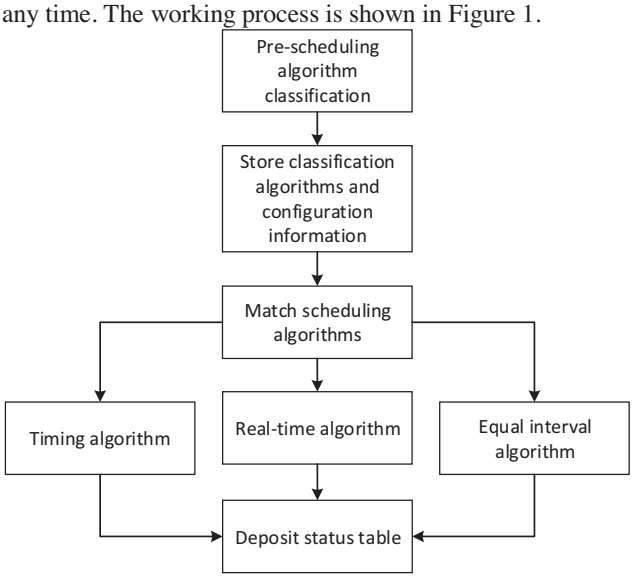

Figure 1. Scheduling algorithm framework

III. ALGORITHM CONFIGURATION TABLE

The algorithm configuration table is placed in the database. The table contains the algorithm name (AL), algorithm category identifier, algorithm execution time setting (time), and whether the algorithm performs the setting (Run). The algorithm class identifier includes: a timing algorithm flag (I: timing task), a real-time algorithm flag (R: Realtime task), and an equal interval algorithm flag (I: interval task). When the flag of the algorithm is 1, it indicates that is it. When the flag of the algorithm is 0, it means not.

The algorithm execution time is set differently for three different algorithms. 09:00:00 in the timing algorithm represents to execute this algorithm at 9:00 everyday; 00:00:00 in the real-time algorithm is the default fixed value because the real-time algorithm is executed from the beginning of the entire control system; 00:30:00 in the equal interval algorithm means that it is executed every 30 minutes and can be modified by modifying the corresponding fields. Whether the algorithm needs to execute the flag: 1 means the algorithm needs to be executed, 0 means the algorithm does not need to be executed. It supports setting the algorithm module running on the same day as needed.

TABLE I. ALGORITHM CONFIGURATION TABLE

AL	T	R	I	time	Run
A	1	0	0	09:00:00	1
B	0	1	0	00:00:00	0
C	0	0	1	00:30:00	1

IV. SCHEDULING METHOD WORKFLOW

The program starts, and the scheduling method scans the configured algorithm configuration table every minute. Obtain the algorithm that matches the current time, arrange the algorithm and ready to execute. Mutually exclusive judgment and then execute the algorithm. The mutual exclusion judgment mechanism guarantees the uniqueness of the algorithm execution and prevents the repeated scheduling algorithm, resulting in algorithm confusion. When the judgment result is YES, it indicates that the program is being executed, and the status log is not updated and the status log is updated. When the judgment result is no, it indicates that the scheduler should be executed, and the scheduler will match the corresponding scheduling algorithm and monitoring mode.

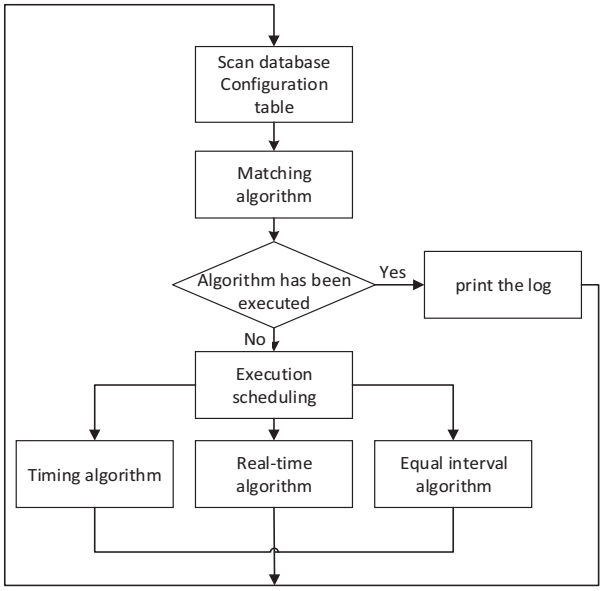

Figure 2. Scheduling system operation process

V. ALGORITHM STATUS TABLE

Algorithm state table, which is a data table that stores the execution of the algorithm. The scheduling method will feedback and update the values in the status table based on the operational status during execution. The scheduler and monitor call the values in the state table at any time during execution. The values in the status table also provide the user with algorithm profile information. For fault algorithms, fault information is stored in the log for analysis.

TABLE II. ALGORITHM STATUS TABLE

date	AL	Start time	Finish time	Run time	status
2017-06-20	A	2017-06-20 14:35:42	2020-01-01	40	0
2017-06-21	B	2017-06-21 14:33:12	2020-01-01	40	0
2017-06-22	C	2017-06-21 14:30:33	2020-01-01	40	0

The table includes: the execution date of the algorithm (date), the configuration item of the algorithm (AL), the execution start time (start time), and the execution end time of the algorithm (finish time), Algorithm execution time (runtime), State flag for algorithm execution (status). The execution start time needs to be accurate to the second. The end time of the algorithm is filled in by default 2020-01-01 00:00:00. The status flag of the algorithm execution: 0 for failure, 1 for success, and 2 for execution.

VI. ALGORITHM SCHEDULING AND MONITORING

A. *Timing Algorithm*

The timing algorithm is an algorithm that is executed at a specific time every day. The algorithm monitors the process of data processing by iteratively checking the state table of the algorithm and feedback the state of the data processing algorithm in time. The figure 3 shows how the timing algorithm execution. The timing algorithm is roughly divided into two parts, and the first part detects whether the data processing algorithm is successfully scheduled. The second part is whether the data processing algorithm is executed successfully.

The scheduling system scans the algorithm state table and obtains state information of the algorithm to be scheduled. The state information of the algorithm is represented by the status flag in the algorithm state table. There are four cases of status flag: the status flag is empty to indicate that the algorithm for processing data has not been scheduled; 0 indicates that the algorithm failed to execute; 1 indicates that the algorithm executed successfully; 2 indicates that the algorithm is being executed. Through the information of the status flag, the system can detect whether the algorithm for processing the data has been scheduled. If not scheduled, the scheduling system schedules the algorithms that need to be executed and feeds back the state of the algorithm to the data state table. Otherwise, the algorithm has been scheduled. The already scheduled data processing algorithm needs to judge whether the operation is successful according to the status flag. If this algorithm fails, it needs to be rescheduled. Each time after the system operates the data processing algorithm, it will feed back its running status to the algorithm status table. Because the timing algorithm only needs to be run once a day, the successful data processing algorithm does not need to be repeatedly scheduled. Finally, the program will detect whether the scheduled algorithm is successful according to the algorithm status flag.

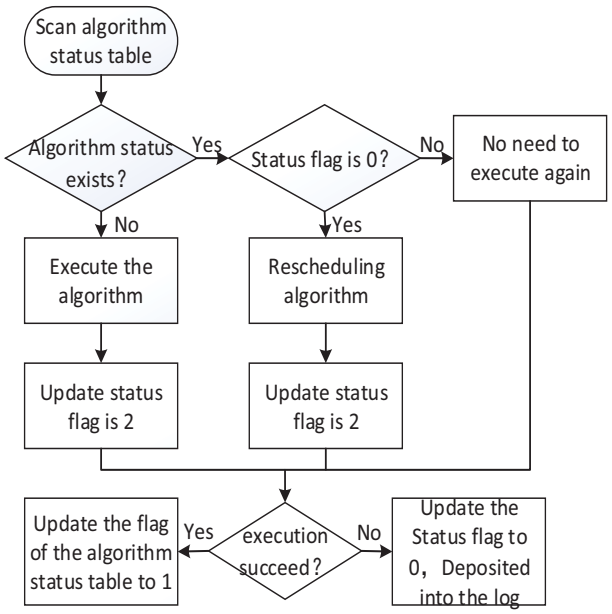

Figure 3. Timing algorithm execution process, 0 indicates that the algorithm failed to execute; 1 indicates that the algorithm executed successfully; 2 indicates that the algorithm is being executed.

B. *Real-time Algorithm*

A real-time algorithm is an algorithm that is always executed in the background. The execution process of the real-time algorithm is basically similar to the execution process of the timing algorithm. Figure 4 shows how the real-time algorithm works. However, there are only three states in the algorithm state table: the state flag is empty to indicate that the algorithm for processing data has not been scheduled to be executed; 0 indicates that the algorithm failed to execute; 2 indicates that the algorithm is being executed. Because the data processing algorithm scheduled by the real-time algorithm needs to be executed in the background, there is no need for a successful state, and only need to detect whether it fails.

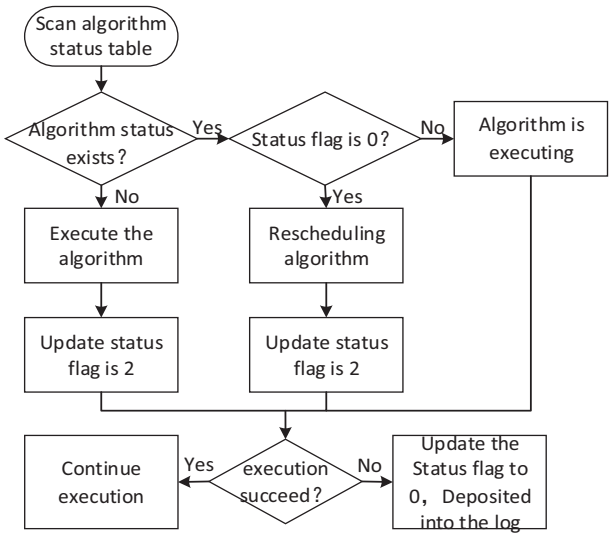

Figure 4. Timing algorithm execution process, 0 indicates that the algorithm failed to execute; 2 indicates that the algorithm is being executed.

C. *Equal Interval Algorithm*

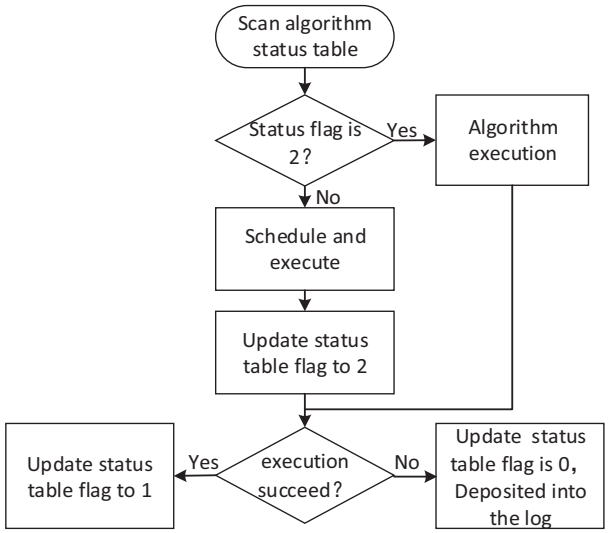

Figure 5. Equal interval algorithm execution process, 0 indicates that the algorithm failed to execute; 1 indicates that the algorithm executed successfully; 2 indicates that the algorithm is being executed.

The equal interval algorithm is an algorithm that is executed once at a regular interval. The fixed duration can be minutes and hours, and does not support this mix of 1 hour and 15 minutes. Figure 5 shows how the equal interval algorithm execution. The equal interval algorithm is roughly divided into two parts, and the first part detects whether the data processing algorithm is successfully scheduled. The second part is whether the data processing algorithm is executed successfully.

The scheduling system scans the algorithm state table and obtains state information of the algorithm to be scheduled. The state information of the algorithm is represented by the status flag in the algorithm state table. The status flag has three states: 0 indicates that the algorithm failed to execute, 1 indicates that the algorithm executed successfully, and 2 indicates that the algorithm is being executed. The execution status of the program is judged by scanning the status flag. If it is already in execution, there is no need to reschedule. Otherwise, the algorithm that needs to be executed is scheduled. The scheduling system needs to feed back the execution status of the data processing algorithm to the algorithm status table each time when the new data processing algorithm is executed. The system makes the program execute more stable by monitoring the status flags. Finally, the system will judge if the algorithm execution was successful. The system feeds back the status of the success or failure to the system status table. If the program fails to execute, you will also need to print a failed log. Because the equal interval algorithm is executed every once in a while, as long as the flag is not 0, the algorithm is always running.

VII. ALGORITHM RUNNING RESULTS AND ANALYSIS

We have made the scheduling strategy introduced in this article into a scheduling system. The front end of the scheduling system uses the Bootstrap framework combined with jQuery and Html5 technologies. The back end of the scheduling system uses the Django framework. The programming language of the system is Python 3.5 and R3.4.3. The server of scheduling system is configured with a 12-core processor and 32 GB RAM. The scheduling system runs continuously for many days, and 15 data processing algorithms are scheduled every day. The amount of data processed by the algorithm is 3.5G. Figure 6(a) shows the processing results in a visual way.

Algorithm of equal interval data process processing are placed in interval tasks. This category includes an algorithm for processing the inverter alarm data (jarolalarm.py), an algorithm for processing the operating power data of the inverter (jarolworkpower.py), and an algorithm for processing the operating state of the inverter (jarolworkstate.py). They only need to be executed once at regular intervals. This kind of algorithm takes about 10 seconds to process the data on average (figure 6(b)). Algorithm of timing data process processing are placed in timing tasks. This category includes some algorithms for fault prediction and some algorithms related to inverter running (figure 6(c)). They only need to run at a fixed time. For example, the fault prediction algorithm (fault prediction.py) worked at 1 am.

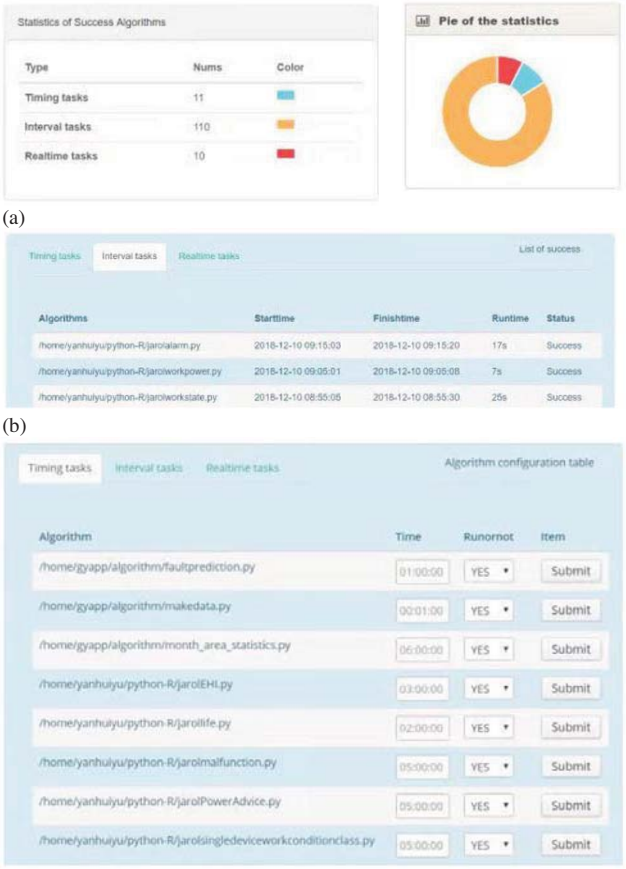

Figure 6. (a) The scheduling system classifies the data processing algorithm with a visual method.(b) Data processing algorithm under the equal interval algorithm task.(c) Data processing algorithm under the Timing algorithm task.

VIII. CONCLUSION

We use this approach to simplify the complexity of algorithmic scheduling control and improve the ability to process dynamic data quickly. Different scheduling methods are provided for different types of algorithms to ensure the scheduling accuracy and security of each type of algorithm. The configuration data is stored in the database. The configuration data includes the category identifier of the scheduled algorithm, the execution time identifier, and whether the identifier is executed, etc., which is convenient for change and setting. Feedback The status of the scheduled algorithm, including start-stop time, algorithm time-consuming, algorithm status (failure, success, execution) and other information are stored in the database table and in the log. This method has a mutual exclusion mechanism to ensure that different types of algorithms do not repeat scheduling during execution, and can implement automatic scheduling execution and monitoring without human manipulation.

ACKNOWLEDGEMENT

This work is supported by National Key R&D Program of China (No.2018YFF0214704). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

REFERENCES

[1] Luebbers, Dominik, U. Grimmer, and M. Jarke. "Systematic Development of Data Mining-Based Data Quality Tools." Proc. VLDB. 29th International Conference on Very Large Data Bases, Morgan Kaufmann, Sep. 2003, pp. 548-559.

[2] Chung Hyunji, Park Jungheum, Lee Sangjin, Kang Cheulhoon." Digital forensic investigation of cloud storage services". Digital Investigation,Vol.9,Nov.2012,pp.81-95,DOI: 10.1016/j.diin.2012.05.015.

[3] Shahidinejad, Soheil, E. Bibeau, and S. Filizadeh. "Statistical Development of a Duty Cycle for Plug-in Vehicles in a North American Urban Setting Using Fleet Information." IEEE Transactions on Vehicular Technology, vol.59, Oct. 2010, pp. 3710-3719, DOI: 10.1109/TVT.2010.2061243.

[4] Strelcov, Evgheni, et al. "Deep Data Analysis of Conductive Phenomena on Complex Oxide Interfaces: Physics from Data Mining." ACS Nano, Vol.8, Jun. 2014, pp.6449-6457, DOI: 10.1021/nn502029b.

[5] Ganguly, A. R, et al. "Toward enhanced understanding and projections of climate extremes using physics-guided data mining techniques." Nonlinear Processes in Geophysics, Vol.21, Jul.2014 pp.777-795, DOI : 10.5194/npg-21-777-2014.

[6] Tuchbreiter, Arno , et al. "High Output Polymer Screening: Exploiting Combinatorial Chemistry and Data Mining Tools in Catalyst and Polymer Development." Macromolecular Rapid Communications, Vol.24, Jan.2010, pp.47-62, DOI: 10.1002/marc.200390014.

[7] Li, Jin , et al. "A data mining paradigm for identifying key factors in biological processes using gene expression data." Scientific Reports Vol.8, Jun.2018, pp.9083, DOI:10.1038/s41598-018-27258-8.

[8] Zou Hui. " Protection of personal information security in the age of big data." Proc.CIS. 12th International Conference on Computational Intelligence and Security, Institute of Electrical and Electronics Engineers Inc. Jan. 2017, pp.586-589, DOI: 10.1109/CIS.2016.141.

[9] Ma, Xiao Xing , and D. Wu . "Research on Information Security Issues Facing the Era of Big Data." Proc ICAEMAS. 3rd International Conference on Advanced Engineering Materials and Architecture Science. Trans Tech Publications Ltd, Jul.2014, Applied Mechanics and Materials, Vol. 651-653, pp.1913-1916, DOI: 10.4028/www.scientific.net/AMM.651-653.1913.

[10] Jindal, Anish , N. Kumar , and M. Singh . "A unified framework for big data acquisition, storage and analytics for demand response management in smart cities." Future Generation Computer Systems, 2018, DOI: 10.1016/j.future.2018.02.039, in press.

[11] Au, Wai Ho, K. C. C. Chan, and X. Yao. "A novel evolutionary data mining algorithm with applications to churn prediction." IEEE Transactions on Evolutionary Computation, Vol.7, Dec.2003,pp.532-545, DOI: 10.1109/TEVC.2003.819264.

[12] Burges, Christopher J. C. "A Tutorial on Support Vector Machines for Pattern Recognition." Data Mining and Knowledge Discovery, Vol.2,Jun.1998, pp.121-167, DOI: 10.1023/A:1009715923555.

.