Improve Repairing Data Process by Using Multiple Based-Rules

Yuan Feng, Guohua Liu
Shanghai, Songjiang district, Renmin North Road 2999, Donghua University
yuanfeng5865@gmail.com, ghliu@dhu.edu

Abstract. This paper proposes a method on solving the problem of dealing with dirty data in the database. Considering the complexity of the structure of the data, based on the previous methods that work on this problem, our method combines the methods that use regular expression and methods that use conditional functional dependencies, to complete the data quality improvement. This method uses dependencies to improve the repairing speed and the searching time on the data. The repairing based on the regular expression is regular while there exist questions that the repairing efficient is influenced by the amount of data. When dealing with the database from company Standard Solution Group (SSG) which is from the reality world data, we have tried other related methods and inspired by these methods, we propose this method. The experiments on the data from SSG shows that this method is much efficient.

1. Introduction
Nowadays, in the organizations there is an increasing need for dealing with data and at the same time growing problems arise within the large amount of data. With the technology of data science and artificial intellectual technologies developing, varies types of data and complex structures of data are becoming disturbing the efficiency and accuracy of the whole process of technology development [1,2]. In this case, the quality of the data and database is becoming utmost important.

The quality of data in the database of a company influence the results of prediction or evaluation of products in the company [3]. High quality of data directly influences the quality of the results of the company and will improve the efficiency of the production and prediction in the company. The methods work on promoting the quality of data are varies: some are rule-based methods that requires the data to obey a set of rules [4]; some are learning-based methods to improve data quality by repairing data using machine learning models. By using models such as decision tree, neural network and classification methods to predict the candidates that may be the right data to be used on the data correction. Some are based on the discover truth from the data that may exist confliction.

This paper inspired by the method of regular expression to work on the correction of the data and the method of function dependencies that are working on the consistency of the database. In the previous works, the detection and correction of the data and the consistence of the data are two problems, which are discussed separately. While in this paper, we talk about the two problems which can be considered together to improve the quality of the database more efficiently.

2. regular expression on data repairing
On the data repairing process, regular expression is used to detect and correct the data. In the most situations, based on the different columns in the dataset, different data structure can be expressed by the related and unique regular expressions. And like any other language grammars, regular language
grammar expresses regular language efficiently. By using non-determined finite automaton (NFA), the regular language can be equally expressed by an automaton. When using regular expression to correct the data, the NFA which we define and design before is a useful tool. In the previous work, the edit-distance is used to measure the distance between the two strings. In the process of correction, edit-distance is adopted to be a tool to determine the token values to take place.

2.1. Non-determined finite automaton
The non-determined finite automaton is an important tool to recognize and express regular languages and regular expressions. A non-determined finite automaton can be defined by a 5-tuple \((Q, (\Sigma \cup \{\varepsilon\}), \delta, q_0, F)\), where

1) \(Q\) is a finite set called the states,
2) \((\Sigma \cup \{\varepsilon\})\) is a finite set called the alphabet,
3) \(\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow Q\) is the transition function,
4) \(q_0 \in Q\) is the start state, and
5) \(F \subseteq Q\) is the set of accept states.

2.2. RSR algorithm
In the research before, repairing data with regular expression is popular. A lot of data can be expressed by regular expression and non-determined automaton. And by using this method, the data can be corrected. While recently the repairing processing has improved by using edit-distance to make the choose of the token repairing. The new method proposed is known as RSR algorithm[5]. We here use this method to do the correction.

2.2.1. operations of regular expression
Let \(M\) and \(N\) be languages. The regular language or regular expression can be expressed by using the regular operations union, concatenation and star. They are defined as follows:

1) Union: \(M \cup N = \{x | x \in M \text{ or } x \in N\}\),
2) Concatenation: \(M \cdot N = \{xy | x \in M \text{ and } y \in N\}\).
3) Star: \(M^* = \{x_1x_2 \ldots x_k \mid k \geq 0 \text{ and each } x_i \in M\}\).

The union operation is combining the strings in the language \(M\) and \(N\). Strings in the operation \(U\) is considered to be put in one language. Concatenation operation is connecting the strings in \(N\) and \(M\). In the process of connection, first choose one string from language \(M\) and then choose one from \(N\), the order is concrete but each time the amount of choosing is limited to one. Star operation is selecting strings from language \(M\) and connecting them together to form a new language. The selection can be null and \(\varepsilon\) can be put in the new language.

2.2.2. Algorithm processing
First input the string \(s\), and NFA corresponding that can represent a regex that with an edge set and a set of accept state. The edge set is the set of each step of processing the \(s\) to NFA step, and the edit operator sequence will evaluate the edit distance to convert \(s\) to match the regex. The outputs of the algorithm are the smallest edit distance and edit operator sequence that in the comparing processing.

2.2.3. Example
The repairing data in one specific column which can be expressed by regular language. And at the same time, regular language can be expressed by either non-determined finite automaton or regular expression which uses the above operations to form complex languages. When dealing with the data repairing, we first give the data expressions to express the data and the second step is design a specific NFA for this expression. For example, the data we met from the real world is like the table I below:
TABLE I. EXAMPLE OF DATA

| SKC-nummer | Benämning | Grupp | Varumärke/Norm | Referens (artikelnr, typbet, rit.nr) |
|------------|-----------|-------|----------------|-----------------------------------|
| 11111111   | FEOJEW    | 695954| OEWIEORW       | FIOWIEOFJ                         |
| 492374839  | FEWFEWO   | 54309504| REKWLKR       | FEIOW93200                        |
| 43245333   | FEWOFJKS  | 424320| REWRKEE        | 504892049332                      |
| 333333363  | VLKL      | 43517000| REW           | 5904294032                        |
| 766643225  | POKI      | 533843200| REWREW       | FJEOWF2MKPGR0903                 |
| 540939855  | OFEOW     | 4324340| AXRON         | 4302985024930FJEIOJFLS            |

As the data above showed, column “Benämning”, “Grupp” and “Varumärke/Norm” are much constant because they are constant in some way because the name, group and the norm can be stored in a dataset and when we detect one value, we can compare whether there is one corresponding to it. If we find that value, we can determine whether it is correct. And by using conditional functional dependencies below, we can use the previous data which are correct to do the correction.

While the column “Referens (artikelnr, typbet, rit.nr)”, this column’s structure is much complex. In this case, we use RSR algorithm to illustrate and express this column and in this way, this column can be detected and corrected. In the process of expressing this column, we find a problem that is important and needs to be discussed, which is the expressions and NFAs are a lot. The number of rules grows which influence the speed of repairing. To solve this problem, we propose this method to improve.

3. repairing process with conditional functional dependencies (CFDs)

3.1. Conditional functional dependencies discover algorithm
In the data mining process, the rules underlying is meaningful in the database. In the formal research, there are a lot of work on the discovery of functional dependencies [6, 7] and then the research reaches the discover of conditional functional dependencies. Here we adopt the algorithm CTANE[8,9] to discover the associate rules.

As the discovery of association rules work, many works have done on this area. Cause the based on the association rules, the repairing process can be done in most data. While we adopt the association rule here is not for the repairing but for the search speed of the regex, because the association rules are much more lesser and the association rules are related with the columns. We use the rules to determine the regex that the specific regex obey.

In our repairing process, we choose conditional functional dependencies to find the associate rules in the database. Besides the column of reference, we apply the discover of CFDs among the other columns.

3.2. use cfd improve repairing process
To make the repairing process more efficient, we build a hash map data structure to store the rules. The structure is as follows:

Hash map: [(CFD), (regular expression1, regular expression2, regular expression3, ...)]

Considering the regular expression searching process and the regular expressions’ amount grows, we propose this data structure to store the rules and help improve the repairing process. In the first part, we store the CFD that is found by using CFD discovery algorithm. Based on the database, because the CFDs a database has are a lot, here we should choose some basic and essential associate rules that the database follows. While the second process is storing the regular expressions related. When we search the rules that the reference data obeys, we use hash map to map this data by using CFD. And then try the following regular expressions to test the column, like reference column.
3.3. the processes of the whole repairing

In the process of repairing, we use CFD discovery algorithm CTANE to get CFDs and construct regular expressions to create repairing rules. And then we use the repairing rules and RSR algorithm to correct the data in the database. The improvement part of repairing is that the repairing based on regular expression is influenced by the amount of regular expressions, so we improve the process by using CFDs. We first determine the data obeying the CFDs and then by using hash map to determine which regular expression it obeys. This process improves the speed of searching the expression. The whole process is shown in the figure 1.

The processes of the whole repairing are as follows:
1) use corrected database to discover CFDs and construct regular expressions and NFAs;
2) use CFDs and constructed regex to form the repairing rules;
3) then the rules constructed can be used to applied on the repairing data.

4. experiment and results

The experiment is done on a PC with Intel core i7 CPU 2.1GHz, 8GB memory and 64bit Windows 10.0.

4.1. the table of rules of RSR algorithm

Considering the data in the column with data that can be expressed by regular expression, we construct the regex based on the dataset shown on the following table II.

| TABLE II. RULES OF RSR ALGORITHM |
|----------------------------------|
| Regex | Rules |
|-------|-------|
| regex1 | (0-9)*abeF0(A-Z)*(a-z)*(ijk)|(white)(blue) |
| regex2 | (white)*|((0-9)*(0-9)*(0-9)*)|(back) |
| regex3 | (a-c)*(0-9)*269|(cs)*982 |

4.2. the result of CFDs

By using the CTANE algorithm, we can get CFDs as follows, shown in the table II. While the number of CFDs is huge, we choose the useful and important columns to construct the final rules. The rules chosen are based on the data in the real world or the component s of the data. Here we choose no more than three columns to determine the rules. In the table III, rules are constructed.

| TABLE III. RESULT OF CFDS |
|---------------------------|
| CFDs | Table Column Head |
|------|-------------------|
| CFD 1 | ['J', 'K', ('\-\-\-'), ("2",)] |
| CFD 2 | ['J', 'T', ("\-\-\-"), ("1",)] |
| CFD 3 | ['H', 'T', ("\-\-\-"), ("1",)] |
| CFD 4 | ['BH', 'G', ("\-\-\-"), ("\-\-\-"), ("\-\-\-")]] |

4.3. The Table of Hashmap

As we construct the table of rules, we use the CFDs found before and regex constructed, we can get the table IV as follows:
4.4. Number of CFDs Influenced by K-frequency
By using the data to get CFDs, the number of CFDs are influenced by the frequency. The experiment shows the number of CFDs in the figure 2. As we can see, the number of CFDs decreases as frequency improves. And when the frequency is large than 3, the number of CFDs decreases a lot. When we construct the rules, considering the cover of rules in the data, we should choose the lower frequency to get more dependencies.

![Fig. 2. Number of CFDs](image)

4.5. Correction Rate
In the figure 3, the accuracy of data is measured by the formula:

\[
\text{accuracy} = \frac{\text{repaired}}{\text{incorrected}}
\]

In the figure, we can see that the accuracy is not improved when the data amount increases. The method does not influence the accuracy of the data repairing.

![Fig. 3. Accuracy of Data Repair](image)

4.6. Improvement on Time
The improvement on time is obvious because of the rules reconstruct. The figure 4 shows the improvement compared with the RSR algorithm alone. The result shows the data repairing is much efficient and time is saved a lot. The time consume formula is as follows:

\[
\text{time consuming} = \frac{\text{repairing without dependencies}}{\text{repairing with dependencies}}
\]
4.7. Error Rate
In the figure 5, we give the influence of the frequency k of the time improvement. The experiment shows that the frequency k influences the result of time consuming. The result shows that when the k is smaller, the time consumes more. The time consuming is compared with the repairing process without the dependencies. The measure results are as follows:

5. Conclusion
In the research of our study, we use RSR algorithm to detect and correct the data which can be expressed by regular expression and NFA. Conditional functional dependencies used here are to help the repairing process to search the expression that the column obeys. By using the formal discover algorithm, the hash map can be built and the experiment using real world data shows that this method is useful to improve repairing process.

References
[1] W. Eckerson. Data quality and the bottom line. Journal of Radioanalytical & Nuclear Chemistry, 160(4): 355-362, 1992
[2] C. May_eid, J. Neville, and S. Prabhakar. ERACER: a database approach for statistical inference and data cleaning. In SIGMOD, pages 75-86, 2010.
[3] F.Chiang and R.J.Miller. Discovering data quality rules. PVLDB, 1(1):1166-1177, 2008
[4] R.Agrawal and R.Srikant. Fast algorithms for mining association rules in large databases. In VLDB, pages 487-499, 1994.
[5] Zeyu Li, Hongzhi Wang, Wei Shao, Jianzhong Li, Hong Gao, Repairing Data through Regular
[6] Yka Huhtala, Juha Karkkainen, Pasi Porkka, Hannu Toivonen, TANE: An efficient algorithm for discovering functional and approximate dependencies, the computer journal, vol. 42, No.2, 1999

[7] P. Bohannon, W. Fan, F. Geerts, X. Jia and A. Kementsietsidis. Conditional functional dependencies for data cleaning. In ICDE, pages 746-755, 2007.

[8] Wenfei Fan, Floris Geerts, Jianzhong Li, Ming Xiong, Discovering Conditional Functional Dependencies, 2011

[9] Philip Bohannon, Wenfei Fan, Floris Geerts, Xibei Jia, Anastasios Kementsietsidis, Conditional Functional Dependencies for Data Cleaning, 2007 IEEE