QUALITATIVE CLEANING METHODS ON DISTRIBUTED IOT DATASETS

_____

A Thesis

Presented to

The Faculty of the Department Computer Science

Sam Houston State University

_____

In Partial Fulfillment

of the Requirements for the Degree of

Master of Science

_____

by

George Ayodeji Ogungbemile

May, 2019

QUALITATIVE CLEANING METHODS ON DISTRIBUTED IOT DATASETS

by

George Ayodeji Ogungbemile

_____

APPROVED:

Bing Zhou, PhD
Thesis Director

Min Kyu An, PhD
Committee Member

Khaled Rabieh, PhD
Committee Member

John B. Pascarella, PhD
Dean, College of Science & Engineering Technology

DEDICATION

I dedicate this dissertation to the Almighty God, the source of all knowledge, who has been so gracious to me all my life. I also dedicate this dissertation to my family, who always motivate and support me through my decision making process and the entire congregation of RCCG Rhema Int'l Ministries, Huntsville Texas. A special dedication goes to my mother for always going the extra mile, even in inconvenient situations and to my sister for her prayers and moral support. I also dedicate this to my wife to be, for going out of her way to convince me about pursuing a graduate degree abroad. Finally, I dedicate this to all the faculty and students in the computer science department for bearing with my learning pace.

ABSTRACT

George Ayodeji, Ogungbemile, *Qualitative cleaning methods on distributed IoT datasets*. Master of Science (Computing and Information Science), May, 2019, Sam Houston State University, Huntsville, Texas.

Data analysis encompasses a set of individual steps that allows a typically large data set to be remodeled such that actionable information can be extracted from the data set, which can then be used to support decision-making. Data generated from multiple distributed sources is usually dirty by default and dirty data will often lead to inaccurate or incomplete data analysis. As a result, without first performing data cleaning, wrong or fatally flawed business decisions is inevitable. IoT describes a network of physical and virtual objects containing software, electrical components and sensors that exchange data with other connected devices over the internet. The data generated from these sensors is distributed by design and my aim for this thesis is to explore qualitative data cleaning methods such as integrity constraints and functional dependency violations to perform error detection and in place error repairing techniques on the distributed data set generated from these devices. This approach is relatively new since most of the prior data cleaning research in this domain have focused on quantitative techniques such as outlier detection.

The next goal for my thesis will then be to perform exploratory data analysis on the data sets from these IoT sources using data wrangling tools on open source frameworks such as Optimus under Apache Spark to handle the unstructured and semi structured formats of the data generated from these sources. The end goal will be to generate clean data from these data sources such that insights can be gained to support decision making for the purpose of product improvement.

KEY WORDS: IoT, Apache, Optimus.

ACKNOWLEDGEMENTS

First, I would like to express my sincere gratitude to my Thesis Director Dr. Bing Zhou, for the support and insight she provided for my research. Her area of expertise also aided me in my research and her guidance and patience have been immensely beneficial to me on this journey. She exposed me to the best tools to source for my datasets and environments to achieve optimal results for my project.

Besides my advisor, I would also like to thank the committee, Dr. Khaled Rabieh and Dr. Min Kyu An, for their insightful comments, valuable suggestions, and encouragement for the improvement of my thesis in different dimensions. I would also like to thank the department for providing the facility and a conducive environment for my learning.

Finally, I would like to thank all the professors and students, for their knowledge sharing and for always being available to provide valuable feedback towards the overall improvement of my work.

PREFACE

Qualitative cleaning methods have existed long before this project, however, this work aims at exploring the effect of such methods, particularly utilizing the concept of integrity constraints, functional dependencies etc. on datasets from IoT devices. The nature of these datasets, which in most cases is distributed, poses a challenge at the error-cleaning phase of the data cleaning process and this research aims at presenting viable solutions to the challenges that this phase constitutes. A combination of unique environments and languages are combined in executing this approach and results are presented.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure **Page**

CHAPTER I

Introduction

Data cleaning approaches exist in two different variants namely quantitative and qualitative techniques. For this thesis, the focus will be on the qualitative approach for cleaning data. Qualitative error cleaning is further divided into 2 broad steps, which are error detection and error repairing. In the error detection phase of the qualitative approach, the detection mechanism utilizes database integrity constraints for identifying violations in a dataset. Integrity constraint concepts such as functional dependency violations, primary key and foreign key violations are then used to categorize data as being in a valid state or a non-valid state. This work will focus on using the steps in this qualitative approach to perform error detection on a distributed data set and later use the apache spark framework to complete the error-repairing phase in the data cleaning process.

Data quality is essential to the success of any data analysis project, as the results of any analysis will mostly depend on the initial data that is fed into the analytical system used in arriving at the results. Errors are often encountered in the process of data acquisition, these errors range in severity from relatively common errors such as duplicated values, which may be easy to address, to more complex errors such as missing values or data in unrecognized formats. In many published approaches employed in cleaning data, two phases are usually referenced which are error detection and error repairing (X Chu et.al, 2016) and the detection phase can either be quantitative or qualitative. Quantitative error detection techniques use known statistical approaches such as outlier detection where, for example, in a company database "*a salary that is three standard deviation away from the average salary is an error*" (J.M. Hellerstein, 2008). Qualitative error detection

approaches use database concepts such as referential integrity, functional dependencies and other constraint patterns to determine the validity of a dataset. Any field that violates these patterns or constraints are then identified as errors.

Integrity constraints are an important tool for ensuring one aspect of data quality, which is data consistency. Integrity constraints were originally designed to enforce data quality by preventing data from becoming inconsistent (M. Volkovs et.al, 2013). This thesis will attempt to apply these qualitative approaches as described above to data sets from IoT devices and examine its results. IoT data storage architecture is usually distributed, thus it is necessary to consider data access in these storage clusters. All of the data resident on the distributed system are periodically synchronized and a mechanism in place, usually ensures that data updates and deletions performed on one node will be automatically reflected on all other nodes in which other components of the data is related. Therefore, to achieve efficient cleaning in a distributed IoT data stream for the purpose of this work, the Apache Spark framework with Optimus will be employed. This provides accessibility to data resident in clusters or across multiple nodes. The Optimus framework (Optimus Documentation Release 2.1.0, 2018) is the missing framework for cleaning, pre-processing and exploring data in a distributed environment. It uses all the power of Apache Spark (optimized via Catalyst) to do so.

In many IoT devices, data acquisition, storage or transmission is a secondary purpose and not the main purpose for their design. For example, while a smart TV might acquire some data intended for the purpose of feedback and design improvement for its manufacturers, its primary function is still to provide entertainment to its users and therefore, the quality of the data it acquires while performing its primary function may not

be of immediate concern. However, in the context of this thesis, acquiring that data and applying qualitative techniques as described above to derive insights for business improvement serves as a central focus.

CHAPTER II

Related Work

In recent data cleaning approaches, the problem of detecting and repairing dirty data using qualitative techniques have focused on using data quality rules [14]. However, many of these data cleaning implementations are applicable only to data that is static or warehoused, not on IoT data that is distributed, and frequently accessed real time or distributed data, which requires a synchronization of data updates. [14] provides a summary of incremental algorithms that detect errors in distributed data when it is updated which is similar to its own implementation of violation detection in a data cleaning system called BLEACH. BLEACH uses dynamic rule management to detect and repair streaming data in a distributed environment by updating each rule to modify tuples in the cleaning process without losing its state. This technique applied to streaming data will be explained further in this section. Apache Spark framework uses machine learning and data science to provide a solution that can connect to any database or file system to handle data cleaning, working in both clusters in a parallelized fashion or locally on a laptop (Optimus Documentation Release 2.1.0, 2018). It has a fast and open source library to prepare and process distributed data such as IoT data using spark and python (PySpark). Finally, (X Chu et.al, 2016) (S Garcia et.al, 2016) (I.F. Ilyas 2015) (A Assadi 2018) detail qualitative techniques using integrity constraints violations and fixing rules for duplication removal and null entry detection all of which are also detailed in this section.

In (A Assadi 2018) qualitative techniques are used to detect and repair errors in a cleaning system called DANCE. DANCE identifies suspicious tuples whose update may contribute to the violation resolution and builds a graph that records the likelihood of an

error in one tuple to occur and affect the other. This approach uses a combination of violation sets and suspicious tuples to build a graph. Violation sets and suspicious tuples are further explained below. These implementations also do not address distributed data from IoT sources but rather for data at rest. This situation presents an opportunity for further research into applying qualitative techniques used on static data for streaming data from IoT devices. Some of the common qualitative techniques are functional dependency and integrity constraint violation, both of these are employed in error detection algorithms. A fixing rule is then developed and employed for error repairing in the same algorithm, one such example that combines fixing rules with integrity constraints is in the commodity cleaning system called NADEEF (N Tang 2014). These constraints violations are further explained below.

Functional Dependency – these are relationships between attributes in a relation, typically between a key attribute and other non-key attributes. An attribute A is said to be functional dependent on another attribute B if for every valid instance of A, the value of A determines the value of B.

(N Tang 2014) focuses on this detection based error-cleaning technique. An illustration is presented to demonstrate how a functional dependency constraint works for error detection. The image below depicts a dataset of travel records;

	name	country	capital	city	conf
r_1:	George	China	Beijing	Beijing	SIGMOD
r_2:	Ian	China	Shanghai (Beijing)	Hongkong (Shanghai)	ICDE
r_3:	Peter	China (Japan)	Tokyo	Tokyo	ICDE
r_4:	Mike	Canada	Toronto (Ottawa)	Toronto	VLDB

Figure 1. Instance of travel schema (N Tang 2014)

The following schema specifies the dataset; **Travel (name, country, capital, city, conference)** where a tuple specifies a person identified by a name, who has traveled to a conference, held at a city in a country with a capital.

The specific error in our example is highlighted as r2[capital] = Shanghai. The correct value is expected to be Beijing. Suppose that a functional dependency (FD) is specified for the Travel table as:

Travel ([country] ⟶ [capital])

which states that country uniquely determines capital. One can verify that in Fig. 1, the two tuples (r1; r2) violate the FD, since they have the same country but carry different capital values, so do (r1; r3) and (r2; r3).

Integrity Constraint Violations - these are rules or criteria that have to be satisfied over a given set of key or non-key attributes. Some of these constraints prevent data duplication or restrict the values that are acceptable in a certain domain. For example, over a data set, NULL values may be defined as restricted. Therefore, integrity constraint **violations** occur when the restriction criteria set over a set of attributes **are** not satisfied. Figure 3 gives an example that shows an integrity constraint violation, syntactic error, duplicates and missing value (X Chu et.al, 2016).

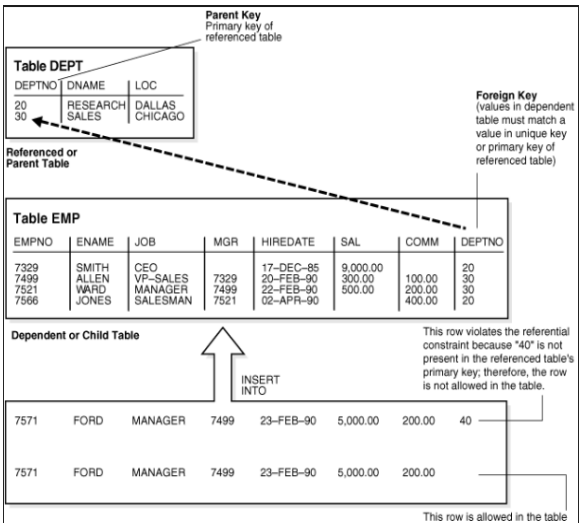


Figure 2. Example of data errors (Google Image: Foreign Key Violation)

Fixing Rules - After errors have been detected, the error observation results in a set of fixing rules, which are incorporated in some cleaning algorithm, such as NADEEF (N Tang 2014). For clarity, a fixing rule contains an evidence pattern, a set of negative patterns and a fact value, which are all combined to indicate how to correct any tuples, detected to have errors. Below is an example of a fixing rule for errors detected in the travel records example (N Tang 2014).

φ_1:

country	$\{capital^-\}$	$capital^+$
China	Shanghai	Beijing
	Hongkong	

φ_2:

country	$\{capital^-\}$	$capital^+$
Canada	Toronto	Ottawa

Figure 3. Fixing Rules (N Tang 2014)

The evidence pattern, negative patterns and fact in this example are China, {Shanghai, Hong Kong}, and Beijing respectively.

Violation Sets - A violation set of a constraint φ in a database D is a minimal set of tuples in D that implies existence of an assignment v (from the tuples values) that is not satisfying the given constraint φ. Intuitively, each violation set is a set of tuples that caused the

database D to violate the constraint φ. For a set of constraints, the violation set is the union of the violation sets of each individual constraint (A Assadi et.al, 2018).

Suspicious Tuples - For a database D and a set *f* of constraints, the set of suspicious tuples includes all the tuples in the proofs, excluding those that have already been validated through update questions. The violation sets are included in the suspicious tuples as they are part of a set of value proofs whose value has not yet been verified.

CHAPTER III

Methodology

3.1 Overview

In accomplishing the objective of this thesis, data set(s) from a single IoT device/source will first be extracted for pre-processing as the first phase of work. The next steps will be to perform error detection by applying the qualitative techniques described in section I, i.e. using referential integrity to define what valid data is and checking for any data with constraints violations. The final phase will be to then use the Optimus framework for data cleaning to perform error repairing on the data set. The results of the data cleaning exercise will then be examined for readiness with respect to suitability for gaining insights that were not readily available at the initial extraction.

The data from a single IoT device acquired at the initial phase will be to test for noise. The intended IoT device data to be used will be from an ATM device. ATM devices capture data real time and online 24 hours of the day as transactions are being carried out. Preliminary research shows that much of the data ATMs capture are for the purpose of reconciliations and payment disputes at the back end. The nature of the data from the source is distributed and in multiple formats as many new ATM machines capture audio, images while recording transactional data from their users. Some of the data usually captured also include diagnostics data while the machines are not in use by customers. This data usually contains the state of the ATM network connection or availability of stationery such as journal rolls or if the reject bins in the machine needs to be emptied by a custodian. Other vital data captured by these machines include the cash levels on each of the cassettes in the machine and rejected transactions perhaps due to suspected fraud or incorrect pin usage.

Once the live/raw data from the ATM device is acquired, qualitative cleaning techniques as described above will be applied to the data using some of the techniques referenced in the related work. The final step will be to feed the pre-processed data into Optimus and examine the results for readiness in terms of business insights such as usage trends, downtime and availability statistics et al.

Due to the cost implication of a live ATM machine, data samples to be used for this thesis will come from free data repositories online such as “redpixie” (free IoT data repository), “re3data” (free IoT data repository)and “data source” (free IoT data repository). Below is a diagrammatic representation of the steps involved in this research.

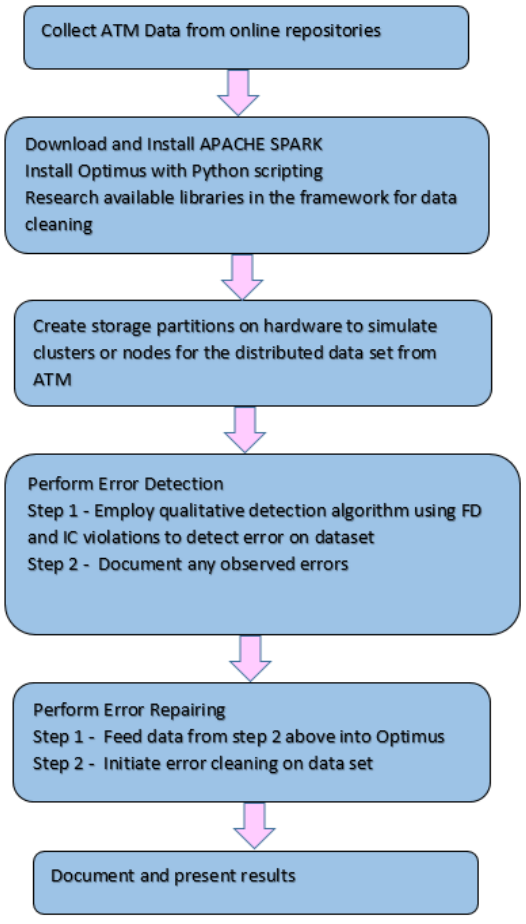


Figure 4. Diagrammatic Representation of Methodology

3.2 Data collection and environment setup

Due to the nature of the data intended for use in this research, multiple data sources were examined online, these searches were narrowed down to three online repositories providing IoT data for different industries ranging from health care to finance. Data from www.re3data.org “re3data” (free IoT data repository) was finally used. This is due to many factors, most important of which was the amount of work required for pre-processing the data in order for it to be best suited for the experimentation. Further details of pre-processing and business rules adopted for the experimentation can be found in Section 4.2. All the software tools used for this work are free for download or trial use online, however the optimus library for cleaning, which was required for use after the error detection stage of the research could not completely install due to unavailable versions of the dependencies required for it to function in the environment. Its use was mainly substituted with the pandas library in python. Physical storage partitions also became unnecessary as pyspark contains internal mechanisms or data distribution using resilient distributed datasets.

3.3 Algorithm

1: **data** ← dataset from csv file

2: isolate account_id column and extract unique ids

3: create empty python dictionary compatible with lists

4: **my_List** ← create second python dictionary for **key/value** pairs of unique account_ids and first occurrences of account_nbr

5: append unique account_ids and 1st occurrences of account_nbr to **my_List**

6: def **Compare** function: ← Function to compare two items in a list

```
        Compare (a, b)
        return (a > b) – (a < b)
7: def func(x): ← Function to handle violations
Counter = 0
#Loop
for value in x.items():
        for i,e in enum(value):
                if Compare(value[0],e) != 0
                        Counter += 1
for values in x.values():
        for ,e in values:
                if Compare(value[0],e) != 0
                        append(e)
return
```

8: **func(data)**

9: aggregate and print violations and append to list in step 3

10: **cleaned_data**): ← data.drop(violations)

11: return **cleaned_data**

3.4 Error detection

In implementing the algorithm for this project, there are 2 steps used in detecting violations in the data. First step considers all cells in the dataset with a null value in the account number column. Since the python pandas library used in developing this detection logic can simply detect null values with the ".isnull()" method, all null values

are selected from the dataset and assigned to a list. Second step uses a “truth list” which is generated based on the functional dependency earlier defined above. Every tuple that is not captured in the truth list is aggregated in a separate violation list which is printed out on the screen for the user while the code is running. These violations do not form part of the exported csv file generated after the dataset has been cleaned. Consider **Figure 8**, account id **2** has an account number **66487163** as its first occurrence. Based on our business rule that the first occurrence of the account number for a unique id is the correct value for the account number, and our FD of [account_id account_nbr], it follows that every tuple in the dataset with an id of **2** but a different account number value from **66487163** is a violation. The “.isnull()” method handles all the null value occurrences in cells for that column, these cells are marked in red in the figure below, while the logic of the truth list excludes the other cells which have values which are not null but not equal to the value of the first occurrence which is **66487163** in this case. See diagrams below for sample dataset error detection and the running code displaying non null value violations in the dataset.

1	trans_id	account_id	account_nbr	date	type	operation	amount	balance	k_symbol	bank
2	276	2	66487163	PRIJEM	VKLAD	1100	1100			
3	279	2	[redacted]	PRIJEM	PREVOD Z	20236	21336		ST	66487163
4	697	2	66487163	PRIJEM	VKLAD	3700	25036			
5	3530483	2	66487163	PRIJEM		13.5	25049.5	UROK		
6	280	2	66487163	PRIJEM	PREVOD Z	20236	45285.5		ST	66487163
7	698	2	[redacted]	VYDAJ	VYBER	11000	34285.5			
8	3530484	2	[redacted]	PRIJEM		109.5	34394.9	UROK		
9	281	2	66487163	PRIJEM	PREVOD Z	20236	54630.9		ST	66487163
10	699	2	[redacted]	VYDAJ	VYBER	17600	37030.9			
11	3530485	2	[redacted]	PRIJEM		144.7	37175.6	UROK		
12	282	2	66487163	PRIJEM	PREVOD Z	30354	67529.6		ST	66487163
13	700	2	66487163	VYDAJ	VYBER	22400	45129.6			
14	3530486	2	66487163	PRIJEM		159.9	45289.5	UROK		
15	355	2	0	VYDAJ	VYBER	3200	42089.5			0
16	283	2	66487163	PRIJEM	PREVOD Z	20236	62325.5		ST	66487163
17	497	2	[redacted]	VYBER	VYBER	13145	49180.5			
18	701	2	[redacted]	VYDAJ	VYBER	10300	38880.5			
19	632	2	[redacted]	VYDAJ	VYBER	14.6	39068.9	SLUZBY		
20	3530487	2	[redacted]	PRIJEM		203	39083.5	UROK		
21	428	2	13943797	VYDAJ	PREVOD N	7266	31802.9	SIPO	QR	13943797
22	356	2	0	VYDAJ	VYBER	2900	28902.9			0
23	284	2	66487163	PRIJEM	PREVOD Z	20236	49138.9		ST	66487163
24	493	2	[redacted]	VYBER	VYBER	21286	27852.9			
25	702	2	[redacted]	VYDAJ	VYBER	1000	26852.9			
26	633	2	[redacted]	VYDAJ	VYBER	14.6	26991.8	SLUZBY		
27	3530488	2	[redacted]	PRIJEM		153.5	27006.4	UROK		
28	429	2	13943797	VYDAJ	PREVOD N	7266	19725.8	SIPO	QR	13943797
29	285	2	66487163	PRIJEM	PREVOD Z	20236	39961.8		ST	66487163
30	357	2	0	VYDAJ	VYBER	3600	36361.8			0

Figure 5. Sample dataset

```
if __name__ == "__main__":
        start = time.time()
        fd(groups(my_dict))
        end = time.time()
        print(end - start)
```

```
For account_id 2 violation detected with account_number 0 in position 15
For account_id 2 violation detected with account_number 13943797 in position 21
For account_id 2 violation detected with account_number 0 in position 22
For account_id 2 violation detected with account_number 13943797 in position 28
For account_id 2 violation detected with account_number 0 in position 30
```

Figure 6. Displaying violations on screen

3.5 Generating repairs over the Dataset

A set of cells for an account_id is clean if the account number value for that cell matches the first occurrence in the truth list. The truth list is generated by populating a dictionary of key/value pairs containing only the unique account_id values and their

corresponding account_number values. Every subsequent account_number is compared to the account number in this dictionary to determine their validity.

A tuple violates the functional dependency [account_id → account_nbr] if its account number does not comply with the first occurrence condition of the business rules as defined or if it is captured in the violation set. The algorithm compiles the truth list and deletes the content of the violations from the original dataset.

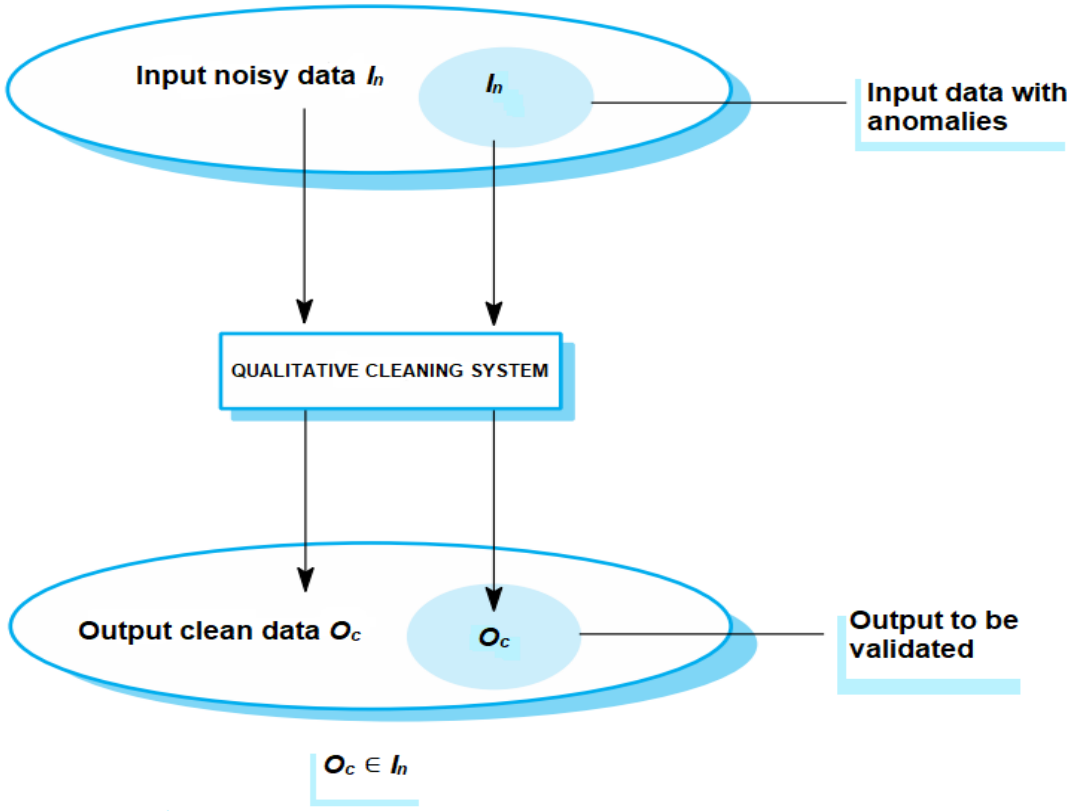


Figure 7. Repair generation

3.6 Serial and Parallel implementation of the Algorithm

Both the serial and parallel implementations use the same error repairing logic, however, different python libraries are employed in executing the functions. For the serial implementation, the libraries include mainly pandas, numpy. The parallel implementation

employs pyspark, which is a combination of python libraries and apache for achieving internal distribution of the imported dataset. Apache uses resilient distributed dataset RDDs to internally create a cluster that houses the dataset in chunks. The parallelized version is achieved by calling the internal SparkContext method, which is imported at source and can be used to create distributed datasets from any storage source or collection including local file systems. The spark.read.csv () method was used in this case and it takes a URI for the file (local path). It reads in the local path and converts it to a collection of lines. The toPandas() method was then used to convert individual routines in the algorithm to make it compatible with the pyspark version in order for the code to be fully functional.

CHAPTER IV

Implementation and Experiment

4.1 Tools and Software

The following software and hardware resources were used to implement the algorithm for detecting errors using functional dependency and cleaning the violations.

Programming Languages.

Python. Python was the language of choice because of its versatility and simplicity of code writing. In addition, it provides a platform for integration of multiple APIs, which makes code integration much faster.

PySpark. PySpark is the python API for Spark. Apache Spark is an open source distributed general-purpose distributed computation environment. The PySpark public classes, one of which is the RDD: (Resilient distributed dataset) provides the representation of the dataset used for this project in a partitioned format, which can be operated on in parallel.

Optimus. Optimus provides a data-cleaning library that was intended for use in inserting value corrections for the detected violations. This feature, although installed was later not used due to dependency challenges.

Software Tools.

Anaconda Navigator. This is an excellent platform to launch python and other python compatible libraries. It also provides multiple libraries and dependencies that can easily be updated or backdated as needed, all within a user friendly interface.

Spark version 2.3.0. Apache Spark is a fast and general-purpose cluster computing system. It provides high-level APIs in Java, Python and an optimized engine

that supports general execution graphs. It also supports a rich set of higher-level tools including Spark SQL and structured data processing.

Jupyter Notebook. The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations and visualizations.

Hardware Tools.

HP Pavilion laptop computer with 1TB of Hard Drive space and 8GB of Random Access Memory. The processor specification is intel Core i3.

4.2 Dataset Preprocessing

The default schema of the dataset sourced from the online repository "re3data" (Free IoT data repository) contains the following attributes as depicted below:

ATM_DATASET (trans_id, account_id, date, type, operation, amount, balance, k_symbol, bank, account). For compactness, certain operations were performed on this dataset such as reduction of the attributes, rearrangement of the attributes by relevance and sorting in ascending order. These operations do not affect the dataset significantly, and were only performed to ensure the data is well ordered before the error detection and cleaning algorithms are applied.

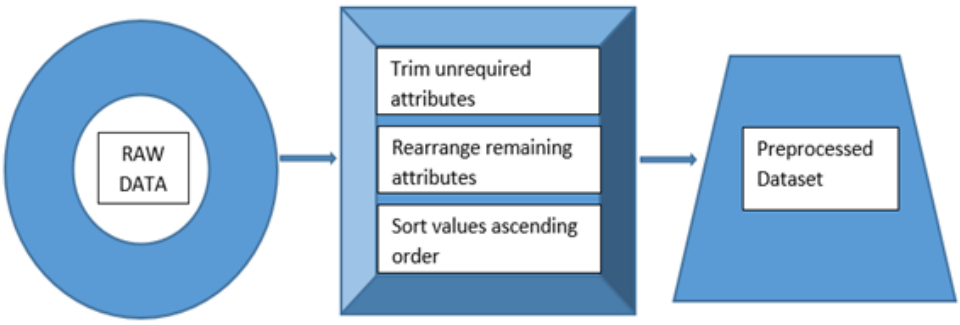


Figure 8. Preprocessed Dataset Generation for Cleaning

Functional Dependency over the Dataset

From definition, a functional dependency [A→B] in a relation holds if two tuples having same value of attribute A also have same value for attribute B. For this project, we are specifying a business rule with the following postulations and functional dependency [account_id → account_nbr]. This therefore creates violations on numerous tuples in the dataset, which forms the basis of the cleaning process.

Postulation

A set of business rules will be followed in defining what constitutes valid data and invalid data. For this project, we are determining validity based on the functional dependency defined above, and determining invalidity with the following application logic. In creating a violation set over the dataset, the first occurrence of any account number value for a given unique account id value will be considered as the correct account number for that account id. All other occurrences not equal to the first occurrence are violations of the dependency. Consider the following;

Account number 66487163 is the correct value for account id 2

Account number 13943797 is the correct value for account id 9

Account number 24265569 is the correct value for account id 15 and so forth.

account_id	account_nbr	
2	66487163	✓
2	*null*	✗
2	6487	✗
9	13943797	✓
9	13943798	✗

9	13947	
15	24265569	
15	*null*	
15	24265	

Figure 9. Instance of dataset

In the algorithm implementation, the first occurrences are all compiled and appended in to a list, which serves as the basis for comparing every other occurrence of an account_id in determining if or not they constitute a violation. A violation set contains all the tuples with account number values that do not match the account number value for the first occurrence of a unique account_id. A depiction of a violation set is as seen in figure 7 below.

account_id	account_nbr	
2	*null*	
2	6487	
9	13943798	
9	13947	
15	*null*	
15	24265	

Figure 10. Violation set

From our business rule above, [account_id → account_nbr] therefore every value of account_nbr for a cell different from the first occurrence of the (account_id/account_nbr) pair will automatically form part of the violation set.

4.3 Results

The experimental results below show that the error detection and error cleaning algorithm are effective with respect to their purpose. Every single error introduced into the dataset was detected in line with the business rules guiding the system. However, the performance of the distributed implementation of the algorithm when evaluated in comparison to the serial implementation shows a sharp divergence. Both implementations result in a linear relationship between the size of the dataset and the execution time, however, it is important to note that the initial assumption was that the distributed implementation of any algorithm is generally expected to produce a reduction in execution time with significant increases in the size of the dataset inputted into the system.

Table 1.

Table of results

Size	Serial time	Parallel time	Errors Introduced	Errors detected	Detection efficiency
500 rows	0.0269	4.247	212	212	100%
1,000 rows	0.0239	4.291	696	696	100%
30,000 rows	0.123	5.942	22,272	22,272	100%
65,000 rows	0.234	7.57	44,540	44,540	100%
100,000 rows	0.319	8.992	58,612	58,612	100%
125,000 rows	0.47	10.777	89,088	89,088	100%
200,000 rows	0.792	15.113	164,213	164,213	100%
250,000 rows	1.002	18.259	203,818	203,818	100%

325,000 rows	1.269	23.562	261,113	261,113	100%
450,000 rows	1.642	29.421	363,520	363,520	100%
500,000 rows	1.863	31.51	375,913	375,913	100%
1m+ rows	4.863	56.702	720,103	720,103	100%

Error detection

```
In [8]: if __name__ == "__main__":
            start = time.time()
            fd(groups(my_dict))
            end = time.time()
            print(end - start)
```

```
For account_id 2 violation detected with account_number 0 in position 15
For account_id 2 violation detected with account_number 13943797 in position 21
For account_id 2 violation detected with account_number 0 in position 22
For account_id 2 violation detected with account_number 13943797 in position 28
For account_id 2 violation detected with account_number 0 in position 30
For account_id 2 violation detected with account_number 0 in position 35
For account_id 2 violation detected with account_number 13943797 in position 36
For account_id 2 violation detected with account_number 13943797 in position 41
For account_id 2 violation detected with account_number 0 in position 43
For account_id 2 violation detected with account_number 13943797 in position 47
For account_id 2 violation detected with account_number 0 in position 48
For account_id 2 violation detected with account_number 13943797 in position 56
For account_id 2 violation detected with account_number 89597016 in position 58
For account_id 2 violation detected with account_number 13943797 in position 66
For account_id 2 violation detected with account_number 89597016 in position 68
For account_id 2 violation detected with account_number 13943797 in position 72
For account_id 2 violation detected with account_number 89597016 in position 74
For account_id 2 violation detected with account_number 13943797 in position 78
For account_id 2 violation detected with account_number 89597016 in position 80
For account_id 2 violation detected with account_number 13943797 in position 84
For account_id 2 violation detected with account_number 89597016 in position 85
For account_id 2 violation detected with account_number 0 in position 106
For account_id 2 violation detected with account_number 13943797 in position 112
For account_id 2 violation detected with account_number 0 in position 113
For account_id 2 violation detected with account_number 13943797 in position 119
```

Figure 11. Error detection in execution

Error detection with time lapse

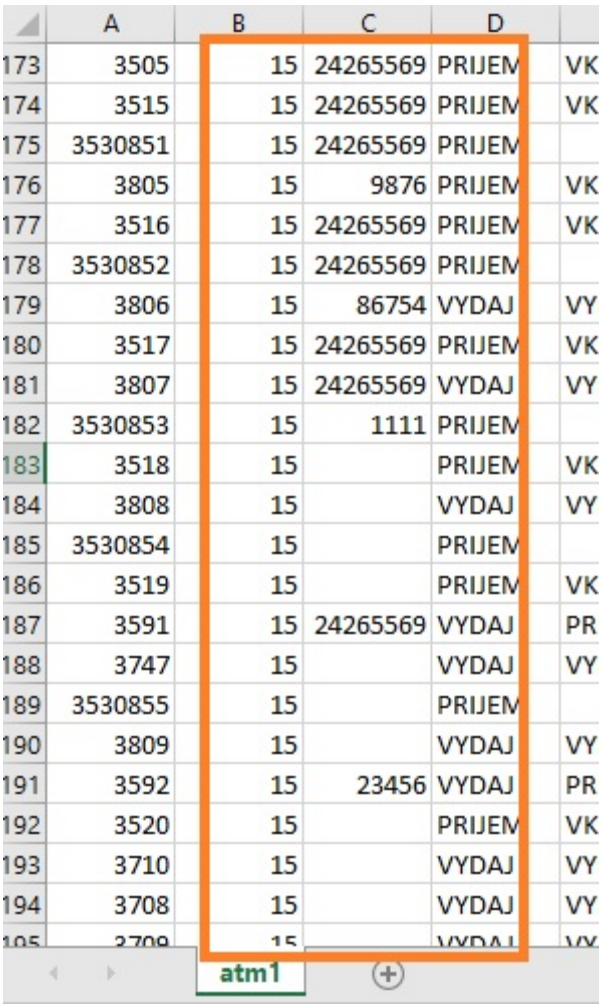
```
For account_id 34 violation detected with account_number 19993174 in position 20
For account_id 34 violation detected with account_number 58999049 in position 23
For account_id 34 violation detected with account_number 19993174 in position 27
For account_id 34 violation detected with account_number 58999049 in position 28
For account_id 34 violation detected with account_number 19993174 in position 34
For account_id 34 violation detected with account_number 58999049 in position 36
For account_id 34 violation detected with account_number 19993174 in position 43
For account_id 34 violation detected with account_number 58999049 in position 45
For account_id 34 violation detected with account_number 58999049 in position 60
For account_id 34 violation detected with account_number 19993174 in position 64
For account_id 34 violation detected with account_number 58999049 in position 67
For account_id 34 violation detected with account_number 19993174 in position 71
For account_id 34 violation detected with account_number 58999049 in position 72
For account_id 34 violation detected with account_number 19993174 in position 78
For account_id 34 violation detected with account_number 58999049 in position 80
For account_id 34 violation detected with account_number 19993174 in position 87
For account_id 34 violation detected with account_number 58999049 in position 89
For account_id 35 violation detected with account_number 99848401 in position 8
For account_id 35 violation detected with account_number 56788617 in position 9
For account_id 35 violation detected with account_number 99848401 in position 16
For account_id 35 violation detected with account_number 56788617 in position 17
For account_id 485 violation detected with account_number 4968433 in position 5
For account_id 485 violation detected with account_number 4968433 in position 10
For account_id 1844 violation detected with account_number 8 in position 5
For account_id 1844 violation detected with account_number 4298841 in position 7
For account_id 1844 violation detected with account_number 8 in position 11
For account_id 1844 violation detected with account_number 4298841 in position 13
There are 696 account number violations and null values in the dataset
0.27187609672546387
```

Figure 12. Error detection in execution with time stamp

Dirty dataset sample

	trans_id	account_i	account_n	date	ty
2	276	2	66487163	PRIJEM	VK
3	279	2		PRIJEM	PF
4	697	2	66487163	PRIJEM	VK
5	3530483	2	66487163	PRIJEM	
6	280	2	66487163	PRIJEM	PF
7	698	2		VYDAJ	VY
8	3530484	2		PRIJEM	
9	281	2	66487163	PRIJEM	PF
10	699	2		VYDAJ	VY
11	3530485	2		PRIJEM	
12	282	2	66487163	PRIJEM	PF
13	700	2	66487163	VYDAJ	VY
14	3530486	2	66487163	PRIJEM	
15	355	2	0	VYDAJ	VY
16	283	2	66487163	PRIJEM	PF
17	497	2		VYBER	VY
18	701	2		VYDAJ	VY
19	632	2		VYDAJ	VY
20	3530487	2		PRIJEM	
21	428	2	13943797	VYDAJ	PF
22	356	2	0	VYDAJ	VY
23	284	2	66487163	PRIJEM	PF

atm1



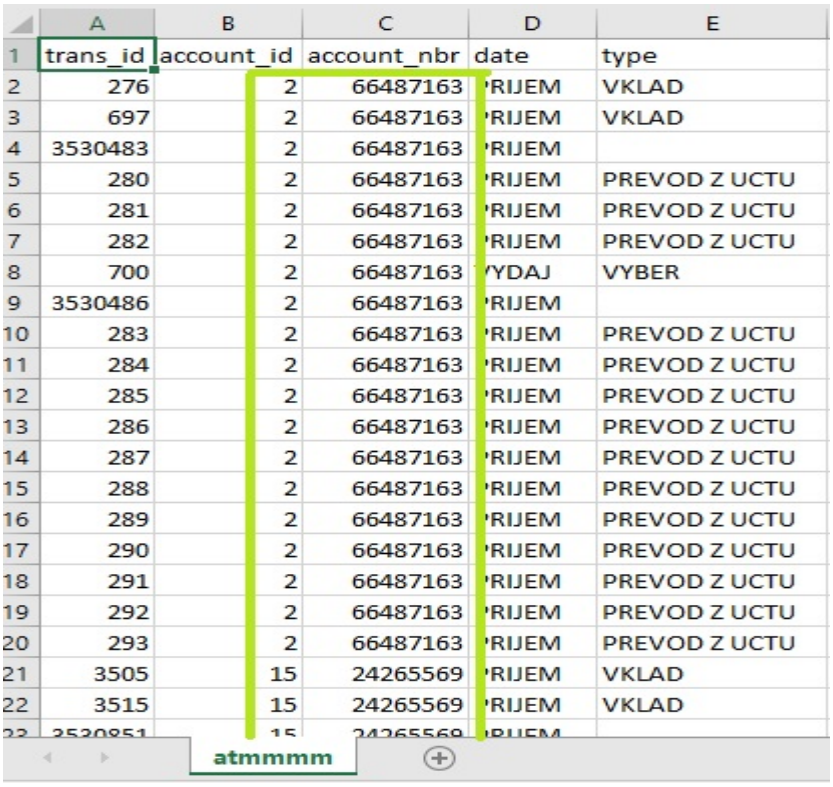
	A	B	C	D	
173	3505	15	24265569	PRIJEM	VK
174	3515	15	24265569	PRIJEM	VK
175	3530851	15	24265569	PRIJEM	
176	3805	15	9876	PRIJEM	VK
177	3516	15	24265569	PRIJEM	VK
178	3530852	15	24265569	PRIJEM	
179	3806	15	86754	VYDAJ	VY
180	3517	15	24265569	PRIJEM	VK
181	3807	15	24265569	VYDAJ	VY
182	3530853	15	1111	PRIJEM	
183	3518	15		PRIJEM	VK
184	3808	15		VYDAJ	VY
185	3530854	15		PRIJEM	
186	3519	15		PRIJEM	VK
187	3591	15	24265569	VYDAJ	PR
188	3747	15		VYDAJ	VY
189	3530855	15		PRIJEM	
190	3809	15		VYDAJ	VY
191	3592	15	23456	VYDAJ	PR
192	3520	15		PRIJEM	VK
193	3710	15		VYDAJ	VY
194	3708	15		VYDAJ	VY
195	3709	15		VYDAJ	VY

atm1

Figure 13. Sample dirty input

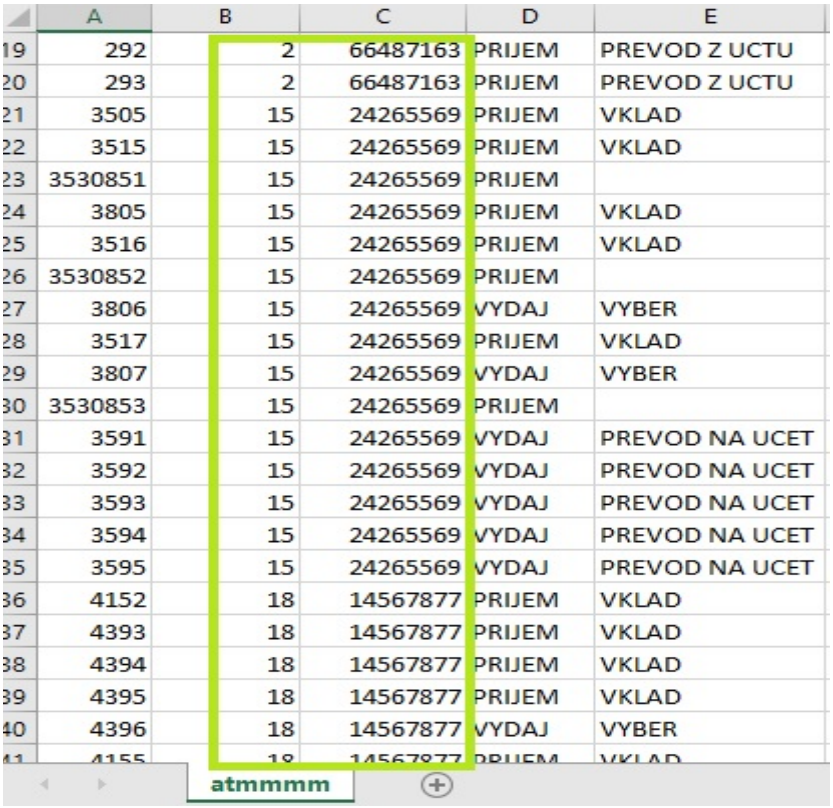
Every occurrence of a unique account ID. must have a unique account number associated with it, in which case has to be the first occurrence of the account number for that ID. For this sample dataset, account ID ‘2’ must have an account number of ‘66487163’ for every occurrence of that account ID.

Cleaned output sample



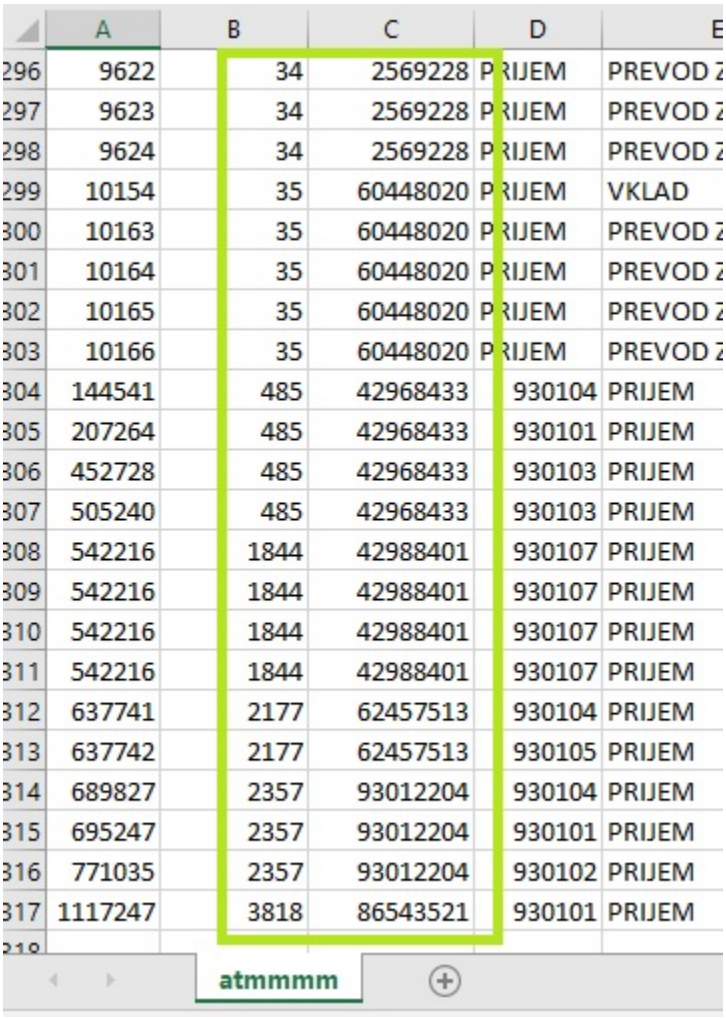
	A	B	C	D	E
1	trans_id	account_id	account_nbr	date	type
2	276	2	66487163	PRIJEM	VKLAD
3	697	2	66487163	PRIJEM	VKLAD
4	3530483	2	66487163	PRIJEM	
5	280	2	66487163	PRIJEM	PREVOD Z UCTU
6	281	2	66487163	PRIJEM	PREVOD Z UCTU
7	282	2	66487163	PRIJEM	PREVOD Z UCTU
8	700	2	66487163	VYDAJ	VYBER
9	3530486	2	66487163	PRIJEM	
10	283	2	66487163	PRIJEM	PREVOD Z UCTU
11	284	2	66487163	PRIJEM	PREVOD Z UCTU
12	285	2	66487163	PRIJEM	PREVOD Z UCTU
13	286	2	66487163	PRIJEM	PREVOD Z UCTU
14	287	2	66487163	PRIJEM	PREVOD Z UCTU
15	288	2	66487163	PRIJEM	PREVOD Z UCTU
16	289	2	66487163	PRIJEM	PREVOD Z UCTU
17	290	2	66487163	PRIJEM	PREVOD Z UCTU
18	291	2	66487163	PRIJEM	PREVOD Z UCTU
19	292	2	66487163	PRIJEM	PREVOD Z UCTU
20	293	2	66487163	PRIJEM	PREVOD Z UCTU
21	3505	15	24265569	PRIJEM	VKLAD
22	3515	15	24265569	PRIJEM	VKLAD

atmmmm



	A	B	C	D	E
19	292	2	66487163	PRIJEM	PREVOD Z UCTU
20	293	2	66487163	PRIJEM	PREVOD Z UCTU
21	3505	15	24265569	PRIJEM	VKLAD
22	3515	15	24265569	PRIJEM	VKLAD
23	3530851	15	24265569	PRIJEM	
24	3805	15	24265569	PRIJEM	VKLAD
25	3516	15	24265569	PRIJEM	VKLAD
26	3530852	15	24265569	PRIJEM	
27	3806	15	24265569	VYDAJ	VYBER
28	3517	15	24265569	PRIJEM	VKLAD
29	3807	15	24265569	VYDAJ	VYBER
30	3530853	15	24265569	PRIJEM	
31	3591	15	24265569	VYDAJ	PREVOD NA UCET
32	3592	15	24265569	VYDAJ	PREVOD NA UCET
33	3593	15	24265569	VYDAJ	PREVOD NA UCET
34	3594	15	24265569	VYDAJ	PREVOD NA UCET
35	3595	15	24265569	VYDAJ	PREVOD NA UCET
36	4152	18	14567877	PRIJEM	VKLAD
37	4393	18	14567877	PRIJEM	VKLAD
38	4394	18	14567877	PRIJEM	VKLAD
39	4395	18	14567877	PRIJEM	VKLAD
40	4396	18	14567877	VYDAJ	VYBER

atmmmm



	A	B	C	D	E
296	9622	34	2569228	PRIJEM	PREVOD Z
297	9623	34	2569228	PRIJEM	PREVOD Z
298	9624	34	2569228	PRIJEM	PREVOD Z
299	10154	35	60448020	PRIJEM	VKLAD
300	10163	35	60448020	PRIJEM	PREVOD Z
301	10164	35	60448020	PRIJEM	PREVOD Z
302	10165	35	60448020	PRIJEM	PREVOD Z
303	10166	35	60448020	PRIJEM	PREVOD Z
304	144541	485	42968433	930104	PRIJEM
305	207264	485	42968433	930101	PRIJEM
306	452728	485	42968433	930103	PRIJEM
307	505240	485	42968433	930103	PRIJEM
308	542216	1844	42988401	930107	PRIJEM
309	542216	1844	42988401	930107	PRIJEM
310	542216	1844	42988401	930107	PRIJEM
311	542216	1844	42988401	930107	PRIJEM
312	637741	2177	62457513	930104	PRIJEM
313	637742	2177	62457513	930105	PRIJEM
314	689827	2357	93012204	930104	PRIJEM
315	695247	2357	93012204	930101	PRIJEM
316	771035	2357	93012204	930102	PRIJEM
317	1117247	3818	86543521	930101	PRIJEM

atmmmm

Figure 14. Sample cleaned output

Based on the specified FD over this dataset, every occurrence of a unique account ID value has a corresponding account number value which is constant over every such occurrence.

4.4 Analysis

While the overall goal of this project is to explore the feasibility of using qualitative techniques to detect errors in a dataset, the algorithm designed to clean the detected errors on varying sizes of datasets fed into the system also has to be evaluated. The variations in

the performance of this algorithm in both a serial implementation and a parallel implementation are presented below on increasing dataset sizes.

Analysis on implementation for a range of 500 to 65,000 tuples

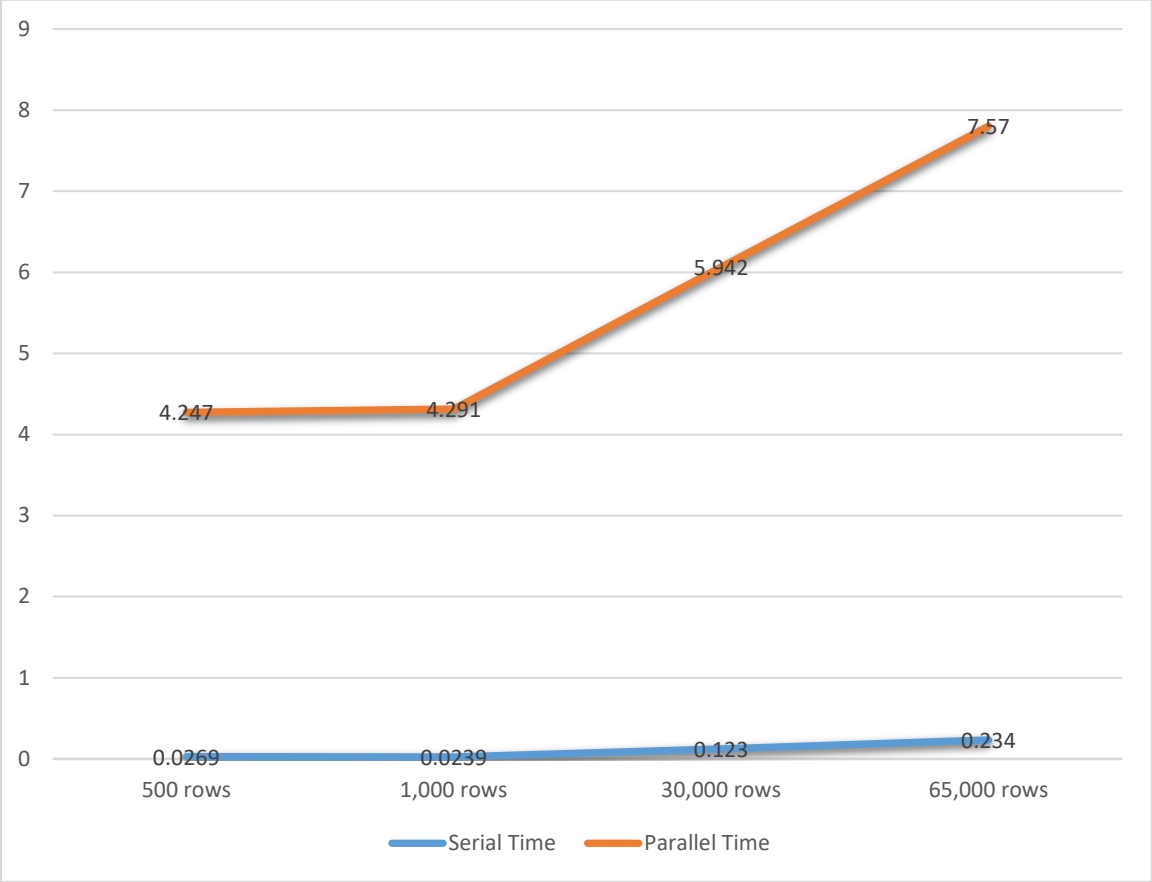


Figure 15. Serial vs parallel implementation of algorithm

Analysis on implementation for a range of 100,000 to 250,000 tuples

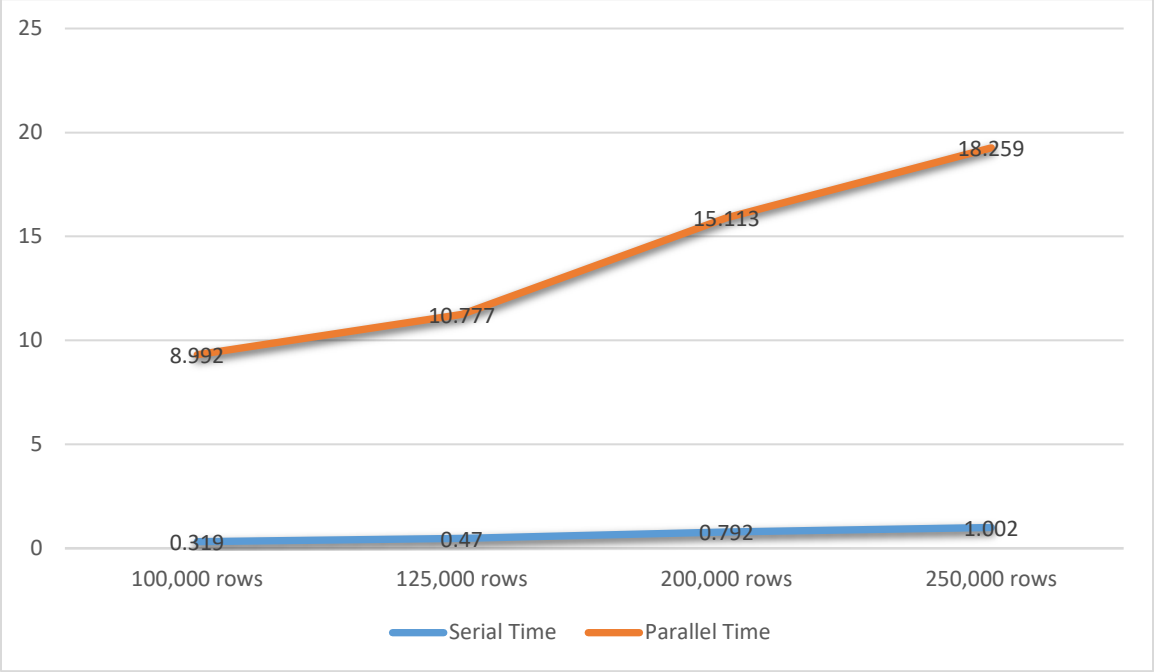


Figure 16. Serial vs parallel implementation of algorithm

Analysis on implementation for a range of 325,000 to 1,000,000 tuples

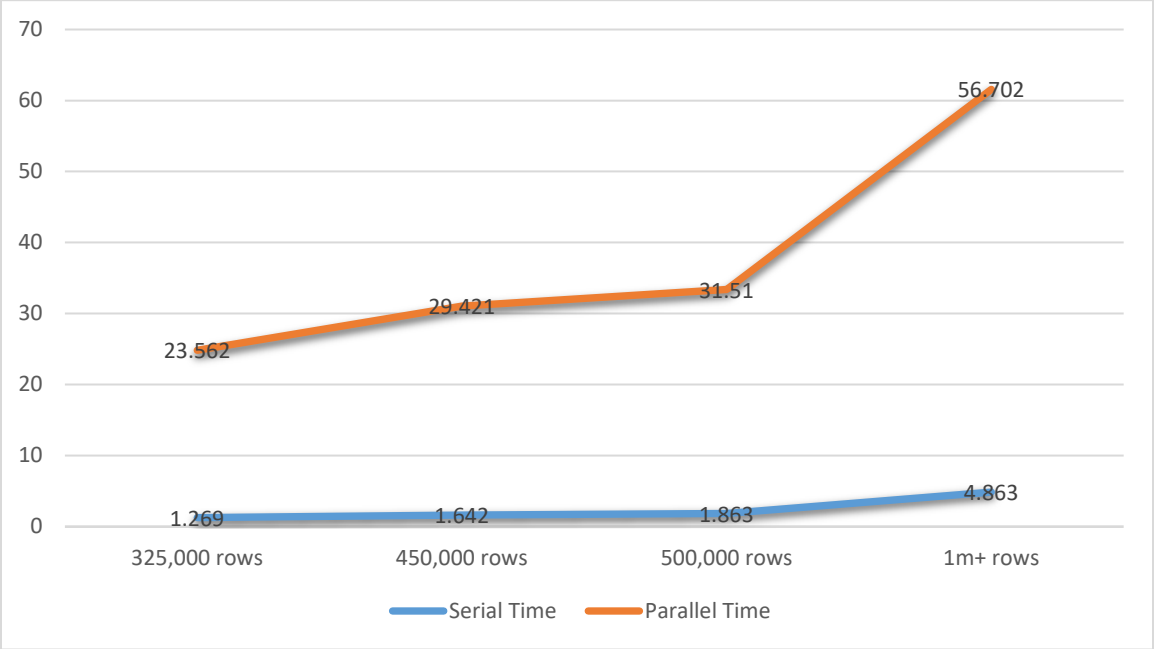


Figure 17. Serial vs parallel implementation of algorithm

4.5 Validation

Most of the existing literature in the data-cleaning field employ quantitative techniques in error cleaning, and as such, very few literatures exist that employ qualitative algorithms in error detection and removal. Furthermore, lesser research works exist that combine the qualitative technique described in this research on IoT datasets. Below is a tabular comparison of our results with respect to existing research in this field;

Table 2.

Validation of results

	NADEEF(N Tang 2014)	DANCE(A Assadi 2018)	THIS RESEARCH
Application to IoT Streaming data	✓	-	✓
Algorithm execution	-	-	linear time
Distribution mechanism	Clustering	-	Internal, through PySpark RDDs*. Hardware clusters not required
Error Detection Accuracy	-	-	100%

• (continued)

Fixing rules specified over dataset	• Integrity Constraints • Contextual functional dependency • Suspicious tuples • Violation sets	Human experts required	• 1 FD Rule • Business rules
Deployment	Easy to deploy	-	Easy to deploy
Human manipulation	Automated	Human involvement	Automated
Tool set	Proprietary	Proprietary	Python, Apache, Windows/Linux
Development environment	Proprietary	Proprietary	Open source
Repeatability of experiment	✓	-	✓
Data size	Unspecified	Unspecified	500 – 1million rows
Running time of 1m+ dataset size	-	-	• Serial– 4.863 • Distributed- 56.702

*RDD – Resilient distributed datasets (PYSPARK DOCUMENTATION REFERENCE)

*Linear – Running time grows at most linearly with the size of the input for worst case

*FD – Functional Dependency

CHAPTER V

Summary and Remark

A summary of the findings for this research is presented in this chapter as well as certain limitations, which were not earlier anticipated during background studies. The future directions of this research will also be discussed.

5.1 Summary

In this project, we examined various approaches for data cleaning, many of which utilize either quantitative or qualitative cleaning techniques using established algorithms and systems. We further narrowed down on these techniques by discussing how qualitative error detection techniques can be used on a slightly different form of datasets, which are distributed in nature and a typical source of these datasets are from IoT devices or other forms of data streams.

We depicted this technique in use over a dataset from the finance industry using bank ATM data sourced from an online repository containing over a million records. The main purpose of this approach was to show that qualitative cleaning techniques, specifically, functional dependency constraints, can be applied on these types of datasets with an optimal performance.

The performance of this implementation is measured over two metrics, correctness and time of execution. The correctness is justified by the efficiency of error detection i.e. the number of errors introduced into the dataset before it is inputted into the application and the number of errors the application detects. This currently stands at a 100% success rate. The trade off for this approach is the time it takes in performing error cleaning on datasets with sizes of over one million records. Table 1. gives a

comprehensive breakdown of the performance measurements while Figure 9. to Figure 12. provide a graphical analysis.

Overall, the aim of this research was largely accomplished based on the correctness of the error detection algorithm and the performance of the error cleaning process. However, certain limitations that exist are discussed in the next section.

5.2 Limitations

We specified a fixed data file type - ".csv" - for the input files in order to render the input compatible with the programming language. This can be considered as a limitation, which requires other data sources to be converted first into this format. The business rules specified in order to establish the functional dependency over the attributes of the dataset require a knowledge of databases, this is not anticipated to be a major challenge drawback to the use of the system.

In addition, the processing time for the distributed version of the algorithm is considerably high for large datasets. This is suspected to be due to the number of loops in the algorithm or an undetected master/slave connection delay in the spark cluster.

5.3 Future Work

This research can be extended to take data in any format. In addition, the performance of the algorithm with respect to time is an area that needs improvement. Regardless of the dataset size, it is a reasonable expectation that distributed processing should provide an output possibly within a unit time of microseconds. Also, this research can be expanded to allow processing of real time data that is constantly being updated in a production environment. The algorithm can dynamically detect errors in these

environments and pass the cleaned data to external servers that feed the data to live systems that perform data analytics.

REFERENCES

J. M. Hellerstein. 2008. Quantitative data cleaning for large databases. *United Nations Economic Commission for Europe (UNECE).*

OPTIMUS Documentation Release 2.1.0, Oct 3, 2018. Retrieved from https://media.readthedocs.org/pdf/Optimus-ironmussa/latest/Optimus-ironmussa.pdf

X Chu and I.F. Ilyas 2016 'Qualitative Data Cleaning', *University of Waterloo, September*

M. Volkovs, F Chiang, J Szlichta, R. J. Miller 2013 'Continuous Data Cleaning'. *University of Toronto, McMaster University.*

N Tang, 2014 'Big Data Cleaning', *Qatar Computing Research Institute, Doha Qatar.*

X Chu and I.F. Ilyas, S Krishnan, J Wang. 2016 'Data Cleaning: *Overview and Emerging Challenges'. University of Waterloo, UC Berkeley, Simon Fraser University.*

S Garcia, S Ramirez-Gallego, J Luengo, J Manuel Benitez, F Herrera. 2016 '*Big Data Preprocessing: methods and prospects'. University of Granada.*

I.F. Ilyas, 2015 'Effective Data Cleaning with Continuous Evaluation'. *University of Waterloo.*

A Assadi, T Milo, S Novgorodov. June 2018 'Cleaning Data with Constraints and Experts'. *Tel Aviv University.*

Free IoT data repository – Retrieved from 'https://www.re3data.org/search?query=&contentTypes%5B%5D=Raw%20data&contentTypes%5B%5D=Audiovisual%20data'

Free IoT data repository. Retrieved from https://www.redpixie.com/blog/iot-big-data-datasets-finance

Free IoT data repository. Retrieved from https://datasource.kapsarc.org/explore/dataset/atm-stats/api/?disjunctive.periodicity&disjunctive.quarter&disjunctive.month&disjunctive.indicator&sort=time_period&dataChart=eyJxdWVyaWVzIjpbeyJjaGFydHMiOlt7InR5cGUiOiJjb2x1bW4iLCJmdW5jIjoiQVZHIiwieUF4aXMiOiJpbmRpY2F0b3JfdmFsdWUiLCJzY2llbnRpZmljRGlzcGxheSI6dHJ1ZSwiY29sb3IiOiJyYW5nZS1BY2NlbnQifV0sInhBeGlzIjoidGltZV9wZXJpb2QiLCJtYXhhb2ludHMiOm51bGwsInNvcnQiOiIiLCJ0aW1lc2NhbGUiOiJ5ZWFyIiwic2VyaWVzQnJlYWtkb3duIjoiaW5kaWNhdG9yIiwic3RhY2tlZCI6Im5vcm1hbCIsImNvbmZpZyI6eyJkYXRhc2V0IjoiYXRtLXN0YXRzIiwib3B0aW9ucyI6eyJkaXNqdW5jdGl2ZS5wZXJpb2RpY2l0eSI6dHJ1ZSwiZGlzanVuY3RpdmUucXVhcnRlciI6dHJ1ZSwiZGlzanVuY3RpdmUubW9udGgiOnRydWUsImRpc2p1bmN0aXZlLmluZGljYXRvciI6dHJ1ZSwic29ydCI6InRpbWVfcGVyaW9kIn19fV0sInRpbWVzY2FsZSI6IiIsImRpc3BsYXlMZWdlbmQiOnRydWUsImFsaWduTW9udGgiOnRydWV9 (accessed 2019).

V.J Maccio, F Chiang, D.G Down, 2014 ‘Models for Distributed, Large scale Data Cleaning’. *McMaster University, Ontario Canada.*

Y Tian, P Michiardi, M Vukolic, June 2017 ‘A Distributed Stream Data Cleaning System’. *Eurecom & IBM Research – Zurich.*

Pyspark 2.4.0 Documentation – Apache Spark. Retrieved from https://spark.apache.org/docs/latest/api/python/pyspark.html#pyspark.SparkContext

Pyspark 2.4.0 Resilient distributed dataset – Retrieved from https://spark.apache.org/docs/latest/api/python/pyspark.html#pyspark.RDD

W Fan, F Geerts, X Jia. 2008 ‘A Revival of Integrity Constraints for Data Cleaning’. *University of Edinburgh. Bell Laboratories*

H Muller, J-C Freytag. 2005 ‘Problems, Methods and Challenges in Comprehensive Data Cleansing’. *Humboldt-Universitat zu Berlin, Berlin Germany*

E Rahm, H H Do. 2000 ‘Data Cleaning: Problems and Current Approaches’. *University of Leipzig, Germany*

J I Maletic, A Marcus. 2009 ‘Data Cleansing: A Prelude to Knowledge Discovery’. *Kent State University. Wayne State University*

S Devi, Dr. A Kalia. March 2015 ‘Study of Data Cleaning & Comparison of Data Cleaning Tools’. *International Journal of Computer Science and Mobile Computing Vol. 4, Issue. 3 ISBN 2320-088X. Himachal Pradesh University Shimla, India.*

P Jermyn, M Dixon, B J Read. 1999 ‘Preparing Clean Views of Data for Data Mining’. *CISM, London Guildhall University. ITD, CLRC Rutherford Appleton Laboratory*

K-B Yue. 2012 ‘A Realistic Data Cleansing and Preparation Project’. *Department of Computer Information Systems. University of Houston, Clear Lake Houston Tx USA. Journal of Information Systems Education, Vol. 23(2)*

JVD Broeck, SA Cunningham, R Eeckels, K Herbst. 2005 ‘Data Cleaning: Detecting, Diagnosing and Editing Data Abnormalities’. *PLos journals.plos.org*

Y. Huhtala, J. Karkkainen, P. Porkka, and H. Toivonen. 1999. ‘TANE: An efficient algorithm for discovering functional and approximate dependencies’. *Computer Journal, 42(2):100–111.*

APPENDIX

Repository

- Final - https://github.com/georgeayodeji/Functional-dependency/blob/master/modified%20fd.ipynb
- Parallel – https://github.com/georgeayodeji/Functional-dependency/blob/master/PARALLEL%20FD.ipynb
- Serial - https://github.com/georgeayodeji/Functional-dependency/blob/master/SERIAL%20FD.ipynb

Schema

trans_id	account_id	account_nbr	date	type	amount	balance	bank
276	2	66487163	PRIJEM	VKLAD	1100		
279	2		PRIJEM	PREVOD Z UCTU	21336		66487163
697	2	66487163	PRIJEM	VKLAD	25036		
3530483	2	66487163	PRIJEM		25049.5	UROK	
280	2	66487163	PRIJEM	PREVOD Z UCTU	45285.5		66487163
698	2		VYDAJ	VYBER	34285.5		
3530484	2		PRIJEM		34394.9	UROK	
281	2	66487163	PRIJEM	PREVOD Z UCTU	54630.9		66487163
699	2		VYDAJ	VYBER	37030.9		
3530485	2		PRIJEM		37175.6	UROK	
282	2	66487163	PRIJEM	PREVOD Z UCTU	67529.6		66487163
700	2	66487163	VYDAJ	VYBER	45129.6		
3530486	2	66487163	PRIJEM		45289.5	UROK	
355	2	0	VYDAJ	VYBER	42089.5		0
283	2	66487163	PRIJEM	PREVOD Z UCTU	62325.5		66487163

497	2		VYBER	VYBER	49180.5		
701	2		VYDAJ	VYBER	38880.5		
632	2		VYDAJ	VYBER	39068.9	SLUZBY	
3530487	2		PRIJEM		39083.5	UROK	
428	2	13943797	VYDAJ	PREVOD NA UCET	31802.9	SIPO	13943797
356	2	0	VYDAJ	VYBER	28902.9		0
284	2	66487163	PRIJEM	PREVOD Z UCTU	49138.9		66487163
493	2		VYBER	VYBER	27852.9		
702	2		VYDAJ	VYBER	26852.9		
633	2		VYDAJ	VYBER	26991.8	SLUZBY	
3530488	2		PRIJEM		27006.4	UROK	
429	2	13943797	VYDAJ	PREVOD NA UCET	19725.8	SIPO	13943797
285	2	66487163	PRIJEM	PREVOD Z UCTU	39961.8		66487163
357	2	0	VYDAJ	VYBER	36361.8		0
703	2		VYDAJ	VYBER	31861.8		
592	2		VYDAJ	VYBER	21661.8		
634	2		VYDAJ	VYBER	21774	SLUZBY	
3530489	2		PRIJEM		21788.6	UROK	
358	2	0	VYDAJ	VYBER	17774		0
430	2	13943797	VYDAJ	PREVOD NA UCET	10508	SIPO	13943797
286	2	66487163	PRIJEM	PREVOD Z UCTU	30744		66487163
704	2		VYDAJ	VYBER	27744		
635	2		VYDAJ	VYBER	27830.4	SLUZBY	
3530490	2		PRIJEM		27845	UROK	
431	2	13943797	VYDAJ	PREVOD NA UCET	20564.4	SIPO	13943797
287	2	66487163	PRIJEM	PREVOD Z UCTU	40800.4		66487163
359	2	0	VYDAJ	VYBER	37000.4		0
705	2		VYDAJ	VYBER	31700.4		
636	2		VYDAJ	VYBER	31824.9	SLUZBY	
3530491	2		PRIJEM		31839.5	UROK	
432	2	13943797	VYDAJ	PREVOD NA UCET	24558.9	SIPO	13943797

360	2	0	VYDAJ	VYBER	21058.9		0
546	2		VYDAJ	VYBER	16858.9		
288	2	66487163	PRIJEM	PREVOD Z UCTU	47212.9		66487163
590	2		VYDAJ	VYBER	44212.9		
495	2		VYBER	VYBER	30816.9		
706	2		VYDAJ	VYBER	27716.9		
637	2		VYDAJ	VYBER	27840.6	SLUZBY	
3530492	2		PRIJEM		27855.2	UROK	
433	2	13943797	VYDAJ	PREVOD NA UCET	20574.6	SIPO	13943797
289	2	66487163	PRIJEM	PREVOD Z UCTU	40810.6		66487163
361	2	89597016	VYDAJ	PREVOD NA UCET	35337.9	UVER	8959701 6
563	2		VYDAJ	VYBER	38710.6		
560	2		VYDAJ	VYBER	32937.9		
561	2		VYDAJ	VYBER	26937.9		
707	2		VYDAJ	VYBER	26537.9		
562	2		VYDAJ	VYBER	19937.9		
638	2		VYDAJ	VYBER	20038.7	SLUZBY	
3530493	2		PRIJEM		20053.3	UROK	
434	2	13943797	VYDAJ	PREVOD NA UCET	12772.7	SIPO	13943797
290	2	66487163	PRIJEM	PREVOD Z UCTU	33008.7		66487163
362	2	89597016	VYDAJ	PREVOD NA UCET	29636	UVER	8959701 6
708	2		VYDAJ	VYBER	26136		
639	2		VYDAJ	VYBER	26229.6	SLUZBY	
3530494	2		PRIJEM		26244.2	UROK	
435	2	13943797	VYDAJ	PREVOD NA UCET	18963.6	SIPO	13943797
291	2	66487163	PRIJEM	PREVOD Z UCTU	39199.6		66487163
363	2	89597016	VYDAJ	PREVOD NA UCET	35826.9	UVER	8959701 6
709	2		VYDAJ	VYBER	30926.9		
640	2		VYDAJ	VYBER	31041.5	SLUZBY	
3530495	2		PRIJEM		31056.1	UROK	

436	2	13943797	VYDAJ	PREVOD NA UCET	23775.5	SIPO	13943797
292	2	66487163	PRIJEM	PREVOD Z UCTU	44011.5		66487163
364	2	89597016	VYDAJ	PREVOD NA UCET	40638.9	UVER	89597016
710	2		VYDAJ	VYBER	33038.9		
641	2		VYDAJ	VYBER	33174.2	SLUZBY	
3530496	2		PRIJEM		33188.8	UROK	
437	2	13943797	VYDAJ	PREVOD NA UCET	25908.2	SIPO	13943797
365	2	89597016	VYDAJ	PREVOD NA UCET	42771.5	UVER	89597016
293	2	66487163	PRIJEM	PREVOD Z UCTU	46144.2		66487163
602	2		VYDAJ	VYBER	33471.5		
711	2		VYDAJ	VYBER	28571.5		
599	2		VYDAJ	VYBER	20171.5		
642	2		VYDAJ	VYBER	20293.4	SLUZBY	
3530497	2		PRIJEM		20308	UROK	
438	2		VYDAJ	PREVOD NA UCET	13027.4	SIPO	13943797
2176	9	13943797	PRIJEM	VKLAD	400		
2190	9		PRIJEM	VKLAD	1200		
2432	9		PRIJEM	VKLAD	12100		
3459107	9		PRIJEM		12105.4	UROK	
2191	9	13943797	PRIJEM	VKLAD	12905.4		
2433	9	13943797	PRIJEM	VKLAD	18705.4		
3459108	9	13943797	PRIJEM		18759.4	UROK	
2192	9	13943797	PRIJEM	VKLAD	19559.4		
2178	9	13943797	PRIJEM	VKLAD	33804.4		
2434	9	13943797	VYDAJ	VYBER	27004.4		
3459109	9	13943797	PRIJEM		27089	UROK	

2193	9	13943797	PRIJEM	VKLAD	27889		
2435	9	13943797	VYDAJ	VYBER	23489		
3459110	9	13943797	PRIJEM		23599.2	UROK	
2365	9	13943797	VYDAJ	VYBER	23584.6	SLUZBY	
2194	9	13943797	PRIJEM	VKLAD	24384.6		
2273	9	13943797	VYDAJ	VYBER	15484.6		
2436	9	13943797	PRIJEM	VKLAD	21084.6		
3459111	9	13943797	PRIJEM		21171	UROK	
2366	9	13943797	VYDAJ	VYBER	21156.4	SLUZBY	
2195	9	13943797	PRIJEM	VKLAD	21956.4		
2437	9	13943797	VYDAJ	VYBER	20856.4		
2328	9	13943797	VYDAJ	VYBER	19606.4		
3459112	9	13943797	PRIJEM		19693.5	UROK	
2327	9	13943797	VYDAJ	VYBER	17578.9		
2367	9	13943797	VYDAJ	VYBER	19678.9	SLUZBY	
2329	9	13943797	VYDAJ	VYBER	16278.9		
2196	9	13943797	PRIJEM	VKLAD	17078.9		
2438	9	13943797	PRIJEM	VKLAD	18378.9		
3459113	9	13943797	PRIJEM		18452.2	UROK	
2368	9	13943797	VYDAJ	VYBER	18437.6	SLUZBY	
2197	9	13943797	PRIJEM	VKLAD	19237.6		
2439	9	13943797	PRIJEM	VKLAD	22837.6		
3459114	9	13943797	PRIJEM		22918.7	UROK	
2369	9	13943797	VYDAJ	VYBER	22904.1	SLUZBY	
2198	9	13943797	PRIJEM	VKLAD	23704.1		
2440	9	13943797	PRIJEM	VKLAD	26104.1		
3459115	9	13943797	PRIJEM		26202.9	UROK	
2370	9	13943797	VYDAJ	VYBER	26188.3	SLUZBY	

2199	9	13943797	PRIJEM	VKLAD	26988.3		
2441	9	13943797	VYDAJ	VYBER	26888.3		
3459116	9	13943797	PRIJEM		26998.2	UROK	
2371	9	13943797	VYDAJ	VYBER	26983.6	SLUZBY	
2200	9	13943797	PRIJEM	VKLAD	27783.6		
2285	9	13943797	VYDAJ	VYBER	25683.6		
2442	9	13943797	VYDAJ	VYBER	25383.6		
3459117	9	13943797	PRIJEM		25493	UROK	
2372	9	13943797	VYDAJ	VYBER	25478.4	SLUZBY	
2297	9	13943797	VYDAJ	VYBER	24478.4		
2268	9	13943797	VYDAJ	VYBER	23378.4		
2201	9	13943797	PRIJEM	VKLAD	24178.4		
2298	9	13943797	VYDAJ	VYBER	23278.4		
2443	9	13943797	PRIJEM	VKLAD	26678.4		
2302	9	13943797	VYDAJ	VYBER	25978.4		
2300	9	13943797	VYDAJ	VYBER	24528.4		
2301	9	13943797	VYDAJ	VYBER	23578.4		
2299	9	13943797	VYDAJ	VYBER	22728.4		
2280	9	13943797	VYDAJ	VYBER	18628.4		
3459118	9	13943797	PRIJEM		18728.4	UROK	
2373	9	13943797	VYDAJ	VYBER	18713.8	SLUZBY	
2202	9	13943797	PRIJEM	VKLAD	19513.8		
2444	9	13943797	PRIJEM	VKLAD	24113.8		
3459119	9	13943797	PRIJEM		24198.1	UROK	
2374	9	13943797	VYDAJ	VYBER	24183.5	SLUZBY	
2203	9	13943797	PRIJEM	VKLAD	24983.5		
2332	9	13943797	VYDAJ	VYBER	23183.5		
2445	9	13943797	PRIJEM	VKLAD	23483.5		
3459120	9	13943797	PRIJEM		23582.4	UROK	
2375	9	13943797	VYDAJ	VYBER	23567.8	SLUZBY	
2204	9	13943797	PRIJEM	VKLAD	24367.8		

2446	9	13943797	PRIJEM	VKLAD	25367.8		
3459121	9	13943797	PRIJEM		25468.1	UROK	
2376	9	13943797	VYDAJ	VYBER	25453.5	SLUZBY	
2179	9		PRIJEM	VKLAD	36584.5		
2205	9		PRIJEM	VKLAD	37384.5		
2447	9		VYDAJ	VYBER	30284.5		
3459122	9		PRIJEM		30405	UROK	
2377	9		VYDAJ	VYBER	30390.4	SLUZBY	
2180	9		PRIJEM	VKLAD	33858.4		
2262	9		VYDAJ	VYBER	13558.4		
3505	15	24265569	PRIJEM	VKLAD	800		
3515	15	24265569	PRIJEM	VKLAD	17865		
3530851	15	24265569	PRIJEM		17927.4	UROK	
3805	15	24265569	PRIJEM	VKLAD	22327.4		
3516	15	24265569	PRIJEM	VKLAD	39392.4		
3530852	15	24265569	PRIJEM		39541.6	UROK	
3806	15	24265569	VYDAJ	VYBER	32241.6		
3517	15	24265569	PRIJEM	VKLAD	57839.6		
3807	15	24265569	VYDAJ	VYBER	42066.1		
3530853	15	24265569	PRIJEM		58066.1	UROK	
3518	15		PRIJEM	VKLAD	59131.1		
3808	15		VYDAJ	VYBER	43531.1		
3530854	15		PRIJEM		43706.8	UROK	
3519	15		PRIJEM	VKLAD	60771.8		
3591	15	24265569	VYDAJ	PREVOD NA UCET	57699.8	SIPO	24265569
3747	15		VYDAJ	VYBER	57925.1	SLUZBY	
3530855	15		PRIJEM		57939.7	UROK	

3809	15		VYDAJ	VYBER	42625.1		
3592	15	24265569	VYDAJ	PREVOD NA UCET	56618.1	SIPO	2426556 9
3520	15		PRIJE M	VKLAD	59690.1		
3710	15		VYDAJ	VYBER	51698.1		
3708	15		VYDAJ	VYBER	49298.1		
3709	15		VYDAJ	VYBER	47978.1		
3748	15		VYDAJ	VYBER	48190.2	SLUZBY	
3810	15		VYDAJ	VYBER	38390.2		
353085 6	15		PRIJE M		48204.8	UROK	
3521	15		PRIJE M	VKLAD	55455.2		
3593	15	24265569	VYDAJ	PREVOD NA UCET	52383.2	SIPO	2426556 9
3811	15		VYDAJ	VYBER	40377.2		
3749	15		VYDAJ	VYBER	52577.2	SLUZBY	
353085 7	15		PRIJE M		52591.8	UROK	
3594	15	24265569	VYDAJ	PREVOD NA UCET	54370.2	SIPO	2426556 9
3522	15		PRIJE M	VKLAD	57442.2		
3812	15		VYDAJ	VYBER	40270.2		
3750	15		VYDAJ	VYBER	40487	SLUZBY	
353085 8	15		PRIJE M		40501.6	UROK	
3595	15	24265569	VYDAJ	PREVOD NA UCET	63013	SIPO	2426556 9
3523	15		PRIJE M	VKLAD	66085		
3651	15		VYDAJ	VYBER	16293		
4152	18	14567877	PRIJE M	VKLAD	1100		
4391	18		PRIJE M	VKLAD	14600		
4392	18		PRIJE M	VKLAD	17200		
4393	18	14567877	PRIJE M	VKLAD	20700		
4394	18	14567877	PRIJE M	VKLAD	23200		
4395	18	14567877	PRIJE M	VKLAD	23400		

4396	18	14567877	VYDAJ	VYBER	21500		
4155	18	14567877	PRIJEM	VKLAD	34902		
4397	18	14567877	VYDAJ	VYBER	28902		
3459193	18	14567877	PRIJEM		28944.7	UROK	
4398	18	14567877	VYDAJ	VYBER	27944.7		
3459194	18	14567877	PRIJEM		28065.1	UROK	
4173	18	14567877	PRIJEM	VKLAD	35771.1		
4399	18	14567877	VYDAJ	VYBER	29171.1		
4163	18	14567877	PRIJEM	VKLAD	74078.1		
4171	18	14567877	PRIJEM	VKLAD	109095.1		
3459195	18	14567877	PRIJEM		109277	UROK	
4400	18	14567877	VYDAJ	VYBER	68277		
3459196	18	14567877	PRIJEM		68731.6	UROK	
4154	18	14567877	PRIJEM	VKLAD	78979.6		
4161	18	14567877	PRIJEM	VKLAD	90838.6		
4401	18	14567877	VYDAJ	VYBER	57438.6		
3459197	18	14567877	PRIJEM		57789.8	UROK	
4336	18	14567877	VYDAJ	VYBER	57759.8	SLUZBY	
4174	18		PRIJEM	VKLAD	97847.8		
4232	18		VYBER	VYBER	79482.8		
4300	18		VYDAJ	VYBER	72282.8		
4273	18		VYDAJ	VYBER	2082.8		
5180	22	97847811	PRIJEM	VKLAD	1100		
5184	22		PRIJEM	VKLAD	17973		
5402	22	97847811	PRIJEM	VKLAD	21473		
3530968	22	97847811	PRIJEM		21499.8	UROK	
5185	22	97847811	PRIJEM	VKLAD	38372.8		
5403	22	97847811	VYDAJ	VYBER	32672.8		

3530969	22		PRIJEM		32777.4	UROK	
5516	23	20774019	PRIJEM	VKLAD	800		
5522	23	20774019	PRIJEM	VKLAD	33932		
5785	23	20774019	VYDAJ	VYBER	28632		
3531025	23	20774019	PRIJEM		28671.8	UROK	
5523	23	20774019	PRIJEM	VKLAD	50759.8		
5786	23	20774019	VYDAJ	VYBER	39359.8		
3531026	23	20774019	PRIJEM		39494.9	UROK	
5524	23	20774019	PRIJEM	VKLAD	61582.9		
5787	23	20774019	VYDAJ	VYBER	42182.9		
3531027	23	20774019	PRIJEM		42351.5	UROK	
5525	23	20774019	PRIJEM	VKLAD	64439.5		
5788	23	20774019	VYDAJ	VYBER	47139.5		
3531028	23	20774019	PRIJEM		47323.5	UROK	
5526	23	20774019	PRIJEM	VKLAD	69411.5		
5789	23		VYDAJ	VYBER	46511.5		
5723	23		VYDAJ	VYBER	46692.2	SLUZBY	
3531029	23		PRIJEM		46706.8	UROK	
5599	23	20774019	VYDAJ	PREVOD NA UCET	40885.2	SIPO	20774019
5527	23		PRIJEM	VKLAD	62973.2		
5790	23		VYDAJ	VYBER	45773.2		
5684	23		VYDAJ	VYBER	41693.2		
5724	23		VYDAJ	VYBER	41885.6	SLUZBY	
3531030	23		PRIJEM		41900.2	UROK	
5600	23	20774019	VYDAJ	PREVOD NA UCET	36078.6	SIPO	20774019
5528	23		PRIJEM	VKLAD	69210.6		
5791	23		VYDAJ	VYBER	48910.6		
5673	23		VYDAJ	VYBER	45710.6		

5725	23		VYDAJ	VYBER	45911.2	SLUZBY	
3531031	23		PRIJEM		45925.8	UROK	
5601	23	20774019	VYDAJ	PREVOD NA UCET	40104.2	SIPO	20774019
5668	23		VYDAJ	VYBER	25544.2		
5529	23		PRIJEM	VKLAD	47632.2		
5792	23		VYDAJ	VYBER	35632.2		
5726	23		VYDAJ	VYBER	35784.6	SLUZBY	
3531032	23		PRIJEM		35799.2	UROK	
5602	23	20774019	VYDAJ	PREVOD NA UCET	29977.6	SIPO	20774019
5530	23		PRIJEM	VKLAD	52065.6		
5793	23		VYDAJ	VYBER	39365.6		
5727	23		VYDAJ	VYBER	39523.9	SLUZBY	
3531033	23		PRIJEM		39538.5	UROK	
5603	23	20774019	VYDAJ	PREVOD NA UCET	33716.9	SIPO	20774019
5531	23		PRIJEM	VKLAD	55804.9		
5794	23		VYDAJ	VYBER	39404.9		
5728	23		VYDAJ	VYBER	39574.4	SLUZBY	
3531034	23		PRIJEM		39589	UROK	
5604	23	20774019	VYDAJ	PREVOD NA UCET	33767.4	SIPO	20774019
5532	23		PRIJEM	VKLAD	55855.4		
5795	23		VYDAJ	VYBER	42455.4		
5729	23		VYDAJ	VYBER	42629.3	SLUZBY	
3531035	23		PRIJEM		42643.9	UROK	
5605	23	20774019	VYDAJ	PREVOD NA UCET	36822.3	SIPO	20774019
5533	23		PRIJEM	VKLAD	58910.3		
5796	23		VYDAJ	VYBER	40510.3		
5730	23		VYDAJ	VYBER	40686.4	SLUZBY	
3531036	23		PRIJEM		40701	UROK	

5606	23	20774019	VYDAJ	PREVOD NA UCET	34879.4	SIPO	20774019
5534	23		PRIJEM	VKLAD	68011.4		
6517	26		PRIJEM	VKLAD	800		
6525	26	73418420	PRIJEM	PREVOD Z UCTU	25602		73418420
3531145	26		PRIJEM		25670.8	UROK	
6826	26		PRIJEM	VKLAD	26970.8		
6526	26	73418420	PRIJEM	PREVOD Z UCTU	51772.8		73418420
3531146	26		PRIJEM		51948.6	UROK	
6827	26		VYDAJ	VYBER	36648.6		
6527	26	73418420	PRIJEM	PREVOD Z UCTU	61450.6		73418420
3531147	26		PRIJEM		61692.3	UROK	
6828	26		VYDAJ	VYBER	42692.3		
6528	26	73418420	PRIJEM	PREVOD Z UCTU	67494.3		73418420
3531148	26		PRIJEM		67766.3	UROK	
6829	26		VYDAJ	VYBER	46266.3		
6529	26	73418420	PRIJEM	PREVOD Z UCTU	83469.3		73418420
6716	26		VYDAJ	VYBER	81669.3		
6766	26		VYDAJ	VYBER	81978.4	SLUZBY	
3531149	26		PRIJEM		81993	UROK	
6830	26		VYDAJ	VYBER	52878.4		
6602	26		VYDAJ	PREVOD NA UCET	42491.4	SIPO	12891853
6728	26		VYDAJ	VYBER	57993.4		
6530	26	73418420	PRIJEM	PREVOD Z UCTU	67293.4		73418420
6676	26		VYBER	VYBER	35114.4		
6767	26		VYDAJ	VYBER	35331.1	SLUZBY	
3531150	26		PRIJEM		35345.7	UROK	
6831	26		VYDAJ	VYBER	30631.1		
6603	26	12891853	VYDAJ	PREVOD NA UCET	20244.1	SIPO	12891853

6531	26	73418420	PRIJEM	PREVOD Z UCTU	45046.1		73418420
6673	26		VYBER	VYBER	37548.1		
6768	26		VYDAJ	VYBER	37689.5	SLUZBY	
3531151	26		PRIJEM		37704.1	UROK	
6832	26		VYDAJ	VYBER	25602.5		
6604	26	12891853	VYDAJ	PREVOD NA UCET	27302.5	SIPO	12891853
6532	26	73418420	PRIJEM	PREVOD Z UCTU	50404.5		73418420
6727	26		VYDAJ	VYBER	40804.5		
6670	26		VYBER	VYBER	29169.5		
6769	26		VYDAJ	VYBER	29301.3	SLUZBY	
3531152	26		PRIJEM		29315.9	UROK	
6833	26		VYDAJ	VYBER	27501.3		
6605	26	12891853	VYDAJ	PREVOD NA UCET	17114.3	SIPO	12891853
6533	26	73418420	PRIJEM	PREVOD Z UCTU	41916.3		73418420
6770	26		VYDAJ	VYBER	42046.9	SLUZBY	
3531153	26		PRIJEM		42061.5	UROK	
6834	26		VYDAJ	VYBER	33546.9		
6606	26	12891853	VYDAJ	PREVOD NA UCET	23159.9	SIPO	12891853
6534	26	73418420	PRIJEM	PREVOD Z UCTU	47961.9		73418420
6771	26		VYDAJ	VYBER	48123	SLUZBY	
3531154	26		PRIJEM		48137.6	UROK	
6835	26		VYDAJ	VYBER	34423		
6607	26	12891853	VYDAJ	PREVOD NA UCET	24036	SIPO	12891853
6955	27	28175699	PRIJEM	VKLAD	900		
6965	27	28175699	PRIJEM	VKLAD	9527		
7255	27	28175699	PRIJEM	VKLAD	17827		
3531210	27	28175699	PRIJEM		17841.4	UROK	
6966	27	28175699	PRIJEM	VKLAD	26468.4		

7256	27	28175699	VYDAJ	VYBER	24768.4		
3531211	27	28175699	PRIJEM		24848.4	UROK	
6967	27	28175699	PRIJEM	VKLAD	37788.4		
7257	27	28175699	VYDAJ	VYBER	29388.4		
3531212	27	28175699	PRIJEM		29496	UROK	
6968	27	28175699	PRIJEM	VKLAD	38123		
7258	27	28175699	VYDAJ	VYBER	29523		
3531213	27		PRIJEM		29645.6	UROK	
6969	27		PRIJEM	VKLAD	38272.6		
7259	27		VYDAJ	VYBER	31172.6		
7197	27		VYDAJ	VYBER	31282.5	SLUZBY	
3531214	27		PRIJEM		31297.1	UROK	
7042	27	28175699	VYDAJ	PREVOD NA UCET	27228.5	SIPO	28175699
6970	27		PRIJEM	VKLAD	35855.5		
7158	27		VYDAJ	VYBER	31555.5		
7260	27		VYDAJ	VYBER	25855.5		
7198	27		VYDAJ	VYBER	25967.3	SLUZBY	
3531215	27		PRIJEM		25981.9	UROK	
7043	27	28175699	VYDAJ	PREVOD NA UCET	21913.3	SIPO	28175699
6971	27		PRIJEM	VKLAD	30540.3		
7261	27		VYDAJ	VYBER	25840.3		
7199	27		VYDAJ	VYBER	25943.9	SLUZBY	
3531216	27		PRIJEM		25958.5	UROK	
7044	27	28175699	VYDAJ	PREVOD NA UCET	21889.9	SIPO	28175699
6972	27		PRIJEM	VKLAD	30516.9		
7262	27		VYDAJ	VYBER	26716.9		
7200	27		VYDAJ	VYBER	26821.1	SLUZBY	
3531217	27		PRIJEM		26835.7	UROK	

7045	27	28175699	VYDAJ	PREVOD NA UCET	22767.1	SIPO	28175699
6973	27		PRIJEM	VKLAD	35707.1		
8563	31	12345679	PRIJEM	VKLAD	800		
8585	31		PRIJEM	VKLAD	6139		
8588	31		PRIJEM	VKLAD	21182		
8567	31		PRIJEM	VKLAD	63442		
8886	31	12345679	VYDAJ	VYBER	41942		
3531366	31		PRIJEM		42081.1	UROK	
8579	31		PRIJEM	VKLAD	91618.1		
9220	33	33047658	PRIJEM	VKLAD	600		
9229	33	33047658	PRIJEM	VKLAD	24189		
9475	33	33047658	PRIJEM	VKLAD	26189		
3531421	33	33047658	PRIJEM		26210.8	UROK	
9230	33	33047658	PRIJEM	VKLAD	49799.8		
9476	33	33047658	VYDAJ	VYBER	37399.8		
3531422	33	33047658	PRIJEM		37519.5	UROK	
9231	33	33047658	PRIJEM	VKLAD	61108.5		
9477	33		VYDAJ	VYBER	44508.5		
3531423	33		PRIJEM		44671.2	UROK	
9232	33		PRIJEM	VKLAD	80055.2		
9478	33		VYDAJ	VYBER	53055.2		
3531424	33		PRIJEM		53249.9	UROK	
9233	33		PRIJEM	VKLAD	76838.9		
9479	33		VYDAJ	VYBER	50038.9		
9416	33		VYDAJ	VYBER	50241.7	SLUZBY	
3531425	33		PRIJEM		50256.3	UROK	

9306	33	33047658	VYDAJ	PREVOD NA UCET	46349.7	SIPO	33047658
9234	33		PRIJEM	VKLAD	69938.7		
9480	33		VYDAJ	VYBER	45138.7		
9417	33		VYDAJ	VYBER	45366.2	SLUZBY	
3531426	33		PRIJEM		45380.8	UROK	
9307	33	33047658	VYDAJ	PREVOD NA UCET	41474.2	SIPO	33047658
9235	33		PRIJEM	VKLAD	65063.2		
9378	33		VYDAJ	VYBER	62063.2		
9481	33		VYDAJ	VYBER	44363.2		
9384	33		VYDAJ	VYBER	40563.2		
9381	33		VYDAJ	VYBER	40063.2		
9418	33		VYDAJ	VYBER	40270.4	SLUZBY	
3531427	33		PRIJEM		40285	UROK	
9308	33	33047658	VYDAJ	PREVOD NA UCET	36378.4	SIPO	33047658
9236	33		PRIJEM	VKLAD	59967.4		
9379	33		VYDAJ	VYBER	58967.4		
9482	33		VYDAJ	VYBER	43067.4		
9419	33		VYDAJ	VYBER	43260.8	SLUZBY	
3531428	33		PRIJEM		43275.4	UROK	
9309	33	33047658	VYDAJ	PREVOD NA UCET	39368.8	SIPO	33047658
9237	33		PRIJEM	VKLAD	62957.8		
9483	33		VYDAJ	VYBER	45257.8		
9420	33		VYDAJ	VYBER	45462.9	SLUZBY	
3531429	33		PRIJEM		45477.5	UROK	
9380	33		VYDAJ	VYBER	41062.9		
9310	33	33047658	VYDAJ	PREVOD NA UCET	37170.9	SIPO	33047658
9238	33		PRIJEM	VKLAD	72554.9		
9606	34	2569228	PRIJEM	VKLAD	600		
9616	34	2569228	PRIJEM	PREVOD Z UCTU	32445		2569228

10027	34		VYDAJ	VYBER	29545		
3531485	34		PRIJEM		29604.6	UROK	
9617	34	2569228	PRIJEM	PREVOD Z UCTU	61449.6		2569228
10028	34		VYDAJ	VYBER	41049.6		
3531486	34		PRIJEM		41196.6	UROK	
9618	34	2569228	PRIJEM	PREVOD Z UCTU	88964.6		2569228
10029	34		VYDAJ	VYBER	55964.6		
3531487	34		PRIJEM		56167.7	UROK	
9619	34	2569228	PRIJEM	PREVOD Z UCTU	88012.7		2569228
10030	34		VYDAJ	VYBER	55112.7		
3531488	34		PRIJEM		55344	UROK	
9620	34	2569228	PRIJEM	PREVOD Z UCTU	87189		2569228
9836	34	58999049	VYDAJ	PREVOD NA UCET	81119		58999049
10031	34		VYDAJ	VYBER	53219		
9969	34		VYDAJ	VYBER	53428.7	SLUZBY	
3531489	34		PRIJEM		53458.7	UROK	
9765	34	19993174	VYDAJ	PREVOD NA UCET	52926.7	SIPO	19993174
9902	34		VYBER	VYBER	39667.7		
9621	34	2569228	PRIJEM	PREVOD Z UCTU	71512.7		2569228
9837	34	58999049	VYDAJ	PREVOD NA UCET	65442.7		58999049
10032	34		VYDAJ	VYBER	47442.7		
9970	34		VYDAJ	VYBER	47632.8	SLUZBY	
3531490	34		PRIJEM		47662.8	UROK	
9766	34	19993174	VYDAJ	PREVOD NA UCET	47130.8	SIPO	19993174
9838	34	58999049	VYDAJ	PREVOD NA UCET	72905.8		58999049
9622	34	2569228	PRIJEM	PREVOD Z UCTU	78975.8		2569228
10033	34		VYDAJ	VYBER	51105.8		
9933	34		VYDAJ	VYBER	45325.1		

9971	34		VYDAJ	VYBER	45295.1	SLUZBY	
3531491	34		PRIJEM		51325.1	UROK	
9767	34	19993174	VYDAJ	PREVOD NA UCET	44793.1	SIPO	19993174
9899	34		VYBER	VYBER	36429.1		
9839	34	58999049	VYDAJ	PREVOD NA UCET	62204.1		58999049
9623	34	2569228	PRIJEM	PREVOD Z UCTU	68274.1		2569228
9900	34		VYBER	VYBER	38360.1		
10034	34		VYDAJ	VYBER	33260.1		
9901	34		VYBER	VYBER	20848.1		
9972	34		VYDAJ	VYBER	20955.8	SLUZBY	
3531492	34		PRIJEM		20985.8	UROK	
9768	34	19993174	VYDAJ	PREVOD NA UCET	20453.8	SIPO	19993174
9624	34	2569228	PRIJEM	PREVOD Z UCTU	68221.8		2569228
9840	34	58999049	VYDAJ	PREVOD NA UCET	62151.8		58999049
10154	35	60448020	PRIJEM	VKLAD	600		
10162	35		PRIJEM	PREVOD Z UCTU	6046	DUCHOD	60448020
10163	35	60448020	PRIJEM	PREVOD Z UCTU	11492	DUCHOD	60448020
10164	35	60448020	PRIJEM	PREVOD Z UCTU	16938	DUCHOD	60448020
10165	35	60448020	PRIJEM	PREVOD Z UCTU	22384	DUCHOD	60448020
10166	35	60448020	PRIJEM	PREVOD Z UCTU	27830	DUCHOD	60448020
10310	35	99848401	VYDAJ	PREVOD NA UCET	26468		99848401
10238	35	56788617	VYDAJ	PREVOD NA UCET	25446	SIPO	56788617
144541	485	42968433	930104	PRIJEM	300	300	
207264	485	42968433	930101	PRIJEM	1000	1000	
452728	485	42968433	930103	PRIJEM	600	600	
497211	485	4968433	930103	PRIJEM	200	200	

505240	485	42968433	930103	PRIJEM	1000	1000	
542216	1844	42988401	930107	PRIJEM	3242	3742	ST
542216	1844	42988401	930107	PRIJEM	3242	3742	ST
542216	1844	42988401	930107	PRIJEM	3242	500	ST
542216	1844	8	930107	PRIJEM	300	200	ST
542216	1844	42988401	930107	PRIJEM	100	400	ST
542216	1844	4298841	930107	PRIJEM	250	150	ST
637741	2177	62457513	930104	PRIJEM	800	800	YZ
637742	2177	62457513	930105	PRIJEM	5123	5923	YZ
689827	2357	93012204	930104	PRIJEM	800	800	
695247	2357	93012204	930101	PRIJEM	700	700	
771035	2357	93012204	930102	PRIJEM	1100	1100	
1117247	3818	86543521	930101	PRIJEM	600	600	
9767	34	19993174	VYDAJ	PREVOD NA UCET	44793.1	SIPO	19993174
9899	34		VYBER	VYBER	36429.1		
9839	34	58999049	VYDAJ	PREVOD NA UCET	62204.1		58999049
9623	34	2569228	PRIJEM	PREVOD Z UCTU	68274.1		2569228
9900	34		VYBER	VYBER	38360.1		
10034	34		VYDAJ	VYBER	33260.1		
9901	34		VYBER	VYBER	20848.1		
9972	34		VYDAJ	VYBER	20955.8	SLUZBY	
3531492	34		PRIJEM		20985.8	UROK	
9768	34	19993174	VYDAJ	PREVOD NA UCET	20453.8	SIPO	19993174
9624	34	2569228	PRIJEM	PREVOD Z UCTU	68221.8		2569228
9840	34	58999049	VYDAJ	PREVOD NA UCET	62151.8		58999049

10154	35	60448020	PRIJEM	VKLAD	600		
10162	35		PRIJEM	PREVOD Z UCTU	6046	DUCHOD	60448020
10163	35	60448020	PRIJEM	PREVOD Z UCTU	11492	DUCHOD	60448020
10164	35	60448020	PRIJEM	PREVOD Z UCTU	16938	DUCHOD	60448020
10165	35	60448020	PRIJEM	PREVOD Z UCTU	22384	DUCHOD	60448020
10166	35	60448020	PRIJEM	PREVOD Z UCTU	27830	DUCHOD	60448020
10310	35	99848401	VYDAJ	PREVOD NA UCET	26468		99848401
10238	35	56788617	VYDAJ	PREVOD NA UCET	25446	SIPO	56788617
144541	485	42968433	930104	PRIJEM	300	300	
207264	485	42968433	930101	PRIJEM	1000	1000	
452728	485	42968433	930103	PRIJEM	600	600	
497211	485	4968433	930103	PRIJEM	200	200	
505240	485	42968433	930103	PRIJEM	1000	1000	
542216	1844	42988401	930107	PRIJEM	3242	3742	ST
542216	1844	42988401	930107	PRIJEM	3242	3742	ST
542216	1844	42988401	930107	PRIJEM	3242	500	ST
542216	1844	8	930107	PRIJEM	300	200	ST
542216	1844	42988401	930107	PRIJEM	100	400	ST
542216	1844	4298841	930107	PRIJEM	250	150	ST

VITA

GEORGE OGUNGBEMILE

EDUCATION

MS Computing and Information Science **2017 - 2019**
SAM HOUSTON STATE UNIVERSITY
BSc Computer Science **2006 - 2010**
UNIVERSITY OF LAGOS

EMPLOYMENT HISTORY

Graduate Assistant **Aug 2018 - Present**
COMPUTER SCIENCE DEPARTMENT – SHSU
Assist professors in grading, teaching and organization of lectures to students of the department.

Internship **May 2018 – Aug 2018**
VISA INC
ATM Development & Devops. Wrote programs in proprietary programming languages for terminal and communication handlers of Automated Teller Machines. Worked on CICD tools to configure automation pipelines for continuous software and update deployment.

Database Analyst - Volunteer **Nov 2017 – May 2018**
TEXAS RESEARCH INSTITUTE FOR ENVIRONMENTAL STUDIES – SHSU
Modification of facility database for compliance with the research industry standard.

Graduate Executive (Operations and Technology) **Jul 2013 – Aug 2017**
DIAMOND BANK PLC
Cash & tellering supervisor, ATM custodianship, currency vault management, trade desk officer (imports, exports and invisibles), customer service supervision

IT Helpdesk Administrator **Nov 2011 – Jul 2013**
CHEMICAL AND ALLIED PRODUCTS PLC
Regularly conducted IT infrastructure project management, daily routines involved server administration (Virtualization, RAID deployment), active directory administration, disaster recovery planning and implementation, Hardware configuration and support.

Internship **May – October 2008**
DEPARTMENT OF PETROLEUM RESOURCES
Collaborated on a pipeline and products pricing project which included a software development and inventory automation phase. This availed me an opportunity for industrial application of structured query language and Microsoft access functions.

ADDITIONAL WORK EXPERIENCE

Voluntary Youth Service **Jun 2010 - Aug 2011**

NATIONAL YOUTH SERVICE CORPS

Participated in an optional service scheme, teaching high school students in rural areas of Africa about computers and internet

PROFESSIONAL CERTIFICATIONS

Cisco Certified Network Associate - CCNA

Microsoft Certified IT Professional