J. Basic. Appl. Sci. Res., 6(6)21-31, 2016

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Data Matching: An Algorithm for Detecting and Resolving Anomalies in Data Federation

Ejaz Ahmed[1], Nik Bessis[2], Waseem Shahzad[3]

[1, 3]Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan
[2]Department of Computing, Edge Hill University, Ormskirk, United Kingdom

Received: March 14, 2016
Accepted: May 9, 2016

ABSTRACT

Accessing federated data sources and databases leads to several concerns related to anomalies. Specifically, data from integrated and federated data sources can have anomalies such as inconsistency, redundancies and interpretation conflicts in terms of different business contexts. Normally, data is accessed from federated data sources using multi-vendors' database management systems (DBMS) own repository, catalogue or metadata schema. However, catalogues do not provide complete description of data elements during the processes of search and match. In this paper, we propose a generic algorithm to help in integrating catalogues of distinct multi-vendors DBMSs. Our focus is to provide seamless data search of matching of schema's structure and objects. Current database environments are highly heterogeneous and our algorithm used to play an essential role as implementation for federated data found in grid and inter-cloud e-infrastructure. Within this context, our algorithm performs search by matching information, extract anomalies in federated data sources or DBMSs, and our experiments demonstrate and detect the evaluations of anomalies on data matching.

KEYWORDS: Data Integration, Federated Environment, Staging DBMS, Web Services, Metadata, Pattern Matching, Grid, Inter-Cloud.

1 INTRODUCTION

Recent developments in science and technology have enabled the search and matching of information from distinct heterogeneous data sources. This has created an immense opportunity for data matching and detecting data anomalies. Traditionally, integrated data is accessed and searched with the help of complex ad-hoc programming (need basis), software tools and utilities (normally used for data warehouses) and programs are tuned for better usage [6]. Current database applications domains such as data migration, data warehousing, e-business, semantic query processing and social data network have challenges in accessing data in terms of data matching [3, 9]. Mainly, we use data source or database as a major resource for storing or accessing data. Data is being accessible and shared using emerging web technologies such as Web Services (WS) when it is dispersed and distributed across network heterogeneous computers [1]. Currently, data federation enhancing traditional data integration approaches and requires more attention in emerging environments such as grids, and inter-cloud with virtualization of multi-vendor heterogeneous data sources [2, 8, 9, 11, 16]. Emerging infrastructure is supporting distributed computational resources and data access under cloud environment [17]. We categorized traditional and emerging concepts of accessing and sharing data by introducing e-infrastructures named as Traditional Distributed Environment (TDE) and Emerging Distributed Environment (EDE). In these e-infrastructures, data matching of elements can be managed in single as well as federated databases. Our research work uses same meanings and interpretations described in the survey paper [3]. Occurrences of such data elements with associated interpretation are known as anomalies [14]. Usually data anomalies can be identified in the form of inconsistent and redundant data elements. In the context of this paper, the term of 'inconsistent' refers to a data element having different meanings in different contexts whilst the term of 'redundant' refers to identical data elements found in more than one data elements or databases.

Distributed methods using DBMS usually adopt an internet model but fail to achieve a service oriented approach like a grid or cloud. Usage of big data in cloud environment has significant advantages using DBMS [20]. State-of-the-art data anomalies such as data inconsistencies and redundancies are emerged when data is accessed or shared from source to target schema due to inconsistent schema structures or data interpretations [4, 6]. Schema is a logical structure of the database where data resides for manipulation purpose. Mapping is a transformation that

***Corresponding Author:** Ejaz Ahmed, Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan. ejaz.ahmed@nu.edu.pk,

identifies, relates or maps relevant data instances of some structure elements when source and target data sources are integrated. A data matching process is managed on the basis of identification and relationship of structured elements with data instances of source and target data sources [3]. Matching is performed on data instance values for the corresponding mapping of structure elements such as similar attribute names of schema tables [18].

In general, data anomalies can be explored using metadata repositories of data sources or DBMSs. However, papers [1, 3, 4, 17] do not highlight a significant research in this direction. It is noted that metadata repositories do not capture meaning and interpretation of data elements except an actual string or text matching. We further support this by arguing that even if meaning and interpretations were available, these would have been contextualized within their local data level and thus these may erroneously interpret and be misleading at global data level. However, metadata definitions may have different meanings under different contexts and/or when stored on distinct and heterogeneous e-infrastructures. During data cleansing, same real-world objects or entities are described in distinct data sources [19]. Such similar entities are managed using GenLink algorithm with linkage rules and references of genetic programming [21].

We are particularly interested on how can we ensure that the extracted data from an object is consistent or not with other identical object(s) that is available from integrated data sources. We will use DBMS architecture of data federation for data matching by introducing staging database as a mediator for processing. In an EDE, identical information or data instance values may exist in federated databases. Such identical occurrences may be identical, redundant or inconsistent and this leads to data anomalies. Thus, there is a need to filter out, interpret and match data objects with their correct meaning and context, so we can take informed decisions during the data objects matching process [3].

Within this in mind, main purpose of this research work is to develop an algorithm to detect and explore data anomalies, and thus support informed decisions on data objects matching over heterogeneous e-infrastructures TDE and EDE. To achieve this, requirements and challenges of data objects matching are discussed in section 2. Section 3 presents functionality related issues. Section 4 presents how our framework is designed to perform data object matching. In section 5, we provide mathematical formulation of algorithm, and section 6 describes implementation of an algorithm with the help of experiment. Finally, a conclusion is provided in section 7.

2 Requirements and Challenges of Data Matching

A frequent concern with EDE heterogeneous e-infrastructures is that the same information is interpreted in more than one data sources when data is shared on multi vendor heterogeneous DBMSs. The concern appears in the form of data anomalies including data redundancies and inconsistencies. It is analyzed how EDE technology can be used to solve a particular search problem - finding duplicate or redundant items in distributed databases [10]. In such a scenario, there is a need to formulate a uniform strategy that will operate effectively to share and access data or search string patterns seamlessly. Following are the key requirements for searching and matching of information:

- Software application programmer works with few requirements. This will enhance application with uniform strategy of integrating DBMSs.
- In EDE, software application for matching can be easily enhanced and modified for execution efficiently.
- Identify pre-requisites on the resource provider like schemas of federated DBMS, so that new resources for integrations in TDE can be easily incorporated into EDE.
- Analyze schema structure and elements level data integration mapping and matching

There are two major issues to manage data matching; one matching of information using metadata in one vendor's DBMS is different from another vendor's DBMS [5]. For example, data matching is performed in Oracle DBMS by using metadata USER_TAB_COLS. This metadata repository searches structure of tables such as attributes' data types and their sizes. IBM DB2 is another prominent vendor of DBMS, in which similar structure of tables can be managed using its own metadata that is SYSIBM.SYSTABLES. Both USER_TAB_COLS and SYSTABLES are recognized in DBMSs Oracle and IBM DB2 respectively. This means metadata used in distinct vendors' databases are not generic. Both Oracle and IBM DB2 cannot communicate with each other using their metadata. These metadata are helpful for searching purpose but identical meaning and interpretation are still a challenge for a particular word to be searched. This means such repositories of vendors are developed independently with slightly different information of structure and terminology [3]. This can be obviously occurred when database application systems such as customer retail system and telecommunication billing system are from different domains. For example, in retail system customer is not necessarily registered or known whereas in billing system customer is treating as a consumer or client and registered entity because these application systems were designed

and developed by different experts with user requirements. Meaning and interpretation of customer is still a challengeable in data matching of heterogeneity, structure and terminology.

Our contribution resides to that - to the best of our knowledge - there is no formal method to enable the full satisfaction of all aforementioned requirements and challenges. Although a detailed survey about data matching in TDE has been covered in [3]. In TDE, it is evident that distributed file systems are providing similar easy access to distributed data, but require deployment of substantial technology with inter-organizational cooperation. Similarly, in EDE, a framework such as Open Grid Service Architecture - Data Access Integration (OGSA-DAI) – that is the de-facto specification framework for integrating data over grids - provides a transparent access to remote resources using web technology, but it also requires additional issues of mapping of structure to analyze granularity level data matching for the detection of data anomalies [1, 5]. In addition there is no work on piloting the effectiveness of OGSA-DAI specification framework for integrating data over inter-cloud environment. The IoT-oriented data storage framework in the cloud platform is expected to provide IoT data storage access, and the integrated management service [23]. Within this context, our work offers a novel approach to detect such data anomalies in TDE and EDE. The following strategies were addressed to accomplish results pf experiments, such that this research contributed by defining mechanisms that can:

- Be implemented with least specialized services at a participating data sources or database sites.
- Facilitated application schema programmers to provide ordinary services of regular schema;
- Allowed programmers and software developers to manage data search with the selection of data sources, staging database and filtering of matched data.
- Focused on the data management issues, namely, providing the means to manage and search from high volume engine data archives or data sources.

There are various problems and causes in generating data anomalies. Some problems exemplifying data anomalies in context of data mapping are [14]:

1. Width size of some attribute of a table in an integrated searched database called target schema is less than the size of a corresponding matching attribute in source schema.
2. An attribute of a table in a target schema has a Not Null constraint that refers to corresponding matching attribute in source schema that has a Null or blank value
3. Same data value exists in more than one database. For example, it is likely be possible that a bank credit card number appears in more than one banks' databases. In one database relation or table, it appears to identify personal information of a card holder. The same credit card number appears in some other bank's database table to identify the card holder is a defaulter.
4. An attribute of table in a target schema has referential integrity constraint that refers to a corresponding attribute of same or different parent table. The value is not populating in a target attribute if such value does not match a referring attribute of a parent table
5. An attribute of a table in a target schema has unique constraint. The value from a source attribute is already exists in an attribute of a target table. This causes unique constraint violation during the loading of the same data instance.

Experiments with such problems shows emerging of significant anomalies with the range of applications demonstrate practical utility and analysis on multi occurrences of data. The following sections describe the nature of mechanism and implementation issues.

3 Classification of Linguistic Naming Matching

In TDE, search is based on linguistic matching by using string, text and names. These string, text and names can be words or short sentences, finding semantically matched schema objects or elements. Names can be similar and their similarity can be defined semantically [3, 7]. Some of the semantics are defined as follows:

- Prefix/ suffix symbols or words are normally used in an equality of canonical name representations. For example, Customer# → Customer Number or Customer No → Customer Number, Item ID → Item Barcode or Item Identifier.
- Equality symbol (≅) is used for similar synonyms. For instance, Employee ≅ Person, Automobile ≅ Vehicle or Car and made-of ≅ model or brand, make etc.

- Common substrings produce similarity of names based on soundex. This means how names are recognized with their sounds rather than their spellings and edit distance? For example, Ship Destination ≅ Ship2, Invoicing ≅ Ordering, Invoice Type ≅ Order Type, Represented By ≅ Salesman or Representative.
- Hyponyms are used as equality. For example, article is-a publication and book is-a publication can be interpreted as article ≅ publication, book ≅ publication, and book ≅ article.
- Name matching is also significant. For example, works For ≅ manager, report To ≅ manager, report To ≅ supervisor, error ≅ bug, software Fault ≅ bug.

Linguistic conventional names are associated with each schema element. For example, attributes in source schema S_1 contain Cust# which means customer number and attributes Cust Address in S_2 means customer address.

Apart from classifications of matching, an issue of state-of-the-art anomalies will occur if information is inconsistent. This means

- Some attributes have different interpretation based on domain values but same meaning. For example, attribute gender contains domain values 'Male and Female' in one schema whereas 'M' and 'F' or '1' and '0' are domain values in some other schemas.
- Sometimes, size of data values of same matched attributes can be different and it may be significant to create anomalies. For example, if size of attribute First Name of person is larger than appropriate size then in some schema domain may contain full name of some persons.

4 Matching Process of String Patterns

Section 3 describes significant and prominent challenges of matching and mapping. This section describes process or framework of accessing data using matching as shown in figure 1. We propose a generic algorithm in section 5 that can help in reducing the mismatching of heterogeneous representations in terms of pattern search.

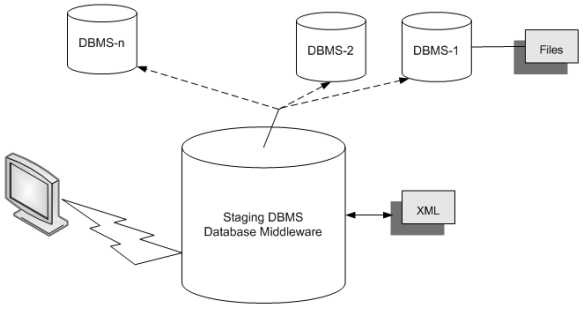

Figure 1: Federation of integrated heterogeneous databases

We build up a scenario by integrating heterogeneous DBMSs with staging DBMS and web interfaces are deployed to access data from these databases using WSs as shown in figure 1. Data is accessed from target schemas of heterogeneous mapped databases. We assume that the aforementioned heterogeneous architecture setting reassembles with federated databases, usually found in EDE e-infrastructures. DBMS software with tools and database levels – there exist the problem of communication between the software applications. Multi-vendor DBMSs cannot communicate directly with each other, our scenario of integrating heterogeneous DBMSs with staging DBMS provides seamless facility of data communication with each other via staging DBMS. All heterogeneous databases are connected with a staging DBMS using WS with the help of connectivity drivers such as ODBC and ODBC-JDBC [1]. To validate data federation concept staging DBMS provides a service oriented platform to make seamless virtualized data accessing and sharing. This virtualization helps in data loading, data transformations, data migration, data matching and temporary data storage services. In cloud computing, it is increasingly popular that data owners can outsource their data (e.g., staging DBMS) to public cloud server [22]. Following examples elaborate how mapping is performed in staging DBMS.

Example 1: There is n-number of heterogeneous databases integrated with staging DBMS as shown in figure 2. Information can be retrieved or accessed from these databases through staging DBMS. Information can be retrieved from these databases using SQL with the help of web interface. Temporary buffer storage of the staging schema or DBMS will be used while fetching or displaying data from federated databases.

Example 2: Similar data values or information can be searched or matched from integrated target schemas of databases in a grid or inter-cloud environment as shown in figure 2. Such occurrences of information may lead to the misconception of anomalies found including data redundancies or inconsistencies. Data search is performed using queries with WSs.

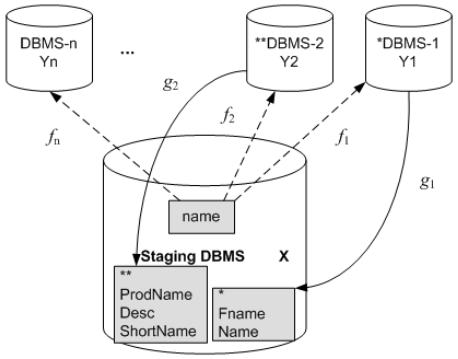

Figure 2: Searching patterns of string element *name* from federated DBMSs

Now our research work is moving towards formulation of mathematical algorithm for aforementioned scenario of framework of federated databases. Consider a search function *f* that is directed or mapped from a staging DBMS to any target federated database as shown in figure 2. If x is a string initiated at staging database (X) and searched from each federated database (Y) then *f* can be mapped as

f_i: X → Y^i , i = 1, 2, 3, …, n

The range Y may have more than one image of x ∈ X, i.e., more than one domain match exists in range Y. The possible matched values will be mapped in a staging database X defined by function *g* as

g_i: Y^i → X , i = 1, 2, 3, …, n

$$x \cong g_i(f_i(x)) \qquad \text{Eq (1)}$$

Equation (1) shows X is mapped from target database into staging database. Function g_i is used as a response of requesting function *f*. Sometimes, it may be possible g_i is a requesting function of a federated database in which some search is needed from other federated databases through staging DBMS. It is noted that x ≅ φ in case if no match found or function g is null.

$f(str) = \rho_{DB}(str, \psi, \delta)$ is a function defined in [12] and it is oversight because ψ is the union of instance values and structure or schema elements. Where δ represents an instance or domain value that needs to be searched from ψ = d (domain) and ψ = s (structure). Search is performed to explore structure elements and instance values or domain values. It is further elaborated in the form of an algorithm in the following section.

5 Proposed Algorithm for Detecting State-of-the-Art Data Anomalies

In above section, we formulate analytically how search values such as instance values are retrieved from integrated target schema of DBMSs. In this section, we extend our matching strategy for formulating proposed algorithm. We follow most of the basic nomenclature and notations as described in section 3 in which different elements denote different objects or meanings in terms of understanding and interpretation.

A database schema often contains structure of objects or elements known as data types, names, range values, constraints, optionally relationship type, cardinalities and uniqueness etc. Structures can be managed by a matcher to determine the similarity when two input schemas contain information and needs their comparisons of structures [7]. This means similarity of elements can be mapped between two sources in terms of data types and domain values [3]. This may include key characteristics such as unique, referential, entity integrity and cardinalities such as one-to-one, one-to-many and many-to-many. However, a certain matching can be based on the structural data elements such as object names. In figure 3, an attribute Name of source schema is structurally mapped to target schema attributes Cust Name, P Name and Name of an employee. Although a match of domain's data instance values exists [18], for example Name may appear in the description of a product.

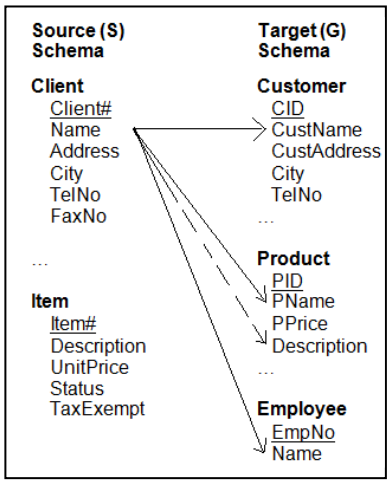

Figure 3: Matching of elements from source to target database schemas

To reduce the complexity of the heterogeneity of databases, their schemas are mapped using matching in a uniform manner of structure and domain values. Such kind of mapping cannot perform by using utilities of DBMSs such as import/export or XML-data-extractions [3].

Our research work is based on a global mapping method in which **G** represents global or target schema and **S** represents a source schema. A data transformation system $\boldsymbol{\Gamma}$ is a triplet $\boldsymbol{\Gamma}$ = (G, S, $M_{G,S}$), where $\mathbf{M}_{G,S}$ is a mapping between source (**S**) and target schema (**G**) [4]. A staging database DBMS or schema is used with this mapping in the federated environment and assuming staging schema acts just like a source. In example 3, staging schema is a source and one of the integrated DBMSs is a target schema.

Example-3: We are choosing an expression triplet $\boldsymbol{\Gamma}^1 = (G^1, S^1, \mathbf{M}^1_{G,S})$ by assigning mapping between source S and target schema G. This shows how elements are semantically related for matching of S and G as shown in figure 3. An attribute Name in Client table is mapped to three similar attributes of schema **G**, Cust Name in table Customer, P Name in table Product and Name in table Employee as shown in figure 3.

In this example, both **S** and **G** are from a same domain application but in a different business scenario, local requirement of the system or their interpretations can be different. A mapping between **S** and **G** may be constituted with Client. Client# and Customer. CID, by mapping expression or value correspondences. That is,

Client. Client# = Customer. CID

Similar mapping of other elements can be made from figure 3.

This is a simple case of mapping between two schemas whereby correspondences are created with attributes from schemas matching objects or relations. We will explore further the anomalies and redundancies that are highly likely to be occurred due to such mapping.

For further exploring of our algorithm, assume there are m data sources or databases; each database contains number of schemas S. There are many schema objects or contents say T such as table, view, procedures etc. This means

$$T_{j_i} \in S^j \qquad \text{where } i, j = 1, 2, \ldots, n \qquad \text{Eq (2)}$$

If C represents component or column of T then using equation (2) searching of C from database can be written as

$$C_k = C_{j_{i,k}} \in T_{j_i} \qquad \text{where } k = 1, 2, 3, \ldots, m \qquad \text{Eq (3)}$$

Based on equations (2), (3), matching expression in the form of search function g can be written as

$$(S^j, T_{j_i}, C) \rightarrow g_m(f(S^j, T_{j_i}, C_k))$$

$$\equiv \begin{cases} \{f(S^j, T_{j_i}, C_{j_{i,k}})\}_m \cup \{f(S^j, T_{j_i}, t_i[C_{j_{i,k}}]\}_m \\ \Phi_m \end{cases} \qquad \text{Eq (4)}$$

Where $g_m f$ is a composite function, returns image f of a search string C from data source or database m. It returns no value Φ when no image f is found in m. C_k matching patterns are verified using staging database repository of interpretation of schema objects such as column names. Two functions used in union, one for structured values and another for instance values in equation (1). Matching is performed by using equation (4), our well-defined derived algorithm.
We choose two columns from plug-in relations repository [12] that are $<L_1, L_2>$ of integrated data sources of databases m in some schema of staging database. In table 1, column L_1 represents matching words and L_2 represents their possible meanings and interpretations.

L1	L2
P Name	Name of product, description, generic name
Cust Name	Customer name, client name
Tel No	Telephone of customer, mobile number
Desc	Name, title, description
Name	Name of an employee
…	…

Table 1: Samples data names and their possible meanings or synonyms

Where i represents ith row number. If C is a "name" then C will be some of L_2 values corresponding to L_1, referring to figure 1. It means

C→ $<L_{11}, L_{21}>$, $<L_{12}, L_{22}>$, $<L_{14}, L_{24}>$, $<L_{15}, L_{25}>$

Incase, if no match is found from table $< L_1, L_2>$ then

C→ Φ from some database m.

In terms of an equation (4), we get an expression of table $<L_1, L_2>$.

$$(S^j, T_{j_i}, C) \equiv \{f(S^j, T_{j_i}, L_{11}), f(S^j, T_{j_i}, L_{12}), f(S^j, T_{j_i}, L_{14}), f(S^j, T_{j_i}, L_{15})\}_m \quad \text{Eq. (5)}$$

In equation (5), only instance value matching is explored for searching string name and each L represent domain or instance value. Using algorithm of equations (4), Equation (5) can be interpreted using flow diagram as shown in figure 4.

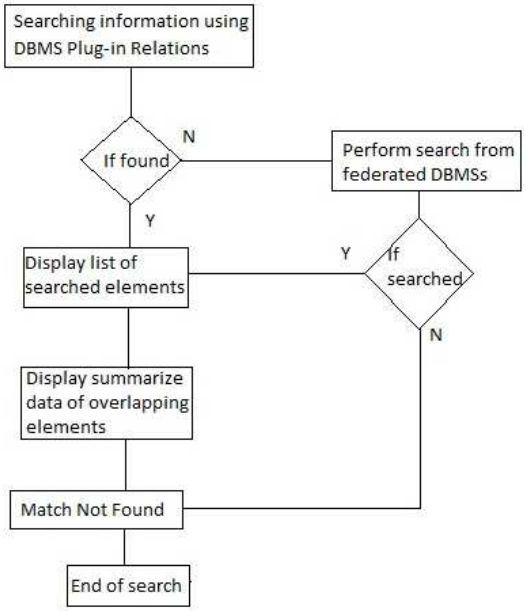

Figure 4: Flow of searching and matching text

Example-4: This example shows how instance values of attributes can be searched using equation (1). Assuming an instance value "5158134" of attribute telephone number (tno) can be searched or matched from number of federated databases schemas S1, S2, ..., Sn in an EDE.
Referring to equation (1), function *f* can be written as
$f(\text{str}) = \rho_{DB}(\text{"5158134"}, d, \{\text{tel\#, telno , tno, mobileno, mobile\#, faxno }\})$
Reference to equation (5), returned values of this function [12] would be
$\text{str} \cong g_1(f(\text{str}), \{\text{S2.Customer.telno, S2.EMP.mobile\#}\})$.
$\text{str} \cong g_1(f(\text{str}), \{\text{S1.Client.telno}\})$ that is t[S1.Client.telno] = "5158134"

Example 4 depicts a data inconsistency if some customer's telephone number may emerge as a mobile number of some employee in some other federated database. Such occurrences of distinct information are proved analytically with the help of aforementioned algorithm in equation (4).

6 Implementation and Experimental Results

In order to handle state-of-the-art data anomalies in federated databases, we implemented a novel approach by introducing public schema segments in federated databases whereby read privileges of schema objects are granted. We implement a replica schema of all federated DBMSs metadata catalogues in staging DBMS. We call such metadata catalogue a Staging CATalogue (SCAT), similar to table 1.

A front end WS interface called Pattern Matching Controller accesses all objects or elements using SCAT of pattern match data management architecture described in [14]. Herein, we extend our experimental functionality in an inter-cloud SCAT pattern where the architecture described in [14] remains the same. Readers are pointed to [16] for further information inter-cloud architectures.

SCAT contains structure of relations which is termed as plug-in relation [24] as given in equation (6). To improve the effectiveness of mappings, plug-in relations provide some auxiliary information in addition to information of metadata dictionary of any DBMS. The structure of plug-in relations can be changeable and created by the users. Defining the following plug-in relations and Table 1 is a schema state of an equation (6).

Data Table (Table Name, Schema Name, Description, Date Created)
Table Details (Table Name, Schema Name, Serial#, Attribute, Data Type, Size, Constraints, Short Desc, Detail Desc)

Eq (6)

The relations of equation (6) will act as an enterprise dictionary, catalogue and act as taxonomies of a schema. Underlined attributes represents entity integrity constraints. It is noted that attributes *Descriptions*, *Short Desc* and *DetailDesc* play an important role for the name or linguistic matching, and the description matching as described in section 3.

A granular search is performed under the layers of various segments. This means a search of single match is performed in federated databases through multiple layers of databases, schema objects, tables and their attributes. This indicates complexity in terms of granularity for match searching.

6.1 Data Matching and Mapping Implementation

A case study called Health Care Information System has been implemented and deployed to a collaborative diagnosis analysis. Data used in the experiments is partially synthetic and encrypted from real. Therefore, our research team chooses hospitals which are using real-world vendor databases with different schemas, adopted from the actual health care sites across Saudi Arabia [13, 14, 15]. Initially, these health care hospitals' sites are known as nodes of federated databases, named as H1 (University Hospital in Dhahran), H2 (City Hospital in Dhahran), H3 (Central Hospital in Dammam), H4 (Abdul Aziz Hospital in Riyadh) [14]. A staging database used locally to integrate data from databases of these hospitals. These all nodes are using multi-vendor federated DBMSs such as an Oracle, MySQL, IBM DB2, MS SQL Server etc. Table 2 shows some structured records of searching string "name" from city hospital Dhahran. Description column shows semantics anomalies of various matching of string "name". We have tested various scenarios from all hospitals using structure and instance value matching, the sample detection of anomalies are shown in figure 5.

Table 2: Matching string "name" exists in schema tables

Table Name	Database.Schema Name	Description	Date Created
PATIENT	Dhahran.H2	First_Name	
PATIENT	Dhahran.H2	Middle_Name	
PATIENT	Dhahran.H2	Family_Name	
PATIENT	Dhahran.H2	Name_A	
MEDICAL_LOC	Dhahran.H2	Loc_Name	
DEPENDENT	Dhahran.H2	First_Name	
DEPENDENT	Dhahran.H2	Middle_Name	
DEPENDENT	Dhahran.H2	Family_Name	
DEPENDENT	Dhahran.H2	Name_A	
CATEGORY	Dhahran.H2	Cat_Desc	
LAB_REQUISITION	Dhahran.H2	LabReq_Desc	
LAB_SUBREQUISITION	Dhahran.H2	LabReq_Desc	
RAD_REQUISITION	Dhahran.H2	RadReq_Desc	
TRMNT_REQUISITION	Dhahran.H2	TrmntReq_Desc	
HOSPITAL_CONTACT	Dhahran.H2	H_Name	
HOSPITAL_CONTACT	Dhahran.H2	H_Name_A	
MED_STOCK	Dhahran.H2	Generic_Name	
MED_STOCK	Dhahran.H2	Brand_Name	
ST_SUPPLIER	Dhahran.H2	Supplier_Name	
ST_SUPPLIER	Dhahran.H2	Supplier_Name_A	
MAST_DICT_CNTRY	Dhahran.H2	Country_Name	
MAST_DICT_CNTRY	Dhahran.H2	Country_Name_A	
COMPANY_CONTACT	Dhahran.H2	Company_SubTitle	
COMPANY_CONTACT	Dhahran.H2	Company_MainTitle	
CP_LAB_REQUISITION	Dhahran.H2	LabReq_Desc	

6.2 Experimental Results of Anomalies Detections

An experiment of matching process of an algorithm of equations (4), (5) and (6) is performed. Factors s and d are showing matching anomaly results of structure and domain instance values respectively. Time t is a factor that shows slowness of searching due to granular complexity of structure described in an equation (3) and resources such as distant network traffic, busy resources etc. Various experiments of matching strings such as searching of "name" are performed. Results are extracted similar to list of matched strings of Table 2. In figure 5, there is a group of five anomalies are detected near t = 7 with large values of s and t. This means large number of s and d predicts high time t values and vice versa. This validates an impact of an equation (7) and anomalies are clustered with large value impacts of s, t, and d axes.

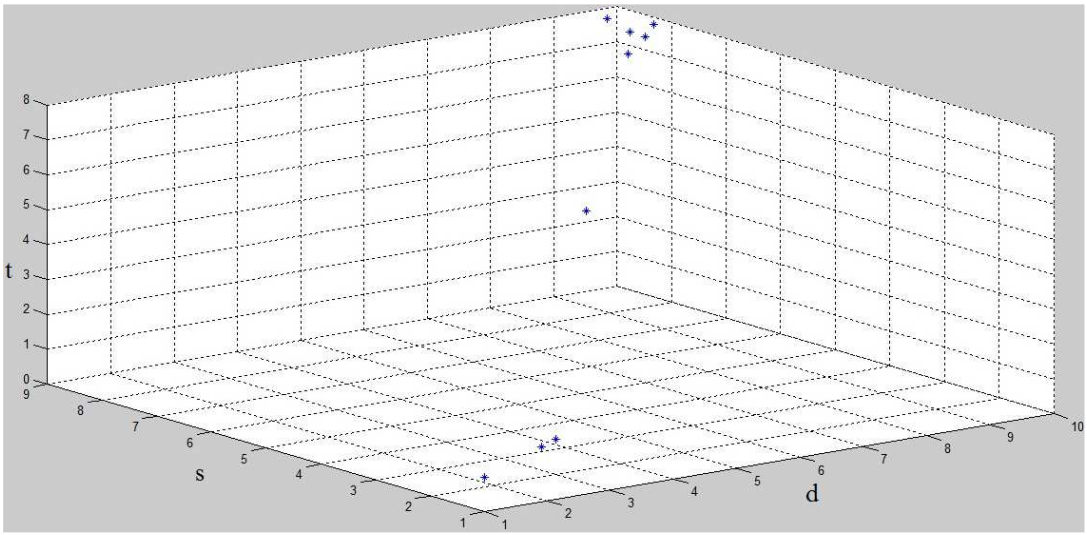

Figure 5: Detections of data anomalies of matching string "name"

During experimentation of detecting anomalies, medical practitioners have explored an anomaly that is some patients are visited more than one hospitals with same disease. Data has been searched and collected using instance level matching algorithm equation (4) as shown in figure 6.

Data for a period of Jan-Apr

Infectious Disease	City Hospital Dhahran (H2)		Central Hospital Dammam		Riyad Hospital (H4)	
	H2-Single	H2-Dup	H3-Single	H3-Dup	H4-Single	H4-Dup
Diphtheria	31	4	29	6	33	2
Measles	5	2	12	2	25	0
Mumps	22	1	31	1	24	1
Yellow Fever	24	2	71	1	84	3
Total	82	9	143	10	166	6

Figure 6: Data of infectious diseases from federated data sources of three hospitals

In figure 6, number of patients are reported once in one hospital under columns H2-Single, H3-Single and H4-Single whereas number of patients who visited more than one hospital (duplicated) with same disease under columns H2-Dup, H3-Dup and H4-Dup. In figure 7, graph shows overlap of number of records in each disease, represents a kind of anomaly.

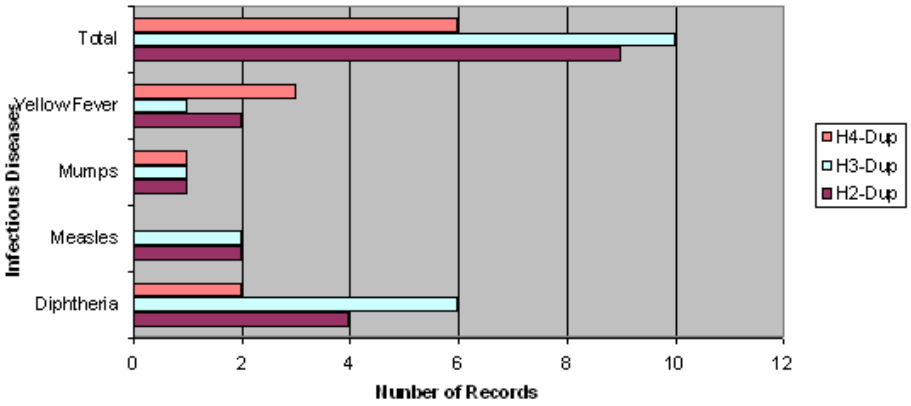

Figure 7: Number of same patients is reported in more than one hospital

7 Conclusion and Future Works

In this paper, we discussed a challenging and critical problem of state-of-the-art data anomalies emerged from data matching and mapping. In TDE, data is usually distributed amongst heterogeneous data sources. Such data is managed and designed locally by multi-vendors of the data sources, which have no ability to match or map data fully across federated DBMSs. Algorithm along with work flow of matching and mapping provides a basis for detecting and resolving anomalies. Our discussion of data matching when number of DBMSs is integrated through staging DBMS, analytical interpretation of algorithm with implementation will serve as a comprehensive source for TDE and EDE. Additionally, our algorithm is validated by using case study of federated databases of hospitals. This identifies how anomalies are significant by using searching information with different meanings and interpretations. Performance issues with big data can be further explored in cloud environment. Data mining techniques can also be applied for further data matching enhancements and secured search is being performed on clouds [22]. Our research work emphasizes on the detection of anomalies of various matches, it can be further utilized across other software domains such as data warehousing, data integration, cloud based systems, enterprise information system [25] and knowledge based systems etc.

REFERENCES

1. Rezenda, F., Georgian, U. H., Rutschlin, J.: A practical approach to access heterogeneous and distributed databases. CAiSE*99, Berlin Heidelberg, pp. 317-332 (1999)
2. Foster, I., Kesselman, C., Nick, J., Tueke, S.: Grid services for distributed system integration. IEEE computer, 35(6), pp. 397-398 (2002)
3. Rahm, E., and Bernstein, P. A.: A survey of approaches to automatic schema matching, Online publication, VLDB Journal 10, pp. 334-350, (2001)
4. Cali, C. A., Calvanese, D., Giacomo, G., Lenzerini, M.: Data integration under integrity constraints, CAiSE, Springer LNCS 2348, pp. 262-279, (2002)

5. Ahmed, E., Bessis, N., Yue, Y., and Stephens, D.: data loading and mapping using staging DBMS in the grid, 21st IEEE Annual Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1887-1893, Ontario, Canada, (2008)
6. R. J. Miller, L. M. Haas, M. A. Hernandez, "Schema Mapping as Query Discovery", Proceedings of the 26th VLDB Conference, Cairo, Egypt, pp. 77-88, (2000).
7. Larson, J. A., Navathe, S.B., El-Masri, R.: A theory of attribute equivalence in databases with application to schema integration. IEEETrans Software Eng 16(4), pp. 449-463, (1989)
8. Bessis, N., French, T., Burakova-Lorgnier, M., Huang, W.: Using grid technology for data sharing to support intelligence in decision making, Xu, M. (ed) Managing Strategic Intelligence: Techniques and Technologies, Idea Group Publishing Inc, pp. 179-201, (2007)
9. Alpdemir, N., Mukherjee, A., Foster, I., Paton, N.W., Watson, P., Fernandes, A. A. A., Gounaris, A., and Smith, J., Service-based distributed query processing on the grid, 1st Intl. Conference on Service-Oriented Computing (ICSOC), Springer-Verlag, Trento, Italy, pp. 467-482, (2003).
10. Austin, J., Turner, A., and Alwis, S., Grid enabling data de-duplication, 2nd IEEE International Conference on e-Science and Grid Computing, pp. 2-8, (2006)
11. Austin, J., Davis, R., Fletcher, M., Jackson, T., Jessop, M., Liang, N., Pasley A.: DAME: Searching large data sets with in a grid-enabled engineering application, Proceedings of the IEEE, Vol. 93, pp. 496-509, (2005)
12. Ahmed, E., Bessis, N., Yue, Y., and Sarfraz, M.: Matching multiple elements in grid databases: a practical approach, 3rd International Conference on Digital Information Management (IEEE-ICDIM), London, U.K., (2008)
13. Truban, E., Sharda, R., and David, J. E. A., Business Intelligence: A Managerial Approach, Upper Saddle River, N. J., (2008)
14. Ahmed, E., Bessis, N., Yue, Y., Norrington, P., and Sarfraz M.: Data mapping, matching and loading using Grid services, 24th AINA IEEE International Conference, Advanced Information Networking and Applications, pp. 679-775, Singapore, (2011)
15. Raman, V., Narang, I., Crone, C., Haas, L., Malaika, S., Mukai, T., Wolfson, D., and Baru, C.: Services for Data Access and Data Processing on Grids, IBM Almaden Research Center and IBM Silicon Valley lab, DAIS Working Group, Global Grid Forum, (2003)
16. Bessis, N., Sotiriadis, S., Xhafa, F., Pop, F., Cristea, V.: Meta-scheduling Issues in Interoperable HPCs, Grids and Clouds, International Journal of Web and Grid Services, InderScience, Vol. 8(2), ISSN: 1741-1106, (2012).
17. Bell, G. S., Sethi, A.: Matching records in a national medical patient index, CACM 44(9), pp. 83-88, (2001).
18. P. Ajitha and E. Chandra, A Survey on Outliers Detection in Distributed Data Mining for Big Data, Journal of Basic and Applied Scientific Research, TextRead Publication, vol. 5(2), pp. 31-38, (2015).
19. Isele, R. and C. Bizer, Learning expressive linkage rules using genetic programming, Proceedings of the VLDB Endowment, **5**(11): p. 1638-1649, (2012).
20. Neepa K. Shah, Big data and Cloud computing: Pitfalls and Advantages in Data Management, International Conference on Computing for Sustainable Global Development, IEEEXPlore CFP1583W-ART, India, March (2015).
21. M. G. de Carvalho, A. H. F. Laender, M. A. Goncalves, and A. S. da Silva. A genetic programming approach to record duplication, IEEE Transactions on Knowledge and Data Engineering, vol. 24(3):399–412, (2012).
22. Wei Zhang, Yaping Lin, Sheng Xiao, Jie Wu and Siwang Zhou, Privacy Preserving Ranked Multi-Keyword Search for Multiple Data Owners in Cloud Computing, IEEE TRANSACTIONS ON COMPUTERS, vol. 65(5), MAY (2016).
23. Q. Li, X. Wang, W. Li, J. Li, C. Wong, and R. Du, Application integration in a hybrid cloud computing environment: Modeling and platform, Enterprise Information Systerm, vol. 7(3), pp. 237-271, (2013).
24. E. Ahmed, N. Bessis, Y. Yue, and M. Sarfraz, "Data Mapping, Matching and Loading using Grid Services", 24th AINA IEEE International Conference, Advanced Information Networking and Applications, pp. 1158-1164, Perth, Australia, (2010).
25. J. Aliakbari, A. Hassanzadeh, A. Aliakbari, and M. Taherkhani, The Investigation of the Impacts of Advanced Consistency Mechanism on Organizational Process Integration, Journal of Basic and Applied Scientific Research, TextRead Publication, Vol. 3(5), pp. 27-33, (2013).