ICIC Express Letters
Volume **10**, Number **12**, December **2016**
 ISSN 1881-803X
pp. **2829–2835**

ACTIVE INCONSISTENCY DETECTION IN DATA INTEGRATION

Pei-Ju Lee[1], Yi-Chih Hsieh[2] and Yung-Cheng Lee[3,*]

[1]Department of Information Management
National Chung Cheng University
No. 168, Sec. 1, University Rd., Min-Hsiung Township, Chia-Yi County 621, Taiwan
pjlee@mis.ccu.edu.tw

[2]Department of Industrial Management
National Formosa University
No. 64, Wunhua Rd., Huwei Township, Yunlin County 632, Taiwan
yhsieh@nfu.edu.tw

[3]Department of Security Technology and Management
WuFeng University
No. 117, Sec. 2, Chiankuo Rd., Minhsiung, Chia-Yi County 62153, Taiwan
*Corresponding author: yclee@wfu.edu.tw

Received May 2016; accepted August 2016

Abstract. *The challenges of data integration involve inconsistent data and redundant data in the tremendous volume of heterogeneous data sources. A uniform schema for integrated data is time-consuming and cannot detect data inconsistency before merging. In this paper, we propose a conflict detection system that utilizes the linear system with temporal or spatial data. The system generalizes estimated value for each time- or space-interval as well as the delta value to indicate the conflict level for multiple overlapping reports. We build upon these foundations to develop an interpretation in terms of multisensory integration and data fusion.*
Keywords: Information integration, Data fusion, Consistency, Data conflict

1. **Introduction.** The amount of data generated from various data sources is growing rapidly. In addition to the basic database management problem, the data integration issue becomes more important. In basic database integration process, each database has to be unified into one global schema; however, some data conflicts such as time/location/name confliction will not be revealed at the stage of integration. This data conflict is not easy to be detected since the data are following the same schema but it can cause tremendous errors of the query tables if users adopted the wrong number of cases. The major problem for both multisensory data fusion and information integration data fusion is the tremendous volume of heterogeneous data. Take multisensory data fusion as an example, types of sensors are usually classified by their physical nature such as electromagnetic spectrum, vision (e.g., video camera), sound waves (e.g., sonar), touch (e.g., tactile sensor), odor, or the absolute position of the system (e.g., rangefinder) [1]. Systems usually use multiple types of sensors or duplicate sensors for their tasks. Therefore, merging data from large-scale data resources becomes critical.

With the improvement of the Internet speed and storage devices, access data sources worldwide to obtain more information becomes easier but integrating these data from these heterogeneous data sources becomes more difficult. The two major challenges of largescale data integration, heterogeneous data and conflicting data start getting more attention [2]. Temporal and spatial databases include a wide range of time duration and area coverage of numerous events and they may have overlap due to the type-in error or redundancy of records. To utilize the temporal and spatial data sources in a global

repository, some problems such as data inconsistency may be encountered. From the database point of view, data integration may be performed when there is heterogeneity at the schema level, tuple level, or value level. The heterogeneous data may have tuple conflicts; therefore, we cannot perform operations over domains even though these are semantically similar [3]. Other studies analyze the relation of the relevant information from distributed data sources to prevent data conflict; however, these methods spend a large amount of time and effort to examine records from data sources. The techniques used in these methods are such as Data Exchange that adopts possible answers to represent each single but conflict tuple in [4] or Conflict Handling in [5,6] that utilized various strategies: conflict ignoring, conflict avoiding, or averaging in the global repository. These methods require techniques to examine the conflict data details in order to change these data values and this complicated process becomes the disadvantage of traditional integration methods.

In order to reduce the computation cost and time of comparison in the traditional methods, we propose the usage of the linear programming algorithm performing the internal reliability assessment of integrated data. The linear programming algorithm can detect the existence of inconsistent events in a global repository. This inconsistency detection is conducted before the detailed data comparison as well as the merging process; therefore, this linear programming algorithm can reduce computational costs. The proposed algorithm can increase users' awareness of data conflict without exhaustive comparisons. The remainder of this paper is structured as follows. Related work and problem statement will be introduced in Section 2. The proposed algorithm and methodologies will be explained in Section 3. The experimental results will be shown in Section 4. Section 5 concludes discussion and application of our proposed algorithm.

2. **Problem Statement and Preliminaries.**

Regarding the data characteristics of time and space, dynamic changing and records continuity, the temporal and spatial database integration requires a comprehensive consolidation of data sets. These continuous data reports exist in different areas such as environmental data (ex. climate change), health data (ex. disease contagion), biological data (ex. species migration), or financial data (ex. stock rating) that data analysis is based on events within some time intervals. The consolidated data composed of heterogeneous data sources and various time intervals that involve the great potential of data overlap in time, name, and location. This paper focuses on solving data conflicts on temporal and spatial data reports.

The earliest approaches to data integration are given by U.S. Naval Observatory in 1960. The data fusion model is divided into functional model (i.e., model contains primary functions, relevant database, and interconnectivity to perform data fusion), architectural model (i.e., focus on the hardware/software, the data flow and external operator interfaces), and mathematical model (i.e., describes the algorithm performing data fusion and logical process) in [7]. Data fusion is most used in multisensor environment and the advantages of using multiple sensors over a single sensor including higher signal-to-noise ratio, robustness and reliability in the evidence of sensor failure, parameter coverage, dimensionality of the measurement, confidence and resolution, hypothesis discrimination with the aid of more complete information arriving from multiple sensors, obtaining information regarding independent features in the system, and lower uncertainty, measurement time, as well as possible costs [1,6,8,9].

The data fusion process can happen in a hierarchical or sequential manner; moreover, a hybrid of these two. Some data integration models and their process levels are described in [1] such as Thomopoulos architecture that is divided into signal level fusion (i.e., data correlated through learning), evidence level fusion (i.e., data correlated through statistical model or decision making), and dynamics level fusion (i.e., data correlated through mathematical models) [10]; Luo and Kay's framework is divided into signal, pixel, feature,

and symbol levels of fusion as the level of representation increases from signal to symbol, and the level of information provided to users also increases [8]; or the Waterfall model is divided into signal (i.e., preprocessing the raw data), feature (i.e., feature extraction and pattern processing), and interrogation (i.e., situation assessment and decision making) levels [11].

Since big data has become a popular topic nowadays, many studies have focused on the techniques to process the tremendous amount of data; these techniques are such as the MapReduce which allows users to process large datasets [12], Hadoop [13], and Sparks [14]. In addition, the thriving of social network made a number of studies find that these techniques not only need to deal with the traditional data types such as image, text, or audio but also need to focus on data sets and the platforms of social media such as Facebook or Twitter [15,16].

The major problems that multisensory data fusion faces are the huge amount of heterogeneous data and time-consuming integration process. In this paper, we consider merging data of reports from heterogeneous data sources with temporal or spatial overlapping of events. In addition, we also consider the overlap condition of these events. To analyze the dynamic trend of the reports and find a potential conflict between them, our system models these overlapping reports into an underdetermined linear programming. The solution sets of this model are generated from the least square method, which provides the estimation value of the reports. The model can also be used as an indicator showing reports consistency (i.e., values are consistent across reports or there are conflicts in report values).

3. **Methodologies.** We adopt the linear programming model for merging data from heterogeneous data sources with temporal or spatial overlapping of events. In a temporal database, the reports represent the occurrence of events (i.e., diseases, temperature changes, etc.) at specific locations. R_i represents report ID and the corresponding number of cases is noted as V_i where i stands for report ID. These reports can be allocated on a timeline by their occurrence time (the earliest reported data to the latest), and the timeline will be separated by intervals sorted between reports time. The number of the interval is varied by overlapping condition of reports: more overlapping, more intervals and finer units of intervals. The range of the number of intervals on each timeline varies from 1 to $2n-1$ (n: the number of the report); in other words, there will be six intervals if we have four reports. These intervals are noted as X_j where j stands for interval ID. Ideally, reports from heterogeneous data sources may be redundant but are not inconsistent with reports values. Each interval there will have an identical value between reports and the summation of corresponding intervals will be equal to the actual value of the report. However, if the system cannot identify identical values for these intervals between reports then these incomparable interval values indicate the existence of report inconsistency.

When data sources are integrated, reports can be grouped in several linear systems depending on their overlapping conditions. The unknown variable vector $\boldsymbol{X}$ represents unknown event density for each time interval. The size of vector $\boldsymbol{X}$ depends on the overlapping condition of these reports; in other words, n is different for every linear system. The coefficient matrix $\boldsymbol{A}$ denotes the existence of reports at a corresponding time interval of $\boldsymbol{X}$. The aggregated statistic value of reports represents as a constraint value vector $\boldsymbol{b}$.

Definition.

- R_i reports represent the occurrence of events
- V_i report values
- X_j unknown event density for each interval
- $\boldsymbol{A}$ the existence of reports at corresponding interval of $\boldsymbol{X}$

$\boldsymbol{b}$ the aggregated statistic value of reports

Assumption.
The linear system in matrix form is represented as

$$\boldsymbol{AX} = \boldsymbol{b} \tag{1}$$

The non-negative least square method [17] is used in this paper to solve linear equations (1) and $\boldsymbol{X}$ can be computed by

$$\boldsymbol{X'} = \boldsymbol{A^T} \left(\boldsymbol{AA^T}\right)^{(-1)} \boldsymbol{b} \tag{2}$$

The estimated report value $\boldsymbol{b'}$ will be generated by

$$\boldsymbol{AX'} = \boldsymbol{b'} \tag{3}$$

Then we check if the report value $\boldsymbol{b'}$ is the same with the real report value $\boldsymbol{b}$ without any error and $\boldsymbol{b'} = \boldsymbol{b}$. Otherwise, we will have non-zero delta ($\boldsymbol{\delta}$) for

$$\boldsymbol{AX} + \boldsymbol{\delta} = \boldsymbol{b} \tag{4}$$

Model.

$$\begin{aligned} \text{Minimize} \quad & \|\boldsymbol{b} - \boldsymbol{AX}\|_2^2 && (5)\\ \text{s.t.} \quad & \boldsymbol{AX} = \boldsymbol{b} && (6)\\ & \boldsymbol{X} \geq 0 && (7) \end{aligned}$$

Proof:

$$\begin{aligned} & \|\boldsymbol{X}\|^2 + \boldsymbol{\lambda}^T(\boldsymbol{b} - \boldsymbol{AX}) && (8)\\ & 2\boldsymbol{X} - \boldsymbol{A}^T\boldsymbol{\lambda} = 0 && (9)\\ & 2\boldsymbol{AX} - \boldsymbol{AA}^T\boldsymbol{\lambda} = 0 && (10)\\ & 2\boldsymbol{b} - \boldsymbol{AA}^T\boldsymbol{\lambda} = 0 && (11)\\ & 2\boldsymbol{b} = \boldsymbol{AA}^T\boldsymbol{\lambda} && (12)\\ & \boldsymbol{\lambda} = 2\left(\boldsymbol{AA}^T\right)^{-1}\boldsymbol{b} && (13)\\ & \boldsymbol{X} = \boldsymbol{A}^T\left(\boldsymbol{AA}^T\right)^{-1}\boldsymbol{b} && (14) \end{aligned}$$

Below is the pseudo code for this linear model:

The non-negative least square method to solve linear systems:

Input: matrix $\boldsymbol{A}$, matrix $\boldsymbol{b}$.
Suppose $\boldsymbol{AX} = \boldsymbol{b}$, and $\boldsymbol{X} = \boldsymbol{A}^T\left(\boldsymbol{AA}^T\right)^{-1}\boldsymbol{b}$.
$\boldsymbol{X'} = \boldsymbol{A}^T\left(\boldsymbol{AA}^T\right)^{-1}\boldsymbol{b}$; $\boldsymbol{AX'} = \boldsymbol{b'}$.
if $\boldsymbol{b'} = \boldsymbol{b}$ then $\boldsymbol{AX'} = \boldsymbol{b}$; $\boldsymbol{X'} = \boldsymbol{X}$ is the solution set for system;
else
$\boldsymbol{b'} \neq \boldsymbol{b}$, $\boldsymbol{b'} + \boldsymbol{\delta} = \boldsymbol{b}$;
$\boldsymbol{AX'} + \boldsymbol{\delta} = \boldsymbol{b}$, $\boldsymbol{\delta} \neq \boldsymbol{0}$; there is an error existing in $\boldsymbol{A}$ or $\boldsymbol{b}$.
end

4. **Experimental Results.** In this section, we experimentally evaluate the different report values and overlapping structures for the proposed linear model, and their impact on inconsistent report values detection. Experiments were conducted using simulated datasets of two reports B$(1,1)$, indicating report 1, and B$(2,1)$, indicating report 2. These two reports reflect two data sets of the same entity name (i.e., for the same disease, the same area, etc.) but report from different repositories with different recorded time. We placed these two reports fully overlapped, and report B$(1,1)$ is fully subsumed by report

$B(2,1)$ (i.e., $B(1,1)$ has a shorter period of time recorded compared with $B(2,1)$). We initially consider no conflict values between these two reports and the initial delta value will be zero. In condition 1 (figure upper left), the initial values for both reports are the same as 100, and then the value of $B(1,1)$ increases 10 and the value of $B(2,1)$ decreases 10 in each following round; i.e., at the second round, $B(1,1)$ has the reported value 110 and $B(2,1)$ has the reported value 90. This induces conflicting between these two reports since $B(1,1)$ has higher value but shorter period of time compared with $B(2,1)$. Under this circumstance that two reports are overlapped and one is exactly subsumed by the other, any simple discrepancy of report values will cause inconsistency and mismatch of these data sets. Condition 2 has the report value of $B(1,1)$ increases 10 and report value of $B(2,1)$ remains the same, Condition 3 has the report value of $B(1,1)$ increases 100 and report value of $B(2,1)$ remains the same, and Condition 4 has exactly the same report value of $B(1,1)$ and $B(2,1)$. The settings of these four conditions are also explained in Table 1.

The value difference between reports $B(1,1)$ and $B(2,1)$ is increasing while the summation of delta value is also increasing. In the previous session, we have introduced that the non-zero delta value can be used to represent the discrepancy between report values since we will have zero delta value if the report value b' is the same with the real report value b. The delta values of four conditions in each run are shown in the y-axis and the iterations of each run are shown in the x-axis. In Figure 1, the delta value is changing in different conditions: in Condition 1 (figure upper left), the delta value increases proportionally while the difference between two reports increases; in Condition 2 (figure upper right), the value of one report remains the same and the other with report value keeps increasing. The delta value still increases proportionally with the report value difference; Condition 3 (figure bottom left) follows the same rule of Condition 2 but has larger report values, and the delta value is larger than that in Condition 2 with the same trend; and Condition 4 (figure bottom right) shows the case that values of two reports increase at the same time; therefore, data conflict will not happen and the delta value remains zero.

Table 1. Experiment conditions

<table>
<tr><td></td><td colspan="2">x-axis: Simulation round</td></tr>
<tr><td rowspan="2">y-axis: Report value differences</td><td>[Condition 1]

Initial report value: $B(1,1) = 100$, $B(2,1) = 100$.
Iteration:
Report value of $B(1,1)$ increases 10, and report value of $B(2,1)$ decreases 10; i.e., in round 2, $B(1,1) = 110$, $B(2,1) = 90$.</td><td>[Condition 2]

Initial report value: $B(1,1) = 100$, $B(2,1) = 100$.
Iteration:
Report value of $B(1,1)$ increases 10, and report value of $B(2,1)$ remains the same; i.e., in round 2, $B(1,1) = 110$, $B(2,1) = 90$.</td></tr>
<tr><td>[Condition 3]

Initial report value: $B(1,1) = 0$, $B(2,1) = 100$.
Iteration:
Report value of $B(1,1)$ increases 100, and report value of $B(2,1)$ remains the same; i.e., in round 2, $B(1,1) = 200$, $B(2,1) = 100$.</td><td>[Condition 4]

Initial report value: $B(1,1) = 100$, $B(2,1) = 100$.
Iteration:
Report value of $B(1,1)$ increases 10, and report value of $B(2,1)$ also increases 10; i.e., in round 2, $B(1,1) = 110$, $B(2,1) = 100$.</td></tr>
</table>

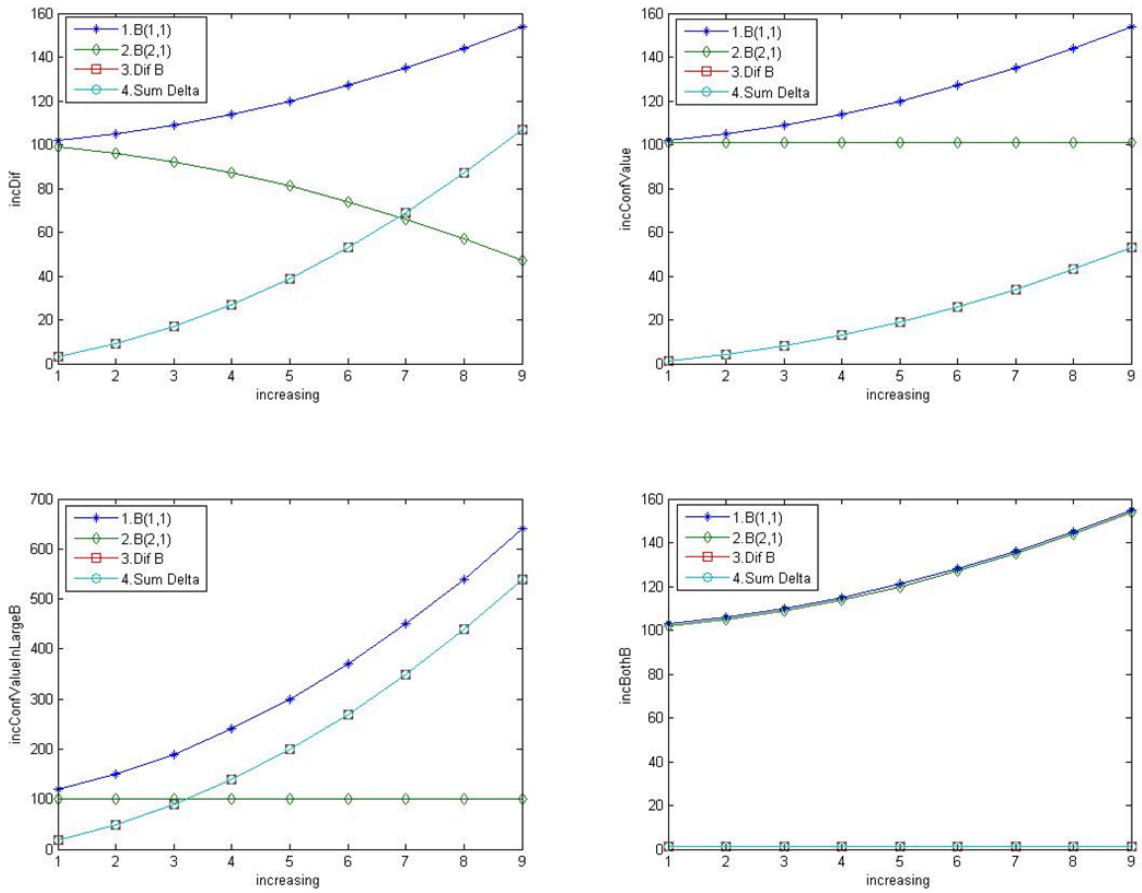

FIGURE 1. The delta value

5. **Conclusions.** The main contribution of this paper is to provide an efficient approach to detect report value inconsistency before merging the data from heterogeneous sources. The proposed model includes inconsistency detection and level of inconsistency indication. The value difference between reports with overlap affects the delta value when there are conflicts between these reports. The inconsistency detecting is getting more important while analysis of large quantities of data is thriving. Our work on the linear system continues along several directions. We are studying the extent to which structure of reports affects the conflict level, and adapting our model to have better performance for different conditions. We are also in the process of deploying our system with collaborative data.

REFERENCES

[1] J. Esteban, A. Starr, R. Willetts, P. Hannah and P. Bryanston-Cross, A review of data fusion models and architectures: Towards engineering guidelines, *Neural Computing and Applications*, pp.1-27, 2005.

[2] V. Zadorozhny and Y.-F. Hsu, Conflict-aware historical data fusion, *Scalable Uncertainty Management*, pp.331-345, 2011.

[3] L. G. DeMichiel, Resolving database incompatibility: An approach to performing relational operations over mismatched domains, *IEEE Trans. Knowledge and Data Engineering*, vol.1, pp.485-493, 1989.

[4] R. Fagin, P. G. Kolaitis and L. Popa, Data exchange: Getting to the core, *ACM Trans. Database Systems*, vol.30, pp.174-210, 2005.

[5] J. Bleiholder and F. Naumann, Data fusion, *ACM Computing Surveys*, vol.41, pp.1-41, 2008.

[6] F. Naumann, A. Bilke, J. Bleiholder and M. Weis, Data fusion in three steps: Resolving inconsistencies at schema-, tuple-, and value-level, *IEEE Data Eng. Bull*, vol.29, pp.21-31, 2006.

[7] D. L. Hall and S. A. H. McMullen, *Mathematical Techniques in Multisensor Data Fusion*, Artech House Publishers, 2004.
[8] R. C. Luo and M. G. Kay, Multisensor integration and fusion in intelligent systems, *IEEE Trans. Systems, Man, and Cybernetics*, vol.19, pp.901-931, 1989.
[9] J. K. Hackett and M. Shah, Multi-sensor fusion: A perspective, *IEEE Robotics and Automation*, pp.1324-1330, 1990.
[10] S. C. A. Thomopoulos, Sensor integration and data fusion, *Robotic Systems*, vol.7, pp.337-372, 1990.
[11] C. J. Harris, A. Bailey and T. J. Dodd, Multi-sensor data fusion in defense and aerospace, *The Aeronautical Journal*, vol.102, pp.229-244, 1998.
[12] J. Dean and S. Ghemawat, Mapreduce: Simplified data processing on large clusters, *Symposium on Operating Systems Design and Implementation*, 2004.
[13] T. White, *Hadoop: The Definitive Guide*, O'ReillyMedia, 2009.
[14] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker and I. Stoica, Spark: Cluster computing with working sets, *USENIX Conference on Hot Topics in Cloud Computing (HotCloud)*, p.10, 2010.
[15] G. Bello-Orgaz, J. Jung and D. Camacho, Social big data: Recent achievements and new challenges, *Information Fusion*, vol.28, pp.45-59, 2016.
[16] S. Caton, C. Haas, K. Chard, K. Bubendorfer and O. F. Rana, A social compute cloud: Allocating and sharing infrastructure resources via social networks, *IEEE Trans. Services Computing*, vol.7, pp.359-372, 2014.
[17] B. K. P. Horn and M. J. Brooks, Solving over- and under-determined sets of equations, *LISP*, 1981.