Directed bipartite Hypergraph: Representation of data edits for constraint-based data cleaning

Wa Ode Zuhayeni Madjida¹, Takdir², Sulisetyo Puji Widodo¹

¹Directorate of Statistical Information System, BPS – Statistics of Indonesia
²Politeknik Statistika STIS

Email: zuhayeni@bps.go.id

Abstract. Constraint-based data cleaning captures data violations to a set of constraints called data quality constraints. Data edits is one of constraint type besides integrity constraint that used for checking data inconsistencies which come from census or survey questionnaire (questionnaire schema). Data edits contain some variables and describe their relationship using AND and OR operator. The relationship needs to be represented in a structure that can find the best data repair solution. Graph is a generic structure to represent a relationship. In previous studies, hypergraph is used as a solution to represent variable relationships of the violated integrity constraint. Such solution is not efficient for data edits. Hypergraph cannot show the relationship between data edits as a whole. This can trigger more new errors. In this paper, we use graph representation namely directed bipartite hypergraph to illustrate the relationship between overall data edits. Nodes in the graph not only contain variable information of data edits, but also the data edits itself. This makes the interaction between data edits can be seen as a basis to prevent new errors. We also introduce four parameters as determining the level of variables that are priorities for improvement. The goal is to minimize the number of variables must be fixed, but can eliminate all violations that occur. We evaluate the quality of the proposed structure by simulating data repairing. The results show that 100% of the data has decreased violations. 84% of them can be repaired to zero violations.

1. Introduction
Data cleaning is an important process to guarantee data quality. The challenge is how to decide that the data is clean. Constraint-based is one approach to answer that issue. The constraint-based method detects anomalies in data based on specific constraints. Data is said to be dirty if violates the constraint. The constraint is formulated based on data heuristics or built by experts.

There are two kinds of constraints, namely integrity constraint and data edits [1]. Integrity constraint defines rules into relational database schema, while data edit defines rules in questionnaire schema separated from the database schema. The rules made implicitly state relations between attributes in data that will affect the values that can be applied to those attributes. In constraint-based data cleaning, both based on integrity constraints and data edits, data repair is accomplished by making each row of data meet the constraints or rules that have been defined.

Various studies proposed constraint-based data repair algorithms, such as [2] [3] [4] [5] [6] [7] [8]. However, the algorithm does not explore a relationship and interaction between the rules made. It is very important to know the relationship between defined rules because it affects the data repairing process. Fixing violations on one rule can trigger new violations. This makes minimum repair [9] more challenging.
In [9], they developed a minimum repair algorithm by using hypergraph to describe relationship between violations of integrity constraints that occur. The relationship illustrated in the hypergraph will produce a minimum vertex cover (MVC) that is expected to be an optimum repair candidate. [10] also utilized this structure in developing its data repair algorithm. However, this structure cannot be applied to data edits constraint, because of rule characteristics and structure differences, namely the AND and OR operators in data edits rule. On the other hand, only representing violations is still not enough to maximize the data repair algorithm. The algorithm does not explore the relationship and interaction between violated rules and clean rules. Improvement of data based on violated rules can introduce new violations from previously clean rules, even with a greater number of violations than before.

This paper introduces a representation of data edits which provides an overview of relationships between all defined rules, both those violated by data and those are not. Graphs have proven as excellent languages for communicating the relationship of problem-domain variables [11]. This paper takes advantage of graph to represent data edits structure called directed bipartite hypergraph. This structure describes how an attribute or variable is involved in several rules. This involvement can become a weight in determining steps for repairing data. We also present four parameters to evaluate the candidate variable of data improvement. The result indicates the priority of variables in data repairing, whereby making minimum changes, all violations can be eliminated.

We verify the proposed structure by implementing a data repair algorithm in [12]. Assumptions are used to assess the use of directed bipartite hypergraphs:

- Graph structure is appropriate if there is some reduction of violations number.
- The level node (based on the proposed parameter) indicates a priority attribute that must be corrected. Priority can deliver a trend of decreasing the number of violations, even eliminating all violations that occur.

Organization of this paper as follows: Section 2 explains data edits structure. Section 3 explores previous studies related to constraint-based data cleaning. We introduce directed bipartite hypergraph in Section 4. Experiments are reported in Section 5. We explain how the structure is used to repair data and the results itself in this section. Finally, we give concluding remarks in Section 6.

2. Structure of Data Edit

In statistical fields, data are collected based on census questionnaires that follow a structure called questionnaire schema. The rules defined are very similar to the structure of rules in integrity constraints, specifically Conditional Functional Dependency (CFD) [13][14]. Both of them consist of left hand side (LHS) and right hand side (RHS). If values on LHS are met, then data must satisfy the requirements on RHS. The relationship between attribute value in questionnaire is defined as a constant resembles the structure of pattern tableau in constant-CFD, because it determines value points that must be met by an attribute. However, constant-CFD only defines AND (\(\land\)) relationship between attributes, not include OR (\(\lor\)) relationship. In mathematical notation, we denote a rule derived from data edits rules by including constants directly in the attribute. The rules that must be met are then used in the analysis and design of data repair algorithms.

For example, there are inconsistent answers on questionnaire:

Marital status = ‘married’, age = ‘5 years old’

Rules for detecting violations:
If marital status is married, age must not be less than 14

These rules can be made in form of an edit that expresses violation conditions, namely:
Marital status = ‘married’ \(\land\) age < 14

Based on the rule above, we can specify a pattern that must be met by data:
If marital status is married, then age must be equal or greater than 14 years old
‘Married’ is constant for ‘marital status’, and ‘equal or greater than 14 years old’ is constant for ‘age’. Denoted that $\varphi$ is data edits rule, which refer to rules that must be satisfied by data, then the rule mathematically can be noted as:

$$\varphi: \text{[marital status = ‘married’]} \rightarrow \text{[age} \geq 14\text{]} \tag{1}$$

3. Literature Review

To design data repair algorithms on data cleaning process, [9] represents integrity constraint violations for functional dependency (FD) and conditional functional dependency (CFD) by using undirected hypergraph. [10] improved this study to design holistic data repairing algorithm. The violation representation helps algorithm to define attribute that must be repaired and get interaction between other violation.

We give an illustration how previous study do their violation representation. Consider data in table 1, and following rules ($\varphi_1$ and $\varphi_2$ are FD, $\varphi_3$ is CFD):

$\varphi_1: A \rightarrow B$

$\varphi_2: C \rightarrow B$

$\varphi_3: R[D = 2] \rightarrow R[B = 2]$

|        | A | B | C | D |
|--------|---|---|---|---|
| t1     | 1 | 3 | 1 | 1 |
| t2     | 1 | 2 | 3 | 1 |
| t3     | 2 | 3 | 3 | 2 |

Figure 1 shows conflict hypergraph, namely hypergraph that built on rule violations of $\varphi_1$, $\varphi_2$, and $\varphi_3$. Based on the image, we can see that the rule is represented in one edge. One edge may cover several nodes, where the nodes represent the attribute involved in the rule.

Figure 1. Example of Conflict Hypergraph

Undirected hypergraph that is used cannot describe AND and OR operators of the rules. In addition, representations that are only carried out on violations can trigger more new violations, even though the rules were previously clean. To overcome this problem, this paper proposes a representation of data edits in form of directed bipartite hypergraph.

4. Proposed Design

In this section, we first introduce rule parser. This step generalize data edits rule before represent the rule in directed bipartite hypergraph.

4.1. Rule Parser

Rule parser is used for generalization of all given rules. If a rule contains two operators AND and OR, the rule would be converted to disjunctive normal form (DNF) by using distributive law. The
conversions were performed because of the use of directed hypergraph as graph representation. In directed hypergraph, nodes with AND operator is fused in one edge, while nodes with OR operator have a separate edge. Additionally, rule parser would elaborate rule expression which contains operator != (not equal), <, ≤, >, ≥ based on domain attribute. Illustration:

\[ \varphi_1: [A \neq \text{null}] \rightarrow [B = \text{null}]; \text{dom}(A) = \{1, 2\} \]

The rule become:

\[ \varphi_2: [A = 1] \lor [A = 2] \rightarrow [B = \text{null}] \]

At this phase, condition \([A! = \text{null}]\) in the rule will be mapped to variable value domain of A and elaborating the expression into \([A = 1]\) and \([A = 2]\):

\[ \varphi_2: ([A = 1] \land [B = 1]) \lor ([A = 2] \land [B = 1]) \rightarrow [C = 1] \lor [C = 5] \]

\[ \varphi_2: ([A = 1] \land [B = 1]) \lor ([A = 2] \land [B = 1]) \rightarrow [C = 1] \lor [C = 5] \]

4.2. Data Edits Representation

We propose directed bipartite hypergraph as a model representation of data edits. Data consists of multiple attributes where the value of among attribute has a relationship. A rule in data edits states the relationship. On the other hand, a rule may have a linkage with other rules, RHS of the rule can be LHS in other rules. Thus, the node of the graph consists of two sets (bipartite graph), they are sets of attribute and sets of rules. Nodes for sets of the attributes are denoted with rectangular and circle for sets of the rule.

Here some basic form of rule, where \(\varphi_8\) is output from rule parser.

- Relatedness between two attributes.
  \[ \varphi: [A = 5] \rightarrow [B \neq \text{null}], \text{dom}(B) = \{1, 2, 3\} \]
  \[ \varphi_8: [A = 5] \rightarrow [B = 1] \lor [B = 2] \lor [B = 3] \]

  Figure 2 is graph for this rule. The graph shows that \([A = 5]\) is LHS of rule and \([B \neq \text{null}]\) is RHS of rule. \([B \neq \text{null}]\) is elaborated into multiple node based on attribute domain, namely \([B = 1]\), \([B = 2]\), and \([B = 3]\).

- There are only AND operator, involving at least two attributes on LHS. Graph can be seen at figure 3. \([A = 5] \land [B = 1]\) are fused in one edge.
  \[ \varphi: [A = 5] \rightarrow [B \neq \text{null}], \text{dom}(B) = \{1, 2, 3\} \]
  \[ \varphi_8: [A = 5] \rightarrow [B = 1] \lor [B = 2] \lor [B = 3] \]

- There are only OR operator, involving at least two attributes on LHS. Figure 4 shows the graph.
  \[ \varphi: [A \neq \text{null}] \lor [B \neq \text{null}] \lor [C \neq \text{null}] \rightarrow [D = \text{null}], \text{domain} \text{ of A, B, and C is} \{1, 2\} \]
  \[ \varphi_8: [A = 1] \lor [A = 2] \lor [B = 1] \lor [B = 2] \lor [C = 1] \lor [C = 2] \rightarrow [D = \text{null}] \]

- There are AND and OR operator in one rule.
  \[ \varphi: [A = 1] \land [B = 3] \land [[C = 3] \land [D = 1]] \lor [[C = 4] \land [D = 2]] \rightarrow [E = 5]. \]
  \[ \varphi_8: ([A = 1] \land [B = 3] \land [C = 3] \land [D = 1]) \lor ([A = 1] \land [B = 3] \land [C = 4] \land [D = 2]) \rightarrow [E = 5]. \]
A rule with another rule may have associated. Some rules below may have link with each other, RHS in a rule can be LHS in another. These association can be seen in figure 6.

\[ \varphi_1: [hamil = 1] \rightarrow [ALH \neq null] \]
\[ \varphi_2: [hamil = 5] \rightarrow [ALH = null] \]
\[ \varphi_3: [ALH = 1] \rightarrow [waktuALH \neq null] \]
\[ \varphi_4: [ALH = 5] \rightarrow [waktuALH = null] \]
\[ \varphi_5: [hamil = 5] \rightarrow [waktuALH = null] \]
\[ \varphi_6: [waktuALH = 1] \rightarrow [tptALH \neq null] \]
\[ \varphi_7: [hamil = 5] \rightarrow [tptALH = null] \]
\[ \varphi_8: [ALH = 5] \rightarrow [tptALH null] \]
\[ \varphi_9: [waktuALH = 2] \rightarrow [tptALH = null] \]
\[ \varphi_{10}: [waktuALH = null] \rightarrow [tptALH = null] \]
\[ \varphi_{11}: [waktuALH = 1] \rightarrow [siapaALH \neq null] \]
\[ \varphi_{12}: [waktuALH = 2] \rightarrow [siapaALH = null] \]
\[ \varphi_{13}: [waktuALH] = null \rightarrow [siapaALH = null] \]
This study proposed some parameters in selecting repair expression, they are:

- Number of ingoing degree that come from rule node that has error flag in bipartite-directed hypergraph (IER).
- Number of ingoing degree that come from flagged attribute node to connected error rule node (IHV).
- Number of outgoing degree to rule node that has outgoing edge to flagged attribute node (OHR).
- Number of outgoing degree to flagged attribute node from connected rule node (OHV).

5. Experiments

5.1. Experimental Settings

**Dataset.** We use National Socio-Economic Survey dataset, which is Statistics of Indonesia survey. There are 483 rules that must be satisfied by data. These rules consist of 286 rules without AND and OR operator, 95 rules that only cover AND operator, 38 rules that only OR operator, and 64 rules which consist of AND and OR. This experiment uses three scenarios derived from the dataset by generating random errors of 10%, 20%, and 30%. Table 2 shows a detailed scenario.

**Table 2.** Characteristics of dataset used.

| Scenario | Records | Rules          | Error Rate |
|----------|---------|----------------|------------|
|          |         | Without AND and OR | AND | OR | AND and OR |          |
| 1        | 50      | 286 | 96 | 38 | 64 | 10%         |
| 2        | 50      | 286 | 96 | 38 | 64 | 20%         |
| 3        | 50      | 286 | 96 | 38 | 64 | 30%         |

**Figure 6.** Graph representation of relationship between rules.
Algorithms. We implement a data repairing algorithm in [12] where undirected graph as violation representation model.

**Metric.** We evaluate the assumptions previously given using indicators: (1) there is a decrease in the number of errors in each repair (capability). Capability is obtained by comparing the number of initial violations with the number of violations at the end of repair iteration; (2) the number of remaining violations in data is zero, and; (3) the number of repair iteration to produce zero violation.

### 5.2. Experimental Results

**Table 3. Results.**

| Scenario | Error rate | Capability | Number of zero violation | Number of repair iteration of zero violation |
|----------|------------|------------|--------------------------|---------------------------------------------|
|          |            |            |                          | Min | Max | Average |
| 1        | 10%        | 100%       | 84.00%                   | 7   | 29  | 11-12   |
| 2        | 20%        | 94%        | 57.45%                   | 15  | 40  | 24-25   |
| 3        | 30%        | 96%        | 31.25%                   | 2   | 57  | 33-34   |

Table 3 shows the simulation results for all three scenarios. In scenario 1 with an error rate of 10%, capability gives 100%. This shows that overall data can provide a decrease of violations number in the last iteration of the repair process. In the second scenario with an error rate of 20%, there are 47 records or 94% records that decrease the number of violations. Another 6% of the record gives higher violations than the number of initial violations. The same results are also given in scenario 3, not much different from scenario 2 which is 96% or 48 records.

Although all three scenarios provide relatively the same capability, there are significant differences in the success of the improvement (resulting in zero violation). The higher the error rate, the lower the number of data with zero violation. Scenario 1 can repair 84.00% of data so that there are no violations. 57.45% of data can be repaired for data with an error rate of 20% and 31.25% for data with an error rate of 30%.

We illustrate some cases that makes unsuccessful to produce zero violation, as follows:

**Given rule below:**
\[ \phi: [A = 1] \rightarrow [B = 2], \text{dom}(B) = \{1, 2\} \]

Based on these rule, node [B = 2] is chosen as repairing solution. This raises new violations in other rule, namely:
\[ \phi: [C > 17] \rightarrow [B = \text{null}], \text{dom}(B) = \{1, 2\} \]

Where value of attribute B violates rule [B = null]. This kind of repairing causes unlimited looping.

Based on the rule structure, there is no relationship between nodes and rules for the same variable, so both rules do not have a connection in the graph. Figure 7 visualizes this pattern.

![Figure 7. Examples of unconnected rules](image-url)
Conversely, the improvements that succeeded in reducing the number of violations to zero violations are in the rule that has a relationship of both node and its rule for the same variable. For example, for variable B there are two rules that affect the value of the improvement.

\[ \varphi: \{A = 1\} \rightarrow \{B = 1\} \lor \{B = 2\}, \text{dom}(B) = \{1, 2, 3\} \]

\[ \varphi: \{C = 2\} \land \{D = 1\} \rightarrow \{B = 2\} \lor \{B = 3\}, \text{dom}(B) = \{1, 2, 3\} \]

Figure 8 gives an illustration of the structure.

![Diagram of connected rules](image)

**Figure 8.** Examples of connected rules

In terms of improvement iteration number made, to produce zero violation, in scenario 1 requires an average of 11-12 repairs on each iteration. The minimum iteration produced is 7 and the maximum iteration achieved is 29. With an increase in error rate, the iteration also increases. In the second scenario, the average requires 24-25 iterations, and in scenario 3 needs 34-35 iterations.

### 6. Conclusion

In this paper, we proposed a graph structure of data edits rules, namely directed bipartite hypergraph. The proposed representation can be used as a basis for constructing constraint-based data repair algorithms. Based on the experimental study, error reduction can be achieved by 100% in data with 10% error rate, 94% for 20% error rate, and 96% for 30% error rate. The success of resolving violations into zero violations decreases with increasing of the error rate in the data. Likewise with the many iterations in conducting violations. The next research will develop other graph parameters that can be used in determining repairing variables to provide more optimum improvement results.

### References

[1] Batini Carlo and Scannapieca M 1998 *Data Quality Concepts, Methodologies and Techniques* (Italy: Springer)

[2] Beskales G, Ilyas I F and Golab L 2010 Sampling the repairs of functional dependency violation under hard constraints *Proc. of the VLDB Endowment* 3(1-2) pp 197-207

[3] Chiang F and Miller R J 2011 A unified model for data and constraint repair *IEEE 27th Int. Conf. on Data Engineering* pp 446-457

[4] Diallo T, Petit J M and Servigne S 2012 Discovering editing rules for data cleaning *Proc. of Aqb Conf.* 40

[5] Fan W, Li J, Ma S, Tang N and Yu W 2012 Towards certain fixes with editing rules and master data *The VLDB J.* 21(2) pp 213-238

[6] Bertossi L, Bravo L, Franzoni E and Lopatenko A 2008 The complexity and approximation of fixing numerical attributes in databases under integrity constraint *Information Systems* 33(4-5) pp 407-434

[7] Cong G, Fan W, Geerts F, Jia X, and Ma S 2007 Improving data quality: consistency and
accuracy Proc. of the 33rd Int. Conf. on Very Large Databases pp 315-326

[8] Yakout M, Elmagarmid A, Neville J, Ouzzani M and Ilyas I F. 2011 Guided data repair Proc. of VLDB Endowment 4(5) pp 279-289

[9] Kolahi S and Lakshmanan L V S 2009 On approximating optimum repairs for functional dependency violations Proc. of the 12th Int. Conf. on Database Theory pp 53-62

[10] Chu X, Ilyas I F and Papotti P 2013 Holistic data cleaning: putting violations into context IEEE 29th Int. Conf. on Data Engineering pp 458-469

[11] Kjaerulff U B and Madsen A L 2013 Bayesian networks and influence diagrams: a guide to construction and analysis (London: Springer)

[12] Madjida W O Z and Nugraha I G B B 2018 Gradit: graph-based data repair algorithm for multiple data edits rule violations J. Phys. Conf. Series 971

[13] Medina R and Nourine L 2008 A unified hierarchy for functional dependencies, conditional functional dependencies, conditional functional dependencies and association rules Research Report LIMOS

[14] Fan W, Geerts F, Jia X and Kementsietsidis A 2008 Conditional functional dependencies for capturing data inconsistencies ACM Transaction on Database System 33(2) pp 6:1-6:48