# Rule-Based Method for Entity Resolution

Lingli Li, Jianzhong Li, and Hong Gao

**Abstract—**The objective of entity resolution (ER) is to identify records referring to the same real-world entity. Traditional ER approaches identify records based on pairwise similarity comparisons, which assumes that records referring to the same entity are more similar to each other than otherwise. However, this assumption does not always hold in practice and similarity comparisons do not work well when such assumption breaks. We propose a new class of rules which could describe the complex matching conditions between records and entities. Based on this class of rules, we present the rule-based entity resolution problem and develop an on-line approach for ER. In this framework, by applying rules to each record, we identify which entity the record refers to. Additionally, we propose an effective and efficient rule discovery algorithm. We experimentally evaluated our rule-based ER algorithm on real data sets. The experimental results show that both our rule discovery algorithm and rule-based ER algorithm can achieve high performance.

**Index Terms—**Entity resolution, rule learning, data cleaning

## 1 Introduction

In many applications, a real-world entity may appear in multiple data sources so that the entity may have quite different descriptions. For example, there are several ways to represent a person's name or a mailing address. Thus, it is necessary to identify the records referring to the same real-world entity, which is called Entity Resolution (ER). ER is one of the most important problems in data cleaning and arises in many applications such as information integration and information retrieval. Because of its importance, it has attracted much attention in the literature [27].

Traditional ER approaches obtain a result based on similarity comparison among records, assuming that records referring to the same entity are more similar to each other (*compact set property* [17]). However, such property may not hold so traditional ER approaches cannot identify records correctly in some cases. We use the following example to illustrate one of the cases.

**Example 1.** Table 1 shows seven authors with name "wei wang" identified by $o_{ij}$s. By accessing to the authors home pages containing their publications, we manually divide the seven authors into three clusters. The records with IDs $o_{11}$, $o_{12}$, and $o_{13}$ refer to the person in UNC, denoted as $e_1$, the records with IDs $o_{21}$ and $o_{22}$ refer to the person in UNSW, denoted as $e_2$, and the records with IDs $o_{31}$ and $o_{32}$ refer to the person in Fudan University, denoted as $e_3$. The task of entity resolution is to identify $e_1$, $e_2$ and $e_3$ using the information in Table 1. Since all these records have identical name but different set of coauthors in different papers, the similarity between any two records $X$ and $Y$, denoted by $Sim(X, Y)$, is determined by the similarity of `coauthors`. To measure the similarity between sets, Jaccard similarity [3] is often used, and thus the similarity between any two records, $X$ and $Y$, is defined as: $Sim(X,Y) = \frac{|\texttt{coauthors}(X) \cap \texttt{coauthors}(Y)|}{|\texttt{coauthors}(X) \cup \texttt{coauthors}(Y)|}$. Thus, we have the following facts:

- $Sim(o_{11}, o_{12}) < Sim(o_{11}, o_{31})$ since $Sim(o_{11}, o_{12}) = 0$ and $Sim(o_{11}, o_{31}) = \frac{1}{2}$, and
- $Sim(o_{12}, o_{13}) < Sim(o_{12}, o_{21})$ since $Sim(o_{12}, o_{13}) = \frac{1}{5}$ and $Sim(o_{12}, o_{21}) = \frac{1}{4}$.

The result shows that the similarity between $o_{11}$ and $o_{12}$ is smaller than the similarity between $o_{11}$ and $o_{31}$ even though $o_{11}$ and $o_{12}$ refer to the same entity while $o_{11}$ and $o_{31}$ refer to different entities. It is obvious that we are unable to get the correct ER result of the example by applying similarity comparison between records. Similar to Jaccard, other similarity functions, such as cosine similarity and TF-IDF, also have the same problem. As similarity comparisons can not be applied in this case, we have the following observations.

**Observation 1.** The existence of some attribute-value pairs are useful to identify records.

Take Table 1 as an example. The attribute-value pair (`coauthors`,"lin") occurs only in the records referring to $e_2$. Thus, the existence of (`coauthors`, "lin") can be used to identify records referring to $e_2$. Similarly, the existence of (`coauthors`, "kum") and (`coauthors`, "shi") can be used to identify records referring to $e_1$ and $e_3$ respectively.

**Observation 2.** The nonexistence of some attribute-value pairs are also useful to identify records.

Taking Table 1 as an example again, the `coauthors` of the record $o_{11}$ includes only "zhang". Since "zhang" occurs in both $o_{11}$ and $o_{31}$, the existence of (`coauthors`, "zhang") can distinguish the records referring to $e_1$ or $e_3$ from the other records, but cannot distinguish the records referring to $e_1$ and $e_3$. However, the nonexistence of (`coauthors`, "shi") can be used to rule out the possibility of $o_{11}$ referring to $e_3$ since the existence of (`coauthors`, "shi") can identify all the records referring to $e_3$. Thus, the existence of (`coauthors`, "zhang") and the nonexistence of (`coauthors`, "shi") can be used together to identify the records referring to $e_1$.

- *L. Li, J. Li and H. Gao are with the Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China. E-mail: lwsbrr@gmail.com, {lijzh, honggao}@hit.edu.cn.*

*Manuscript received 29 Jul. 2013; revised 16 Apr. 2014; accepted 19 Apr. 2014. Date of publication 28 Apr. 2014; date of current version 1 Dec. 2014. Recommended for acceptance by J. Pei.*
*For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.*
*Digital Object Identifier no. 10.1109/TKDE.2014.2320713*

TABLE 1
Paper-Author Records

| | id | name | coauthors | title |
|---|---|---|---|---|
| $e_1$ | $o_{11}$ | wei wang | zhang | inferring... |
| | $o_{12}$ | wei wang | duncan, kum, pei | social... |
| | $o_{13}$ | wei wang | cheng, li, kum | measuring... |
| $e_2$ | $o_{21}$ | wei wang | lin, pei | threshold... |
| | $o_{22}$ | wei wang | lin, hua, pei | ranking... |
| $e_3$ | $o_{31}$ | wei wang | shi, zhang | picturebook... |
| | $o_{32}$ | wei wang | pei, shi, xu | utility... |

Based on the observations, we are able to develop the following rules to identify records in Table 1.

- $R_1$: $\forall o_i$, if $o_i$[name] is "wei wang" and $o_i$[coauthors] includes "kum", then $o_i$ refers to entity $e_1$;
- $R_2$: $\forall o_i$, if $o_i$[name] is "wei wang" and $o_i$[coauthors] includes "lin", then $o_i$ refers to entity $e_2$;
- $R_3$: $\forall o_i$, if $o_i$[name] is "wei wang" and $o_i$[coauthors] includes "shi", then $o_i$ refers to entity $e_3$;
- $R_4$: $\forall o_i$, if $o_i$[name] is "wei wang" and $o_i$[coauthors] includes "zhang" and excludes "shi", then $o_i$ refers to entity $e_1$.

This example shows that the disadvantages of the traditional ER methods can be overcome by employing rules generated from the entities' information. This findings motivate us to develop a rule-based entity resolution method. This gives rise to the following challenges.

1) How to define the rules as described in Example 1 and how to define the properties to be satisfied by the rules to ensure a good performance of entity resolution both in efficiency and effectiveness?
2) How to discover the rules from a given training data set to support ER efficiently and effectively?

   Note that, to discover rules to resolve entities, the information of real-world entities are required. In practice, such information can be collected from many different sources. For example, the information of clients can be obtained from the master data maintained by companies, the information, such as skills, education, etc., of professional persons can be collected from LinkedIn,[1] and the information of researchers and scientists can be collected from ResearchGate.[2] In this paper, we assume that the information of entities are already collected.
3) How to use the rules to identify records efficiently and effectively?
4) How to maintain the rules when entity information is changed?

This paper aims at the aforementioned problems, and the main contributions of the paper are as following.

1) The syntax and semantics of the rules for ER are designed, and the independence, consistency, completeness and validity of the rules are defined and analyzed.
2) An efficient rule discovery algorithm based on training data is proposed and analyzed.
3) An efficient rule-based algorithm for solving entity resolution problem is proposed and analyzed.
4) A rule maintaining method is proposed when entity information is changed.
5) Experiments are performed on real data to verify the effectiveness and efficiency of the proposed algorithms.

In fact, our method and traditional ER approaches can be considered as the complementary to each other and be applied together. This is because our rule-based method can identify records which cannot be resolved by traditional ER methods and traditional ER methods can identify most of the records effectively and do not require the availability of correct entity set. In this way, the limitations of both methods can be overcome.

## 2 Rules for Entity Resolution

In this section, a rule system for entity resolution, called ER-rule, is defined. We can see that each rule in Example 1 consists of two clauses. (1) The *If* clause includes constraints on attributes of records, such as "including *zhang* in *coauthors*", and (2) the *Then* clause indicates the real world entity referred by the records that satisfy the first clause of the rule, such as "refers to entity $e_1$". Thus, we use $A \Rightarrow B$ to express the rules "$\forall o$, *If* $o$ satisfies A *Then* $o$ refers to B" for ER. We denote the left-hand side and the right-hand side of a rule $r$ as $\mathsf{LHS}(r)$ and $\mathsf{RHS}(r)$ respectively.

### 2.1 Syntax

An ER-rule is syntactically defined as $T_1 \wedge \cdots \wedge T_m \Rightarrow e$, where $T_i (1 \leq i \leq m)$ is a clause with the form of $(A_i\ op_i\ v_i)$, $(v_i\ op_i\ A_i)$, $\neg(A_i op_i\ v_i)$ or $\neg(v_i\ op_i\ A_i)$, where $A_i$ is an attribute, $v_i$ is a constant in the domain of $A_i$ and $op_i$ can be any domain-dependent operator defined by users, such as exact match operator $=$, fuzzy match operator $\approx$ [16] for string value, $\leq$ for numeric value, or $\in$ for set value. The clause with form $(A_i\ op_i\ v_i)$ or $(v_i\ op_i\ A_i)$ is called *positive clause*, and the clause with form $\neg(A_i\ op_i\ v_i)$ or $\neg(v_i\ op_i\ A_i)$ is called *negative clause*.

Each ER-rule $r$ can be assigned a weight $w(r)$ in $[0, 1]$ to reflect the level of confidence that $r$ is correct. Intuitively, the more records are identified by an ER-rule $r$, the more possible $r$ is correct. Therefore, given a data set $S$, we define the weight of each ER-rule $r$ as:

$$w(r) = \frac{|S(r)|}{|S(\mathsf{RHS}(r))|},$$

where $S(r)$ denotes the records in $S$ that are identified by $r$ and $S(\mathsf{RHS}(r))$ denotes the records in $S$ that refer to entity $\mathsf{RHS}(r)$.

In the rest of the paper, we assume that the operator $op_i$ for each attribute $A_i$ is given. In this way, a positive clause $(A_i\ op_i\ v_i)$ can be abbreviated as an attribute-value pair $(A_i, v_i)$. Similarly, a negative clause $\neg(A_i\ op_i\ v_i)$ can be abbreviated as $\neg(A_i, v_i)$.

**Example 2.** The rules given in Example 1 can be expressed as the following ER-rules respectively. For simplicity we write coa rather than coauthors.

$r_1$: (name ="wei wang")$\wedge$("kum"$\in$ coa) $\Rightarrow e_1$,
$r_2$: (name ="wei wang")$\wedge$("lin"$\in$ coa) $\Rightarrow e_2$,

1. http://www.linkedin.com/.
2. https://www.researchgate.net/.

$r_3$: (name ="wei wang")$\wedge$("shi"$\in$ coa) $\Rightarrow e_3$,
$r_4$: (name ="wei wang")$\wedge$("zhang"$\in$ coa) $\wedge$
$\neg$("shi"$\in$ coa)$\Rightarrow e_1$,

where $x \in Y$ represents that attribute $Y$ includes value $x$. For $r_4$, (name = "wei wang"), ("zhang"$\in$ coa) are positive clauses and $\neg$("shi" $\in$ coa) is a negative clause, where name, coa are attributes, $=, \in$ are operators and "wei wang", "zhang", "shi" are values. $r_4$ can also be simplified as (name,"wei wang")$\wedge$(coa,"zhang")$\wedge\neg$(coa,"shi")$\Rightarrow e_1$.

## 2.2 Semantics

In the following definitions, we let $o$ be a record, $S$ be a data set, $r$ be an ER-rule and $R$ be an ER-rule set. For the convenience of discussion, we assume the mapping from each record in $S$ to its actual entity is given.

Since an ER-rule does not include disjoint clauses, we define the condition of matching the left-hand and right-hand sides of rule as follows.

**Definition 1.** $o$ matches *the* LHS *of* $r$ *if* $o$ *satisfies all the clauses in* $\mathsf{LHS}(r)$. $o$ matches *the* RHS *of* $r$ *if* $o$ *refers to entity* $\mathsf{RHS}(r)$.

**Definition 2.** $o$ satisfies $r$, *denoted by* $o \vdash r$, *if* $o$ *does not match* $\mathsf{LHS}(r)$ *or matches* $\mathsf{RHS}(r)$.

In another word, record $o$ does not satisfy ER-rule $r$ if and only if $o$ matches $\mathsf{LHS}(r)$ and does not match $\mathsf{RHS}(r)$. This definition is based on that "if $A$ then $B$" is equivalent to "$\neg A \vee B$" in first order logic system.

**Example 3.** Consider records in Table 1 and ER-rule $r_1$ in Example 2. For $o_{12}$ and $o_{13}$, they satisfy $r_1$ because they both match $\mathsf{LHS}(r_1)$ and $\mathsf{RHS}(r_1)$. For the other records in Table 1, they also satisfy $r_1$ because none of them matches $\mathsf{LHS}(r_1)$. Therefore $r_1$ is satisfied by all the records in Table 1.

Definition 1 and 2 presents the semantics of an ER-rule, that is, if a record $o$ makes $\mathsf{LHS}(r)$ true then $o$ must refer to the entity denoted by $\mathsf{RHS}(r)$.

Moreover, Definition 2 ($\vdash$) can be extended as follows.

- $S \vdash r$ if $o \vdash r$ for $\forall o \in S$, and we say $r$ is a *valid* rule for data set $S$ if $S \vdash r$.
- $o \vdash R$ if $o \vdash r$ for $\forall r \in R$.

Based on the above definitions, we say $R$ *entails* $r$, denoted by $R \models r$, if $o \vdash R$ then $o \vdash r$ for any possible record $o$.

**Definition 3.** $o$ *is* identified *by* $r$, *if* $o$ *matches both* $\mathsf{LHS}(r)$ *and* $\mathsf{RHS}(r)$. *Note that, if* $o$ *is identified by* $r$, $o$ *must satisfy* $r$. *If* $o$ *satisfies* $r$, $o$ *might not be identified by* $r$.

**Example 4.** Consider $o_{11}$ in Table 1 and the ER-rule set $R = \{r_1, r_2, r_3, r_4\}$ in Example 2. $o_{11}$ matches $\mathsf{LHS}(r_4)$ because $o_{11}$[name] = "wei wang", "zhang"$\in o_{11}$[coa] and "shi"$=\notin o_{11}$[coa]. $o_{11} \vdash r_4$ and $o_{11}$ is identified by $r_4$ because $o_{11}$ matches $\mathsf{LHS}(r_4)$ and $\mathsf{RHS}(r_4)$. $o_{11} \vdash r_1$ but $o_{11}$ is not identified by $r_1$ because $o_{11}$ does not match $\mathsf{LHS}(r_1)$. $o_{11} \vdash R$ because $o_{11} \vdash r$ for $\forall r \in R$.

## 2.3 Properties of ER-Rule Set

Given an ER-rule set $R$ and a data set $S$, to ensure $R$ performs well on $S$, we require that (1) there is no false matches between record and entity (validity); (2) there is no conflicting decisions by $R$ (consistency); (3) each record in $S$ can be mapped to an entity by $R$ (completeness) and (4) there is no redundant rules in $R$ (independence). Now we present the formal definitions of these properties.

**Definition 4 (Validity).** $R$ *is valid for* $S$ *if each ER-rule in* $R$ *is valid for* $S$.

**Definition 5 (Consistency).** $R$ *is consistent for* $S$ *if* $o$ *matches both* $\mathsf{LHS}(r_1)$ *and* $\mathsf{LHS}(r_2)$ *then* $\mathsf{RHS}(r_1) = \mathsf{RHS}(r_2)$ *for* $\forall o \in S$ *and* $\forall r_1, r_2 \in R$.

**Definition 6 (Completeness).** $R$ *is complete for* $S$ *if for* $\forall o \in S$, $\exists r \in R$ *such that* $o$ *matches* $\mathsf{LHS}(r)$.

**Definition 7 (Independence).** $R$ *is independent if each ER-rule* $r$ *in* $R$ *satisfies that* $R - \{r\}$ *does not entail* $r$, *denoted as* $R - \{r\} \not\models r$.

The following proposition shows that an ER-rule set will contain redundant rules if it is not independent.

**Proposition 1.** *If both* $r$ *and* $R$ *are valid for* $S$ *and* $R \models r$, *then for any record* $o$ *in* $S$, *if* $o$ *is identified by* $r$, $o$ *must be identified by* $R$.

**Proof.** We prove that if there exists one record $o$ in $S$ such that $o$ is identified by $r$ but $o$ is not identified by $R$, then there exists no rule $r'$ in $R$ such that $o$ matches $\mathsf{LHS}(r')$. Otherwise, let $o$ be a record which is not identified by $R$ and there exists one rule $r'$ in $R$ such that $o$ matches $\mathsf{LHS}(r')$. Since $o$ is not identified by $R$, $o$ does not match $\mathsf{RHS}(r')$. Then $o \not\vdash r'$. Thus $R$ is not valid for $S$, which contradicts to our assumption. Therefore the conclusion that if there exists one record $o$ in $S$ such that $o$ is identified by $r$ but $o$ is not identified by $R$, then there exists no rule $r'$ in $R$ such that $o$ matches $\mathsf{LHS}(r')$ is true.

Now we prove that if there exists one record $o$ in $S$ such that $o$ is identified by $r$ but $o$ is not identified by $R$, then there exists a record $o'$ such that $o' \vdash R$ but $o' \not\vdash r$. We construct the record $o'$ as follows: let $o' = o$ and $o'$ does not refer to $\mathsf{RHS}(r)$. Since $o' = o$ and there exists no rule $r'$ in $R$ such that $o$ matches $\mathsf{LHS}(r')$, $o' \vdash R$. Thus $R \not\models r$ because $o' \vdash R$ and $o' \not\vdash r$. This is in contradiction with our assumption that $R \models r$. Therefore, the clause is true. □

**Proposition 2.** *If* $R$ *is valid for* $S$, $R$ *is consistent for* $S$.

**Proof.** Suppose $R$ is valid but not consistent for $S$. There must exist a record $o \in S$ and ER-rules $r_1, r_2 \in R$ such that $o$ matches $\mathsf{LHS}(r_1)$ and $\mathsf{LHS}(r_2)$, and $\mathsf{RHS}(r_1) \neq \mathsf{RHS}(r_2)$. Since $o$ matches both $\mathsf{LHS}(r_1)$ and $\mathsf{LHS}(r_2)$ and $R$ is valid, $o$ should match both $\mathsf{RHS}(r_1)$ and $\mathsf{RHS}(r_2)$. Therefore, $\mathsf{RHS}(r_1) = \mathsf{RHS}(r_2)$, which is in contradiction with the assumption that $R$ is valid but not consistent for $S$. Thus, the proposition is true. □

Theorem 1 shows the expressive power of ER-rules.

**Theorem 1.** *Given a data set* $S = \{o_1, \ldots, o_n\}$, *there is at least one ER-rule set* $R$ *that is independent, consistent, valid and complete for* $S$.

**Proof.** For each record $o_i$ in $S$, let $e_i$ denote the entity to which $o_i$ refers, $U$ denote the set of attribute-value pairs that occur in records in $S$. We construct an ER-rule set $R = \{r_1, \ldots, r_n\}$ as follows. For each rule $r_i$ in $R$, $\mathsf{RHS}(r_i) = e_i$, and for $\forall (A_j, v_j) \in U$, if $(A_j, v_j) \in o_i$ then

$(v_j \in A_j)$ is in $\mathsf{LHS}(r_i)$, otherwise $\neg(v_j \in A_j)$ is in $\mathsf{LHS}(r_i)$. Note that, the equality operator(=) can be considered as a special case of operator $\approx$ and $\in$.

Hence, $R$ has the following properties. For $1 \leq i, j \leq n$,

(1) if $i = j$, $o_i$ matches $\mathsf{LHS}(r_j)$ and $\mathsf{RHS}(r_j)$;
(2) if $i \neq j$, $o_i$ does not match $\mathsf{LHS}(r_j)$.

According to property (1), $R$ is complete. According to the properties (1) and (2), $R$ is valid. According to the Proposition 2, $R$ is consistent.

To prove $R$ is independent, we prove that $R - \{r_i\} \not\models r_i$ for each ER-rule $r_i \in R$. For each ER-rule $r_i \in R$, we construct a record $o_i'$ as follows: $o_i' = o_i$ but $o_i'$ does not refer to $\mathsf{RHS}(r_i)$. Since $o_i'$ matches $\mathsf{LHS}(r_i)$ and $o_i$ does not match $\mathsf{RHS}(r_i)$, then $o_i' \not\vdash r_i$. Moreover, since $o_i'$ does not matches $\mathsf{LHS}(r_j)$ for any ER-rule $r_j \in R - \{r_i\}$, then $o_i' \vdash R - \{r_i\}$. Therefore $R - \{r_i\} \not\models r_i$ for each ER-rule $r_i \in R$. Hence $R$ is independent.

Therefore, given a data set $S$, there exists at least one rule set $R$ that is independent, consistent, valid and complete for $S$. □

## 3 Rule Discovery

Since it might be too expensive to construct ER-rules manually, we discuss how to discover useful rules from a training data set for efficient and effective entity resolution in this section. We assume that the operator on each attribute can be any domain-dependent operator defined by users.

First, we discuss the requirements of the discovered rule sets and present our framework of rule discovery (Section 3.1). Then we describe the algorithms in the rule discovery framework (Sections 3.2 and 3.3) and study the correctness and complexity of our algorithm (Section 3.4). For exposition, proofs have been deferred to the Appendix.

For the convenience of discussion, some concepts are introduced first.

We classify ER-rules into two categories according to whether negative clauses are included.

**Definition 8 (PR).** PR *is an ER-rule which only includes positive clauses.*

**Definition 9 (NR).** NR *is an ER-rule which includes at least one negative clause.*

For example, $r_1$, $r_2$ and $r_3$ in Example 2 are all PRs while $r_4$ is an NR.

**Definition 10 (coverage).** *The* coverage of clause $T$ *on* $S$, *denoted as* $Cov_S(T)$, *is the subset of* $S$ *such that* $Cov_S(T) = \{o | o \in S, o \text{ satisfies } T\}$.

*Accordingly, the* coverage of rule $r$ *on* $S$, *denoted as* $Cov_S(r)$, *is the intersection of the coverage of each clause in* $r$, *such that* $Cov_S(r) = Cov_S(T_1) \cap Cov_S(T_2) \cap \cdots \cap Cov_S(T_k)$, *where* $\mathsf{LHS}(r) = T_1 \wedge T_2 \cdots \wedge T_k$. *Clearly,* $Cov_S(r) = \{o | o \in S, o \text{ matches } \mathsf{LHS}(r)\}$.

The *coverage of rule set* $R$ on $S$, denoted as $Cov_S(R)$, is the union of the coverage of each rule $r \in R$ on $S$, such that $Cov_S(R) = \cup_{r \in R} Cov(r)$.

Note that, when $S$ is clear from the context, $Cov_S()$ is simplified as $Cov()$.

**Example 5.** Let $T =$ ("zhang" $\in$ coa), $S$ be the set of records in Table 1, we have $Cov_S(T) = \{o_{11}, o_{31}\}$. Consider rules in Example 2, we have $Cov(r_1) = \{o_{12}, o_{13}\}$, $Cov(r_2) = \{o_{21}, o_{22}\}$, $Cov(r_3) = \{o_{31}, o_{32}\}$, $Cov(r_4) = \{o_{11}\}$ and $Cov_S(R) = S$ where $R = \{r_1, r_2, r_3, r_4\}$.

**Proposition 3.** *If* $\mathsf{RHS}(r) = e_j$, *then* $r$ *is valid for* $S$ *iff each record in* $Cov_S(r)$ *refers to* $e_j$.

Clearly, if rule $r$ is valid for data set $S$, $Cov_S(r)$ is the set of records which are identified by $r$. Accordingly, if rule set $R$ is valid for $S$, $Cov_S(R)$ is the set of records in $S$ that are identified by rules in $R$.

### 3.1 Rule Discovery Problem

In this section, we define the problem of rule discovery. Let $\mathbb{S} = \{S_1, \ldots, S_m\}$ be the training data of data set $S$, where each $S_j$ in $\mathbb{S}$ is a subset of $S$ which refer to the same entity, denoted by $e_j$. As discussed, the discovered rule set should be independent, valid and complete for $S$ to ensure a good performance of ER on $S$. Thus these three properties should be taken into consideration for the rule discovery problem.

#### 3.1.1 Requirements

Even though these properties are satisfied on the training data set, it cannot be ensured that the generated rule set can also perform well on the other data sets. To make the rule set suitable for ER for many data sets other than only suitable for the training data set, we require the discovered ER-rule set, denoted by $R$, should also satisfy two requirements described as follows.

• *Length Requirement: Given a threshold* $l$, *each rule* $r$ *in* $R$ *satisfies that* $|r| \leq l$.

To determine whether record $o$ matches the LHS of ER-rule $r$, we should check whether $o$ satisfies each clause in $\mathsf{LHS}(r)$. Thus to guarantee the efficiency of rule-based ER (R-ER) and avoid overfitting, the length of each rule (the number of clauses) should be no more than a threshold.

• *PR Requirement: each rule* $r$ *in* $R$ *is a* PR.

The reason why we give priority to PRs is that, positive literals lead to bounded spaces while negative literals lead to unbounded spaces. Therefore the discovered PRs are more possible to identify other data sets effectively than the discovered NRs.

However, both the length requirement and the PR requirement can decrease the expression power of ER-rules, so that there might be some records in $S$ that cannot be identified by any valid PR with length no more than $l$. To guarantee the completeness, the requirements should be relaxed. Specifically, for each record $o$ in $S$ that cannot be identified by any valid PR with length no more than $l$, a valid PR with the smallest length that identifies $o$ should be discovered; if no valid PR can identify $o$, a valid NR with the smallest length that identifies $o$ should be discovered.

The following propositions(see Appendix for proofs) prove the hardness of our problem. Without loss of generality, it is assumed that there exists no primary key in the data set.

**Proposition 4.** *Given a length-threshold* $l$ *and a data set* $S$, *for any record* $o$ *in* $S$, *determining whether there exists a valid* PR

*$r$ such that $o \in Cov(r)$ and $|r| \leq l$(we call this problem "LPR" for brief) is NP-Complete.*

**Proposition 5.** *Given a data set $S$, for any record $o$ in $S$, finding a valid* PR *$r$ with the smallest length that identifies $o$ (we call this problem "SPR" for brief) is NP-Hard.*

This proposition can be easily proved from Proposition 4.

**Proposition 6.** *Given a data set $S$, for any record $o$ in $S$, finding a valid* NR *$r$ with the smallest length that identifies $o$ (we call this problem "SNR" for brief) is NP-Hard.*

#### 3.1.2 Framework

Given a record $o$ in $S$, since finding a valid PR (NR) with the smallest length that identifies $o$ is NP-hard, to make the problem tractable, we relax the requirements to find a minimal valid PR (NR) that identifies $o$.

**Definition 11 (sub-rule).** *ER-rule $r_1$ is a sub-rule of ER-rule $r_2$ if $\mathsf{RHS}(r_1) = \mathsf{RHS}(r_2)$ and the set of the clauses in $\mathsf{LHS}(r_1)$ is a proper subset of the clauses in $\mathsf{LHS}(r_2)$.*

**Definition 12 (minimal rule).** *ER-rule $r$ is a minimal rule for data set $S$, if $Cov(r) \backslash S_{e_r} \subset Cov(r') \backslash S_{e_r}$ for any sub-rule $r'$ of $r$, where $S(e_r) = \{o | o \in S$ and $o$ refers to entity $\mathsf{RHS}(r)\}$.*

**Example 6.** Consider the following rules. As can be seen, $r_2$ is a sub-rule of $r_1$. $r_1$ is not a minimal rule because $Cov(r_1) \backslash S_1 = Cov(r_2) \backslash S_1 = \emptyset$. In contrast, $r_2$ is a minimal rule.
$r_1$: (name,"wei wang")$\wedge$(coa,"zhang")$\wedge\neg$(coa,"shi") $\Rightarrow e_1$,
$r_2$: (coa,"zhang")$\wedge\neg$(coa,"shi")$\Rightarrow e_1$.

Clearly, $r$ is a *minimal valid rule* for $S$, if $r$ is valid for $S$ and $r'$ is not valid for $S$ for any sub-rule $r'$ of $r$.

Now we present the framework of ER-rule set discovery (shown in Algorithm 1), denoted by DISCR.

**Algorithm 1** Framework of Rule Discovery(DISCR)

**Input:** length-threshold $l$, training data $\mathbb{S} = \{S_1, \cdots, S_m\}$
**Output:** ER-rule set $R$

```
1:  S ← S_1 ∪ ··· ∪ S_m;
2:  R ←GEN-PR(l, 𝕊);
3:  if Cov(R) ≠ S then
4:      S' ← S\Cov(R);
5:      for each o in S' do
6:          if r_o is valid then
7:              R.insert(MIN-RULE(r_o));
8:          else
9:              R.insert(GEN-SINGLENR(o));
10:         end if
11:     end for
12: end if
13: R_min ←GREEDY-SETCOVER(R, S);
14: Return R;
```

Given a training data set $\mathbb{S}$ and a length-threshold $l$, the valid PRs with length no more than $l$ are generated at first by GEN-PR (line 2), denoted by $R$.

If $R$ is not complete for $S$ (line 3), for each record $o$ that is not covered by $R$, if the corresponding rule $r_o$: $\wedge_{t_i \in o} t_i \Rightarrow e_o$ is valid, a minimal valid sub-rule of $r_o$ is generated by MIN-RULE and is added into $R$ (line 7); otherwise, according to Proposition 7 there exists no valid PR that identifies $o$, then a minimal valid NR to identify $o$ is generated by GEN-SINGLENR and added into $R$ (line 9).

Finally, to ensure the independence (see Proposition 8) and reduce the number of rules to accelerate ER, after a valid and complete ER-rule set $R$ is generated, a minimal subset $R_{min}$ of $R$ is generated by applying the greedy algorithm GREEDY-SETCOVER [34] for the set covering problem, where $S$ is the universe to be covered and $R$ is the collection of subsets of $S$ in which each rule $r$ in $R$ is mapped to the subset $Cov(r)$ (line 10). During the greedy selection step, for two rules $r_1$ and $r_2$ with the same incremental gain, we choose the rules based on the following priority: if $r_1$ is PR and $r_2$ is NR, $r_1$ is selected; or if $r_1$ and $r_2$ are both PRs (NRs), but $|r_1| \leq |r_2|$, $r_1$ is selected.

TABLE 2
Data Structure of ER-Rules

| rid | eid | exp | coverage |
|---|---|---|---|
| $r_1$ | $e_1$ | $t_1$ | $H(r_1, e_1)$,$H(r_1, e_2)$ |
| $r_2$ | $e_2$ | $t_2 \wedge t_3$ | $H(r_2, e_2)$ |

**Proposition 7.** *Given a data set $S$, for any record $o$ in $S$, there exists a valid* PR *$r$ that identifies $o$ iff $r_o$: $\wedge_{t_i \in o} t_i \Rightarrow e_o$ is valid.*

**Proposition 8.** *If $R$ is minimal for data set $S$, then $R$ is independent.*

To support the algorithm, we use (rid, eid, exp, coverage) to describe each rule $r$, where rid is the id of $r$, eid is $\mathsf{RHS}(r)$, exp is $\mathsf{LHS}(r)$, and coverage is the sets of records covered by $r$. A hash table $H$ is maintained to store the information of coverage. Specifically, given $(r_i, e_j)$ as the key, $H(r_i, e_j) = Cov(r_i) \cap S_j$. For instance, in Table 2, $H(r_1, e_2)$ returns the records in $S_2$ which match $\mathsf{LHS}(r_1)$. Each time a new rule is generated, the hash table is used for checking its validity.

### 3.2 Gen-PR

We first define the following concepts and propositions for the convenience of discussion.

**Definition 13 (preliminary rule).** *$r$ is called a preliminary rule for $S$, if $r$ is not valid for $S$ but $r$ identifies at least one record in $S$.*

According to Proposition 3, if $\mathsf{RHS}(r) = e_j$, $r$ is a preliminary rule for $S$ iff $Cov(r) \nsubseteq S_j$ and $Cov(r) \cap S_j \neq \emptyset$.

Since a valid ER-rule with empty coverage does not need to be discovered, we require that the coverage of each valid ER-rule is not empty.

**Proposition 9.** *If $r$ is a minimal valid rule for $S$ and $|r| > 1$, then any sub-rule of $r$ must be a preliminary rule of $S$.*

**Definition 14 (conjunction).** *Let $r_1, r_2$ be two ER-rules in which $\mathsf{RHS}(r_1) = \mathsf{RHS}(r_2)$. The conjunction of $r_1$ and $r_2$ is an ER-rule $\mathsf{LHS}(r_1) \wedge \mathsf{LHS}(r_2) \Rightarrow \mathsf{RHS}(r_1)$, denoted as $r_1 \wedge r_2$.*

For simplicity, $r$ is called an "atomic" rule if its length equals to 1. $r$ is called a rule of entity $e_j$, if $\mathsf{RHS}(r) = e_j$ and $R$ is called a rule set of entity $e_j$, if each rule in $R$ is a rule of $e_j$.

Since minimal valid rules can be generated by conjunctions of preliminary rules according to Proposition 9, Gen-PR first generates all the atomic PRs and then iteratively generates PRs with length of $k+1$ by conjunctions of preliminary PRs of length $k$ and preliminary atomic PRs.

In our algorithm (shown in Algorithm 2), for each entity $e_j$, the generated preliminary PRs of $e_j$ are stored in $L_j^y$, the preliminary atomic PRs of $e_j$ are stored in $L_j^x$, and valid PRs of $e_j$ are stored in $R_j$.

**Algorithm 2** GEN-PR

**Input:** $l$, $\mathbb{S}$

1: **for** each $S_j$ in $\mathbb{S}$ **do**
2: **for** each attribute-value pair $t$ in $S_j$ **do**
3: **if** $Cov(t) \subseteq S_j$ **then**
4: $R_j$.insert($r(t)$);
5: **else**
6: $L_j^x$.insert($r(t)$);
7: $L_j^y$.insert($r(t)$);
8: **end if**
9: **end for**
10: **for** each $r_i \in L_j^y$ **do**
11: **if** $|r_i| > l$ **then**
12: break;
13: **end if**
14: **for** each $r_k \in L_j^x$ **do**
15: **if** $Cov(r_i \wedge r_k) \neq \emptyset$ **then**
16: **if** $Cov(r_i \wedge r_k) \subseteq S_j$ **then**
17: $R_j.insert(r_i \wedge r_k)$;
18: **else**
19: $L_j^y.insert(r_i \wedge r_k)$;
20: **end if**
21: **end if**
22: **end for**
23: **end for**
24: **end for**
25: $R = R_1 \cup R_2 \cup \cdots \cup R_{|\mathbb{S}|}$
26: Return $R$;

*Step 1. Generate atomic* PRs *(lines 2-9).* To find all the preliminary and valid atomic PRs of $e_j$, for each attribute-value pair $t$ that appears in the records in $S_j$, we should check whether the corresponding rule $r(t)$: $t \Rightarrow e_j$ is preliminary or valid. Specifically, if $r(t) \subseteq S_j$, then $r(t)$ is valid (according to Proposition 3) and is added to $R_j$ (lines 3-4); otherwise $r(t)$ is a preliminary and $r(t)$ is added to both $L_j^x$ and $L_j^y$ for further conjunction (lines 5-7).

*Step 2. Generate* PRs *with length* $>1$ *(lines 10-23).* In this step, we conjunct each preliminary rule $r_i$ in $L_j^x$ (line 8) with each preliminary rule $r_k$ in $L_j^y$ (line 9) to generate a new PR, $r_i \wedge r_k$. This new rule is added to $R_j$ if it is valid and its coverage is not empty (lines 10-12); or added to $L_j^y$ for further conjunctions if it is preliminary (lines 13-14); otherwise it is useless and can be ignored.

In contrast to traditional rule generation method, such as FOIL [42], where each rule is generated by iteratively adding the best literal into the rule until the rule becomes valid, we generate each rule by enumerating all possibilities. The reason why we do not use the greedy strategy similar as FOIL is that our problem does not satisfy the greedy selection property. That is, the best literal does not have to be included in the optimal result.

Now we use an example to illustrate the process of Gen-PR.

**Example 7.** Consider $o_{11}$-$o_{32}$ in Table 1. Let the operator for coa be $\in$. Suppose "pei" is also in $o_1$[coa].

TABLE 3
An Example of GEN-PR

**clusters of records**

| | |
|---|---|
| $S_1$ | $o_{11}$, $o_{12}$, $o_{13}$ |
| $S_2$ | $o_{21}$, $o_{22}$ |
| $S_3$ | $o_{31}$, $o_{32}$ |

**steps of GEN-PR**

| | $L_1^x$ | $L_1^y$ | $R_1$ |
|---|---|---|---|
| Step 1: | $r_1$, $r_2$ | $r_1$, $r_2$ | $r_3$, $r_4$ |
| Step 2: | $r_1$, $r_2$ | $r_1$, $r_2$ | $r_3$, $r_4$, $r_5$ |

PRs

| ER-rule | $Cov_{S_1}(r_j)$ | $Cov_{S_2}(r_j)$ | $Cov_{S_3}(r_j)$ |
|---|---|---|---|
| $r_1$: (coa,"zhang")$\Rightarrow e_1$ | $o_{11}$ | | $o_{31}$ |
| $r_2$: (coa,"pei")$\Rightarrow e_1$ | $o_{11}, o_{12}$ | $o_{21}, o_{22}$ | $o_{32}$ |
| $r_3$: (coa,"kum")$\Rightarrow e_1$ | $o_{12}, o_{13}$ | | |
| $r_4$: (coa,"duncan")$\Rightarrow e_1$ | $o_{12}$ | | |
| $r_5$: $r_1 \wedge r_2$ | $o_{11}$ | | |

First, we generate the valid PR set $R_1$ to identify records referring to $e_1$. The steps of generating $R_1$ are shown in Table 3. In Step 1, atomic PRs ($r_1$, $r_2$, $r_3$ and $r_4$) are generated. Among these rules, $r_3$, $r_4$ are stored in $R_1$ since they are valid, and $r_1$, $r_2$ are stored in both $L_1^x$ and $L_1^y$ since they are preliminary. In Step 2, we conjunct rules in $L_1^y$ with rules in $L_1^x$ ($L_1^x = L_1^y = \{r_1, r_2\}$). Then a new rule, $r_1 \wedge r_2$, is generated. $r_1 \wedge r_2$ is verified to be valid and is inserted into $R_1$. In this way, the PR set $R_1$ of $e_1$ is generated. Similarly, $R_2$ and $R_3$ are generated. Finally, the algorithm outputs the result, $R_1 \cup R_2 \cup R_3$.

The following Lemma shows the correctness of Algorithm 2.

**Lemma 1.** *Given a length-threshold $l$ and a training data set $\mathbb{S}$, the result $R$ output by Gen-PR is a valid* PR *set in which the length of each rule is no more than $l$.*

### 3.3 Gen-SingleNR

GEN-SINGLENR composes two steps. Given a record $o$, a valid NR $r$ that identifies $o$ is generated at first by SEL-CLAUSES; then a minimal sub-rule of $r$ is output by MIN-RULE.

SEL-CLAUSES. To find a valid rule $r$ that identifies $o$, SEL-CLAUSES works as follows. First $r$ is initialized as $r_o$ ($r_o$: $\wedge_{t_i \in o} t_i \Rightarrow e_o$), then for each record $o_i$ in $Cov(r_o) \backslash S_o$ ($S_o$ denotes the cluster in $\mathbb{S}$ that includes $o$), a negative clause $T_i$ which is satisfied by $o$ and not satisfied by $o_i$ is found and added into LHS($r$). This step is shown in Algorithm 3.

**Algorithm 3** Sel-Clauses

**Input:** record $o$

1: $T \leftarrow o$;
2: **for** each $o_i$ in $Cov(r_o) \backslash S_o$ **do**
3: **for** each $t_j$ in $o_i$ **do**
4: **if** $o$ satisfies $\neg t_j$ **then**
5: $T$.insert($\neg t_j$);
6: **end if**
7: **end for**
8: **end for**
9: Return $\wedge_{T_j \in T} T_j \Rightarrow e_o$;

Proposition 10 shows the correctness of Algorithm 3 (proof is in Appendix).

TABLE 4
An Example of GEN-SINGLENR

| record | entity |
|---|---|
| $o_{11} = \{t_1, t_2\}$ | $e_1$ |
| $o_{21} = \{t_1, t_2, t_3\}$ | $e_2$ |
| $o_{22} = \{t_1, t_2, t_4\}$ | $e_2$ |

**Proposition 10.** *Given a record $o$, SEL-CLAUSES outputs a valid ER-rule $r$ that identifies $o$.*

MIN-RULE. Given an ER-rule $r$, MIN-RULE (shown in Algorithm 4) checks each clause $T_i$ in $r$ (line 1) to identify whether $T_i$ can be removed (line 3). If so, $T_i$ is removed from $r$ (line 4).

**Algorithm 4** Min-Rule

**Input:** ER-rule $r : T_1 \wedge T_2 ... \wedge T_k \Rightarrow e_j$
1: **for** each clause $T_i$ in $r$ **do**
2: let $r'$ be the sub-rule of $r$ that excludes $T_i$;
3: **if** $Cov(r) \backslash S_j = Cov(r') \backslash S_j$ **then**
4: $r \leftarrow r'$;
5: **end if**
6: **end for**
7: Return $r$;

Proposition 11 shows the correctness of Algorithm 4 (proof is in Appendix).

**Proposition 11.** *Given a valid ER-rule $r$, MIN-RULE outputs a minimal sub-rule $r'$ of $r$.*

The following lemma can be easily proved from Proposition 10 and Proposition 11.

**Lemma 2.** *Given a record $o$, the result $r$ output by GEN-SINGLENR is a minimal valid* NR *that identifies $o$.*

We use an example to illustrate the whole process of GEN-SINGLENR.

**Example 8.** As shown in Table 4, there are four records, $o_{11}$, $o_{21}$, $o_{22}$ and $o_{31}$. Suppose the operator for each attribute is $=$. We want to find a minimal valid NR that identifies $o_{11}$.

SEL-CLAUSES. First we initialize $r$ as $t_1 \wedge t_2 \Rightarrow e_1$. Since $Cov(r) = \{o_{11}, o_{21}, o_{22}\}$ and $S_1 = \{o_{11}\}$, $Cov(r) \backslash S_1 = \{o_{21}, o_{22}\}$. For all the attribute-value pairs in $o_{21}$, only $t_3$ does not appear in $o_{11}$, then $o_{11}$ satisfies $\neg t_3$. Thus we add $\neg t_3$ into LHS$(r)$. For $o_{22}$, as $t_4$ does not appear in $o_{11}$, we add $\neg t_4$ into LHS$(r)$. Now a valid rule $r$ that identifies $o_{11}$ is generated such that $r = t_1 \wedge t_2 \wedge \neg t_3 \wedge \neg t_4 \Rightarrow e_1$.

MIN-RULE. For each clause $T_i$ in LHS$(r)$, we check whether the sub-rule of $r$ excluding $T_i$ is still valid. Since $t_2 \wedge \neg t_3 \wedge \neg t_4 \Rightarrow e_1$ is still valid, $t_1$ is removed. Since $\neg t_3 \wedge \neg t_4 \Rightarrow e_1$, $t_2 \wedge \neg t_4 \Rightarrow e_1$ and $t_2 \wedge \neg t_3 \Rightarrow e_1$ are all invalid, $t_2$, $t_3$ and $t_4$ should not be removed. Thus, the minimal sub-rule $t_2 \wedge \neg t_3 \wedge \neg t_4 \Rightarrow e_1$ is output.

### 3.4 Analysis

The following theorem shows the correctness of Algorithm 1.

**Theorem 2.** *Given a training data set $\mathbb{S}$, Algorithm 1 outputs an ER-rule set $R_{min}$ that is valid, complete and independent.*

Now we analyze the time complexity of our algorithms.

**Lemma 3.** *The running time of Gen-PR is $O(n_o \cdot l_o \cdot n_c + n_t^l)$, where $n_o$ is the number of records in $S$, $l_o$ is the maximum number of attribute-value pairs in each record, $n_c$ is the maximum size of all groups in the training data set and $n_t$ is the maximum number of attribute-value pairs in all the groups in the training data set.*

Although Gen-PR has super-linear time complexity, the values of $l_o$, $n_c$ and $l$ are quite small in practice that can be considered as constants. Therefore, Gen-PR is linear in the number of records.

**Lemma 4.** *The running time of Sel-Clauses(o) is $O(n_o \cdot l_o)$.*

**Lemma 5.** *The running time of Min-Rule(r) is $O(n_o \cdot |r|^2)$.*

**Lemma 6.** *The running time of GREEDY-SETCOVER(R, S) is $O(n_o \cdot |R| \cdot min(|R|, n_o))$.*

**Theorem 3.** *Let $l_m = max\{l_r^2, l_o\}$ where $l_r$ is the largest length of a rule. The running time of Algorithm 1 is $O(n_o^2 \cdot l_m + n_t^l + n_o \cdot |R| \cdot min\{|R|, n_o\})$.*

Although the running time of our algorithm in the worst case is quadratic in number of records, our proposed solution actually scales linearly in our experiments.

## 4 RULE-BASED ENTITY RESOLUTION

In this section, we discuss the algorithm of entity resolution by leveraging ER-rules. We first define the rule-based ER problem. Next we develop an online algorithm for rule-based ER problem. Finally, we describe how to incorporate this algorithm into a generalized ER framework.

**Problem 1 (Rule-based ER).** *Rule-based ER takes $U$ and $R_E$ as input, and outputs $\mathbb{U}$. $U$ is a data set, $R_E$ is an ER-rule set of entity set $E = \{e_1, \ldots, e_m\}$, $\mathbb{U} = \{U_1, \ldots, U_m\}$ is a partition of records where each group $U_j (1 \leq j \leq m)$ is a subset of $U$ which are determined to refer to the entity $e_j$ and $\cup_{1 \leq j \leq m} U_j$ is a subset of $U$.*

Our rule-based ER algorithm R-ER (shown in Algorithm 5) scans records one by one and determines the entity for each record. The determination process can be divided into the following steps.

First, we find all the rules satisfied by $o$ (FINDRULES). Second, for each entity $e$ to which $o$ might refer, we compute the confidence that $o$ refers to $e$ according to the rules of $e$ that are satisfied by $o$ (COMPCONF). Third, we select the entity $e$ with the largest confidence to which $o$ might refer, and if this confidence is larger than a confidence threshold, it is determined that $o$ refers to $e$ (SELENTITY). These procedures are described as follows.

FINDRULES (lines 14-25). It takes a record $o_i$ and find all the rules that are satisfied by $o_i$. The intuitive idea is to compare $o_i$ with each rule. In practice, $o_i$ does not match the LHS of most of the rules. In order to find the rules whose LHS are matched by $o_i$ efficiently, we construct an inverted index and a B-tree for rules, denoted by $L_R$ and $T_R$ respectively. $L_R$ is for rules including clauses with $=$ and $\in$ operators and $T_R$ is for rules including clauses with range operators, such as $\leq$. For example, given an attribute-value pair $t =$ (name,"wang"), $L_R(t)$ (or $T_R(t)$) stores the rules which include $t$.

**Algorithm 5** R-ER

**Input:** $U$, $R_E$, $\theta_C$
**Output:** $\mathbb{U}$

1: Initialize:
2: **for** each entity $e_j$ in $E$ **do**
3: $U_j \leftarrow \emptyset$;
4: **end for**
5: **for** each $o_i$ in $U$ **do**
6: $R(o_i) \leftarrow$FINDRULES($o_i$);
7: **for** each entity $e_j$ in $E$ **do**
8: $R(e_j) \leftarrow \{r|\mathsf{RHS}(r) = e_j\}$;
9: $C(o_i, e_j) \leftarrow$COMPCONF($R(o_i) \cap R(e_j)$);
10: **end for**
11: SELENTITY($o_i, \theta_C$);
12: **end for**
13: Return $\mathbb{U} \leftarrow \{U_1, U_2, ..., U_m\}$;

14: **procedure** FINDRULES($o_i$)
15: $R(o_i) \leftarrow \emptyset$;
16: **for** each attribute-value pair $t$ in $o_i$ **do**
17: $R(o_i) \leftarrow R(o_i) \cap L_R(t) \cap T_R(t)$;
18: **end for**
19: **for** each $r$ in $R(o_i)$ **do**
20: **if** $o_i$ does not match $\mathsf{LHS}(r)$ **then**
21: $R(o_i) \leftarrow R(o_i) - \{r\}$;
22: **end if**
23: **end for**
24: return $R(o_i)$;
25: **end procedure**

26: **procedure** COMPCONF($R$)
27: $C \leftarrow \sum_{r \in R} w(r)$;
28: return $C$;
29: **end procedure**

30: **procedure** SELENTITY($o_i, \theta_C$)
31: $j' \leftarrow argmax\{C(o_i, e_j)|1 \leq j \leq m\}$
32: **if** $C(o_i, e_{j'}) \geq \theta_C$ **then**
33: add $o$ to $U_{j'}$;
34: **end if**
35: **end procedure**

**Example 9.** Let us consider record $o_1$ in Table 1 and the rule set in Example 2. An inverted index for rules is shown in Table 5. As $o_1$ is named "wei wang" and has a coauthor "zhang", we first find the related rules of (name,"wei wang") which are $\{r_1, r_2, r_3, r_4\}$; then we find the related rules of (coa,"zhang") which is $\{r_4\}$; since the intersection of the two rule sets is $\{r_4\}$, $o_1$ should compare with only one rule, $r_4$.

COMPCONF (lines 26-29). Let $R(o)$ denote the rule set satisfied by $o$ and $R(e)$ denote the rule set of entity $e$. Intuitively, the more rules in $R(o) \cap R(e)$, the larger the weight $w(r)$ of each rule $r$ in $R(o) \cap R(e)$, the more confident that $o$ refers to $e$. Thus the confidence of $o$ referring to $e$ (denoted by $C(o, e)$) is defined as below. $C(o, e) = \sum_{r \in R(o) \cap R(e)} w(r)$.

**Example 10.** Consider $o_{12}$ in Table 1, the rule set in Example 2 and another rule $r_5$: (name, "wei wang")$\wedge$(coa, "duncan")$\Rightarrow e_1$. Let the weight of each rule be 1. By running FINDRULES on $o_{12}$, we can get $R(o_{12}) = \{r_1, r_5\}$. Since $R(e) = \{r_1, r_4, r_5\}$, the confidence results for $o_{12}$ are $C(o_{12}, e_1) = w(r_1) + w(r_5) = 2$; $C(o_{12}, e_2) = C(o_{12}, e_3) = 0$.

TABLE 5
Index for ER-Rules

| attribute-value pair | rule |
|---|---|
| (name,"wei wang") | $r_1$, $r_2$, $r_3$, $r_4$ |
| (coa,"kum") | $r_1$ |
| (coa,"lin") | $r_2$ |
| (coa,"shi") | $r_3$, $r_4$ |
| (coa,"zhang") | $r_4$ |

SELENTITY (lines 30-35). Given a record $o_i$, we select the entity $e_j$ which has the largest confidence among all the entities. If the largest confidence $C(o_i, e_j)$ is also larger than a given confidence threshold, then $o_i$ is determined to refer to $e_j$ and is put into the group $U_j$; otherwise, it is determined that $o_i$ does not refer to any entity in $E$.

**Example 11.** Let us consider the confidence results for $o_{12}$ in Example 11. Let $\theta_C = 0$. As $C(o_{12}, e_1) = 2$ is the maximum and $C(o_{12}, e_1) > \theta_C$, $o_{12}$ is then determined to refer to $e_1$ and is put into group $U_1$.

We now analyze the complexity of Algorithm 5.

**Theorem 4.** *Let $n_o$ be the number of records, $n_r$ be the maximal size of related rule set for each record, $l_m$ be the maximum length of rules, and $l_o$ be the average number of attribute-value pairs for each record. Algorithm 5 runs in $O(n_o \cdot n_r \cdot l_m \cdot l_o)$ time.*

In practice, $l_o$, $l_m$ and $n_r$ are quite small which can be considered as constants, then the running time of Algorithm 5 is linear to the number of records.

*Application in generic ER.* Now we present a generic ER framework, shown in Algorithm 6, where our approach is combined with traditional ER methods to resolve entities.

**Algorithm 6** GENERAL-ER

**Input:** $U$, $R$, $E$
**Output:** A partition $\mathbb{U}$ of $U$

1: $\mathbb{U}'$ =R-ER($U, R$);
2: $U'$ =MERGE($\mathbb{U}'$);
3: $\mathbb{U}$ =T-ER($U \cup U' - \cup_{U_i \in \mathbb{U}'} U_i$);
4: Return $\mathbb{U}$

Given a data set, we first use existing ER-rules to identify records (line 1). These rules can be discovered from existing high quality data such as master data or manually identified data. Inspired by the swoosh method [21], each cluster is then merged into a composite record via a merge function (line 2). Finally a traditional ER method, denoted by T-ER, can be applied (line 3) to identify the new data set.

Moreover, in order to identify more records, the current ER result can be used as the training data to discover new ER-rules. The training data can also be obtained by using techniques, such as relevant feedback, crowd sourcing and knowledge extraction from the web. Therefore, with the accumulated information, ER-rules for more entities can be discovered.

## 5 RULE UPDATE

The discovered rule set might be invalid, incomplete, or contain useless rules if the training data is incomplete or out-of-date. To

ensure the performance of the discovered rule set on new records, we introduce an evolution method of rules in this section. First, we introduce the problems that rules might have.

*Invalid rule*. A rule $r$ is invalid if there exist records that match $\mathsf{LHS}(r)$ but do not refer to $\mathsf{RHS}(r)$. Invalid rules might be discovered when the information of entities is not comprehensive. For example, suppose the training data set involves the records in Table 1 except $o_{31}$. The rule $r$: (name ="wei wang")$\wedge$(coa $\in$"zhang")$\Rightarrow e_1$ can be generated. For $o_{31}$, it matches $\mathsf{LHS}(r)$ but does not refer to $e_1$. Therefore, $r$ is an invalid rule.

*Useless rule*. An ER-rule $r$ is called a useless rule if $Cov(r) = \emptyset$, since no records are identified by $r$. Rules will become useless when entity features change. For instance, authors may change their research areas or teams. As a result, the topics, conferences and coauthors of their papers will change correspondingly. Then some rules may become useless. For example, the author *wei wang* in UNSW has changed his research interest from $XML$ to *keyword search* and *similarity join*; and his coauthors from *ju* and *jiang* to *lin*, etc.

*Incomplete rule set*. An ER-rule set $R$ of entity set $E$ is incomplete if there are records referring to entities in $E$ that are not covered by $R$. Both the incomprehensive information of entities and continuous changes of entity features would cause a rule set become incomplete.

To solve these problems, we develop some methods to identify candidate invalid rules and candidate useless rules and discover new effective ER-rules.

*Identify invalid rules*. Rules $r_1$ and $r_2$ are candidate invalid rules if there is a record $o$ that matches both $\mathsf{LHS}(r_1)$ and $\mathsf{LHS}(r_2)$ but $\mathsf{RHS}(r_1) \neq \mathsf{RHS}(r_2)$.

*Identify useless rules*. Given a time threshold $\theta_t$(days) and a number threshold $\theta_n$, we determine a rule $r$ to be a candidate useless rule if for $\theta_t$(days), more than $\theta_n$ new records have been determined to refer to $\mathsf{RHS}(r)$ but none of them is identified by $r$.

*Discover new rules*. By considering the R-ER result of a new data set as a training data, the rule discovery strategy can be applied to discover new rules.

After candidate invalid rules, candidate useless rules or new rules are discovered, by exploiting users' feedback, we can finally determine among these rules, which should be deleted or inserted and the rule set can then be updated accordingly.

## 6 Experimental Evaluation

In this section we perform extensive experiments to validate our methods. Using real data sets, we evaluate (1) the effectiveness of our rule learning algorithm (DiscR) and our rule-based ER approach, (2) the impact of training data size on ER accuracy and the number of generated rules, (3) the impact of rule length threshold on ER accuracy, and (4) the scalability of DiscR and R-ER with the size of data.

Note that, to evaluate the effectiveness of both DiscR and R-ER, we compute the ER-results output by R-ER, using the rules discovered by DiscR.

### 6.1 Experimental Setting

Considering paper-author identification is one of the most difficult ER problems, we use the following data sets to evaluate our algorithms.

TABLE 6
Names Corresponding to Multiple Authors

| name | #aut | name | #aut |
|---|---|---|---|
| "jing wang" | 11 | "ping zhang" | 4 |
| "yan liu" | 10 | "hui wang" | 11 |
| "jian zhang" | 8 | "xin zhang" | 13 |
| "lei chen" | 4 | "jun yang" | 9 |
| "jun sun" | 7 | "wei wang" | 25 |

(1) *dblp data* is a selection from DBLP Bibliography[3] containing 1,812 paper-authors, which is divided into groups according to the authors' identities in DBLP. The author names in this data set (shown in Table 6) are quite representative, since each name is shared by a large number of authors. Hence, these records might be the most difficult to identify in DBLP.

(2) *kdd data*[4] is the validation data set for Track 1 of KDD-Cup 2013, which contains 47,081 paper-authors. It is the ground truth obtained from the user edits at the Academic Search website, where an assignment of a paper to an author is known to be incorrect if an author deleted the paper from the profile, or correct if an author confirmed it [40].

The overlap between these two data sets is little since they only have 26 paper-authors in common, which is 0.05 percent of kdd data and 1.4 percent of dblp data.

(3) *training data* are produced by random sampling from *dblp data* and *kdd data*, controlled by the parameter trainpercent: the percentage of records sampled from each cluster. For instance, the training data with train% $= 20\%$ is generated by sampling $max\{0.2|c|, 1\}$ records from each cluster $c$ in the original data. Note that, there is at least one selected record for each cluster.

To test the effectiveness of our rule discovery algorithm, the average cluster size in the training data should be small, such as $\leq$3. Hence, we fix train% $= 20\%$ for *dblp data* and train% $= 5\%$ for *kdd data* due to their different average cluster sizes.

*Algorithms*. We compare our proposed algorithms against GHOST [11]. GHOST is one of the state-of-the-art name disambiguation algorithms which focuses on the problem of identifying authors with identical names in publications. The basic idea of GHOST is to build a graphical model of the input such that each node represents a record and each edge represents a co-authorship and then computes similarities between records by exploiting the relationships among every pair of publications.

We also compared our method against the leading algorithm [41] in the KDD cup 2013, denoted by CFR. This algorithm extracted several features from the provided data set at first, and then trained classification and ranking models using these features, and finally combined these models to boost the performance.

Since most of the paper-authors can be identified based on the author name and one of the coauthors, we set the rule length threshold $l = 2$ by default. To permit the match between a full name and its abbreviation, we define the operator for attribute name as a fuzzy matcher $\approx$, such that two names are matched only if one name can be

3. http://dblp.uni-trier.de.
4. http://www.kaggle.com/c/kdd-cup-2013-author-paper-identification-challenge/data.

TABLE 7
Comparison

| F-measure | | GHOST | R-ER |
|---|---|---|---|
| dblp | "hui wang" | 0.32 | 0.92 |
| | "jian zhang" | 0.28 | 0.76 |
| | "jing wang" | 0.95 | 0.98 |
| | "jun sun" | 0.95 | 1.00 |
| | "jun yang" | 0.75 | 0.97 |
| | "lei chen" | 0.23 | 0.93 |
| | "yan liu" | 0.42 | 0.86 |
| | "ping zhang" | 0.28 | 0.99 |
| | "xin zhang" | 0.93 | 1.00 |
| | "wei wang" | 0.43 | 0.90 |
| | **average** | 0.55 | 0.93 |
| **MAP** | | CFR | R-ER |
| kdd | | 0.983 | 0.989 |

transformed into the other by inserting several letters (neither deletion nor replacement is allowed). Specifically, $s_1 \approx s_2$ if edit-distance($s_1$, $s_2$) $< 1$, where the costs of insertion, deletion and replacement are 0.1, 1 and 1 respectively.

Our algorithms were implemented in C++ and compiled using Microsoft Visual Studio 2010. The experiments are conducted on a core i7 2.00 GHz PC with 8GB RAM, running Microsoft Windows 7.

*Accuracy Measures.* To ensure a fair comparison with other ER approaches, two accuracy measures are used. One is F-measure used by GHOST, which is the harmonic mean of precision and recall. The other is mean average precision (MAP) used by CFR, which is also a well-known measure from information retrieval that factors in precision at all recall levels [40].

### 6.2 Comparison

In the first set of experiments, we compare the effectiveness of our methods with GHOST and CFR.

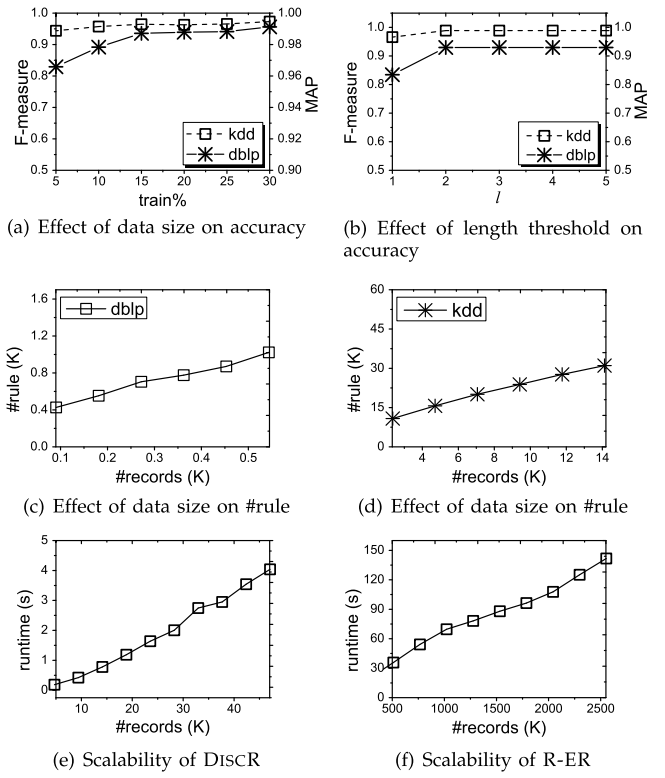
The comparison results are reported in Table 7. We have the following observations. (1) R-ER outperforms GHOST and CFR by up to 41 and 1 percent, respectively. This verifies the benefits of both our rule-based entity resolution method and our rule discovery algorithm on the effectiveness. (2) The lowest F-measure of R-ER is 76 percent, while it is 28 percent for GHOST. This result verifies the robustness of our method.

### 6.3 Effect of Updating Rules

In this section, we evaluate the impact of updating rules on the accuracy of ER-result. We compare the accuracy of ER-results based on the rules which are not updated and the rules which are updated. The results are reported in Table 8, where "no-upt" represents the R-ER results with rules not updated, and "upt" represents the R-ER results with the rules updated automatically based on testing data sets without correctness check by users. It is observed that *upt* outperforms *no-upt* on both *dblp* and *kdd* by up to 5.4 and 0.3 percent respectively. Specifically, *upt* outperforms *no-upt* for all the cases on *dblp*. This verifies that updating rules is indeed useful for improving the quality of rules.

TABLE 8
Effect of Updating Rules

| F-measure | | no-upt | upt |
|---|---|---|---|
| dblp | "hui wang" | 0.85 | 0.92 |
| | "jian zhang" | 0.70 | 0.76 |
| | "jing wang" | 0.97 | 0.98 |
| | "jun sun" | 1.00 | 1.00 |
| | "jun yang" | 0.95 | 0.97 |
| | "lei chen" | 0.81 | 0.93 |
| | "yan liu" | 0.77 | 0.86 |
| | "ping zhang" | 0.96 | 0.99 |
| | "xin zhang" | 0.96 | 1.00 |
| | "wei wang" | 0.88 | 0.90 |
| | average | 0.88 | 0.93 |
| **MAP** | | no-upt | upt |
| kdd | | 0.986 | 0.989 |




(a) Effect of data size on accuracy
(b) Effect of length threshold on accuracy
(c) Effect of data size on #rule
(d) Effect of data size on #rule
(e) Scalability of DISCR
(f) Scalability of R-ER

Fig. 1. Experimental results.

### 6.4 Effect of Training Data Size and Threshold $l$

In the first experiment, we examine the effect of varying the training data size on accuracy. The results are reported in Fig. 1a. We have following observations. (1) The accuracy on *dblp* reaches 90 percent when train% $= 10\%$, and the accuracy on *kdd* reaches 98.8 precent when train% $= 5\%$, which shows that our method can achieve high accuracy under a small training data. (2) With the growth of the training data size, although there is a narrow fluctuation caused by random selection of training data sets, the accuracy increases gradually as expected.

In the second experiment, we examine the effect of varying the training data size on the number of generated rules. The results are reported in Figs. 1c and 1d. It shows that the number of rules is larger than the number of training records on both data sets. However, the size of rules would not be large since it grows slowly with the training data size.

In the third experiment, we compare the accuracy as the length threshold $l$ varies from 1 to 5. The results are reported in Fig. 1b. It shows that the accuracy reaches to the highest value when $l = 2$ for both data sets, which means

most rules with length larger than 2 are not effective to identify records in our experiments.

### 6.5 Efficiency and Scalability

In this section, we investigated the runtime performance of DiscR and R-ER algorithms as we increase the number of records.

Fig. 1e shows the performance of DiscR on *kdd*. It can be observed that DiscR scales linearly with the number of records. The results tell us although the runtime of DiscR in the worst case is quadratic in number of records, it actually scales quite well in practise.

Fig. 1f shows the performance of R-ER on *dblp*. As expected, the runtime of R-ER is approximately linear to the number of records which is accordance with our time complexity analysis in Section 4. Thus, the result verifies the scalability of R-ER.

*Summary.* From the experimental results, we can draw the following conclusions. In our experiments, (a) DiscR and R-ER can achieve a high accuracy using a small training data; (b) updating rules indeed help identify records; (c) the number of generated rules scales well with the training data size on both data sets; (d) rules with length larger than 2 are seldom needed to identify records; and (e) both DiscR and R-ER scales well with the size of data.

## 7 Related Work

The work on entity resolution can be broadly divided into three categories.

*Pairwise ER.* Most works on ER focus on record matching [1], which involves comparing record pairs and identifying whether they match. A major part of work on record matching focuses on similarity functions [2], [3], [4], [5]. To capture string variations, [6] proposed a transformation-based framework for record matching. Some machine-learning-based approaches [7], [8] can identify matching strings which are syntactically far apart. Similarity based on record relationships [9], [11] are also proposed to solve the people identification problem.

Since in our work, records are not compared with each other, our work is orthogonal to record matching. However, string similarity functions can be applied to fuzzy match operator (denoted by $\approx$) in ER-rules. For example, given a string $s$, we say $s \approx$ "wei wang" if the edit distance between $s$ and "wei wang" is smaller than a given threshold.

Decision trees are employed to teach record matching rules in [10]. However, decision trees cannot be used to discover ER-rules. This is because the domain of the right-hand side of record matching rules is {yes, no} (two records are mapped or not mapped), while the domain of the right-hand side of ER-rules is an entity set.

*Non-pairwise ER.* The research on non-pairwise ER includes clustering strategies [11], [15], [16], [32] and classifiers [18], [19]. Most strategies solve ER based on the relationship graph among records, by modeling the records as nodes and the relationships as edges. Machine learning approaches [12], [31] are also proposed by using global information to solve ER effectively. However, these methods are not suitable for massive data because of efficiency issues. We choose a representative work [11] for comparison.

*Scaling.* Some other works [13], [20], [22] treat the ER algorithm as black box and focus on developing scalable framework for ER. Indexing techniques used for ER have been surveyed by Christen[23]. [14] focuses on how to update ER results efficiently when ER logic evolves. These techniques are orthogonal to our work and can be applied to accelerate our rule-based ER algorithm.

Note that, among the existing works on pair-wise ER, rule-based approaches [35], [36] are closer to our work. These rules differ from our work as they focus on determining whether two records refer to the same entity while our work focuses on determining whether a record refers to an existing entity.

Our preliminary work [29] proposed rule-based ER and rule discovery strategies. However, the preliminary work only proposed a heuristic method for rule discovery, without efficiency and accuracy guarantee. In this paper, we propose a new definition of rules and effective algorithms for rule discovery.

## 8 Conclusion

This paper developed a class of ER-rules which are capable to describe the complex matching conditions between records and entities. Based on these rules, we developed an ER algorithm R-ER. We experimentally evaluated our algorithms on real data sets. The experimental results show that our algorithm can achieve a good performance both on efficiency and accuracy. For future work, we would like to extend our techniques to more general cases. For instance, how to discover ER-rules when the operator for each attribute is not given? We would also like to consider how to incorporate human resources, such as Crowd, into our rule-discovery framework to improve the quality of rules.

## Appendix A: Proofs for Section 3

### A.1 Proof of Proposition 4

**Proof.** The problem of verifying an ER-rule $r$ whether it is valid and its length is no more than $l$ is in NP.

We use a reduction from the Set Cover Problem (SCP) [34]. SCP is known to be NP-Complete. Let ($U_e$, $C$, $k$) be an instance of SCP, where $U_e = \{o_1, o_2, \ldots, o_n\}$ is a finite set of elements and $\ldots C = \{C_1, C_2, \ldots C_m\}$ is a collection of subsets of $U$. The question is that does $C$ contain a cover for $U$ of size $k$.

We build an instance of LPR in the following way.

1. Construct a data set $S = \{o_1, \ldots, o_n, o_{n+1}\}$. Let $\mathbb{S} = \{S_1, S_2\}$ be the training data set of $S$, where $S_1 = U_e = \{o_1, \ldots, o_n\}$ is the data set referring to $e_1$ and $S_2 = \{o_{n+1}\}$ is the data set referring to entity $e_2$.

2. Let $o_{n+1} = \{t_1, t_2, \ldots, t_m\}$ where each $t_i$ in $o_{n+1}$ is an attribute-value pair, such that $\overline{Cov(t_i)} = (S_1 \backslash C_i) \cup \{o_{n+1}\} = (S_1 \backslash C_i) \cup S_2 = S \backslash C_i$. Thus $\overline{Cov(t_i)} = C_i$.

The question is that whether there exists a valid PR $r$ such that $r$ identifies $o_{n+1}$ and $|r| \leq k$. Clearly, $r$ is valid and identifies $o_{n+1}$ iff $Cov(r) = \{o_{n+1}\} = S_2$, that is $\overline{Cov(r)} = S \backslash S_2 = S_1$.

If $C' = \{C_{i1}, C_{i_2}, \ldots, C_{i_k}\}$ is the solution to the instance of SCP, let $r$ be the ER-rule $t_{i_1} \wedge t_{i_2} \cdots \wedge t_{i_k} \Rightarrow e_2$, then we have,

$$\begin{aligned}\overline{Cov(r)} &= \overline{Cov(t_{i_1}) \cap Cov(t_{i_2}) \ldots Cov(t_{i_k})} \\ &= \overline{Cov(t_{i_1})} \cup \overline{Cov(t_{i_2})} \ldots \cup \overline{Cov(t_{i_k})} \\ &= C_{i1} \cup C_{i_2}, \ldots \cup C_{i_k} \\ &= U_e = S_1.\end{aligned}$$

Thus, $r$ is the solution to the instance of LPR since $Cov(r) = \{o_{n+1}\}$ and $|r| = k$.

On the contrary, if $r = t_{i_1} \wedge t_{i_2} \cdots \wedge t_{i_k} \Rightarrow e_2$ is the solution to the instance of LPR, then $C' = \{C_{i1}, C_{i_2}, \ldots, C_{i_k}\}$ is the solution to the instance of SCP. Clearly, this reduction is polynomial. Hence, LPR is NP-Complete.

### A.2 Proof of Proposition 6

**Proof.** It is similar to the proof of Proposition 4. We use a reduction from the smallest set covering problem (SCP). Given an instance $(U, C)$ of SCP, we build an instance of SNR in the following way.

Let $\mathbb{S} = \{S_1, S_2, S_3\}$ be the training data set of $S = \{o, o', o_1, \ldots, o_n\}$, where $S_1 = \{o\}$ is the data set referring to $e_1$, $S_2 = \{o'\}$ is the data set referring to $e_2$ and $S_3 = U = \{o_1, \ldots, o_n\}$ is the data set referring to $e_3$. Suppose that the operator for each attribute is $=$. Let $T_1$, $T_2$ and $T_3$ be the sets of attribute-value pairs that appear in $S_1$, $S_2$ and $S_3$ respectively, where $T_1 = \{t_0\}$, $T_2 = \{t'_0\}$ and $T_3 = \{t_0, t_1, t_2, \ldots, t_m\}$. Moreover, for each $t_i (1 \leq i \leq m)$, $Cov(t_i) = C_i$.

Our goal is to find a valid NR $r$ such that $r$ identifies $o$ and $|r|$ is minimized. Clearly, $r$ is valid and identifies $o$ iff $Cov(r) = \{o\}$.

If $C' = \{C_{i1}, C_{i_2}, \ldots, C_{i_k}\}$ is the solution to the instance of SCP, let $r$ be the ER-rule $r$: $t_0 \wedge \neg t_{i_1} \wedge \neg t_{i_2} \cdots \wedge \neg t_{i_k} \Rightarrow e_1$, then we have

$$\begin{aligned}\overline{Cov(r)} &= \overline{Cov(\neg t_{i_1}) \cap Cov(\neg t_{i_2}) \cdots \cap Cov(\neg t_{i_k}) \cap Cov(t_0)} \\ &= Cov(t_{i_1}) \cup Cov(t_{i_2}) \ldots \cup Cov(t_{i_k}) \cup \overline{Cov(t_0)} \\ &= C_{i1} \cup C_{i_2}, \ldots \cup C_{i_k} \cup \{o'\} = U \cup \{o'\}.\end{aligned}$$

Then we have $Cov(r) = \{o\}$. Thus $r$ is a valid rule with length $k$ that identifies $o$. We prove that $r$ is the solution to the instance of SNR. If the solution to the instance of SNR is not $r$, but $r' = t_0 \wedge \neg t_{j_1} \wedge \neg t_{j_2} \cdots \wedge \neg t_{j_{k'}} \Rightarrow e_1$, where $k' < k$. Then $C'' = \{C_{j_1}, C_{j_2}, \ldots, C_{j_{k'}}\}$ is a smaller subset than $C'$ that covers $U$, which contradicts to the assumption that $C'$ is the solution to the instance of SCP. Similarly, if $r = t_0 \wedge \neg t_{i_1} \wedge \neg t_{i_2} \cdots \wedge \neg t_{i_k} \Rightarrow e_1$ is the solution to the instance of SNR, by contradiction it can be proved that $C' = \{C_{i1}, C_{i_2}, \ldots, C_{i_k}\}$ is the solution to the instance of SCP. Clearly, this reduction is polynomial. Hence, SNR is NP-Hard. □

### A.3 Proof of Proposition 8

**Proof 6.** If $R$ is minimal for $S$ and $R$ is not independent, then there exists a rule $r \in R$ such that $R - \{r\} \models r$. Then according to Proposition 1, any record in $S$ that is identified by $r$ is also identified by $R - \{r\}$, that is $Cov(r) \subseteq Cov(R - \{r\})$. Therefore $Cov(R) = Cov(R - \{r\})$, which contradicts to the assumption that $R$ is minimal. □

Now we prove that $r$ is valid. $\forall o_i \in Cov(r_o) \backslash S_o$, since there exists a clause $T_k$ in $\mathsf{LHS}(r)$ such that $o_i$ does not satisfy $T_k$, then $o_i$ does not match $\mathsf{LHS}(r)$. Thus $Cov(r) \cap (Cov(r_o) \backslash S_o) = \emptyset$. Then $Cov(r) \subseteq \overline{Cov(r_o) \backslash S_o}$. Since $r_o$ is a sub-rule of $r$, $Cov(r) \subseteq Cov(r_o)$. Then we have the following. Thus, $r$ is valid. □

$$\begin{aligned}Cov(r) &\subseteq Cov(r_o) \cap \overline{Cov(r_o) \backslash S_o} \\ &= Cov(r_o) \cap (\overline{Cov(r_o)} \cup S_o) \\ &= S_o \cap Cov(r_o) \subseteq S_o.\end{aligned}$$

### A.4 Proof of Proposition 11

**Proof.** Clearly, $r'$ output by MIN-RULE is valid. Now we prove that $r'$ is minimal. Let $r = t_1 \wedge t_2 \cdots \wedge t_m \Rightarrow e$ and $\mathsf{LHS}(r') = t_{i_1} \wedge t_{i_2} \cdots \wedge t_{i'_m}$. If $r'$ is not minimal, then there exists a clause $i_j$ $(1 \leq i_j \leq m)$ in $r'$ such that the rule $r''$: $t_{i_1} \wedge t_{i_2} \cdots \wedge t_{i_{j-1}} \wedge t_{i_{j+1}} \cdots \wedge t_{m'} \Rightarrow e$ is still valid. The only reason that $t_{i_j}$ is in $r'$ is that, the rule $r'''$: $t_{i_1} \wedge t_{i_2} \cdots \wedge t_{i_j} \wedge t_{i_j+1} \cdots \wedge t_m \Rightarrow e$ is not valid. Since $r''$ is a sub-rule of $r'''$, $r$ should be valid. Hence, the assumption that $r'$ is not minimal is not true. □

### A.5 Proof of Theorem 2

**Proof.** From Lemma 1 and Lemma 2, each rule in $R_{min}$ is valid. Thus $R_{min}$ is valid. As Algorithm 1 shows (lines 5-11), each record in $\mathbb{S}$ can be identified by learned rules. Moreover, $R_{min}$ is a minimal subset of the learned rule set that covers $\mathbb{S}$. Thus $R_{min}$ is both complete and independent. □

### A.6 Proof of Lemma 3

**Proof.** The time required for line 3 is no more than $n_c$, then the time required for Step 1 (line 2-7) is $O(n_o \cdot l_o \cdot n_c)$. The time required for Step 2 (line 8-14) is $C^2_{n_t} + \cdots + C^l_{n_t}$. Since $C^2_{n_t} + \cdots + C^l_{n_t} \leq l \cdot C^l_{n_t} \leq n_t(n_t - 1) \cdots (n_t - l + 1) \leq n_t^l$, Algorithm 2 runs in $O(n_o \cdot l_o \cdot n_c + n_t^l)$. □

### A.7 Proof of Lemma 4

**Proof.** Since $|Cov(r) \backslash S_o| < n_o$, there are at most $n_o \cdot l_o$ clauses that should be checked whether they are satisfied by $o$. Thus Algorithm 3 runs in $O(n_o \cdot l_o)$. □

### A.8 Proof of Lemma 5

**Proof.** Since the size of the coverage of each clause in $r$ is no more than $n_o$ and each sub-rule of $r$ has no more than $|r|$ clauses, the time required for computing the coverage of each sub-rule is no more than $n_o \cdot r$. Thus the running time of Algorithm 4 is $O(n_o \cdot |r|^2)$. □

### A.9 Proof of Theorem 3

**Proof.** According to Algorithm 1, the total running time of the whole algorithm is the sum of the following parts (1) the time of GEN-PR (line 2), (2) for all records, the time for checking the validity of $r_o$ for each $o$ (line 6) and the maximal time of MIN-RULE and GEN-SINGLENR for each $o$ (line 3-9); and (3) the time of GREEDY-SETCOVER (line 10).

By Lemma 3, the time complexity of part (1) is $O(n_o \cdot l_o \cdot n_c + n_t^l)$.

For each record $o$, the time complexity of validation is $O(n_o \times l_o)$. According to Lemma 5, since MIN-RULE is

invoked in GEN-SINGLENR, the maximal time complexities of MIN-RULE and GEN-SINGLENR is that of GEN-SINGLENR. That is $O(n_o \cdot |r|^2)$. Thus the time complexity of part (2) is $n_o \times O(n_o \cdot |r|^2)$ since $l_o \leq |r|^2$. With $|r| \leq l_m$, the time complexity of part (2) is $O(n_o^2 \cdot l_m^2)$.

According to Lemma 6, the time complexity of part (3) is $O(n_o \cdot |R| \cdot min\{|R|, n_o\})$.

As a sum of the time complexities of these three parts, the total time complexity is $O(n_o \cdot l_o \cdot n_c + n_t^l \, n_o^2 \cdot l_m^2 + n_o \cdot |R| \cdot min(|R|, n_o))$. Thus the time complexity of the algorithm is $O(n_o^2 \cdot l_m + n_t^l + n_o \cdot |R| \cdot min(|R|, n_o))$ since $n_c \leq n_o$ and $l_o \leq l_m$. □

## ACKNOWLEDGMENTS

This paper was partially supported by NGFR 973 grant 2012CB316200 and NGFR 863 grant 2012AA011004.

## REFERENCES

[1] N. Koudas, S. Sarawagi, and D. Srivastava, "Record linkage: Similarity measures and algorithms," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2006, pp. 802–803.

[2] M. Bilenko and R. J. Mooney, "Adaptive duplicate detection using learnable string similarity measures," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 39–48.

[3] W. W. Cohen, "Integration of heterogeneous databases without common domains using queries based on textual similarity," *ACM SIGMOD Rec.*, vol. 27, no. 2, pp. 201–212, 1998.

[4] L. Gravano, P. G. Ipeirotis, N. Koudas, and D. Srivastava, "Text joins in an RDBMS for web data integration," in *Proc. 12th Int. Conf. World Wide Web*, 2003, pp. 90–101.

[5] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida," *J. Amer. Statist. Assoc.*, vol. 84, no. 406, pp. 414–420, 1989.

[6] A. Arasu, S. Chaudhuri, and R. Kaushik, "Transformation-based framework for record matching," in *Proc. 24th Int. Conf. Data Eng.*, 2008, pp. 40–49.

[7] S. Chaudhuri, B. C. Chen, V. Ganti, and R. Kaushik, "Example-driven design of efficient record matching queries," in *Proc. 33rd Int. Conf. Very Large Databases*, 2007, pp. 327–338.

[8] A. Arasu, S. Chaudhuri, and R. Kaushik, "Learning string transformations from examples," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 514–525, 2009.

[9] R. Bekkerman and A. McCallum, "Disambiguating web appearances of people in a social network," in *Proc. 14th Int. Conf. World Wide Web*, 2005, pp. 463–470.

[10] S. Tejada, C. Knoblock, and S. Minton, "Learning object identification rules for information integration," *Inf. Syst.*, vol. 26, no. 8, pp. 607–633, 2001.

[11] X. Fan, J. Wang, X. Pu, L. Zhou, and B. Lv, "On graph-based name disambiguation," *J. Data Inf. Quality*, vol. 2, no. 2, p. 10, 2011.

[12] L. Shu, B. Long, and W. Meng, "A latent topic model for complete entity resolution," in *Proc. 25th Int. Conf. Data Eng.*, 2009, pp. 880–891.

[13] R. Vibhor, N. D. Nilesh, and N. G. Minos, "Large-scale collective entity matching," *Proc. VLDB Endowment*, vol. 4, no. 4, pp. 208–218, 2011.

[14] S. E. Whang and H. Garcia-Molina, "Entity resolution with evolving rules," *Proc. VLDB Endowment*, vol. 3, no. 1, pp. 1326–1337, 2010.

[15] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Mach. Learn.*, vol. 56, no. 1–3, pp. 89–113, 2004.

[16] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani, "Robust and efficient fuzzy match for online data cleaning," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2003, pp. 313–324.

[17] S. Chaudhuri, V. Ganti, and R. Motwani, "Robust identification of fuzzy duplicates," in *Proc. 21st Int. Conf. Data Eng.*, 2005, pp. 865–876.

[18] P. Singla and P. Domingos, "Object identification with attribute-mediated dependencies," in *Proc. Eur. Conf. Principles Practice Knowl. Discov. Databases: PKDD*, 2005, pp. 297–308.

[19] V. S. Verykios, G. V. Moustakides, and M. G. Elfeky, "A Bayesian decision model for cost optimal record matching," *VLDB J.*, vol. 12, no. 1, pp. 28–40, 2003.

[20] S. E. Whang, D. Menestrina, G. Koutrika, M. Theobald, and H. Garcia-Molina, "Entity resolution with iterative blocking," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2009, pp. 219–232.

[21] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom, "Swoosh: A generic approach to entity resolution," *VLDB J.*, vol. 18, no. 1, pp. 255–276, 2009.

[22] M. Bilenko, B. Kamath, and R. J. Mooney, "Adaptive blocking: Learning to scale up record linkage," in *Proc. IEEE Int. Conf. Data Mining*, 2006, pp. 87–96.

[23] P. Christen, "A survey of indexing techniques for scalable record linkage and deduplication," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 9, pp. 1537–1555, Sep. 2012.

[24] C. Xiao, W. Wang, X. Lin, and J. X. Yu, "Efficient similarity joins for near duplicate detection," in *Proc. 17th Int. Conf. World Wide Web*, 2008, pp. 131–140.

[25] C. Xiao, W. Wang, X. Lin, and H. Shang, "Top-k set similarity joins," in *Proc. IEEE Int. Conf. Data Eng.*, 2009, pp. 916–927.

[26] X. Yin, J. Han, and P. S. Yu, "Object distinction: Distinguishing objects with identical names," in *Proc. IEEE 23rd Int. Conf. Data Eng.*, 2007, pp. 1242–1246.

[27] A. K. Elmagarmid, G. I. Panagiotis, and S. V. Vassilios, "Duplicate record detection: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 1, pp. 1–16, Jan. 2007.

[28] V. V. Vazirani, *Approximation Algorithms*, New York, NY, USA: Springer, 2001, pp. 1–378.

[29] L. Li, J. Li, H. Wang, and H. Gao, "Context-based entity description rule for entity resolution," in *Proc. 20th ACM Int. Conf. Inf. knowl. Manag.*, 2011, pp. 1725–1730.

[30] H. Kopcke, A. Thor, and E. Rahm, "Evaluation of entity resolution approaches on real-world match problems," *Proc. VLDB Endowment*, vol. 3, no. 1, pp. 484–493, 2010.

[31] I. Bhattacharya and L. Getoor, "Collective entity resolution in relational data," *Proc. VLDB Endowment*, vol. 3, no. 1, p. 5, 2010.

[32] H. Kopcke, A. Thor, and E. Rahm, "Domain-independent data cleaning via analysis of entity-relationship graph," *ACM Trans. Database Syst.*, vol. 31, no. 2, pp. 716–767, 2006.

[33] M. Herschel, F. Naumann, S. Szott, and M. Taubert, "Scalable iterative graph duplicate detection," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 11, pp. 2094–2108, Nov. 2011.

[34] C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT press, 2001.

[35] E.-P. Lim, J. Srivastava, S. Prabhakar, and J. Richardson, "Entity identification in database integration," in *Proc. 9th Int. Conf. Data Eng.*, 1993, pp. 294–301.

[36] F. Wenfei, J. Xibei, L. Jianzhong, and M. Shuai, "Reasoning about record matching rules," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 407–418, 2009.

[37] D. Loshin, *Master Data Management*. San Mateo, CA, USA: Morgan Kaufmann, 2009.

[38] A. Arasu and R. Kaushik, "A grammar-based entity representation framework for data cleaning," in *Proc. ACM SIGMOD, Int. Conf. Manag. data*, 2009, pp. 233–244.

[39] S. Abiteboul, H. Richard, and V. Victor, *Foundations of Databases*, vol. 8, Reading, MA, USA: Addison-Wesley, 1995.

[40] S. B. Roy, M. D. Cock, V. Mandava, S. Savanna, B. Dalessandro, C. Perlich, W. Cukierski, and B. Hamner, "The microsoft academic search dataset and kdd cup 2013," in *Proc. KDD Cup 2013 Workshop*, 2013, p. 1.

[41] C. Li, Y. Su, T. Lin, C. Tsai, W. Chang, K. Huang, T. Kuo, S. Lin, Y. Lin, Y. Lu, C. Yang, C. Chang, W. Chin, Y. Juan, H. Tung, J. Wang, C. Wei, F. Wu, T. Yin, T. Yu, Y. Zhuang, S. Lin, H. Lin, and C. Lin, "Combination of feature engineering and ranking models for paper-author identification in KDD Cup 2013," in *Proc. KDD Cup 2013 Workshop*, 2013, p. 2.

[42] J. R. Quinlan, "Learning logical definitions from relations," *Mach. Learn.*, vol. 5, no. 3, pp. 239–266, 1990.

**Lingli Li** received the master's degree in computer science and engineering from HIT, China. She is currently working toward the PhD degree at Harbin Institute of Technology (HIT). Her research interests include data quality, entity resolution.

**Jianzhong Li** is a professor and doctoral supervisor at Harbin Institute of Technology. He is a senior member of CCF. His research interests include database, parallel computing, and wireless sensor networks, etc.

**Hong Gao** is a professor and doctoral supervisor at Harbin Institute of Technology. She is a senior member of CCF. Her research interests include data management, wireless sensor networks and graph database, etc.