The Interaction between Schema Matching and Record Matching in Data Integration

Binbin Gu, Zhixu Li, Xiangliang Zhang, An Liu, Guanfeng Liu, Kai Zheng, Lei Zhao, and Xiaofang Zhou *Senior Member, IEEE*

Abstract—Schema Matching (SM) and Record Matching (RM) are two necessary steps in integrating multiple relational tables of different schemas, where SM unifies the schemas and RM detects records referring to the same real-world entity. The two processes have been thoroughly studied separately, but few attention has been paid to the interaction of SM and RM. In this work we find that, even alternating them in a simple manner, SM and RM can benefit from each other to reach a better integration performance (i.e., in terms of precision and recall). Therefore, combining SM and RM is a promising solution for improving data integration. To this end, we define novel matching rules for SM and RM respectively, that is, every SM decision is made based on intermediate RM results, and vice versa, such that SM and RM can be performed alternately. The quality of integration is guaranteed by a Matching Likelihood Estimation model and the control of semantic drift, which prevent the effect of mismatch magnification. To reduce the computational cost, we design an index structure based on q-grams and a greedy search algorithm that can reduce around 90% overhead of the interaction. Extensive experiments on three data collections show that the combination and interaction between SM and RM significantly outperforms previous works that conduct SM and RM separately.

Index Terms—Data Integration, Schema Matching, Record Matching

1 Introduction

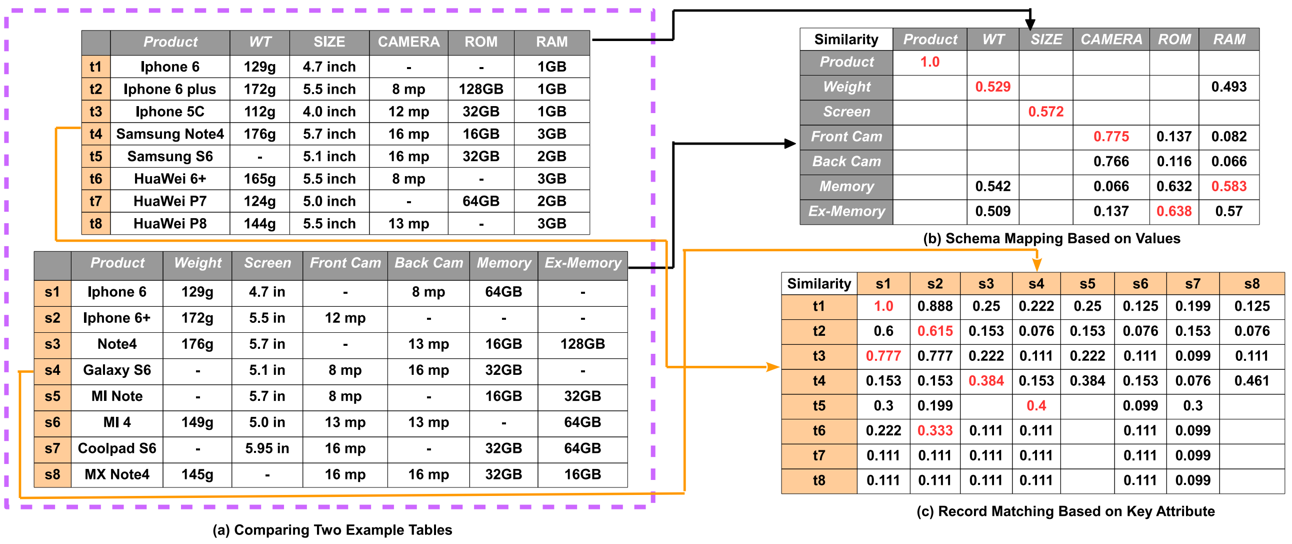
Due to the data explosion in the big data era, the inconsistency between data sources becomes a critical issue in two dimensions: *schema-level inconsistency* and *tuple-level inconsistency*. As a result, merging data from multiple relational databases requires two necessary steps, namely Schema Matching (SM) and Record Matching (RM), in order to achieve a uniform and consistent data view. Here, SM unifies the schemas of different data sets; while RM finds pairs of linked records referring to the same entity.

There have been a host of works on SM or RM (see [28] or [14] for a survey). Briefly, the state-of-the-art SM method considers both the similarity (or semantic correlation) between the attribute names [20] and the similarity between the set of attribute values (or selected/sampled subsets of attribute values) under the two attributes [29]; while the most advanced RM methods inspect linguistic similarities and structural/relational similarities [3], [30] between key attribute values [4] or indicative non-key attribute values [35] when deciding the matching records among data sets, where key attribute is the one that can uniquely determine a record in a relational table while all the others are non-key attributes. Recently, external domain knowledge and human interventions are also employed to improve the quality of SM [12] or RM [21].

All existing efforts, however, consider the two tasks independently, that is, they first perform SM, and then perform RM subsequently in only one run, which do not pay any attention on the possible interaction between SM and RM in the data integration process. This strategy inevitably gives us only one chance to make decisions, and deprives us further chances to update the links when more and more valuable information is collected from the other task. As a result, these SM and RM methods may easily make wrong decisions without further refined updates. Besides, there are two critical issues with existing works: (1) Existing instance-based SM methods rely heavily on the assumption that the distributions of attribute values under linked attributes should be similar to each other; otherwise it will suffer from low similarity between selected subsets of attribute values with the attributes should be linked, such as the situation in which only a small part of records are shared by the two data sets. (2) The RM linking results are greatly determined by SM linking results. As a result, both missed attribute-pairs and mistaken attribute-pairs would degrade the quality of RM linking results.

We study in this paper the interaction between SM and RM, by performing them alternately for data integration. To achieve this, novel matching rules are proposed: at each RM step, we identify a set of highly possible matching record-pairs based on the already linked attribute-pairs; Likewise, at each SM step, we identify a set of highly-possible matched attribute-pairs based on the already linked record-pairs. For instance, assume a start-up linked key attribute-pairs (`Product`,`Product`) between the two tables in Fig. 1 (a), at the first RM step, we may identify

- *B. Gu, Z. Li, A. Liu, G. Liu, K. Zheng, L. Zhao are with the School of Computer Science and Technology, Soochow University, China. Email: gu.binbin@hotmail.com, {zhixuli, anliu, gfliu,kaizheng, zhaol}@suda.edu.cn.*
- *X. Zhang is with the King Abdullah University of Science and Technology, Jeddah, Saudi Arabia. Email: xiangliang.zhang@kaust.edu.sa*
- *X. Zhou is with the School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane QLD 4072, Australia, and the School of Computer Science and Technology, Soochow University, China. Email: zxf@itee.uq.edu.au*



	Product	WT	SIZE	CAMERA	ROM	RAM
t1	Iphone 6	129g	4.7 inch	-	-	1GB
t2	Iphone 6 plus	172g	5.5 inch	8 mp	128GB	1GB
t3	Iphone 5C	112g	4.0 inch	12 mp	32GB	1GB
t4	Samsung Note4	176g	5.7 inch	16 mp	16GB	3GB
t5	Samsung S6	-	5.1 inch	16 mp	32GB	2GB
t6	HuaWei 6+	165g	5.5 inch	8 mp	-	3GB
t7	HuaWei P7	124g	5.0 inch	-	64GB	2GB
t8	HuaWei P8	144g	5.5 inch	13 mp	-	3GB

	Product	Weight	Screen	Front Cam	Back Cam	Memory	Ex-Memory
s1	Iphone 6	129g	4.7 in	-	8 mp	64GB	-
s2	Iphone 6+	172g	5.5 in	12 mp	-	-	-
s3	Note4	176g	5.7 in	-	13 mp	16GB	128GB
s4	Galaxy S6	-	5.1 in	8 mp	16 mp	32GB	-
s5	MI Note	-	5.7 in	8 mp	-	16GB	32GB
s6	MI 4	149g	5.0 in	13 mp	13 mp	-	64GB
s7	Coolpad S6	-	5.95 in	16 mp	-	32GB	64GB
s8	MX Note4	145g	-	16 mp	16 mp	32GB	16GB

(a) Comparing Two Example Tables

Similarity	Product	WT	SIZE	CAMERA	ROM	RAM
Product	1.0					
Weight		0.529				0.493
Screen			0.572			
Front Cam				0.775	0.137	0.082
Back Cam				0.766	0.116	0.066
Memory		0.542		0.066	0.632	0.583
Ex-Memory		0.509		0.137	0.638	0.57

(b) Schema Mapping Based on Values

Similarity	s1	s2	s3	s4	s5	s6	s7	s8
t1	1.0	0.888	0.25	0.222	0.25	0.125	0.199	0.125
t2	0.6	0.615	0.153	0.076	0.153	0.076	0.153	0.076
t3	0.777	0.777	0.222	0.111	0.222	0.111	0.099	0.111
t4	0.153	0.153	0.384	0.153	0.384	0.153	0.076	0.461
t5	0.3	0.199		0.4		0.099	0.3	
t6	0.222	0.333	0.111	0.111		0.111	0.099	
t7	0.111	0.111	0.111	0.111		0.111	0.099	
t8	0.111	0.111	0.111	0.111		0.111	0.099	

(c) Record Matching Based on Key Attribute

Fig. 1. Two Example Tables for Integration (a) and the Integration Results with Previous Methods ((b) and (c))

$(t_1 \leftrightsquigarrow s_1)$ and $(t_2 \leftrightsquigarrow s_2)$ as linked records as they share the same `Product` values. We then identify (`Weight`, `WT`) as linked attribute-pair given that the two linked records share the same value under the two attributes. We repeat this process iteratively until no more attributes or records can be linked. Finally, we will have all the four attribute-pairs and six record-pairs be correctly linked as demonstrated in Fig. 2. By contrast, traditional methods perform SM and RM in only one run, which as a result introduce $(\mathtt{Ex-Memory} \leftrightsquigarrow \mathtt{ROM})$ and $(t_4 \leftrightsquigarrow s_8)$ as wrong matches, and also miss pairs $(\mathtt{Size} \leftrightsquigarrow \mathtt{Screen\ Size})(\mathtt{t}_3 \leftrightsquigarrow \mathtt{s}_3), (\mathtt{t}_4 \leftrightsquigarrow \mathtt{s}_4), (\mathtt{t}_5 \leftrightsquigarrow \mathtt{s}_5)$ and $(t_6 \leftrightsquigarrow s_6)$ as matched pairs with similarities and thresholds given in Fig. 1(b)(c), where the similarities between attribute values are measured by Levenshtein similarity. Instance-based SM methods in [13], [23] also use instances to facilitate SM. However, the instance-pairs are not from the results RM steps, and thus could be wrongly selected from mismatched attributes such as `ROM` and `Memory` in Figure 1 due to the similar values they have.

Nevertheless, the interaction model raises two challenging issues: Firstly, new linking decisions made at each SM (or RM) step based on intermediate RM (or SM) results should be **reliable**. Otherwise, they may lead to mistaken linking decisions in later stage. Secondly, the potential **semantic drift** in the interaction process should be controlled to prevent mismatch magnification in the subsequent iterations. Both of the two issues are crucial to the quality of the matching results.

To address the two problems, we design a probabilistic model to estimate the *Matching Likelihood* of each matching-record-pair. In particular, we first measure each individual attribute's ability to identify matching-record-pairs, and then estimate the likelihood of each matching-record-pair by jointly considering the identification ability of multiple attributes. The key difficulty lies on how to calculate the dependencies between attributes. A traditional model based on Inclusion-Exclusion principle [6] estimates the matching likelihood by calculating the dependencies between attributes comprehensively, but the computation cost grows exponentially with n (the number of attributes in a table). Another famous model Noisy-All [1], [25] completely neglects the dependencies between attributes for efficiency, with a sacrifice of estimation accuracy. To reach a balance between accuracy and efficiency, we propose a novel combination model which employs the logistic sigmoid function [5] to simplify the function of calculating the dependencies among the attributes into a linear one. Besides, to prevent from the semantic drift issue, we introduce different strategies to check the correctness of each matching-record-pair and matching-attribute-pair respectively. One checks the degree of deviation of every matching-record-pair from the other matching-record-pairs according to the unbiased variance [33], while the other employs cross-validation to use matching attribute-pairs to validate each other.

Computational cost is always an issue when comparing a large number of attribute value pairs in RM and SM. According to our analysis, without any optimizations, the computational complexity of the interaction algorithm can be as high as $O(min(p, q)mn)$, where m and n are the numbers of records in the two tables for integration respectively, and p and q are the number of attributes in the two tables respectively. To reduce the high computational cost, we design an index structure based on q-grams [34] to index all possible matched record-pairs w.r.t. a single attribute. Potentially matchable record-pairs between the two tables are grouped into (possibly overlapped) blocks such that matching-record-pairs are only identified within one block. We then propose a greedy algorithm selecting only one block at a time from all blocks, which brings the maximum benefit (i.e., linking the most matching-record-pairs or matching-attribute-pairs at the next step) with the minimum cost (i.e., comparing the least attribute values). After each step, the algorithm updates the indices and does the greedy block selection again until no more blocks left.

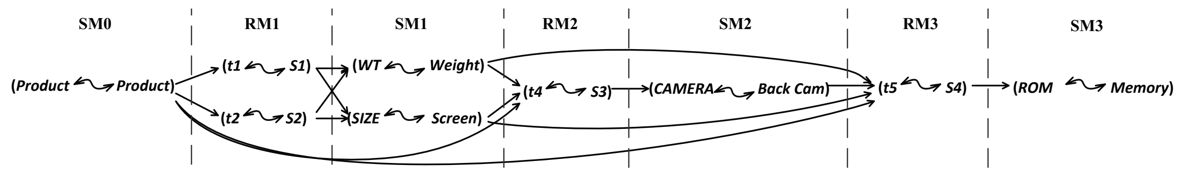


Fig. 2. Example Interaction Workflow of SM and RM for Integrating Tables in Fig. 1

Our main contributions are summarized as follows:

1) We first study the combination and interaction between SM and RM by performing them alternately when integrating multiple data sources. Novel matching rules are proposed as the foundation of the combination.
2) We design a probabilistic model based on the logistic sigmoid function to estimate the Matching Likelihood of a matching pair. Our model can be adopted to the situation with small overhead when many incidents are dependent, since a parameter acts on the logistic sigmoid function to smooth the dependency among attributes.
3) We propose two effective ways to check the correctness of each matching pair. One uses unbiased variance of the similarity between the attribute values pairs of the two records, while the other employs cross-validation to use all matching pairs to validate each other.
4) We greatly reduce the time complexity of the interaction algorithm by around 90% with a special-designed index structure and several optimization techniques proposed based on the index structure.

The rest of the paper is organized as follows: We give an overview of the interaction in Sec. 2. We discuss on the matching likelihood estimation scheme in Sec. 3, and then present how we control the integration quality in Sec. 4. The optimization on efficiency is given in Sec. 5. After reporting the experiments in Sec. 6, the related work is covered in Sec. 7. We conclude in Sec. 8.

2 Interaction Overview

Given two relational tables $T_1 = \{t_1, t_2, ..., t_n\}$ under schema $\mathcal{S}_1 = \{\mathtt{A}_1, \mathtt{A}_2, ..., \mathtt{A}_\mathtt{p}\}$ and $T_2 = \{s_1, s_2, ..., s_m\}$ under schema $\mathcal{S}_2 = \{\mathtt{B}_1, \mathtt{B}_2, ..., \mathtt{B}_\mathtt{q}\}$, where n, m, p, q are positive integers, t_i $(1 \leq i \leq n)$ denotes a record in T_1, s_i $(1 \leq i \leq m)$ denotes a record in T_2, $\mathtt{A}_\mathtt{i}$ $(1 \leq i \leq p)$ denotes an attribute in T_1, and $\mathtt{B}_\mathtt{i}$ $(1 \leq i \leq q)$ denotes an attribute in T_2, assume $\mathcal{S}_1 \cap \mathcal{S}_2 \neq \emptyset$, i.e., the two tables have common attributes, $T_1 \cap T_2 \neq \emptyset$, i.e., the two tables have records referring to the same real-world entity, the two fundamental tasks of Data Integration[1] is to perform **Schema Matching (SM)** between $\mathcal{S}_1$ and $\mathcal{S}_2$, and **Record Matching (RM)** between T_1 and T_2: While the objective of SM is to unify $\mathcal{S}_1$ and $\mathcal{S}_2$ by finding out all pairs of attributes $(\mathtt{A}_\mathtt{i}, \mathtt{B}_\mathtt{j})$ between $\mathcal{S}_1$ and $\mathcal{S}_2$, where each attribute-pair refers to the same property of the records, or denoted as $(\mathtt{A}_\mathtt{i} \leftrightsquigarrow \mathtt{B}_\mathtt{j})$, the object of RM is to find out all pairs of records (t_i, s_j) referring to the same entity, or denoted as $(t_i \leftrightsquigarrow s_j)$, between T_1 and T_2. Here we assume that each attribute in $\mathcal{S}_1$ matches with no more than one attribute in $\mathcal{S}_2$, and vice versa. Also, we assume that each record in T_1 matches with no more than one record in T_2, and vice versa. Besides, we do not consider the situation that an attribute in one table corresponds to the combination of several attributes in the other table.

1. We present our interactive idea between two tables for ease of presentation, and it is extendable to the integration of multiple tables.

The integration quality can be reflected in four dimensions, i.e., *Precision of SM, Recall of SM, Precision of RM*, and *Recall of RM*. In particular, the precision of SM is the percentage of correctly matched attribute-pairs among all matched attribute-pairs while the recall of SM is the percentage of correct matching-attribute-pairs among all matching-attribute-pairs that should be identified between the two tables. Similarly, the precision of RM is the percentage of correct matching-record-pairs among all matching-record-pairs while the recall of RM is the percentage of correct matching-record-pairs among all matching-record-pairs that should be identified between the two tables.

To implement data integration, existing work does SM firstly and RM secondly in one run, but often fails to reach a satisfied integration quality (that could be achieved). In this paper, we work on the interaction between SM and RM in the process of performing them alternately, with the expectation that the interaction can provide us further chances to improve the integration quality. For easier presentation, we use **IntSRM** to denote the interaction process. In the following, we first use an example to demonstrate how we perform SM and RM alternately in Sec. 2.1, and then formally state the problems in Sec. 2.2.

2.1 Basic Interaction Scheme

The basic interaction process is described as follows: starting with seed linked attribute-pairs, we perform SM and RM in turn to detect linked attributes or linked records iteratively until no more links can be detected. While every RM step identifies more linked entities to help the next SM step find more not yet linked attribute-pairs, every SM step detects more linked attribute-pairs to benefit the followed RM in finding more not yet linked record-pairs.

Briefly, a RM matching is made based on temporary attribute-pairs that are already linked, and an SM matching is made based on temporary instance pairs that are already linked. More specifically, we describe the two rules below:

Rule 1: (**Linked-Attributes-based RM**). Given linked attribute-pairs $\{(\mathtt{A}_{l1} \leftrightsquigarrow \mathtt{B}_{l1}), (\mathtt{A}_{l2} \leftrightsquigarrow \mathtt{B}_{l2}), ..., (\mathtt{A}_{ll} \leftrightsquigarrow \mathtt{B}_{ll})\}$ between $\mathcal{S}_1$ and $\mathcal{S}_2$ in integrating T_1 with Schema $\mathcal{S}_1$ and T_2 with schema $\mathcal{S}_2$, we say whether a record-pair $t \in T_1$ and $s \in T_2$ will be linked (temporarily) in the next RM step is determined jointly by the similarity between the pair $(t[\mathtt{A}_\mathtt{i}]$, $s[\mathtt{B}_\mathtt{i}])$, and the ability of $(\mathtt{A}_\mathtt{i}, \mathtt{B}_\mathtt{i})$ in recognizing matching-record-pairs, where $i \in \{l1, l2, ..., ll\}$.

Basically, the more attribute values the two records share under matching-attribute-pairs and the stronger the ability of these matched attribute-pairs in recognizing matching-record-pairs, the more likely that the two records can be a matching-record-pair referring to the same entity. For example in Fig. 1, given linked attributes ($\texttt{Weight} \leftrightsquigarrow \texttt{WT}$) and ($\texttt{SIZE} \leftrightsquigarrow \texttt{Screen}$). Assume the two attributes can effectively differentiate a record from the others, we may produce a record-pair ($t_4 \leftrightsquigarrow s_3$), since it shares the same "Weight or WT" and "SIZE or Screen" values.

Rule 2: (**Linked-Records-based SM**). Given linked record-pairs $\{(t_{l1} \leftrightsquigarrow s_{l1}), (t_{l2} \leftrightsquigarrow s_{l2}), ..., (t_{ll} \leftrightsquigarrow s_{ll})\}$ between T_1 and T_2 in integrating T_1 with Schema $\mathcal{S}_1$ and T_2 with schema $\mathcal{S}_2$, we say whether an attribute-pair $\texttt{A} \in \mathcal{S}_1$ and $\texttt{B} \in \mathcal{S}_2$ will be linked (temporarily) in the next SM step is determined jointly by the similarity between each ($t_i[\texttt{A}]$, $s_i[\texttt{B}]$) pair, where $i \in \{l1, l2, ..., ll\}$.

The more record-pairs support the matching of the two attributes, the more likely that they should be matched pairs. If this situation is observed with several pairs of linked records between two tables, there will be a high matching likelihood that the two attributes actually refer to the same one. For example in the two tables in Fig. 1, Weight and WT might be linked since they share the same attribute values under several record-pairs like (t_1, s_1) and (t_2, s_2).

Based on the two matching rules above, the interaction scheme can be simply represented by a sequence of attribute/record-pair set. Let $\mathcal{P}_i^S = \{(\texttt{A}^{\texttt{i},1} \leftrightsquigarrow \texttt{B}^{\texttt{i},1}), (\texttt{A}^{\texttt{i},2} \leftrightsquigarrow \texttt{B}^{\texttt{i},2}), ...\}$ denote the attribute-pair set identified at the i−th SM step, and $\mathcal{P}_i^R = \{(\texttt{t}^{\texttt{i},1} \leftrightsquigarrow \texttt{s}^{\texttt{i},1}), (\texttt{t}^{\texttt{i},2} \leftrightsquigarrow \texttt{s}^{\texttt{i},2}), ...\}$ denote the record-pair set identified at the i−th RM step, an interaction scheme between T_1 and T_2 can be denoted as

$$Q = \langle \mathcal{P}_0^S, \mathcal{P}_1^R, \mathcal{P}_1^S, \mathcal{P}_2^R, \mathcal{P}_2^S, ..., \mathcal{P}_k^R, \mathcal{P}_k^S, ...\rangle$$

, where $\forall i \neq j, \mathcal{P}_i^S \cap \mathcal{P}_j^S = \mathcal{P}_i^R \cap \mathcal{P}_j^R = \emptyset$.

We now describe a basic interaction workflow between SM and RM with an example scenario in Fig. 2.

Example 1: Initially, we have $\mathcal{P}_0^S = \{(\texttt{Product} \leftrightsquigarrow \texttt{Product})\}$, according to which we can match ($t_1 \leftrightsquigarrow s_1$) and ($t_2 \leftrightsquigarrow s_2$). Then, we find that the two matched records share the same values under (WT, Weight) and (SIZE, Screen). Thus, ($\texttt{WT} \leftrightsquigarrow \texttt{Weight}$) and ($\texttt{SIZE} \leftrightsquigarrow \texttt{Screen}$) can be our newly-linked attribute-pairs. Until now, we would have three attribute-pairs, according to which we can find a new record-pair ($t_4 \leftrightsquigarrow s_3$), given that the three matching-attribute-pairs support this record-pair. Next, since $t_4[\texttt{CAMERA}]$ equals to $s_3[\texttt{BackCam}]$ rather than $s_3[\texttt{FontCam}]$, we may have ($\texttt{CAMERA} \leftrightsquigarrow \texttt{BackCam}$). We continue with RM and SM alternatively in this way to have ($t_5 \leftrightsquigarrow s_4$) and ($\texttt{ROM} \leftrightsquigarrow \texttt{Memory}$).

2.2 Problems Statement

There are several crucial issues in the interaction workflow. Firstly, the way of estimating the matching likelihood of an attribute-pair (or a record-pair) is the key factor to ensure the matching quality. As we mentioned in Rule 1, the matching likelihood between two records depends on two aspects, i.e., the number of linked attribute-pairs that support the matching, and the ability of the linked attribute-pair in recognizing matching-record-pairs. Therefore, the matching likelihood issue can be resolved by one sub-task of estimating the ability of the linked attribute-pairs in recognizing records referring to the same entity, and the other more challenging sub-task: how to combine the contributions from multiple attribute-pairs to the calculation of matching likelihood of the two concerned records. We will give our solution to this problem in Sec. 3.

Second, "semantic drift" problem should be controlled for preventing the mistake magnification from an SM (or RM) step to the following RM (or SM). The linking decisions made at each SM (or RM) step based on temporary RM (or SM) results should be validated. We will discuss the details of matching quality control in Sec. 4.

Last but not the least, the large overhead produced by comparing a large number of value pairs should be reduced. Our analysis shows that, without any optimizations, the computational complexity of the interactive algorithm can be as high as $O(min(p, q)mn)$, where m and n are the number of records in the two tables for integration respectively, and p and q are the number of attributes in the two tables respectively. Sec. 5 will introduce how to reduce the computational cost.

3 Matching Likelihood Estimation

We present the models for estimating the matching likelihood for record-pairs and attribute-pairs respectively.

3.1 Matching Likelihood Estimation for RM

We provide a way to estimate the ability of the linked attribute-pair in recognizing records referring to the same entity, and then discuss how to combine the contributions from multiple attributes to the matching likelihood of the two concerned records.

1) IdC Score. We call the ability of an attribute A in differentiating a record from the other records as the *Identification Confidence* of the attribute A, denoted as $IdC(A)$. Basically, the IdC of an attribute can be learned from a large training data set with a probabilistic model, where the training data consisted of a set of labeled matched pairs denoted as Pos_T and a set of labelled unmatched pairs denoted as Neg_T. More specifically, we estimate the IdC score of an attribute A based on a labeled training set from its table as:

$$IdC(A) = \frac{Pos_T(A)}{Pos_T(A) + Neg_T(A)} \tag{1}$$

where $Pos_T(A)$ is the number of record-pairs matched on attribute A among all labeled matched pairs in Pos_T, and $Neg_T(A)$ is the number of record-pairs matched on attribute A among all labeled unmatched pairs in Neg_T.

Given that two attributes, A from one table and B from the other, are matched, the IdC of the attribute-pair $A \leftrightsquigarrow B$ denoted by $IdC(A \leftrightsquigarrow B)$ can be estimated as:

$$IdC(A \leftrightsquigarrow B) = \sqrt{\frac{[IdC(A)]^2 + [IdC(B)]^2}{2}} \tag{2}$$

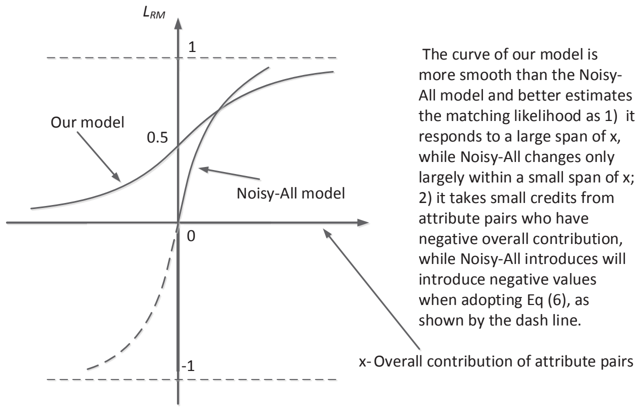


Fig. 3. Comparison between Models for Illustration

2) Contributions Combination. Let $\{\mathbb{A} \leftrightsquigarrow \mathbb{B}\} = \{(A_1 \leftrightsquigarrow B_1), (A_2 \leftrightsquigarrow B_2), ..., (A_n \leftrightsquigarrow B_n)\}$ denote the set of linked attribute-pairs, we discuss on how to estimate the matching likelihood of two records, say $t \in T_1$ and $s \in T_2$, based on: (1) the similarity of the values under linked attribute-pairs in s and t, denoted as $sim(s[A_i], t[B_i])$ $(1 \le i \le n)$, and (2) the IdC score of every linked attribute-pair $(A_i \leftrightsquigarrow B_i)$.

Two existing models are potentially usable for estimating the matching likelihood of two records, but have limitations discussed below.

(1) *Inclusion-Exclusion:* A classical way based on the inclusion-exclusion principle [6] calculates the matching likelihood of (t, s) as: $\mathcal{L}_{RM}(t,s) = \sum_{k=1}^{n}(-1)^{k+1}P(\Psi(k,n,t,s))$, where $\Psi(k,n,t,s)$ is a set containing all the k-size combinations generated from the set of linked attribute-pairs, and $P(\cdot)$ is the likelihood score of a set as: $P(C_1 \cap \cdots \cap C_i) = \mu_{(C_1 \cap \cdots \cap C_i)} \cdot \prod_{j=1}^{i} IdC(A_j \leftrightsquigarrow B_j) \cdot sim(A_j, B_j) \cdot sim(t[A_j], s[B_j])$, where $C_j = A_j \leftrightsquigarrow B_j$ and $\mu_{(C_1 \cap \cdots \cap C_i)}$ is the dependency factor of $\{C_1, ..., C_i\}$, which, however, needs to be estimated in advance. Although this model can accurately estimate the matching likelihood between two records by calculating the dependencies between attributes comprehensively, the computation cost grows exponentially with n.

(2) *Noisy-All:* The Noisy-All model [1], [25] is another popular model that calculates the matching likelihood as: $\mathcal{L}_{RM}(t,s) = 1 - \prod_{(A \leftrightsquigarrow B) \in \{\mathbb{A} \leftrightsquigarrow \mathbb{B}\}} sim(t[A], s[B]) \cdot (1 - IdC(A \leftrightsquigarrow B))$, which indicates that the likelihood estimation is simplified by assuming all attributes are independent. However, the dependency between attributes should not be neglected in real practice. In addition, the noisy-all model has the so called *Accumulative-Error* problem: the matching likelihood of record-pairs increases with the number of matched attributes. That is to say, two records can be wrongly estimated to have high matching likelihood even the similarity of values under their linked attribute-pairs is low, just because the number of matched attributes is high. For example, even $sim(t[A_i], s[B_i]) = 0.4$, $IdC(\ A_i \leftrightsquigarrow B_i)) = 0.5$ $(1 \le i \le n)$ but $n = 10$, then $\mathcal{L}_{RM}(t,s) = 0.9$.

We propose a new model to take the advantages of both the two models but addressing their limitations. Firstly, instead of the comprehensive way to calculate the dependencies among the attributes, we simplify the function into a linear one with the logistic sigmoid function [5] and then rely on only one parameter to control the influence among the attributes, i.e. we use the logistic sigmoid function to smooth the influence among the attributes for matching records in an explicit way. Besides, to overcome the Accumulative-Error problem, we define a contribution function and employ a logarithmic function to map the value from $[0,1]$ into $[0,+\infty)$.

To achieve this, we first assume that all the IdC of attributes are independent such that a linear model (similar to Noisy-All) can be used to calculate the matching likelihood, and then we compensate for the dependence between attributes in the model by introducing a damping factor. The matching likelihood of the record-pair (t,s) is then calculated as:

$$\mathcal{L}_{RM}(t,s) = \frac{1}{1 + e^{-\lambda \cdot S(\mathbb{A} \leftrightsquigarrow \mathbb{B}, t, s)}} \tag{3}$$

where λ is the damping factor to compensate for the dependence between attributes (the parameter can be tuned on a validation dataset), and $S(\mathbb{A} \leftrightsquigarrow \mathbb{B}, t, s)$ is the overall contribution score of the set of linked attribute-pairs $\{\mathbb{A} \leftrightsquigarrow \mathbb{B}\}$ to the matching of the record-pair (t,s), computed as:

$$S(\mathbb{A} \leftrightsquigarrow \mathbb{B}, t, s) = \sum_{(A \leftrightsquigarrow B) \in \{\mathbb{A} \leftrightsquigarrow \mathbb{B}\}} \phi(A,B) \cdot ctr(t[A], s[B]) \tag{4}$$

where $\phi(A,B) \in [0,+\infty)$ employs a logarithmic function to map the value between 0 to 1 into the whole real axis as:

$$\phi(A,B) = -ln(1 - \mathcal{L}_{SM}(A,B) \cdot IdC(A \leftrightsquigarrow B)) \tag{5}$$

where $\mathcal{L}_{SM}(A,B)$ is the matching likelihood of the two attributes A and B. Finally, $ctr(t[A], s[B])$ is the contribution of the similarity between two values $t[A]$ and $s[B]$

$$ctr(t[A], s[B]) = \begin{cases} 0, & \text{if } t[A] = null \text{ or } s[B] = null \\ \frac{sim(t[A],s[B]) - \theta}{1-\theta}, & \text{if } sim(t[A], s[B]) \ge \theta \\ \frac{sim(t[A],s[B]) - \theta}{\theta}, & \text{if } sim(t[A], s[B]) < \theta \end{cases} \tag{6}$$

where $sim(t[A], s[B])$ is the similarity between $t[A]$ and $s[B]$ measured by string similarity function such as edit distance, and θ is an expert-defined tipping point to decide whether this value pair produces positive or negative contributions. The contribution defined in Eq. (6) resolves the issue of accumulative error when using $sim(t[A], s[B])$ directly as contribution. That is, a large but wrong$S(\mathbb{A} \leftrightsquigarrow \mathbb{B}, t, s)$ score can be obtained by accumulating small similarity $sim(t[A], s[B])$ of a large number of attributes $(A \leftrightsquigarrow B)$ in $\{\mathbb{A} \leftrightsquigarrow \mathbb{B}\}$.

To summarize, the proposed model has the following two advantages: (1) it can be adopted to the situation when the contributions of the linked attribute-pairs are not independent with a low overhead; (2) it solves the Accumulative-Error problem. As can be observed in the comparison of $\mathcal{L}_{RM}$ curve between the Noisy-All model and our model in Fig. 3, the Noise-all $\mathcal{L}_{RM}$ curve is abrupt and changes largely when x is small, while our $\mathcal{L}_{RM}$ curve is smooth and responds to a large span of x. In addition, our model does not introduce negative values like Noisy-All model when adopting Eq. (6).

3.2 Matching Likelihood Estimation for SM

We now present how to estimate the matching likelihood between two attributes $A \in S_1$ and $B \in S_2$. Let $\{\mathbb{T}_t \leftrightsquigarrow \mathbb{T}_s\} = \{(t_1 \leftrightsquigarrow s_1), (t_2 \leftrightsquigarrow s_2), ..., (t_n \leftrightsquigarrow s_n)\}$ denote the set of linked record-pairs so far, we adopt the same model to calculate the matching likelihood between two attributes, where the damping factor is not required since the records are usually independent with each other. Thus we have:

$$\mathcal{L}_{SM}(A, B) = \frac{1}{1 + e^{-S(\mathbb{T}_t \leftrightsquigarrow \mathbb{T}_s, A, B)}} \tag{7}$$

where $S(\mathbb{T}_t \leftrightsquigarrow \mathbb{T}_s, A, B)$ is the overall contribution of the set of linked record-pairs $\mathbb{T}_t \leftrightsquigarrow \mathbb{T}_s$ to the matching of the attribute-pair A and B, which can be computed as:

$$S(\mathbb{T}_t \leftrightsquigarrow \mathbb{T}_s, A, B) = \alpha \cdot \sum_{(t \leftrightsquigarrow s) \in \{\mathbb{T}_t \leftrightsquigarrow \mathbb{T}_s\}} \varphi(t, s) \cdot ctr(t[A], s[B]) \tag{8}$$

where α is a parameter to control the contribution score about the related record-pairs, and $\varphi(t, s) = -ln(1 - \mathcal{L}_{RM}(t, s))$ is also a logarithmic function to map the value from $[0, 1]$ into $[0, +\infty)$.

In the interactive process, our strategy is to keep a high precision with a strict constraint and improve the recall step by step. That is, at each RM or SM step, the $\mathcal{L}_{RM}$ or $\mathcal{L}_{SM}$ for a number of candidate record-pairs or attribute-pairs will be calculated, and only those satisfying predefined threshold will be taken as linked pairs for further interactions.

Example 2: Let $(\texttt{Product} \leftrightsquigarrow \texttt{Product})$ be the seed attribute-pair to initiate the interaction between the two tables as depicted in Fig. 1. Assume the $IdC(\texttt{Product} \leftrightsquigarrow \texttt{Product})$ is 0.97, and let 0.7 be the threshold to the matching likelihood of both SM and RM for illustration, $\theta = 0.1$ in Equation 6.

At the first RM step, we have $(t_1 \leftrightsquigarrow s_1)$ and $(t_2 \leftrightsquigarrow s_2)$ given that $\mathcal{L}_{RM}(t_1, s_1) = \frac{1}{1+e^{-3.5066}} = 0.97 > 0.7$ and $\mathcal{L}_{RM}(t_2, s_2) = \frac{1}{1+e^{-0.8723}} = 0.705 > 0.7$.

At the first SM step, we have $(Weight \leftrightsquigarrow WT)$ since they share the same two values in the two pairs of linked records such that the matching likelihood can be calculated as: $\mathcal{L}_{SM}(\texttt{Weight}, \texttt{WT}) = \frac{1}{1+e^{-[-ln(1-0.97)-ln(1-0.705)]}} = 0.991$.

4 Quality Control

Although the interaction scheme introduced above tends to select more promising matching pairs for iterative interaction to keep a high matching precision, still, the linking decisions made during the interaction may involve errors, since: (1) the decisions made at each SM (or RM) step based on temporary RM (or SM) results should be selected for filtering out the unreliable ones; and (2) once a mistaken matching happens at an iteration, more mistaken matchings might be introduced in later iterations, i.e., the "semantic drift" problem happens.

In the following, we introduce how to control semantic drift, and how to iteratively update the linking pairs in the interaction for higher-quality linking results.

4.1 Semantic Drift Control

We validate the newly-linked records and newly-linked attributes separately to prevent semantic drift from happening. After each RM step, we identify "risky" record-pairs by checking the unbiased variance of the similarity between their value pairs under various attribute-pairs, while after each SM step, we identify "outlier" attribute-pairs by applying cross-validation techniques to validate all the linked attributes.

1) Unbiased Variance Checking for "Risky" Records. Intuitively, two records having similar values under more attribute-pairs tend to have higher matching likelihood. However, relying on the matching likelihood only cannot effectively differentiate high-quality record-pairs from "risky" ones due to the ubiquitously existing errors, various formats and so on. To identify the risky pairs, we measure the degree of instability of each record-pair by calculating the variance of the similarity between their attribute values under various attribute-pairs. More specifically, for a record-pair $(t \leftrightsquigarrow s)$, we get the degree of instability under different attribute-pairs by calculating the Unbiased Variance [33] (short for UV) of the similarity between their value pairs under various attribute-pairs as:

$$UV(t, s) = \frac{1}{m-1} \sum_{i=1}^{m} [sim(t[A_i], s[B_i]) - \overline{sim}(A_i, B_i)]^2 \tag{9}$$

where $\overline{sim}(A_i, B_i)$ is the average similarity of all the attribute values under the attribute-pair (A_i, B_i) and m is the number of linked attribute-pairs. We remove the record-pairs whose UV values are larger than a user-defined threshold, which is set as the average UV values of all the record-pairs to be checked in a data set by default. Note that $\overline{sim}(A, B)$ will change after removing some record-pairs, but we can obtain the final fixed record-pairs after several iterations. This process is similar to the k-means clustering [27] whose convergence property has been proved well. The complexity of the UV-checking method is $O(pq^2)$, where p is the number of linked attribute-pairs and q is the number of linked record-pairs to be checked.

This UV-checking is crucial to matching quality control, since it not only guarantees the high-quality record-pairs for next SM step, but also helps us identify different presentations of the same entity, for example "in" is an abbreviation of "inch" in the example in Fig. 1, since every pair of linked attribute values shares the same distance "ch".

2) Cross-Validation for Detecting "Outlier" Attributes. We adopt the cross-validation techniques to validate the linked attribute-pairs. Intuitively, if a pair of linked attributes does not consist with the other attribute-pairs, it is very likely a risky "outlier" that should be dropped.

Specifically, we denote the set of all attribute-pairs as $\mathcal{P}$, and partition $\mathcal{P}$ into k disjoint subsets denoted as

$\mathcal{P} = \{P_1, P_2, \cdots, P_k\}$ and let the number of attribute-pairs in each subset in $\mathcal{P}$ be $\frac{|\mathcal{P}|}{k}$ (We initialize k as the number of attribute pairs at one step. If $|\mathcal{P}|$ is too large, we can repartition them to reduce the computational cost). And we denote $\mathcal{P} - P_i$ the attribute-pair set which excludes P_i from $\mathcal{P}$. At each verification, we take $\mathcal{P} - P_i$ as a training set and P_i as a validation set. We exploit $\mathcal{P} - P_i$ to infer record-pairs, and then we compute the matching likelihood of attribute-pairs in P_i according to the inferred records pairs. Typically, we let the number of attribute-pairs in P_i be one (considering not too many attribute-pairs) and denote it as $(A \leftrightsquigarrow B)$. We adopt a linear loss function F in regression $F(S, ((A, B), \eta))) = S_{(t \leftrightsquigarrow s) \in \mathcal{R}}(t[A], s[B]) - \eta$, where $\mathcal{R}$ is the record-pairs inferred by $\mathcal{P} - P_i$, S is a similarity function and η is a qualified threshold. And we define the error $\{0, 1\}$-loss in the judgement $F(S, ((A, B), \eta))) = 1$, if $S_{(t \leftrightsquigarrow s) \in \mathcal{R}}(t[A], s[B]) < \eta$. Then we calculate the errors (short for Er) of the attribute-pair (A, B) as follows:

$$Er(A, B) = \frac{1}{|\mathcal{R}|} \sum_{(t \leftrightsquigarrow s) \in \mathcal{R}} F(S, ((A, B), \eta)) \tag{10}$$

We repeat this verification process k times with different validation sets and then we obtain the Er values of each attribute-pair. We then drop the attribute-pairs whose Er is lower than a predefined threshold.

4.2 Iterative Updating and Adjusting

As the interaction proceeds, more and more attribute-pairs and record-pairs are linked. After each iteration, the matching likelihood we calculated in previous iterations need to be updated, according to which we also need to adjust the attribute and record-pairs that are already linked.

The relationship between the record-pairs and attribute-pairs is mutually reinforced. We can use a bi-graph to illustrate the relationship between the record-pairs and attribute-pairs, where the weight on an edge means the contribution score of a record-pair or an attribute-pair. Typically, the weight on an edge which points to an attribute-pair (A, B) from a record-pair (t, s) is $\varphi(t, s) \cdot ctr(t[A], s[B])$, and the weight on an edge which points to a record-pair (t, s) from an attribute-pair (A, B) is $\phi(A, B) \cdot ctr(t[A], s[B])$. Then the matching likelihood of a record-pair is to apply the function $f(x) = \frac{1}{1+e^{-\lambda \cdot x}}$ to the summation of all the weights on the edges pointed to the record-pair itself. And this calculation method can be also adapted to an attribute-pair. Denote the matching likelihood of record and attribute-pairs respectively as a vector $\vec{r}$ and $\vec{a}$. We compute $\vec{r}$ and $\vec{a}$ alternatively until they all reach a stable state. We prove that the vectors $\vec{r}$ and $\vec{a}$ will converge to a constant vector. Formally, we have the following conclusion:

Theorem 1: The iterative algorithm is convergent. That is, the matched record and attribute-pairs will be uniquely determined finally.

Proof: Please see the appendix.[2] □

2. Due to the limitation of space, we put our appendix online which can be accessed through http://ada.suda.edu.cn/Uploads/File/201512/04/1449212591654/proof.pdf

4.3 Algorithm Analysis and Other Issues

One issue of the algorithm is how to set the thresholds for the matching likelihood of record-pairs and attribute-pairs respectively. Although we set the thresholds empirically for different data sets, the experiment results show that the thresholds can be set to values in a quite large range where the performance of our algorithm has no significant variation. The reason is that the initial threshold setting will not affect the number of matched record or attribute-pairs at the later steps. Specifically, if the threshold is too high, we just need more iterations to find more matched record or attribute-pairs. If the threshold is too low, the quality control process will gradually amend them as the interaction algorithm goes. Generally, the thresholds should be larger than 0.5, which means one record or attribute-pair is more likely to be a correct one than a wrong one. In our experiments, we set both of them to 0.6. Also, the threshold setting for Er is similar to the threshold setting to matching likelihood.

The time complexity of the algorithm is mainly decided by both the time complexity of SM steps and that of RM steps. The cost of SM mainly depends on the comparison times between two subsets of instances that are used for schema matching, while the cost of RM mainly depends on the number of records and the size of schemas in the two tables. Formally, we analyze the time complexity of our algorithm as follows:

Theorem 2: Given tables $T_1 = \{t_1, t_2, ..., t_n\}$ under schema $S_1 = \{A_1, A_2, ..., A_p\}$, and $T_2 = \{s_1, s_2, ..., s_m\}$ under schema $S_2 = \{B_1, B_2, ..., B_q\}$ for integration. The upper-bound of the time complexity that an interaction scheme for IntSRM can reach is: $O(min(p, q)mn)$, where p and q are the number of records in T_1 and T_2 respectively, m and n are the number of attributes in S_1 and S_2 respectively.

Proof: Please see the appendix. □

5 Optimizations on Efficiency

As can be seen from the proof of Theorem 2, the efficiency bottleneck of the interaction usually lies on the RM step since there are usually a lot more records than attributes in the table. To minimize the number of record-pairs for comparison, some state-of-the-art indexing techniques [8] have been proposed for scalable record linkage and successfully applied on RM based on the key attribute. In this paper, we extend the q-gram index [9], [22], [34] to multiple pairs of attributes scenario, and split potential matched record-pairs between the two tables into (possibly overlapped) blocks so that matching-record-pairs are only identified within every block.

5.1 Indices for SM and RM Interaction

Before introducing how the indices are built on linked attribute-pairs, some definitions and lemmas are given first:

Definition 1: Given a string s, a set of q-grams (q is a constant to denote the length of each gram) can be generated from s as: $Gms(s, q) = \{gm_1, gm_2, ..., gm_{|s|-q+1}\}$, where gm_i consists of the characters from i to $(i+q-1)$ by their natural order in s. Then an **ℓ-length consecutive q-gram sequence** (or **(ℓ,q)-seq** for short) of s can be defined as a string that consisted of a sequence of q-grams consecutively in their natural order in $Gms(s,q)$.

Lemma 1: Let $F(s_1,s_2) = \frac{|Gms(s_1,q)\cap Gms(s_2,q)|}{|Gms(s_1,q)\cup Gms(s_2,q)|+\varepsilon}$, where ε gets very close to 1. Given a q-gram overlap threshold ω, if $F(s_1,s_2)\geq\omega$, then they should share at least one (ℓ,q)-seq, where $\ell\geq q\cdot\lfloor(max(|s_1|,|s_2|)-q+1)\cdot\omega\rfloor$.

Proof: Please see the appendix. □

1) Index Building on a Single Pair of Attributes: Based on the definition and the lemma above, the index building process can be described as follows: Given a pair of linked attributes, for every distinct attribute value s under the linked attributes in either table, we generate all (ℓ,q)-seq from this value, where $\ell\geq q\cdot\lfloor(|s|-q+1)\cdot\omega\rfloor$, where q is a constant to define the length of grams. Based on Lemma 1, each (ℓ,q)-seq of an attribute value s will be taken as a **key index value** for s, according to which s will be indexed into a corresponding block. For instance, assume a product value "huawei", whose bi-gram list is ['hu', 'ua', 'aw', 'we', 'ei']. Let $\omega=0.8$, then $\ell\geq q\cdot\lfloor(|s|-q+1)\cdot\omega\rfloor=8$, thus we will generate six $(\ell,2)$-seq from "huawei", i.e., 'huuaawweei', 'uaawweei', 'huawweei', 'huuaweei', 'huuaawei', 'huuaawwe' as the key index values for "huawei". As a result, "huawei" will be put into the six blocks corresponding to the six key index values.

2) Dynamic Indices Building between the Two tables: It is a dynamic process to build the index on multiple pairs of linked attributes in the interaction process. Initially, we only build the index under the seed attribute-pair (such as (`Product` ⇜ `Product`)). When more linked attributes are identified at each SM step, we build an index under each of these linked attributes, as long as its IdC score is higher than a predefined threshold. For easier presentation, we call the index we build on the two databases for integration as a *Qgram-based Multiple-Line Indices for the Interaction between RM and SM* (or **MLineIndex** for short).

Lemma 2 describes the relationship between the threshold ω we mentioned above and the edit similarity threshold between two attribute values.

Lemma 2: Let ω denote the threshold that works for controlling the generation of key value indexes in building the MLineIndex, if two attribute values s_1 and s_2 are assigned into one block, then their edit similarity must be no less than $\frac{\lfloor(max(|s_1|,|s_2|)-q+1)\cdot\omega\rfloor+q-1}{max(|s_1|,|s_2|)}$, and if s_1 and s_2 are not in the same block, their edit similarity must be no larger than $1-\frac{2-q+max(|s_1|,|s_2|)-\lfloor(max(|s_1|,|s_2|)-q+1)\cdot\omega\rfloor}{q\cdot max(|s_1|,|s_2|)}$.

Proof: Please see the appendix. □

5.2 Greedy RM based on MLineIndex

We now describe a greedy RM algorithm with the MLineIndex at a particular RM step. Assume we already have a set of linked attributes, and every potential matched record-pairs w.r.t. a linked attribute-pair is put into a block and indexed under this linked attribute-pair. For the sake of processing the interaction between SM and RM with the minimum RM comparison times, each RM step only greedily selects one particular block of records for RM comparison from all the blocks indexed by the MLineIndex, which should satisfy the following two conditions: (1) it requires the least RM comparison times; (2) it has a high probability to generate matching record-pairs. More specifically, given a block $Block=(\{LR\},\{RR\})$, where LR is the set of records from one table, and RR is the set of records from the other table, we estimate the priority of doing RM to this block as:

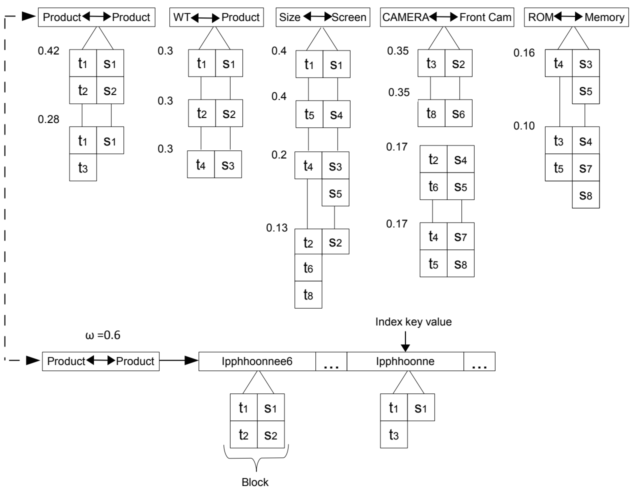


Fig. 4. The MLineIndex built on the Two Example Tables in Figure 1

$$priority(Block)=\frac{IdC(AttrPair_{Block})}{Max(|LR|,|RR|)} \qquad (11)$$

where $AttrPair_{Block}$ is the attribute-pair under which the block is indexed, and $Max(|LR|,|RR|)=\frac{|LR|\times|RR|}{Min(|LR|,|RR|)}$ denotes the average comparison times needed for generating a matching record-pair from $Block$, since the block contains at most $Min(|LR|,|RR|)$ record-pairs. When a block $Block=(\{LR\},\{RR\})$ is selected for RM comparison, for every pair of records between LR and RR, we calculate the similarity between their attribute values under each linked attribute-pairs respectively, and then get their RM matching likelihood according to Eq. (3). We set a maximum number of record-pairs to be used for SM (1000 in our experiments), such that the time consumption of SM steps will not be deferred too much by RM steps.

We observe that the block with higher priority often produces correct record-pairs based on MLineIndex, although we have not considered the similarity of attribute values in Eq. (11). The reason is that the high priority of a block often means there are less records in the block, but they often have quite a lot same q-grams i.e. their similarities are often very large.

Another advantage of this strategy is that we can rapidly identify the matched record-pairs, since many attribute-pairs have been identified. Such that, for the rest records, they can reduce many unnecessary comparisons due to the less and less number of records.

How to set the threshold ω_i for each attribute-pair A_i is the key problem for the indices-based greedy RM algorithm. An optimal setting is demanded to make a trade-off between the precision of matching results and the efficiency. In the following, we discuss the setting of ω_i for maximizing the precision of matching results while reducing the time complexity as much as possible.

5.2.1 Setting the Parameter ω_i

Computing time cost is mainly affected by the number of blocks and the number of record-pairs in blocks. Generally, to reduce the number of blocks and the record-pairs in blocks, it is preferred to have a relatively high threshold ω to those attribute-pairs with a relatively low IdC. As the number of q-grams generated from an attribute values grows exponentially as the threshold ω decreases [8], we employ a convex decreasing function to set the threshold for an attribute-pair $A \leftrightsquigarrow B$ according to its IdC. In particular, let $\{A_1 \leftrightsquigarrow B_1, A_2 \leftrightsquigarrow B_2, ..., A_m \leftrightsquigarrow B_m\}$ denote the set of matched attribute-pairs sorted in a non-decreasing order of their IdC scores, and let ω_i $(1 \leq i \leq m)$ denote the q-gram overlap threshold for the attribute-pair $A_i \leftrightsquigarrow B_i$, we choose ω_1 as a base value (it will be set automatically in later discussions), and then obtain ω_i with the following equation:

$$\omega_i = e^{\frac{IdC(A_i \leftrightsquigarrow B_i)\cdot ln\omega_1}{IdC(A_1 \leftrightsquigarrow B_1)}} \tag{12}$$

and we will show how to set ω_1 below.

According to Lemma 2, if two records t and s are not in a same block pair under the index of $A_i \leftrightsquigarrow B_i$, the similarity between their attribute values under $A_i \leftrightsquigarrow B_i$ has a upper bound denoted by sim_i^u. Thus we can rewrite sim_i^u as the following equation combining with Eq. (12):

$$sim_i^u = 1 - \frac{2 - q + M_i - \lfloor (M_i - q + 1)\cdot \omega_1^{\frac{IdC(A_i \leftrightsquigarrow B_i)}{IdC(A_1 \leftrightsquigarrow B_1)}} \rfloor}{q \cdot M_i} \tag{13}$$

where M_i is the maximal length of values under $A_i \leftrightsquigarrow B_i$. Thus if one record-pair (t, s) is not in any block pair of all the indices, we can obtain its upper bound of matching likelihood as follows:

$$\mathcal{L}_{RM}^U(t,s) = \frac{1}{1 + e^{-\lambda \cdot \sum_{i=1}^m -ln(1-IdC(A_i \leftrightsquigarrow B_i))\cdot ctr(t[A_i],s[B_i])}} \tag{14}$$

Recall Eq. (6), since $sim(t[A_i], s[B_i])$ is larger than θ, $ctr(t[A_i], s[B_i])$ here can be substituted with $\frac{sim_i^u - \theta}{1-\theta}$. Given a quality threshold τ_p^R to matched record-pairs, and let $\mathcal{L}_{RM}^U(t,s) = \tau_p^R$, we can derive the value of ω_1 by replacing sim_i^u with Eq. (12), and ω_1 will satisfy the following equation.

$$\frac{1}{1 + e^{-\gamma \cdot \sum_{i=1}^m -ln(1-IdC(A_i \leftrightsquigarrow B_i))\cdot (sim_i^u - \theta)/(1-\theta)}} = \tau_p^R \tag{15}$$

Eq. (15) is an equation with only one unknown parameter ω_1. Therefore, ω_1 can be derived by analysing Eq. (15). And all the other thresholds about ω_i can be derived according to Eq. (12).

Theorem 3: By setting w_1 which satisfies Eq. (15) and w_i $(1 \leq i \leq m)$ according to Eq. (12), all possible matched pairs (whose matching likelihood is larger than τ_p^R) can be covered by the minimum number of blocks under linked attribute-pairs.

Proof: Please see the appendix. □

5.2.2 Bounding for RM Step

Although setting thresholds can prune a large percent of unmatched record-pairs for comparison, there are still many unmatched record-pairs waiting to be filtered in the blocks. Here we employ a bounding-based strategy to further identify unmatched record-pairs from matched ones.

For a record-pair (t, s) in a block under a set of attribute-pairs $\{A_i \leftrightsquigarrow B_i\}$, where $1 \leq i \leq m$, it satisfies $\frac{\lfloor (max(|s_1|,|s_2|)-q+1)\cdot\omega\rfloor + q - 1}{max(|s_1|,|s_2|)} \leq sim(t[A_i], s[B_i]) \leq 1$ according to Lemma 2. Then the lower bound of the matching likelihood $\mathcal{L}_{RM}^L(t,s)$ between t and s can be calculated with the lower bound of $sim(t[A_i], s[B_i])$, while the upper bound of the matching likelihood between t and s is:

$$\mathcal{L}_{RM}^U(t,s) = \frac{1}{1 + e^{-\lambda\cdot\sum_{i=1}^m -ln(1-IdC(A_i \leftrightsquigarrow B_i))}} \tag{16}$$

If $\mathcal{L}_{RM}^L(t,s) > \tau_p^R$, then (t, s) will be a candidate record-pair, and if $\mathcal{L}_{RM}^U(t,s) < \tau_p^R$, then (t, s) will be pruned directly. In order to tighten the bounds as early as possible, we compute the similarity of the attribute values under the attribute-pair with higher IdC in priority, and then the upper-bound and lower-bound of the likelihood can be updated correspondingly.

After all, we will analyze the time complexity of this greedy and bounded interaction algorithm based on the MLineIndex in the appendix.

6 Experiments

6.1 Data Sets and Metrics

We conduct experiments on 2 real and 1 synthetic data sets:

- *Mobile:* We collect cellphones on sale from Tmall[3] and PConline[4] respectively. The Tmall table contains 40k tuples under 53 attributes, while the PConline table contains 56k tuples under 46 attributes. The two tables share 3.8k records and 38 common attributes including *Release Date, Operation System, RAM, Screen Size, Type* etc.
- *Camera:* We collect digital cameras on sale from Yesky[5] and PConline respectively. The Yesky table contains 25k tuples under 50 attributes, while the PConline table contains 34k tuples under 44 attributes. The two tables share 25k records and 31 common attributes including *Type, Pixels, Panel, Wifi, Manufacturer* etc.
- *Synthetic:* We also generate two synthetic tables sharing 100k tuples and 60 common attributes and use certain rules to let the distribution of data close to real data sets. For instance, the similarities between

3. https://www.tmall.com
4. http://www.pconline.com.cn
5. http://www.yesky.com

attribute values in linked record-pairs are uniformly distributed between 0 and 1. Note that since there are some missing attribute values in the two real data sets, we also generate a random number of missing values under each non-key attribute for the synthetic data set.

Metrics: We evaluate the effectiveness of the integration methods in four dimensions, i.e., **Precision of SM**, **Recall of SM**, **Precision of RM**, and **Recall of RM**. Also, the F_1**-score** of SM and RM are also concerned. We use **Time Cost** to evaluate the efficiency of a method.

6.2 Integration Quality Comparison

We compare the integration quality of our method (short for **IntSRM**) with several state-of-the-art methods on the three data sets. For a fair comparison, for each method testing on every dataset, we tuned its setting to make the method reach the best performance on that specific dataset.

a) Name-based SM (**NBSM**): This SM method uses the edit similarity between the attribute names [20] for SM.
b) Value-based SM (**VBSM**): This SM method uses the overlap between the selected subset of attribute values under the two attributes [19] for SM, where each selected subset contains the top-k highest frequency attribute values under the attribute.
c) Linkage-Points-based SM (**LPSM**): This is a state-of-the-art instance-based SM method proposed in [23], which treats the matching function as a black-box and uses specific measures to have reliable SM results from the overlapping instances.
d) Key-based RM (**Key**): This RM method inspects the string similarities between key attribute values [4] only for RM, which also uses q-gram together with inverted index [31], as well as prefix-based pruning [32] and batch-based matching [7], for improving efficiency.
e) Key+Non-key RM (**NokeaRM**): This is a state-of-the-art RM methods using both key and non-key attributes for RM [35], [11]. Briefly, it builds a probabilistic rule-based decision tree based on all attributes according to the ability of each attribute in identifying matching records and the ability in identifying un-matching records, and then relys on this decision tree to make RM decisions. To further improve its efficiency, q-gram-based blocking techniques are used: we generate q-grams for each value, and only those satisfying the minimum q-gram overlap will be compared.

In the following, we first compare the integration quality of RM with previous methods, and then that of SM with previous methods. Since the overlap ratio of the records between two tables has a great influence on the integration quality, we conduct our comparison experiments at various overlap ratios (from 10% to 90%) on the three data sets.

(1) F_1 **Comparison for RM:** As demonstrated in Fig. 5(a)(b)(c), pervious RM methods work poorly (with F_1-score around 0.5-0.6 for NokeaRM and F_1 around 0.2-0.4 for Key) when the overlap ratio is low (such as 10%, 30%), while IntSRM can reach F_1-score as high as 0.6-0.8 on all the three data collections. As the overlap ratio increases, the integration quality of all methods increases gradually. But always, our method IntSRM reaches 20% higher F_1 than the other methods.

(2) F_1 **Comparison for SM:** Similar comparison results can be observed for SM. As demonstrated in Fig. 5(d)(e)(f), pervious SM methods work poorly (with F_1-score less than 0.6) when the overlap ratio is low (such as 10%, 30%), while IntSRM can reach 0.7-0.8 on all the three data collections. As the overlap ratio increases, the integration quality of all methods increases gradually. But always, our method IntSRM reaches about 15% higher F_1-score than the other methods.

(3) PR (Precision & Recall) Comparison for SM and RM Respectively: For more comprehensive comparison, we also draw the Precision-recall graphs for RM and SM comparison by setting the overlap ratio 70% on all the three data collections. As shown in Fig. 6, basically, our method IntSRM can always reach the best performance on either precision or recall over the three data collections.

6.3 Efficiency Comparison

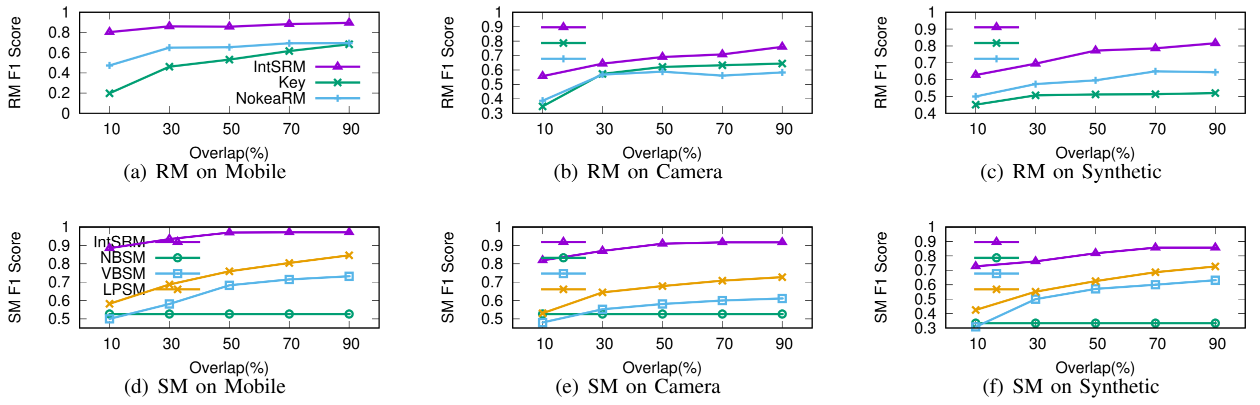
We compare the efficiency of our method with previous methods at the setting of the overlap ratio 70%. For fair comparison, we also compare the time cost of SM or RM with previous SM or RM methods separately. For IntSRM, the time cost of SM includes not only the time cost of all SM steps, but also the time cost of all the RM steps that processed before the last SM step. The time cost of RM for IntSRM includes the time cost of all SM and RM steps.

(1) Time Cost for RM: As described in Fig. 7(a)(b)(c), the Key method uses much less time than NokeaRM and IntSRM since the Key method uses the key attribute only for RM. Compared with NokeaRM, IntSRM uses a bit more time since NokeaRM employs a special decision tree to help prune a large percentage of record-pairs for comparison, which on the one hand, greatly improves the efficiency, but on the other hand, hurts the precision and recall as reported in the experiments in the last subsection. Overall, the time cost spend in RM by IntSRM is acceptable as we greatly improve the precision and recall of the integration.

(2) Time Cost for SM: As described in Fig. 7(e)(f)(g), VBSM uses the least time than both LPSM and IntSRM. The time cost of IntSRM is more than LPSM since the IntSRM spends more time on finding record-pairs under more attribute-pairs than LPSM. But the time cost of IntSRM is acceptable in practice since we can greatly improve the precision and recall.

6.4 Quality Improvement Evaluation

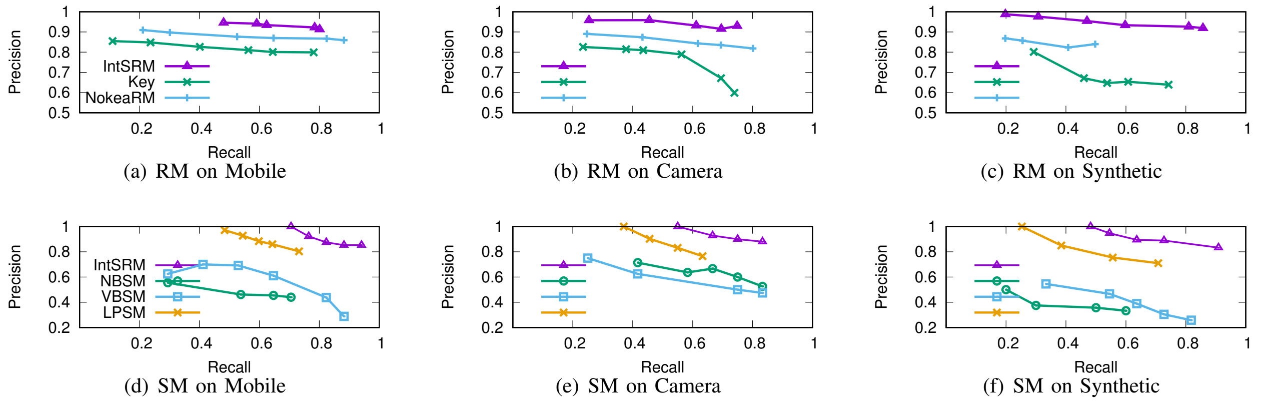
We now evaluate the effectiveness of our proposed techniques for improving the quality of the integration results on the two real data sets. For differentiation, we call the interaction without any quality control techniques as the **Baseline**, the one with validating new-linked records as **UV-Check**, the one with validating new-linked attributes as **Cross-Validate**, and the one with both the two techniques as **UV+Cross**.



(a) RM on Mobile (b) RM on Camera (c) RM on Synthetic

(d) SM on Mobile (e) SM on Camera (f) SM on Synthetic

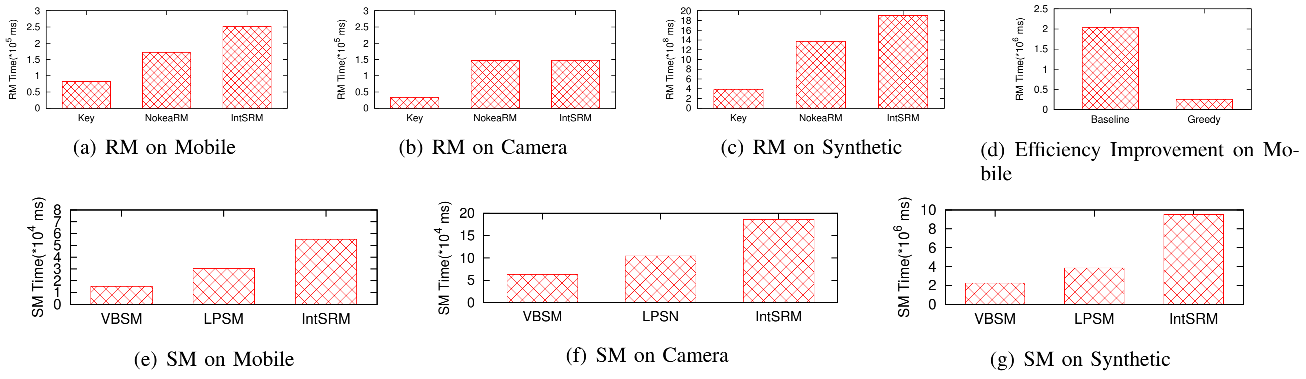
Fig. 5. Comparing the F_1-score of RM and SM Methods respectively on the Three Data Sets



(a) RM on Mobile (b) RM on Camera (c) RM on Synthetic

(d) SM on Mobile (e) SM on Camera (f) SM on Synthetic

Fig. 6. Comparing the Precision and Recall on Three Data Sets



(a) RM on Mobile (b) RM on Camera (c) RM on Synthetic (d) Efficiency Improvement on Mobile

(e) SM on Mobile (f) SM on Camera (g) SM on Synthetic

Fig. 7. Comparing the Time Cost with Other Methods on Three Data Sets

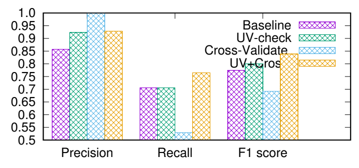
(1) Semantic Drift Control: As demonstrated in Fig. 8(a)(b)(c)(d), the two techniques have different performance on the two data collections, but the combination of them apparently improves the integration quality of both SM and RM. Specifically, UV-Check tends to improve the precision of the Baseline by around 10% for SM without hurting the recall, while Cross-Validate tends to improve the precision of the Baseline by around 10-15% but may hurt the recall of SM. This is because Cross-Validate uses a strict rule in deciding unqualified matching pairs. We set the threshold of Er value as 80% here. Overall, the combination of the two always improves the precision and recall of both SM and RM.

(2) Iterative Updating: As demonstrated in Fig. 9(a)(b), the quality of RM and SM can also make a further improvement and they can hold steady with a satisfied result as the iteration goes.

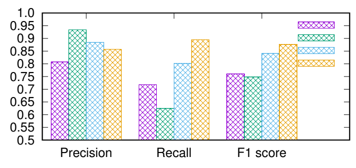
6.5 Efficiency Improvement Evaluation

We evaluate the effectiveness of our proposed techniques on improving the efficiency of our interaction algorithm. We use **Baseline** to denote the algorithm without any optimization on efficiency, and **Greedy** to denote our interaction algorithm based on the MLineIndex we build.

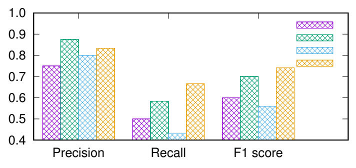
(1) Efficiency Improvement: As shown in Fig. 7(d), we can find that our Greedy method actually can save



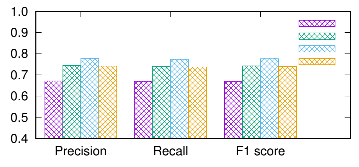
(a) SM on Mobile



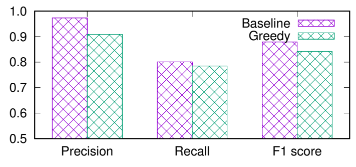
(b) RM on Mobile



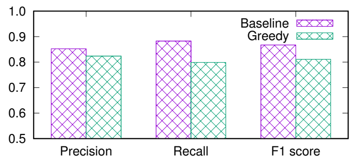
(c) SM on Camera



(d) RM on Camera

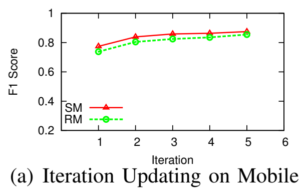


(e) Loss of Quality about SM on Mobile

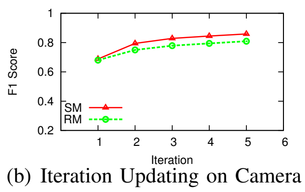


(f) Loss of Quality about RM on Mobile

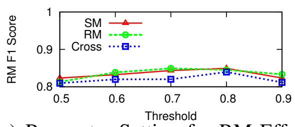
Fig. 8. Quality Improvement on ”Mobile” and ”Camera” Data Sets and Loss of Quality on “Mobile” Data Set



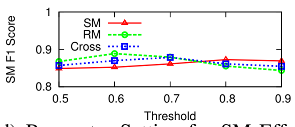
(a) Iteration Updating on Mobile



(b) Iteration Updating on Camera

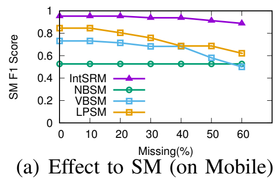


(c) Parameter Setting for RM Effectiveness

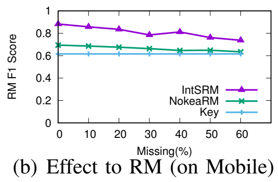


(d) Parameter Setting for SM Effectiveness

Fig. 9. Quality Improvement with Interaction(on Mobile and Camera) and Parameter Setting Test (on Syn)



(a) Effect to SM (on Mobile)



(b) Effect to RM (on Mobile)

Fig. 10. Effect of Missing Values to Integration Quality

almost 90% time cost of the baseline which proves the effectiveness of the proposed techniques.

(2) Side-effect: Loss in Quality: As a side-effect of the efficiency improvement, there is a little decrease on both precision and recall. As shown in Fig. 8(e)(f), there is less than 5% decrease on precision and 2%-5% decrease on recall, which is an acceptable price to pay for the reduction of 90% of the time cost.

6.6 Effect of Missing Values

We also conduct experiments on evaluating the effect of missing values to the performance of the proposed approaches and the other methods by randomly removing some non-key values from the table. As can be observed in Fig. 10(a)(b), as the missing ratio increases from 0% to 60%, the integration quality (including the F_1-score of SM and RM) of all approaches using non-key attribute values decreases. For SM, LPSM decreases the most from more than 0.80 to about 0.57, while our approach IntSRM can still reach about 0.88 when the missing ratio becomes 60%. For RM, NokeaRM is also robust and only decreases from about 0.7 to about 0.63, while our approach IntSRM decreases from 0.89 to about 0.75. Generally, our method always reaches nearly 10% higher F_1-score than the other methods.

6.7 Parameter Setting

We evaluate the effect of the three key parameters in our algorithm to the quality of data integration, which are: (1) the threshold to the SM matching likelihood; (2) the threshold to the RM matching likelihood; and (3) the threshold of Er value of Cross-Validate. To avoid potential biases, we fix the other two when we evaluate one of the three parameters. As can be seen from Fig. 9(c)(d), the F_1-score of SM or RM has no significant variation to different parameter settings. This coincides with the discussion in Sec. 4.3: different parameter settings of IntSRM will generate almost the same quality after different numbers of interactive steps. We also find that the time cost of them has no much difference.

7 Related Work

A host of works have been done on Schema Matching [28] and Record Matching [14]. While typical SM approaches are based on the similarity (or semantic correlation) between Attribute Names [15], [10], or Attribute Value Sets (i.e., instance-based SM) [2], [24] or combination of the two [12], typical RM methods measure the similarities between either key attribute values [4] or non-key attribute values [35]. Excellent surveys to the two problems can be found in [28] and [14], respectively.

For SM, our work is closely related to instance-based SM method, which has been recognized as an effective approach due to its robustness for matching heterogenous schemas. Generally, the instance-based SM method leverages the classified instance data to process SM by measuring the similarity between sets of annotated instances (e.g., the construction of links between attributes based on the co-occurrence of instances). The basic idea of instance-based SM method is that the more significant overlap among common instances, the more relevant the two attributes are. The challenge here lies on how to define the significance for the overlap. [16] determines the similarity of two attributes by executing a pair-wise comparison of instance values using a similarity function, while [18] defines the similarity of two attributes by considering both the specificity and generalization of instances when two attributes are linked. In this paper, we propose a new SM rule by measuring the similarity of two attributes between the sets of linked record-pairs. More advanced, our method tends to link attributes with more explicit evidence (i.e. linked record-pair should have similar values under the same attribute), even there are just a few overlapped instances.

In addition, instance-based SM methods [18], [23] spend much time on comparing two set of attribute values. To save the time cost, some methods calculate the similarity between two attributes based on selected small subsets of attribute values that are generated from the two large original attribute value sets respectively. In [13], [23], hashing method and filtering strategies were proposed to improve the scalability of the instance-based schema matching. In this paper, we extend the q-gram index [9], [22], [34] to multiple pairs of attributes scenario, and split potential matched record-pairs between the two tables into (possibly overlapped) blocks such that matching-record-pairs are only identified within every block. At each SM and RM step, we only process those blocks that will bring the maximum benefits to the future interaction with the minimum cost.

Efficiency is a more serious problem for RM since there are usually many more records than attributes in databases. So far, various techniques have been proposed to reduce the overhead of RM, including Q-Grams together with inverted indices [31], prefix-based pruning techniques [32], batch-based matching techniques [7]. However, these techniques are only applied on the key attributes. A recent work [35] uses both key and non-key attributes for RM and relies on a special decision-tree to improve the efficiency of RM. In this paper, we do RM based on both key and non-key attribute values under the linked attributes, which outperforms previous methods on accuracy. Meanwhile, inspired by the previous methods, we extend the indices over all attributes to improve the efficiency of RM and the interaction process, which has been demonstrated to significantly reduce time cost in our experimental results.

There are also works on using structural/relational similarity of records for SM or RM. [3] and [30] resolve related entities collectively and combine attribute similarity with relational evidence to improve the quality of RM results. In [26], a graph based on attributes, attribute types and their relations is built to capture inherent structural information of the table, which can be utilized in SM. These approaches can improve the integration quality on data sets that have strong structural/relational information to utilize. However, the improvement of integration quality will be limited when data sets have weak structural information.

Recently, external domain knowledge and human interventions are also employed to improve the quality of SM [12] or RM [21]. These methods get external knowledge from either external knowledge base like wikipedia, or crowd workers, and use these external knowledge to label some attributes or instances and then extract effective features for better SM or RM.

So far, all existing efforts take SM and RM as independent steps in data integration. There has been little discussion about the interaction between them. A similar interesting study has been conducted on the interaction problem between RM and Data Cleaning [17], but the problem setting and the key challenges in that work are different from ours in various aspects.

8 Conclusions and Future Work

In this paper, we study the interaction between SM and RM by performing them alternately in data integration. Extensive experiments on three data collections show that the combination and interaction between SM and RM significantly outperforms previous works that conduct SM and RM separately. Based on special-designed indices, we reduce around 90% overhead of the interaction algorithm.

Nonetheless, our approach has its own limitations: it can only work well with the case that one attribute (record) in a table matches with no more than one attribute (record) in the other table. As a future work, we will consider to extend our approach to deal with multiple matches. Our future work also includes addressing the problem when data is too large to be loaded into memory at a time.

Acknowledgements: This research is partially supported by Natural Science Foundation of China (Grant No. 61303019, 61402313, 61472263, 61572336), the King Abdullah University of Science and Technology, the Australian Research Council (Grants No. DP120102829), the Postdoctoral scientific research funding of Jiangsu Province (No. 1501090B) and the National Postdoctoral Funding (No. 2015M581859, 2016T90493).

References

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ACMDL*, pages 85–94. ACM, 2000.
[2] P. A. Bernstein, J. Madhavan, and E. Rahm. Generic schema matching, ten years later. *PVLDB*, 4(11):695–701, 2011.
[3] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):5, 2007.
[4] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *ACM SIGKDD*, pages 39–48. ACM, 2003.
[5] C. M. Bishop et al. *Pattern recognition and machine learning*. springer New York, 2006.
[6] R. A. Brualdi. *Introductory combinatorics*. New York, 1992.
[7] A. Chandel, P. Nagesh, and S. Sarawagi. Efficient batch top-k search for dictionary-based entity recognition. In *ICDE*, pages 28–28, 2006.
[8] P. Christen. A survey of indexing techniques for scalable record linkage and deduplication. *TKDE*, 24(9):1537–1555, 2012.
[9] P. Christen et al. *Towards parameter-free blocking for scalable record linkage*. Department of Computer Science, Faculty of Engineering and Information Technology, Australian National University, 2007.
[10] C. Comito, S. Patarin, and D. Talia. A semantic overlay network for p2p schema-based data integration. In *Computers and Communications, 2006. ISCC'06. Proceedings. 11th IEEE Symposium on*, pages 88–94. IEEE, 2006.
[11] D. Dey, V. S. Mookerjee, and D. Liu. Efficient techniques for online record linkage. *Knowledge and Data Engineering, IEEE Transactions on*, 23(3):373–387, 2011.
[12] R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. imap: discovering complex semantic matches between database schemas. In *SIGMOD*, pages 383–394. ACM, 2004.
[13] S. Duan, A. Fokoue, O. Hassanzadeh, A. Kementsietsidis, K. Srinivas, and M. J. Ward. Instance-based matching of large ontologies using locality-sensitive hashing. In *The Semantic Web–ISWC 2012*, pages 49–64. Springer, 2012.

[14] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *TKDE*, 19(1):1–16, 2007.

[15] D. W. Embley, D. Jackman, and L. Xu. Multifaceted exploitation of metadata for attribute match discovery in information integration. In *IIWeb*, pages 110–117, 2001.

[16] D. Engmann and S. Massmann. Instance matching with coma+. *University of Leipzig*, 2007.

[17] W. Fan, S. Ma, N. Tang, and W. Yu. Interaction between record matching and data repairing. *Journal of Data and Information Quality (JDIQ)*, 4(4):16, 2014.

[18] A. Ferraram, A. Nikolov, and F. Scharffe. Data linking for the semantic web. *Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications*, 169, 2013.

[19] A. Gal. Managing uncertainty in schema matching with top-k schema mappings. In *Journal on Data Semantics VI*, pages 90–114. Springer, 2006.

[20] F. Giunchiglia, M. Yatskevich, and P. Shvaiko. Semantic matching: Algorithms and implementation. In *Journal on Data Semantics IX*, pages 1–38. Springer, 2007.

[21] C. Gokhale, S. Das, A. Doan, J. F. Naughton, N. Rampalli, J. Shavlik, and X. Zhu. Corleone: Hands-off crowdsourcing for entity matching. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 601–612. ACM, 2014.

[22] L. Gravano, P. G. Ipeirotis, H. V. Jagadish, N. Koudas, S. Muthukrishnan, D. Srivastava, et al. Approximate string joins in a database (almost) for free. In *VLDB*, volume 1, pages 491–500, 2001.

[23] O. Hassanzadeh, K. Q. Pu, S. H. Yeganeh, R. J. Miller, L. Popa, M. A. Hernández, and H. Ho. Discovering linkage points over web data. *Proceedings of the VLDB Endowment*, 6(6):445–456, 2013.

[24] T. Kirsten, A. Thor, and E. Rahm. Instance-based matching of large life science ontologies. In *Data Integration in the Life Sciences*, pages 172–187. Springer, 2007.

[25] W. Lin, R. Yangarber, and R. Grishman. Bootstrapped learning of semantic classes from positive and negative examples. In *ICML Workshop on The Continuum from Labeled to Unlabeled Data*, page 21, 2003.

[26] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 117–128, 2002.

[27] D. Pollard. Quantization and the method of k-means. *IEEE Transactions on Information theory*, 28(2):199–204, 1982.

[28] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10(4):334–350, 2001.

[29] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. In *Journal on Data Semantics IV*, pages 146–171. Springer, 2005.

[30] J. Tang, A. C. Fong, B. Wang, and J. Zhang. A unified probabilistic framework for name disambiguation in digital library. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):975–987, 2012.

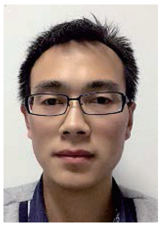
[31] E. Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theoretical computer science*, 92(1):191–211, 1992.

[32] W. Wang, C. Xiao, X. Lin, and C. Zhang. Efficient approximate entity extraction with edit distance constraints. In *SIGMOD*, pages 759–770, 2009.

[33] L. Wasserman. *All of statistics*. Springer Science & Business Media, 2011.

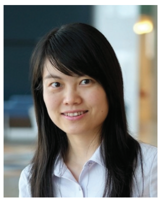
[34] C. Xiao, W. Wang, and X. Lin. Ed-join: an efficient algorithm for similarity joins with edit distance constraints. *PVLDB*, 1(1):933–944, 2008.

[35] Q. Yang, Z. Li, J. Jiang, P. Zhao, G. Liu, A. Liu, and J. Zhu. Nokearm: Employing non-key attributes in record matching. In *Web-Age Information Management*, pages 438–442. Springer, 2015.

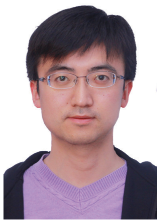


Binbin Gu is a master student at the Research Center on Advanced Data Analytics (ADA) in Soochow University, China. He has visited Prof. Xiangliang Zhang at KAUST for half a year in 2016. His research interests include Knowledge Fusion, Information Extraction and NLP. He has published several papers at DASFAA. He is the external reviewer of several International Conferences such as ADC and WAIM.

Zhixu Li is an associate professor in the Department of Computer Science & Technology at Soochow University, China. He used to work as a research fellow at KAUST. He received his Ph.D. degree from the University of Queensland in 2013, and his B.S. and M.S. degree from Renmin University of China in 2006 and 2009 respectively. His research interests are data cleaning, big data applications, information extraction and retrieval.



Xiangliang Zhang is an Assistant Professor and directs the Machine Intelligence and kNowledge Engineering (MINE) Laboratory in King Abdullah University of Science and Technology (KAUST). She earned her Ph.D. degree in computer science from INRIA-University Paris-Sud 11, France, in July 2010. Her main research interests and experiences are in diverse areas of machine learning and data mining. She has published over 60 papers in referred journals and conferences, including TKDE, SIGKDD, VLDB J, AAAI, IJCAI, ICDM, ECML/PKDD, CIKM, InfoCom etc.



An Liu is an associate professor in the Department of Computer Science & Technology at Soochow University. Prior to that in 2014, he was a Senior Research Associate in the Joint Research Center of City University of Hong Kong (CityU) and University of Science & Technology of China (USTC). He received his Ph. D. from both CityU and USTC in 2009. His research interests include security, privacy, and trust in emerging applications; cloud computing; and services computing.

Guanfeng Liu is an Associate Professor in the School of Computer Science and Technology at Soochow University, China. He received his Ph.D degree in Computer Science from Macquarie University, Australia in 2013. His research interests include social network mining and trust. He has published over 40 papers in the most prestigious journals and conferences such as ICDE, AAAI, ICWS and IEEE Transactions. He was the PC Chair of BDMS 2015.

Kai Zheng is a Professor with the School of Computer Science and Technology at Soochow University. He received his PhD degree in Computer Science from The University of Queensland in 2012. His research focus is to find effective and efficient solutions for managing, integrating and analyzing big data for business, scientific and personal applications. He has been working in the area of spatial-temporal databases, uncertain databases, trajectory computing, social-media analysis and bioinformatics. He has published over 60 papers in the highly referred journals and conferences such as SIGMOD, ICDE, EDBT, The VLDB Journal, ACM Transactions and IEEE Transactions.

Lei Zhao is a Professor with the School of Computer Science and Technology at Soochow University. He received his Ph.D. degree in Computer Science from Soochow University in 2006. His research focuses on graph databases, social media analysis, query outsourcing, parallel and distributed computing. His recent research is to analyze large graph database in an effective, efficient, and secure way. He has published over 100 papers including more than 20 published in well-known journals and conferences such as ICDE, DASFAA, WISE, JCST.

Xiaofang Zhou is a Professor of computer science with The University of Queensland. He is the Head of the Data and Knowledge Engineering Research Division. He is specially appointed Adjunct Professor under the Chinese National Qianren Scheme hosted by Renmin University of China (2010-2013), and by Soochow University since July 2013 where he leads the Research Center on Advanced Data Analytics (ADA). He has been working in the area of spatial and multimedia databases, data quality, high performance query processing, Web information systems and bioinformatics, co-authored over 250 research papers with many published in top journals and conferences.