Group Linkage

Byung-Won On
Penn State Univ.
on@cse.psu.edu

Nick Koudas
Univ. of Toronto
koudas@cs.toronto.edu

Dongwon Lee
Penn State Univ.
dongwon@psu.edu

Divesh Srivastava
AT&T Labs – Research
divesh@research.att.com

Abstract

Poor quality data is prevalent in databases due to a variety of reasons, including transcription errors, lack of standards for recording database fields, etc. To be able to query and integrate such data, considerable recent work has focused on the record linkage problem, i.e., determine if two entities represented as relational records are approximately the same. Often entities are represented as groups of relational records, rather than individual relational records, e.g., households in a census survey consist of a group of persons. We refer to the problem of determining if two entities represented as groups are approximately the same as group linkage.

Intuitively, two groups can be linked to each other if (i) there is high enough similarity between "matching" pairs of individual records that constitute the two groups, and (ii) there is a large fraction of such matching record pairs. In this paper, we formalize this intuition and propose a group linkage measure based on bipartite graph matching. Given a data set consisting of a large number of groups, efficiently finding groups with a high group linkage similarity to an input query group requires quickly eliminating the many groups that are unlikely to be desired matches. To enable this task, we present simpler group similarity measures that can be used either during fast pre-processing steps or as approximations to our proposed group linkage measure. These measures can be easily instantiated using SQL, permitting our techniques to be implemented inside the database system itself. We experimentally validate the utility of our measures and techniques using a variety of real and synthetic data sets.

1 Introduction

The quality of the data residing in databases gets degraded due to a multitude of reasons. Such reasons include transcription errors (e.g., lexicographical errors, character transpositions), lack of standards for recording database fields (e.g., person names, addresses), and various errors introduced by poor database design (e.g., update anomalies, missing key constraints). Data of poor quality can result in significant impediments to popular business practices: sending products or bills to incorrect addresses, inability to locate customer records during service calls, inability to correlate customers across multiple services, etc.

To be able to query and integrate such data in the presence of data quality errors, a central problem is the ability to identify whether two entities are approximately the same. When each entity is represented as a relational record, this problem is referred to as the *record linkage problem*. Depending on the type of data under consideration, various similarity measures (approximate match predicates) have been defined to quantify the closeness of a pair of records in a way that common mistakes are captured. Given any specific similarity measure, a key algorithmic problem in this context is the *approximate match problem*: given a large relation of data records and a single query record, identify all data records whose similarity with the query record exceeds a specified similarity threshold.

Often entities are represented as groups of relational records (sharing a group ID), rather than individual relational records. For example, a household in a census survey consists of a group of persons, where each person is represented by a relational record. Similarly, an author in a digital bibliography consists of a group of publications, where each publication is represented by a relational record. In such cases, there is often a need to identify whether two entities (represented by groups of relational records) are approximately the same. For example, re-sampling strategies used in statistical census surveys would need to identify matching households between the original sampled set and the re-sampled set. Similarly, when integrating different digital bibliographies (e.g., DBLP and ACM Digital Library), there is a need to identify an author in one that matches an author in the other. We refer to the problem of determining if two entities represented as groups are approximately the same as the **group linkage** problem.

Determining if two entities, represented as groups, can be linked to each other typically requires (approximately) matching elements (each of which is a relational record)

across the two groups using record linkage techniques. Intuitively, two groups can be linked to each other if:

- There is high enough similarity between "matching" pairs of individual records that constitute the two groups; the matching pairs of records do not need to be identical.
- A large fraction of records in the two groups form matching record pairs; not all records in each of the two groups need to match.

Our first technical contribution in this paper is to formalize this intuition and propose a group linkage measure based on bipartite graph matching, referred to as $BM_{sim,\rho}$, where ρ is a lower bound on the similarity between pairs of records in the two groups using record-level similarity measure sim. This measure is a natural generalization of the popular Jaccard similarity measure between two sets. In particular, if we constrain records from the two groups to match only when they are identical, our group linkage measure reduces to the Jaccard similarity measure.

Given the $BM_{sim,\rho}$ group linkage measure, a key algorithmic problem in this context is the *approximate match* problem: given a large relation of records (each associated with a group ID) D and a single query group of records g, identify all groups of records $g_i \in D$ such that $BM_{sim,\rho}(g, g_i) \geq \theta$, where $0 \leq \theta \leq 1$ is a group similarity threshold. Computing the group linkage measure $BM_{sim,\rho}$ requires the computation of maximum weight bipartite matching, which is an expensive operation. Efficiently finding groups $g_i \in D$ with a high group linkage similarity to an input query group requires quickly eliminating the many groups that are unlikely to be desired matches. To enable this task, our second contribution is the development of simpler group similarity measures that can be used either as lower or upper bounds to $BM_{sim,\rho}$, or as a blocking technique during a fast pre-processing step. We show that these simpler group similarity measures can be easily instantiated using SQL, permitting our techniques to be implemented inside the database system itself.

Our third contribution is a detailed experimental study validating the utility of our measures and techniques using a variety of real and synthetic data sets. We show that our measures can be implemented efficiently, are more robust and can achieve better recall than competing approaches.

2 Related Work

The group linkage problem is related with more general class of problems, known by various names – record linkage (e.g., [7, 3]), citation matching (e.g., [18]), identity uncertainty (e.g., [20]), merge-purge (e.g., [12]), object matching (e.g., [4]), duplicate detection (e.g., [21, 1]), approximate string join (e.g., [9]), etc. However, by and large, conventional approaches treat contents of a group as bag of tokens. On the other hand, we view that a group contains a set of records, and take advantage of record-level matching under the bipartite matching framework.

Bilenko et al. [3] have studied name matching for information integration using string-based and token-based methods. Cohen et al. have also compared the efficacy of string-distance metrics for the name matching task [6]. Blocking was first proposed by Kelley et al. [15] in the record linkage literature. Our blocking scheme is also similar in flavor to the two-step citation matching schemes proposed in [13, 18] where initial rough but fast clustering (or "Canopy") is followed by more exhaustive citation matching step. Another stream of works that are relevant to our work is name/entity disambiguation and authority controls in NLP community. For instance, work done in [23] aims at detecting name variants automatically using data mining or heuristic techniques. Similarly, [16, 11] study methods to find matching variants of a named entity in the digital library context. [19] studied the notion of "grouped entity resolution" using semantic graph partitioning, similar to our group linkage problem. Our proposal in this paper is more general and systematic, regardless of the existence of semantics in groups.

In a recent trend in the entity resolution problem, (e.g., [17, 14]), there is an attempt to exploit additional information beyond string comparison. A more extensive and systematic study is needed to investigate the usefulness and limitations of context in various flavors of the entity resolution problem. Unlike the traditional methods exploiting textual similarity, "constraint-based entity matching" [22] examines semantic constraints in an unsupervised way. [2] presents a generic framework, *Swoosh* algorithms, for the entity resolution problem.

3 Group Linkage: Problem

We use D to denote a relation of multi-attribute records, one of whose attributes is a group ID; r (possibly with subscripts) to denote records in D; and g (possibly with subscripts) to denote groups of records in D, i.e., a set of records that share the same value of the group ID attribute. Let sim denote an arbitrary record-level similarity measure, such that $sim(r_i, r_j) \in [0..1]$, where 1 denotes a perfect match. Table 1 summarizes the notations used throughout the paper.

Definition 1 (Group Linkage Measure). *Consider two groups of records* $g_1 = \{r_{11}, r_{12}, \ldots, r_{1m_1}\}$ *and* $g_2 = \{r_{21}, r_{22}, \ldots, r_{2m_2}\}$. *The* group linkage measure $BM_{sim,\rho}$ *is the normalized weight of the maximum weight matching* M *in the bipartite graph*

Notation	Description
D	relation of multi-attribute records
g_1, g_2, ...	groups of records in D
r_1, r_2, ...	records in D
$sim(r_i, r_j)$	arbitrary record-level similarity function
θ	group-level similarity threshold
ρ	record-level similarity threshold
M	maximum weight bipartite matching
BM	bipartite matching based group linkage
UB, LB	upper and lower bound of BM
MAX	$max()$ based heuristic group linkage

Table 1. Summary of notations.

ID	Group name	Description
g_{127381}	Amelie Marian	canonical group
g_{583980}	Amelie Marian	matching group
g_{136440}	Jean-Claude Mamou	false positive
g_{32238}	Brendan Hills	false positive

Table 2. Example of "Amelie Marian".

$(N = g_1 \cup g_2, E = g_1 \times g_2)$, *defined as:*

$$BM_{sim,\rho}(g_1, g_2) = \frac{\Sigma_{(r_{1i}, r_{2j}) \in M}(sim(r_{1i}, r_{2j}))}{m_1 + m_2 - |M|}$$

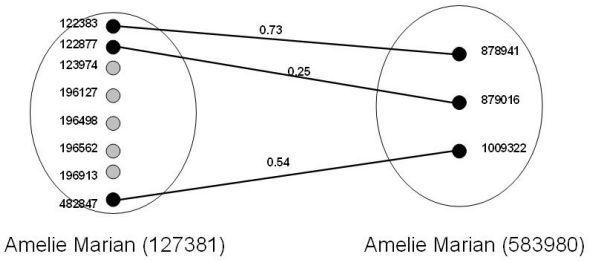
such that each $sim(r_{1i}, r_{2j}) \geq \rho$. □

The numerator of $BM_{sim,\rho}$ is the weight of the maximum weight bipartite matching M using only edges with record-level similarity no less than ρ. The denominator adds up the number of edges in the matching M and the number of "unmatched" elements (i.e., records) in each of g_1 (i.e., $m_1 - |M|$) and g_2 (i.e., $m_2 - |M|$). $BM_{sim,\rho}$ is guaranteed to be between 0 and 1. When the numerator is large, it captures the intuition that there is high enough similarity between "matching" pairs of individual records that constitute the two groups; the matching pairs of records do not need to be identical. When the denominator is small, it captures the intuition that a large fraction of records in the two groups form matching record pairs; not all records in each of the two groups need to match.

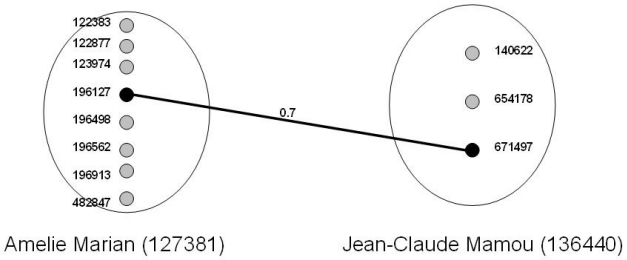
The group linkage measure $BM_{sim,\rho}$ is a natural generalization of the popular Jaccard similarity measure between two sets s_1 and s_2, defined as $\frac{|s_1 \cap s_2|}{|s_1 \cup s_2|}$. In particular, if we constrain the records from the two groups to match only when they are identical, our group linkage measure results in the same similarity value as the Jaccard similarity measure. Likewise, since $BM_{sim,\rho}$ is a generalization of the Jaccard measure, it can identify matching groups when the Jaccard fails as shown in the following example.

Example 1. Consider a real example in Table 2 derived from the data set $R1_{DB}$ in Section 5. The goal is to match g_{127381} with g_{583980}, without being confused with false positives, g_{136440} and g_{32238}. Note that these false positives share many common tokens with g_{127381}. For instance, g_{136440} ("Jean-Claude Mamou") shares common co-author tokens of {`brendan`, `bruno`, `cassio`, `dos`, `hills`, `jean-claude`, `marian`, `mignet`, `sophie`, `tova`, `vercoustre`, `abiteboul`, `aguilera`, `amann`, `anne-marie`, `bernd`, `cluet`, `hubert`, `laurent`, `mamou`, `milo`, `santos`, `serge`, `souza`, `tessier`, `vincent`} and title tokens of {`changes`, `documents`, `evaluating`, `optimizing`} with g_{127381} ("Amelie Marian").

On the other hand, matching group g_{583980} ("Amelie Marian") shares common co-author tokens of {`gravano`, `luis`} and title tokens of {`active`, `demonstration`, `repository`, `views`, `xml`, `queries`, `databases`, `detecting`, `multimedia`, `repositories`, `selection`, `xml`} with g_{127381} ("Amelie Marian"). When the Jaccard measure is applied, it would return g_{136440} ("Jean-Claude Mamou") as the most similar group to g_{127381} ("Amelie Marian") with the similarity of 0.22. However, the real matching group g_{583980} only comes second with the similarity of 0.2. On the other hand, as illustrated in Figure 1, when $BM_{cosine,0.1}$ is used (using $cosine$ similarity as the record-level similarity measure with 0.1 as threshold), it is able to identify the correct matching group, partly due to the fact that the two matching groups share three similar record-level pairs. The result is shown in Table 3. □



(a) Maximum weight bipartite matching for g_{583980}.



(a) Maximum weight bipartite matching for g_{136440}.

Figure 1. Illustration of $BM_{cosine,0.1}$.

Rank	g_1	g_2	$BM_{cosine,0.1}(g_1, g_2)$
1	g_{127381}	g_{583980}	0.19
2	g_{127381}	g_{32238}	0.07
3	g_{127381}	g_{136440}	0.07

Table 3. Result of $BM_{cosine,0.1}$.

Note that when $\rho = 0$, the group linkage measure permits any pair of records $(r_{1i}, r_{2j}) \in E$ to be part of the maximum weight matching M. In practice, however, for two records to be considered approximately the same their similarity needs to exceed some threshold $\rho > 0$ using a record-level similarity function sim, both of which are parameters to the group linkage measure BM.

The main problem addressed in this paper is the *approximate match problem*. Formally, our problem can be defined as follows.

> Given a large relation of records (each associated with a group ID) D and a single query group of records g, identify all groups of records $g_i \in D$ such that $BM_{sim,\rho}(g, g_i) \geq \theta$, where $0 \leq \theta \leq 1$ is a group similarity threshold.

A related problem is the top-k version of the problem where we are interested in identifying the groups of records $g_i \in D$ whose $BM_{sim,\rho}(g, g_i)$ values are the k highest among all groups in D.

Addressing these problems are essential to be able to query and integrate entities (such as households in census surveys and authors in digital bibliographies) that are represented as groups of multi-attribute records, in the presence of data quality errors.

4 Group Linkage: Solution

4.1 Bipartite Matching

The solution to the approximate match problem requires that we first identify all pairs of records $r_i \in g, r_j \in D$ such that the record-level similarity $sim(r_i, r_j) \geq \rho$. Efficient techniques are known for identifying such record pairs for a variety of record-level similarity measures, including edit distance [8] and tf.idf cosine similarity [9]. Recently, Chaudhuri et al. [5] proposed the SSJoin operator for optimizing and efficiently evaluating a variety of record-level similarity measures inside the database.

Once we have identified all such record pairs, the next step is to identify groups g_i in D such that $BM_{sim,\rho}(g, g_i) \geq \theta$. This requires the use of maximum weight bipartite matching. Bipartite matching is a well-studied problem for which efficient algorithmic solutions are known. Guha et al. [10] presented SQL realizations of the Hungarian algorithm and of an incremental strategy referred to as SSP for identifying maximum weight perfect bipartite matchings. While reasonably efficient for identifying the maximum weight bipartite matching (and hence $BM_{sim,\rho}$) for a pair of groups, applying such a strategy for every pair of groups is infeasible when the database relation contains a very large number of groups.

Efficiently finding groups $g_i \in D$ with a high group linkage similarity to input query group g requires quickly eliminating the many groups that are unlikely to be desired matches. To enable this task, we next develop simpler group similarity measures that can be used either as bounds to $BM_{sim,\rho}$, or as a blocking technique during a fast preprocessing step.

4.2 Greedy Matching

Maximum weight bipartite matching is computationally expensive because of the requirement that no node in the bipartite graph can have more than one edge incident on it. We can quickly obtain (upper and lower) bounds on $BM_{sim,\rho}$ by relaxing this requirement, using the following greedy strategy for each pair of groups g_1, g_2.

- For each record $r_i \in g_1$, find a record $r_j \in g_2$ with the highest record-level similarity among those with $sim() \geq \rho$. For the pair of groups g_1, g_2, let $S1$ denote the set of all such record pairs.
- Similarly, for each record $r_j \in g_2$, find a record $r_i \in g_1$ with the highest record-level similarity among those with $sim() \geq \rho$. For the pair of groups g_1, g_2, let $S2$ denote the set of all such record pairs.

Note that neither $S1$ nor $S2$ may be a matching. In $S1$, the same record in g_2 may be the target of more than one record in g_1. Similarly, in $S2$, the same record in g_1 may be the target of more than one record in g_2. However, these sets can be used to quickly obtain bounds for $BM_{sim,\rho}$, using the following formulas.

Definition 2 (Upper and Lower Bounds). *Consider two groups of records $g_1 = \{r_{11}, r_{12}, \ldots, r_{1m_1}\}$ and $g_2 = \{r_{21}, r_{22}, \ldots, r_{2m_2}\}$. The upper and lower bounds of* group linkage measure $BM_{sim,\rho}$ *is defined as:*

$$UB_{sim,\rho}(g_1, g_2) = \frac{\Sigma_{(r_{1i}, r_{2j}) \in S1 \cup S2}(sim(r_{1i}, r_{2j}))}{m_1 + m_2 - |S1 \cup S2|}$$

$$LB_{sim,\rho}(g_1, g_2) = \frac{\Sigma_{(r_{1i}, r_{2j}) \in S1 \cap S2}(sim(r_{1i}, r_{2j}))}{m_1 + m_2 - |S1 \cap S2|}$$

where $S1$ and $S2$ are defined above. □

Comparing $BM_{sim,\rho}$ of Definition 1 and $UB_{sim,\rho}$ of Definition 2, we can infer that the numerator of $UB_{sim,\rho}$

is at least as large as the numerator of $BM_{sim,\rho}$, and the denominator of $UB_{sim,\rho}$ is no larger than the denominator of $BM_{sim,\rho}$. Similarly, by comparing $BM_{sim,\rho}$ of Definition 1 and $LB_{sim,\rho}$ of Definition 2, we can infer that the numerator of $LB_{sim,\rho}$ is no larger than the numerator of $BM_{sim,\rho}$, and the denominator of $LB_{sim,\rho}$ is at least as large as the denominator of $BM_{sim,\rho}$. As a result, we have the following result.

Theorem 1. *Consider two groups of records g_1 and g_2. Then, for this pair of groups, we have that*

$$LB_{sim,\rho}(g_1, g_2) \leq BM_{sim,\rho}(g_1, g_2) \leq UB_{sim,\rho}(g_1, g_2)$$

That is, $BM_{sim,\rho}$ is bounded. ∎

The advantage of these simpler group similarity measures $UB_{sim,\rho}$ and $LB_{sim,\rho}$ is that these bounds can be easily and efficiently instantiated using SQL, permitting them to be implemented inside the database system itself. Quickly computing $UB_{sim,\rho}$ and $LB_{sim,\rho}$ can help us efficiently address our approximate match problem as follows.

1. Since $UB_{sim,\rho}$ is an upper bound on $BM_{sim,\rho}$, if $UB_{sim,\rho}(g_1, g_2) < \theta$, then it must be the case that $BM_{sim,\rho}(g_1, g_2) < \theta$. Hence, (g_1, g_2) is guaranteed to not be part of the answer to the approximate match problem and can be pruned away.
2. Since $LB_{sim,\rho}$ is a lower bound on $BM_{sim,\rho}$, if $LB_{sim,\rho}(g_1, g_2) \geq \theta$, then it must be the case that $BM_{sim,\rho}(g_1, g_2) \geq \theta$. Hence, (g_1, g_2) is guaranteed to be part of the answer to the approximate match problem and can be selected without computing the more expensive $BM_{sim,\rho}$.
3. Only when $LB_{sim,\rho}(g_1, g_2) < \theta \leq UB_{sim,\rho}(g_1, g_2)$, the more expensive $BM_{sim,\rho}(g_1, g_2)$ computation would be needed.

Proposition 1. *The group similarity measures $UB_{sim,\rho}$ and $LB_{sim,\rho}$ can be used in a pre-processing step to speed up computation of the answers to the approximate match problem*, without incurring any false negatives.

4.3 Heuristic Measure

In this section, we describe a group similarity measure that is even simpler (and hence faster to compute) than $UB_{sim,\rho}$ and $LB_{sim,\rho}$, and can also be used during a preprocessing step (called *blocking*).

Definition 3 (Heuristic Group Linkage Measure). *For a pair of groups (g_1, g_2), a heuristic measure using the max function is defined as follows:*

$$MAX_{sim,\rho}(g_1, g_2) = \max_{(r_{1i}, r_{2j}) \in g_1 \times g_2} (sim(r_{1i}, r_{2j}))\Box$$

The key intuition behind the use of $MAX_{sim,\rho}$ is that it is quite likely in practice that pairs of groups with a high value of $BM_{sim,\rho}$ will share at least one record with a high record-level similarity. By quickly identifying groups with the largest values of $MAX_{sim,\rho}$, we can avoid computing $BM_{sim,\rho}$ on a significant number of groups. Unlike the use of $UB_{sim,\rho}$ and $LB_{sim,\rho}$ as described above, however, this is a heuristic, without any guarantees of avoiding false negatives. We experimentally evaluate the utility of $MAX_{sim,\rho}$ in Section 5.

4.4 Implementation

It becomes increasingly important to efficiently instantiate data quality algorithms (e.g., record or group linkage algorithms) using SQL, enabling implementation on top of any DBMS. Recent examples of this trend include SQL implementation of matching algorithms [9] and the merge algorithms in [10]. We built $BM_{sim,\rho}$ by extending SSP in [10], and implemented all of the $UB_{sim,\rho}$, $LB_{sim,\rho}$, and $MAX_{sim,\rho}$ in SQL. In the interest of space, we only discuss $UB_{sim,\rho}$.

The basic logic of $UB_{sim,\rho}$ is: for each element in g_i, find the best match in g_j. It is possible that g_i has multiple edges incident on it (i.e., match). Then, the similarity between g_i and g_j is computed as the sum of edge weights divided by (number of matched edges + number of unmatched nodes in g_i and g_j). Some of the tables being used include:

- $Group_k$(gid, name, numOfElm): records the number of elements per group.
- $GrpToElm_k$(gid, eid): associates each group with elements in it.
- $ElmSim$(eid1, eid2, sim): records, for all pair-wise records, record-level similarity ($\geq \rho$) using cosine measure with TF/IDF weighting.

Table 4, for instance, shows a snippet of SQL statement to implement $UB_{sim,\rho}$.

5 Experimental Validation

We have introduced two versions of the group linkage problem – threshold and top-k. In the experiments, we used the top-k version: given a group g_i and the answer window k, return the top-k matching groups to g_i. We choose to do so in order to be able to evaluate both *performance* as well as *quality* of the answers.

5.1 Set-up

For evaluating our group linkage proposal, we extracted various data sets, as summarized in Table 5, from the ACM

```
insert UB select T.gid1 as gid1, T.gid2 as gid2,
   T.s/cast(T.cnt+abs(G1.numOfElm-T.cnt)+abs(G2.numOfElm-T.cnt) as float) as sim
from (
        select count(*) as cnt, sum(B.sim) as s, B.gid1 as gid1, B.gid2 as gid2
        from
        (
           select A.eid1 as eid1, max(A.sim) as sim, A.gid1 as gid1, A.gid2 as gid2
           from
           (
              select S.eid1 as eid1, S.sim as sim, E1.gid as gid1, E2.gid as gid2
              from GrpToElm1 E1, GrpToElm2 E2, ElmSim S
              where E1.eid = S.eid1 and E2.eid = S.eid2
           ) A
           where A.eid1 in (select S.eid1
                            from GrpToElm1 E1, GrpToElm2 E2, ElmSim S
                            where E1.eid = S.eid1 and E2.eid = S.eid2)
           group by A.eid1, A.gid1, A.gid2
        ) B
        group by B.gid1, B.gid2
     ) T, Group1 G1, Group2 G2
where G1.gid = T.gid1 and G2.gid = T.gid2
...
// if there is a threshold
having T.s/cast(T.cnt+abs(G1.numOfElm-T.cnt)+abs(G2.numOfElm-T.cnt) as float) > 0
```

Table 4. The SQL statement for $UB_{sim,\rho}$.

Notation	Left (all authors with at least 5 citations)	Right
$R1_a$	279 authors with a keyword *XML* in titles from DBLP	700,000 authors from ACM
$R1_b$	265 authors with a keyword *Query* in titles from ACM	400,000 authors from DBLP
$R2_{DB}$	100 authors with keywords *Query* and *Schema* in titles from DBLP	700,000 authors from ACM
$R2_{AI}$	100 authors with keywords *Reasoning* and *Recognition* in titles from DBLP	700,000 authors from ACM
$R2_{Net}$	100 authors with keywords *ATM* and *TCP* in titles from DBLP	700,000 authors from ACM
$S1_a$	279 authors with a keyword *XML* in titles from DBLP	700,000 authors from ACM + 279 $\frac{1}{3}$-type dummies
$S1_b$	279 authors with a keyword *XML* in titles from DBLP	700,000 authors from ACM + 279 $\frac{1}{3}$-type dummies
$S2_s$	100 authors with keywords *Memory* and *Power* in titles from ACM	100 authors (same as left) + 100 authors with 30% errors
$S2_m$	100 authors with keywords *Memory* and *Power* in titles from ACM	100 authors (same as left) + 100 authors with 45% errors
$S2_l$	100 authors with keywords *Memory* and *Power* in titles from ACM	100 authors (same as left) + 100 authors with 60% errors

Table 5. Summary of data sets.

and DBLP citation digital libraries. These libraries typically have a list of citations per author. Therefore, we can treat each author as a *group*, citations of an author as *records* in a group, and linkage between authors as the *group linkage* problem. By using author names as a key to link, we evaluate how well a method can link a group in the left data set (e.g., "Vannevar Bush" in DBLP) to matching groups in the right data set (e.g., "V. Bush" in ACM).

Real data sets. First, we prepared two kinds of real data sets, $R1$ and $R2$, each of which in turn has several variations, $R1_a$ and $R1_b$ for $R1$ and $R2_{DB}$, $R2_{AI}$, and $R2_{Net}$ for $R2$, respectively. In both $R1_a$ and $R1_b$ data sets, authors in the left are selected from DB venues such as SIGMOD, VLDB, and ICDE, and have at least 5 citations. The average number of citations in the left and right authors are 41 and 25 for $R1_a$ and 40 and 55 for $R1_b$. Therefore, these are rather *uniform* data sets.

Next, we create skewed data sets in three domains – (1) $R2_{DB}$ with authors in the left from DB venues such as SIGMOD, VLDB, and ICDE, (2) $R2_{AI}$ with authors in the left from AI venues such as AAAI, IJCAI, and ICML, and (3) $R2_{Net}$ with authors in the left from Network venues such as INFOCOM, MOBICOM, and SIGCOMM. The average numbers of citations in the left and matching right author sets are 30 and 9 for $R2_{DB}$, 31 and 10 for $R2_{AI}$, 22 and 6 for $R2_{Net}$. Therefore, these are rather *skewed* data sets.

For all these data sets, we have manually verified if an author in the left matches an author in the right. Note that some matching authors in the left and right have little resemblance in their names. For instance, "Shlomo Argamon" in DBLP and "Sean Engelson" in ACM are matching authors.[1] On the other hand, some authors in the left have

[1]This is verified at his home page `http://lingcog.iit.edu/ argamon/`.

multiple matching authors in the right. For instance, "Jeffrey D. Ullman" in DBLP has numerous matching authors in ACM such as "J. Ullman", "J. D. Ullman", and "Ullman, Jeffrey", each of which has a varying number of citations in it. In such a case, we merge all matching authors in ACM into an arbitrary one (e.g., merge "J. Ullman" and "J. D. Ullman" to "Ullman, Jeffrey"). In the end, for all data sets of Table 5, for a group in the left, there is only *one* matching group in the right.

Synthetic data sets. To see the effect of varying degrees of error in the data sets, we introduce controlled errors into the real data sets. First, we prepare two synthetic data sets, $S1_a$ and $S1_b$. Both are the same as the real data set $R1_a$, except that dummy authors are injected to the right. Two types of dummy authors are prepared. For an author A, its dummy author D has to use the same bag of words in citations as A, but the number of citations in D is either $\frac{1}{3}$ (for $S1_a$) or 3 (for $S1_b$) times the number of citations in A. Therefore, in the dummy author for $S1_a$, each citation will have many more words than a citation in A so that the number of citations becomes $\frac{1}{3}$ that of A. Similarly, in the dummy author for $S1_b$, each citation will have fewer words than a citation in A so that the number of citations becomes 3 times that of A. Note that a non-robust method such as Jaccard will get confused and return the injected dummy author from the right as the matching author of the left, incorrectly. The goal is to see if the bipartite matching group linkage method is also confused or not.

Finally, since it is not feasible to run complete bipartite matching against the 700,000 authors in the right, we generated another synthetic data set. This time, the right hand side has also a small number of authors like the left hand side so that bipartite matching can be applied without any blocking. Using the dbgen tool from the University of Toronto, for each author in the left, we have generated a dummy author with varying levels of errors (i.e., typos, abbreviation, omission, contraction, etc) and inserted it to the right data set. The goal is to see if a method is able to detect the correct matching author, without being confused with the dummy author.

Evaluation Metrics. For evaluating the methods, we used the average recall with an answer window size of k as well as the running time. Note that for each author a_1 in the left, there is only one other matching author a_2 in the right. Therefore, if a_2 is included in the top-k answer window for a_1, then recall becomes 1, and 0 otherwise. As the window size k increases, recall is likely to increase. At the end, we measure the *average recall* for all authors.

Compared Methods. Four methods, as summarized in Table 7, as well as their hybrid combinations are evaluated. Given two groups, the Jaccard (JA) method treats each group as a bag of tokens, and measures the size of intersected tokens over the size of union of tokens. When a method A is used with an answer window size of k, then it is denoted as $A(k)$. When a method A with a window k_1 is used in step 1, followed by a method B with a window k_2 in step 2, then the hybrid approach is denoted as $A(k_1)|B(k_2)$. For instance, the method $MAX(5)|BM(1)$ refers to the hybrid of MAX with top-5 in step 1 and BM with top-1 in step 2. As a record-level similarity measure, we used cosine similarity with TF/IDF weighting. This computation is done once at a pre-processing stage. The running times are shown in Table 6 for the case of $R2$. Other data sets took similar time, except $S2$ which took only minutes.

Data set	Pre-processing time
$R2_{DB}$	1:20:05
$R2_{AI}$	1:23:57
$R2_{Net}$	1:32:48

Table 6. Pre-processing time for computing cosine similarity of $R2$. (hh:mm:ss).

Notation	Description
JA	Jaccard based group linkage measure
BM	$BM_{sim,\rho}$ group linkage measure
UB	$UB_{sim,\rho}$ group linkage measure
MAX	$MAX_{sim,\rho}$ group linkage measure

Table 7. Notation in experimentation.

Summary of Set-up. Using this set-up, we seek to answer the following questions:

- Q1: Compared to the naive approach (e.g., Jaccard measure), how does $BM_{sim,\rho}$ behave? We use the $R1$, $S1$, and $S2$ data sets.
- Q2: How robust is $BM_{sim,\rho}$ in the presence of errors? We use the $S1$ and $S2$ data sets.
- Q3: How does the heuristic based $MAX_{sim,\rho}$ behave? We use the $R2$ data set.
- Q4: Can $UB_{sim,\rho}$ be used as fast pre-processing step (i.e., blocking) for $BM_{sim,\rho}$? We use the $R2$ data set.

All experiments were conducted using the Microsoft SQL Server 2000 on Pentium III 3GHZ/512MB machine.

5.2 Results

1. $R1$ real data set. Against $R1_a$ and $R1_b$, in Figure 2, JA(1) (i.e., JA with window size of 1) achieved recalls of 0.84 and 0.93, respectively. Since $R1$ data sets are rather

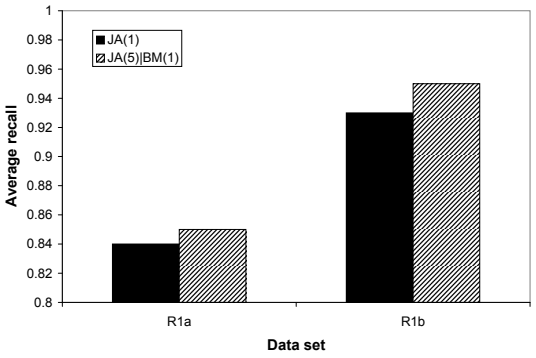


Figure 2. $JA(1)$ **vs.** $JA(5)|BM(1)$ **against** $R1$**.**

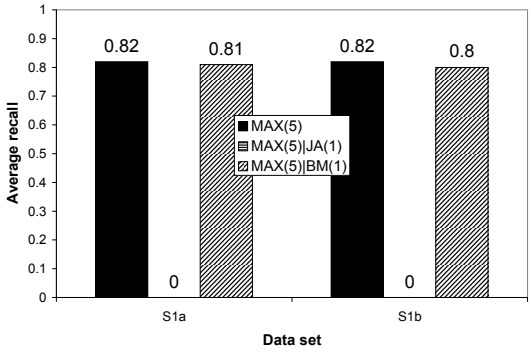


Figure 3. $JA(1)$ **vs.** $BM(1)$ **against** $S1$**.**

uniform with respect to the number of citations between two author groups, it is an ideal data set for JA. This explains the relatively high recall of 0.84 and 0.93. Next, we examined whether using the bipartite matching (BM) idea improves the overall recall or not. Since the graph-based BM has non-linear running time, it is not feasible to apply BM directly to $R1_a$ and $R1_b$ which has half million authors in the right. Therefore, we first apply JA(5) as the blocking method to get five candidates per canonical author, then apply BM(1) to get the top-1 matching author. Figure 2 shows that JA(5)|BM(1) archives recalls of 0.85 for $R1_a$ and 0.95 for $R1_b$, respectively. Note that using BM does not worsen the recall but the improvement is only minor – 1-2%. Therefore, when two groups share many common tokens and when there is no other *confusing* group with common tokens, JA works relatively well. Next, we show that this is no longer the case when JA is faced with confusing groups.

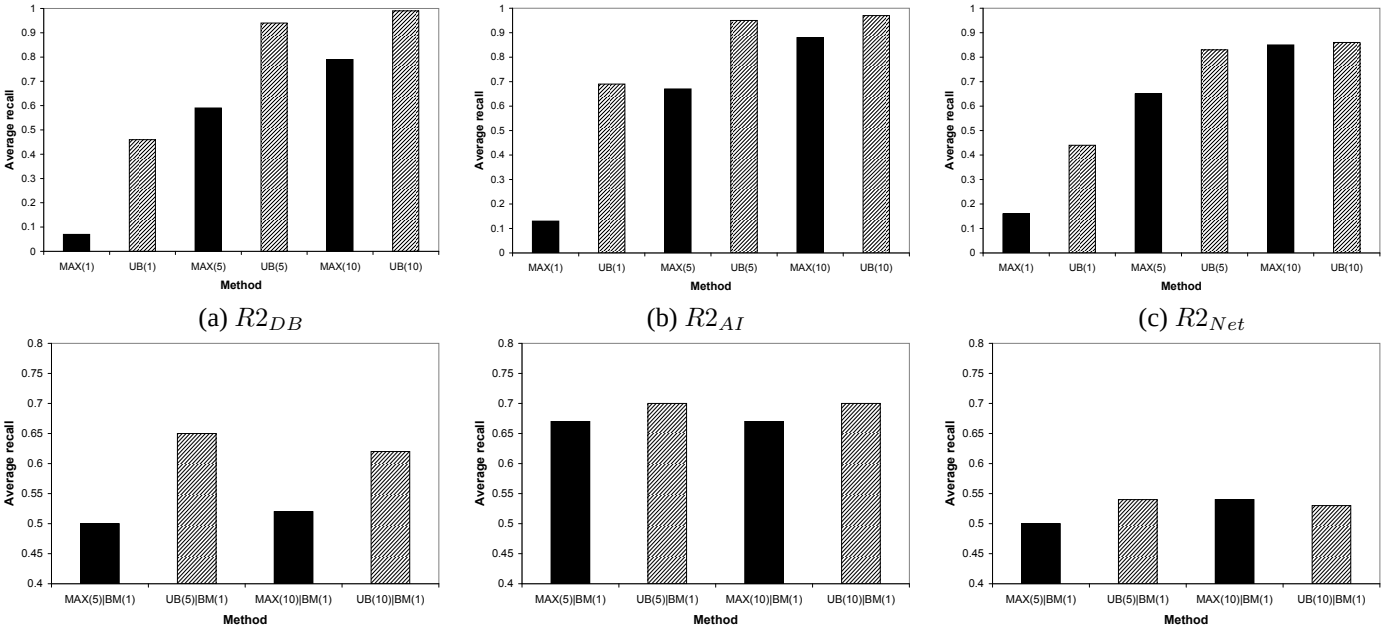
2. $S1$ **and** $S2$ **synthetic data sets.** To demonstrate that our proposed BM is more robust than JA, we created an extreme case that is the worst for JA, and see how BM behaves under the same environment. Note that JA determines a matching group purely based on the commonality of tokens. Therefore, if we take a canonical group, a, with N records from the left and inject a copy of it, a_c, to the right, then JA will always return a_c from the right as the matching author of a since both a and a_c have an identical bag of tokens. Based on this idea, we generate two dummy authors: (1) a_c with $\frac{N}{3}$ citations and the same bag of tokens as a ($S1_a$), and (2) a_c with $3 \times N$ citations and the same bag of tokens as a ($S1_b$).

Figure 3 shows the result of this comparison. To speed-up, we used MAX(5) as the *blocking* (i.e., pre-processing step), and both JA(1) and BM(1) are applied at a later step. Against the $S1_a$ data set, first, MAX(5) achieves a recall of 0.82 in the blocking stage. When both JA(1) and BM(1) are applied to the five candidate authors that MAX(5) found, MAX(5)|JA(1) has a recall of 0 while MAX(5)|BM(1) has a recall of 0.81. As per design, JA incorrectly selects dummy authors from the right, yielding 0 as recall. However, the proposed BM does not get confused and correctly identifies matching authors. Since the recall of 0.82 by MAX(5) is in a sense the ideal recall for subsequent methods, the recall of 0.81 by MAX(5)|BM(1) is a satisfactory result. The same pattern emerges for the $S1_b$ data set as well. Next, we repeat the experimentation using $S2$. This time, we did not use blocking. Instead, both JA and BM are directly applied to $S2$. Figure 4 shows that $BM(1)$, regardless of the error levels, outperforms $JA(1)$ by 16-17% in recall.

Therefore, JA is a fast and simple method which can be used for group linkage, but is prone to errors. If there exist groups with tokens similar to the canonical group, or matching groups have intolerable noises, then JA is easily confused, as shown in Figure 3. On the other hand, regardless of data types or error levels, BM is able to perform better group linkage.

3. $R2$ **real data set.** $R2$ data sets are more challenging than $R1$ since the average number of elements per group between left and right is *skewed*. For instance, on average, canonical authors in the left have three times more citations than corresponding authors in the right. First, Figures 5 (a)-(c) show the comparison of MAX and UB. For all three data sets, UB outperforms MAX in recall. Since MAX is heuristic based, it tends to be looser than UB (i.e., as long as there is one record pair with high similarity, group linkage will have a high similarity), allowing more errors. On the other hand, for the same reason, MAX is slightly faster than UB, as shown in Table 8 (first six columns).

Next, Figures 5 (d)-(f) show how BM behaves when either UB or MAX is first applied as pre-processing. Regardless of data sets, UB followed by BM outperforms



(a) $R2_{DB}$ (b) $R2_{AI}$ (c) $R2_{Net}$

(d) $R2_{DB}$ (e) $R2_{AI}$ (f) $R2_{Net}$

Figure 5. MAX **vs.** UB **for (a)-(c) and** $MAX|BM$ **vs.** $UB|BM$ **for (d)-(f) against** $R2$**.**

	MAX(1)	UB(1)	MAX(5)	UB(5)	MAX(10)	UB(10)	UB(5)|BM(1)	UB(10)|BM(1)
$R2_{DB}$	0.23	0.35	0.24	0.38	0.24	0.38	4.91	7.45
$R2_{AI}$	0.2	0.28	0.2	0.28	0.2	0.28	4.44	6.66
$R2_{Net}$	0.27	0.4	0.26	0.43	0.27	0.43	4.61	6.41

Table 8. Running time (per group) of $MAX(k)$**,** $UB(k)$**, and** $UB(k)|BM(1)$ **against** $R2$ **(in sec).**

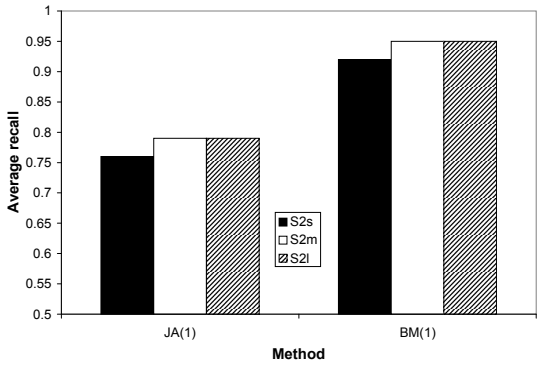


Figure 4. $JA(1)$ **vs.** $BM(1)$ **against** $S2$**.**

MAX followed by BM by 5-14%. This is consistent with Proposition 1, supporting our claim that UB is a good preprocessing step for BM. The answer window size in step 2 is somewhat correlated with the recall. For instance, Figure 6 is the case of $UB(10)|BM(k)$ against $R2_{AI}$, where $k = 1, ..., 10$ on the Y-axis.

6 Conclusions and Future Work

In this paper, we have studied a new problem of *group linkage*, and proposed a bipartite matching based group similarity measure, $BM_{sim,\rho}$, which naturally generalizes the Jaccard measure. In addition, we proved the upper and lower bounds of $BM_{sim,\rho}$ can be used as a filtering step for speed-up. Finally, through extensive experiments and SQL implementations, we have validated that $BM_{sim,\rho}$ is a more robust group similarity measure than others, and can efficiently detect matching groups together with proper preprocessing step.

Many directions are ahead for future work. First, we

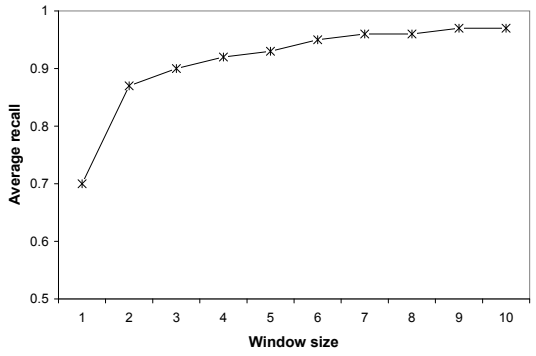


Figure 6. $UB(10)|BM(k)$ **against** $R2_{AI}$**.**

plan to compare our proposed measures against data mining measures for hierarchical clustering (e.g., single link, complete link). In a sense, these are heuristic measures similar to our $MAX_{sim,\rho}$. Second, we plan to do user study to evaluate how humans perform given two groups to match. It is to find out if matching groups that mechanical algorithms such as ours detect are indeed the same as what humans would choose. Third, we also plan to explore issues when there is a hierarchy among groups. Finally, our group linkage technique can be applied to other applications such as region-based image retrieval. We plan to conduct more experiments on this application.

Acknowledgement. The research of Dongwon Lee is partially supported by Microsoft SciData Award (2005) and IBM Eclipse Innovation Award (2006).

References

[1] R. Ananthakrishna, S. Chaudhuri, and V. Ganti. "Eliminating Fuzzy Duplicates in Data Warehouses". In *VLDB*, 2002.

[2] O. Benjelloun, H. Garcia-Molina, Q. Su, and J. Widom. "Swoosh: A Generic Approach to Entity Resolution". Technical report, Stanford University, 2005.

[3] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. "Adaptive Name-Matching in Information Integration". *IEEE Intelligent System*, 18(5):16–23, 2003.

[4] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani. "Robust and Efficient Fuzzy Match for Online Data Cleaning". In *ACM SIGMOD*, 2003.

[5] S. Chaudhuri, V. Ganti, and R. Kaushik. "A Primitive Operator for Similarity Joins in Data Cleaning". In *IEEE ICDE*, 2006.

[6] W. Cohen, P. Ravikumar, and S. Fienberg. "A Comparison of String Distance Metrics for Name-matching tasks". In *IIWeb Workshop held in conjunction with IJCAI*, 2003.

[7] I. P. Fellegi and A. B. Sunter. "A Theory for Record Linkage". *J. of the American Statistical Society*, 64:1183–1210, 1969.

[8] L. Gravano, P. G. Ipeirotis, H. V. Jagadish, N. Koudas, S. Muthukrishnan, and D. Srivastava. "Approximate String Joins in a Database (Almost) for Free". In *VLDB*, pages 491–500, 2001.

[9] L. Gravano, P. G. Ipeirotis, N. Koudas, and D. Srivastava. "Text Joins in an RDBMS for Web Data Integration". In *Int'l World Wide Web Conf. (WWW)*, 2003.

[10] S. Guha, N. Koudas, A. Marathe, and D. Srivastava. "Merging the Results of Approximate Match Operations". In *VLDB*, pages 636–647, 2004.

[11] H. Han, C. L. Giles et al. "Two Supervised Learning Approaches for Name Disambiguation in Author Citations". In *ACM/IEEE Joint Conf. on Digital Libraries (JCDL)*, Jun. 2004.

[12] M. A. Hernandez and S. J. Stolfo. "The Merge/Purge Problem for Large Databases". In *ACM SIGMOD*, 1995.

[13] J. A. Hylton. *"Identifying and Merging Related Bibliographic Records"*. PhD thesis, Dept. of EECS, MIT, 1996. LCS Technical Report MIT/LCS/TR-678.

[14] D. V. Kalashnikov, S. Mehrotra, and Z. Chen. "Exploiting Relationships for Domain-independent Data Cleaning". In *SIAM Data Mining (SDM) Conf.*, 2005.

[15] R. P. Kelley. "Blocking Considerations for Record Linkage Under Conditions of Uncertainty". In *Proc. of Social Statistics Section*, pages 602–605, 1984.

[16] D. Lee, B.-W. On, J. Kang, and S. Park. "Effective and Scalable Solutions for Mixed and Split Citation Problems in Digital Libraries". In *ACM SIGMOD Workshop on Information Quality in Information Systems (IQIS)*, Jun. 2005.

[17] B. Malin. "Unsupervised Name Disambiguation via Social Network Similarity". In *SIAM SDM Workshop on Link Analysis, Counterterrorism and Security*, 2005.

[18] A. McCallum, K. Nigam, and L. H. Ungar. "Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching". In *ACM KDD*, Boston, MA, Aug. 2000.

[19] B.-W. On, E. Elmacioglu, D. Lee, J. Kang, and J. Pei. "An Effective Approach to Entity Resolution Problem Using Quasi-Clique and its Application to Digital Libraries, ". In *ACM/IEEE Joint Conf. on Digital Libraries (JCDL)*, Jun. 2006.

[20] H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. "Identity Uncertainty and Citation Matching". In *Advances in Neural Information Processing Systems*. MIT Press, 2003.

[21] S. Sarawagi and A. Bhamidipaty. "Interactive Deduplication using Active Learning". In *ACM KDD*, 2002.

[22] W. Shen, X. Li, and A. Doan. "Constraint-Based Entity Matching". In *AAAI*, 2005.

[23] J. W. Warnner and E. W. Brown. "Automated Name Authority Control". In *ACM/IEEE Joint Conf. on Digital Libraries (JCDL)*, 2001.