

Improving Grouped-Entity Resolution using Quasi-Cliques

Byung-Won On, Ergin Elmacioglu, Dongwon Lee*
The Pennsylvania State University
{on, ergin, dongwon}@psu.edu

Jaewoo Kang†
NCSU & Korea Univ.
kangj@korea.ac.kr

Jian Pei
Simon Fraser Univ.
jpei@cs.sfu.ca

Abstract

The entity resolution (ER) problem, which identifies duplicate entities that refer to the same real world entity, is essential in many applications. In this paper, in particular, we focus on resolving entities that contain a group of related elements in them (e.g., an author entity with a list of citations, a singer entity with song list, or an intermediate result by GROUP BY SQL query). Such entities, named as grouped-entities, frequently occur in many applications. The previous approaches toward grouped-entity resolution often rely on textual similarity, and produce a large number of false positives. As a complementing technique, in this paper, we present our experience of applying a recently proposed graph mining technique, Quasi-Clique, atop conventional ER solutions. Our approach exploits contextual information mined from the group of elements per entity in addition to syntactic similarity. Extensive experiments verify that our proposal improves precision and recall up to 83% when used together with a variety of existing ER solutions, but never worsens them.

1 Introduction

Large-scale data repositories often suffer from duplicate entities whose representations are different, but yet refer to the same real world object. For instance, in digital libraries, there may be various name variants of an author due to errors introduced in data entry or errors from imperfect data collection softwares. Let us denote that, among the duplicate entities, the single authoritative one as *canonical entity*¹ while the rest as *variant entities* or *variants* in short.

*Contact author: dongwon@psu.edu. His research was partially supported by Microsoft SciData Award (2005) and IBM Eclipse Innovation Award (2006).

†His work was partially supported by a gift funding from Microsoft.

¹Our proposal is orthogonal to the issue of canonical entity since it works as long as one of entities is designated as the canonical one. Therefore, the issue of determining the *canonical entity* among many candidate entities is not pursued further in this paper.

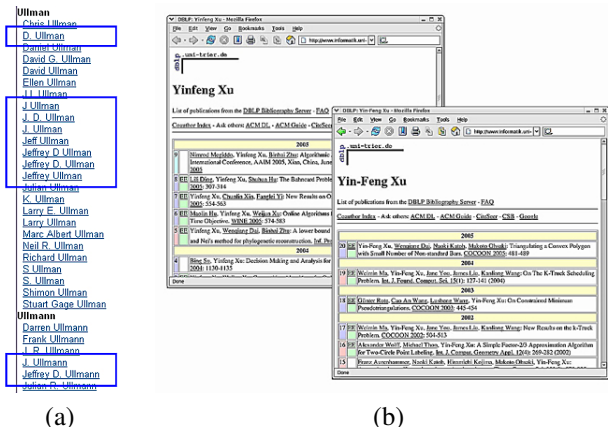


Figure 1. Real examples of the GER problem: (a) Split case of “J. D. Ullman” in ACM, and (b) Split case of “Yin-Feng Xu” in DBLP.

Since the existence of variant entities degrades the quality of the collection severely, it is important to de-duplicate them. Such a problem is, in general, known as the **Entity Resolution (ER)** problem. The ER problem frequently occurs in many applications and is exacerbated especially when data are integrated from heterogeneous sources. Note that the ER problem cannot be completely avoided since not all entities in data collections carry an ID system such as digital object ID (DOI).

In this paper, in particular, we focus on the ER problem where the goal is to detect all variant entities – what we call “split entities”² – and consolidate them into one entity. With an array of extensive research on the ER problem (to be surveyed in Section 6), in general, there are various efficient and effective methods to identify split entities. However, we observed that a particular type of entities occur quite common in real applications, and a more aggressive method can exploit the characteristics of the type better. That is, we noticed that many entities contain “a group of elements” in

²The other type of ER problem is to detect distinct entities from a pool of “mixed entities” due to homonyms. In general, mixed entity case can be casted as k -way clustering problem.

them. We refer to such an entity as the **Grouped-Entity** and the ER problem on grouped-entity as **Grouped-Entity Resolution (GER)** problem. Unlike a regular entity, a grouped-entity contains a wealth of information in its elements. How to unearth such hidden information for resolving grouped-entities is the focus of this paper. Throughout the rest of the paper, we simply use “entity” to refer to the grouped-entity. The GER problem frequently occurs in many situations as follows:

Example 1. Examples of grouped-entities include an author entity with a paper list or an actor entity with a movie list. Figure 1 illustrates the screen-shots of the GER problem from two real digital libraries – ACM and DBLP. Here, each grouped-entity (i.e., author) has a group of citations in it. Due to various errors, however, citations of the same author may be split under multiple author names. For instance, in the ACM, the citations of computer scientist “Jeffrey D. Ullman” appear under ten different name variants. If we take entity “Jeffrey D. Ullman” as the canonical one, then the other nine (e.g., “D. Ullman” and “J. Ullmann”) are variants, and should be consolidated. Similarly, in DBLP, a partial list of citations of “Yin-Feng Xu” appears under the variant entity “Yinfeng Xu.” □

Example 2. The US census bureau³ needs to keep track of people in families. To do this, they often use people’s names or known addresses. However, due to data-entry errors or confusing homonyms, the tracking is not always successful. Moreover, comparing two people by their names alone yields many false positives. Now, suppose one uses as *context* the family information of each person – two persons are similar if their “families” are similar, e.g., they share the similar family members such as names of spouse and children. That is, each grouped-entity (i.e., person) has a group of elements (i.e., family names). Then, in determining if “John Doe” is the same person of “Jonathan Doe,” for instance, one may combine the textual similarity, $sim(\text{“John Doe”}, \text{“Jonathan Doe”})$, as well as the contextual similarity – if the family of “John Doe” shares similar names with that of “Jonathan Doe.” If so, it is likely that “Jonathan Doe” is a name variant of “John Doe.” □

By and large, previous approaches to the GER problem (e.g., [4, 10, 6]) work as follows: (1) the information of an entity, e , is captured in a data structure, $D(e)$, such as a multi-attribute tuple or an entropy vector; (2) a binary distance function, f , is prepared; (3) the distance of two entities, e_1 and e_2 , is measured as that of the corresponding data structures, $D(e_1)$ and $D(e_2)$, using function f : $dist(e_1, e_2) = f(D(e_1), D(e_2))$; and (4) finally, if the result, $r = dist(e_1, e_2)$, is less than certain threshold, ϕ , then

³The example is derived from the discussion with Divesh Srivastava at AT&T Labs.

the two entities are variants: $r < \phi \rightarrow e_1 \sim e_2$. Although working well in many scenarios, this approach often suffers from a large number of *false positives* (i.e., an entity determined to be a variant when it is not). Consequently, the overall recall and precision suffer. If a user asks for top- k answers, such false positives can even override correct variants out of the answer window of $|k|$, degrading the precision substantially.

Example 3. Consider the example of Figure 1(a) again. Suppose the distance of two entities is measured as that of author name spellings themselves. If we use Edit distance, for instance, $dist(\text{“J. D. Ullman”}, \text{“J. Ullman”}) = 2$. If the threshold ϕ is set to 3, then “J. Ullman” can be correctly identified as a variant of “J. D. Ullman”. However, other entities such as “J. L. Ullman” or “K. Ullmann” whose distances are less than 3 will also be detected as variants. However, both are in fact *false positives*. Even if we use more sophisticated data structures to capture entities, the problem may still persist. For instance, suppose an author entity of Figure 1(a) is represented as a multiple-attribute tuple, $(coauthor, title, venue)$, where each attribute is a vector of tokens from the citations. That is, “J. D. Ullman” and “K. Ullman” are represented as $([Hopcroft, Aho], [Cube, Query], [KDD, ICDM])$ and $([Cruise, Kidman], [Mining, Query], [KDD, SIGMOD, ICDM])$, respectively. Then, depending on the distance function, it is possible that “K. Ullman” is identified as a variant of “J. D. Ullman” since both share many tokens. However, “K. Ullmann” is a false positive which happens to bear certain textual similarity to “J. D. Ullman” (since maybe both share research interests). □

The culprit of this false positive problem is in essence the use of distance functions that solely rely on the “textual similarity” of two entities, regardless of the adopted data structures. Toward this problem, in this paper, we present a novel graph partition based approach that boosts up precision significantly. We unearth the relationship hidden under the grouped-entities, and exploit it together with textual similarities. Experimental results using real and synthetic data sets validate our claim.

Our contributions are as follows:

- We formulate the GER problem as a specialized form of the ER problem. Since the grouped-entities in the GER problem contain a wealth of information (i.e., a group of elements), its exploitation can result in better outcome.
- We introduce how to capture “contextual information” hidden in a group of elements in grouped-entities. In particular, we propose to use the technique of *superimposition* to mine hidden relationships into graphs.

- To capture the “contextual distance” between grouped-entities, we exploit the notion of *Quasi-Clique*, a measure to see how strong inter-relationships between two graphs are, and propose a simple yet effective two-step GER algorithm, *distQC*.
- The proposed techniques of superimposed graphs and subsequent *Quasi-Clique*-based distance measurement are implemented and tested extensively against five test cases (1 real and 4 synthetic cases). Experiments verify that our proposal improves precision and recall up to 83% (when used with a variety of existing ER solutions) but never worsens them.

2 Problem Formulation

Typically, a group of elements in grouped-entities follow certain formats, and contain a lot of tokens, “not” all of which are directly indicative of the entity. For instance, the grouped-entity of “Yin-Feng Xu” of Figure 1(c) contains 20 citations, each of which contains co-authors, title, venue, year, etc. Then, some tokens in this long list are not real indicative of the entity “Yin-Feng Xu” and may confuse distance functions. Therefore, often, detecting variant grouped-entities yields a large number of false positives. Formally, the GER problem is:

Given a set of grouped-entities, E , where each contains a group of elements, for each canonical entity, $e_c (\in E)$, identify all variant entities, $e_v (\in E)$, such that $dist(e_c, e_v) < \phi$ with as few false positives as possible.

Note that our proposal works as long as “one” entity is selected as the canonical one.

3 General Idea

We consider a web of entities connected via relationships present in data sets. Our approach is based on the hypothesis that “if two entities are variants, there is a high likelihood that they are strongly connected to each other through a web of relationships, implicit in the database” [13]. In other words, among “J. D. Ullman,” “J. Ullman,” and “J. K. Ullman” entities, in order not to mistakenly mark “J. K. Ullman” as a variant of “J. D. Ullman” (i.e., no false positive), we may unearth the hidden relationships between two and exploit them (e.g., although both are Database researchers, the cores of their frequent co-author list are slightly different). If such relationships are first captured as some graphs, then subsequently the distance between two graphs can be used to measure the distance between two entities. Since graphs may contain richer information than

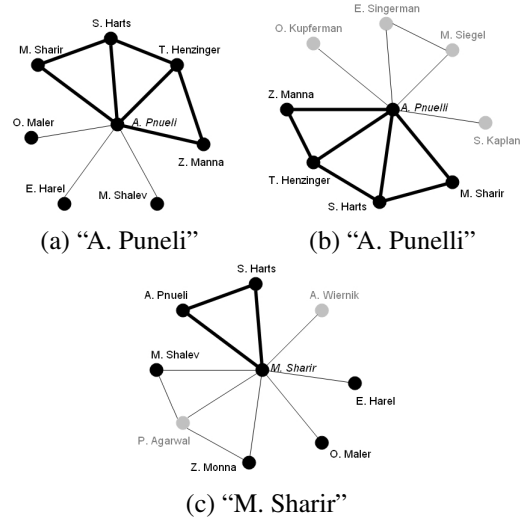


Figure 2. Graph representations of the “contexts” (i.e., co-authors) of three entities, $\{E_a, E_b, E_c\}$, and their superimposed quasi-cliques (solid lines). This *real* example is drawn from ACM data set.

simple strings or vectors, the corresponding distance computed from graphs are likely to help identify real variants.

3.1 Context as Additional Information

In the general ER problem (e.g., record linkage, reference resolution, name disambiguation), often, the underlying assumption is that there are some textual similarities among variant entities. For instance, to identify if two given records are duplicate or not (i.e., record linkage problem), one may apply either a token-based distance function (e.g., cosine) or an Edit distance based function (e.g., Jaro) to measure how similar two records are. However, in the situations where matching entities may “not” bear syntactical similarities, the general ER methods do not work well. On the other hand, an interesting observation from the previous studies is that a method often shows a good performance if it considers some “additional information” beyond textual similarities. We call this additional information as **context** in this paper. When contextual information is captured as a graph, let us call the graph as **context graph**.

In particular, note that grouped-entities have a group of elements in it. That is, each has abundant repetition of related tokens – a set of co-author names, a tokens of words used in paper titles, or a set of venue names to which they submit often, etc. We may use these as *context* and measure the corresponding contextual similarity. Let us first see an example where the notion of context is useful.

Consider the real cases of three entities from ACM data set, E_a (“A. Puneli” entity), E_b (“A. Punelli” entity), and E_c (“M. Sharir” entity), of which E_a and E_b are name vari-

ants and E_c is a false positive of E_a . Each entity has the following co-authors in their group of elements:

- $E_a = \{\text{“M. Sharir”, “S. Harts”, “T. Henzinger”, “Z. Manna”, “M. Shalev”, “E. Harel”, “O. Maler”}\}$
- $E_b = \{\text{“M. Sharir”, “S. Harts”, “T. Henzinger”, “Z. Manna”, “O. Kupferman”, “S. Kaplan”, “E. Singerman”, “M. Siegel”}\}$
- $E_c = \{\text{“A. Pnueli”, “S. Harts”, “Z. Monna”, “A. Wiernik”, “P. Agarwal”, “M. Shalev”, “E. Harel”, “O. Maler”}\}$

If we draw graphs where the entity itself becomes the center node and its co-authors become neighboring nodes attached to the center node, then we get Figure 2. Furthermore, if there are known co-authorship among co-authors, edges are created between them. For instance, “S. Harts” and “T. Henzinger” co-authored elsewhere (i.e., other than in entities E_a , E_b , and E_c), and thus an edge connecting them is created in Figure 2(a).

First, suppose we compare three entities by counting how many common co-authors they have. Then, E_b and E_c have 5 and 7 common co-authors with E_a , respectively. Therefore, if we use distance metrics that are based on the number of common tokens of two entities such as Jaccard distance, then E_c would be probably returned as a variant of E_a over E_b . However, this is wrong, and we have a case of false positive. Second, if we compare three entities in terms of how large the maximum common subgraph (where all vertices and adjacent edges match) is, then E_b and E_c have a common subgraph with 5 and 3 vertices with E_a , respectively. Therefore, E_b would be returned as a variant of E_a over E_c – the opposite result of previous case.

Note that entities E_a and E_b have four common co-authors, $\{\text{“M. Sharir”, “S. Harts”, “T. A. Henzinger”, “Z. Manna”}\}$, who are all well connected among themselves. This can be used as a clue to unearth the hidden similarity between the two entities, E_a and E_b . On the other hand, E_c shares only small-sized well-connected subgraph although overall it has more number of common co-authors. In other words, if we look at the relationships existing in the whole graph, instead of individual vertices, then we may be able to detect that E_c is not a variant of E_a .

3.2 Quasi-Clique

Once hidden relationships among two entities are captured as two context graphs, one needs a way to measure distance between two graphs. For this purpose, we propose to use a graph mining technique using the notion of *Quasi-Clique*.

Given a graph G , let $V(G)$ and $E(G)$ be the sets of vertices and edges in the graph, respectively. Let $U \subseteq V(G)$

be a subset of vertices. The *subgraph induced on U* , denoted by $G(U)$, is the subgraph of G whose vertex-set is U and whose edge-set consists of all edges in G that have both endpoints in U , i.e., $G(U) = (U, E_U)$, where $E_U = \{(u, v) | (u, v) \in E(G) \wedge u, v \in U\}$.

Definition 1 (γ -Quasi-Clique) A connected graph G is a γ -Quasi-Clique graph ($0 < \gamma \leq 1$) if every vertex in the graph has a degree at least $\gamma \cdot (|V(G)| - 1)$. Clearly, a 1-Quasi-Clique graph is a complete graph (i.e., clique). In a graph G , a subset of vertices $S \subseteq V(G)$ is a γ -Quasi-Clique ($0 < \gamma \leq 1$) if $G(S)$ is a γ -Quasi-Clique graph, and no proper superset of S has this property. \square

As shown in [20], an interesting property of *Quasi-Cliques* is that when γ is not too small, a γ -Quasi-Clique is compact – the diameter of the *Quasi-Clique* is small. That is, the diameter of G , $D(G)$, is:

$$D(G) \begin{cases} = 1 & \text{if } 1 \geq \gamma > \frac{n-2}{n-1} \\ \leq 2 & \text{if } \frac{n-2}{n-1} \geq \gamma \geq \frac{1}{2} \\ \leq 3 \lfloor \frac{n}{\gamma(n-1)+1} \rfloor - 3 & \text{if } \frac{1}{2} > \gamma \geq \frac{2}{n-1} \text{ and} \\ & n \bmod (\gamma(n-1)+1) = 0 \\ \leq 3 \lfloor \frac{n}{\gamma(n-1)+1} \rfloor - 2 & \text{if } \frac{1}{2} > \gamma \geq \frac{2}{n-1} \text{ and} \\ & n \bmod (\gamma(n-1)+1) = 1 \\ \leq 3 \lfloor \frac{n}{\gamma(n-1)+1} \rfloor - 1 & \text{if } \frac{1}{2} > \gamma \geq \frac{2}{n-1} \text{ and} \\ & n \bmod (\gamma(n-1)+1) \geq 2 \\ \leq n-1 & \text{if } \gamma = \frac{1}{n-1} \end{cases}$$

The upper bounds are realizable. This property gives *Quasi-Cliques* a lift to be a good data structure to identify compact subgraphs.

In a network (e.g., a social network or a citation network) scenario, a *Quasi-Clique* is a set of objects that are highly interactive with each other. Therefore, a *Quasi-Clique* in a graph may strongly indicates the existence of a potential community. Since a *Quasi-Clique* contains a group of highly interacting (and thus likely highly similar in role) objects, it may be more reliable in representing relationships than individual objects. Heuristically, we can use *Quasi-Cliques* to annotate the relationships in large scale subgraphs, which is highly desirable for solving the GER problem.

While γ value indicates the compactness of *Quasi-Clique*, another parameter that is of interest is the number of vertices of *Quasi-Clique*. We denote this parameter as \mathcal{S} . For a graph G , therefore, functions:

- $\text{QC}(G, \gamma, \mathcal{S})$ returns a γ -Quasi-Clique graph g from G with $|V(g)| \geq \mathcal{S}$ if it exists, and
- $\text{QC}(G_1, G_2, \gamma, \mathcal{S})$ returns a common γ -Quasi-Clique graph g of G_1 and G_2 with $|V(g)| \geq \mathcal{S}$.

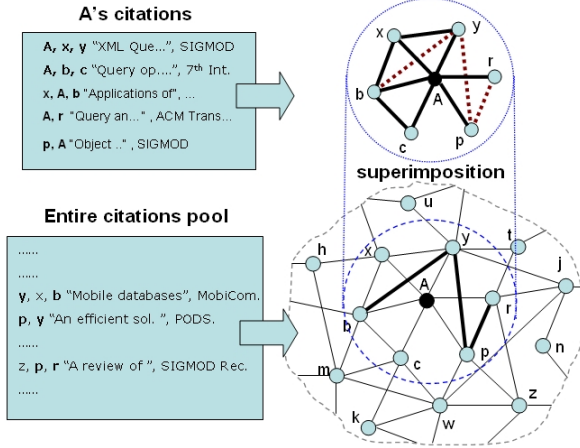


Figure 3. Illustration of “superimposition” of co-author data set. After superimposition, three new edges, (b, y) , (y, p) , and (p, r) , are added to the context graph (top), where solid edges are due to A ’s co-authors while dotted edges are due to superimposition.

Then, $|g|$ indicates how strongly two graphs G_1 and G_2 are related, and can be used as a “distance.” This function $QC(G_1, G_2, \gamma, \mathcal{S})$ is used to measure the contextual distance in our `distQC` algorithm in Section 4.2.

4 The Two-Step GER Algorithm

Based on the ideas of Section 3, here, we introduce our proposal of using graph-based partition to tackle the GER problem. We capture contextual information in context graphs through superimposition (step 1), and measure their contextual similarity in terms of *Quasi-Clique* (step 2).

4.1 Step 1: Mining Context Graphs

Suppose we want to mine a graph out of an author entity A ’s co-author tokens: B, C, D , and E . First, a vertex is prepared for tokens, A through E , referred to as $V(A)$ through $V(E)$. Then, four co-author vertices, $V(B)$ through $V(E)$, are connected to the main vertex $V(A)$, forming a graph, g_a . Next, g_a is “superimposed” to the collaboration graph, G , that is pre-built using the entire set of co-authors from all entities. For instance, if an author C had co-authored with an author D elsewhere, then now g_a will have an edge connecting $V(C)$ and $V(D)$. At the end, if all neighboring co-authors of A have co-authored each other, then g_a becomes a clique. This process is illustrated in Figure 3. Note that the pre-built collaboration graph works as the *base graph*.

Similarly, for venue information, once we create an initial graph, g_a , we can superimpose it against a base graph.

Algorithm 1: `distQC`

Input: A grouped-entity e , an ER method \mathcal{M} , and three parameters $(\alpha, \gamma$ and $\mathcal{S})$.

Output: k variant grouped-entities, $e_v (\in E)$, such that $e_v \sim e$.

- 1 Using \mathcal{M} , find top $\alpha \times k$ candidate entities, e_X ;
 - 2 $G_c(e) \leftarrow$ context graph of e ;
 - 3 **forall** $e_i (\in e_X)$ **do**
 - 4 $G_c(e_i) \leftarrow$ context graph of e_i ;
 - 5 $g_i \leftarrow QC(G_c(e), G_c(e_i), \gamma, \mathcal{S})$;
 - 6 Sort $e_i (\in e_X)$ by $|g_i|$, and return top- k ;
-

For instance, one may use, as a base graph, a venue relation graph where an edge between two venue vertices represents the “semantic” similarity of two venues (e.g., how many authors have published in both venues). The superimposition works as long as there is a base graph (e.g., collaboration graph, venue relation graph) onto which an entity’s graph can be superimposed. For more general cases, a base graph can be also constructed using the co-occurrence relationship among tokens.

4.2 Step 2: Measuring Contextual Distance

Once the contexts of entities are captured and represented as graphs, their similarity can be properly modeled using *Quasi-Clique*. Two factors matter here. First, Since parameter γ indicates the compactness of a γ -*Quasi-Clique*, two entities in a *Quasi-Clique* with a higher γ value are more similar to each other than two in a *Quasi-Clique* with a lower γ value. Second, two entities in a *Quasi-Clique* with more vertices suggests that the two entities are very “similar” not only in their vertices but also in their relationships. For instance, in the *Quasi-Clique* jargon, in Figure 2, there is a 0.5-*Quasi-Clique* between E_a and E_b with at least 4 vertices. However, there exists clearly no common 0.5-*Quasi-Clique* between E_a and E_c , which prevents us from concluding that E_c is a variant entity.

Since one of the motivations of this work is to reduce false positives, we can adopt the *Quasi-Clique* into a two-step GER algorithm, `distQC`, as shown in Algorithm 1. Given an entity $e (\in E)$, to locate matching k variant entities, the `distQC` algorithm first relies on any existing ER solutions, and selects α times more number of candidates than k as an extra. Since we try to improve precisions by reducing false positives, once we get more candidates, if we can boost up those correct variants up into higher ranks in the subsequent step, then our aim can be met.

Table 1. Summary of data sets.

Data set	Domain	# of grouped-entities	# of elements in all entities
ACM	Comp. Sci.	707,368	1,037,968
BioMed	Medical	6,169	24,098
IMDB	Entertainment	446,016	935,707

5 Experimental Validation

5.1 Set-up

For validating our proposal, we first gathered three real data sets from diverse domains – ACM, BioMed, and IMDB – as shown in Table 1.

Real Case. From ACM data set, we have manually identified 47 real cases of canonical and variant grouped-entities by contacting authors or visiting their home pages. Each case has one designated canonical entity and on average two variant entities. Furthermore, each grouped-entity has on average about 24.6 elements in it (i.e., each author has 24.6 citations). Real cases include variants as simple as “Dongwon Lee” vs. “D. Lee” (i.e., abbreviation) and “Alon Levy” vs. “Alon Halevy” (i.e., last name change) to as challenging as “Shlomo Argamon” vs. “Sean Engelson” (i.e., little similarity) or even ten variants of “Jeffrey D. Ullman” of Figure 1. Since there are a total of 707,368 entities in ACM data set, locating all k variants correctly without yielding any false positives is a challenging task.

Synthetic Case. We synthetically generated a test case for each data set as follows. For each data set, we first pick 100 grouped-entities that has at least 10 elements in them. Then, we made up a variant entity by either abbreviating the first name or injecting invalid characters to the last name in 7:3 ratio. Then, both canonical and variant entities carry halves, 133, of the original elements. The goal is then to identify the artificially generated variant out of the entire entities – e.g., 707,368 in ACM and 446,016 in IMDB data sets. For the details, please refer to [18].

Evaluation Metrics. Suppose there are R variants hidden. When top- k candidate variants are returned by an algorithm, furthermore, suppose that only r candidates are correct variants and the remaining $k - r$ candidates are false positives. Then, $Recall = \frac{r}{R}$ and $Precision = \frac{r}{k}$. Since this traditional recall and precision do not account for the quality of rankings in the answer window k , the *Ranked Precision* metric measures the precision at different cut-off points [12]. For instance, if the topmost candidate is a correct variant while the second one is a false positive, then we have 100% precision at a cut off of 1 but 50% precision at a cut off of 2. In our context, we dynamically set the cut-off points, C , to be the same as R . That is, if we know that there are 4 variants to identify, we set $C = 4$. Formally,

Table 2. Summary of notation.

Notation	Meaning
JC	Jaccard
TI	TFIDF
IC	IntelliClean
QC	<i>Quasi-Clique</i>
JC+QC	Jaccard + <i>Quasi-Clique</i>
TI+QC	TFIDF + <i>Quasi-Clique</i>
IC+QC	IntelliClean + <i>Quasi-Clique</i>

$Ranked\ Precision = \frac{\sum_{i \in C} precision_i}{r}$, where $precision_i$ is the precision at a cut off of i . Finally, the *Average Recall (AR)* and *Average Ranked Precision (ARP)* are the averages of recall and ranked precision over all cases (i.e., 47 cases for real ACM test set and 100 cases for each of the four synthetic test sets).

Compared Methods. Recall that the goal of the proposed two-step algorithm is to avoid false positives, and thus improve overall precision and recall. To see the effectiveness of our *distQC*, therefore, we conducted the most effective distance metrics in the first step to get initial candidate variants. Then, in the second step, we applied the *Quasi-Clique*-based metric to the candidate variants to get the final answers. Then, we compared the performance of “before” and “after” *Quasi-Clique*-based metric was applied.

In the first step, we experimented the following distance metrics: *Jaccard*, *TF/IDF*, and *IntelliClean (IC)*. Due to the space limitation, we omit the definitions of each metric (please refer to [23] [15]).

At the end, we have implemented *distQC* and *IntelliClean* in Java, borrowed implementations of *TFIDF*, *Jaccard*, etc. from *SecondString* [23], and *Quasi-Clique* from [20]. Table 2 summarizes notations of methods evaluated in the experimentation.

5.2 Results

1. *Quasi-Clique* on ACM real case. We experiment to see if *Quasi-Clique* can improve the performance of one-step distance functions. We first ran and measured the performance of three distance methods *Jaccard (JC)*, *TFIDF (TI)* and *IntelliClean (IC)*. Then, to each, *Quasi-Clique* is applied as a second step, and the performance is measured as *JC+QC*, *TI+QC*, and *IC+QC*. Hierarchical clustering algorithm is used to generate a concept hierarchy for *IC*. Figure 4(a-b) illustrate *AR* and *ARP* of these six schemes. Note that *Quasi-Clique* improved the precision visibly. For instance, the precision of *JC+QC* (resp. *TI+QC*) significantly improves from *JC* (resp. *TI*) on co-authors. On average, precision improves by 63%, 83%, and 46% for three attributes, respectively.

In general, JC and TI are simple distance metrics, measuring distances based on the occurrences of common tokens. Since some authors have name initials and common first names on co-author data, therefore, these metrics tend to generate a large number of false positives as shown in Figure 4(a-b). Since *Quasi-Clique* uses additional information as to how closely co-authors of the authors are correlated each other on graph representation, it overcomes the limitations of the simple metrics.

2. Quasi-Clique on Synthetic cases. We experiment to see how effective *Quasi-Clique* is against large-scale data sets with synthetically generated solution sets. For this purpose, we prepared three data sets with diverse domains – ACM, BioMed, and IMDB. Figure 4(c-h) illustrate the final results of AR and ARP for all data sets. Regardless of the distance function used in step 1 (either JC or TI), type of attributes (co-authors, titles or venues), or type of data sets, *distQC* consistently improved the ARP up to 75% (BioMed/title case).

In particular, Figure 4(g-h) illustrate the performance result of IMDB – a movie data set with attributes like movie ID, country, distributor, editor, genre, keyword, language, location, producer, production company, and release date. Then, we selected three attributes, “location”, “production companies”, and “release date”, as these have more contextual information than other attributes (e.g., French actors tend to star french movies more often)⁴.

Figure 4(g-h) illustrate AR and ARP of JC and TI, and their two-step correspondents, JC+QC and TI+QC. Overall, the location attribute shows relatively lower AR and ARP than release date and production companies, in all of the methods. This is because there are a lot of uncommon tokens such as the city “Victorville” in the location attribute. Nevertheless, *distQC* improved the ARP by 13%, suggesting its effectiveness. Both AR and ARP of TI are low, compared to those of JC. This is because there are common attribute values, such as “USA” in location, and “pictures” or “movies” in production companies. In TI, for weighting tokens, common tokens (e.g., “movies”) would have lower weight via IDF, negatively affecting matching process. In AR, our *distQC* methods show much higher than both JC and TI.

Compared with citation data sets, note that our *distQC* algorithm performs slightly better than string distance metrics in IMDB data set. This is because (1) records and attribute values of an actor and his/her variant entities have no strong relationships unlike those of citations; (2) attribute values of citations are long while those of IMDB data set are short, carrying not-so-rich information; (3) many attributes in IMDB data set contains empty values and noises.

⁴The “location” shows where the movie was primarily made. Although the location may not show strong correlation for superstars, majority of actors tend to appear in movies of only a few locations (e.g., country).

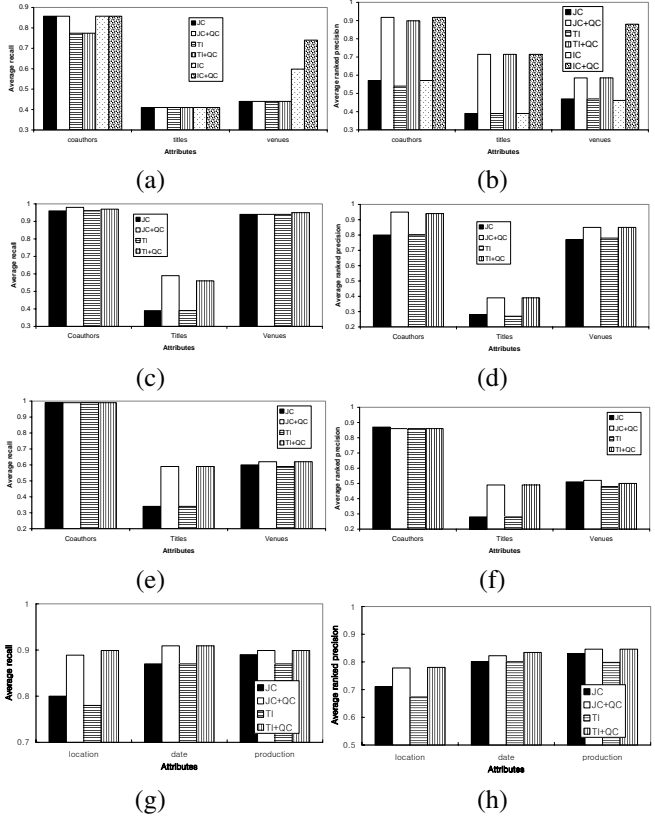


Figure 4. Test cases for three data sets: (a-b) real cases for ACM data set, and (c-d), (e-f), and (g-h) synthetic cases for ACM, BioMed, and IMDB data sets, respectively.

6 Related Work

The general ER problem has been known as various names – record linkage (e.g., [8, 4]), citation matching (e.g., [17]), identity uncertainty (e.g., [19]), merge-purge (e.g., [10]), object matching (e.g., [6]), duplicate detection (e.g., [21, 1]), authority control (e.g., [24, 11]), and approximate string join (e.g., [9]) etc. When entities are “grouped-entities,” these methods do not distinguish them, while our *distQC* tries to exploit them using *Quasi-Clique*. Our method can be used together with any of these methods as the first step.

Bilenko et al. [4] have studied name matching for information integration using string-based and token-based methods. Cohen et al. [7] have also compared the efficacy of string-distance metrics, like JaroWinkler, for the name matching task. [14] experimented with various distance-based algorithms for citation matching, with a conclusion that word based matching performs well. We have implemented all these methods in our framework and compared

how much our `distQC` improves when used together. Before we process each element in a grouped-entity (e.g., citation or movie record), we assume that field segmentation and identification have been completed using some methods (e.g., [5]). ALIAS system in [21] proposes a framework to detect duplicate entities such as citations or addresses, but its focus is on the learning aspect.

Unlike the traditional methods exploiting textual similarity, Constraint-Based Entity Matching (CME) [22] examines “semantic constraints” in an unsupervised way. They use two popular data mining techniques, Expectation-Maximization (EM) and relaxation labeling for exploiting the constraints. [2] presents a generic framework, *Swoosh* algorithms, for the entity resolution problem. Our `distQC` can be used as a type of distance metric in their framework.

The recent trend in the ER problem shows similar direction to ours (e.g., [4, 3, 16, 13]) – Although each work calls its proposal under different names, by and large, most are trying to “exploit additional information beyond string comparison.” A more extensive and systematic study is needed to investigate the usefulness and limitations of the context in a multitude of the ER problem. In this paper, we especially exploit the graph-based partitioning technique to capture “contexts.” Recent work by Kalashnikov et. al. [13] presents a relationship-based data cleaning (ReIDC) which exploits context information for entity resolution, sharing similar idea to ours. ReIDC constructs a graph of entities connected through relationships and compares the connection strengths across the entities on the graph to determine correspondences. The main difference is the notion of data structure used (i.e., we used *Quasi-Clique* along with superimposition to derive distances between graphs).

7 Conclusion

Toward the grouped-entity resolution problem, we present a graph partition based approach using *Quasi-Clique*. Unlike string distance or vector-based cosine metric, our approach examines the relationship hidden under the grouped-entities. Experimental results verify that our proposed approach improves precision up to 83% at best (in Figure 4(a-d)), but never worsens it.

References

- [1] R. Ananthkrishna, 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] I. Bhattacharya and L. Getoor. “Iterative Record Linkage for Cleaning and Integration”. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2004.
- [4] 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.
- [5] V. R. Borkar, K. Deshmukh, and S. Sarawagi. “Automatic Segmentation of Text into Structured Records”. In *ACM SIGMOD*, Santa Barbara, CA, May 2001.
- [6] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani. “Robust and Efficient Fuzzy Match for Online Data Cleaning”. In *ACM SIGMOD*, 2003.
- [7] W. Cohen, P. Ravikumar, and S. Fienberg. “A Comparison of String Distance Metrics for Name-matching tasks”. In *IJWeb Workshop held in conjunction with IJCAI*, 2003.
- [8] I. P. Fellegi and A. B. Sunter. “A Theory for Record Linkage”. *J. of the American Statistical Society*, 64:1183–1210, 1969.
- [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] M. A. Hernandez and S. J. Stolfo. “The Merge/Purge Problem for Large Databases”. In *ACM SIGMOD*, 1995.
- [11] Y. Hong, B.-W. On, and D. Lee. “System Support for Name Authority Control Problem in Digital Libraries: OpenDBLP Approach”. In *European Conf. on Digital Libraries (ECDL)*, Bath, UK, Sep. 2004.
- [12] D. Hull. “Using Statistical Testing in the Evaluation of Retrieval Experiments”. In *ACM SIGIR*, pages 329–338, Pittsburgh, 1993.
- [13] D. V. Kalashnikov, S. Mehrotra, and Z. Chen. “Exploiting Relationships for Domain-independent Data Cleaning”. In *SIAM Data Mining (SDM) Conf.*, 2005.
- [14] S. Lawrence, C. L. Giles, and K. Bollacker. “Digital Libraries and Autonomous Citation Indexing”. *IEEE Computer*, 32(6):67–71, 1999.
- [15] M. L. Lee, W. Hsu, and V. Kothari. “Cleaning the Spurious Links in Data”. *IEEE Intelligent System*, 19(2):28–33, 2004.
- [16] B. Malin. “Unsupervised Name Disambiguation via Social Network Similarity”. In *SIAM SDM Workshop on Link Analysis, Counterterrorism and Security*, 2005.
- [17] 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.
- [18] B.-W. On, D. Lee, J. Kang, and P. Mitra. “Comparative Study of Name Disambiguation Problem using a Scalable Blocking-based Framework”. In *ACM/IEEE Joint Conf. on Digital Libraries (JCDL)*, Jun. 2005.
- [19] 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.
- [20] J. Pei, D. Jiang, and A. Zhang. “On Mining Cross-Graph Quasi-Cliques”. In *ACM KDD*, Aug. 2005.
- [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] SecondString: Open source Java-based Package of Approximate String-Matching. <http://secondstring.sourceforge.net/>.
- [24] J. W. Warnner and E. W. Brown. “Automated Name Authority Control”. In *ACM/IEEE Joint Conf. on Digital Libraries (JCDL)*, 2001.