# Interactive Entity Resolution in Relational Data: A Visual Analytic Tool and Its Evaluation

Hyunmo Kang, *Member*, *IEEE Computer Society*,
Lise Getoor, *Member*, *IEEE Computer Society*,
Ben Shneiderman, *Member*, *IEEE Computer Society*,
Mustafa Bilgic, *Student Member*, *IEEE Computer Society*, and
Louis Licamele, *Student Member*, *IEEE Computer Society*

**Abstract**—Databases often contain uncertain and imprecise references to real-world entities. Entity resolution, which is the process of reconciling multiple references to underlying real-world entities, is an important data cleaning process required before accurate visualization or analysis of the data is possible. In many cases, in addition to noisy data describing entities, there is data describing the relationships among the entities. This relational data is important during the entity resolution process; it is useful both for the algorithms that determine likely database references to be resolved and for visual analytic tools that support the entity resolution process. In this paper, we introduce a novel user interface, D-Dupe, for interactive entity resolution in relational data. D-Dupe effectively combines relational entity resolution algorithms with a novel network visualization that enables users to make use of an entity's relational context for making resolution decisions. We describe resolution strategies based on pairs or sets of references and show appropriate visualizations for each. Since resolution decisions often are interdependent, D-Dupe facilitates understanding this complex process through animations that highlight combined inferences and a history mechanism that allows users to inspect chains of resolution decisions. An empirical study with 12 users confirmed the benefits of the relational context visualization on the performance of entity resolution tasks in relational data in terms of time as well as users' confidence and satisfaction.

**Index Terms**—Information visualization, visual analytics, data and knowledge visualization, data mining, graphical user interfaces, user-centered design.

## 1 Introduction

Databases often contain uncertain and imprecise references to real-world entities. The absence of unique identifiers for the underlying entities frequently results in databases that contain multiple references to the same entity. This can lead not only to data redundancy and inconsistency but also to inaccuracies in information visualization, query processing, and knowledge extraction. Entity resolution, which is the data cleaning process of reconciling multiple references to underlying real-world entities, is an active research topic within a number of fields including information visualization, database management, data mining, and machine learning.

In many cases, in addition to the attributes such as name, address, and so on, which describe the entities themselves, there are relationships among the entities, and these relationships can be useful during the entity resolution process. The relationships are often co-occurrence relationships, for example, email addresses may co-occur in to: and cc: lists, author names co-occur in the author lists of scientific papers, and actors are related to each other in social networks. We refer to this type of entity resolution problem as *relational entity resolution*, to highlight the importance of using the relationships among the entities, in addition to the available information about each entity's attributes. One of the research questions that we address is how to visually present this relational information to users during the entity resolution process.

Most existing entity resolution methods focus on automated entity resolution. Automated techniques are not perfect, and they face a precision-recall trade-off. If they are tuned to have a high precision, they rarely merge duplicates, leaving many duplicates in the database. If they are tuned to have a high recall, they mistakenly merge references that are in fact distinct. By contrast, handcleaning methods, even with visualization support, can be slow and inefficient in finding duplicates. These approaches tend to be high precision, because there is a human-in-the-loop making the final resolution decision. However, inspecting a large data set and hunting for duplicates can be like looking for the proverbial needle in a haystack. Thus, while these approaches may have high precision, they tend to have low recall.

In this paper, we present D-Dupe (http://www.cs.umd.edu/linqs/ddupe), a visual analytic tool, which supports relational entity resolution. D-Dupe provides an interactive analyst-centric approach to the problem, which tightly integrates entity resolution algorithms with a visualization suited to the task. D-Dupe provides access to sophisticated entity resolution algorithms and enables users to flexibly combine them to uncover duplicates. A variety of similarity measures can be used and their results can be combined in an extremely flexible manner. In addition, D-Dupe provides users with a simple network visualization, which displays the

- *H. Kang is with the Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742.
E-mail: kang@cs.umd.edu, hyunmo.kang@gmail.com.*
- *L. Getoor, B. Shneiderman, M. Bilgic, and L. Licamele are with the Department of Computer Science, University of Maryland, College Park, MD 20742. E-mail: {getoor, ben, mbilgic, licamele}@cs.umd.edu.*

*Manuscript received 28 Sept. 2007; revised 18 Jan. 2008; accepted 21 Feb. 2008; published online 17 Mar. 2008.
Recommended for acceptance by J.T. Stasko.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2007-09-0148.
Digital Object Identifier no. 10.1109/TVCG.2008.55.*

relational context for potential duplicates. The relational context viewer shows, for any two potential duplicates, their relational neighborhood. The network visualization allows users to quickly identify shared and nonshared relational context and base their exploration and resolution decisions on the context. Emerging principles from information visualization, such as laying out the nodes on a meaningful substrate, are combined with representations for uncertainty, resulting in a tool that is especially well suited to the entity resolution task. Powerful filtering and search techniques are also integrated into the tool. One of the findings from our user study is that users are faster and have more confidence in their resolution decisions when they are able to see the relational context.

One of the properties of relational entity resolution is that resolution decisions are often chained together and depend on each other. For example, once users determine that two author references refer to the same underlying entity, that then provides new evidence, which makes their coauthor references have more in common, and hence, the coauthors are more likely to also be coreferent. D-Dupe provides a lookahead mechanism for viewing the potential dependencies, before users commit to them, which helps users understand the potential impact of their decisions. D-Dupe also has a flexible history mechanism that allows users to see the chains and undo earlier decisions if needed.

D-Dupe's tight integration of relational entity resolution and simple interactive interface provides a scalable approach to semiautomated data cleaning. References can be resolved on a pairwise basis, for each instance of a reference, or for all references matching user-specified criteria. If users prefer more automation, all references with a similarity above a given threshold can be automatically merged in bulk.

Our initial work on a prototype system was described in [27]. Since then, we have completely redesigned and reimplemented the system, adding many additional features, not all of which are described in this paper. This paper describes the new version, D-Dupe v2.0. Some of the most important new contributions include

1. support for both data deduplication and data integration;
2. support for a variety of resolution strategies, including pairwise, cluster-based, and bulk;
3. support for visualization of chains of resolutions;
4. history and lineage support.

In addition, an important contribution of this paper is a user study, which examined the performance of 12 users in terms of speed, success rates, and confidence both with and without the graphical relational context viewer.

This paper is organized as follows: We begin with a discussion of related work in Section 2. Section 3 provides a brief overview of the design principles and overall D-Dupe interface and describes the entity resolution process at a high level. Section 4 describes the data model supported in D-Dupe. Section 5 goes into detail about the design of the tool, highlighting the powerful integration of machine learning and visualization methods for entity resolution, the novel pairwise task-specific visualization, which is particularly well-suited to the entity resolution problem, and the support for understanding the complex interactions possible in relational entity resolution. Section 6 describes the system architecture, and Section 7 provides a detailed description of a quantitative user study of D-Dupe, which studies the impact of the D-Dupe interface on users' speed, success rates, and confidence on the relational entity resolution task. Section 8 concludes with a discussion of future directions.

## 2 Related Work

### 2.1 Entity Resolution Approaches

The entity resolution problem has been studied in many different areas under different names such as deduplication, record linkage, coreference resolution, reference reconciliation, object consolidation, and others. Within the machine learning community, most of this work has focused on automatic methods rather than interactive support. Much of the work has concentrated on entity resolution based on attributes of the entity references. Extensive research has been done on defining approximate string similarity measures [1], [10], [26], [31] that may be used for unsupervised entity resolution.

A number of recent approaches take relations into account for entity resolution and data integration [5], [15], [17], [29], [30], [41]. The relational methods have been shown to increase performance significantly over the attribute-based solutions for the same sets. However, they pay a price in terms of computational complexity. The similarity computations become more expensive, and in the case where the dependence of the resolution decisions is taken into account, iterative solutions that make multiple passes over the data are required.

Unlike these fully automatic methods, D-Dupe allows users to use a mix of strategies for resolving duplicates. Users can define a variety of similarity measures that are based solely on attribute values, and D-Dupe will visually show the relational context for selected pairs; users can also define more complex similarity measures that take into account the overlapping neighbors when searching for potential duplicates. Users have the choice of inspecting each potential duplicate or automatically merging all duplicates with a similarity score above a user-selected threshold.

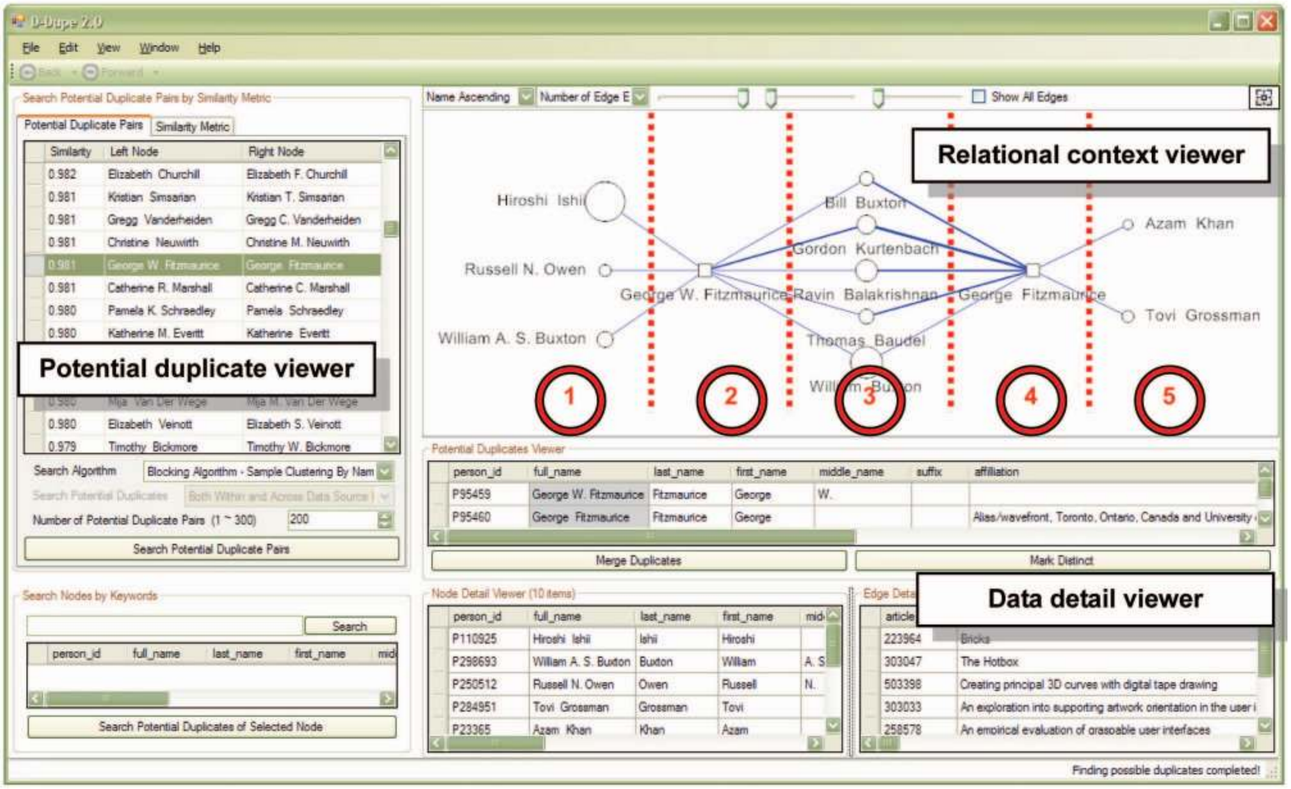
### 2.2 Interactive Data Cleaning

Within the database community, there has been work on interactive data cleaning [36], [38], and there is an increasing number of commercial tools, which support interactive data cleaning. The research and tools allow users to define rules for correcting mistakes in data and allow users to inspect and resolve mistakes in the data. However none of these tools support resolving references, which are linked via relationships. The tools use tabular views, which show only the attribute values of the potential duplicates, and there has been little research into appropriate visualizations.

Within the machine learning community, there has been work on automatically learning similarity functions based on user feedback [23], [32], [34]. This work assumes there is a single similarity measure that will be accurate across all instances. D-Dupe takes an alternate approach, which allows users to use a variety of similarity metrics for ranking potential duplicates and combine the results as appropriate.

### 2.3 Network Visualization

Visual analysis of networks is an integral component of the field of social network analysis and is in many ways fundamental to its very definition [22]. There are several excellent surveys describing social network visualization



| Similarity | Left Node | Right Node |
|---|---|---|
| 0.982 | Elizabeth Churchill | Elizabeth F. Churchill |
| 0.981 | Kristian Simsarian | Kristian T. Simsarian |
| 0.981 | Gregg Vanderheiden | Gregg C. Vanderheiden |
| 0.981 | Christine Neuwirth | Christine M. Neuwirth |
| 0.981 | George W. Fitzmaurice | George Fitzmaurice |
| 0.981 | Catherine R. Marshall | Catherine C. Marshall |
| 0.980 | Pamela K. Schraedley | Pamela Schraedley |
| 0.980 | Katherine M. Everitt | Katherine Everitt |
| 0.980 | Mija Van Der Wege | Mija M. Van Der Wege |
| 0.980 | Elizabeth Veinott | Elizabeth S. Veinott |
| 0.979 | Timothy Bickmore | Timothy W. Bickmore |

Search Algorithm: Blocking Algorithm - Sample Clustering By Nam

Search Potential Duplicates: Both Within and Across Data Source

Number of Potential Duplicate Pairs (1 ~ 300): 200

Search Potential Duplicate Pairs

| person_id | full_name | last_name | first_name | mid |
|---|---|---|---|---|

| person_id | full_name | last_name | first_name | middle_name | suffix | affiliation |
|---|---|---|---|---|---|---|
| P95459 | George W. Fitzmaurice | Fitzmaurice | George | W. | | |
| P95460 | George Fitzmaurice | Fitzmaurice | George | | | Alias/wavefront, Toronto, Ontario, Canada and University |

| person_id | full_name | last_name | first_name | mid |
|---|---|---|---|---|
| P110925 | Hiroshi Ishii | Ishii | Hiroshi | |
| P298693 | William A. S. Buxton | Buxton | William | A. S |
| P250512 | Russell N. Owen | Owen | Russell | N. |
| P284951 | Tovi Grossman | Grossman | Tovi | |
| P23365 | Azam Khan | Khan | Azam | |

| article | |
|---|---|
| 223964 | Bricks |
| 303047 | The Hotbox |
| 503398 | Creating principal 3D curves with digital tape drawing |
| 303033 | An exploration into supporting artwork orientation in the user i |
| 258578 | An empirical evaluation of graspable user interfaces |

Fig. 1. D-Dupe consists of three coordinated windows: the potential duplicate viewer on the left, the relational context viewer on the upper right corner, and the data detail viewer on the lower right corner. The potential duplicate viewer shows the list of potential duplicate author pairs that were identified based on the user-defined similarity metric. The relational context viewer visualizes the coauthorship relation between the potential duplicate author pair selected in the potential duplicate viewer. The data detail viewer shows all the attribute values of the nodes (authors) and edges (papers) displayed in the relational context viewer.

within the social sciences literature [21], [33] and the information visualization literature [9], [18], [37]. In addition, several websites [2], [39] show a rich variety of network visualization examples, and there are many useful social network software packages that focus on interactive visualization of the complete network [6], [19], [20], [23], [28]. While we make use of network visualization in D-Dupe, our work is quite distinct since we focus on visualizing the relationships between pairs of references rather than the complete network.

## 3 Interface Design Principles

The challenge of entity resolution in large relational data requires an interface that provides tight integration of statistical data mining algorithms, meaningful presentation of carefully selected subnetworks, and ready access to rich details to confirm or refute user conjectures.

Our interface design provides simple access to sophisticated entity resolution algorithms and enables users to flexibly apply sequences of actions to identify duplicates effectively. In addition, users are provided with a simple network visualization, which displays the relational context between potential duplicates and allows users to make quick resolution decisions based on the context. The subnetwork visualization is based on the information visualization principle of laying out the entities on a meaningful substrate [4], [12] and is well suited to entity resolution tasks.

In this section, we introduce an interactive entity resolution tool, D-Dupe, and illustrate how our design principles are applied to the overall interface design by demonstrating a simple entity resolution scenario (resolving duplicate authors) using a real-world bibliographic data set. The data set consists of a subset of ACM Digital Library, which contains 4,073 papers from the ACM CHI conference from 1982 to 2004 authored by 6,358 people.

Fig. 1 shows the overall D-Dupe interface, which is composed of three coordinated windows: the potential duplicate viewer, the relational context viewer, and the data detail viewer. The potential duplicate viewer (on the left) shows a list of potential duplicate author pairs that are identified and ranked based on user-defined similarity metrics. Users can select a potential duplicate author pair, and then the relational context viewer (in the upper right corner) shows the coauthor relationships between the potential duplicate author pair. Fig. 1 shows a simple semantic graph layout that represents the coauthorship relationships of the two potential duplicate authors, “George W. Fitzmaurice” and “George Fitzmaurice.” Finally, the data detail viewer (in the lower right corner) shows all the attribute values of nodes (authors) and edges (papers) displayed in the relational context viewer.

Users begin the entity resolution process by loading one or multiple data files depending on their task (deduplication or data integration). Before searching for potential duplicates, users need to define a similarity metric, which describes what information should be used to determine if two records may match. For example, if users consider authors likely to match when they have similar names and affiliations, users can select the attributes such as author’s name and affiliation for computing the author similarity.

After the similarity metric is defined as described above, users can search for the potential duplicate authors using the search button located in the potential duplicate viewer. Once the search has been completed, users can see a list of potential duplicate author pairs, sorted by their similarity, in the potential duplicate viewer. Users can scroll through the list of potential duplicate pairs and select a potential duplicate pair for further analysis.

Users can then see the coauthorship relations between the potential duplicate authors in the relational context viewer. The relational context viewer shows a subnetwork that only contains the potential duplicate authors and their coauthors rather than an entire network. The relational context viewer is divided into five regions (separated by red dotted lines in Fig. 1). The potential duplicate authors are located in the second and fourth regions, the shared coauthors in the third region, and the nonshared coauthors in the first and fifth regions. With this simple but stable and meaningful layout, users can easily recognize the coauthorship relation between the potential duplicate authors. Users can dynamically resize, sort, filter, and brush the nodes and edges in the viewer for better understanding of the relationship between the potential duplicate authors.

Users can resolve the potential duplicate authors in three ways: 1) merge the potential duplicate authors, 2) mark them as distinct authors to exclude from further search results, and 3) leave them for later or other users' decision. Once duplicate authors are merged, D-Dupe automatically searches for the next potential duplicate authors for the merged author and shows it in the relational context viewer. With this feature, users can make a series of resolution decisions for the multiple references to the same author and resolve them incrementally.

## 4 Data Model

We designed D-Dupe to work with several schemas for relational co-occurrence data. The most generic is the *affiliation network schema* [32], which describes data consisting of two types of entities, referred to as *Actors* and *Events,* and there is one relationship between the entities, a *participation* relationship, which links actors to events. In our earlier example, the actors are the authors, the events are the papers, and the participation relationship represents the author relationship between an author and a paper. These are referred to as affiliation networks in the social network research community and are often viewed as bipartite graphs, with actors as one set of nodes, events as the other set of nodes, and participation links between actors and events. Each of the entities in the network may have additional associated attributes, and we also assume they have an associated unique identifier *Id*. It is straightforward to describe the data model using the following relational schema over an actor relation $\mathcal{A}$, event relation $\mathcal{E}$, and participation relation $\mathcal{P}$:

- $A(Id_a, a_1, a_2, \ldots, a_m)$;
- $E(Id_e, e_1, e_2, \ldots, e_n)$;
- $P(Id_a, Id_e)$.

In addition, D-Dupe also supports a data schema that can be described as a standard node-link graph model, which has advantages in representing the data with directed edges or the data with only one type of entity (either actor or event).

## 5 Implementation Challenges

In this section, we explore the challenges in designing an interface for interactive entity resolution and our proposed solutions by describing the features of D-Dupe following the steps of the entity resolution process.

### 5.1 Data Integration and Deduplication

Entity resolution is useful for two related tasks, deduplication and data integration. Deduplication (also known as data cleaning) is the process of reconciling multiple references within a single data source to the same real-world entity. The main goals are avoiding data redundancy and data inconsistency, enhancing the correctness of data statistics and facilitating knowledge discovery [16], [24]. Data integration is a closely related but slightly different task. Data integration is the process of determining approximate matches or joins for consolidating data from multiple sources. The availability of data from a wide variety of sources often requires the process of finding and matching multiple references across multiple data sets.

To support both entity resolution tasks, we designed D-Dupe to import multiple data input files and let users choose a method for comparing entities in multiple data sources depending upon their specific entity resolution tasks. If the data schemas of imported data files are not exactly the same, D-Dupe unions them by default and indicates which attributes are common among the data files so that those attributes can be used for defining a similarity metric for deduplication and data integration tasks. This mechanism can be enhanced further by using various schema matching algorithms [7].

### 5.2 Similarity Metric Definition

The entity resolution process generally starts with the similarity search to find potential duplicate pairs to explore. There is a great deal of flexibility in how the similar pairs are found. The decision can be based on similar attribute values or it can be based on the relational similarity between the entities. Once users have defined a similarity criterion, the database is searched, and pairs of similar entities are presented.

Our component design for defining a similarity metric is based on the definition of the similarity [16], [17] between two entities, $e_i$ and $e_j$, which is the weighted combination of the attribute similarity of the two entities and the relational similarity:

$$sim(e_i, e_j) = (1 - \alpha) \times sim_A(e_i, e_j) + \alpha \times sim_R(e_i, e_j), \quad 0 \leq \alpha \leq 1,$$

where $sim_A(e_i, e_j)$ represents the similarity of attribute, and $sim_R(e_i, e_j)$ represents the relational similarity between the entities. $sim_A(e_i, e_j)$ is further defined as

$$sim_A(e_i, e_j) = \sum_{k=1}^{n} w_k \times sim\_fun_k(e_i \cdot a_k, e_j \cdot a_k),$$
$$-1 \leq w_k \leq 1 \quad \text{and} \quad \sum_{k=1}^{n} |w_k| = 1,$$

where $sim\_fun_k()$ is a similarity measure function for $k$th attribute, $e_i \cdot a_k$ is the $k$th attribute value of entity $e_i$, $w_k$ represents a weight of $k$th similarity measure function, and $n$ represents the number of entity attributes.

D-Dupe supports several measures of attribute similarity including common string similarity measures [8], [10], [25], [40] such as

- Levenstein (edit distance): the minimum number of insertions, deletions, and substitutions required to transform one string to the other.
- Jaccard: the size of the intersection among the characters divided by the size of the union of the characters occurring.
- Jaro and JaroWinkler: more sophisticated string similarity scores, which look at the similarity within a certain neighborhood; the JaroWinkler score is based on Jaro and weights matches at the beginning more highly.
- MongeElkan: another more sophisticated string similarity measure, which looks at matching subcomponents of the strings; it is good at finding swapped fields, such as first and last names.

We use the implementations from the open source library SimMetrics [35] for these measures.

The relational similarity $sim_R(e_i, e_j)$ uses neighborhood information (neighboring entities) of $e_i$ and $e_j$. D-Dupe supports several measures of neighborhood similarity. For example, one simple approach for measuring commonness between the entities is counting the overlap in their neighbors:

$$CommonNbr(e_i, e_j) = \frac{1}{K} \times |Nbr(e_i) \cap Nbr(e_j)|,$$

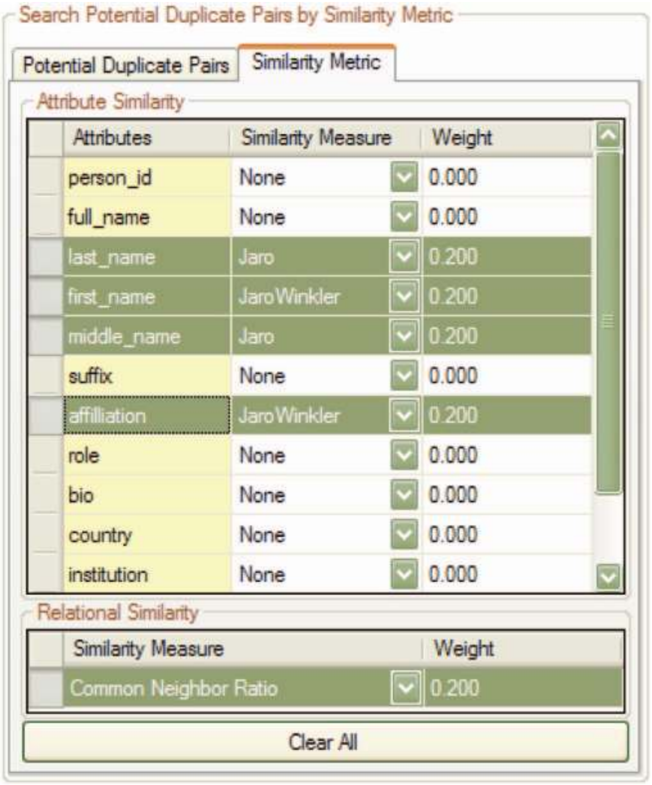
where $Nbr(e_i)$ represents the set of neighboring entities of $e_i$, and $K$ is a large enough constant such that the measure is less than 1. This can be normalized by the total number of neighbors:

$$CommonNbrRatio(e_i, e_j) = \frac{|Nbr(e_i) \cap Nbr(e_j)|}{|Nbr(e_i) \cup Nbr(e_j)|}.$$

Users can use larger neighborhoods (Higher Order Neighborhood [16]) to calculate the number of neighboring entities of $e$. For example, the second-order neighborhood can be calculated by recursively taking the set union of the neighborhoods of all neighboring entities just as

$$Nbr^2(e) = \bigcup\nolimits_{e' \in Nbr(e)} Nbr(e').$$

We designed the similarity metric tab to be part of the potential duplicate viewer so that users can define a similarity metric using both attribute similarity and relational similarity. In the attribute similarity groupbox, users can see all the entity attributes available in the imported data set in the first column. Fig. 2 shows there are 13 author attributes in the sample data set such as person_id, first_name, middle_name, last_name, affiliation, bio, and so on. The second column, the similarity measure column, allows users to choose a similarity measure function for each attribute. Each cell in this column displays different similarity function options depending on the attribute data type (e.g., string or number). For example, users can use string matching functions described earlier for computing the name similarity. Each similarity measure function has its advantages; for example, Levenstein is able to capture misspellings and JaroWinkler performs well for matching person names and abbreviations. The choice of similarity measure function depends upon the application, and one of D-Dupe's advantages is the ease and flexibility with which users can explore multiple similarity function combinations and orderings. In the last column, users can assign either positive or negative weights to the corresponding similarity functions. In Fig. 2, first_name, middle_name, last_name, and affiliation attributes are selected, and Jaro, JaroWinkler string match functions are applied to the selected attributes, respectively, for computing the similarity of author entity. In addition, the Common Neighbor Ratio relational similarity (bottom of Fig. 2) is selected in defining a similarity metric.


Search Potential Duplicate Pairs by Similarity Metric
Potential Duplicate Pairs
Similarity Metric
Attribute Similarity
Relational Similarity
Clear All

| Attributes | Similarity Measure | Weight |
|---|---|---|
| person_id | None | 0.000 |
| full_name | None | 0.000 |
| last_name | Jaro | 0.200 |
| first_name | JaroWinkler | 0.200 |
| middle_name | Jaro | 0.200 |
| suffix | None | 0.000 |
| affiliation | JaroWinkler | 0.200 |
| role | None | 0.000 |
| bio | None | 0.000 |
| country | None | 0.000 |
| institution | None | 0.000 |

| Similarity Measure | Weight |
|---|---|
| Common Neighbor Ratio | 0.200 |

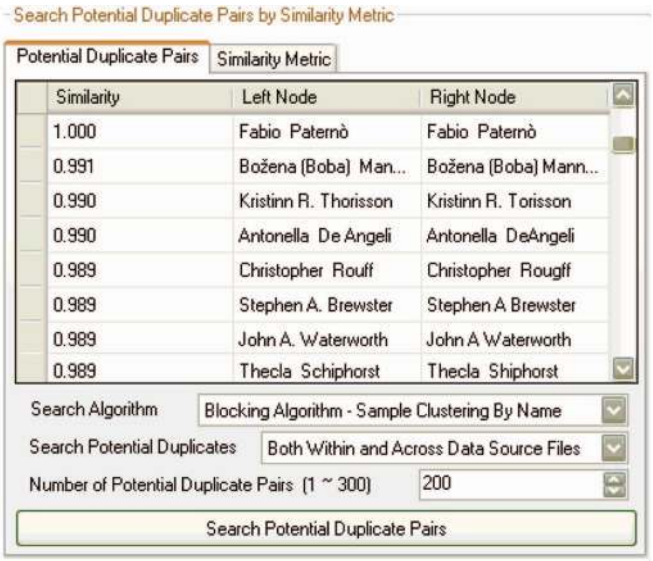
Fig. 2. The similarity metric tab in the potential duplicate viewer allows users to use both attribute and relational similarity measures for computing the similarity between the comparing entities.

The overall design of the interface can be extended for supporting user-defined domain-specific similarity functions.

### 5.3 Potential Duplicate Search

Once the similarity metric is defined, users can search for the potential duplicate entities in the imported data sets. Using the search options provided in the potential duplicate viewer, users can control the performance and completeness of search depending upon their tasks.

D-Dupe is designed to have logical data partitions in each data file for improving the search performance. By default, D-Dupe compares all the possible pairs of entities to find the potential duplicates. However, unless the data sets are very small, it is often impractical to perform exhaustive search. Apart from the scaling issue, most pairs checked by $O(n^2)$ comparisons will be rejected since usually only a few of all pairs are true matches [16]. To handle this issue, D-Dupe supports blocking techniques. Blocking algorithms [24] generate candidate matches and partition



Search Potential Duplicate Pairs by Similarity Metric

| Similarity | Left Node | Right Node |
|---|---|---|
| 1.000 | Fabio Paternò | Fabio Paternò |
| 0.991 | Božena (Boba) Man... | Božena (Boba) Mann... |
| 0.990 | Kristinn R. Thorisson | Kristinn R. Torisson |
| 0.990 | Antonella De Angeli | Antonella DeAngeli |
| 0.989 | Christopher Rouff | Christopher Rougff |
| 0.989 | Stephen A. Brewster | Stephen A Brewster |
| 0.989 | John A. Waterworth | John A Waterworth |
| 0.989 | Thecla Schiphorst | Thecla Shiphorst |

Search Algorithm: Blocking Algorithm - Sample Clustering By Name

Search Potential Duplicates: Both Within and Across Data Source Files

Number of Potential Duplicate Pairs (1 ~ 300): 200

Fig. 3. Search options located in the potential duplicate viewer to support both deduplication and data integration tasks with blocking algorithms.

the full cross product of entity comparisons into mutually exclusive blocks as a preprocessing step and compare only pairs of references within each block. This can significantly improve the search performance.

Fig. 3 shows the search options located in the potential duplicate viewer. From the search algorithm option, users can choose either exhaustive search or one of the predefined blocking algorithms. The internal design allows developers to plug in their own blocking algorithms that are appropriate for the specific tasks. In addition, users can select how references are matched in multiple data files from the "Search Potential Duplicates" option. By default, the "Both Within and Across Data Source Files" option is selected to support both deduplication and data integration tasks at once, but users can only focus on either deduplication or data integration tasks. Finally, users can also specify the number of potential duplicate entity pairs to be displayed in the potential duplicate viewer so that they can focus only on the most likely duplicates in data.

Sometimes rather than examining the most likely duplicates, users may want to search for a specific entity using keywords in its attribute values. The "Search Entity by Keyword" panel located at the bottom of the potential duplicate viewer (lower left corner of Fig. 1) supports the single entity search by keyword. In addition, users can search for all the potential duplicate entities of the selected entity by double clicking an entity in the list.

### 5.4 Relational Context Visualization

Instead of visualizing the entire network, D-Dupe shows only the subnetwork relevant for the entity resolution task with a novel semantic layout. The nodes are laid out on a stable and meaningful substrate where the potential duplicates and other related entities always appear at the same location. This simplification not only reduces the users' cognitive workload in scanning the network structure but also allows D-Dupe to scale to large networks.

In addition, the semantic layout is designed to enable users to easily find chains of potential resolutions by detecting unidentified possible duplicate entities across nonshared neighbors. For example, if the potential duplicate entities are from different data sets, it is not unusual that initially they do not have any shared neighboring entities since the neighboring entities of each potential duplicate entity are also from different data sets. Fig. 4a shows a semantic graph layout that does not have any shared coauthors between the potential duplicate authors. The relational context viewer is designed to provide a similarity threshold slider in the toolbar so that users can check if there are any potential duplicates across the nonshared neighbors. If any entity pairs across the nonshared neighbors have higher similarity than the threshold specified by the slider, they are automatically moved in between the potential duplicates and placed next to each other, as shown in Fig. 4b. With this transformed layout, users cannot only anticipate the possible matches across the nonshared neighboring entities but also understand the actual underlying relationship more clearly.

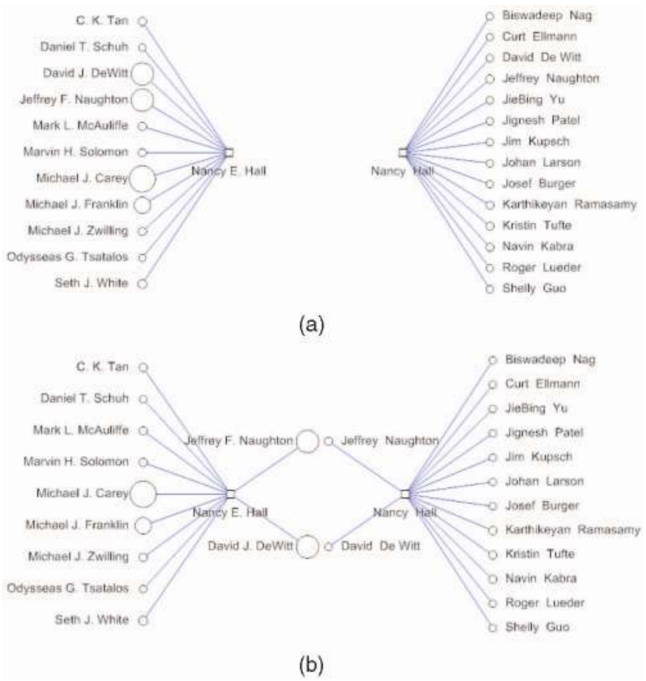


Fig. 4. Visualization of the potential duplicates across the nonshared neighbors. (a) An original semantic layout for the potential duplicate authors identified across different data sets. (b) A transformed layout in which the matching pairs identified across nonshared neighbors are moved and placed in the shared coauthors region.

In the example, the relational context viewer visualizes authors as nodes and papers as edges, but their representations can be switched due to the structural symmetry in data. D-Dupe is designed to support this graph duality so that users can focus on resolving both types of entity and explore the network structure in different perspectives.

### 5.5 Clusterwise Relational Context Visualization

The pairwise relational context visualization is further extended to the *clusterwise* relational context visualization. This visualization shows users the relationships between potential duplicate clusters rather than individual potential duplicate pairs and enables users to accomplish both deduplication and data integration tasks at once more efficiently.

Fig. 5a shows an example of clusterwise relational context visualization, which illustrates two potential duplicate author clusters found across the ACM-KDD publication data and the ACM SIGMOD publication data. The

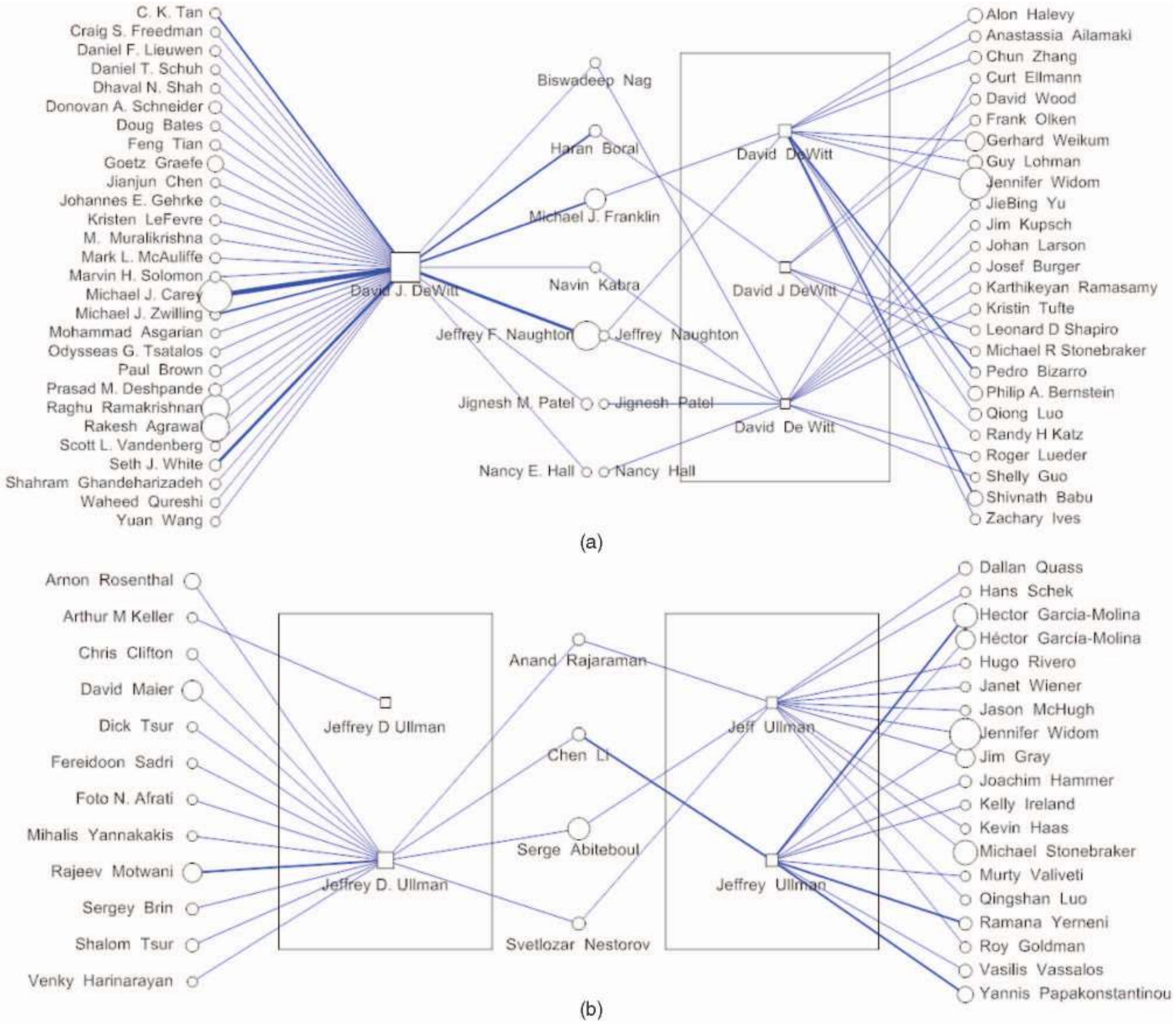
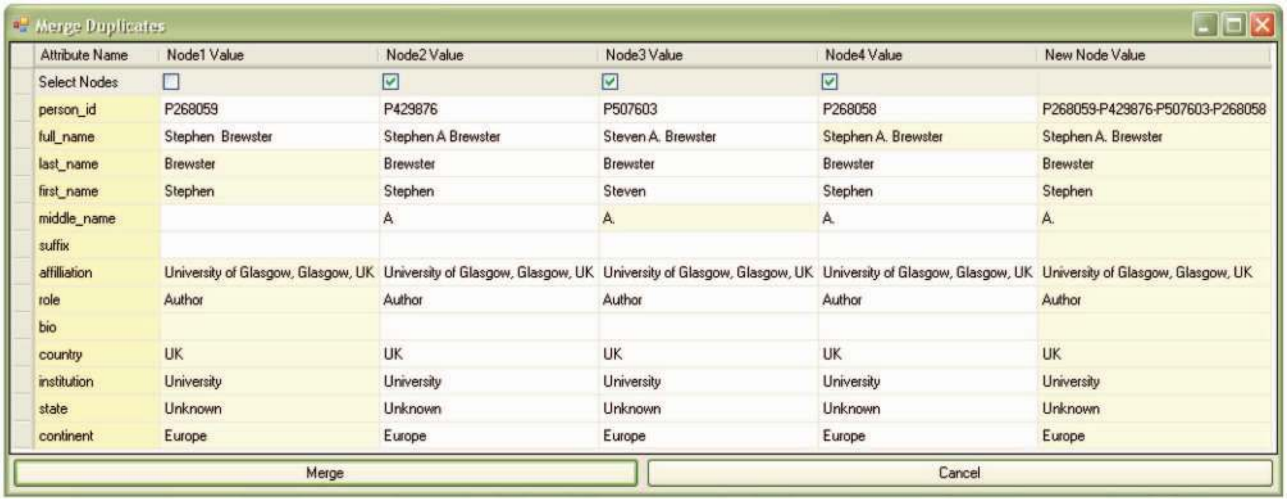


Fig. 5. A clusterwise relational context visualization not only illustrates the relationship between the clusters but also the relationship among the individual entities within the clusters. Each cluster can be collapsed into a single node, which enables users to observe either one-to-one, one-to-many, or many-to-many relationship. (a) Relational context visualization between the potential duplicate clusters found across the ACM-KDD and the ACM SIGMOD data sets. (b) Relational context visualization of two potential duplicate clusters detected within the ACM SIGMOD data set.

potential duplicate author on the left, "David J. Dewitt" is from the ACM-KDD data. On the other hand, three potential duplicate authors enclosed in a rectangle on the right {"David DeWitt," "David J DeWitt," "David De Witt"} are from the ACM SIGMOD data. Just as in the pairwise layout, the potential duplicate clusters are placed in the middle of the screen. All the shared coauthors of both clusters are placed in between them, and all the nonshared coauthors of both clusters are placed on the sides. In Fig. 5a, even though the author clusters were found across two different data sets, they have four shared coauthors, "Biswadeep Nag," "Haran Boral," "Michael J. Franklin," and "Navin Kavra," because those authors have unique identifiers in the ACM database. On the other hand, there are three potential duplicate coauthor pairs found across the nonshared coauthors, thus they are moved in between the author clusters and placed next to each other. Both the shared coauthors and the nonshared coauthors can be either sorted or filtered by their attributes such as names, node size, edge size, and so forth, so that users can understand the connections more clearly while reducing the edge crossings. With this clusterwise relational context visualization, users can easily recognize not only the coauthorship relation between the clusters but also the relationship among the individual authors in the clusters.

The potential duplicate entities in the cluster rectangle can be collapsed into a single node by double clicking it. With this mechanism, users can focus only on the relationship that they want to observe such as one-to-one (pairwise intercluster relationship), one-to-many (relationship between individual node and cluster), and many-to-many (relationship between all the nodes across the clusters). In addition, users can merge all the potential duplicate authors in the clusters with a single interaction. This single interaction accomplishes both deduplication and data integration tasks at once, which would require many more iterative steps using the pairwise entity resolution method. Users can choose only a few individual authors in the clusters for merging, if the clusters contain any hard-to-decide duplicate authors.



| Attribute Name | Node1 Value | Node2 Value | Node3 Value | Node4 Value | New Node Value |
|---|---|---|---|---|---|
| Select Nodes | ☐ | ☑ | ☑ | ☑ | |
| person_id | P268059 | P429876 | P507603 | P268058 | P268059-P429876-P507603-P268058 |
| full_name | Stephen Brewster | Stephen A Brewster | Steven A. Brewster | Stephen A. Brewster | Stephen A. Brewster |
| last_name | Brewster | Brewster | Brewster | Brewster | Brewster |
| first_name | Stephen | Stephen | Steven | Stephen | Stephen |
| middle_name | | A | A. | A. | A. |
| suffix | | | | | |
| affiliation | University of Glasgow, Glasgow, UK | University of Glasgow, Glasgow, UK | University of Glasgow, Glasgow, UK | University of Glasgow, Glasgow, UK | University of Glasgow, Glasgow, UK |
| role | Author | Author | Author | Author | Author |
| bio | | | | | |
| country | UK | UK | UK | UK | UK |
| institution | University | University | University | University | University |
| state | Unknown | Unknown | Unknown | Unknown | Unknown |
| continent | Europe | Europe | Europe | Europe | Europe |

Fig. 6. The “Merge Duplicates” dialog window shows all the potential duplicate entities along with their attribute values. Users can either pick values from the potential duplicate entities’ cells (selected cells are highlighted light yellow) or type in new values for the new entity that is created by merging the selected potential duplicate entities.

The clusterings implemented in D-Dupe can be used not only for detecting the potential duplicate clusters across the different data files but also for finding clusters within the same data file. Fig. 5b shows two potential duplicate author clusters, {“Jeffrey D Ullman,” “Jeffrey D. Ullman”} and {“Jeff Ullman,” “Jeffrey Ullman”}, found within the same ACM SIGMOD publication data.

To be more flexible in clustering, D-Dupe dynamically searches for the potential duplicates of the selected entity pair based on the currently selected similarity metric and blocking algorithm. Users can adjust the similarity metric or the search options to identify the appropriate potential duplicate clusters depending on their resolution tasks.

### 5.6 Entity Resolution

In our interface design, users can make resolution decisions by using two buttons, “Mark Distinct” and “Merge Duplicates,” in the data detail viewer. If the potential duplicate entities are not actual duplicates, users can record this using the *Mark Distinct* button. Then, the marked pairs are excluded from further search results by default but can also be shown with a different background color in the search results in case users want further investigation. On the other hand, when the Merge Duplicates button is pressed, the Merge Duplicates dialog window pops up. As shown in Fig. 6, the dialog window shows all the entity attribute names in the first column and corresponding attribute values for each potential duplicate entity in the following consecutive columns. The last column shows the attribute values of the entity to be created by merging the selected potential duplicate entities. The merge duplicates window allows users to either pick values from the potential duplicate entities’ cells or type in new values to the cells in the last column. Users can selectively merge the entities by choosing a check box in the first row, if the dialog window contains any entities the user wishes to exclude.

When the *Merge* button is pressed, D-Dupe creates a new entity node with the assigned attribute values and replaces all the potential duplicate entities in the data set with this new node. Instead of removing the entities that were merged to a new entity, D-Dupe attaches those entities to the newly created entity as children. Each entity node has both parent and children link fields. With this hierarchical structure, D-Dupe is able to easily keep track of the merging history of the potential duplicate entities and make resolution process reversible without affecting other resolution decisions in the same history thread just by splitting child entity nodes from their parent.

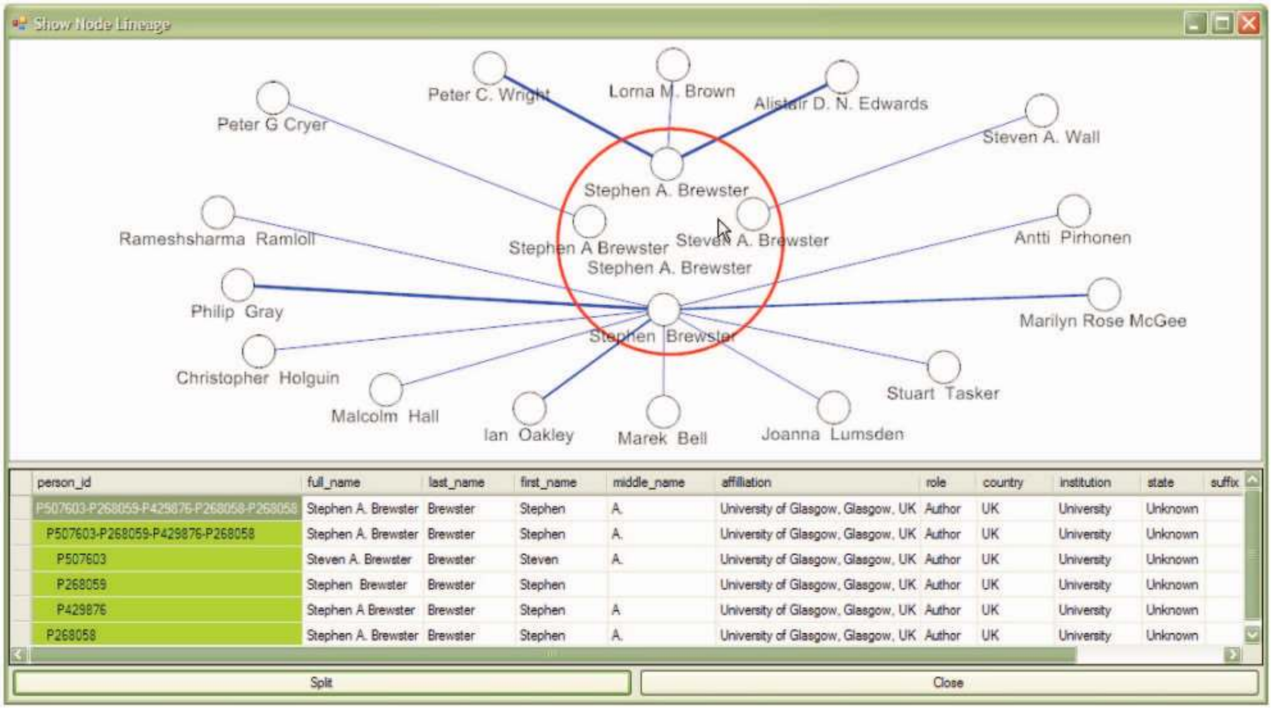
D-Dupe allows fine-level user control over resolution decisions. In addition to pairwise and clusterwise entity resolution mechanisms, we have added support for *bulk merging*, which enables users to merge multiple potential duplicate pairs with a single interaction. In this mechanism, users set a similarity threshold in advance so that all the retrieved potential duplicate pairs whose similarity values are greater than the specified threshold are automatically merged in sequence from higher similarity duplicate pair to lower similarity duplicate pair. However, like other automatic machine data cleaning approaches, it is not easy to choose a proper threshold value that can assure both high precision and high recall. Therefore, we implemented a supplementary mechanism that allows users to select a subset of potential duplicate pairs displayed in the potential duplicate viewer and apply *bulk merging* only to the selected pairs.

### 5.7 Entity Resolution History

The *Node Lineage* dialog (Fig. 7) is designed to allow users to inspect the merging history of an entity. Users can bring up the dialog either by double clicking the entity or using a popup menu in the data detail viewer.

The *tree view* (bottom of Fig. 7) shows the order as well as the hierarchy of the merged entities along with their prior attribute values. The first row represents the current entity that was created by merging two prior entities (second and sixth rows). The entity in the second row was also created by merging three prior entities (rows 3 through 5). Alternatively, the *graph view* (top of Fig. 7) shows a subnetwork with the selected entity in the center and all its neighboring entities in a concentric circle. By double clicking the center node, users can see how the prior duplicate entities (entity nodes within the red circle in Fig. 7) used to be connected to the current neighboring entities.

If users decide they want to reverse a resolution decision from the *Node Lineage* dialog, they can restore the prior entities by pressing the split button located at the bottom of the window (Fig. 7). Because of the hierarchical structure of



| person_id | full_name | last_name | first_name | middle_name | affiliation | role | country | institution | state | suffix |
|---|---|---|---|---|---|---|---|---|---|---|
| P507603-P268059-P429876-P268058-P268058 | Stephen A. Brewster | Brewster | Stephen | A. | University of Glasgow, Glasgow, UK | Author | UK | University | Unknown | |
| P507603-P268059-P429876-P268058 | Stephen A. Brewster | Brewster | Stephen | A. | University of Glasgow, Glasgow, UK | Author | UK | University | Unknown | |
| P507603 | Steven A. Brewster | Brewster | Steven | A. | University of Glasgow, Glasgow, UK | Author | UK | University | Unknown | |
| P268059 | Stephen Brewster | Brewster | Stephen | | University of Glasgow, Glasgow, UK | Author | UK | University | Unknown | |
| P429876 | Stephen A Brewster | Brewster | Stephen | A | University of Glasgow, Glasgow, UK | Author | UK | University | Unknown | |
| P268058 | Stephen A. Brewster | Brewster | Stephen | A. | University of Glasgow, Glasgow, UK | Author | UK | University | Unknown | |

Fig. 7. The *Node Lineage* dialog: the graph view (on the top) visualizes how prior duplicate entities used to be connected to the current neighboring entities of the merged entity, while the tree view (on the bottom) shows the hierarchy of previously merged duplicate entities.

entity resolution history, users can split only the incorrect decision link without affecting other prior resolution decisions in the same history thread.

Generally, users want only the final cleaned or integrated data to be saved for use by other applications. However, sometimes users may want to see all the history of intermediate entity resolution decisions and processes. For example, if multiple users collaborate on the entity resolution task, they may need to examine others' resolution decisions or correct them if necessary. Therefore, we designed D-Dupe to support both cases in exporting data. Users can export only the cleaned final data results to a new data file, or they can export the data along with all the intermediate resolution decisions as well.

## 6 System Architecture

D-Dupe was written in C# using the University of Maryland's open source toolkit, Piccolo [3], for network visualization. The D-Dupe architecture is composed of three parts: Model, View, and Controller [11]. The D-Dupe Model manages all the data-related processes as well as data mining algorithms so that D-Dupe View can request the necessary data from the model and visualize them independently from the internal data structures or algorithms that the model uses. The D-Dupe Model consists of three modules: the data mining algorithm module, the graph data structure module, and the data input/output (I/O) module. They are designed to be modular so that other developers can add their own domain-specific algorithms and similarity measures to improve accuracy and performance for entity resolution. Each module has descriptions about the input and output formats for the customizable functions so that developers can implement their own algorithm functions or similarity measure functions in the module without caring about any internal data structures as long as they use the same input and output formats for their functions. In addition, developers can add other I/O modules to support various types of databases such as XML DB and OODB. The current data mining algorithm module uses an open source library SimMetrics [35] for string distance measure functions. The D-Dupe Controller manages all the events raised by the viewers, coordinates the viewers, and controls all the data flow among the viewers. In addition, it manages the history mechanism so that users can switch back and forth to the previously searched sets of potential duplicates without researching the same set repeatedly, which often slows down the entity resolution process.

## 7 Evaluation

Our earlier work [27] demonstrates the benefits of integrating the data mining algorithms with a meaningful layout of the selected subnetwork by using several case studies on bibliographic data sets such as CiteSeer and PubMed. In addition, the scalability and the utility of the D-Dupe approach has been well demonstrated in our earlier work [14] by applying D-Dupe to real-world geospatial data sets, which contain more than 200,000 records of locations and over a million neighborhood relationships. In this study, we were able to identify more than 200 duplicate locations and deduplicate them while integrating 135 matching locations across the data sets within a half hour of use of D-Dupe, even though these data sets were carefully prepared for public release by National Geospatial-Intelligence Agency. To confirm these benefits and the payoff from still richer visual features, we conducted an empirical study to measure task completion times, success rates, and user confidence.

### 7.1 Study Design

We used a $2 \times 2 \times 4$ (two interfaces with two different bibliographic data sets with four different types of potential duplicates) repeated-measure within-subject design. To control for the effect of order and learning, we prepared two sets of potential duplicate pairs with similar difficulties for each bibliographic data set and counterbalanced the order of presentation of the interfaces and the data sets. Four dependent variables were collected in this study: task completion time, success rate, errors, and user confidence. Participants were given 12 potential duplicate pairs in each data set and each interface (a total of 48 potential duplicate pairs). Task completion time was measured by recording the time when users press a start button until they make a final resolution decision by pressing one of three decision buttons ("They are same," "They are different," or "I don't know"). Success rate is the percentage of users' correct resolution decisions (based on the ground truth data) with respect to the total number of tasks in each set. There are two types of errors users can make: marking actual duplicates as distinct entities or marking nonduplicates as the same entity. Participants were asked to indicate their confidence (1-9 range) by selecting a radio button in a popup dialog on every resolution decision they made (except the "I don't know" decision) during the study.

**Participants.** We recruited 12 participants (10 males and 2 females, ages between 25 and 39). They were all graduate students (between three and six years) in the Department of Computer Science, University of Maryland, College Park, who were comfortable with computers, familiar with scientific publications, and able to quickly understand the concept of relational databases and entity resolution (data cleaning). Among the participants, four of them were somewhat familiar with the ACM CHI publications. They either have submitted research papers to the CHI conference or have referenced papers published in the CHI conference. However, they were not familiar with the ACM SIGMOD publications. Another group of four participants was somewhat familiar with the ACM SIGMOD publications but not familiar with the ACM CHI publications. Finally, the remaining four participants were not familiar with either ACM CHI or ACM SIGMOD publications. The participants received $15 for their participation. To increase motivation, an extra $5 prize was given to the participant with the highest success rate and the fastest completion time for each interface. The study took 60-90 minutes depending on participants' performance.

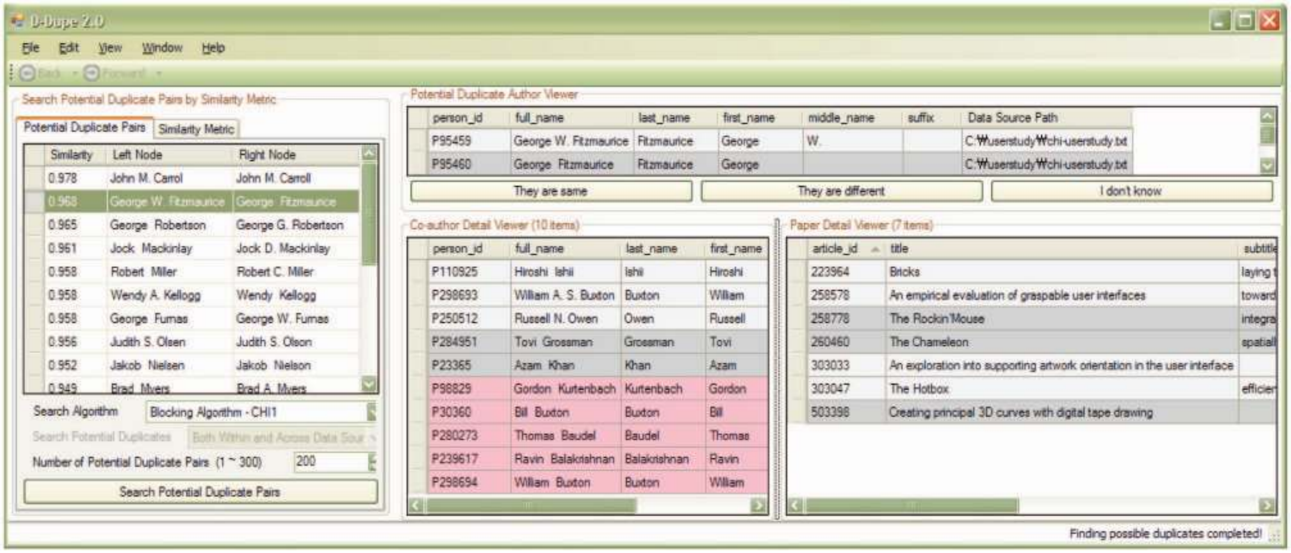
**Data sets.** Two real-world bibliographic data sets were used for this study. Both data sets were from the ACM Digital Library. The first data set contains 4,073 papers published in ACM CHI conferences over 20 years (from 1982 to 2004) and 6,358 authors of those papers. The second data set consists of 2,833 authors and their 1,637 papers published in ACM SIGMOD conferences from 1986 to 2005. From each bibliographic data set, we searched for potential duplicate authors using D-Dupe and prepared two sets of potential duplicate author pairs for each data set. The data sets were named CHI1, CHI2, SIGMOD1, and SIGMOD2, respectively. Each data set contains 12 potential duplicate author pairs, which were carefully chosen by the strategy described below.

First, potential duplicate author pairs were searched in each data set based on authors' full name. No other author's attributes such as affiliation, nationality, role, and so forth, were used. The intent was not to exclude any potential duplicate authors in the data set. An exhaustive comparison with the JaroWinkler string matching algorithm was performed to compute the similarity between all candidate pairs. From the potential duplicate author pairs whose similarity value was greater than a threshold (0.9), we kept only the potential duplicate author pairs that have more than five published papers in the conference. This was intended to pick relatively influential and well-known authors in both the CHI and SIGMOD publication venues. From the above search results, 12 potential duplicate author pairs were randomly selected for each data set, CHI1, CHI2, SIGMOD1, and SIGMOD2. Finally, we adjusted and balanced the data sets to have similar average similarity values and difficulties. In addition, we also adjusted the data sets to have all types of potential duplicate author pairs, which can be classified into four categories based on the ground truth data:

- type-1: duplicate authors with shared coauthors;
- type-2: duplicate authors without shared coauthors;
- type-3: nonduplicate authors with shared coauthors;
- type-4: nonduplicate authors without shared coauthors.

As a result, each data set contains seven type-1 pairs, two type-2 pairs, two type-3 pairs, and one type-4 pair. Their distribution rates (7 : 2 : 2 : 1) reflect the actual distribution rate of four types of potential duplicate author pairs in the searched results, making the empirical study data sets more realistic.

**Interfaces.** Two interfaces were used for the study. The first (interface-1) is a tabular style interface without any graph visualization (Fig. 8). This interface is composed of two data viewers: the potential duplicate viewer (on the left) and the data detail viewer (on the right). The data detail viewer is further divided into three subpanels: the duplicate author viewer, the coauthor viewer, and paper viewer. The duplicate author viewer shows the potential duplicate author pair selected in the potential duplicate viewer. Each row shows the author's person id, first, middle, and last names, and has its own color (white or gray). There are resolution action buttons located in the duplicate author viewer corresponding to the three potential resolution decisions: "They are same," "They are different," and "I don't know." Participants use these buttons to make a resolution decision for a given potential duplicate author pair during the study. The coauthor viewer shows a list of coauthors of the potential duplicate author pair displayed in the duplicate author viewer. As in the duplicate author viewer, each row shows the coauthor's attributes. Each coauthor row is colored white, gray, or pink to represent the relational information between the potential duplicate authors. A coauthor is colored white if he or she coauthored papers only with the white author in the duplicate author viewer. Similarly a coauthor is colored gray if he or she has a coauthorship relation only with the gray author. On the other hand, if a coauthor has written papers with both authors, the row is highlighted pink in the coauthor viewer. Finally, the paper viewer shows a list of papers authored by the potential duplicate authors. Just as in the coauthor viewer, each paper row is highlighted


D-Dupe 2.0
File Edit View Window Help
Back Forward
Search Potential Duplicate Pairs by Similarity Metric
Potential Duplicate Pairs
Similarity Metric
Search Algorithm
Blocking Algorithm - CHI1
Search Potential Duplicates
Both Within and Across Data Sour
Number of Potential Duplicate Pairs (1 ~ 300)
200
Search Potential Duplicate Pairs
Potential Duplicate Author Viewer
They are same
They are different
I don't know
Co-author Detail Viewer (10 items)
Paper Detail Viewer (7 items)
Finding possible duplicates completed!

| Similarity | Left Node | Right Node |
|---|---|---|
| 0.978 | John M. Carrol | John M. Carroll |
| 0.968 | George W. Fitzmaurice | George Fitzmaurice |
| 0.965 | George Robertson | George G. Robertson |
| 0.961 | Jock Mackinlay | Jock D. Mackinlay |
| 0.958 | Robert Miller | Robert C. Miller |
| 0.958 | Wendy A. Kellogg | Wendy Kellogg |
| 0.958 | George Furnas | George W. Furnas |
| 0.956 | Judith S. Olsen | Judith S. Olson |
| 0.952 | Jakob Nielsen | Jakob Nielson |
| 0.949 | Brad Myers | Brad A. Myers |

| person_id | full_name | last_name | first_name | middle_name | suffix | Data Source Path |
|---|---|---|---|---|---|---|
| P95459 | George W. Fitzmaurice | Fitzmaurice | George | W. | | C:₩userstudy₩chi-userstudy.txt |
| P95460 | George Fitzmaurice | Fitzmaurice | George | | | C:₩userstudy₩chi-userstudy.txt |

| person_id | full_name | last_name | first_name |
|---|---|---|---|
| P110925 | Hiroshi Ishii | Ishii | Hiroshi |
| P298693 | William A. S. Buxton | Buxton | William |
| P250512 | Russell N. Owen | Owen | Russell |
| P284951 | Tovi Grossman | Grossman | Tovi |
| P23365 | Azam Khan | Khan | Azam |
| P98829 | Gordon Kurtenbach | Kurtenbach | Gordon |
| P30360 | Bill Buxton | Buxton | Bill |
| P280273 | Thomas Baudel | Baudel | Thomas |
| P239617 | Ravin Balakrishnan | Balakrishnan | Ravin |
| P298694 | William Buxton | Buxton | William |

| article_id | title | subtitle |
|---|---|---|
| 223964 | Bricks | laying t |
| 258578 | An empirical evaluation of graspable user interfaces | toward |
| 258778 | The Rockin'Mouse | integra |
| 260460 | The Chameleon | spatiall |
| 303033 | An exploration into supporting artwork orientation in the user interface | |
| 303047 | The Hotbox | efficien |
| 503398 | Creating principal 3D curves with digital tape drawing | |

Fig. 8. A tabular style interface (interface-1): the potential duplicate viewer is located on the left and the data detail viewer is located on the right.

white, gray, or pink to represent which author wrote the paper. In addition to the coauthor and paper information provided by the data detail viewer, participants are allowed to search the web to find out more information about authors and papers for their resolution decision. Double clicking on any row in the data detail viewer initiates the automatic Google search with author's full name or paper title.

The second interface (interface-2) is the D-Dupe interface that has an additional visualization component, the relational context viewer (Fig. 1). As described earlier, the relational context viewer is tightly coupled with the data detail viewer so that the selected nodes (author) or links (papers) in the relational context viewer are highlighted in the data detail viewer, and vice versa. In order to control for the effect of the relational context visualization, participants were not allowed to change the predefined settings of the relational context viewer such as node order, node size, and visible edges.

In both interfaces, participants are asked if they are ready to proceed to a new entity resolution task. Once the start button is pressed, the interface records how long it takes until participants make a resolution decision by pressing one of three resolution decision buttons. After a decision has been made, the measured task completion time is recorded into a log file, and a dialog box pops up and asks the participants to indicate their confidence in the decision (confidence values are between 1 and 9). This process is repeated for each pair in the data set.

**Procedures.** Before beginning the study, participants were asked to complete a background survey. The survey contains questions about their experience with computers, major study area, knowledge about the databases and entity resolution, familiarity with data sets, and so forth. Participants first received 5-minute training on the first interface. The training session consisted of a brief review of entity resolution concepts, a quick demonstration of the interface, and detailed instruction on the usage of the interface.

The tutorial was administered by the same person following a basic script (explanations and demonstrations). In addition to demonstrating the features of the interfaces, the administrator explained basic strategies to complete the tasks (for example, comparing the topics of the potential duplicate authors' papers). If participants did not recall the strategies, they were reminded during the practice trials. After the 5-minute training session, participants had time to play around with it and try a set of practice tasks. Participants were allowed to ask questions during the practice tasks but not during the timed ones. When confident to continue, they were given our study tasks. Tasks did not have a time limit and participants were allowed to give up on a task at any time. Once they completed all the tasks for the first interface, the same procedure was repeated with the second interface. To control for the effect of order and learning, we counterbalanced the order of presentation of the interfaces and the data sets by choosing one of four sequences shown in Table 1 for each participant.

After the participants finished all the tasks, they were asked to complete post-study questionnaires for each interface. Quantitative factors such as preferences, satisfactions, usability, and learnability were collected along with participants' comments and suggestions during the post-study survey session.

**Apparatus.** Participants used a PC running Windows XP (3-GHz Pentium 4 with 2.5-Gbyte RAM) with a 24'' wide screen LCD monitor at 1,920 × 1,200 resolution. The interfaces were maximized to the screen as default, but participants were allowed to resize it while searching using web browsers to search the web. Both testing interfaces were modified from the original versions to collect log data such as participants' resolution decisions, user confidence, task completion time, and so on, during the study.

### 7.2 Results

Fig. 9 shows bar charts that represent the overall average task completion time, success rate, and user confidence, respectively, with the error bars representing the confidence

TABLE 1
Four Sequences of Combined Interfaces and Data Sets Designed to Control the Learning Effect

| Seq. | Order | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| A | Interface-1 (CHI1) | Interface-2 (CHI2) | Interface-1 (SIGMOD1) | Interface-2 (SIGMOD2) |
| B | Interface-1 (SIGMOD1) | Interface-2 (SIGMOD2) | Interface-1 (CHI1) | Interface-2 (CHI2) |
| C | Interface-2 (CHI1) | Interface-1 (CHI2) | Interface-2 (SIGMOD1) | Interface-1 (SIGMOD2) |
| D | Interface-2 (SIGMOD1) | Interface-1 (SIGMOD2) | Interface-2 (CHI1) | Interface-1 (CHI2) |

interval with a 0.05 confidence level. Overall, it was remarkable to see that all the participants were able to successfully complete 48 entity resolution tasks with more than 90 percent success rate within an hour of use of our tools after having just a 5-minute training.

We used paired $t$-test to examine if there are any significant differences in completion time, success rate, and user confidence for each task with respect to each interface. The results show that participants were able to accomplish the given entity resolution tasks significantly faster with the relational context viewer ($t$-value: $-3.86$, $P$-value $< 0.0001$) than without it (Fig. 9a). As for the success rate, the relational context viewer helped participants make somewhat more accurate resolution decisions, but the $t$-test shows that the difference was not statistically significant ($t$-value: 1.24, $P$-value: 0.1107) with 0.05 confidence level. However, participants were more confident about their resolution decisions when they used the relational context viewer than when they did not use it ($t$-value: 4.71, $P$-value $< 0.0001$).

We investigated the correlation between the success rate and the user confidence to examine if user's confidence affects the success rate and also to see if the relational context viewer generates false confidence. Fig. 10a shows a scatter plot in which each plot represents a task (a total of 48 resolution decisions) with overall average success rate and user confidence ($x$-axis: success rate, $y$-axis: user confidence). There is a positive correlation between success rate and user confidence. On the other hand, Fig. 10b shows a scatter plot of 48 decisions resolved only with the relational context viewer. The slope of the trend line is less steep because each decision has much larger changes in success rate than the changes in user confidence. Note that we still can see a positive correlation between the success rate and the user confidence, which suggests that the relational context viewer did not result in misguided confidence to the participants despite a few outliers.

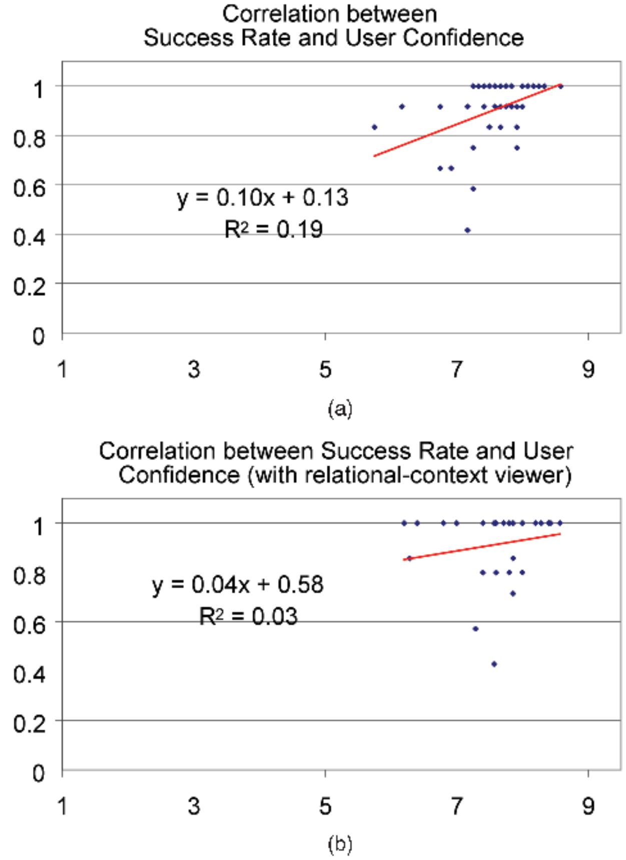


Fig. 10. Correlation between success rate and user confidence. (a) Scatter plot of 48 study tasks with their overall success rate and average user confidence. (b) Scatter plot of 48 study tasks with average success rate and user confidence when the relational context viewer was used.

**Analysis by task types.** In addition to the analysis of overall average of completion time, success rate, and user

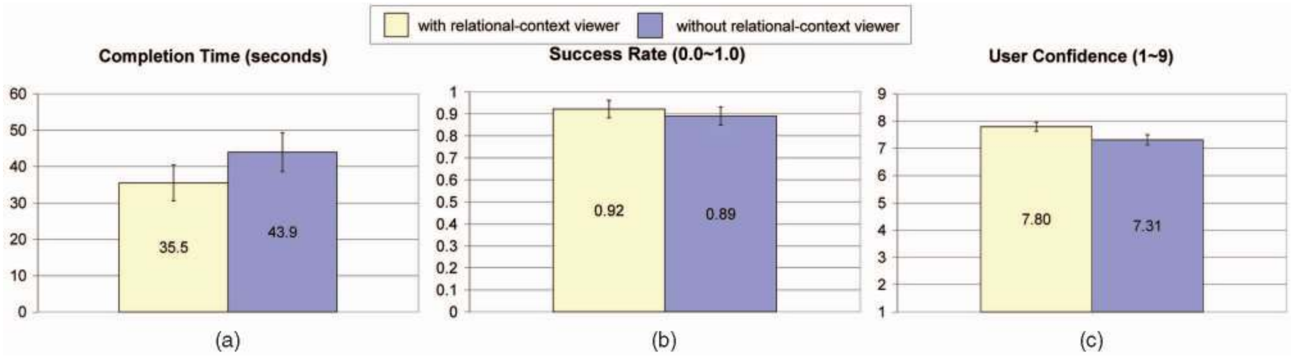


Fig. 9. Average task completion time, success rate, and user confidence for the interface with the relational context viewer and the tabular interface. Error bar indicates the confidence interval with 0.05 confidence level. (a) Overall average task completion time. (b) Overall success rate. (c) Overall user confidence.

confidence for each interface, we also analyzed the study results in terms of four types of the potential duplicate author pairs: type-1: duplicate authors with shared coauthors; type-2: duplicate authors without shared coauthors; type-3: nonduplicate authors with shared coauthors; and type-4: nonduplicate authors without shared coauthors. We were interested in analyzing the effectiveness of the relational context viewer for the different types of potential duplicate pairs.

Not surprisingly, the participants spent less time on type-1 and type-3 tasks (Fig. 11a) with higher user confidence (Fig. 11b) and better success rate (Fig. 11c), regardless which interface they used. Our hypothesis is that this is because the participants were able to make use of the relational information between the potential duplicate authors in their resolution decisions. On the other hand, they were more careful in making decisions for type-2 and type-4 tasks and spent more time searching the web since the relational information in these tasks was not that helpful.

As shown in Fig. 11a, the participants were able to complete all types of tasks faster with the relational context viewer than without it in terms of task completion time.

However, it is interesting that the relational context viewer has statistically significant improvements only for type-1 and type-2 tasks. The variance of completion time for type-4 tasks is relatively large because there are only four tasks of this type in the data sets (one in each data set). The paired $t\text{-test}$ shows that type-1 tasks have $t\text{-value} = -3.15$ and $P\text{-value} < 0.001$, and the type-2 tasks have $t\text{-value} = -2.13$ and $P\text{-value} = 0.035$, respectively.

The analysis of user confidence shows similar results (Fig. 11b). The overall average user confidence with the relational context viewer is higher than without it for all types of tasks. However, only type-1 tasks ($t\text{-value} = 5.00$, $P\text{-value} < 0.001$) and type-2 tasks ($t\text{-value} = 2.57, P\text{-value} = 0.0185$) have statistically significant differences. On the other hand, there are no significant improvements in success rate for any specific task type.

From the above analysis, it is apparent that the relational context viewer assists users in accomplishing certain types of entity resolution tasks significantly faster and makes them feel much more confident about their decisions. However, we did not find any clear evidence in this study whether it also helps users make more accurate resolution decisions. This may be due to the small number of participants in this study or the relative lack of difficulty in the tasks, and further research is warranted.

**Analysis of the effect of users' familiarity with data.** We investigated further to see if users' familiarity with the data sets had an effect on the utility of the interfaces. As mentioned earlier, we classified participants into three groups based on their familiarity with the data sets (participants familiar with CHI data set, participants familiar with SIGMOD data set, and participants familiar with neither). Figs. 12a and 12b show the average task completion time for the SIGMOD data set and the CHI data set, respectively, with respect to the three participant groups. There seems to be a relationship between the task completion time and users' familiarity with the data set. In other words, the participants accomplished the tasks in familiar domains more quickly than in unfamiliar domains. In addition, each group shows better performance in terms of task completion time, success rate, and user confidence with the relational context viewer whether or not they were familiar with the data sets. However, unfortunately, due to the wide differences (i.e., large variances) in skills among participants in each group, the paired $t\text{-tests}$ failed to show that those average value differences in task completion time, success rate, and user confidence are statistically significant. In Fig. 12, it is interesting to speculate about skill differences between groups. The SIGCHI group's rapid (but not significantly different) performance may be because of their greater familiarity with GUIs and visual user interfaces. In addition, despite our attempts to control for differences in

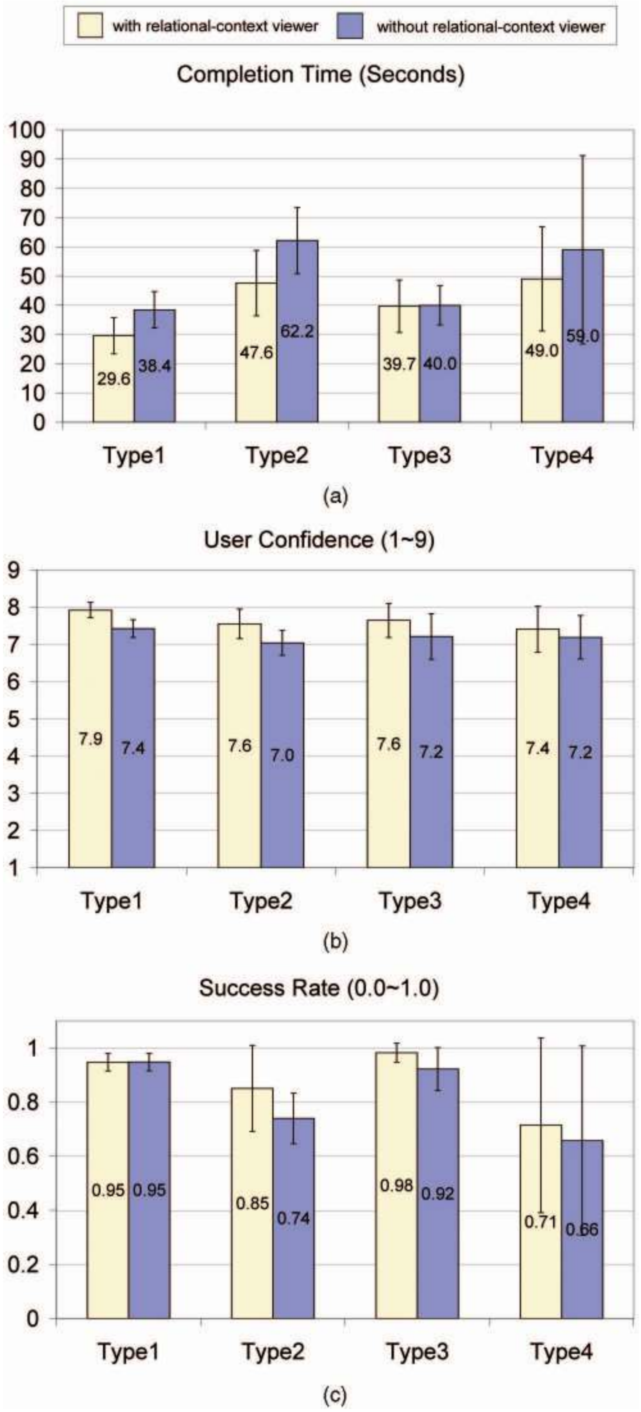


Fig. 11. Bar charts comparing the average values of three dependent variables in terms of four task types based on the existence of the relational context viewer. (a) Average task completion time (seconds) based on four task types. (b) Average user confidence (1-9) based on four task types. (c) Average success rate (0-1.0) based on four task types.

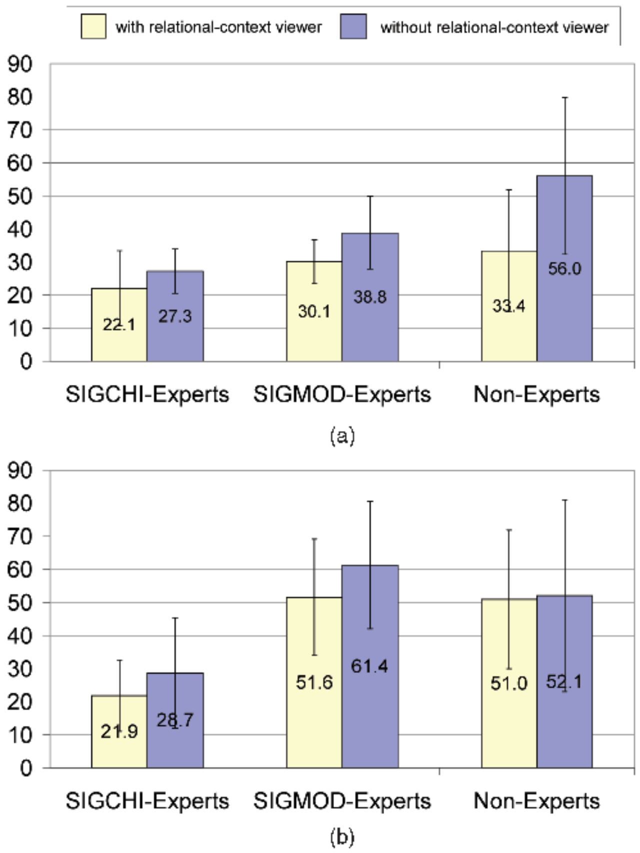


Fig. 12. Average task completion times with respect to participants' familiarity (four participants per group) with the data sets. (a) Average task completion time (seconds) for the SIGMOD data set. (b) Average task completion time (seconds) for the CHI data set.

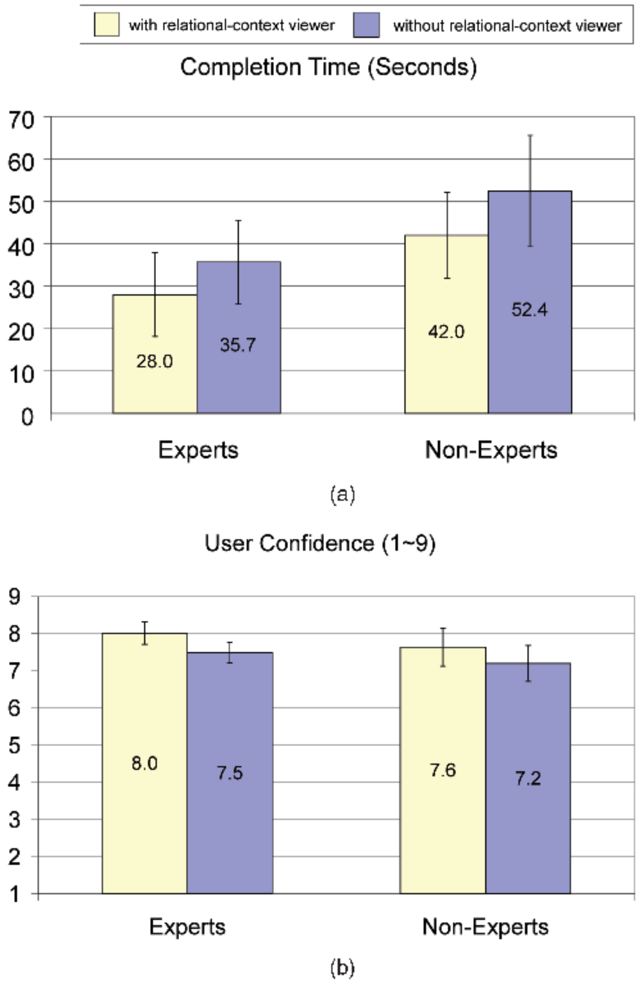


Fig. 13. Comparison of task completion time and user confidence between expert group and nonexpert group. (a) Average task completion time (seconds) of each group depending upon the existence of the relational context viewer. (b) Average user confidence (1-9) of each group depending upon the existence of the relational context viewer.

difficulty between the two data sets, the larger CHI data set may have been more difficult. Larger groups of participants will be needed to detect significant performance differences based on familiarity with the data sets.

**Analysis by users' skills with interface.** Based on the participants' background survey as well as our observations of the participants' skills in using computer software during the study, we partitioned 12 participants into two groups (i.e., expert group and nonexpert group) to examine if the relational context viewer can help both groups of users accomplish entity resolution tasks with better performance. Interestingly, it turns out that the relational context viewer facilitates the entity resolution process significantly for both expert users and nonexpert users in terms of task completion time and user confidence.

Fig. 13a shows the average task completion times of both the expert user group and the nonexpert user group, respectively. Both user groups were able to complete the given tasks much faster with the relational context viewer than without it, and their paired $t$-tests show that their differences are statistically significant ($t\text{-value} = -2.06$, $P\text{-value} = 0.03$ for expert group and $t\text{-value} = -2.03$, $P\text{-value} = 0.0337$ for nonexpert group).

As for the user confidence, Fig. 13b shows that the average user confidence between the two groups are almost the same while the participants in both groups were more confident about their resolution decisions when they used the relational context viewer. Their paired $t$-test results also show there exist significant differences in user confidence depending on the existence of the relational context viewer ($t\text{-value} = 2.31$, $P\text{-value} = 0.0208$ for expert group and $t\text{-value} = 2.77$, $P\text{-value} = 0.0091$ for nonexpert group). However, not surprisingly, given previous results, neither expert group nor nonexpert group has any statistically significant differences in success rate whether or not they used the relational context viewer.

**Post-study survey.** In the post-study survey, the participants were asked to answer four questions (ease of use, ease of learning, overall satisfaction, and overall confidence) for each interface with ratings on a nine-point Likert scale. The overall results are shown in Fig. 14. While the participants did not feel the interface with the relational context viewer was much harder to learn or use compared with the tabular interface, their overall satisfaction and confidence toward the interface with the relational context viewer was remarkably strong. There were significant differences in their indication of satisfaction and confidence between two interfaces ($t\text{-value} = -2.66$, $P\text{-value} = 0.011$ for satisfaction, $t\text{-value} = -3.56$, $P\text{-value} = 0.002$ for confidence). More importantly, it turns out that their overall confidence for each interface in the poststudy survey matches the analysis results of log data.

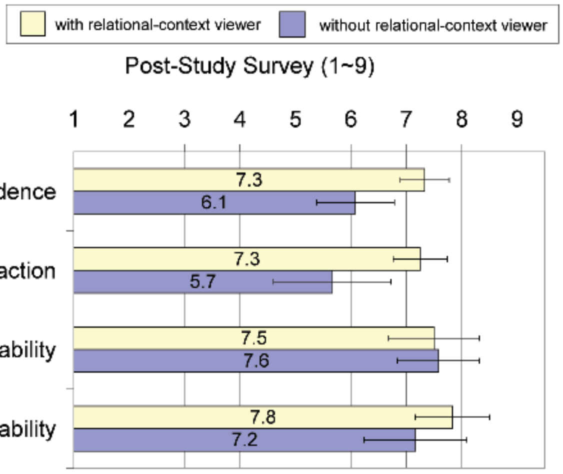


Fig. 14. The results of post-study survey for each interface.

All the participants mentioned that the relational information was very helpful in making the resolution decisions regardless of interface types, and 11 out of 12 mentioned that they preferred the relational context visualization and it helped them in resolving the author entities. Only one participant mentioned that he felt more comfortable with the tabular style interface and did not think the relational context visualization helped him much. Curiously, this participant (who was in the SIGMOD expert group and took the sequence C in Table 1) won the $5 prize for the best performance (highest success rate and fastest completion time) with the interface with the relational context viewer. Moreover, his log showed that his performance (speed and accuracy) with the relational context viewer was much better than without it, which suggests that the relational context viewer facilitates the entity resolution tasks whether or not users believe in its utility. This remarkable outcome underlines the need for performance testing in addition to subjective satisfaction measures.

Many participants mentioned that since publication date of papers (event entity) played an important role in their resolution decisions, it would be helpful if the interface supported visualization of publication date information. In fact, we already recognized the importance of time information for entity resolution and analytic tasks and included it in another visual analytic tool [13].

## 8 Conclusion

We have presented D-Dupe, a novel task-specific visual analytic tool for entity resolution in relational data. By focusing on the task at hand, we were able to develop a simple network visualization that is well suited to the task. As the empirical user study showed, this visualization helped users make decisions statistically significantly faster, while producing greater confidence in decisions. This preliminary evaluation should be followed up with larger numbers of subjects and more tasks.

Anecdotally, users who have performed entity resolution or data cleaning with other methods have been very excited about the interactive approach in D-Dupe. We have found the stable pairwise visualization using semantic substrate ideas to be useful in more general settings than just entity resolution and have recently included them in a tool for visualization of changing group membership over time, which uses a similar paradigm. Obviously, entity resolution is just one part of the analysis process, and in future work, we will more tightly integrate D-Dupe with tools that support other analytic tasks.

## Acknowledgments

The authors appreciate the participants of the user study for their efforts and valuable comments. This work was supported in part by the US National Science Foundation (NSF), NSF #0423845 and NSF #0438866, and the National Geospatial-Intelligence Agency.

## References

[1] A. Monge and C. Elkan, "The Field Matching Problem: Algorithms and Applications," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD),* 1996.

[2] *An Atlas of Cyberspace,* www.cybergeography.org/atlas, 2008.

[3] B. Bederson, J. Grosjean, and J. Meyer, "Toolkit Design for Interactive Structured Graphics," *IEEE Trans. Software Eng.,* vol. 30, no. 8, pp. 535-546, Aug. 2004.

[4] B. Shneiderman and A. Aris, "Network Visualization by Semantic Substrates," *IEEE Trans. Visualization and Computer Graphics,* vol. 12, no. 5, pp. 733-740, Sept./Oct. 2006.

[5] D. Kalashnikov, S. Mehrotra, and Z. Chen, "Exploiting Relationships for Domain-Independent Data Cleaning," *Proc. SIAM Int'l Conf. Data Mining (SIAM SDM),* 2005.

[6] E. Adar, "Guess: A Language and Interface for Graph Exploration," *Proc. Conf. Human Factors in Computing Systems (CHI '06),* pp. 791-800, 2006.

[7] E. Rahm and P. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *The VLDB J.,* vol. 10, no. 4, 2001.

[8] E.S. Ristad and P.N. Yianilos, "Learning String-Edit Distance," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 5, pp. 522-532, May 1998.

[9] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice-Hall, 1999.

[10] G. Navarro, "A Guided Tour to Approximate String Matching," *ACM Computing Surveys,* vol. 33, no. 1, pp. 31-88, 2001.

[11] G.E. Krasner and S.T. Pope, "A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80," *J. Object-Oriented Programming,* vol. 1, no. 3, pp. 26-49, 1988.

[12] H. Kang and B. Shneiderman, "Exploring Personal Media: A Spatial Interface Supporting User-Defined Semantic Regions," *J. Visual Language and Computing,* vol. 17, no. 3, pp. 254-283, 2006.

[13] H. Kang, L. Getoor, and L. Singh, "Visual Analysis of Dynamic Group Membership in Temporal Social Networks," *SIGKDD Explorations: Special Issue on Visual Analytics,* vol. 9, no. 2, pp. 13-21, 2007.

[14] H. Kang, V. Sehgal, and L. Getoor, "GeoDDupe: A Novel Interface for Interactive Entity Resolution in GeoSpatial Data," *Proc. Int'l Conf. Information Visualisation (IV '07),* pp. 489-496, 2007.

[15] I. Bhattacharya and L. Getoor, "Collective Entity Resolution in Relational Data," *ACM Trans. Knowledge Discovery from Data (TKDD '07),* vol. 1, no. 1, 2007.

[16] I. Bhattacharya and L. Getoor, "Entity Resolution in Graphs," *Mining Graph Data,* L.B. Holder and D.J. Cook, eds., Wiley, 2006.

[17] I. Bhattacharya and L. Getoor, "Iterative Record Linkage for Cleaning and Integration," *Proc. ACM SIGMOD Workshop Data Mining and Knowledge Discovery (DMKD '04),* pp. 11-18, 2004.

[18] I. Herman, G. Melançon, and M.S. Marshall, "Graph Visualization and Navigation in Information Visualization: A Survey," *IEEE Trans. Visualization and Computer Graphics,* vol. 6, no. 1, pp. 24-43, Jan.-Nar, 2000.

[19] J. Heer, S.K. Card, and J.A. Landay, "Prefuse: A Toolkit for Interactive Information Visualization," *Proc. Conf. Human Factors in Computing Systems (CHI '05),* pp. 421-430, 2005.

[20] J. O'Madadhain, D. Fisher, P. Smyth, S. White, and Y.B. Boey, "Analysis and Visualization of Network Data Using JUNG," *J. Statistical Software,* 2005.

[21] L.C. Freeman, "Visualizing Social Networks," *J. Social Structure,* vol. 1, no. 1, 2000.

[22] L. Freeman, *The Development of Social Network Analysis: A Study in the Sociology of Science.* Empirical Press, 2004.

[23] M. Baur, M. Benkert, U. Brandes, S. Cornelsen, M. Gaertler, B. Köpf, J. Lerner, and D. Wagner, "Visone Software for Visual Social Network Analysis," *Graph Drawing Software,* P. Mutzel, M. Jünger, and S. Leipert, eds., pp. 463-464, Springer, 2002.

[24] M. Bilenko, B. Kamath, and R.J. Mooney, "Adaptive Blocking: Learning to Scale Up Record Linkage," *Proc. Int'l Conf. Data Mining (ICDM '06),* pp. 87-96, 2006.

[25] M. Bilenko and R.J. Mooney, "Adaptive Duplicate Detection Using Learnable String Similarity Measures," *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '03),* pp. 39-48, 2003.

[26] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, "Adaptive Name Matching in Information Integration," *IEEE Intelligent Systems,* vol. 18, no. 5, pp. 16-23, Sept./Oct. 2003.

[27] M. Bilgic, L. Licamele, L. Getoor, and B. Shneiderman, "D-Dupe: An Interactive Tool for Entity Resolution in Social Networks," *Proc. IEEE Symp. Visual Analytics Science and Technology (VAST '06),* pp. 43-50, 2006.

[28] *Netminer II: Social Network Mining Software,* http://www.netminer.com/NetMiner/home 01.jsp, 2008.

[29] P. Singla and P. Domingos, "Multi-Relational Record Linkage," *Proc. ACM SIGKDD Workshop Multi-Relational Data Mining (MRDM),* 2004.

[30] R. Ananthakrishna, S. Chaudhuri, and V. Ganti, "Eliminating Fuzzy Duplicates in Data Warehouses," *Proc. Int'l Conf. Very Large Databases (VLDB),* 2002.

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

[32] S. Sarawagi and A. Bhamidipaty, "Interactive Deduplication Using Active Learning," *Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '02),* pp. 269-278, 2002.

[33] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications.* Cambridge Univ. Press, 1994.

[34] S. Tejada, C. Knoblock, and S. Minton, "Learning Object Identification Rules for Information Integration," *Information Systems J.,* vol. 26, no. 8, pp. 635-656, 2001.

[35] *SimMetrics: Open Source Similarity Measure Library,* http://www.dcs.shef.ac.uk/~sam/simmetrics.html, 2007.

[36] T. Dasu and T. Johnson, *Exploratory Data Mining and Data Cleaning.* John Wiley and Sons, 2003.

[37] U. Brandes, T. Raab, and D. Wagner, "Exploratory Network Visualization: Simultaneous Display of Actor Status and Connections," *J. Social Structure,* vol. 2, no. 4, 2001.

[38] V. Raman and J. Hellerstein, "Potter's Wheel: An Interactive Data Cleaning System," *Proc. Int'l Conf. Very Large Databases (VLDB '01),* pp. 381-390, 2001.

[39] *Visual Complexity,* http://www.visualcomplexity.com, 2007.

[40] W.W. Cohen, P. Ravikumar, and S.E. Fienberg, "A Comparison of String Distance Metrics for Name-Matching Tasks," *Proc. IJCAI Workshop Information Integration on the Web (IIWeb '03),* pp. 73-78, 2003.

[41] X. Dong, A. Halevy, and J. Madhavan, "Reference Reconciliation in Complex Information Spaces," *Proc. ACM SIGMOD,* 2005.

**Hyunmo Kang** received the PhD degree in computer science from the University of Maryland, College Park, in 2003. He is currently a researcher in the Institute for Advanced Computer Studies, University of Maryland. His research interests include information visualization, visual analytics, GUI design and implementation, and personal information management. He is a member of the IEEE Computer Society.

**Lise Getoor** received the PhD degree in computer science from Stanford University in 2001. She is an associate professor in the Department of Computer Science and a member of the Institute for Advanced Computer Studies, University of Maryland, College Park. Her research interests include machine learning, reasoning under uncertainty, database, and artificial intelligence. She is a member of the IEEE Computer Society.

**Ben Shneiderman** is a professor in the Department of Computer Science, founding director (1983-2000) of the Human-Computer Interaction Laboratory, and a member of the Institute for Advanced Computer Studies, University of Maryland, College Park. His research interests include information visualization, universal usability, and creativity support tools. He is a member of the IEEE Computer Society.

**Mustafa Bilgic** received the MS degree in computer science from the University of Maryland, College Park, in 2006. He is currently a PhD student in the same institution. His research interests are in the areas of machine learning, information acquisition, and decision theory. He is a student member of the IEEE Computer Society.

**Louis Licamele** received the MS degree in computer science from the University of Maryland, College Park, in 2006. He is a PhD student in computer science at the University of Maryland. His research interests are in the areas of machine learning, data mining, bioinformatics, and social network analysis. He is a student member of the IEEE Computer Society.