Query-Driven Approach to Entity Resolution
Amir Ilkhechi, Stavros Sintos

Paper is by Hotham Altwaijry, Dmitri V. Kalashnikov, Sharad Mehrotra

Used the video presentation by Kalashnikov
Take away points

- Query driven entity resolution
  - Significantly reduce the cleaning overhead by resolving records that influence the query’s answer.
- Notion of vestigiality.
  - Use vestigiality to reduce computation.
- Different levels of clustering representation
  - Exact, Distinct, Representative
- Using blocking with QDA have a large improvement in the number of resolves.
Overview:

- Motivation
- High-level introduction to the problem and solutions
- Formal Problem Definition
- Algorithms (detailed explanation of the solutions)
- Experiments
Motivation:

- More than 80% of data mining researchers spend >40% of their project time on cleaning and preparation of data.
- Analysis on bad data can lead to incorrect results:
  - Fix errors before analysis (common approach)
  - Account for them during the analysis
Introduction

- ER phases:
  - Blocking
  - Similarity Computation
  - Clustering
Query Driven Approach

- Traditional ETL Process and ER:
  - Extract data from static data resources
  - Transform (and clean)
  - Load
  - Save as Data Warehouse
  - Queries are on the warehouse

- Query-driven ER:
  - Query on the raw data
  - Extract data (based on the query)
  - Do the necessary cleaning
Why query-driven

- Big data
- Queries on online data
- Know what to clean only at the query time
- Small organizations with large data sets and limited computational resources
  - Suppose that only a small portion of the data needs to be analyzed
## Running Example

<table>
<thead>
<tr>
<th>p_id</th>
<th>p_title</th>
<th>cited</th>
<th>venue</th>
<th>authors</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1$</td>
<td>Towards efficient entity resolution</td>
<td>65</td>
<td>Very Large Data Bases</td>
<td>Alon Halevy</td>
<td>2000</td>
</tr>
<tr>
<td>$p_7$</td>
<td>Towards efficient ER</td>
<td>45</td>
<td>VLDB</td>
<td>Alon Halevy</td>
<td>2000</td>
</tr>
<tr>
<td>$p_2$</td>
<td>Entity Resolution on dynamic data</td>
<td>25</td>
<td>ACM SIGMOD</td>
<td>Alon Halevy, Jane Doe</td>
<td>2005</td>
</tr>
<tr>
<td>$p_3$</td>
<td>ER on dynamic data</td>
<td>20</td>
<td>Proc of ACM SIGMOD Conf</td>
<td>A. Y. Halevy, J. Doe</td>
<td>2005</td>
</tr>
<tr>
<td>$p_4$</td>
<td>Entity-Resolution for dynamic data</td>
<td>15</td>
<td>SIGMOD Conf</td>
<td>A. Halevy, Jane D.</td>
<td>2005</td>
</tr>
<tr>
<td>$p_5$</td>
<td>Entity-Resolution for census data</td>
<td>10</td>
<td>ICDE Conf</td>
<td>Alon Halevy</td>
<td>2002</td>
</tr>
<tr>
<td>$p_6$</td>
<td>ER on census data</td>
<td>5</td>
<td>Proc of ICDE Conf</td>
<td>Alon Y. Halevy</td>
<td>2002</td>
</tr>
</tbody>
</table>

Dirty relation (R)
<table>
<thead>
<tr>
<th>Cluster</th>
<th>Pid</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1\oplus 7</td>
</tr>
<tr>
<td>B</td>
<td>2\oplus 3\oplus 4</td>
</tr>
<tr>
<td>C</td>
<td>5\oplus 6</td>
</tr>
</tbody>
</table>

Ground Truth

Record with id = 7 not included
Returned Answer Semantics:

- Answer returned by first cleaning and then querying:
  - Let's call it $Q^*$

- Exact Semantics:
  - It matches both in terms of clusters and their representations with $Q^*$
  - E.g., \{p_1 \oplus p_7, p_2 \oplus p_3 \oplus p_4\}

- Distinct Semantics
  - It matches in terms of clusters with $Q^*$, but representations might be different
  - E.g., \{p_1 \oplus p_7, p_2 \oplus p_3\}

- Representative Semantics
  - It might include duplicates but it matches $Q^*$ in terms of clusters.
  - E.g., \{p_1, p_7, p_2 \oplus p_3\}
Example Query

- Select * From R Where cited >= 45

<table>
<thead>
<tr>
<th>p_id</th>
<th>p_title</th>
<th>cited</th>
<th>venue</th>
<th>authors</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>Towards efficient entity resolution</td>
<td>65</td>
<td>Very Large Data Bases</td>
<td>Alon Halevy</td>
<td>2000</td>
</tr>
<tr>
<td>p7</td>
<td>Towards efficient ER</td>
<td>45</td>
<td>VLDB</td>
<td>Alon Halevy</td>
<td>2000</td>
</tr>
<tr>
<td>p2</td>
<td>Entity Resolution on dynamic data</td>
<td>25</td>
<td>ACM SIGMOD</td>
<td>Alon Halevy, Jane Doe</td>
<td>2005</td>
</tr>
<tr>
<td>p3</td>
<td>ER on dynamic data</td>
<td>20</td>
<td>Proc of ACM SIGMOD Conf</td>
<td>A. Y. Halevy, J. Doe</td>
<td>2005</td>
</tr>
<tr>
<td>p4</td>
<td>Entity-Resolution for dynamic data</td>
<td>15</td>
<td>SIGMOD Conf</td>
<td>A. Halevy, Jane D.</td>
<td>2005</td>
</tr>
<tr>
<td>p5</td>
<td>Entity-Resolution for census data</td>
<td>10</td>
<td>ICDE Conf</td>
<td>Alon Halevy</td>
<td>2002</td>
</tr>
<tr>
<td>p6</td>
<td>ER on census data</td>
<td>5</td>
<td>Proc of ICDE Conf</td>
<td>Alon Y. Halevy</td>
<td>2002</td>
</tr>
</tbody>
</table>

Dirty relation(R)
Standard solution:

- Step 1: De-duplicate the dirty relation thoroughly
  - Perhaps by calling the resolve function on all pairs of records

<table>
<thead>
<tr>
<th>cluster</th>
<th>p_id</th>
<th>p_title</th>
<th>cited</th>
<th>venue</th>
<th>authors</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>C_1</td>
<td>p_1 ⊕ p_7</td>
<td>Towards efficient entity resolution</td>
<td>110</td>
<td>Very Large Data Bases</td>
<td>Alon Halevy</td>
<td>2000</td>
</tr>
<tr>
<td>C_2</td>
<td>p_2 ⊕ p_3 ⊕ p_4</td>
<td>Entity-Resolution on dynamic data</td>
<td>60</td>
<td>Proc of ACM SIGMOD Conf</td>
<td>Alon Halevy, Jane Doe</td>
<td>2005</td>
</tr>
<tr>
<td>C_3</td>
<td>p_5 ⊕ p_6</td>
<td>Entity-Resolution for census data</td>
<td>15</td>
<td>Proc of ICDE Conf</td>
<td>Alon Halevy</td>
<td>2002</td>
</tr>
</tbody>
</table>

- Step 2: Compute the query result over the acquired clean relation

Issues:
- Large number of calls to resolve function
- Resolve itself is generally expensive
Resolve Function

- Resolve is a pairwise function $R(r_i, r_j)$
Merge Function

- It consolidates two duplicate records to produce a new record.
- Combine (@) function is used to combine each attribute

<table>
<thead>
<tr>
<th>25</th>
<th>Entity Resolution on dynamic data</th>
<th>20</th>
<th>ACM SIGMOD Proc of ACM SIGMOD Conf</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>ER on dynamic data</td>
<td></td>
<td>Alon Halevy, Jane Doe</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A. Y. Halevy, J. Doe</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2005</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>45</th>
<th>ADD semantics: ( v_i \oplus v_j = v_i + v_j )</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>MAX semantics: ( v_i \oplus v_j = \max(v_i, v_j) )</td>
</tr>
<tr>
<td>20</td>
<td>MIN semantics: ( v_i \oplus v_j = \min(v_i, v_j) )</td>
</tr>
</tbody>
</table>

\{\text{Alon Halevy, A. Y.Halevy}\} \quad \text{UNION semantics: } v_i \oplus v_j = v_i \cup v_j

\{\text{Alon Halevy}\} \quad \text{EXEMPLAR semantics: } v_i \oplus v_j = \text{choose } v_i \text{ or } v_j
Exploiting query semantics

- Query: Select * From R Where cited >= 30
  - Assume ADD semantics
  - We can prune the resolves

answer 1 = \{1\}
answer 2 = \{1, 2 \oplus 3\}

Call R(p_2, p_3) = MustMerge
Call R(p_4, p_5) = MustSeparate
Gains compared to other approaches:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># of resolve calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query-driven algorithm</td>
<td>2</td>
</tr>
<tr>
<td>Standard Transitivity Closure Algorithm</td>
<td>11</td>
</tr>
<tr>
<td>Correlation Clustering Algorithm</td>
<td>15</td>
</tr>
</tbody>
</table>
Formal Problem Definition

- Notations:
  - The following represents a relation in the database \( R = \{ r_1, r_2, \ldots, r_{|R|} \} \)
  - The set of attributes are given as \( \langle a_1, a_2, \ldots, a_n \rangle \)
  - The \( k \)’th record is represented as \( r_k = \langle \nu_{k1}, \nu_{k2}, \ldots, \nu_{kn} \rangle \)

- The goal of traditional ER:
  - Partition records in \( R \) into a set of non-overlapping clusters
  - 2 records from the same cluster correspond to the same entity
  - 2 records from distinct clusters correspond to different entities

- Queries:
  - \[
  \text{SELECT [DISTINCT] * FROM } R \text{ WHERE } a_\ell \text{ op } t
  \]
  - \( op \) is \( \{<, \leq, >, \geq, \text{ or } =\} \) if \( a_\ell \) is a numeric attribute;
  - \( = \) if \( a_\ell \) is a categorical attribute.
Problem: Optimization

- Given: Query Q
- Minimize: # of calls to resolve function
- Subject to:
  - Query satisfaction:
    - Each cluster returned by QDA must satisfy Q.
  - User-defined equivalence:
    - The answer generated by QDA must be (exactly, distinctly, or representatively) equivalent to that of TC.
Vestigiality
Vestigiality

- Categorize triples \((p,\oplus, a_i)\)
- Deal with multi-predicate selection queries
- Construction of the labeled graph
- Use relevant clique, minimal clique to test for vestigiality.
Triple \( (p, \oplus, a_i) \) Categorization

- \( p \): Query predicate
- \( \oplus \): Combine function
- \( a_i \): Attribute

- Help to significantly reduce the cleaning overhead by resolving only those edges that may influence the answer of \( Q \).
Definition 5. Triple \((p, \oplus, a_\ell)\) is in-preserving, if for all possible values \(\nu_{i\ell}, \nu_{j\ell} \in a_\ell\), if \(p\) is true for \(\nu_{i\ell}\), then \(p\) is also true for all \(\nu_{i\ell} \oplus \nu_{j\ell}\).

- (cited>=45, ADD, cited): in-preserving
- (cited<=45, ADD, cited): not in-preserving
Triple \((p, \oplus, a_\ell)\) Categorization

**Definition 6.** Triple \((p, \oplus, a_\ell)\) is out-preserving, if for all possible values \(\nu_{i\ell}, \nu_{j\ell} \in a_\ell\), if \(p\) is false for \(\nu_{i\ell}\), then it is also false for all \(\nu_{i\ell} \oplus \nu_{j\ell}\).

- (cited\(\leq 45\), ADD, cited): out-preserving
Multi-Predicate Selection Queries

- More complex selection queries (AND, OR, NOT).

<table>
<thead>
<tr>
<th>$\tau_i = (p_i, \oplus_i, a_i)$</th>
<th>$\tau_j = (p_j, \oplus_j, a_j)$</th>
<th>$\tau_i \land \tau_j$</th>
<th>$\tau_i \lor \tau_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>in-preserving</td>
<td>in-preserving</td>
<td>in-preserving</td>
<td>in-preserving</td>
</tr>
<tr>
<td>in-preserving</td>
<td>out-preserving</td>
<td>neither</td>
<td>neither</td>
</tr>
<tr>
<td>out-preserving</td>
<td>out-preserving</td>
<td>out-preserving</td>
<td>out-preserving</td>
</tr>
<tr>
<td>in-preserving</td>
<td>neither</td>
<td>neither</td>
<td>neither</td>
</tr>
<tr>
<td>out-preserving</td>
<td>neither</td>
<td>neither</td>
<td>neither</td>
</tr>
<tr>
<td>neither</td>
<td>neither</td>
<td>neither</td>
<td>neither</td>
</tr>
</tbody>
</table>
Multi-Predicate Selection Queries

**Query 2.** SELECT * FROM R WHERE cited ≥ 45 AND cited ≤ 65

- $p_1$: cited ≥ 45
- $p_2$: cited ≤ 65
- $\tau_1 = (\text{cited} \geq 45, \text{ADD}, \text{cited})$: in-preserving
- $\tau_2 = (\text{cited} \leq 65, \text{ADD}, \text{cited})$: out-preserving
- $\tau_1 \cap \tau_2$: neither
- Why?
  - $r_i \rightarrow \text{cited} = 10$
  - $r_j \rightarrow \text{cited} = 40$
  - $r_k \rightarrow \text{cited} = 20$
  - $r_i \oplus r_j \rightarrow \text{cited} = 50$ (IN the answer)
  - $r_i \oplus r_j \oplus r_k \rightarrow \text{cited} = 70$ (OUT the answer)
Creating and Labeling the Graph

- Build and label the Graph.
- Avoid creating as many nodes and edges as possible.
- Blocking to reduce the edges in the graph.
- Remove from consideration nodes and edges that will not influence further processing of \( Q \).

```latex
\begin{algorithm}
\caption{Create-Graph(\( R, Q, A_{cur}, V_{out}, V_{maybe}, \oplus \))}
\begin{algorithmic}[1]
\FOR{each \( r_k \in R \)}
\STATE \( v_k \leftarrow Create-Node(r_k) \)
\IF{Is-In-Preserving(\( p, \oplus, a, p \)) and Satisfy-Qry(\( v_k, Q \))}
\STATE \( A_{cur} \leftarrow A_{cur} \cup \{v_k\} \)
\ELSE\IF{Is-Out-Preserving(\( p, \oplus, a, p \)) and not Satisfy-Qry(\( v_k, Q \))}
\STATE \( V_{out} \leftarrow V_{out} \cup \{v_k\} \)
\ELSE \( V_{maybe} \leftarrow V_{maybe} \cup \{v_k\} \)
\ENDIF
\ENDIF
\STATE \( V \leftarrow \{A_{cur}, V_{out}, V_{maybe}\} \)
\STATE \( E \leftarrow Create-Edges-with-Blocking(V_{maybe}, V_{maybe}) \)
\STATE \( E \leftarrow E \cup Create-Edges-with-Blocking(V_{out}, V_{maybe}) \)
\STATE return \( G(V, E) \)
\end{algorithmic}
\end{algorithm}
```
Create and label the nodes

1. for each \( r_k \in R \) do
2. \( v_k \leftarrow \text{CREATE-NODE}(r_k) \)
3. if \( \text{IS-IN-PRESERVING}(p, \oplus, a_{\ell}) \) and \( \text{SATISFY-QRY}(v_k, Q) \) then
4. \( A_{\text{cur}} \leftarrow A_{\text{cur}} \cup \{v_k\} \)
5. else if \( \text{IS-OUT-PRESERVING}(p, \oplus, a_{\ell}) \)
   and not \( \text{SATISFY-QRY}(v_k, Q) \) then
6. \( V_{\text{out}} \leftarrow V_{\text{out}} \cup \{v_k\} \)
7. else \( V_{\text{maybe}} \leftarrow V_{\text{maybe}} \cup \{v_k\} \)

1. \( \ell[v_k] = \text{in} \) when triple \( (p, \oplus, a_{\ell}) \) is in-preserving and \( v_k \) satisfies \( Q \). Node \( v_k \) is added to \( A_{\text{cur}} \) as it is guaranteed to be in the final answer.
2. \( \ell[v_k] = \text{out} \) when triple \( (p, \oplus, a_{\ell}) \) is out-preserving and \( v_k \) does not satisfy \( Q \). Node \( v_k \) is added to \( V_{\text{out}} \).
3. \( \ell[v_k] = \text{maybe} \), otherwise. Node \( v_k \) is added to \( V_{\text{maybe}} \).
Create and label the edges

9 \quad E \leftarrow \text{CREATE-EDGES-WITH-BLOCKING}(V_{maybe}, V_{maybe})
10 \quad E \leftarrow E \cup \text{CREATE-EDGES-WITH-BLOCKING}(V_{out}, V_{maybe})

- Add the edge \( e_{ij} = (v_i, v_j) \)
  - Nodes \( v_i, v_j \) belong in the same block.
  - \( v_i, v_j \in V_{maybe} \)
  - \( (v_i \in V_{out} \text{ AND } v_j \in V_{maybe}) \text{ OR } (v_j \in V_{out} \text{ AND } v_i \in V_{maybe}) \)
Create and label the edges

1. \( \ell[e_{ij}] = \text{yes} \), when \( \mathcal{R}(r_i, r_j) \) has already been called and returned \text{MustMerge},
2. \( \ell[e_{ij}] = \text{no} \), when \( \mathcal{R}(r_i, r_j) \) has already been called and returned \text{MustSeparate},
3. \( \ell[e_{ij}] = \text{maybe} \), when \( \mathcal{R}(r_i, r_j) \) has already been called and returned \text{Uncertain},
4. \( \ell[e_{ij}] = \text{vestigial} \), when, Definition 8 holds. Note that as QDA proceeds forward, some edges that were not vestigial previously may become vestigial. But once they become vestigial, they remain so.
5. \( \ell[e_{ij}] = \text{unresolved} \), otherwise.

- For efficiency:
  - Label no edge \( \rightarrow \) remove the edge from the graph.
  - Label yes edge \( (v_i, v_j) \) \( \rightarrow \) Merge nodes \( v_i, v_j \).
Create and label the edges

1. \( \ell[e_{ij}] = \text{yes} \), when \( R(r_i, r_j) \) has already been called and returned \text{MustMerge},
2. \( \ell[e_{ij}] = \text{no} \), when \( R(r_i, r_j) \) has already been called and returned \text{MustSeparate},
3. \( \ell[e_{ij}] = \text{maybe} \), when \( R(r_i, r_j) \) has already been called and returned \text{Uncertain},
4. \( \ell[e_{ij}] = \text{vestigial} \), when, Definition 8 holds. Note that as QDA proceeds forward, some edges that were not vestigial previously may become vestigial. But once they become vestigial, they remain so.
5. \( \ell[e_{ij}] = \text{unresolved} \), otherwise.

- For efficiency:
  - Label no edge -> remove the edge from the graph.
  - Label yes edge \((v_i, v_j)\) -> Merge nodes \(v_i, v_j\).

\( A_{\text{cur}} \) is the answer result assuming all vestigial and unresolved edges are NO edges.
Vestigial Edges

**Definition 8.** Let $\mathcal{A}$ be the original entity resolution algorithm. An edge $e_{ij} \in E$ is vestigial when, regardless of what the ground truth for $e_{ij}$ might be, QDA can guarantee that by treating $e_{ij}$ as a no edge, it can still compute an equivalent answer to that of $\mathcal{A}$.

- Identify Vestigial Edges?
Vestigiality Testing using Cliques.

- **Clique:**
  - Given $G(V,E)$ a set $S \subseteq V$ is a clique if for every $v_i, v_j \in S, (v_i, v_j) \in E$.

**Lemma 1.** Nodes (records) co-refer only if they form a clique consisting of only *yes* edges in the ground truth.

- If a group of nodes is not a clique, that group corresponds to at least two distinct entities.
Definition 9. A clique $S$ is called relevant to $Q$, if we can assign labels to its edges such that this labeling might change $C_{cur}$, by either adding (at least one) new cluster to $C_{cur}$, or removing (at least one) cluster from $C_{cur}$.

Theorem 1. Given the current labeled graph $G$, a selection query $Q$ with predicate $p$ on attribute $a_\ell$, if no relevant clique exists that includes $e_{ij}$, then $e_{ij}$ is vestigial. However, the reverse does not hold: a vestigial edge could be part of a relevant clique. Proof is covered in [1].
Resolve $e_{23}$
$p = (cited \geq 45)$
$A_{\text{cur}} = \{p_1, p_2 \oplus p_3\}$
All edges incident to $p_1, p_2 \oplus p_3$ are vestigial.
Nodes 4, 5, 6 form a clique $S$.
Sum up of the cited attribute of $S \rightarrow 5 + 10 + 15 = 30 > 45$.
Merge nodes in $S$ cannot change $A_{\text{cur}}$
Edges $e_{45}, e_{46}, e_{56}$ are not part of any relevant clique, so vestigial edges!
Definition 10. A relevant clique $S$ is called a *minimal clique*, if no subset of nodes in $S$ can form a relevant clique.

Theorem 2. Given a graph $G$ and an in-preserving $(p, \oplus, a_\ell)$, an unresolved edge $e_{ij}$ is vestigial if and only if no minimal clique exists that includes $e_{ij}$. Proof is covered in [1].
Minimal Clique

- Resolve $e_{45}$
- Triple ($cited \geq 45, ADD, cited$) is in-preserving
- $S = \{p_2, p_3, p_4\}$ is a relevant clique ($25 + 20 + 15 = 60 \geq 45$)
- $S_m = \{p_2, p_3\} \subset S$ forms a minimal clique ($25 + 20 = 45 \geq 45$)
- Edges $e_{24}, e_{34}$ are vestigial since both $e_{24}, e_{34}$ do not belong to any minimal clique.
IS_VESTIGIAL()

\[
\text{Is-Vestigial}(e_{ij}, G, Q, \oplus) \\
1 \quad \text{if Is-In-Preserving}(p, \oplus, a_{ij}) \text{ then} \\
2 \quad \text{return not Is-In-A-Minimal-Clique}(e_{ij}, G, Q) \\
3 \quad \text{else return not Is-In-A-Relevant-Clique}(e_{ij}, G, Q)
\]

- NP-hard to test for Vestigiality.
  - Straightforward Reduction from the $k$-Clique problem.
- Worse than $O(n^2)$ calls of the resolve function.
- Challenge: Design QDA that performs vestigiality testing efficiently.
Query-Driven Solution
Overview

- The answer of QDA equivalent to first applying TC on the whole dataset and then querying the cleaned dataset.

- TC Method
  - Choose a pair of nodes to resolve
  - Apply the resolve function
  - Merge nodes if resolve function returns positive answer

- QDA similar to TC
  - Uses its own pair-picking strategy.
  - Instead of calling the resolve function, is it a vestigial edge?
QDA Approach

- Create and label the graph
- Choose an edge to resolve
  - Not the main focus of the paper
  - Quickly add some cluster-representatives or break many relevant cliques.
  - Pick edges according to a weight $w_{ij} = v_i \oplus v_j$
- Lazy edge removal
  - Implemented many optimizations.
  - Check if an edge exists.
  - After merging two nodes only common edges remain
  - $O(|R|)$ in the worst case!
  - Do not remove edges at the time of merge – Check if a node has been merged with another node. $O(1)$ time!
- Vestigiality testing (next)
- Stopping condition
  - If there exists an edge that is neither resolved nor vestigial.
  - Compute the answer (later)
Vestigiality Testing

- Given an edge $e_{ij}$ decide if it is vestigial.
- Recall, NP-hard
- Use an inexact but fast check if $e_{ij}$ is potentially part of any relevant clique.

Vestigiality-Testing($e_{ij}, G, Q, \oplus$)
1. if Is-In-Preserving($p, \oplus, a_\ell$)
   and Might-Change-Answer($\emptyset, v_i \oplus v_j, Q$) then
2. $res \leftarrow \mathcal{R}(v_i, v_j)$
3. if $res = \text{MustMerge}$ then
4.   $A_{cur} \leftarrow A_{cur} \cup \{v_i \oplus v_j\}$
5.   $V_{maybe} \leftarrow V_{maybe} - \{v_i, v_j\}$
6. else if $res = \text{MustSeparate}$ then
7.   $E \leftarrow E - \{e_{ij}\}$
8. else $\ell[e_{ij}] = \text{maybe}$
9. else if Check-Potential-Clique($e_{ij}, G, Q$) then
10. $res \leftarrow \mathcal{R}(v_i, v_j)$
11. if $res = \text{MustMerge}$ then
12.   $v_i \leftarrow v_i \oplus v_j$
13.   $\mathcal{N}[v_i] = \mathcal{N}[v_i] \cap \mathcal{N}[v_j]$
14.   $V_{maybe} \leftarrow V_{maybe} - \{v_j\}$
15. else if $res = \text{MustSeparate}$ then
16.   $E \leftarrow E - \{e_{ij}\}$
17. else $\ell[e_{ij}] = \text{maybe}$
18. else $E \leftarrow E - \{e_{ij}\}$  // this edge is vestigial
Edge Miniclique Check Optimization

- An Optimization
- If $(p,\oplus, a_{i})$ is in-preserving and $e_{ij}$ can change the current answer to $Q$
  - $e_{ij}$ is not vestigial and the algorithm calls the resolve function.

```plaintext
1      if Is-In-Preserving($p, \oplus, a_{i}$) 
     and Might-Change-Answer($\emptyset, v_{i} \oplus v_{j}, Q$) then
2        res ← $R(v_{i}, v_{j})$
3      if res = MustMerge then
4        $A_{cur} ← A_{cur} \cup \{v_{i} \oplus v_{j}\}$
5        $V_{maybe} ← V_{maybe} - \{v_{i}, v_{j}\}$
6      else if res = MustSeparate then
7        $E ← E - \{e_{ij}\}$
8      else $\ell[e_{ij}] = maybe$
```
Check for Potential Clique

- CHECK-POTENTIAL-CLIQUE(): Test if \( e_{ij} \) can potentially be part of any relevant clique.
- If yes, call the resolve function, otherwise it marks as vestigial.

```
9   else if CHECK-POTENTIAL-CLIQUE(e_{ij}, G, Q) then
10     res \leftarrow \mathcal{R}(v_i, v_j)
11     if res = MustMerge then
12       v_i \leftarrow v_i \oplus v_j
13       \mathcal{N}[v_i] = \mathcal{N}[v_i] \cap \mathcal{N}[v_j]
14       V_{maybe} \leftarrow V_{maybe} - \{v_j\}
15     else if res = MustSeparate then
16       E \leftarrow E - \{e_{ij}\}
17       else \ell[e_{ij}] = maybe
18     else E \leftarrow E - \{e_{ij}\} // this edge is vestigial
```
CHECK-POTENTIAL-CLIQUE()

- Quickly check if $e_{ij}$ is involved in any relevant/minimal clique.
- Safe approximation function:
  - False only when $e_{ij}$ is not part of any relevant/minimal clique
  - True when $e_{ij}$ might be part of some relevant clique.
CHECK-POTENTIAL-CLIQUE()

- Merge nodes $v_i, v_j$ and check if their merge change Q’s answer.
- If not, find all common neighbors of $v_i, v_j$ and tries to find the smallest potential clique which might change Q’s answer.
- Keep expanding the size of such clique until no common neighbors left.

```
CHECK-POTENTIAL-CLIQUE(e_{ij}, G, Q)
1  v_{new} ← v_i \oplus v_j
2  if MIGHT-CHANGE-ANSWER(\emptyset, v_{new}, Q) then
3    return true
4  V_{intersect} ← N[v_i] \cap N[v_j]
5  for each $v_k \in V_{intersect}$ do
6    v_{old} ← v_{new}
7    v_{new} ← v_{old} \oplus v_k
8    if MIGHT-CHANGE-ANSWER(v_{old}, v_{new}, Q) then
9      return true
10   return false
```
CHECK-POTENTIAL-CLIQUE()
Computing Answer of Given Semantics

- Compute the final answer $A_{\text{cur}}$ to query Q based on the answer semantics the user requested.

```
COMPUTE-ANSWER($A_{\text{cur}}$, $V_{\text{maybe}}$, Q, S)
1  for each $v_i \in V_{\text{maybe}}$ do
2    if SATISFY-QRY($v_i$, Q) then
3      $A_{\text{cur}} \leftarrow A_{\text{cur}} \cup v_i$
4    if $S = \text{Distinct}$ then
5      for each $v_i \in A_{\text{cur}}$ do
6        for each $v_j \neq v_i \in A_{\text{cur}}$ do
7          if $\mathcal{R}(v_i, v_j) = \text{MustMerge}$ then
8            $v_i \leftarrow v_i \oplus v_j$
9            $A_{\text{cur}} \leftarrow A_{\text{cur}} - \{v_j\}$
10       else if $S = \text{Exact}$ then
11         for each $v_i \in A_{\text{cur}}$ do
12           for each $v_j \neq v_i \in A_{\text{cur}} \cup V_{\text{maybe}}$ do
13             if $\mathcal{R}(v_i, v_j) = \text{MustMerge}$ then
14               $v_i \leftarrow v_i \oplus v_j$
15             if $v_j \in A_{\text{cur}}$ then
16               $A_{\text{cur}} \leftarrow A_{\text{cur}} - \{v_j\}$
```
Representative answer semantics

- Add nodes from $V_{maybe}$ which satisfy $Q$ to $A_{cur}$.
- At this point, $A_{cur}$ satisfies representative answer semantics.

```plaintext
1  for each $v_i \in V_{maybe}$ do
2    if Satisfy-Qry($v_i$, $Q$) then
3      $A_{cur} \leftarrow A_{cur} \cup v_i$
```
Distinct answer semantics

- Clean the representative answers in the current $A_{cur}$ using the original TC algorithm.
  - Remove duplicates by resolving all pairs of nodes in $A_{cur}$.
  - Additional cost: $O(|A_{cur}|^2)$

```latex
4 \text{ if } S = \text{Distinct then}
5 \quad \text{for each } v_i \in A_{cur} \text{ do}
6 \quad \quad \text{for each } v_j \neq v_i \in A_{cur} \text{ do}
7 \quad \quad \quad \text{if } \mathcal{R}(v_i, v_j) = \text{MustMerge then}
8 \quad \quad \quad \quad v_i \leftarrow v_i \oplus v_j
9 \quad \quad \quad A_{cur} \leftarrow A_{cur} - \{v_j\}
```
Exact answer semantics

- Compare all nodes in $A_{cur}$ with all nodes in $A_{cur} \cup V_{maybe}$
- Extra cost: $\Theta(|A_{cur}| \cdot |R|)$

```plaintext
10  else if $S = \text{Exact}$ then
11    for each $v_i \in A_{cur}$ do
12      for each $v_j \neq v_i \in A_{cur} \cup V_{maybe}$ do
13        if $\mathcal{R}(v_i, v_j) = \text{MustMerge}$ then
14          $v_i \leftarrow v_i \oplus v_j$
15        if $v_j \in A_{cur}$ then
16          $A_{cur} \leftarrow A_{cur} - \{v_j\}$
```
Exact answer semantics

- Compare all nodes in $A_{cur}$ with all nodes in $A_{cur} \cup V_{maybe}$
- Extra cost: $O(|A_{cur}| \cdot |R|)$

```plaintext
10  else if $S = \text{Exact}$ then
11     for each $v_i \in A_{cur}$ do
12         for each $v_j \neq v_i \in A_{cur} \cup V_{maybe}$ do
13             if $R(v_i, v_j) = \text{MustMerge}$ then
14                 $v_i \leftarrow v_i \oplus v_j$
15             if $v_j \in A_{cur}$ then
16                 $A_{cur} \leftarrow A_{cur} - \{v_j\}$
```

To produce distinct or exact answer, vestigial edges are considered unresolved.
**Lemma 2.** If the resolve function is always accurate, then QDA will compute answers that are: representationally, distinctly, or exactly equivalent to those in $C_{gt}$.

- Naturally, resolve functions are not always accurate -> no ER technique can guarantee correctness.
- They do not assume that resolve is always accurate.
Experimental Evaluation
Experimental Results

- Dataset:
  - Bibliographical entries from Google Scholar
  - Contains 50 researchers
    - 16396 records, 14.3% of which are duplicates

- Resolve Function
  - Soft-TF-IDF on titles
  - Jaro-Winkler distance on author names

- Blocking Techniques:
  - Bucketize records that might be duplicates
    - 1: First two letters of titles as the hash key
    - 2: Last two letters of titles as the hash key
QDA vs. TC (Transitive Closure)

Query: Select * From R Where cited >= t

Very similar: means bottleneck!
QDA vs. TC [Answer Semantics]

The larger the size of the answer, the higher the extra cost QDA will pay.
QDA speed-up for different types of queries
The effect of resolve function
Combined with blocking...
Effect of Edge Picking
Thank you!
谢谢！
آپ کا شکریہ
Ευχαριστώ!
Teşekkürler!