The Data Linkage Project: A Preliminary Report

The Pennsylvania State University

Dongwon Lee
dongwon@psu.edu

Outline
- Warm-Up
- Motivation & Problem Def.
- Data Linkage
  - Group Linkage
  - Adaptive Linkage
  - Parallel Linkage
  - Googled Linkage
- Conclusion

Penn State University

- Founded in 1855
- 23 campuses throughout PA state
- Main campus at State College, PA
- 84,000 students, 20,800 faculty
- $1.2 billion endowment
- "Nittany Lion"
- Penn State ≠ U. Penn

- Two CompSci-related divisions:
  - Dept. of Computer Science & Engineering (CSE)
  - College of Info. Sciences & Technology (IST)

Penn State University

- 5 DL/DB Faculty
- CSE:
  - Wang-Chien Lee
- IST:
  - C. Lee Giles
  - Dongwon Lee
  - Prasenjit Mitra
  - James Wang

- Active Collaboration

Penn State University

- State College, PA
  - Out of nowhere, but close to everywhere
- West: 2.5 hours to Pittsburgh
- East: 4 hours to New York
- South: 3 hours to Washington DC
- North: 3 hours to Buffalo
Penn State University

- In 2007, plan to hire 1-2 faculty on
  - Security
  - Risk Analysis
  - Data Mining

- Encourage to apply
  - http://ist.psu.edu/ist/facultyrecruiting/

This Talk

- Mainly based on:
  - “Group Linkage”, ICDE 2007
  - “Improving Grouped-Entity Resolution using Quasi-Cliques”, ICDM 2006
  - “Googled Name Linkage”, Penn State TR, 2006
  - “Search Engine Driven Author Name Disambiguation”, JCDL 2006

- Slides for this talk are available at:
  - http://pike.psu.edu => talks

Credits

- Students @ Penn State
  - Ergin Elmacioglu, Hung-sik Kim, Byung-Won On, and Su Yan

- Collaborators
  - Min-Yen Kan (NUS, Singapore)
  - Jaewoo Kang (Korea U., Korea)
  - Nick Koudas (U. Toronto, Canada)
  - Jian Pei (Simon Fraser U., Canada)
  - Divesh Srivastava (AT&T Labs – Research, USA)

Outline

- Warm-Up
- Motivation & Problem Def.
- Data Linkage
  - Group Linkage
  - Adaptive Linkage
  - Parallel Linkage
  - Googled Linkage
- Conclusion

Eg. Records

- Customer Addresses
  - Dongwon Lee, 110 E. Foster Ave. #410, State College, PA, 16802
  - LEE Dong, 110 East Foster Avenue Apartment 410, University Park, PA 16802-2343

- Citations
  - [SM83] G. Salton et al. 1983

Eg. Authors

- Jeffrey D. Ullman
  @ Stanford Univ.
Eg. People Names
- Most common activities of Internet users
  - ~ 30% of search engine queries include person names (R. Guha et al., WWW 2004)
- Highly ambiguous
  - only 90,000 different names for 100 million people (U.S. Census Bureau)
  - Valid changes:
    - Customs: Lee, Dongwon vs. Dongwon Lee vs. LEE Dongwon
    - Marriage: Carol Dusseau vs. Carol Arpaci-Dusseau
    - Misc.: Sean Engelson vs. Shlomo Argamon

Eg. Products
- Products
  - Honda Fix vs. Honda Jazz
  - T-Fal vs. Tefal
  - Apple iPod Nano 4GB vs. 4GB iPod nano 4GB

Eg. Images

The Data Linkage Problem
- Many entities are without IDs
- When entities (e.g., people, products, companies, drugs) have variants \( \Rightarrow \) link them out

Data Linkage Problem: The process of detecting and correcting variant named entities that represent the same real-world object

Terminology
- Entity: real-world object (e.g., tuples, person names, product web pages, images, etc)
- We view that Entity has two main information
  - name: textual description of the entity
  - contents: metadata or contents describing the entity
- Eg:

Technical Landscape
- Abundant research in many disciplines
- Also known as:
  - DB: approximate join, merge/purge, record linkage
  - DL: citation matching, author name disambiguation
  - AI: identity matching
  - NLP: word sense disambiguation
  - IR: web query results clustering
  - LIS: name authority control
Technical Landscape

- In a nutshell, existing approaches often do:
  - For two entities, \( e1 \) and \( e2 \), capture their information in data structures, \( D(e1) \) and \( D(e2) \)
  - Measure the distance or similarity between data structures: \( \text{dist}(D(e1), D(e2)) = d \)
  - Determine for matching:
    - If \( d < \theta \), then \( e1 \) and \( e2 \) are matching entities
  - Work well for common applications

Join vs. Linkage

- **Approximate Join**
  - On short string or numeric data types
  - With Index
  - Match only
  - Tuples

- **Linkage**
  - On long string data types
    - Very expensive
    - Without Index
    - Match, Merge, Match, ...
      - Iterative
    - Tuples, Objects, Images, Documents, ...

New Challenges

- Record vs. Set vs. Vector …
- Millions of data to link
- Entities sometimes have
  - Too many (confusing) contents to use or
  - Too few contents to use
- Solutions for one scenario often do not work well for another

The Data Linkage Project

- We re-visit the linkage problem to be able to link:
  - Large-scale \( \Leftrightarrow \) Parallel Linkage
  - Arbitrary data objects \( \Leftrightarrow \) Group & Googled Linkage
  - Under various scenarios \( \Leftrightarrow \) Adaptive Linkage

Outline

- Warm-Up
- Motivation & Problem Def.
- Data Linkage
  - Group Linkage
  - Adaptive Linkage
  - Parallel Linkage
  - Googled Linkage
- Conclusion

Key Idea

- Often, entities have a wealth of information
  - Can do better than simple token co-occurrence
- When entities have a **group** of elements
  - Authors with a group of citations
  - Tax payers with a family names
  - Images with \( m \times n \) grids
**Popular Group Similarity**

- Jaccard
  \[ \text{sim}(g_1, g_2) = \frac{|g_1 \cap g_2|}{|g_1 \cup g_2|} \]

- Bipartite Matching
  - Cardinality
  - Weighted

- Clustering
  - Single vs. Complete vs. Average Link

**Intuition for better similarity**

- Two groups are similar if:
  - A large fraction of elements in the two groups form matching element pairs
  - There is high enough similarity between matching pairs of individual elements that constitute the two groups

**Group similarity**

- Two groups of elements:
  - \( g_1 = \{ r_{11}, r_{12}, \ldots, r_{1m_1} \} \), \( g_2 = \{ r_{21}, r_{22}, \ldots, r_{2m_2} \} \)
  - The group measure \( BM \) is the normalized weight of the maximum bipartite matching \( M \) in the bipartite graph \( (N = g_1 \cup g_2, E = g_1 \times g_2) \)
  \[
  BM_{g_1, g_2} = \sum_{(r_{1i}, r_{2j}) \in M} (\text{sim}(r_{1i}, r_{2j})) \\
  \text{such that } \text{sim}(r_{1i}, r_{2j}) > \theta \\
  BM(g_1, g_2) \geq 0
  \]

**Challenges**

- Each \( BM \) group measure uses the maximum weight bipartite matching
  - Bellman-Ford: \( O(V^2E) \)
  - Hungarian: \( O(V^3) \)
- Large number of groups to match
  - \( O(NM) \)

**Solution: Greedy matching**

- Bipartite matching computation is expensive because of the requirement
  - No node in the bipartite graph can have more than one edge incident on it
- Let’s relax this constraint:
  - For each element \( e_i \) in \( g_1 \), find an element \( e_j \) in \( g_2 \) with the highest element-level similarity \( \sim S_1 \)
  - For each element \( e_i \) in \( g_2 \), find an element \( e_j \) in \( g_1 \) with the highest element-level similarity \( \sim S_2 \)

**Upper/Lower Bounds**

\[
BM_{g_1, g_2} = \sum_{(r_{1i}, r_{2j}) \in M} (\text{sim}(r_{1i}, r_{2j})) \\
UB_{g_1, g_2} = \sum_{(r_{1i}, r_{2j}) \in M} (\text{sim}(r_{1i}, r_{2j})) \\
LB_{g_1, g_2} = \sum_{(r_{1i}, r_{2j}) \in M} (\text{sim}(r_{1i}, r_{2j}))
\]
Theorem & Algorithm

\[ BM_{\text{sim}, \rho}(g_1, g_2) \leq UB_{\text{sim}, \rho}(g_1, g_2) \]

- IF \( UB(g_1, g_2) < \theta \rightarrow BM(g_1, g_2) < \theta \rightarrow g_1 \neq g_2 \)

\[ LB_{\text{sim}, \rho}(g_1, g_2) \leq BM_{\text{sim}, \rho}(g_1, g_2) \]

- ELSE IF \( LB(g_1, g_2) \geq \theta \rightarrow BM(g_1, g_2) \geq \theta \rightarrow g_1 = g_2 \)
- ELSE, compute BM\((g_1, g_2)\)
  - This step is expensive

Goal: \( BM(g, g_i) \geq \theta \)

ACM Dataset

Left: 300 groups
Right: 700,000 groups

Better precision/recall but 20 times slower!

Summary of Group Linkage

- When entities have a group of elements in them, group linkage is useful and efficient
- But still somewhat slow
- Directions
  - More efficient implementation
  - Hierarchical Group Linkage: OLAP
  - Group => Tree, Graph
  - Application to Image Retrieval

Outline

- Warm-Up
- Motivation & Problem Def.
- Data Linkage
  - Group Linkage
  - Adaptive Linkage
  - Parallel Linkage
  - Googled Linkage
- Conclusion

Key Idea

- An array of parameters play an important role in existing linkage solutions
- Eg:
  - Attributes to use for blocking
  - Sliding window size
  - Choice of similarity functions: Edit, Jacard, TF/IDF
  - Minimum similarity threshold, \( \theta \)
- A linkage solution often picks certain values for these parameters (by human experts) and never change afterward \( \Leftrightarrow \) Why not?
Case Study: SNM

- Sorted Neighborhood Method (SNM)
  - Merge/Purge problem
  - A fixed size window, \( W \), slides from the beginning to the end
  - Within each window, all entities are compared pair-wise
- Adaptive-SNM
  - In watching videos, if subsequent frames are
    - Similar \( \rightarrow \) fast-forward
    - Dissimilar \( \rightarrow \) fast-backward
  - Adaptively adjust \( W \) to maximize objective functions – e.g., accuracy, speed

Example

- Entity set \( E = \{e_1, \ldots, e_6\} \)
  - \( e_1 = \) "iPod"
  - \( e_2 = \) "iPod nano"
  - \( e_3 = \) "iPod mp3 player"
  - \( e_4 = \) "MS Zune"
  - \( e_5 = \) "Apple iPod 512MB"
  - \( e_6 = \) "MS Zune 30GB Player"

Address Dataset (dbgen)

Citation Dataset (CORA)

Summary of Adaptive Linkage

- Linkage solutions need to be
  - Adaptive
  - Flexible
  - Modifiable
- Directions
  - Re-visiting existing linkage solutions
  - Adaptively set their parameters
  - Machine learning and Data Mining techniques

Outline

- Warm-Up
- Motivation & Problem Def.
- Data Linkage
  - Group Linkage
  - Adaptive Linkage
- Parallel Linkage
  - Googled Linkage
- Conclusion
Key Idea

- The linkage problem has two building blocks
  - Match: $e_1 \sim e_2$?
  - Merge: $e_1 + e_2 = e_3$
- Iterative
- More complicated design from Sequential to Parallel
- In parallel processing, when data and tasks are partitioned and later merged
  - One can avoid substantial computation by exploiting the interplay btw. Match and Merge

Match($a_i$, $b_j$) in Nested-Loop

- If $a_i \sim b_j$, do next match($a_{i+1}$, $b_j$)
- If $a_i \sim b_j$:
  - Remove $a_i$ from $A$
  - No need for match($a_i$, $b_k$) s.t. $j<k<|B|$
  - No need for match($a_l$, $b_j$) s.t. $i<l<|A|$
- If $a_i$ contains $b_j$:
  - Remove $b_j$ from $B$
  - Impossible: $b_k$ contains $a_i$ & $b_k \sim a_j$
  - Cannot skip match($a_i$, $b_k$)
- If $a_i$ is contained by $b_j$: …

Summary of Parallel Linkage

- Three sequential linkage scenarios
  - Clean vs. Clean
  - Clean vs. Dirty
  - Dirty vs. Dirty
- Each sequential linkage generates three output sets
- Parallel linkage can merge output sets using only six merges, not nine merges
- N-processor parallel linkage

Outline

- Warm-Up
- Motivation & Problem Def.
- Data Linkage
  - Group Linkage
  - Adaptive Linkage
  - Parallel Linkage
  - Googled Linkage
- Conclusion

Key Idea

- When entities have a wealth of information, we can exploit them by capturing them as either Groups or Graphs
- But when entities do not have a wealth of information or have only noisy information, then what to do?
  - Ask PEOPLE what they think!

Hypothesis

- Use the Web as a collective knowledge of people
  - Hypothesis:
    
    If an entity $e_1$ is a duplicate of another entity $e_2$, and if $e_1$ frequently appears together with information $I$ on the Web, then $e_2$ may appear frequently with $I$ on the Web, too.
Eg. ACM DL Case

- Search results from Google:
  - "Jeffrey D. Ullman" 384,000 pages 45%
  - "Jeffrey D. Ullman" + "aho" 174,000 pages
  - "J. Ullman" 124,000 pages 33%
  - "J. Ullman" + "aho" 41,000 pages
  - "Shimon Ullman" 27,300 pages
  - "Shimon Ullman" + "aho" 66 pages

Overview

Step 1. Select representative data

- What to select
  - A single token "aho"
  - A key phrase "stanford professor"
  - A sentence or more?
- How to select
  - tf, tf*idf, latent topic models, ...
- How many to select
  - 1, 2, ... n
- Where to select from?
  - Contents of canonical entity, variant, or both

Step 2. Acquire the collective knowledge

- How to form the query?
  - Single information "I" (the most important data piece)
    - "J. D. Ullman" AND "Aho"
  - Multiple information "I1", "I2", "I3", ... (the most k important data pieces)
    - Conjunction or Disjunction or Hybrid
      - "J. D. Ullman" AND "Aho" AND/OR "database" AND/OR "vldb"...
- Formal evaluation of the effectiveness of such variations
  - Different heuristics based on
    - Availability, discriminative power of the data content
    - Popularity of the name, variants, other candidates

Step 3. Interpret the collective knowledge

For entities ec, ei, and information tc

- Page Counts
  - Jeffrey D. Ullman: J. Ullman 4/16
  - Jeffrey D. Ullman: Shimon Ullman 1/19
- URLs
  - Jeffrey D. Ullman: J. Ullman
    - portal.acm.org
    - inforlab.stanford.edu
    - en.wikipedia.org
    - theory.lcs.mit.edu
  - Jeffrey D. Ullman: Shimon Ullman
    - portal.acm.org
- Web Page Contents

Web Page Contents

- Use top-k returned Web pages for each entity
- Two alternatives for sim(ec, ei):
  - Group distance between two sets of top-k web pages
  - Represent each set by a single Virtual Document
  - Apply document comparison metrics on Virtual Doc.
- Heuristics for creating Virtual documents:
Step 3. Interpret the collective knowledge

- **Web Page Contents**
  - \( \text{sim}(e_e, e_i) = \text{doc_sim}(vdoc(e_e), vdoc(e_i)) \)
- **Document Similarity metrics**:
  - \( \text{sim}_{\text{secure}}(D_e, D_i) = \frac{\text{internal}(D_e) \text{ internal}(D_i)}{\text{internal}(D_e) + \text{internal}(D_i) - \text{internal}(D_e \cap D_i)} \)
  - \( \text{sim}_{\text{cosine}}(D_e, D_i) = \frac{\sum \text{cos}(vdoc(e_e), vdoc(e_i))}{|D_e| |D_i|} \)
  - \( \text{sim}_{\text{long-model}}(D_e, D_i) = \text{sim}(\theta_e, \theta_i) = -KL(\theta_i \| \theta_e) - \sum_k p(k|\theta_i) \log \frac{p(k|\theta_i)}{p(k|\theta_e)} \)

---

Results with URL and Host

ACM data set:
- 43 authors
- 142 citations/author
- 21 candidates/block
- 3.1 citations/candidate
- 1.8 name variants/block
- 6.7 citations/variant

Results with Web Pages

ACM data set:
- 43 authors
- 142 citations/author
- 21 candidates/block
- 3.1 citations/candidate
- 1.8 name variants/block
- 6.7 citations/variant

Results with Web Pages (cont)

IMDB data set:
- 50 actors
- 24 titles/entity
- 20 candidates/block
- 24 titles/candidate
- 1 name variant/block
- 23.5 titles/variant

Scalability

- Not scalable:
  - A large number of Web accesses
  - Network traffic, load of search engine and web sites

- Solutions:
  - Local snapshot of the Web
  - Stanford WebBase Project
  - ~100 million web pages from >50,000 sites including many .edu domains
  - Downloaded the half of the data & filtered
  - Local snapshot containing 3.5 million relevant pages

Summary of Googled Linkage

- When entities lack evidences for linkage
  - Googled linkage can be useful
  - But terribly slow
- Directions:
  - Balance btw. Googling vs. Local Cache
  - Applications
    - Medical literature mining: acronym-fullname
Conclusion

- Linkage problems are common
- Four novel directions
  - Group Linkage
  - Adaptive Linkage
  - Parallel Linkage
  - Googled Linkage
- We are only in a preliminary stage
- Many interesting yet practical problems
  - Surged interest from DB, DM, WWW, NLP, and AI communities