Large-Scale Entity Extraction, Disambiguation and Linkage

Dr. Flavio Villanustre, HPCC Systems
LexisNexis Risk Solutions
April 9th, 2013

Risk Solutions
LexisNexis

- Over 35 years of really large data experience
- Information solutions to enterprise customers
- More than $6 billion in revenue (2010)
- Acquired by Reed Elsevier in 1994
- RE Stock Symbols: valued at $12 billion in 2010; [NYSE: ENL; NYSE: RUK]; 34,000 employees
- Developing and using the distributed data-intensive HPCC Systems platform for over a decade
- Released the HPCC Systems platform as an Open Source project in 2011
Providing some context

- Multiple data sources containing “attributes” or “observables”, referring to entities and their properties.
- Entities can be people, businesses, assets, licenses, etc., but can also be items in a warehouse, manufacturers, suppliers and customers. Entities could also be research topics and patents, universities, researchers and companies (or any other concepts that correspond to the ontology in your domain field).
- Data can be structured or unstructured, usually large, and quickly builds up to large volumes.
- It’s not just about identifying the entities, it’s also about creating a graph of relationships & properties across them.
Precise entity extraction and resolution are usually critical in many areas, such as Master Data Management, Data Warehousing, Business Intelligence, etc.

Normally involve:
- Linking all attributes belonging to each entity together
- Identifying relationships between entities to create a complete graph
- Categorizing the strength of the relationships
- Having definable levels of precision in this process
- Handling fuzziness (phonetics, string distance, etc.)

Data needs to be loaded, cleansed, normalized and standardized before linkage can start
Correctly identifying entities in structured and unstructured data sources, can be challenging as:

- There are normally no reliable unique identifiers to be leveraged.
- Content can be dirty, incomplete and have errors.
- Heuristic rules-based systems require high maintenance and are not portable across problem domains, geographies and languages.
- Complete pre-existing dictionaries aren’t available.
- Handling the resulting massive graphs creates other problems, particularly around partitioning and distribution.
- Comparing every pair of strings is a $O(N^2)$ problem.
The LexisNexis Open Source Platform: HPCC Systems

High Performance Computing Cluster (HPCC)

Unstructured Semi-structured Big Data

Big Data

Extraction Transformation Loading

THOR Cluster (Data Refinery)

Linux

Concurrent Realtime Delivery

ROXIE Cluster (Data Delivery)

Linux

ESP

Query Results

ECL Developer Using ECL IDE
Detailed HPCC Systems Architecture

- Auxiliary Components
  - Data
  - Landing Zone
  - ESP Server
  - Network Switch

- Nodes
  - Thor Master
  - Node 1
  - Node ...
  - Node n

- Support Nodes
  - Dali
  - Dali Backup
  - Authent / Authority

- Data Flow
  - Data from Thor
  - Data to Roxie

- Thor
  (Batch Job Execution Engine + DFS)
  Physical Layout Schematic Diagram

- Roxie
  (Rapid Data Delivery Engine)
  Physical Layout Schematic Diagram
Enterprise Control Language (ECL)

- Declarative dataflow oriented programming language
- Free from side effects
- Designed for a specific problem domain (data-intensive computing), which makes resulting programs clearer, more compact, and more expressive
- Provides a more natural way to think about data processing problems for large distributed datasets
- Execution is determined from the sequence of dataflows and transformations.
- ECL incorporates transparent and implicit parallelism regardless of the size of the computing cluster
- The ECL compiler generates highly optimized C++ for execution
- Provides a comprehensive IDE and programming tools, including an Eclipse plugin
- There is a large library of efficient modules, to handle common data manipulation tasks
Data preparation process

Load

Profile

Assess Quality

Standardize

Normalize

Parse & Cleanse

Integrate in Base File

Create Specificities

Link
Probabilistic Record Linkage

- Identifying and linking all records referring to the same entity across all data sources
- Also known as “specificity based linking”
- It usually involves:
  - Building specificity tables for all values present in the data
  - Value and field specificity as one of the main metrics for the linkage process
  - Iterative process to progressively identify these clusters of attributes until convergence
  - Use of phonetics, string distance and other metrics to allow for fuzzy matching
How It Works

• What is the probability that two records with identical first and last name, and city are referring to the same entity?
  • For the “John Smith, Atlanta” case
  • For the “Flavio Villanustre, Atlanta” case
• What if there are typos in the data? How do these probabilities change?
• How do we build relationships across entities?
  • Cohabitation
  • Shared ownership of assets
  • Transactional information
  • Correlation
Examples of Probabilistic Records Linkage

Example 1
- First Name: John, John
- Last Name: Smith, Smith
- City: Atlanta, Atlanta
- Phone: 770-555-5555, 770-555-1234

Example 2
- First Name: Flavio, Flavio
- Last Name: Villanustre, Villanustre
- City: Atlanta, Atlanta
- Phone: 770-555-5555, 770-555-1234

Example 3
- First Name: Flavio, Favio
- Last Name: Villanustre, Villanuestre
- City: Atlanta, Atlanta
- Phone: 770-555-5555, 770-555-1234
Data preparation process

Load

- Profile
- Validate

Standardize

- Parse & Cleanse

Integrate in Base File

Create Specificities

Link
• Even in a concise and very high level dataflow programming language like ECL, defining each of these steps can amount to significant work!
• Can an even-higher-level-programming-abstraction help us?
• What if we could let the data scientist focus on the data integration and linking aspects of the process, and leave the rest to a code generator?
• Meet SALT!
Scalable Automated Linking Technology (SALT)

- The acronym stands for “Scalable Automated Linking Technology”
- Template based ECL code generator
- Provides for automated data profiling, QA/QC, parsing, cleansing, normalization and standardization
- Sophisticated specificity based linking and clustering
- Significant productivity boost!

Data Sources

Data Preparation Processes (ETL)

- Profiling
- Parsing
- Cleansing
- Normalization
- Standardization

Match Data Weights and Threshold Computation

Blocking/Searching

Weight Assignment and Record Comparison

Record Match Decision

Linked Data File

Record Linkage Processes

42 Lines of SALT
3,980 Lines of ECL
482,410 Lines of C++
Could we make this even better?

- How about a graphical UI to data integration and data linking?
- No code to write but retain the flexibility to tune the entire process
- Let the data scientist focus on the problem at hand and forget about the implementation details
- Meet SmartView!
Beyond SALT: Smart View

0 Lines of Code!
42 Lines of SALT
3,980 Lines of ECL
482,410 Lines of C++
Beyond SALT: Smart View

- Same Cluster ID
- One Form of the Name
- Another Form of the Same Name
- Same Name, Same Email Address

<table>
<thead>
<tr>
<th>Cluster ID</th>
<th>Full Name Last</th>
<th>Original Cluster ID</th>
<th>Business Name</th>
<th>City</th>
<th>Education School</th>
<th>Email</th>
<th>Enterprise Code</th>
<th>Enterprise Type</th>
<th>Fax</th>
<th>First Name</th>
<th>Last Name</th>
<th>Leg</th>
</tr>
</thead>
<tbody>
<tr>
<td>312396285184991410</td>
<td>9</td>
<td>488-05219148138647</td>
<td>488-05219148138647</td>
<td>长安大学</td>
<td>长安大学</td>
<td><a href="mailto:HUANWEN.QU@GMAIL.COM">HUANWEN.QU@GMAIL.COM</a></td>
<td></td>
<td></td>
<td></td>
<td>9715</td>
<td>男</td>
<td>男</td>
</tr>
<tr>
<td>359628668398610267</td>
<td>9</td>
<td>4268513724477316563</td>
<td>4268513724477316563</td>
<td>北京</td>
<td>北京</td>
<td><a href="mailto:HUANWEN.QU@GMAIL.COM">HUANWEN.QU@GMAIL.COM</a></td>
<td></td>
<td></td>
<td></td>
<td>9715</td>
<td>男</td>
<td>男</td>
</tr>
<tr>
<td>36961845496056218</td>
<td>9</td>
<td>36961845496056218</td>
<td>36961845496056218</td>
<td>范兰大学</td>
<td>范兰大学</td>
<td><a href="mailto:HUANWEN.QU@GMAIL.COM">HUANWEN.QU@GMAIL.COM</a></td>
<td></td>
<td></td>
<td></td>
<td>9715</td>
<td>男</td>
<td>男</td>
</tr>
<tr>
<td>368651845496056218</td>
<td>9</td>
<td>368651845496056218</td>
<td>368651845496056218</td>
<td>范兰大学</td>
<td>范兰大学</td>
<td><a href="mailto:HUANWEN.QU@GMAIL.COM">HUANWEN.QU@GMAIL.COM</a></td>
<td></td>
<td></td>
<td></td>
<td>9715</td>
<td>男</td>
<td>男</td>
</tr>
<tr>
<td>186507393178446218</td>
<td>9</td>
<td>56651845496056218</td>
<td>56651845496056218</td>
<td>范兰大学</td>
<td>范兰大学</td>
<td><a href="mailto:HUANWEN.QU@GMAIL.COM">HUANWEN.QU@GMAIL.COM</a></td>
<td></td>
<td></td>
<td></td>
<td>9715</td>
<td>男</td>
<td>男</td>
</tr>
<tr>
<td>254508447753545260</td>
<td>9</td>
<td>545016575297331469</td>
<td>545016575297331469</td>
<td>北京</td>
<td>北京</td>
<td><a href="mailto:YWHUI.QU@SNA.COM">YWHUI.QU@SNA.COM</a></td>
<td></td>
<td></td>
<td></td>
<td>9715</td>
<td>男</td>
<td>男</td>
</tr>
</tbody>
</table>
Beyond SALT: Smart View
Entity Extraction, Resolution and Linkage are dependent on the entire data integration workflow.

Rules-based data linkage methods fall short of expectations (high maintenance, don’t scale to other problem domains or foreign languages without significant rewrite effort).

Probabilistic record linkage provides an extensive and highly scalable framework, but can require significant computational complexity.

Open Source distributed data-intensive compute solutions, like the HPCC Systems platform, offer the best option, when scaling up to millions of entities and billions of relationships.
Questions

Email: Flavio.Villanustre@LexisNexis.com
Website: http://hpccsystems.com
Links for More Information

- HPCC Systems Platform: http://hpccsystems.com
- SALT data profiling demo: http://hpccsystems.com/demos
- The HPCC Systems blog: http://hpccsystems.com/blog
- Our GitHub portal: https://github.com/hpcc-systems
- Community Forums: http://hpccsystems.com/bb
- Free Online Training http://learn.lexisnexis.com/hpcc