Graph-based keyword search
in heterogeneous data sources

Angelos Anadiotis, Mhd Yamen Haddad, Ioana Manolescu

Cedar Team, Inria and Institut Polytechnique de Paris
Introduction

- Problem: having heterogeneous data sources with different formats, we want to be able to search across these data sources by:
  1. Model the underlying data as a graph.
  2. Given a set of search terms, our algorithm finds links between them within and across the heterogeneous datasets included in the graph.
Introduction

Public officials transparency high authority (CSV)

<table>
<thead>
<tr>
<th>Name</th>
<th>Owner</th>
<th>Location</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dar Gyucy</td>
<td>P. Balkany</td>
<td>Marrakech</td>
<td>Real Estate</td>
</tr>
<tr>
<td>Moulin Cossy</td>
<td>I. Balkany</td>
<td>Giverny</td>
<td>Real Estate</td>
</tr>
</tbody>
</table>

dbpedia.org (RDF)
{
  dbr:Marrakech
    dbr:name      "Marrakech"
    rdf:type      dbo:City ;
    dbo:country   dbr:Morrocco .
  dbr:Morrocco
    dbr:name      "Morocco"
    rdf:type      dbo:Country
    dbo:locatedIn dbr:Africa .
  dbr:CentralAfricanRepublic
    dbr:name      "Central African Republic"
    dbo:locatedIn dbr:Africa .
}

National Directory of Elected Officials (JSON)
[
  {
    name: "Levallois-Perret",
    mayor: "P. Balkany",
    city-council: [
      {name: "I. Balkany"}, ...
    ], ...
  }
]

Libération – Nov. 13, 2014 (Text)

**Balkany mineur de fonds**

L’élue de Levallois-Perret est soupçonnée d’avoir touché 5 millions de dollars de commission en 2009 grâce à son rôle d’intermédiaire entre Areva et la Centrafrique dans le dossier Uranin.

[...]

BDA 2020 A. Anadiotis, M.Y. Haddad, I. Manolescu
Finding Connections in a Heterogeneous Graph

What are the connections between .... ?
Problem statement

● Given the graph $G = (N, E)$ built out of the datasets $D$ and a query keywords $Q = \{w_1, \ldots, w_m\}$, return the $k$ highest-score minimal answer trees.

● An answer tree a set of edges which (i) form a tree (ii) for each $w_i$, contain at least one node whose label matches $w_i$.

● We are interested in minimal answer trees, that is:
  1. Removing an edge from the tree should make it lack some query keywords $w_i$.
  2. If a query keyword $w_i$ matches the label of more than one nodes in the answer tree, then all these matching nodes must be equivalent.
Search space and complexity

- The problem is related to the (Group) Steiner Tree Problem
- Given weighted graph $G$, and nodes $n_1, \ldots, n_m$, the Steiner Tree Problem (STP) consists of finding the smallest-cost tree in $G$ that connects all the nodes.
  - Known NP-hard problem in $|G|$
- Differences with our problem:
  - Each edge can be taken in any direction: exponential increase in search space size
  - We need the $k$ smallest-cost trees, not just one.
  - Each keyword may match several nodes, not just one.
- Our approach: enumerate solutions until time-out or max number of solutions reached.
  - Return best $k$ solutions found
Answering keywords queries (Grow and Merge)

- Each node will form one node partial Tree.
- Two Transformation can be applied:
  - Grow(t,e)
  - Merge(t1,t2)
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of \( p \) nodes requires \( p^2 \) sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of \( p \) nodes requires \( p^2 \) sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Search in heterogeneous graphs

- Equivalent set of $p$ nodes requires $p^2$ sameAs edges.
- Instead: use representative.
- How to modify grow in this case?
- Needs to extend grow.
  - Grow-to-Equivalent.
  - Grow-to-Representative.
Scoring answer trees

● We used a linear combination of:
  1. **Matching score**: reflects the quality of the answer tree (how well its leaves match the query terms).
  2. **Connection score**: reflects the quality of the tree connecting the edges.

● Our connection score based on:
  1. Edge confidence (probability)
  2. Edge specificity: how remarkable (rare) the edge label is for each node
Experiment evaluation

- used a server equipped with 2x10-core Intel Xeon E5-2640 CPUs clocked at 2.40GHz, and 128GB DRAM.
- Postgres 9.6.5 to store and query the graph for nodes, edges and labels.
- The search algorithms are implemented in a Java and used JDBC to connect to the database.
- We run experiments on:
  - Synthetic datasets: for controlled experiments.
  - Real-world dataset: based on data we obtained from journalists about French politics.
Experiments (Real-World dataset)

- Our dataset is a corpus of 462 HTML articles crawled from the French online newspaper Mediapart with the search keywords “gilets jaunes”.
- The resulted Graph consists of 90626 edges and 65868 nodes, out of which 1525 correspond to people, 1240 to locations and 1050 to organizations.
- We query the graph using queries of one, two and three different keywords.
## Experiments (Real-World dataset)

<table>
<thead>
<tr>
<th>Query keyword(s)</th>
<th>Answers</th>
<th>Answer trees</th>
<th>Time to 1st (ms)</th>
<th>Total time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macron</td>
<td>118</td>
<td>0</td>
<td>179</td>
<td>390</td>
</tr>
<tr>
<td>Trump</td>
<td>10</td>
<td>0</td>
<td>26</td>
<td>36</td>
</tr>
<tr>
<td>Melenchon</td>
<td>8</td>
<td>0</td>
<td>31</td>
<td>39</td>
</tr>
<tr>
<td>Christophe, Dettinger</td>
<td>1105</td>
<td>319611</td>
<td>136</td>
<td>123932</td>
</tr>
<tr>
<td>Etienne, Chouard, Rodrigues</td>
<td>1</td>
<td>194</td>
<td>144</td>
<td>146</td>
</tr>
<tr>
<td>Thierry–Paul, Valette, Drouet</td>
<td>0</td>
<td>300813</td>
<td>N/A</td>
<td>120001</td>
</tr>
<tr>
<td>Melenchon, Aubry</td>
<td>9</td>
<td>284</td>
<td>38</td>
<td>929</td>
</tr>
<tr>
<td>Castaner, flashball</td>
<td>17</td>
<td>1724</td>
<td>61</td>
<td>545</td>
</tr>
<tr>
<td>Drouet, Levavasseur</td>
<td>18</td>
<td>518</td>
<td>145</td>
<td>309</td>
</tr>
<tr>
<td>Dupont–Aignan, Chalencon</td>
<td>21</td>
<td>1850</td>
<td>53</td>
<td>393</td>
</tr>
<tr>
<td>Estrosi, Castaner</td>
<td>16</td>
<td>2203</td>
<td>205</td>
<td>529</td>
</tr>
<tr>
<td>Alexis, Corbiere, Ruffin</td>
<td>11</td>
<td>3782</td>
<td>57</td>
<td>1022</td>
</tr>
<tr>
<td>Macron, Nunez</td>
<td>13</td>
<td>4107</td>
<td>1511</td>
<td>1561</td>
</tr>
<tr>
<td>Hamon, Drouet</td>
<td>5</td>
<td>421</td>
<td>71</td>
<td>145</td>
</tr>
<tr>
<td>Drouet, Ludosky</td>
<td>27</td>
<td>486</td>
<td>43</td>
<td>145</td>
</tr>
<tr>
<td>Salvini, Ludosky</td>
<td>17</td>
<td>1156</td>
<td>111</td>
<td>375</td>
</tr>
<tr>
<td>Salvini, Chouard</td>
<td>16</td>
<td>3205</td>
<td>76</td>
<td>710</td>
</tr>
<tr>
<td>Corbiere, Drouet</td>
<td>13</td>
<td>2341</td>
<td>129</td>
<td>673</td>
</tr>
<tr>
<td>Cauchy, Drouet</td>
<td>22</td>
<td>516</td>
<td>96</td>
<td>260</td>
</tr>
<tr>
<td>Benalla, Nunez</td>
<td>15</td>
<td>1027</td>
<td>199</td>
<td>347</td>
</tr>
</tbody>
</table>
Thank you / Questions?

Try ConnectionLens: https://gitlab.inria.fr/cedar/connectionlens
Project Site: https://sourcessay.inria.fr
Back up slides
Experiments (Synthetic datasets)

- Line Graphs:

Line Graph of size K

Figure 7: Line graph execution time
Experiments (Synthetic datasets)

- Chain Graphs:

![Chain Graphs](image)

(a) Time to find the first answer

(b) Time to find all answers

Figure 8: Chain graph execution time
Experiments (Synthetic datasets)

- Star Graph:

![Graph](image_url)

(a) Time to find the first answer
(b) Time to find all answers

Figure 9: Star graph execution time
Edge specificity

- The specificity of an edge \( e = n_1 \rightarrow n_2 \) defined as:

\[
s(e) = \frac{2}{(N_{n_1 \rightarrow}^l + N_{\rightarrow n_2}^l)}
\]
SEARCH IN HETEROGENEOUS GRAPHS

- To tackle the multi-datasets graphs, we need to extend Grow to allow it to traverse not just data edges, but also similarity edges between nodes of the same or different datasets. We proposed the following:
  1. **Grow-to-equivalent**: allow Grow to also add an equivalence edge to the root of a tree. This is very inefficient.
  2. **Grow-to-representative**: instead of growing through all equivalent edges, we only allow growing to the representative. This will be more efficient compare to 1.
Answering keywords queries (GAM Algorithm)

- Priority used in U: at any point, U gives the highest-priority \((t, e)\) pair, which determines the operations performed next.
  1. Trees matching many query keywords are preferable, to go toward complete query answers.
  2. At the same number of matched keywords, smaller trees are preferable in order not to miss small answers.
  3. Finally, among \((t_1, e_1), (t_2, e_2)\) with the same number of nodes and matched keywords, we prefer the pair with the higher specificity edge.

- The algorithm only develops minimal trees. This is guaranteed:
  1. In Grow, Grow2Rep: we check not only that the newly added does not close a cycle, but also that the matches present in the new tree satisfy our minimality condition.
Answering keywords queries (GAM Algorithm)

Algorithm **GAMSearch**(query \( Q = \{w_1, w_2, \ldots, w_k\} \))

1. For each \( w_i, 1 \leq i \leq k \)
   - For each node \( n_i \) matching \( w_i \), let \( t_i \) be the 1-node tree consisting of \( n_i \); process(\( t_i \))
2. Initial \( \text{MERGE}^* \): try to merge every pair of trees from \( E \), and process any resulting answer tree.
3. Initialize \( U \) (empty so far):
   -(a) Create Grow opportunities: Insert into \( U \) the pair \( (t, e) \), for each \( t \in E \) and \( e \) a data or similarity edge adjacent to \( t \)'s root.
   -(b) Create Grow2Rep opportunities: Insert into \( U \) the pair \( (t, n \rightarrow n_{rep}) \) for each \( t \in E \) whose root is \( n \), such that the representative of \( n \) is \( n_{rep} \neq n \).
4. While \( U \) is not empty)
   -(a) Pop out of \( U \) the highest-priority pair \( (t, e) \).
   -(b) Apply the corresponding Grow or Grow2Rep, resulting in a new tree \( t'' \); process(\( t'' \)).
   -(c) If \( t'' \) was not already in \( E \), aggressively \( \text{MERGE}^* \):
      -(ii) Let \( NT \) be a set of new trees obtained from the \( \text{MERGE} \) (initially \( \emptyset \)).
      -(ii) Let \( p_1 \) be the keyword set of \( t'' \)
      -(iii) For each keyword subset \( p_2 \) that is a key within \( K \), and such that \( p_1 \cap p_2 = \emptyset \)
         -(d) For each tree \( t_i \) that corresponds to \( p_2 \), try to merge \( t'' \) with \( t_i \). Process any possible result; if it is new (not in \( E \) previously), add it to \( NT \).
   -(d) Re-plenish \( U \) (add more entries in it). This is performed as in step 3 but based on \( NT \) (not on \( E \)).
Answering keywords queries (GAM Algorithm)

Procedure \texttt{process}(\text{tree } t)

- if $t$ is not already in $E$
- then
  - add $t$ to $E$
  - if $t$ has matches for all the query keywords
  - then post-process $t$ if needed; output the result as an answer
- else insert $t$ into $K$