Schemaless Graph Querying

Shengqi Yang, Yinghui Wu, Huan Sun and Xifeng Yan
UC Santa Barbara
The Big Graphs

Internet graph

Social graph (Facebook)

Software graph (Apache)

Protein interaction graph (human)

Credits: Andrew R. Wade, Facebook, Apache, Ravi Iyengar
The Big Graphs

- It is not only about the scale!

Examples are countless: Google’s knowledge graph, LinkedIn’s professional network, Linked Open Data, etc.
The Graph Querying

- Problem: Subgraph querying
  - a.k.a., subgraph matching, subgraph isomorphism, etc

\[
\text{Pattern query} \quad \rightarrow \quad \text{Molecule graphs (Graph database)}
\]

Query

Actor, \sim 30 \text{ yrs}

UCB \quad M : I

Data graph

(Credit: linkeddata.org)

Answer

Chris Pine (1980) \quad J. J. Abrams

University of California, Berkeley
Mission: Impossible

(a.k.a., match, embedding)
Querying Big Graphs: the Challenges

- The data graph is schemaless
  - The data items, nodes and relations, are from various domains and thus are quite diverse.
  - There is no uniformed conversion followed by the data contributors.
- The query is schemaless
  - Queries are usually not requested in accordance with the metadata, which is transparent to the searchers.
- A “good” querying system should
  - Bridge the gap between query and data: match query to candidates.
  - Rank the results in a principled way.
  - Return top results in query time.
Querying Graphs: the DB approach

- **Keyword Search**
  - Find a root node (LCA in XML or trees) [Liu07][Koutrika06].
  - Group Steiner Tree [Ding07][Talukdar08].
  - Expansion from keyword matches [Bhalotia02][Dalvi08]].
  - Searching based on the score function [Hristidis03][Luo07][Kasneci08].
  - Searching based on indexes (e.g., reachability, distance, etc) [He07][Markowetz09][Li09][Qiao13].

- **Graph Search**
  - Exact subgraph matching, based on indexing techniques.
    - Search on graph databases [Shasha02][Yan04][Zhao07][Zou08].
    - Search on a single large graph [Ullman76][Cordella04][Shang08][Zhang09].
  - Approximate subgraph matching [Tian08][Mongiovi10][Khan13].

- Most of the works were primarily focusing on the efficiency. Result ranking is usually simply considered.
Querying Graphs: the IR and ML approach

- **IR based ranking**
  - Classic IR metrics: TF-IDF, BM25, etc.
    [Hristidis03][Liu06][Luo07]
  - Language models: a probabilistic space for each document
    [Elbassuoni109][Mass12].
  - The works are mostly based on linear combinations of the IR metrics, with human tuned fixed parameters.

- **Machine-learned ranking (MLR) [Liu09]**
  - The research in this domain targets on
    - How to design an effective (ensemble) ranking model.
    - How to estimate the parameters in the ranking model.
  - Application: rank the documents or the web pages.
  - Input: a score vector for a (query, doc) pair and the relevance score.
**Schemaless Graph Querying (SLQ)**

- To the users, no knowledge on the graph data is required.
  - Name the query and the search engine will do the rest.

- An efficient graph search technique that could fast extract the good results.

- A machine learning based ranking algorithm.
  - Robust to noisy and ambiguous queries.
  - Adaptive to the user’s preference.
SLQ - Outline

• The matching strategy

• The ranking and searching technique
  • Offline learning
  • Online query processing

• Performance and Demonstration
The Matching Technique

- A transformation-based matching approach
  - The users could freely post queries, without possessing any knowledge of the underlying data.
  - The querying system should automatically find the matches through a set of transformations.

**Query**

Actor, ~30 yrs

**A match**

Chris Pine (1980)

University of California, Berkeley

J. J. Abrams

Mission: Impossible

- **Acronym transformation** matches ‘UCB’ to ‘University of California, Berkeley’
- **Abbreviation transformation** matches ‘M : I’ to ‘Mission: Impossible’
- **Numeric transformation** matches ‘~30’ to ‘1980’.
The focus of this work is not to study all possible transformations. Any new transformation can be easily plugged into the framework.
The Matching Technique: New Challenges

- The transformation-based matching approach is likely to implicate many more results, compared to classic direct matching.
- To find all matching results (subgraph matching problem) is quite expansive. It is necessary to first suggest the “best” results to the users.
  - IR-based top-K search
  - We resort to ML-based top-K search
The Ranking and Searching Techniques

- The offline learning
  - A ranking model should encode the transformations between the query and the match
  - The ranking model should automatically determine its parameters.
    - User query logs
    - Automatic training data generation

- The online ranking and searching
  - Among many match candidates, an efficient algorithm is required to fast extract the top-K results, in terms of the ranking model.
The Ranking Model

- With a set of transformations \( \{f_i\} \), given a query \( Q \) and its match result \( R \), our ranking model considers
  - the node matching: from a query node \( v \) to its match \( \phi(v) \)
    \[
    F_V(v, \phi(v)) = \sum_i \alpha_i f_i(v, \phi(v))
    \]
  - the edge matching: from query edge \( e \) to its match \( \phi(e) \)
    \[
    F_E(e, \phi(e)) = \sum_i \beta_i f_i(e, \phi(e))
    \]
    (the edge match \( \phi(e) \) in the result could be a path)
- The overall ranking model: a probabilistic model based on Conditional Random Fields.
  \[
P(R \mid Q) \propto \exp(\sum_{v \in V_Q} F_V(v, \phi(v)) + \sum_{e \in E_Q} F_E(e, \phi(e)))
  \]
The Offline Learning

- It is clearly that the parameters \( \{\alpha_i; \beta_j\} \) need to be determined appropriately.
  - Classic IR method: the parameters are tuned by domain experts manually.
    - Specific domain knowledge is not sufficient for big graph data.
  - Supervised method: learning to rank
    - User query logs: not easy to acquire.
    - Manually label the training data: not practical and scalable.
  - Unsupervised method: automatic training data generation
    - A set of high quality training instances could be generated directly from the graph data.
    - Inspired by the advantage in learning high level representations with deep belief networks
      - Denoising Autoencoders [Vincent08]
Automatic training data generation

- The basis of denoising autoencoders
  - An explicit criteria for learning good representations
    - Robustness to partial destruction of the input.
  - Intuition: human is able to recognize partially corrupted images.
  - Application: image classification (search)

\[ \min L(p_1, p_1') \]

Training
random noise (0.1)

Searching
Good result

(picture credit: etidbits.com, youtube.com, blog.sina.com.cn)
Automatic training data generation

- **Sampling**: a set of subgraphs are randomly extracted from the data graph based on the templates.
- **Query generation**: the queries are generated by randomly adding noise (transformation) on the extracted subgraphs.
- **Searching**: search the generated queries on the data graph.
- **Labeling**: the results are labeled based on the original subgraph.
- **Training**: the queries, with the labeled results, are then used to estimate the parameters of the ranking model.
Online Query Processing

• Exact subgraph matching is quite expansive
  • The transformations produce many match candidates.
  • A NP-hard problem by reducing to graph isomorphism.
  • A little (possible) compromise on the quality of the results
could lead to a significant improvement on the efficiency.

• A fast top-K search algorithm
  • Indexing technique.
  • Two effective heuristics in the search algorithm.
    • Approximate inference in the CRFs ranking model.
    • Sketch graph based pruning strategy.
• Inference in graphical model (CRFs)
  • A CRFs model is constructed based on the query (as variables) and the node match candidates (as possible assignments to the variables)
  • Top-1 result: computing the most likely assignment.
    • Exact inference in general graphical model is still NP-hard [Sutton06].
    • Approximate inference: Loopy Belief Propagation (LoopyBP)
      • A message-passing algorithm:
        \[
        m_{ji}^{(t)}(u_i) = \max_{u_j} F_V(v_j, u_j)F_E((v_j, v_i), (u_j, u_i)) \prod_{v_k \in N(v_j) \backslash v_i} m_{kj}^{(t-1)}(u_j)
        \]
      • It is very efficient and empirically successful in many practical applications.
  • From Top-1 to Top-k: best max-marginal first algorithm [Yanover04]
Query Processing - Sketch Graph

- Problem: due to the various transformations, a large number of match candidates should be inspected by the LoopyBP algorithm.

- Solution: sketch graph
  - The candidates through the same transformation preserve the same matching score and thus can be grouped together.

1. Construct sketch graph
2. Search in sketch graph
3. Search in graph
Query Processing - Two-level Search

- The two-level search algorithm
  - Search in sketch graph
    - The size of the sketch graph is very small: only related to the size of the query and the number of the transformations.
    - The score of the upper-level match in the sketch graph is no less (upper bound) than the score of the lower-level match.
  - Search in data graph
    - Based on the upper-level match found in the sketch graph, the algorithm then extracts the lower-level matches in the data graph on a smaller set of match candidates.
    - Pruning: when the score of the lower-level match is smaller than that of a previous upper-level match.
**Other Designs - Indexing**

- An index is constructed based on the transformations. Given a query, its match candidates can be extracted readily from the index.
The Performance Evaluation

- **Dataset**
  
<table>
<thead>
<tr>
<th>Graph</th>
<th>Nodes</th>
<th>Edges</th>
<th>Node types</th>
<th>Relations</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBpedia</td>
<td>3.7M</td>
<td>20M</td>
<td>359</td>
<td>800</td>
<td>40G</td>
</tr>
<tr>
<td>YAGO2</td>
<td>2.9M</td>
<td>11M</td>
<td>6,543</td>
<td>349</td>
<td>18.5G</td>
</tr>
<tr>
<td>Freebase</td>
<td>40.3M</td>
<td>180M</td>
<td>10,110</td>
<td>9,101</td>
<td>88G</td>
</tr>
</tbody>
</table>

- **Query template**
  
  - DBPSB [Morsey11] : the query templates derived from query logs on DBpedia

- **Baseline**
  
  - Spark [Luo07] : a IR based ranking model that takes each node as a document and does not consider any structure information. The parameters are fixed.
  
  - SLQ: our method
  
  - Unit: a variant of SLQ, with equal parameter value
  
  - Card: a variant of SLQ, with parameter value as the selectivity of the corresponding transformation.
The Performance Evaluation – Case Study

Queries on DBpedia

- In the above cases, Spark gives low IR score and cannot identify matches for Query 2.
The Performance Evaluation - Effectiveness

- Queries are randomly generated on YAGO2 based the query templates
- Evaluation: MAP@5
- Change query size while fixing the transformation ratio.
  - SLQ shows the best result.
  - Large query (with more evidence) is more robust to the noise (the transformation).
- Change the transformation ratio.
  - The result degrades along with the increasing of the transformation ratio.
The Performance Evaluation - Efficiency

- Queries are randomly generated on Freebase based the query templates.
- The transformation ratio: 0.2~0.5.
- Baseline
  - Exact: exact search on all candidates.
  - NeMa [Khan13]: a graph search algorithm similar to LoopyBP.
- Increasing the graph size in a “streaming” mode
  - With the sketch graph, the SLQ outperforms the two baselines.
Conclusion

- A novel framework for schemaless graph querying.
- A matching strategy: transformation.
- A ranking model.
  - Incorporates the transformations.
  - Automatic training in the cold-start.
- Top-k search algorithm.
The Architecture and Future Work

- Natural language query process
- Runtime Indexing
- Storage; Multiple data graph
- Online Query Processing
  - 1. Query
  - 2. Matches
  - 3. Graph
  - 4. result
    - 4a. Top-K Query Process
    - 4b. Summarize
  - 5. Clicks
- Logger
- Web GUI / REST Service
- Database
  - Graph Info.
  - Index
- Offline Indexing
- Offline Learning
- Ranking Model
  - Cold-Start (Sampling)
  - Warm-Start (Feedback)
- Better ranking model
- Better top-k search algorithm
- Warm-start training
System Demonstration

Dataset selection
Keyword/SLQ Query
Result navigation bar

Graph Query Drawing / Result rendering panel
Information panel
More Applications

Text Mining/RDF -> Link entities across documents -> Build Information Network

Graph Visualization / Querying / Analysis
References

References (cont.)

References (cont.)