Multi-Example Search in Rich Information Graphs

Matteo Lissandrini, Davide Mottin, Themis Palpanas, Yannis Velegrakis
What are you looking for?
Search by a list of specifications

1. A Movie and an Actor
2. From the Movie return the Director
3. The Director has won an Award
4. The Movie is adapted from a Book
5. From the Book return the Author

Hard to Specify!

Too many options!
What is the schema?
Which specification is important?
<table>
<thead>
<tr>
<th>Search by <strong>Example</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lord of the Rings</td>
</tr>
<tr>
<td>E. Wood</td>
</tr>
<tr>
<td>P. Jackson</td>
</tr>
<tr>
<td>Tolkien</td>
</tr>
</tbody>
</table>
Search by Example

LotR

director

P. Jackson

actor

book

author

E. Wood

Tolkien
Search by Example

LotR
director
actor
E. Wood
P. Jackson
book
Tolkien
actor
book

Harry Potter,
director
actor
book
author
C. Rowling
P. J. Jackson
Search by Example

LotR
actor

tolkien
book author

E. Wood
director

P. Jackson
director

Harry Potter, James Bond, ...
director

actor

book author
Search by Example

LotR
actor

P. Jackson
director

E. Wood
actor

Tolkien
book author

Harry Potter, James Bond, …

director
director

book author
book author

More Intuitive!
Avoid list of specifications

More Intuitive!
Avoid list of specifications
ONE EXAMPLE IS NOT ENOUGH

When known examples are only Partial Specifications
Search by **Multiple Examples**

- **actor**: Harrison Ford (Indiana Jones)
- **spouse**: Michelle Obama
- **spouse**: Angelina Jolie (Tomb Raider)
- **actor**: Brad Pitt

---

ICDE 2018 – Paris | Matteo Lissandrini
ONE EXAMPLE IS NOT ENOUGH

When Results have Different Structures
Search by **Multiple Examples**

![Diagram](image-url)
Multi-Example Search

- **Multiple Simple Examples**
- Each Example describes an **Aspect**
- Results are **Combinations of aspects**
- Results have possibly **Multiple Structures**
Edge-labelled Multigraphs

$G: \langle V, E, L, \ell \rangle$
Exemplar Queries

Input: $Q_e$, an example *element* of interest
Output: set of elements in the *desired result set*

Exemplar Query Evaluation

- *match* $Q_e$ to *sample* $S$ in the graph $G$
- *find* the set of *elements* $A$ similar to $S$
  given a *similarity relation*
- [*OPTIONAL*] return only the *top-k subset* $A^K \subseteq A$

$A : \{ a \in D \mid a \approx S \}$
$\approx \rightarrow \text{similarity}$
Multi-Exemplar Queries

Our Problem Formulation:

Input: \( Q_e \), a set of example elements of interest
Output: set of elements in the desired result set

Exemplar Query Evaluation

- match each \( q \in Q_e \) to the set of samples \( S = \{ s_1, s_2, \ldots \} \) in the graph \( G \)
- find the set of elements \( A \) similar to each element in \( S \) given a similarity relation
- [OPTIONAL] return only the top-k subset \( A^K \)
(Multi-)Exemplar Queries on Graphs

**Single Sample** \( A : \{ a \in D \mid a \approx S \} \)

- **Similarity** \( (\approx) \): graph isomorphism
  
  \[ A : \{ a \text{ subgraph of } G \mid s \text{ isomorphic to } a \} \]

- **Challenge**: find ALL isomorphic graphs

**Multiple Samples** \( A : \{ a \in D \mid \forall s \in S . s \approx a \} \)

- **Similarity** \( (\approx) \): ?Subgraph-Isomorphism
  
  \[ A : \{ a \text{ subgraph of } G \mid \forall s \in S . s \text{ subgraph isomorphic to } a \} \]

**IS THIS CHARACTERIZATION ENOUGH?**

Graph Isomorphism is Transitive and Symmetric!

\( A \) is an Equivalence Class

\( s_i, s_j \in S . s_i \neq s_j \rightarrow A = \emptyset \)

Answers Are Subgraphs That Contain Structures Similar To Each Sample
Multi-Exemplar Answers on Graphs

Graph Similarity ($\approx$) Subgraph-Isomorphism

$$\{ a \in G \mid \forall s \in S . \text{ subgraph isomorphic to } a \}$$

What constitutes a good answer?

With No Restrictions the Entire Graph Is Accepted as Answer
Multi-Exemplar Answers on Graphs

Each answer should be **correct, complete** and **non redundant**: 

Ensure all Aspects are present & Limit Size of Answer Graphs

Answers: **WEAKLY CONNECTED SUBGRAPHS** with **NO SUPERFLOUS NODES** or **EDGES**

1. **Connectedness**
   \[ \forall n_1, n_2 \in V_A \; \exists \text{undirected path} \text{ that connects } n_1 \text{ to } n_2 \]
   ✓ **CORRECT**

2. **Consistency**
   \[ \forall n_A \in V_A \rightarrow \exists s \in S, n_s \in V_s \]
   Such that \( n_A \) maps to \( n_s \)
   ✓ **COMPLETE** ✓ **NON-REDUNDANT**
CHALLENGE!

To find Multi-Exemplar answers

1. Find **ALL isomorphic graphs** to ALL samples

2. Find which samples **combine into one connected answer**

For each sample needs to perform **Subgraph-Isomorphism Search**

**Candidate space = Cartesian Product** for all samples
Search Framework

Multi-exemplar Answering

| Input: Database $G: \langle V, E, \ell \rangle$ |
| Input: Samples $S: \langle s_1, \ldots, s_m \rangle$ |
| Output: Answers $\mathcal{A}$ |

1. $G \leftarrow \text{PARTIAL}(G, S)$
2. $\mathcal{A} \leftarrow \text{SEARCH}(G, S)$
3. return $\mathcal{A}$

Exploit Localized Search
Search Framework Optimizations

1. Find CANDIDATE REGIONS
   a. Remove Unused Edges
   b. Identify SEEDs
   c. Expand Around each seed

2. SEARCH within each region
   a. Avoid Cartesian Product
   b. Fast Merge of Partial Answers

Multi-exemplar Answering

| Input: Database $G : \langle V, E, \ell \rangle$ |
| Input: Samples $S : \langle s_1, \ldots, s_m \rangle$ |
| Output: Answers $\mathcal{A}$ |
| 1: $\mathcal{G} \leftarrow \text{PARTIAL}(G, S)$ |
| 2: $\mathcal{A} \leftarrow \text{SEARCH}(\mathcal{G}, S)$ |
| 3: return $\mathcal{A}$ |

Naïve Algorithm
- 1 single region
- Retrieves ALL Isomorphic-subgraphs
- Hash-JOIN for Fast merge of Partial Answers

2 Advanced Algorithms
- Fast & Fast+
Find Candidate Regions - Fast

Identify SEED:

S1

S2

S3

Min # of matches S1: 4  S2: 6  S3: 7

EXPAND around each seed:
Retrieve candidate Regions
Find Candidate Regions - Fast

Identify SEED:

Min # of matches S1: 4 S2: 6 S3: 7

Precomputed Statistics:
- Label frequency
- Label Pair frequency
- Star cardinality

\[
\mathbb{I}_{\text{star}}(l, c) = |\{G' \subseteq G : (V', E', \ell) \text{ is a star}
\wedge |E'| = c \wedge \exists (v_1, v_2) \in E'\}
\text{s.t. } \ell(v_1, v_2) = l|}
\]
Find Candidate Regions - Fast

Identify SEED:

S1

S2

S3

Min # of matches S1: 4  S2: 6  S3: 7

EXPAND around each seed:
Retrieve candidate Regions

Seed search requires Isomorphic-Search

Some Regions do not contain all structures
Find Candidate Regions – Fast+

Identify SEED:
- S1
- S2
- S3

With cardinality Estimation

Select SINGLE NODE
With bitset-mapping

EXPAND around each seed:
Retrieve candidate Regions

DISCARD incomplete regions
With bitset-mapping & before ISO-search
Fast Pruning with Bit-Vectors

Example: Detect JOIN-Node

\[
\begin{array}{cccc}
\text{actor} & \text{director} & \text{spouse} & \text{author} \\
\hline
\text{Barack Obama (S2)} & - & - & 1 & - \\
\text{Quentin Tarantino (S3)} & - & 1 & - & - \\
\text{Steven Spielberg (A1)} & - & 1 & 1 & - \\
\text{Kate Capshaw (A2)} & - & - & 1 & - \\
\text{BO} \lor \text{QT} & - & 1 & 1 & - \\
(\text{BO} \lor \text{QT}) \land \overline{\text{KC}} & - & 1 & - & - \\
(\text{BO} \lor \text{QT}) \land \overline{\text{SS}} & - & - & - & - \\
\end{array}
\]

(union)

(≠0)

(=0)

Example:

Detect JOIN-Node

\[
(\text{BO} \lor \text{QT}) \land \overline{\text{KC}}
\]

\[
(\text{BO} \lor \text{QT}) \land \overline{\text{SS}}
\]
Top-K

Weight Function for Nodes

Scoring Function

\[ \rho(A) = \frac{1}{|V_A|} \sum_{v \in V_A} w(v) \]

Theorem 2. Given the set of graph samples \( S \), and answers, \( A_1 \) and \( A_2 \), \( \forall s \in S.s \subseteq A_i \), via the isomorphism function \( \mu^s_{A_i} \), the node weight function \( w \), and the ranking \( \rho \) (Equation 1). It holds:

\[ \bar{\rho}(A_2) = \frac{\sum_s \sum_v w(\mu^s_{A_2}(v))}{\max_{s \in S}|V_s|} < \rho(A_1) \rightarrow \rho(A_2) < \rho(A_1) \] (4)

Skip Regions that will not produce answers with score High Enough

Multi-exemplar Answering

Input: Database \( G : (V, E, \ell) \)
Input: Samples \( S : \{s_1, \ldots, s_m\} \)
Output: Answers \( \mathcal{A} \)

1: \( \mathcal{G} \leftarrow \text{Partial}(G, S) \)
2: \( \mathcal{A} \leftarrow \text{Search}(\mathcal{G}, S) \)
3: return \( \mathcal{A} \)
Top-K

Scoring Function \( \rho(A) = \frac{1}{|V_A|} \sum_{v \in V_A} w(v) \)

Skip Regions that will not produce answers with score High Enough

Multi-exemplar Answering

Input: Database \( G : \langle V, E, \ell \rangle \)
Input: Samples \( S : \langle s_1, ..., s_m \rangle \)
Output: Answers \( A \)

1. \( G \leftarrow \text{PARTIAL}(G, S) \)
2. \( A \leftarrow \text{SEARCH}(G, S) \)
3. return \( A \)

Theorem 2. Given the set of graph samples \( S \), and answers, \( A_1 \) and \( A_2 \), \( \forall s \in S. s \subseteq A_i \), via the isomorphism function \( \mu^s_{A_i} \), the node weight function \( w \), and the ranking \( \rho \) (Equation 1). It holds:

\[
\rho(A_2) = \frac{\sum_s \sum_v w(\mu^s_{A_2}(v))}{\max_{s \in S}|V_s|} < \rho(A_1) \rightarrow \rho(A_2) < \rho(A_1)
\]
Experimental Evaluation

3 Algorithms:
   a) Naïve
   b) Fast (iso-graphs as seeds)
   c) Fast+ (nodes as seeds)

2 Large Real Datasets:
   a) YAGO +16.7M Edges
   b) Freebase +300M Edges

Tests: 100 Queries 2-5 Samples
   • Count isomorphic computations
   • Running Time
Evaluation Results

Datasets: Freebase (300M Edges) Yago (16.7M edges)
100 queries, from 2 to 5 samples

Saving Isomorphic Computations

AVERAGE MEDIAN

40–60% Reduction in Iso-Search Computations!
Evaluation Results

Datasets: Freebase (300M Edges) Yago (16.7M edges)
100 queries, from 2 to 5 samples

Running Times

Total time (sec) vs #Query Samples

Triangles: mQ-Naïve
Squares: mQ-Fast
Circles: mQ-Fast+

In some cases the “Fast” algorithm wastes computations

Faster 70% of queries
Saves up to 25secs on AVG

Datasets:
- Freebase (300M Edges)
- Yago (16.7M edges)

100 queries, from 2 to 5 samples

Methods:
- mQ-Naïve
- mQ-Fast
- mQ-Fast+

Average and Median

Datasets:
- Freebase (300M Edges)
- Yago (16.7M edges)

100 queries, from 2 to 5 samples

Methods:
- mQ-Naïve
- mQ-Fast
- mQ-Fast+

Average and Median
Conclusions

**Search via Multiple-Examples**
Find structures similar to a SET of input examples.

**Output:**
Composite Results
Containing characteristics from each Example

a Complete Example with all the desired characteristics is not known.

Characteristics combine in Multiple ways.

**Useful When:**
1. Exploit Localized Search
2. Bitset Pruning
3. Cardinality Estimation
4. Top-K Optimizations

**Optimizations:**
Localized Search is less effective With a Dense Graph!

Thank You!
Questions?

p.s. Now you can hire me!
Thank you!

Questions?
There is more . . .
Search Framework **Optimizations** (bis)

### Algorithm 1 mQ-Naive

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>function <code>PARTIAL(G, S)</code></td>
</tr>
<tr>
<td>2.</td>
<td>return <code>G : {G}</code></td>
</tr>
<tr>
<td>3.</td>
<td>function <code>SEARCH(G, S)</code></td>
</tr>
<tr>
<td>4.</td>
<td><code>A ← ∅</code></td>
</tr>
<tr>
<td>5.</td>
<td>for each <code>C ∈ G</code> do</td>
</tr>
<tr>
<td>6.</td>
<td>for each <code>s_i ∈ S</code> do</td>
</tr>
<tr>
<td>7.</td>
<td>`Ā_i ← {A ⊆ G</td>
</tr>
<tr>
<td>8.</td>
<td>`C ← arg min_{Ā_i ∈ ⟨Ā_1, ..., Ā_{</td>
</tr>
<tr>
<td>9.</td>
<td>while <code>C ≠ ∅</code> do</td>
</tr>
<tr>
<td>10.</td>
<td><code>c ← REMOVEONE(C)</code></td>
</tr>
<tr>
<td>11.</td>
<td>if <code>∀s ∈ S. s ⊆ c</code> then</td>
</tr>
<tr>
<td>12.</td>
<td><code>A ← A ∪ {c}</code></td>
</tr>
<tr>
<td>13.</td>
<td>else `C ← C ∪ CONNECT(c, S, ⟨Ā_1, ..., Ā_{</td>
</tr>
<tr>
<td>14.</td>
<td>return <code>A</code></td>
</tr>
</tbody>
</table>

### Algorithm 2 CONNECT+

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><code>C^+ ← ∅</code></td>
</tr>
<tr>
<td>2.</td>
<td>for each <code>n ∈ V_c</code> do</td>
</tr>
<tr>
<td>3.</td>
<td>for each <code>Ā_i ∈ H(n)</code> s.t. <code>s_i ⊈ c</code> do</td>
</tr>
<tr>
<td>4.</td>
<td><code>C^+ ← C^+ ∪ MERGE(c, Ā_i)</code></td>
</tr>
<tr>
<td>5.</td>
<td>return <code>C^+</code></td>
</tr>
</tbody>
</table>

**Naïve Algorithm**
- 1 single region
- Retrieves ALL Isomorphic-subgraphs
- **Hash-JOIN** for Fast merge of Partial Answers

**2 Advanced Algorithms**
- Fast & Fast+

---

Multi-exemplar Answering

**Input:** Database `G : (V, E, ℓ)`
**Input:** Samples `S : (s_1, ..., s_m)`
**Output:** Answers `A`

1. `G ← PARTIAL(G, S)`
2. `A ← SEARCH(G, S)`
3. return `A`
Cardinality Estimation

Precomputed Statistics:
- Label frequency
- Label Pair frequency
- Star cardinality

\[ \mathbb{I}_{\text{star}}(l, c) = |\{G' \subseteq G | G' : (V', E', \ell) \text{ is a star} \wedge |E'| = c \wedge \exists (v_1, v_2) \in E' \text{ s.t. } \ell(v_1, v_2) = l\}| \]