Design of Declarative Graph Query Languages: On the Choice between Value, Pattern and Object based Representations for Graphs

Hasan Jamil
Department of Computer Science
Wayne State University

IEEE ICDE Workshop on Graph Data Management Workshop
Washington DC
April 5, 2012
Biological Networks
Outline

- Algorithm v Query Language
  - **GraphQL (He and Singh 2008)**
  - SAGA (Tian et al, 2007), TALE (Tian and Patel, 2008), GADDI (Zhang, Li and Yang, 2009), NOVA (Zhu et al, 2010), etc.

- NetQL
  - Subgraph isomorphism (ICTAI 2009, SAC 2011)
    - IsoKEGG (BIBM 2010)
  - Graph reachability (CIKM 2010)
  - Top-k similar graphs (TCBB in press)
  - Network extraction (SAC 2010)
Many incarnations of graphs

(a) Undirected and unlabeled
(b) Directed and node labeled
(c) Undirected and fully labeled
nodes

<table>
<thead>
<tr>
<th>I</th>
<th>N</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td>H</td>
</tr>
<tr>
<td>1</td>
<td>b</td>
<td>Ni</td>
</tr>
<tr>
<td>1</td>
<td>c</td>
<td>Ca</td>
</tr>
<tr>
<td>2</td>
<td>a</td>
<td>H</td>
</tr>
<tr>
<td>2</td>
<td>c</td>
<td>O</td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>H</td>
</tr>
</tbody>
</table>

edges

<table>
<thead>
<tr>
<th>I</th>
<th>F</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>1</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>a</td>
<td>c</td>
</tr>
<tr>
<td>2</td>
<td>a</td>
<td>d</td>
</tr>
<tr>
<td>2</td>
<td>c</td>
<td>d</td>
</tr>
<tr>
<td>2</td>
<td>c</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>c</td>
</tr>
</tbody>
</table>

directed and labeled

<table>
<thead>
<tr>
<th>I</th>
<th>F</th>
<th>T</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td>b</td>
<td>inh</td>
</tr>
<tr>
<td>1</td>
<td>b</td>
<td>a</td>
<td>cat</td>
</tr>
<tr>
<td>2</td>
<td>a</td>
<td>c</td>
<td>phos</td>
</tr>
<tr>
<td>2</td>
<td>c</td>
<td>d</td>
<td>deh</td>
</tr>
<tr>
<td>2</td>
<td>c</td>
<td>a</td>
<td>act</td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>a</td>
<td>inh</td>
</tr>
</tbody>
</table>

undirected and unlabeled

(a) labeled nodes
(b) undirected and unlabeled
(c) directed and labeled
(d)
(e)
Subgraph isomorph of subgraph isomorphs of graphs

(a) Graph $g_1$

(b) Graph $g_2$

(c) Graph $g_3$

(d) Graph $g_4$

(e) Query graph $q$
Declarative graph querying

- The query to find the set of graphs $G$ such that each $g \square G$ has a subgraph isomorph in another set of graphs $G'$ such that a query graph $q$ is a subgraph isomorph of each $g' \square G'$.
  - The answer is $g_4$.
    - Cannot be computed in GraphQL (He and Singh, SIGMOD 2008) or GraphGrep (Giugno and Shasha, ICPR 2002), for example.
Research issues

- Computing this query requires a higher order query language
  - Variables need to range over set of tuples or structures
  - Completeness is at risk
  - Higher processing cost is also expected
- Compromise?
  - Develop operators such as
    - SQL aggregates
    - Data cube
    - Skyline
    - Association rule mining
compute *
from $r$ as $h$
where exists (compute *
  from (compute *
    from $r$ as $t$
    where exists (compute *
      from $r$ as $u$
      where $u.I = q$ and
        subisomorph($u,t$)) as $g$
      where subisomorph($g,h$));

(b)
Main Issues

- Representation that helps
  - Compare arbitrary graphs as single, perhaps complex, objects
    - Values
      - No unified view of a graph
    - Patterns
      - Need enumeration, GraphQL. Query limitations.
    - Objects
      - In the form of a special tuple
        - Allows access to a whole graph through a handle
        - Allows comparing whole graphs without pattern enumeration
      - But
        - Its higher order
        - Model graph comparisons as operators
Technicality

- Represent each graph as a pair \(<I, <V, E>>\) where \(I\) is a graph ID or handle, \(V\) is the set of vertices in graph \(I\), and \(E\) is the set of edges.
  - Extension for labeled graphs
    - \(<I, <<V, L_v>>, <E, L_e>>\>
  - Extension for directed graphs
    - Enforce symmetry for \(E\) v no symmetry

- Define graph operators that satisfy this structure, undefined otherwise

- Consequence?
  - Any relation can be restructured to represent graphs
Dependencies

Relation pairs: nodes and edges

\[ X \subseteq \text{nodes}(R), \text{IN} \in R, \text{IN} \rightarrow X, N \]

discriminator of \text{IN}X

\[ Y \subseteq \text{edges}(S), \text{IFT} \in R, \text{IFT} \rightarrow Y, F \text{ and } T \]

foreign key (N in R), FT discriminator of \text{IFT}Y

Undirected: enforce symmetry of F and T

Labeled: use X and Y
Syntax

create graphview

  nodes(I,N,C, . . .)

graph key (I, N)

graph object I

as SQL statement;

(a)

create graphview

  edges(I,F,T,D, . . .)

graph key (I, F, T)

graph object I

as SQL statement;

(b)
Creating graphs

create graph $G$

notable nodes

edgetable edges

labels $nodes(C)$, $edges(D)$

direction $directed$

source $F$;

(c)

compute $A_1, \ldots, A_n$

from $G$ as $g$, $H$ as $h$, $G'$ as $g'$

where $\text{subisomorph}(g,h)$ without $(g,g')$;

(d)
compute *
from r as h
where exists (compute *
  from (compute *
    from r as t
    where exists (compute *
      from r as u
      where u.I = q and
      subisomorph(u,t))) as g
where subisomorph(g,h));

(b)
Operations allowed in perform

- match, isomorph, subisomorph, similar, \( k \)-similar and circuit
- using library clause in BioFlow/Curray to support analysis tools
- Question?
  - How to implement these operations?
  - Query optimization?
  - Selection, projection, join?
    - Selection conditions, partial constraints a la IsoKEGG (BIBM 2010)
isomorph, match

- For computing isomorph, check if the query graph and the data graph have identical “type” of descriptors – restrict unification to full structure
- For match, do not apply term replacement/mapping
  - Question, how do we allow partial unification to support partial isomorphism?
  - Separate into two groups – no term replacement in one set (BIBM 2010)
Definition 5.1 (Minimum Hub Cover) For a given graph $G = \langle V, E \rangle$, $C \subseteq V$ is a minimum hub cover of $G$, denoted $\mu^g$, if $C$ is the smallest set, and for every edge $< s, d > \in E$, either $d \in C$, or there exists edges $< s, c > \in E$ and $< d, c > \in E$, and $c \in C$. 
MHC of G

(a) Table

(b) Table

(c) Table

(d) Table

(e) Query graph q
Computational model

Similar to deep equality (Abiteboul and den Bussche, *DOOD* 1995).

\[ \nu_1 \leftarrow V \nu I(\pi_{I,N,C}(\text{nodes})) \]
\[ \nu_2 \leftarrow E \nu I(\pi_{I,F,T,D}(\text{edges})) \]
\[ G \leftarrow \pi_{v_1,I,V,E}(\nu_1 \Join_{v_1.I=v_2.I} \nu_2) \]
\[ s \leftarrow \pi_{v.G,v.E}(\sigma_{u.G\triangleleft v.G}(v \times u)) \]
\[ t \leftarrow \pi_{v.G,v.E}(\sigma_{s.G\triangleleft v.G}(v \times s)) \]
Search

(a) Graph \( g \)

(b) Assembling graph \( g \) from minimum parts

(c) Isomorphic assembling of graph \( g \) from minimum parts of another graph \( g' \)

(d) Injective mapping \( \mu \) of nodes in \( g \) and a data graph
Cliques of order 3 or less of n

(a) Triangle  (b) Star  (c) Composite  (d) Composite
Assembly process

(a) Data graph $D$ showing (b) Modified query graph matched subgraph isomorphs from graph in figure 2(a)

(c) Stored graphlets $a$, $j$ and $o$ corresponding to the data graph in figure 4(a)
IsoKEGG (BIBM 2010, IJDMB 2011)
Results
Performance result
(Data Graph size=320, query graph size=200)
Equivalence

Definition 2 (Equivalence of Graphs). Let $g$ and $h$ be two graphs, and $\Omega^g$ and $\Omega^h$ be the corresponding sets of minimum hub covers. Then, graph $g$ and $h$ are equal, denoted $g \equiv h$, if and only if $\exists \mu^g (\mu^g \in \Omega^g \land \mu^g \in \Omega^h)$, where $\mu^g$ is a MHC of $g$.

Definition 3 (Equivalence of Hub Representations). Let $g$ and $h$ be two graphs, and $r$ and $s$ be two hub representation of $g$ and $h$ in $g$-relations. If $g = \text{graph}(r)$ and $h = \text{graph}(s)$ are the graph representations derived from the $g$-relations, then $g \equiv h$, if and only if $\exists \mu^{\text{graph}(r)} (\mu^{\text{graph}(r)} \in \Omega^{\text{graph}(r)} \land \mu^{\text{graph}(r)} \in \Omega^{\text{graph}(s)})$. 
**Definition 4** (Containment of Graphlets). Let $\gamma^a$ and $\gamma^b$ be two graphlets of the form $< n, \varrho^n, \beta^n >$. Then $\gamma^a$ is a substructure of $\gamma^b$, denoted $\gamma^a \leq \gamma^b$, if $\varrho^a \subseteq \varrho^b \land \beta^a \subseteq \beta^b$.

**Theorem 1** (Containment of Graphs). Let $g$ and $h$ be two graphs, and let $\Omega^g$ and $\Omega^h$ be the corresponding sets of minimum hub covers. Then, graph $g$ is contained in graph $h$, denoted $g \subseteq h$, if and only if $\exists \mu^g, \mu^h (\mu^g \in \Omega^g \land \mu^h \in \Omega^h \land \forall \gamma^a (\gamma^a \in \mu^g \Rightarrow (\exists \gamma^b \in \mu^h \land \gamma^a \leq \gamma^b)))$. 

**Containment**
**Theorem 2** (Graph Isomorphism). Let $g$ and $h$ be two graphs, and let $\Omega^g$ and $\Omega^h$ be the corresponding sets of minimum hub covers. Let $f$ be a bijective function between the nodes of $g$ and $h$, and let $f(g)$ denotes the graph $g$ after substitution of the mapping. Then, graph $g$ and $h$ are isomorphic to each other, denoted $g \cong h$, if and only if $f(g) \cong h$.

**Theorem 3** (Subgraph Isomorphism). Let $g$ and $h$ be two graphs, and let $\Omega^g$ and $\Omega^h$ be the corresponding sets of minimum hub covers. Let $f$ be an injective function between the nodes of $g$ and $h$, and let $f(g)$ denotes the graph $g$ after substitution of the mapping. Then, graph $g$ is subgraph isomorphic to $h$, denoted $g \subset h$, if and only if $f(g) \subset h$. 
Conclusions

- Graph querying is challenging
  - Cost versus expressiveness
- Lacking clear model
  - NyQL is a novel proposal
- Memory efficient computation
  - Size is not a factor
- First to allow structure querying and comparison declaratively
Acknowledgement

- NSF Grants
  - CNS 0521454 and IIS 0612203
- My students
  - Anupam, Shafkat, Aminul, Saikat, Zakia
Thank you

Questions