Graph Queries in the Embedded Finite Model Theory

Deniz YILMAZ
Leonid LIBKIN, Cristina SIRANGELO
Université Paris Cité IRIF

Description of the Problem

At present, graph databases have a unifying language: GQL. In April 2024, it became an ISO standard [5]. One may refer to the survey paper [3] as an introduction to GQL. Our overall goal is to study fragments of GQL in order to achieve a theoretical understanding of their expressive power, complexity, and connections with classical logics that we understand well.

Embedded Finite Model Theory (EFMT) involves studying how finite models interact with infinite structures and understanding the computational and logical properties that arise from these interactions. EFMT provides an appropriate setting to study the Constraint Databases. Collapse results (Benedikt , Libkin [2] and independently active-generic collapse Otto and Van den Bussche [8]) greatly improved our understanding of the expressive power of the first order queries.

There are arguably two big categories for graph query languages:

1. Taking graphs as relational data and using classical relational queries, for example SQL, (Data);
2. Querying navigational patterns between nodes, for example *regular path queries* $\mathcal{RPQ}$ (Topology).

One may find a detailed study of query languages combining “Data” and “Topology” in [7]. These logics can be studied in the framework of EFMT. This setting makes the combination of “Data” and “Topology” possible.

For starters, we fix our data model as edge labelled graphs in the EFMT setting and define logics combining topology and data. Typically these would be expected to capture:

- “there exists a path π_s^t from s to t with its label $\lambda(\pi_s^t) \in e$ for some regular expression e” where s, t are two nodes;
- “data on the nodes of the path π_s^t satisfy some logical conditions ”.

Defined logics are studied to understand their expressive power and complexity. Already existing "collapse results" are not sufficient for these more expressive logics. Therefore we are seeking analogous versions of the collapse results.

Embedded Finite Model Theory

Here we present a summary of the EFMT setting. For a detailed treatise, see [6, ch. 5].

Definition 1. Fix a vocabulary τ, let $\mathfrak{M}$ be a τ-structure and $\Sigma = \{R_0, \cdots, R_n\}$ be a finite relational vocabulary where arity of R_i is r_i. An *embedded finite Σ-structure into $\mathfrak{M}$* is

$$\mathfrak{A} = \langle A, R_0^{\mathfrak{A}}, \cdots, R_n^{\mathfrak{A}} \rangle$$

where $R_i^{\mathfrak{A}} \subseteq M^{r_i}$ (M stands for the domain of $\mathfrak{M}$) is a finite subset for all i and $A = \bigcup_{i \leq n} \{a \in M : a \text{ occurs in } R_i\}$.

A is called the *active domain* and denoted $\mathsf{adom}\,\mathfrak{A}$. We will denote the (τ, Σ)-structures as $\langle \mathfrak{M}, \mathfrak{A} \rangle$.

Remark 2. One can also define (τ, Σ)-structures equivalently by specifying an embedding: Let $\mathfrak{A}$ be a finite Σ-structure and $\delta : A \to M$ be an injection. Then $\langle \mathfrak{M}, \mathfrak{A}, \delta \rangle$ defines a (τ, Σ)-structure.

Syntax of the first order (τ, Σ)-formulae, denote $\mathcal{FO}(\tau, \Sigma)$, is $\mathcal{FO}(\tau \cup \Sigma)$ formulas with *active domain quantifiers* $\exists_{\mathsf{adom}}$ and $\forall_{\mathsf{adom}}$.
Semantics of $\mathcal{FO}(\tau, \Sigma)$ is defined in the usual way except the active domain quantification. The latter is defined as follows:

$$\langle \mathfrak{M}, \mathfrak{A} \rangle \models \exists_{\mathsf{adom}}\, x \varphi(x, \bar{a}) \text{ if and only if } \langle \mathfrak{M}, \mathfrak{A} \rangle \models \varphi(b, \bar{a}) \text{ for some } b \in A.$$

Collapse Results

A collapse result typically shows that a logic is equivalent in terms of expressive power to a simpler fragment of it. Natural-active collapse amounts to elimination of unbounded quantifiers. Generic collapses state order-generic queries can be expressed only using order relation from the underlying structure.
A summary of collapse results ([6, ch. 5.6.7]):

$$\begin{tikzcd}
\mathcal{FO}_{\mathsf{act}}(\mathfrak{M}, \Sigma) \arrow[r, equal, "\substack{\mathfrak{M} \text{ is o-minimal and has quantifier elimination} \\ \text{natural-active collapse}}"] \arrow[d, equal, "\substack{\mathfrak{M} \text{ is ordered} \\ \text{active-generic collapse}}"'] & \mathcal{FO}(\mathfrak{M}, \Sigma) \arrow[d, equal, "\substack{\mathfrak{M} \text{ has finite } \textsc{vc}\text{-dimension} \\ \text{natural-generic collapse}}"] \\
\mathcal{FO}_{\mathsf{act}}(<, \Sigma) \arrow[r, equal] & \mathcal{FO}(<, \Sigma)
\end{tikzcd}$$

Remark 3. Note that the collapse results hold if the underlying structure $\mathfrak{M}$ satisfies specific conditions. In particular $\mathrm{Th}(\mathfrak{M})$ being decidable does not guarantee admitting collapse. Yet this is far from being discouraging since the following structures admit collapse: Linear Arithmetic $\langle \mathbb{N}, 0, 1, +, \leq \rangle$, Real Ordered Group $\langle \mathbb{R}, 0, 1, +, \leq \rangle$ and Real Ordered Field $\langle \mathbb{R}, 0, 1, +, \cdot, \leq \rangle$.

An Example of a Graph Database as an EFM

Suppose we have a map of (some sector of) the world with airports marked on it and a list of all flights. We pick two airports, say s and t. Our query is the following:

> Can we find a point p on the map such that there is a flight (direct or not) from s to t such that each airport used is at distance less than r for some fixed r?

One can model this problem by using 2-dimensional embedded data graph on the real ordered field $\langle \mathbb{R}, 0, 1, +, \cdot, \leq \rangle$ by the following assignments:

- V: set of airports with $s, t \in V$;
- $E_a(v, w)$: a direct flight from v to w of a-airline;
- $\delta_0(v)$ is the x-axis coordinate and $\delta_1(v)$ is the y-axis coordinate of v.

Question. From the statement of the query above finding such p seems to require an unbounded quantification over $\mathbb{R}$. Is it necessarily the case?
Consider the map in the Figure I. Suppose that there is a way to travel from s to t following the path

$$\pi_s^t : s \to v_1 \to v_2 \to v_3 \to t.$$

One can eliminate the unbounded quantification by a simple geometric analysis: Figure II.

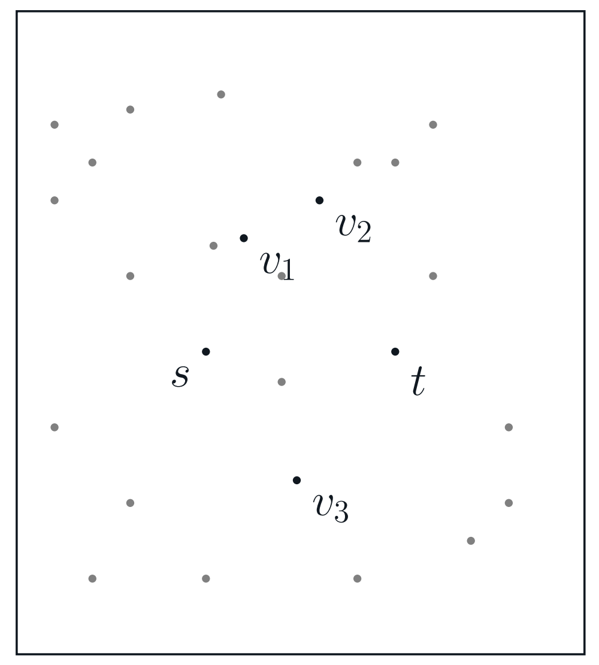

Figure I.

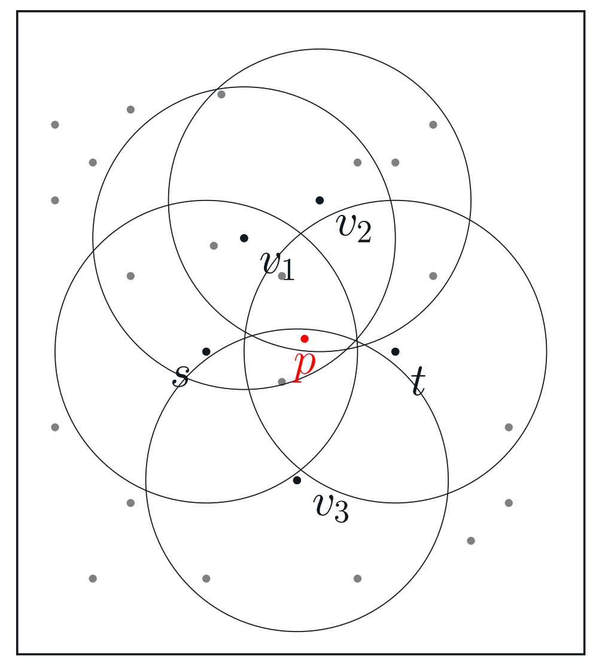

Figure II.

Data Model and Query Languages

Definition 4. An *n-dimensional embedded graph database* into $\mathfrak{M}$ is $\langle \mathfrak{G}, \mathfrak{M}, \delta_i : i \in n \rangle$ where:

- $\mathfrak{G}$ is a Σ-labelled graph;
- $\delta_i : V \to M$ is a mapping such that $\Delta : V \to M^n$ by $x \mapsto (\delta_i(x) : i \in n)$ is an injection.

For querying graphs $\mathcal{FO}$ is not adequate. For example connectivity is not $\mathcal{FO}$-definable. Therefore we need to consider more expressive logics. In our logics we utilise *regular path queries (RPQs)* for pattern matching, following the longstanding tradition (see [1] , [9]) and combine RPQs with first order conditions on data values (in the spirit of [7], [4]); using two different approaches.

1. Extending $\mathcal{FO}$ with $\mathcal{RPQ}(\Sigma)$

Syntax of the language $\mathbb{L}$ is defined recursively as follows:

1. $\varphi \in \mathbb{L}$ for all $\varphi \in \mathcal{FO}(\tau, \Sigma)$.
2. $e_{x,y}[\varphi(x, y, \bar{z})](s, t) \in \mathbb{L}$ for all $e \in RE(\Sigma)$, $\varphi(x, y, \bar{z}) \in \mathcal{FO}(\tau, \Sigma)$ and terms s and t.
3. $\mathbb{L}$ is closed under Boolean operators and the quantifiers $\exists, \forall$.

Semantics of the language $\mathbb{L}$ is defined as usual for (1) and (3). We will define satisfaction relation explicitly for (2): Let $\langle \mathfrak{G}, \mathfrak{M}, \delta_i : i \in n \rangle$ be a (τ, Σ)-structure and $a, b, \bar{c} \in M$.
$\langle \mathfrak{G}, \mathfrak{M}, \delta_i : i \in n \rangle \models e_{x,y}[\varphi(x, y, \bar{z})](s, t)$ iff there is a path π_s^t in G such that the label of the path $\lambda(\pi_s^t) \in e$ and for every edge $(u, v) \in E_a$ of π_s^t we have $(u, v) \in \{(x, y) : \langle \mathfrak{G}, \mathfrak{M} \rangle \models \varphi(x, y, \bar{c})\}$.

Example 5. The query in the example is $\mathbb{L}$-definable by the following formula:

$$\exists p_0 p_1 \Sigma^*_{x,y}[\big((\delta_0(x) - p_0)^2 + (\delta_1(x) - p_1)^2 < r^2\big) \wedge \big(\delta_0(y) - p_0)^2 + (\delta_1(y) - p_1)^2 < r^2\big)](s, t).$$

One can eliminate the unrestricted existential quantification as follows:

$$\Sigma^*_{x,y}[(\delta_0(x) - \delta_0(y))^2 + (\delta_1(x) - \delta_1(y))^2 < 4r^2](s, t).$$

2. Conditioning $\mathcal{RPQ}(\Sigma)$ with $\mathcal{FO}$ in a path sensible setting
To capture the path structure in the logic, we add a predicate P for path nodes, a countable index set $I \subseteq M$, and an index predicate index (realised $\subseteq I$) representing a path index. We include a symbol $<_P$ for ordering the indices and edge relation symbols E'_a for each $a \in \Sigma$ to represent path edges.

Syntax of the language $\mathbb{L}'$ is defined recursively as follows:

1. $e \in \mathbb{L}'$ for all $e \in RE$.
2. $e[\varphi(\bar{x})] \in \mathbb{L}'$ for all $\varphi(\bar{x}) \in \mathcal{FO}(\tau, \{E_a, E'_a, P, \mathsf{index}, <_P : a \in \Sigma\})$

Semantics of the language $\mathbb{L}'$ is defined recursively as follows:

1. $\langle \mathfrak{G}, \mathfrak{M}, \delta_i : i \in n \rangle \models e(s, t)$ iff there exists a path π_s^t.
2. $\langle \mathfrak{G}, \mathfrak{M}, \delta_i : i \in n \rangle \models_{[\bar{x}/\bar{a}]} e[\varphi(\bar{x})](s, t)$ iff there exists a path π_s^t satisfying:

$$\langle \mathfrak{G}, \mathfrak{M}, P^{\pi_s^t}, \mathsf{index}^{\pi_s^t}, {E'_a}^{\pi_s^t}, \delta_i : i \in n, a \in \Sigma \rangle \models \varphi(\bar{a}).$$

Example 6. The query in the example is $\mathbb{L}'$-definable by the following formula:

$$\Sigma^*_{x,y}[\exists p_0 p_1 \forall x P(x) \to \big((\delta_0(x) - p_0)^2 + (\delta_1(x) - p_1)^2 < r^2\big)](s, t).$$

One can eliminate the unrestricted existential quantification as follows:

$$\Sigma^*_{x,y}[\forall xy P(x) \wedge P(y) \to \big((\delta_0(x) - \delta_0(y))^2 + (\delta_1(x) - \delta_1(y))^2 < 4r^2\big)](s, t).$$

References: