Fine-Grained Complexity of Regular Path Queries

Katrin Casel[1], **Markus L. Schmid**[2]

[1] HPI, University of Potsdam, Germany
[2] HU Berlin, Germany

ICDT 2021

Graph Databases and Regular Path Queries

Querying Graphs with Regular Expressions

Graph databases

directed, edge-labelled multigraphs.

Querying Graphs with Regular Expressions

Graph databases

directed, edge-labelled multigraphs.

Graph Databases

Σ	finite alphabet (edge labels)
$V_{\mathcal{D}}$	*vertices* (or *nodes*)
$E_{\mathcal{D}} \subseteq V_{\mathcal{D}} \times \Sigma \times V_{\mathcal{D}}$	*edges* (or *arcs*)
Graph database	$\mathcal{D} = (V_{\mathcal{D}}, E_{\mathcal{D}})$

Querying Graphs with Regular Expressions

Graph databases

directed, edge-labelled multigraphs.

Graph Databases

Σ	finite alphabet (edge labels)
$V_{\mathcal{D}}$	*vertices* (or *nodes*)
$E_{\mathcal{D}} \subseteq V_{\mathcal{D}} \times \Sigma \times V_{\mathcal{D}}$	*edges* (or *arcs*)
Graph database	$\mathcal{D} = (V_{\mathcal{D}}, E_{\mathcal{D}})$

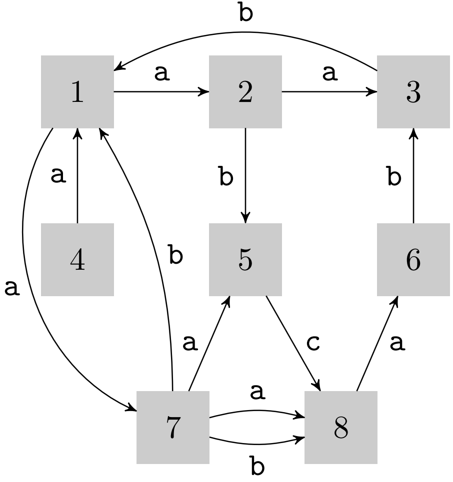
Regular Path Queries (RPQs)

Regular expressions q over Σ.

$q(\mathcal{D}) = \{(u, v) \mid \exists$ u-to-v path labelled by a word from $\mathcal{L}(q)\}$

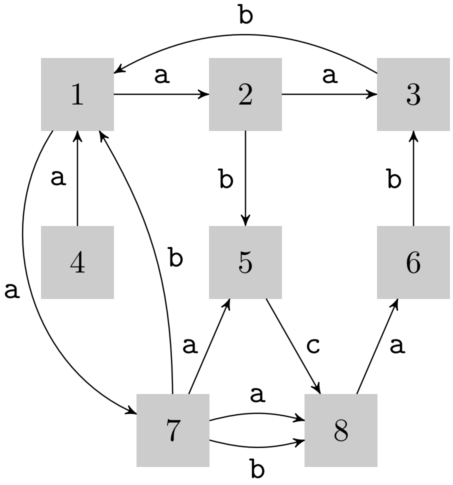
Regular Path Query Example

Graph database $\mathcal{D}$:



Regular Path Query Example

Graph database $\mathcal{D}$:



Regular path query:

$q = \texttt{a}^*(\texttt{b} \vee \texttt{c})$

Different Variants of RPQs

Different Variants of RPQs

Query results:

- Only node pairs (u, v).
- Node pairs (u, v) and a witness path.
- Node pairs (u, v) and *all* witness paths.

Different Variants of RPQs

Query results:

- Only node pairs (u, v).
- Node pairs (u, v) and a witness path.
- Node pairs (u, v) and *all* witness paths.

Path semantics: $(u, v) \in q(\mathcal{D})$ if there is

- an *arbitrary* path.
- a *simple* path.
- a *trail*.
- a *shortest* path.

Product Graph Approach (PG-Approach)

$\mathcal{D}$:	Graph database
q:	Regular path query
M:	NFA for q with state set Q

Product Graph Approach (PG-Approach)

$\mathcal{D}$:	Graph database
q:	Regular path query
M:	NFA for q with state set Q

Product Graph

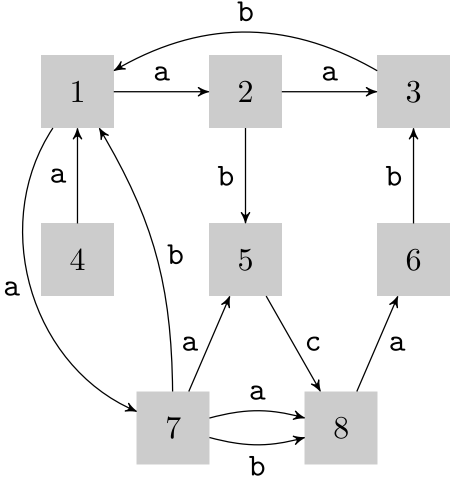
$G(\mathcal{D}, q) = (V(\mathcal{D}, q), E(\mathcal{D}, q))$

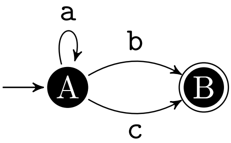
$V(\mathcal{D}, q) = V_{\mathcal{D}} \times Q$

$E(\mathcal{D}, q) \subseteq (V(\mathcal{D}, q) \times V(\mathcal{D}, q))$:

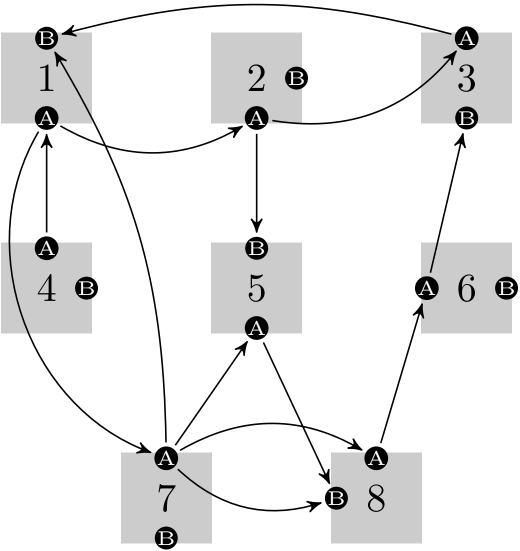
$$(u, p) \rightarrow (v, p') \iff \exists x \in \Sigma : u \xrightarrow{x} v \wedge p \xrightarrow{x} p' .$$

PG-Approach Example





PG-Approach Example



RPQ Evaluation Tasks

Name	Input	Task
RPQ-Boole	$\mathcal{D}$, q	Decide whether $q(\mathcal{D}) = \emptyset$.
RPQ-Eval	$\mathcal{D}$, q	Compute the whole set $q(\mathcal{D})$.
RPQ-Count	$\mathcal{D}$, q	Compute $\|q(\mathcal{D})\|$.
(Sorted) RPQ-Enum	$\mathcal{D}$, q	Enumerate the whole set $q(\mathcal{D})$ (lexicographically ordered).

RPQ Evaluation Tasks

Name	Input	Task
RPQ-Boole	$\mathcal{D}$, q	Decide whether $q(\mathcal{D}) = \emptyset$.
RPQ-Eval	$\mathcal{D}$, q	Compute the whole set $q(\mathcal{D})$.
RPQ-Count	$\mathcal{D}$, q	Compute $\lvert q(\mathcal{D}) \rvert$.
(Sorted) RPQ-Enum	$\mathcal{D}$, q	Enumerate the whole set $q(\mathcal{D})$ (lexicographically ordered).

Updates: Adding/deleting isolated nodes, adding/deleting arcs.

Research Question

- PG-approach good for simple tasks like checking $q(\mathcal{D}) = \emptyset$ or $(u, v) \in q(\mathcal{D})$.
  What about computing, counting or enumerating $q(\mathcal{D})$?
- Is the PG-approach optimal?
- Can we complement upper bounds with conditional lower bounds?

Fine-Grained Complexity and Conditional Lower Bounds

Orthogonal Vectors

Orthogonal Vectors (OV)

Input: Sets A, B each containing n Boolean d-dimensional vectors.
Question: Are there orthogonal vectors $\vec{a} \in A$ and $\vec{b} \in B$?

Orthogonal Vectors

Orthogonal Vectors (OV)

Input: Sets A, B each containing n Boolean d-dimensional vectors.
Question: Are there orthogonal vectors $\vec{a} \in A$ and $\vec{b} \in B$?

OV-Hypothesis

For every $\epsilon > 0$, OV cannot be solved in $O(n^{2-\epsilon} \text{poly}(d))$.

Boolean Matrix Multiplication

Boolean Matrix Multiplication (BMM)

Input: Boolean $n \times n$ matrices A, B.
Task: Compute $A \times B$.

Boolean Matrix Multiplication

Boolean Matrix Multiplication (BMM)

Input: Boolean $n \times n$ matrices A, B.
Task: Compute $A \times B$.

com-BMM-Hypothesis

For every $\epsilon > 0$, BMM cannot be solved in $O(n^{3-\epsilon})$ by a *combinatorial* algorithm.

Boolean Matrix Multiplication

Boolean Matrix Multiplication (BMM)

Input: Boolean $n \times n$ matrices A, B.
Task: Compute $A \times B$.

com-BMM-Hypothesis

For every $\epsilon > 0$, BMM cannot be solved in $O(n^{3-\epsilon})$ by a *combinatorial* algorithm.

SBMM-Hypothesis

BMM cannot be solved in $O(m)$, where $m =$ number of 1-entries.

Our Results

RPQ-Boole

Theorem

RPQ-Boole can be solved in time $O(|\mathcal{D}||q|)$.

RPQ-Boole

Theorem

RPQ-Boole can be solved in time $O(|\mathcal{D}||q|)$.

Theorem

If RPQ-Boole can be solved in time

- $O(|\mathcal{D}|^{2-\epsilon} + |q|^2)$, then OV-hypothesis fails.
- $O(|\mathcal{D}|^2 + |q|^{2-\epsilon})$, then OV-hypothesis fails.
- $O(|V_{\mathcal{D}}|^{3-\epsilon} + |q|^{3-\epsilon})$, com-BMM-hypothesis fails.

RPQ-Boole

Theorem

RPQ-Boole can be solved in time $O(|\mathcal{D}||q|)$.

Theorem

If RPQ-Boole can be solved in time

- $O(|\mathcal{D}|^{2-\epsilon} + |q|^2)$, then OV-hypothesis fails.
- $O(|\mathcal{D}|^2 + |q|^{2-\epsilon})$, then OV-hypothesis fails.
- $O(|V_{\mathcal{D}}|^{3-\epsilon} + |q|^{3-\epsilon})$, com-BMM-hypothesis fails.

Data Complexity

From now on ALL bounds in data complexity!

RPQ-Eval and RPQ-Count

Theorem

RPQ-Eval (and RPQ-Count) can be solved in time $O(|V_{\mathcal{D}}||\mathcal{D}|)$.

RPQ-Eval and RPQ-Count

Theorem

RPQ-Eval (and RPQ-Count) can be solved in time $O(|V_{\mathcal{D}}||\mathcal{D}|)$.

Theorem

If RPQ-Eval can be solved in time

- $O((|V_{\mathcal{D}}||\mathcal{D}|)^{1-\epsilon})$, then com-BMM-hypothesis fails.
- $O((|q(\mathcal{D})| + |\mathcal{D}|))$, then SBMM-hypothesis fails.

RPQ-Eval and RPQ-Count

Theorem

RPQ-Eval (and RPQ-Count) can be solved in time $O(|V_{\mathcal{D}}||\mathcal{D}|)$.

Theorem

If RPQ-Eval can be solved in time

- $O((|V_{\mathcal{D}}||\mathcal{D}|)^{1-\epsilon})$, then com-BMM-hypothesis fails.
- $O((|q(\mathcal{D})| + |\mathcal{D}|))$, then SBMM-hypothesis fails.

Theorem

If RPQ-Count can be solved in time $O((|V_{\mathcal{D}}||\mathcal{D}|)^{1-\epsilon})$ then the OV-hypothesis fails.

RPQ-Enum – Upper Bound

Theorem

Sorted RPQ-Enum can be solved with preprocessing $O(|\mathcal{D}|)$, delay $O(|\mathcal{D}|)$ and $O(1)$ updates.

RPQ-Enum – Upper Bound

Theorem

Sorted RPQ-Enum can be solved with preprocessing $O(|\mathcal{D}|)$, delay $O(|\mathcal{D}|)$ and $O(1)$ updates.

Some Thoughts

- Linear preprocessing is reasonable.
- Linear delay is bad.
- What about updates??

RPQ-Enum – Lower Bounds

Conditional Lower Bounds

Linear preprocessing and

- constant delay? No!

RPQ-Enum – Lower Bounds

Conditional Lower Bounds

Linear preprocessing and

- constant delay? No!
- delay sublinear in $|V_{\mathcal{D}}|$? No!

RPQ-Enum – Lower Bounds

Conditional Lower Bounds

Linear preprocessing and

- constant delay? No!
- delay sublinear in $|V_{\mathcal{D}}|$? No!
- delay sublinear in $|\mathcal{D}|$? Not if we also want updates!

RPQ-Enum – Lower Bounds

Conditional Lower Bounds

Linear preprocessing and

- constant delay? No!
- delay sublinear in $|V_{\mathcal{D}}|$? No!
- delay sublinear in $|\mathcal{D}|$? Not if we also want updates!

Open Question

RPQ-Enum with $O(|\mathcal{D}|)$ preprocessing and $O(|V_{\mathcal{D}}|)$ delay???

RPQ-Enum – Lower Bounds

Conditional Lower Bounds

Linear preprocessing and

- constant delay? No!
- delay sublinear in $|V_{\mathcal{D}}|$? No!
- delay sublinear in $|\mathcal{D}|$? Not if we also want updates!

Open Question

RPQ-Enum with $O(|\mathcal{D}|)$ preprocessing and $O(|V_{\mathcal{D}}|)$ delay???

Next objective:

Just any enumeration that guarantees delay sublinear in $|\mathcal{D}|$.

Three Approaches to Sublinear Delay

First Approach: Representative Subset of Solution Set

A "representative" subset $A \subseteq q(\mathcal{D})$ can be enumerated with linear preprocessing and constant delay.

Three Approaches to Sublinear Delay

First Approach: Representative Subset of Solution Set

A "representative" subset $A \subseteq q(\mathcal{D})$ can be enumerated with linear preprocessing and constant delay.

$\overline{\Delta}(\mathcal{D})$ denotes the *average* degree of $\mathcal{D}$.

Second Approach: Super-Linear Preprocessing

Sorted RPQ-Enum can be solved with preprocessing $\mathrm{O}(\log(\overline{\Delta}(\mathcal{D}))\ \overline{\Delta}(\mathcal{D})\,|\,\mathcal{D}\,|)$ and delay $\mathrm{O}(|V_{\mathcal{D}}|)$.

Three Approaches to Sublinear Delay

First Approach: Representative Subset of Solution Set

A "representative" subset $A \subseteq q(\mathcal{D})$ can be enumerated with linear preprocessing and constant delay.

$\overline{\Delta}(\mathcal{D})$ denotes the *average* degree of $\mathcal{D}$.

Second Approach: Super-Linear Preprocessing

Sorted RPQ-Enum can be solved with preprocessing $O(\log(\overline{\Delta}(\mathcal{D}))\ \overline{\Delta}(\mathcal{D})\,|\,\mathcal{D}\,|)$ and delay $O(|V_{\mathcal{D}}|)$.

$\Delta(\mathcal{D})$ denotes the *maximum* degree of $\mathcal{D}$.

Third Approach: Restricted Class of RPQs

For a $Q \subseteq$ RPQ, RPQ-Enum can be solved with preprocessing $O(|\,\mathcal{D}\,|)$ and delay $O(\Delta(\mathcal{D}))$.

Third Approach: Restricted Class of RPQs

- Short RPQ (S-RPQ):
  $q = (x_1 \vee \ldots \vee x_k)$ or $q = (x_1 \vee \ldots \vee x_k)(y_1 \vee \ldots \vee y_{k'})$,
  where $x_1, \ldots, x_k, y_1, \ldots, y_{k'} \in \Sigma$.

  Example: $q = (\mathtt{a} \vee \mathtt{b})(\mathtt{a} \vee \mathtt{c} \vee \mathtt{d})$.

Third Approach: Restricted Class of RPQs

- Short RPQ (S-RPQ):

  $q = (x_1 \vee \ldots \vee x_k)$ or $q = (x_1 \vee \ldots \vee x_k)(y_1 \vee \ldots \vee y_{k'})$,

  where $x_1, \ldots, x_k, y_1, \ldots, y_{k'} \in \Sigma$.

  Example: $q = (\mathtt{a} \vee \mathtt{b})(\mathtt{a} \vee \mathtt{c} \vee \mathtt{d})$.

- Basic Transitive RPQ (BT-RPQ):

  $q = (x_1 \vee \ldots \vee x_k)^*$ or $q = (x_1 \vee \ldots \vee x_k)^+$,

  where $x_1, \ldots, x_k \in \Sigma$.

  Example: $q = (\mathtt{a} \vee \mathtt{c} \vee \mathtt{d})^+$.

Third Approach: Restricted Class of RPQs

- Short RPQ (S-RPQ):

  $q = (x_1 \vee \ldots \vee x_k)$ or $q = (x_1 \vee \ldots \vee x_k)(y_1 \vee \ldots \vee y_{k'})$,

  where $x_1, \ldots, x_k, y_1, \ldots, y_{k'} \in \Sigma$.

  Example: $q = (\mathtt{a} \vee \mathtt{b})(\mathtt{a} \vee \mathtt{c} \vee \mathtt{d})$.
- Basic Transitive RPQ (BT-RPQ):

  $q = (x_1 \vee \ldots \vee x_k)^*$ or $q = (x_1 \vee \ldots \vee x_k)^+$,

  where $x_1, \ldots, x_k \in \Sigma$.

  Example: $q = (\mathtt{a} \vee \mathtt{c} \vee \mathtt{d})^+$.
- Alternation Closure:

  $\bigvee(\text{S-RPQ} \cup \text{BT-RPQ}) =$
  $\{(q_1 \vee \ldots \vee q_m) \mid q_i \in \text{S-RPQ} \cup \text{BT-RPQ}, 1 \leq i \leq m\}$.

  Example: $q = (\mathtt{ab} \vee \mathtt{c}^* \vee \mathtt{b}(\mathtt{c} \vee \mathtt{d}) \vee (\mathtt{a} \vee \mathtt{b} \vee \mathtt{d})^+)$

Third Approach: Restricted Class of RPQs

Theorem

Semi-sorted Enum($\bigvee$(S-RPQ $\cup$ BT-RPQ)) can be solved with preprocessing $O(|\mathcal{D}|)$ and delay $O(\Delta(\mathcal{D}))$.

Third Approach: Restricted Class of RPQs

Theorem

Semi-sorted Enum($\bigvee$(S-RPQ $\cup$ BT-RPQ)) can be solved with preprocessing $O(|\mathcal{D}|)$ and delay $O(\Delta(\mathcal{D}))$.

Proof Sketch

- *Semi-sorted* Enum(S-RPQ) and Enum(BT-RPQ) can be solved with preprocessing $O(|\mathcal{D}|)$ and delay $O(\Delta(\mathcal{D}))$.

Third Approach: Restricted Class of RPQs

Theorem

Semi-sorted Enum($\bigvee$(S-RPQ $\cup$ BT-RPQ)) can be solved with preprocessing O($|\mathcal{D}|$) and delay O($\Delta(\mathcal{D})$).

Proof Sketch

- *Semi-sorted* Enum(S-RPQ) and Enum(BT-RPQ) can be solved with preprocessing O($|\mathcal{D}|$) and delay O($\Delta(\mathcal{D})$).
- For every $Q \subseteq$ RPQ: Semi-sorted Enum(Q) can be solved with linear preprocessing and some delay, then Enum($\bigvee(Q)$) can be solved with the same preprocessing and delay.

□

Third Approach: Restricted Class of RPQs

Theorem

Semi-sorted Enum($\bigvee$(S-RPQ $\cup$ BT-RPQ)) can be solved with preprocessing $O(|\mathcal{D}|)$ and delay $O(\Delta(\mathcal{D}))$.

Proof Sketch

- *Semi-sorted* Enum(S-RPQ) and Enum(BT-RPQ) can be solved with preprocessing $O(|\mathcal{D}|)$ and delay $O(\Delta(\mathcal{D}))$.
- For every $Q \subseteq$ RPQ: Semi-sorted Enum(Q) can be solved with linear preprocessing and some delay, then Enum($\bigvee(Q)$) can be solved with the same preprocessing and delay.

□

Theorem

If RPQ-Enum(S-RPQ) can be solved with preprocessing $O(|V_{\mathcal{D}}|^{3-\epsilon})$ and delay $O(|\Delta(\mathcal{D})|^{1-\epsilon})$, then the com-BMM-hypothesis fails.

Thank you very much for your attention.