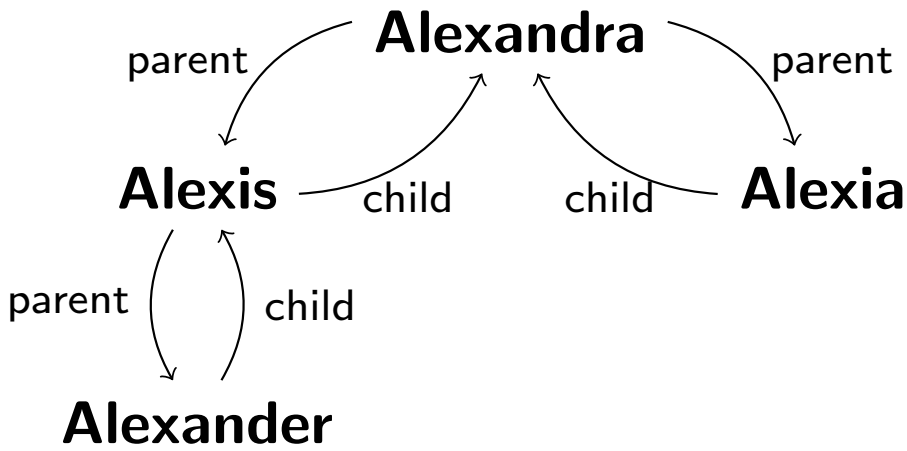
Conjunctive Context-Free Path Queries

Jelle Hellings
Hasselt University and transnational University of Limburg

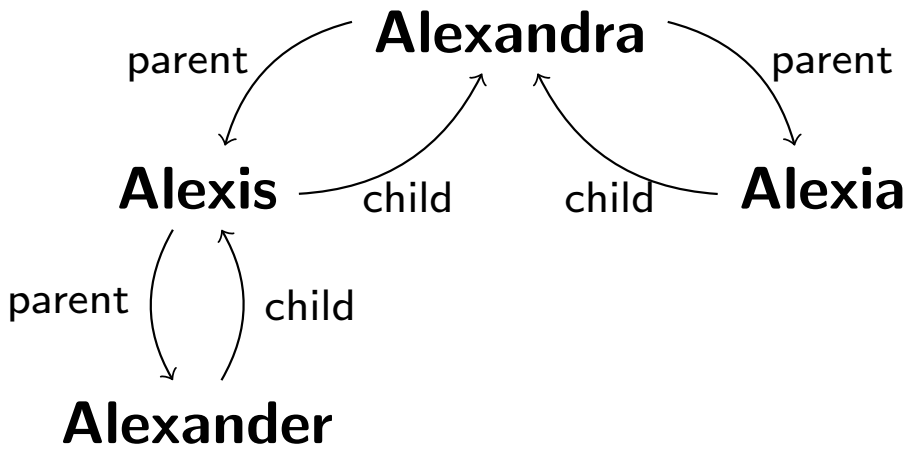
Motivation: graph data and path queries



Example (using CRPQ: conjunctive regular path queries)

- All pairs of relatives: $\mathrm{Q}(n, m) \leftarrow (\mathsf{parent} + \mathsf{child})^*(n, m)$
- All pairs (ancestor, descendants): $\mathrm{Q}(n, m) \leftarrow \mathsf{parent}^+(n, m)$
- All pairs of n-th generation descendants of an ancestor?

Motivation: limitations of CRPQ

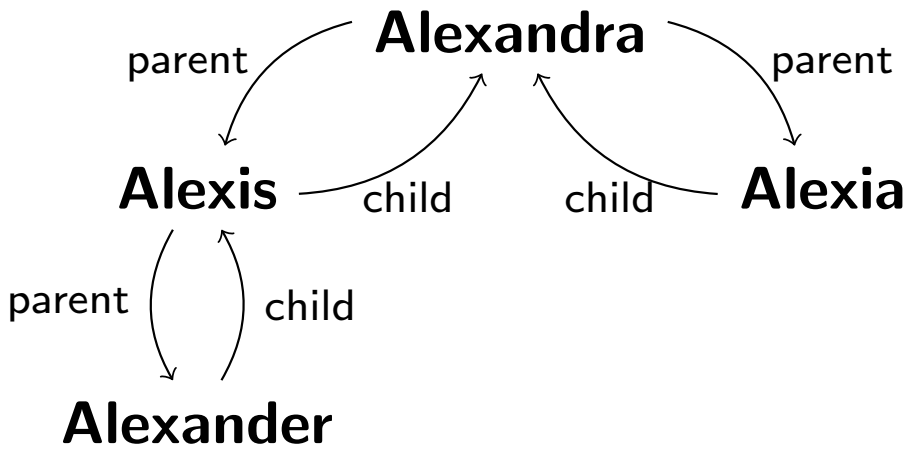


All pairs of n-th generation descendants of an ancestor?

Solution 1: path variables and regular relations [Barceló et al.]

$$\mathrm{Q}(n, m) \leftarrow \exists\pi\exists\rho\exists z \; n\pi z \wedge m\rho z \wedge \mathsf{child}^{+}(\pi) \wedge \mathsf{child}^{+}(\rho) \wedge |\pi| = |\rho|$$

Motivation: limitations of CRPQ



All pairs of n-th generation descendants of an ancestor?

Solution 1: path variables and regular relations [Barceló et al.]

$$\mathrm{Q}(n, m) \leftarrow \exists\pi\exists\rho\exists z\ n\pi z \wedge m\rho z \wedge \mathsf{child}^{+}(\pi) \wedge \mathsf{child}^{+}(\rho) \wedge |\pi| = |\rho|$$

Solution 2: context-free grammars

Production rules: $\mathtt{N} \to \mathsf{parent}\,\mathsf{child}, \quad \mathtt{N} \to \mathsf{parent}\,\mathtt{N}\,\mathsf{child}$

$$\mathrm{Q}(n, m) \leftarrow \mathtt{N}(n, m)$$

Overview

Context-free grammars

Definition (context-free grammar: $G = (N, A, P)$)

- N: a finite set of non-terminals
- A: a finite set of symbols (the alphabet)
- P: a finite set of production rules

Definition (language of context-free grammar G)

$$\mathcal{L}(G_{\mathtt{N}}) = \{s_1 \dots s_j \text{ a finite string over } A \mid \mathtt{N} \to_G s_1 \dots s_j\}$$

Definition (normal form)

All productions are of the form $\mathtt{A} \to \mathtt{B}\,\mathtt{C}$, $\mathtt{A} \to \sigma$, or $\mathtt{A} \to \lambda$

CCFPQ - syntax

Query $\mathrm{Q}(\vec{\nu}) \leftarrow \exists \vec{\mu} \ \bigwedge \mathtt{N}_i(n_i, m_i)$

- $\vec{\nu}$: a tuple of node variables
- $\vec{\mu}$: a tuple of distinct node variables that do not occur in $\vec{\nu}$
- $\mathtt{N}_i$: a non-terminal from a grammar
- n_i and m_i: are node variables taken from $\vec{\nu}$ or $\vec{\mu}$

CCFPQ - semantics

Query $\mathrm{Q}(\vec{\nu}) \leftarrow \exists \vec{\mu} \bigwedge \mathtt{N}_i(n_i, m_i)$

- The *trace* of path π is denoted by $\mathcal{T}(\pi)$
- Define $\mathcal{R}_{\mathtt{N}_i} \subseteq \mathcal{V} \times \mathcal{V}$ as:

$$\mathcal{R}_{\mathtt{N}_i} = \{(n, m) \mid \exists n \pi m \ (\mathcal{T}(\pi) \in \mathcal{L}(G_{\mathtt{N}_i}))\}$$

- Interpret atoms $\mathtt{N}_i(n_i, m_i)$ as relational atoms $\mathcal{R}_{\mathtt{N}_i}(n_i, m_i)$
- Threat result as normal conjunctive query

CCFPQ - query evaluation

Consider graph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ and query $\mathrm{Q}(\vec{\nu}) \leftarrow \exists \vec{\mu} \ \bigwedge \mathtt{N}_i(n_i, m_i)$

- $\mathcal{R}_{\mathtt{N}_i}$ is a binary relation
- Compute $\mathcal{R}_{\mathtt{N}_i}$: reduces CCFPQ to conjunctive queries
- We have $\mathcal{R}_{\mathtt{N}_i} \subseteq \mathcal{V} \times \mathcal{V}$

Context-free recognizer for graphs - algorithm

Input: Edge-labeled directed graph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$
Context-free grammar $G = (N, A, P)$
Output: $\{(\mathtt{N}, n, m) \mid (n, m) \in \mathcal{R}_{\mathtt{N}}\}$

1: $r := \{(\mathtt{N}, n, n) \mid (n \in \mathcal{V}) \wedge (\mathtt{N} \to \lambda \in P)\} \cup$
$\{(\mathtt{N}, n, m) \mid ((n, l, m) \in \mathcal{E}) \wedge (\mathtt{N} \to l \in P)\}$
2: $new := r$
3: **while** $new \neq \emptyset$ **do**
4: pick and remove a $(\mathtt{N}, n, m)$ from new
5: **for all** $(\mathtt{M}, n', n) \in r$ **do**
6: **for all** $(\mathtt{N'} \to \mathtt{M}\,\mathtt{N} \in P) \wedge ((\mathtt{N'}, n', m) \notin r)$ **do**
7: $new := new \cup \{(\mathtt{N'}, n', m)\}$
8: $r := r \cup \{(\mathtt{N'}, n', m)\}$
9: **for all** $(\mathtt{M}, m, m') \in r$ **do**
10: ...
11: **return** r

Context-free recognizer for graphs - analysis

Input: Edge-labeled directed graph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$
Context-free grammar $G = (N, A, P)$
Output: $\{(\mathtt{N}, n, m) \mid (n, m) \in \mathcal{R}_{\mathtt{N}}\}$

Proposition (correctness)

We have $(\mathtt{N}, n, m) \in r$ if and only if $(n, m) \in \mathcal{R}_{\mathtt{N}}$.

Theorem (worst-case complexity)

Running time is $\mathcal{O}(|N||\mathcal{E}| + (|N||\mathcal{V}|)^3)$.

Corollary

For $CCFPQ$ queries, the query evaluation problem has

1. *Polynomial time data complexity,*
2. *NP-complete combined complexity.*

Extending CCFPQ: add path variables

Theorem

Consider $\mathrm{Q}() \leftarrow \exists nm\ \exists n\pi m\ \mathtt{M}(\pi) \wedge \mathtt{M}(\pi)$, *we have:*

1. *For a fixed graph* $\mathcal{D}$, *query evaluation is undecidable*
2. *For a fixed query* Q, *query evaluation is undecidable*

Extending CCFPQ: CRPQ and ECRPQ

All pairs of n-th generation descendants of an ancestor?

Solution 1: path variables and regular relations [Barceló et al.]

$$\mathrm{Q}(n, m) \leftarrow \exists\pi\exists\rho\exists z\ n\pi z \wedge m\rho z \wedge \mathsf{child}^+(\pi) \wedge \mathsf{child}^+(\rho) \wedge |\pi| = |\rho|$$

How to express $|\pi| = |\rho|$?

$|\pi| = |\rho|$ as a regular expression over *tuples of edge labels*:

$$\left[\sum_{\sigma_1,\sigma_2 \in A} \langle \sigma_1, \sigma_2 \rangle\right]^* (\pi, \rho)$$

j-ary context-free grammar

Definition

A *j-ary context-free grammar* over alphabet A is a context-free grammar of the form $G = (N, (A \cup \varsigma)^j, P)$.

Example (subsequences)

'assert', 'liver' *subsequences of* 'Hasselt University'

$$\texttt{Sub} \to \lambda, \quad \texttt{Sub} \to \langle \sigma, \sigma \rangle \, \texttt{Sub} \ (\forall \sigma \in A),$$
$$\texttt{Sub} \to \langle \sigma, \varsigma \rangle \, \texttt{Sub} \ (\forall \sigma \in A).$$

ECCFPQ - syntax

Query $\mathrm{Q}(\vec{\nu}) \leftarrow \exists \vec{\mu}\ \bigwedge \mathtt{N}_i(\vec{p}_i)$

- $\vec{\nu}$: a tuple of node variables
- $\vec{\mu}$: a tuple of distinct node variables that do not occur in $\vec{\nu}$
- $\mathtt{N}_i$: a non-terminal from a j-ary grammar
- $\vec{p}_i$: a j-ary tuple of pairs of node variables taken from $\vec{\nu}$ or $\vec{\mu}$

Example (subsequences)

$$\mathtt{Sub} \to \lambda, \quad \mathtt{Sub} \to \langle \sigma, \sigma \rangle\, \mathtt{Sub}\ (\forall \sigma \in A),$$
$$\mathtt{Sub} \to \langle \sigma, \varsigma \rangle\, \mathtt{Sub}\ (\forall \sigma \in A).$$

$$\mathrm{Q}_{\mathtt{Sub}}(n_1, m_1, n_2, m_2) \leftarrow \mathtt{Sub}(\langle n_1, m_1 \rangle, \langle n_2, m_2 \rangle)$$

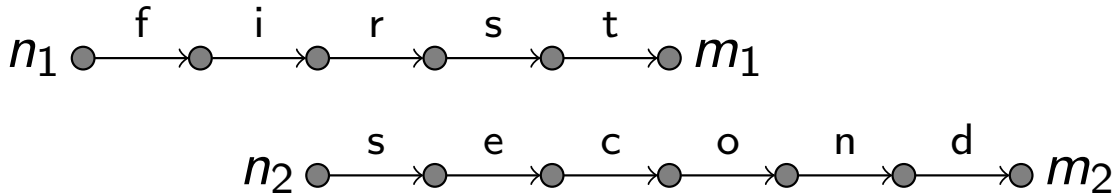
ECCFPQ - skip-traces

Definition

The *skip-traces* of path π, with $\mathcal{T}(\pi) = l_1 \dots l_i$, are defined as

$$\mathcal{T}_\varsigma(\pi) = \{s_1 + l_1 + s_2 + \dots + s_i + l_i + s_{i+1} \mid s_1, \dots, s_{i+1} \text{ are finite sequences of } \varsigma \text{ symbols}\}.$$

Example



$$\mathcal{T}_\varsigma(n_1 \pi_1 m_1) = \{\text{first}, \varsigma\text{first}, \text{f}\varsigma\text{irst}, \varsigma\text{f}\varsigma\text{irst}, \dots\}$$

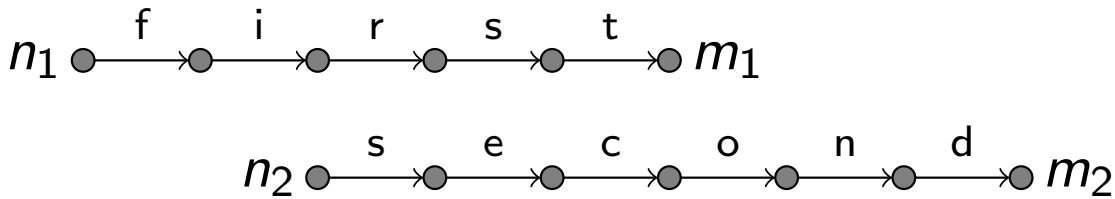
ECCFPQ - j-ary skip-traces

Definition

The *skip-traces* of a tuple of paths $\vec{\pi} = \langle \pi_1, \ldots, \pi_j \rangle$ are defined as

$$\mathcal{T}_{\varsigma j}(\vec{\pi}) = \{\langle t_1^1, \ldots, t_1^j \rangle \ldots \langle t_l^1, \ldots, t_l^j \rangle \mid (t^k = t_1^k \ldots t_l^k) \wedge (t^k \in \mathcal{T}_\varsigma(\pi_k))\}.$$

Example



$$\mathcal{T}_{\varsigma 2}(n_1\pi_1 m_1, n_2\pi_2 m_2) = \{\left\langle \begin{smallmatrix} \varsigma \\ s \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} f \\ e \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} i \\ c \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} r \\ o \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} s \\ n \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} t \\ d \end{smallmatrix} \right\rangle, \left\langle \begin{smallmatrix} \varsigma \\ s \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} f \\ \varsigma \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} i \\ e \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} r \\ c \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} s \\ o \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} t \\ n \end{smallmatrix} \right\rangle \left\langle \begin{smallmatrix} \varsigma \\ d \end{smallmatrix} \right\rangle, \ldots\}$$

ECCFPQ - semantics

Query $\mathrm{Q}(\vec{\nu}) \leftarrow \exists \vec{\mu} \ \bigwedge \mathtt{N}_i(n_i, m_i)$

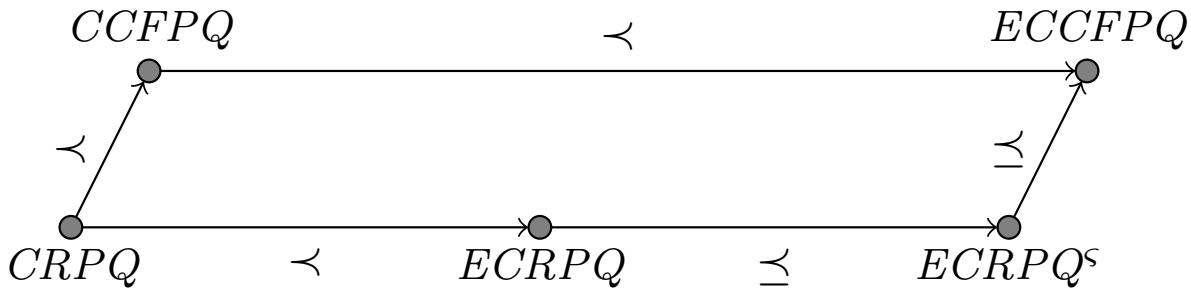
- Define $\mathcal{R}_{\mathtt{N}_i} \subseteq (\mathcal{V} \times \mathcal{V})^j$ as:

$$\mathcal{R}_{\mathtt{N}_i} = \{(\langle n_1, m_1 \rangle, \ldots, \langle n_j, m_j \rangle) \mid \exists n_1 \pi_1 m_1 \ldots \exists n_j \pi_j m_j \\ \exists t \ ((t \in \mathcal{T}_{\varsigma_j}(\langle \pi_1, \ldots, \pi_j \rangle)) \wedge (t \in \mathcal{L}(G_{\mathtt{N}_i})))\}$$

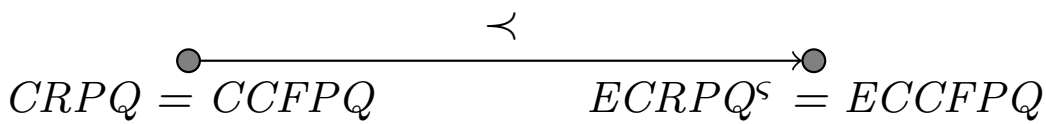
- Interpret atoms $\mathtt{N}_i(\vec{p_i})$ as relational atoms $\mathcal{R}_{\mathtt{N}_i}(\vec{p_i})$
- Threat result as normal conjunctive query

Expressive power of CCFPQ and variants

Theorem (on labeled graphs)



Theorem (on unlabeled graphs)



Extending CCFPQ: adding negation

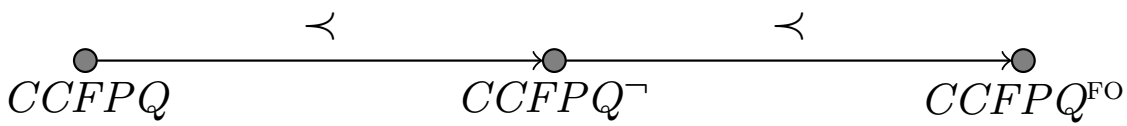
Example (all pairs of *distinct* n-th generation descendants?)

Production rules: $\mathtt{N} \to$ parent child, $\quad \mathtt{N} \to$ parent $\mathtt{N}$ child

$\mathtt{Equal} \to \lambda$

$$\mathrm{Q}(n, m) \leftarrow \mathtt{N}(n, m) \wedge \neg \mathtt{Equal}(n, m)$$

Theorem (on unlabeled graphs)



Conclusions and Future Work

Conclusions

- Introduced context-free grammars for specifying paths
- Efficient query evaluation techniques

Future Work

- Paths as answers instead of nodes
- Determine all relations between expressive power of variants