Exercise 11: Graph Databases and Path Queries

Database Theory

2020-07-06

Maximilian Marx, David Carral

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for “labelled directed graph” here – the technical details of the RDF standard are not important for this exercise.

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for "labelled directed graph" here – the technical details of the RDF standard are not important for this exercise.
Solution.

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for "labelled directed graph" here – the technical details of the RDF standard are not important for this exercise.
Solution.

- Let G be some labelled hypergraph.

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for "labelled directed graph" here – the technical details of the RDF standard are not important for this exercise.
Solution.

- Let G be some labelled hypergraph.
- We construct G_{RDF} by *reifying* hyperedges: for every p-labelled hyperedge $\varphi = p(t_1, t_2, \ldots, t_\ell)$ in G,

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for "labelled directed graph" here – the technical details of the RDF standard are not important for this exercise.
Solution.

- Let G be some labelled hypergraph.
- We construct G_{RDF} by *reifying* hyperedges: for every p-labelled hyperedge $\varphi = p(t_1, t_2, \ldots, t_\ell)$ in G,
- we add labels $p_1, p_2, \ldots, p_\ell$;

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for "labelled directed graph" here – the technical details of the RDF standard are not important for this exercise.
Solution.

- Let G be some labelled hypergraph.
- We construct G_{RDF} by *reifying* hyperedges: for every p-labelled hyperedge $\varphi = p(t_1, t_2, \ldots, t_\ell)$ in G,
- we add labels $p_1, p_2, \ldots, p_\ell$;
- a vertex v_φ; and

Exercise 1

Exercise. It was explained in the lecture that RDF and Property Graph can encode the same graph structures. How could we encode arbitrary hypergraphs (relational databases) in RDF? RDF can be considered as a synonym for "labelled directed graph" here – the technical details of the RDF standard are not important for this exercise.
Solution.

- Let G be some labelled hypergraph.
- We construct G_{RDF} by *reifying* hyperedges: for every p-labelled hyperedge $\varphi = p(t_1, t_2, \dots, t_\ell)$ in G,
- we add labels $p_1, p_2, \dots, p_\ell$;
- a vertex v_φ; and
- edges $p_1(c_\varphi, t_1), p_2(c_\varphi, t_2), \dots, p_\ell(c_\varphi, t_\ell)$ to G_{RDF}.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

1. The "Same generation" Datalog program from the lecture:

$$\mathrm{S}(x, x) \leftarrow \mathrm{human}(x)$$
$$\mathrm{S}(x, y) \leftarrow \mathrm{parent}(x, w) \wedge \mathrm{S}(v, w) \wedge \mathrm{parent}(y, v)$$

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

1. The "Same generation" Datalog program from the lecture:

$$\mathrm{S}(x, x) \leftarrow \mathrm{human}(x)$$
$$\mathrm{S}(x, y) \leftarrow \mathrm{parent}(x, w) \wedge \mathrm{S}(v, w) \wedge \mathrm{parent}(y, v)$$

Solution.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

1. The "Same generation" Datalog program from the lecture:

$$\mathrm{S}(x,x) \leftarrow \mathrm{human}(x)$$
$$\mathrm{S}(x,y) \leftarrow \mathrm{parent}(x,w) \wedge \mathrm{S}(v,w) \wedge \mathrm{parent}(y,v)$$

Solution.

1. ► S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

1. The "Same generation" Datalog program from the lecture:

$$\mathrm{S}(x,x) \leftarrow \mathrm{human}(x)$$
$$\mathrm{S}(x,y) \leftarrow \mathrm{parent}(x,w) \wedge \mathrm{S}(v,w) \wedge \mathrm{parent}(y,v)$$

Solution.

1.
   - S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

1. The “Same generation” Datalog program from the lecture:

$$\mathrm{S}(x, x) \leftarrow \mathrm{human}(x)$$
$$\mathrm{S}(x, y) \leftarrow \mathrm{parent}(x, w) \wedge \mathrm{S}(v, w) \wedge \mathrm{parent}(y, v)$$

Solution.

1. - S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

2. Ancestors born in the same city:

$$\begin{aligned} \mathrm{AncCity}(x, y, x', y') &\leftarrow \mathrm{parent}(x, x') \wedge \mathrm{bornIn}(x, y) \wedge \mathrm{bornIn}(x', y') \\ \mathrm{AncCity}(x, y, x'', y'') &\leftarrow \mathrm{AncCity}(x, y, x', y') \wedge \mathrm{AncCity}(x', y', x'', y'') \\ \mathrm{Query}(x, x', y) &\leftarrow \mathrm{AncCity}(x, y, x', y) \end{aligned}$$

Solution.

1.
   - S matches paths of the form $\mathrm{parent}^{n} \circ \mathrm{human} \circ \mathrm{parent}^{n}$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.

2.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

2. Ancestors born in the same city:

$$\begin{aligned}
\mathsf{AncCity}(x, y, x', y') &\leftarrow \mathsf{parent}(x, x') \wedge \mathsf{bornIn}(x, y) \wedge \mathsf{bornIn}(x', y') \\
\mathsf{AncCity}(x, y, x'', y'') &\leftarrow \mathsf{AncCity}(x, y, x', y') \wedge \mathsf{AncCity}(x', y', x'', y'') \\
\mathsf{Query}(x, x', y) &\leftarrow \mathsf{AncCity}(x, y, x', y)
\end{aligned}$$

Solution.

1. - S matches paths of the form $\mathsf{parent}^n \circ \mathsf{human} \circ \mathsf{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.
2. The following C2RPQ expresses Query:

$$(\mathsf{parent} \circ \mathsf{parent}^*)(x, x') \wedge \mathsf{bornIn}(x, y) \wedge \mathsf{bornIn}(x', y)$$

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

3. Ancestors of Dresden-based family lines:

$$\mathrm{DDAnc}(x,y) \leftarrow \mathrm{parent}(x,y) \wedge \mathrm{bornIn}(x,\mathrm{dresden}) \wedge \mathrm{bornIn}(y,\mathrm{dresden})$$
$$\mathrm{DDAnc}(x,z) \leftarrow \mathrm{DDAnc}(x,y) \wedge \mathrm{parent}(y,z) \wedge \mathrm{bornIn}(z,\mathrm{dresden})$$

Solution.

1.
   - S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.
2. The following C2RPQ expresses Query:

$$(\mathrm{parent} \circ \mathrm{parent}^*)(x,x') \wedge \mathrm{bornIn}(x,y) \wedge \mathrm{bornIn}(x',y)$$

3.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

3. Ancestors of Dresden-based family lines:

$$\mathrm{DDAnc}(x,y) \leftarrow \mathrm{parent}(x,y) \wedge \mathrm{bornIn}(x,\mathrm{dresden}) \wedge \mathrm{bornIn}(y,\mathrm{dresden})$$
$$\mathrm{DDAnc}(x,z) \leftarrow \mathrm{DDAnc}(x,y) \wedge \mathrm{parent}(y,z) \wedge \mathrm{bornIn}(z,\mathrm{dresden})$$

Solution.

1. - S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.
2. The following C2RPQ expresses Query:

$$(\mathrm{parent} \circ \mathrm{parent}^*)(x,x') \wedge \mathrm{bornIn}(x,y) \wedge \mathrm{bornIn}(x',y)$$

3. - DDAnc matches paths where every node has a bornIn-connection to dresden.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

3. Ancestors of Dresden-based family lines:

$$\mathrm{DDAnc}(x,y) \leftarrow \mathrm{parent}(x,y) \wedge \mathrm{bornIn}(x,\mathrm{dresden}) \wedge \mathrm{bornIn}(y,\mathrm{dresden})$$
$$\mathrm{DDAnc}(x,z) \leftarrow \mathrm{DDAnc}(x,y) \wedge \mathrm{parent}(y,z) \wedge \mathrm{bornIn}(z,\mathrm{dresden})$$

Solution.

1.
   - S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.
2. The following C2RPQ expresses Query:

$$(\mathrm{parent} \circ \mathrm{parent}^*)(x,x') \wedge \mathrm{bornIn}(x,y) \wedge \mathrm{bornIn}(x',y)$$

3.
   - DDAnc matches paths where every node has a bornIn-connection to dresden.
   - This is not expressible as a 2RPQ, since $(\mathrm{bornIn} \circ \mathrm{bornIn}^{-1})(x,y)$ will generally be true for $x \neq y$.

Exercise 2.

Exercise. Can the following Datalog programs be encoded using a C2RPQ? In each case, give a suitable C2RPQ or explain why there is none.

3. Ancestors of Dresden-based family lines:

$$\mathrm{DDAnc}(x,y) \leftarrow \mathrm{parent}(x,y) \wedge \mathrm{bornIn}(x,\mathrm{dresden}) \wedge \mathrm{bornIn}(y,\mathrm{dresden})$$
$$\mathrm{DDAnc}(x,z) \leftarrow \mathrm{DDAnc}(x,y) \wedge \mathrm{parent}(y,z) \wedge \mathrm{bornIn}(z,\mathrm{dresden})$$

Solution.

1. - S matches paths of the form $\mathrm{parent}^n \circ \mathrm{human} \circ \mathrm{parent}^n$, with $n \geq 0$.
   - This is not a regular language, and hence cannot be expressed as a 2RPQ.
   - Since the length of a matched path is not accessible in a C2RPQ, this can also not be expressed as a C2RPQ.
2. The following C2RPQ expresses Query:

$$(\mathrm{parent} \circ \mathrm{parent}^*)(x,x') \wedge \mathrm{bornIn}(x,y) \wedge \mathrm{bornIn}(x',y)$$

3. - DDAnc matches paths where every node has a bornIn-connection to dresden.
   - This is not expressible as a 2RPQ, since $(\mathrm{bornIn} \circ \mathrm{bornIn}^{-1})(x,y)$ will generally be true for $x \neq y$.
   - Since the intermediate nodes on a matched path are not accessible in a C2RPQ, this is also not expressible as a C2RPQ.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?
Solution.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide "Containment for RPQs" in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton "on the fly"?

Solution.

- Let E, E' be regular expressions.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

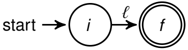
- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$:

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$:
  - if $E = \ell \in L$ is a label, then $\mathcal{N}$ is the following NFA:

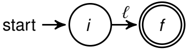


Exercise 3.

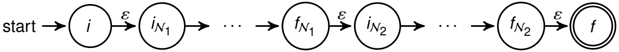
Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$:
  - if $E = \ell \in L$ is a label, then $\mathcal{N}$ is the following NFA:



  - if $E = (E_1 \circ E_2)$, and $\mathcal{N}_1 and \mathcal{N}_2$ are NFAs deciding E_1 and E_2, then $\mathcal{N}$ is the following NFA:

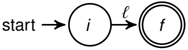


Exercise 3.

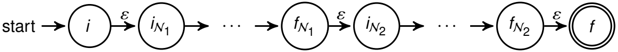
Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

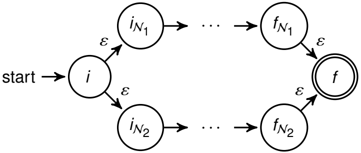
- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$:
  - if $E = \ell \in L$ is a label, then $\mathcal{N}$ is the following NFA:



  - if $E = (E_1 \circ E_2)$, and $\mathcal{N}_1 \mathit{and} \mathcal{N}_2$ are NFAs deciding E_1 and E_2, then $\mathcal{N}$ is the following NFA:



  - if $E = (E_1 + E_2)$, and $\mathcal{N}_1 \mathit{and} \mathcal{N}_2$ are NFAs deciding E_1 and E_2, then $\mathcal{N}$ is the following NFA:

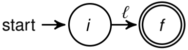


Exercise 3.

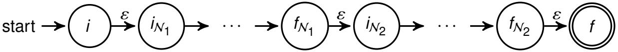
Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

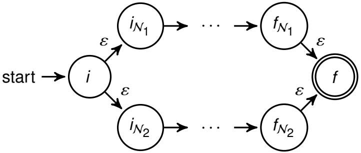
- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$:
  - if $E = \ell \in L$ is a label, then $\mathcal{N}$ is the following NFA:



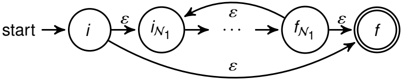
  - if $E = (E_1 \circ E_2)$, and $\mathcal{N}_1 and \mathcal{N}_2$ are NFAs deciding E_1 and E_2, then $\mathcal{N}$ is the following NFA:



  - if $E = (E_1 + E_2)$, and $\mathcal{N}_1 and \mathcal{N}_2$ are NFAs deciding E_1 and E_2, then $\mathcal{N}$ is the following NFA:



  - If $E = E_1^*$ and $\mathcal{N}_1$ is an NFA deciding E_1, then $\mathcal{N}$ is the following NFA:



Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide "Containment for RPQs" in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton "on the fly"?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide "Containment for RPQs" in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton "on the fly"?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.
- Construct the (polynomially large) product automaton $\hat{\mathcal{D}}$ of $\mathcal{D}$ and $\overline{\mathcal{D}'}$; then $\hat{\mathcal{D}}$ decides $E \cap \overline{E'}$.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide "Containment for RPQs" in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton "on the fly"?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.
- Construct the (polynomially large) product automaton $\hat{\mathcal{D}}$ of $\mathcal{D}$ and $\overline{\mathcal{D}'}$; then $\hat{\mathcal{D}}$ decides $E \cap \overline{E'}$.
- $E \sqsubseteq E'$ iff $\mathcal{L}(\hat{\mathcal{D}})$ is empty: if there is $w \in \mathcal{L}(\hat{\mathcal{D}})$, then $w \in \mathcal{L}(E)$ but $w \notin \mathcal{L}(E')$.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide "Containment for RPQs" in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton "on the fly"?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.
- Construct the (polynomially large) product automaton $\hat{\mathcal{D}}$ of $\mathcal{D}$ and $\overline{\mathcal{D}'}$; then $\hat{\mathcal{D}}$ decides $E \cap \overline{E'}$.
- $E \sqsubseteq E'$ iff $\mathcal{L}(\hat{\mathcal{D}})$ is empty: if there is $w \in \mathcal{L}(\hat{\mathcal{D}})$, then $w \in \mathcal{L}(E)$ but $w \notin \mathcal{L}(E')$.
- $\mathcal{L}(\hat{\mathcal{D}})$ is empty iff the final state is not reachable from the initial state.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.
- Construct the (polynomially large) product automaton $\hat{\mathcal{D}}$ of $\mathcal{D}$ and $\overline{\mathcal{D}'}$; then $\hat{\mathcal{D}}$ decides $E \cap \overline{E'}$.
- $E \sqsubseteq E'$ iff $\mathcal{L}(\hat{\mathcal{D}})$ is empty: if there is $w \in \mathcal{L}(\hat{\mathcal{D}})$, then $w \in \mathcal{L}(E)$ but $w \notin \mathcal{L}(E')$.
- $\mathcal{L}(\hat{\mathcal{D}})$ is empty iff the final state is not reachable from the initial state.
- Reachability on directed graphs can be checked in nondeterministic logarithmic space.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide "Containment for RPQs" in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton "on the fly"?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.
- Construct the (polynomially large) product automaton $\hat{\mathcal{D}}$ of $\mathcal{D}$ and $\overline{\mathcal{D}'}$; then $\hat{\mathcal{D}}$ decides $E \cap \overline{E'}$.
- $E \sqsubseteq E'$ iff $\mathcal{L}(\hat{\mathcal{D}})$ is empty: if there is $w \in \mathcal{L}(\hat{\mathcal{D}})$, then $w \in \mathcal{L}(E)$ but $w \notin \mathcal{L}(E')$.
- $\mathcal{L}(\hat{\mathcal{D}})$ is empty iff the final state is not reachable from the initial state.
- Reachability on directed graphs can be checked in nondeterministic logarithmic space.
- Since the state graph of $\hat{\mathcal{D}}$ is exponentially large, we can decide emptiness in nondeterministic polynomial space.

Exercise 3.

Exercise. Consider the method for checking RPQ containment as sketched on slide “Containment for RPQs” in the lecture. Explain the procedure and the resulting complexity bounds in your own words. How could one construct the required automaton “on the fly”?

Solution.

- Let E, E' be regular expressions.
- Construct NFAs $\mathcal{N}$ and $\mathcal{N}'$ deciding $\mathcal{L}(E)$ and $\mathcal{L}(E')$.
- Use the powerset construction to obtain equivalent (but exponentially large) DFAs $\mathcal{D}$ and $\mathcal{D}'$.
- Let $\overline{\mathcal{D}'}$ be the DFA obtained from $\mathcal{D}'$ by making all *accepting* states *reject*, and vice versa. Then $w \in \overline{\mathcal{D}'}$ iff $w \notin \mathcal{D}'$.
- Construct the (polynomially large) product automaton $\hat{\mathcal{D}}$ of $\mathcal{D}$ and $\overline{\mathcal{D}'}$; then $\hat{\mathcal{D}}$ decides $E \cap \overline{E'}$.
- $E \sqsubseteq E'$ iff $\mathcal{L}(\hat{\mathcal{D}})$ is empty: if there is $w \in \mathcal{L}(\hat{\mathcal{D}})$, then $w \in \mathcal{L}(E)$ but $w \notin \mathcal{L}(E')$.
- $\mathcal{L}(\hat{\mathcal{D}})$ is empty iff the final state is not reachable from the initial state.
- Reachability on directed graphs can be checked in nondeterministic logarithmic space.
- Since the state graph of $\hat{\mathcal{D}}$ is exponentially large, we can decide emptiness in nondeterministic polynomial space.
- Because of Savitch’s Theorem, we can thus decide containment in $\textsc{PSpace}$.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\, \mathrm{R}_1(x_1, x_2) \wedge \mathrm{R}_2(x_2, x_3) \wedge \cdots \wedge \mathrm{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathrm{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\, \mathsf{R}_1(x_1, x_2) \wedge \mathsf{R}_2(x_2, x_3) \wedge \cdots \wedge \mathsf{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathsf{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.
Solution.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \dots, x_{k_m}.\ \mathsf{R}_1(x_1, x_2) \wedge \mathsf{R}_2(x_2, x_3) \wedge \cdots \wedge \mathsf{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathsf{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.
Solution.

- Consider, e.g.,

$$\exists x.\ \mathsf{a}(x, x) \wedge \mathsf{b}(x, x),$$

which cannot be expressed as a 2RPQ.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\, \mathsf{R}_1(x_1, x_2) \wedge \mathsf{R}_2(x_2, x_3) \wedge \cdots \wedge \mathsf{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathsf{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.
Solution.

- Consider, e.g.,

  $$\exists x.\, \mathsf{a}(x, x) \wedge \mathsf{b}(x, x),$$

  which cannot be expressed as a 2RPQ.
- Note that this is not a linear C2RPQ.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\, \mathrm{R}_1(x_1, x_2) \wedge \mathrm{R}_2(x_2, x_3) \wedge \cdots \wedge \mathrm{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathrm{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.

Solution.

- Consider, e.g.,

$$\exists x.\, \mathrm{a}(x, x) \wedge \mathrm{b}(x, x),$$

  which cannot be expressed as a 2RPQ.
- Note that this is not a linear C2RPQ.
- Indeed, most linear binary C2RPQ can be expressed by a 2RPQ:

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\ \mathrm{R}_1(x_1, x_2) \wedge \mathrm{R}_2(x_2, x_3) \wedge \cdots \wedge \mathrm{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathrm{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.
Solution.

- Consider, e.g.,

  $$\exists x.\ \mathrm{a}(x, x) \wedge \mathrm{b}(x, x),$$

  which cannot be expressed as a 2RPQ.
- Note that this is not a linear C2RPQ.
- Indeed, most linear binary C2RPQ can be expressed by a 2RPQ:
- Every atom $p(x_i, x_{i+1})$ in the query can be viewed as an RPQ with label p.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\ \mathrm{R}_1(x_1, x_2) \wedge \mathrm{R}_2(x_2, x_3) \wedge \cdots \wedge \mathrm{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathrm{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.
Solution.

- Consider, e.g.,

  $$\exists x.\ \mathrm{a}(x, x) \wedge \mathrm{b}(x, x),$$

  which cannot be expressed as a 2RPQ.
- Note that this is not a linear C2RPQ.
- Indeed, most linear binary C2RPQ can be expressed by a 2RPQ:
- Every atom $p(x_i, x_{i+1})$ in the query can be viewed as an RPQ with label p.
- Since every 2RPQ in the query starts at the endpoint of the previous 2RPQ, the conjunctions can be replaced by composition.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\ \mathrm{R}_1(x_1, x_2) \wedge \mathrm{R}_2(x_2, x_3) \wedge \cdots \wedge \mathrm{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathrm{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.
Solution.

- Consider, e.g.,

  $$\exists x.\ \mathrm{a}(x, x) \wedge \mathrm{b}(x, x),$$

  which cannot be expressed as a 2RPQ.
- Note that this is not a linear C2RPQ.
- Indeed, most linear binary C2RPQ can be expressed by a 2RPQ:
- Every atom $p(x_i, x_{i+1})$ in the query can be viewed as an RPQ with label p.
- Since every 2RPQ in the query starts at the endpoint of the previous 2RPQ, the conjunctions can be replaced by composition.
- Thus, $\exists x_2, \ldots, x_{n-1}.\ (\mathrm{R}_1 \circ \mathrm{R}_2 \circ \cdots \circ \mathrm{R}_{n-1})(x_1, x_n)$ is an equivalent 2RPQ.

Exercise 4.

Exercise. Give an example for a binary C2RPQ that cannot be expressed as a 2RPQ.
By a *binary linear C2RPQ* we mean a C2RPQ of the form

$$\exists x_{k_1}, \ldots, x_{k_m}.\, \mathrm{R}_1(x_1, x_2) \land \mathrm{R}_2(x_2, x_3) \land \cdots \land \mathrm{R}_{n-1}(x_{n-1}, x_n)$$

where each $\mathrm{R}_i(x_i, x_{i+1})$ is an atom or a 2RPQ, and the x_{k_j} are among the variables that occur in the query. Can every linear binary C2RPQ be expressed by a 2RPQ? Explain your answer.

Solution.

- Consider, e.g.,

  $$\exists x.\, \mathrm{a}(x, x) \land \mathrm{b}(x, x),$$

  which cannot be expressed as a 2RPQ.
- Note that this is not a linear C2RPQ.
- Indeed, most linear binary C2RPQ can be expressed by a 2RPQ:
- Every atom $p(x_i, x_{i+1})$ in the query can be viewed as an RPQ with label p.
- Since every 2RPQ in the query starts at the endpoint of the previous 2RPQ, the conjunctions can be replaced by composition.
- Thus, $\exists x_2, \ldots, x_{n-1}.\, (\mathrm{R}_1 \circ \mathrm{R}_2 \circ \cdots \circ \mathrm{R}_{n-1})(x_1, x_n)$ is an equivalent 2RPQ.
- But in a 2RPQ, we lose access to $x_2, \ldots, x_{n-1}$.

Exercise 5.

Exercise. Give an example of a Datalog query that contains both of the following (and maybe also other) rules

$$\begin{aligned} \mathrm{Query}(x, z) &\leftarrow \mathrm{p}_a(x, y) \wedge \mathrm{p}_b(y, z) \\ \mathrm{Query}(x, z) &\leftarrow \mathrm{p}_a(x, x') \wedge \mathrm{Query}(x', z') \wedge \mathrm{p}_b(z', z) \end{aligned}$$

and that can be expressed as a C2RPQ.

Exercise 5.

Exercise. Give an example of a Datalog query that contains both of the following (and maybe also other) rules

$$\mathrm{Query}(x, z) \leftarrow \mathrm{p}_a(x, y) \wedge \mathrm{p}_b(y, z)$$
$$\mathrm{Query}(x, z) \leftarrow \mathrm{p}_a(x, x') \wedge \mathrm{Query}(x', z') \wedge \mathrm{p}_b(z', z)$$

and that can be expressed as a C2RPQ.
Solution.

Exercise 5.

Exercise. Give an example of a Datalog query that contains both of the following (and maybe also other) rules

$$\mathrm{Query}(x, z) \leftarrow \mathrm{p}_a(x, y) \wedge \mathrm{p}_b(y, z)$$
$$\mathrm{Query}(x, z) \leftarrow \mathrm{p}_a(x, x') \wedge \mathrm{Query}(x', z') \wedge \mathrm{p}_b(z', z)$$

and that can be expressed as a C2RPQ.

Solution.

- The query would match paths of the form $a^n b^n$ with $n \geq 0$, which is not a regular language.

Exercise 5.

Exercise. Give an example of a Datalog query that contains both of the following (and maybe also other) rules

$$\begin{aligned}\mathrm{Query}(x,z) &\leftarrow \mathrm{p}_a(x,y) \wedge \mathrm{p}_b(y,z)\\ \mathrm{Query}(x,z) &\leftarrow \mathrm{p}_a(x,x') \wedge \mathrm{Query}(x',z') \wedge \mathrm{p}_b(z',z)\end{aligned}$$

and that can be expressed as a C2RPQ.
Solution.

- The query would match paths of the form $a^n b^n$ with $n \geq 0$, which is not a regular language.
- We add rules so that all paths of the form $a^n b^m$ with $n, m \geq 0$ match, which is a regular language:

$$\begin{aligned}\mathrm{p}_{(a+b)^*}(x,y) &\leftarrow \mathrm{p}_a(x,y) & \mathrm{p}_{(a+b)^*}(x,y) &\leftarrow \mathrm{p}_b(x,y)\\ \mathrm{p}_{(a+b)^*}(x,y) &\leftarrow \mathrm{p}_{(a+b)^*}(x,z) \wedge \mathrm{p}_a(z,y) & \mathrm{p}_{(a+b)^*}(x,y) &\leftarrow \mathrm{p}_{(a+b)^*}(x,z) \wedge \mathrm{p}_b(z,y)\\ \mathrm{Query}(x,y) &\leftarrow \mathrm{p}_{(a+b)^*}(x,y)\end{aligned}$$

Exercise 5.

Exercise. Give an example of a Datalog query that contains both of the following (and maybe also other) rules

$$\mathrm{Query}(x,z) \leftarrow \mathrm{p}_a(x,y) \wedge \mathrm{p}_b(y,z)$$
$$\mathrm{Query}(x,z) \leftarrow \mathrm{p}_a(x,x') \wedge \mathrm{Query}(x',z') \wedge \mathrm{p}_b(z',z)$$

and that can be expressed as a C2RPQ.

Solution.

- The query would match paths of the form $a^n b^n$ with $n \geq 0$, which is not a regular language.
- We add rules so that all paths of the form $a^n b^m$ with $n, m \geq 0$ match, which is a regular language:

$$\mathrm{p}_{(a+b)^*}(x,y) \leftarrow \mathrm{p}_a(x,y) \qquad \mathrm{p}_{(a+b)^*}(x,y) \leftarrow \mathrm{p}_b(x,y)$$
$$\mathrm{p}_{(a+b)^*}(x,y) \leftarrow \mathrm{p}_{(a+b)^*}(x,z) \wedge \mathrm{p}_a(z,y) \qquad \mathrm{p}_{(a+b)^*}(x,y) \leftarrow \mathrm{p}_{(a+b)^*}(x,z) \wedge \mathrm{p}_b(z,y)$$
$$\mathrm{Query}(x,y) \leftarrow \mathrm{p}_{(a+b)^*}(x,y)$$

- The resulting program is equivalent to the C2RPQ

$$(a+b)^*(x,y)$$