Updated 2/29/2022

Topic 2: Complexity of Query Evaluation
Unit 2: Beyond Conjunctive Queries
Lecture 13

Wolfgang Gatterbauer

CS7240 Principles of scalable data management (sp22)

https://northeastern-datalab.github.io/cs7240/sp22/

2/29/2022

Outline: T2-1/2: Query Evaluation & Query Equivalence

- T2-1: Conjunctive Queries (CQs)
  - CQ equivalence and containment
  - Graph homomorphisms
  - Homomorphism beyond graphs
  - CQ containment
  - CQ minimization
- T2-2: Equivalence Beyond CQs
  - Union of CQs, and inequalities
  - Union of CQs equivalence under bag semantics
  - Tree pattern queries
  - Nested queries

Islands of Tractability of CQ Evaluation

- Major Research Program: Identify tractable cases of the combined complexity of conjunctive query evaluation.
- Over the years, this program has been pursued by two different research communities:
  - The Database Theory community
  - The Constraint Satisfaction community
- Explanation: Problems in those community are closely related:

Constraint Satisfaction Problem $\equiv$ Homomorphism Problem $\equiv$ CQ evaluation

[Feder, Vardi 1993] [Chandra, Merlin 1977]

[Kolaitis, Vardi 2000]

Feder, Vardi: Monotone monadic SNP and constraint satisfaction, STOC 1993 https://doi.org/10.1145/167088.167245 / Kolaitis, Vardi: Conjunctive-Query Containment and Constraint Satisfaction, JCSS 2000 https://doi.org/10.1006/jcss.2000.1713 / Chandra, Merlin. "Optimal implementation of conjunctive queries in relational data bases", STOC 1977. https://doi.org/10.1145/800105.803397
Based on Phokion Kolaitis' "Logic and Databases" series at Simons Institute, 2016. https://simons.berkeley.edu/talks/logic-and-databases

Beyond Conjunctive Queries

- What can we say about query languages of intermediate expressive power between conjunctive queries and the full relational calculus?

- Conjunctive queries form the sublanguage of relational algebra obtained by using only cartesian product, projection, and selection with equality conditions.

- The next step would be to consider relational algebra expressions that also involve union.

Beyond Conjunctive Queries

- Definition:
  - A Union of Conjunctive Queries (UCQ) is a query expressible by an expression of the form $q_1 \cup q_2 \cup \ldots \cup q_m$, where each q_i is a conjunctive query.
  - A monotone query is a query expressible by a relational algebra expression which uses only union, cartesian product, projection, and selection with equality condition.
- Fact:
  - Monotone queries are precisely the queries expressible by relational calculus expressions using ∧, ∨, and ∃ only (also assuming restriction to equality here).
  - Every union of conjunctive queries is a monotone query.
  - Every monotone query is equivalent to a union of conjunctive queries
    - but this normal form may have exponentially many disjuncts

$(a+b+c)(d+e+f)(g+h+j) = \ldots$ *how big as sum of products* ?

Beyond Conjunctive Queries

- Definition:
  - A Union of Conjunctive Queries (UCQ) is a query expressible by an expression of the form $q_1 \cup q_2 \cup \ldots \cup q_m$, where each q_i is a conjunctive query.
  - A monotone query is a query expressible by a relational algebra expression which uses only union, cartesian product, projection, and selection with equality condition.
- Fact:
  - Monotone queries are precisely the queries expressible by relational calculus expressions using $\wedge$, $\vee$, and $\exists$ only.
  - Every union of conjunctive queries is a monotone query.
  - Every monotone query is equivalent to a union of conjunctive queries
    - but this normal form may have exponentially many disjuncts

$$(a+b+c)(d+e+f)(g+h+j) = adg + adh + adj + aeg + aeh + \ldots + cfj$$

27 products

Unions of CQs and Monotone Queries

Union of Conjunctive Queries (UCQ)

Given edge relation $E(A,B)$, find paths of length 1 or 2

RA ? (unnamed RA)

DRC ?

Unions of CQs and Monotone Queries

Union of Conjunctive Queries (UCQ)

Given edge relation $E(A,B)$, find paths of length 1 or 2

RA $\quad E \cup \pi_{\$1,\$4}(\sigma_{\$2=\$3}(E \times E))$ (unnamed RA)

DRC ?

Unions of CQs and Monotone Queries

Union of Conjunctive Queries (UCQ)

Given edge relation *E*(*A*,*B*), find paths of length 1 or 2

RA $E \cup \pi_{\$1,\$4}(\sigma_{\$2=\$3}(E \times E))$

DRC $\{(x,y) | E(x,y) \vee \exists z[E(x,z) \wedge E(z,y)]\}$

Unions of CQs and Monotone Queries

Union of Conjunctive Queries (UCQ)

Given edge relation $E(A,B)$, find paths of length 1 or 2

RA $E \cup \pi_{\$1,\$4}(\sigma_{\$2=\$3}(E \times E))$

DRC $\{(x,y) | E(x,y) \lor \exists z[E(x,z) \land E(z,y)]\}$

Monotone Query

Assume schema R(A,B), S(A,B), T(B,C), V(B,C)

Is following query monotone ? $(R \cup S) \bowtie (T \cup V)$

Unions of CQs and Monotone Queries

Union of Conjunctive Queries (UCQ)

Given edge relation *E*(*A*,*B*), find paths of length 1 or 2

RA $E \cup \pi_{\$1,\$4}(\sigma_{\$2=\$3}(E \times E))$

DRC $\{(x,y) | E(x,y) \lor \exists z[E(x,z) \land E(z,y)]\}$

Monotone Query

Assume schema R(A,B), S(A,B), T(B,C), V(B,C)

Following query is monotone: $(R \cup S) \bowtie (T \cup V)$

Equal to a UCQ? ?

Unions of CQs and Monotone Queries

Union of Conjunctive Queries (UCQ)

Given edge relation *E*(*A*,*B*), find paths of length 1 or 2

RA $E \cup \pi_{\$1,\$4}(\sigma_{\$2=\$3}(E \times E))$

DRC $\{(x,y) | E(x,y) \lor \exists z[E(x,z) \land E(z,y)]\}$

Monotone Query

Assume schema R(A,B), S(A,B), T(B,C), V(B,C)

Following query is monotone: $(R \cup S) \bowtie (T \cup V)$

Equal to following UCQ: $(R \bowtie T) \cup (R \bowtie V) \cup (S \bowtie T) \cup (S \bowtie V)$

The Containment Problem for Unions of CQs

THEOREM [Sagiv, Yannakakis 1980]

Let $q_1 \cup q_2 \cup \cdots \cup q_m$ *and* $q'_1 \cup q'_2 \cup \cdots \cup q'_n$ *be two UCQs.*
Then the following are equivalent:

1) $q_1 \cup q_2 \cup \cdots \cup q_m \subseteq q'_1 \cup q'_2 \cup \cdots \cup q'_n$

2) For every $i \leq m$, there is $j \leq n$ such that $q_i \subseteq q'_j$

Proof:

2. ⇒ 1. This direction is obvious.

1. ⇒ 2. Since $D_C[q_i] \models q_i$, we have that $D_C[q_i] \models q_1 \cup q_2 \cup \ldots \cup q_m$.
Because of containment, $D_C[q_i] \models q'_1 \cup q'_2 \cup \ldots \cup q'_n$.
Thus there is some $j \leq n$ with $D_C[q_i] \models q'_j$.
Thus from the CQ homomorphism Theorem $q_i \subseteq q'_j$.

The Complexity of Database Query Languages

	Relational Calculus	CQs	UCQs
Query Eval.: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Eval.: Combined Compl.	PSPACE-complete	NP-complete	NP-complete
Query Equivalence & Containment	Undecidable	NP-complete	NP-complete

Source: Phokion Kolaitis' "Logic and Databases" series at Simons Institute, 2016. https://simons.berkeley.edu/talks/logic-and-databases

Monotone Queries

- Even though monotone queries have the same expressive power as unions of conjunctive queries, the containment problem for monotone queries has higher complexity than the containment problem for unions of conjunctive queries (syntax/complexity tradeoff)

- Theorem: Sagiv and Yannakakis – 1982

  The containment problem for monotone queries is Π_2^p-complete.

- Note: The prototypical Π_2^p-complete problem is $\forall\exists$-SAT, i.e., the restriction of QBF to formulas of the form

$$\forall x_1 \ldots \forall x_m \exists y_1 \ldots \exists y_n \phi.$$

Source: Phokion Kolaitis' "Logic and Databases" series at Simons Institute, 2016. https://simons.berkeley.edu/talks/logic-and-databases

The Complexity of Database Query Languages

	Relational Calculus	CQs	UCQs	Monotone queries
Query Eval.: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Eval.: Combined Compl.	PSPACE-complete	NP-complete	NP-complete	NP-complete
Query Equivalence & Containment	Undecidable	NP-complete	NP-complete	Π_2^p-complete

Source: Phokion Kolaitis' "Logic and Databases" series at Simons Institute, 2016. https://simons.berkeley.edu/talks/logic-and-databases

Conjunctive Queries with Inequalities

- Definition: Conjunctive queries with inequalities form the sublanguage of relational algebra obtained by using only cartesian product, projection, and selection with equality and inequality (≠, <, ≤) conditions.

- Example: Q(x,y):-- E(x,z), E(z,w),E(w,y), z ≠ w, z < y.

- Theorem: (Klug – 1988, van der Meyden – 1992)
  - The query containment problem for conjunctive queries with inequalities is Π_2^p-complete.
  - The query evaluation problem for conjunctive queries with inequalities in NP-complete.

Source: Phokion Kolaitis' "Logic and Databases" series at Simons Institute, 2016. https://simons.berkeley.edu/talks/logic-and-databases

The Complexity of Database Query Languages

	Relational Calculus	CQs	UCQs	Monotone queries / CQs with inequalities
Query Eval.: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Eval.: Combined Compl.	PSPACE-complete	NP-complete	NP-complete	NP-complete
Query Equivalence & Containment	Undecidable	NP-complete	NP-complete	Π_2^p-complete

Source: Phokion Kolaitis' "Logic and Databases" series at Simons Institute, 2016. https://simons.berkeley.edu/talks/logic-and-databases

Outline: T2-1/2: Query Evaluation & Query Equivalence

- T2-1: Conjunctive Queries (CQs)
  - CQ equivalence and containment
  - Graph homomorphisms
  - Homomorphism beyond graphs
  - CQ containment
  - CQ minimization
- T2-2: Equivalence Beyond CQs
  - Union of CQs, and inequalities
  - Union of CQs equivalence under bag semantics
  - Tree pattern queries
  - Nested queries

Following slides are literally from Phokion Kolaitis's talk on "Logic and databases" at "Logical structures in Computation Boot Camp", Berkeley 2016:
https://simons.berkeley.edu/talks/logic-and-databases

Logic and Databases

Phokion G. Kolaitis

UC Santa Cruz & IBM Research – Almaden

Lecture 4 – Part 1

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Thematic Roadmap

- ✓ Logic and Database Query Languages
  - Relational Algebra and Relational Calculus
  - Conjunctive queries and their variants
  - Datalog
- ✓ Query Evaluation, Query Containment, Query Equivalence
  - Decidability and Complexity
- ✓ Other Aspects of Conjunctive Query Evaluation
- Alternative Semantics of Queries
  - Bag Databases: Semantics and Conjunctive Query Containment
  - Probabilistic Databases: Semantics and Dichotomy Theorems for Conjunctive Query Evaluation
  - Inconsistent Databases: Semantics and Dichotomy Theorems

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Alternative Semantics

- So far, we have examined logic and databases under classical semantics:
  - The database relations are sets.
  - Tarskian semantics are used to interpret queries definable be first-order formulas.
- Over the years, several different alternative semantics of queries have been investigated. We will discuss three such scenarios:
  - The database relations can be bags (multisets).
  - The databases may be probabilistic.
  - The databases may be inconsistent.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Sets vs. Multisets

Relation EMPLOYEE(name, dept, salary)

- Relational Algebra Expression:

  $\pi_{salary} (\sigma_{dept = CS} (EMPLOYEE))$

- SQL query:

```
SELECT   salary
FROM     EMPLOYEE
WHERE    dpt = 'CS'
```

- SQL returns a bag (multiset) of numbers in which a number may appear several times, provided different faculty had the same salary.
- SQL does not eliminate duplicates, in general, because:
  - Duplicates are important for aggregate queries (e.g., average)
  - Duplicate elimination takes nlogn time.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Relational Algebra Under Bag Semantics

Operation	Multiplicity
Union $R_1 \cup R_2$	$m_1 + m_2$
Intersection $R_1 \cap R_2$	$\min(m_1, m_2)$
Product $R_1 \times R_2$	$m_1 \times m_2$
Projection and Selection	Duplicates are not eliminated

- R_1

A	B
1	2
1	2
2	3

- R_2

B	C
2	4
2	5

- $(R_1 \bowtie R_2)$

A	B	C
1	2	4
1	2	4
1	2	5
1	2	5

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Conjunctive Queries Under Bag Semantics

Chaudhuri & Vardi – 1993

Optimization of ***Real*** Conjunctive Queries

- Called for a re-examination of conjunctive-query optimization under bag semantics.
- In particular, they initiated the study of the containment problem for conjunctive queries under bag semantics.
- This problem has turned out to be *much more challenging* than originally perceived.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

PROBLEMS

Problems worthy
of attack
prove their worth
by hitting back.

in: *Grooks* by Piet Hein (1905-1996)

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Query Containment Under Set Semantics

Class of Queries	**Complexity of Query Containment**
Conjunctive Queries	NP-complete Chandra & Merlin – 1977
Unions of Conjunctive Queries	NP-complete Sagiv & Yannakakis - 1980
Conjunctive Queries with $\neq$, $\leq$, $\geq$	Π_2^p-complete Klug 1988, van der Meyden -1992
First-Order (SQL) queries	Undecidable Trakhtenbrot - 1949

8

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Bag Semantics vs. Set Semantics

- For bags R_1, R_2:
  $R_1 \subseteq_{BAG} R_2$ if $m(\mathbf{a},R_1) \leq m(\mathbf{a},R_2)$, for every tuple $\mathbf{a}$.
- $Q^{BAG}(D)$: Result of evaluating Q on (bag) database D.
- $Q_1 \subseteq_{BAG} Q_2$ if for every (bag) database D, we have that

$$Q_1^{BAG}(D) \subseteq_{BAG} Q_2^{BAG}(D).$$

Fact:

- $Q_1 \subseteq_{BAG} Q_2$ implies $Q_1 \subseteq Q_2$.
- The converse does **not** always hold.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Bag Semantics vs. Set Semantics

Fact: $Q_1 \subseteq Q_2$ does not imply that $Q_1 \subseteq_{BAG} Q_2$.

Example:

- $Q_1(x)$:- $P(x), T(x)$
- $Q_2(x)$:- $P(x)$

- $Q_1 \subseteq Q_2$ (obvious from the definitions)
- $Q_1 \not\subseteq_{BAG} Q_2$
- Consider the (bag) instance $D = \{P(a), T(a), T(a)\}$. Then:
  - $Q_1(D) = \{a,a\}$
  - $Q_2(D) = \{a\}$, so $Q_1(D) \not\subseteq Q_2(D)$.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Query Containment under Bag Semantics

- Chaudhuri & Vardi - 1993 stated that:

  Under bag semantics, the containment problem for conjunctive queries is Π_2^p-hard.

- Problem:
  - What is the exact complexity of the containment problem for conjunctive queries under bag semantics?
  - Is this problem decidable?

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Query Containment Under Bag Semantics

- 23 years have passed since the containment problem for conjunctive queries under bag semantics was raised.
- Several attacks to solve this problem have failed.
- At least two technically flawed PhD theses on this problem have been produced.
- Chaudhuri and Vardi have withdrawn the claimed Π_2^p-hardness of this problem; no one has provided a proof.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Query Containment Under Bag Semantics

- The containment problem for conjunctive queries under bag semantics remains **open** to date.

- However, progress has been made towards the containment problem under bag semantics for the two main extensions of conjunctive queries:
  - Unions of conjunctive queries
  - Conjunctive queries with ≠

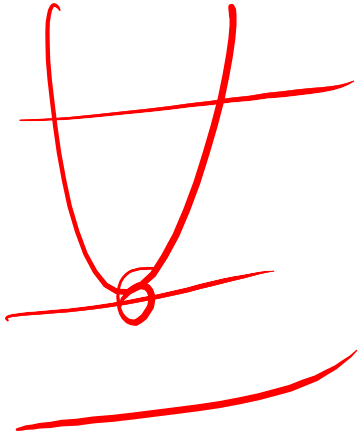
Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Unions of Conjunctive Queries

Theorem (Ioannidis & Ramakrishnan – 1995):
Under bag semantics, the containment problem for unions of conjunctive queries is **undecidable**.

Hint of Proof:
Reduction from Hilbert's 10^{th} Problem.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01



Hilbert's 10th Problem

- Hilbert's 10th Problem – 1900
  (10th in Hilbert's list of 23 problems)

  Given a Diophantine equation with any number of unknown quantities and with rational integral numerical coefficients: To devise a process according to which it can be determined in a finite number of operations whether the equation is solvable in rational integers.

  In effect, Hilbert's 10th Problem is:
  Find an algorithm for the following problem:
  Given a polynomial $P(x_1,\ldots,x_n)$ with integer coefficients, does it have an all-integer solution?

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Hilbert's 10th Problem

- Hilbert's 10th Problem – 1900

  (10th in Hilbert's list of 23 problems)

  Find an algorithm for the following problem:

  Given a polynomial $P(x_1,\ldots,x_n)$ with integer coefficients, does it have an all-integer solution?

- Y. Matiyasevich – 1971

  (building on M. Davis, H. Putnam, and J. Robinson)

  – Hilbert's 10th Problem is **undecidable**, hence **no** such algorithm exists.

16

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Hilbert's 10^{th} Problem

- Fact: The following variant of Hilbert's 10^{th} Problem is undecidable:
  - Given two polynomials $p_1(x_1,\ldots x_n)$ and $p_2(x_1,\ldots x_n)$ with positive integer coefficients and no constant terms, is it true that $p_1 \leq p_2$?

    In other words, is it true that $p_1(a_1,\ldots,a_n) \leq p_2(a_1,\ldots a_n)$, for all positive integers $a_1,\ldots,a_n$?

- Thus, there is no algorithm for deciding questions like:
  - Is $3x_1^4x_2x_3 + 2x_2x_3 \leq x_1^6 + 5x_2x_3$?

17

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Unions of Conjunctive Queries

Theorem (Ioannidis & Ramakrishnan – 1995):

Under bag semantics, the containment problem for unions of conjunctive queries is **undecidable**.

Hint of Proof:

- Reduction from the previous variant of Hilbert's 10th Problem:
  - Use joins of unary relations to encode monomials (products of variables).
  - Use unions to encode sums of monomials.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Unions of Conjunctive Queries

Example: Consider the polynomial $3x_1^4x_2x_3 + 2x_2x_3$

- The monomial $x_1^4x_2x_3$ is encoded by the conjunctive query $P_1(w),P_1(w),P_1(w), P_1(w), P_2(w),P_3(w)$.

- The monomial x_2x_3 is encoded by the conjunctive query $P_2(w),P_3(w)$.

- The polynomial $3x_1^4x_2x_3 + 2x_2x_3$ is encoded by the union having:
  - three copies of $P_1(w),P_1(w),P_1(w), P_1(w), P_2(w),P_3(w)$ and
  - two copies of $P_2(w),P_3(w)$.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Complexity of Query Containment

Class of Queries	Complexity – Set Semantics	Complexity – Bag Semantics
Conjunctive queries	NP-complete CM – 1977	
Unions of conj. queries	NP-complete SY - 1980	Undecidable IR - 1995
Conj. queries with $\neq$, $\leq$, $\geq$	Π_2^p-complete vdM - 1992	
First-order (SQL) queries	Undecidable Trakhtenbrot - 1949	Undecidable

20

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Conjunctive Queries with ≠

Theorem (Jayram, K …, Vee – 2006):
Under bag semantics, the containment problem for conjunctive queries with ≠ is **undecidable**.

In fact, this problem is **undecidable** even if

- the queries use only a single relation of arity 2;
- the number of inequalities in the queries is at most some fixed (albeit huge) constant.

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Complexity of Query Containment

Class of Queries	Complexity – Set Semantics	Complexity – Bag Semantics
Conjunctive queries	NP-complete CM – 1977	**Open**
Unions of conj. queries	NP-complete SY - 1980	Undecidable IR - 1995
Conj. queries with $\neq$, $\leq$, $\geq$	Π_2^p-complete vdM - 1992	Undecidable JKV - 2006
First-order (SQL) queries	Undecidable Trakhtenbrot - 1949	Undecidable

Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Subsequent Developments

- Some progress has been made towards identifying special classes of conjunctive queries for which the containment problem under bag semantics is decidable.

  - Afrati, Damigos, Gergatsoulis – 2010
    - Projection-free conjunctive queries.

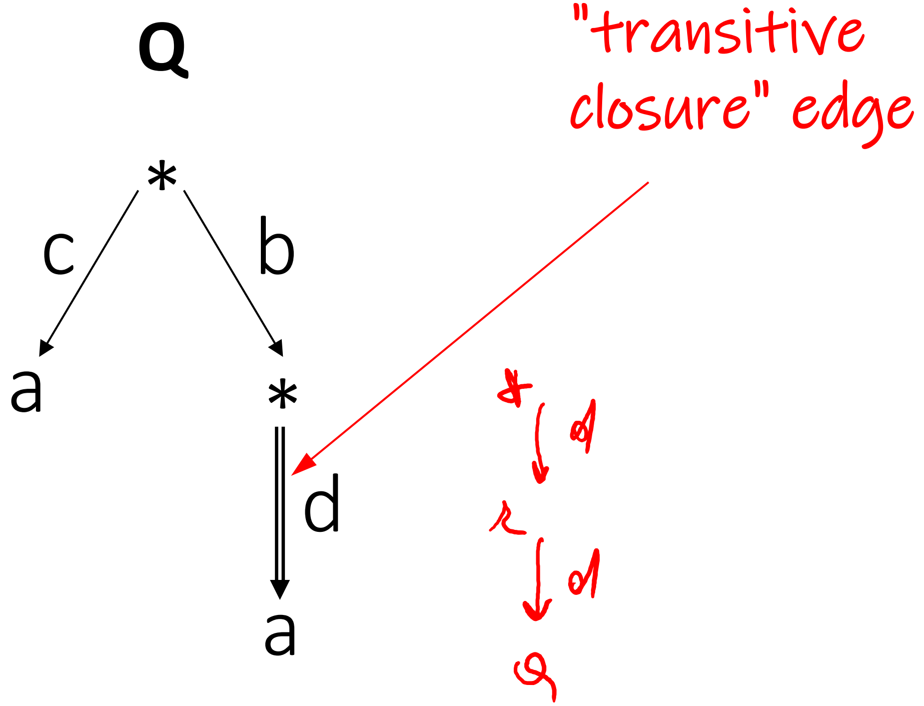
  - Kopparty and Rossman – 2011
    - A large class of boolean conjunctive queries on graphs.

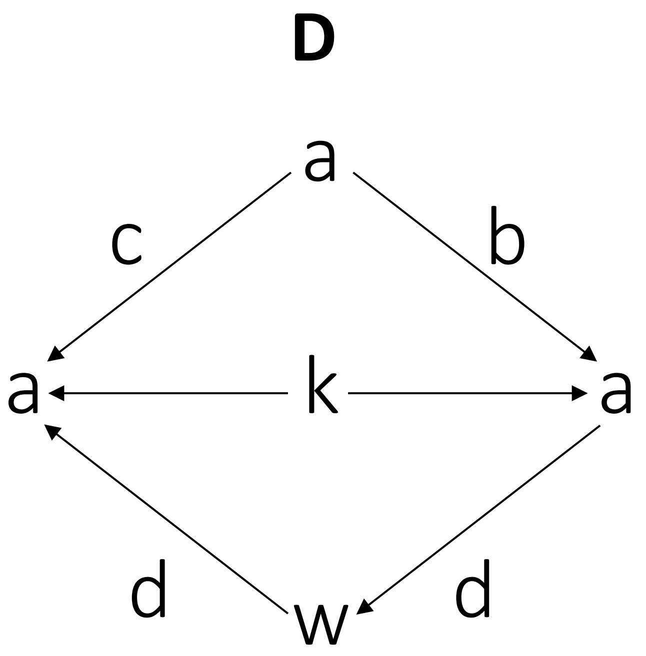
Source: Phokion Kolaitis: https://simons.berkeley.edu/talks/phokion-kolaitis-2016-09-01

Outline: T2-1/2: Query Evaluation & Query Equivalence

- T2-1: Conjunctive Queries (CQs)
  - CQ equivalence and containment
  - Graph homomorphisms
  - Homomorphism beyond graphs
  - CQ containment
  - CQ minimization
- T2-2: Equivalence Beyond CQs
  - Union of CQs, and inequalities
  - Union of CQs equivalence under bag semantics
  - Tree pattern queries
  - Nested queries

Tree pattern queries





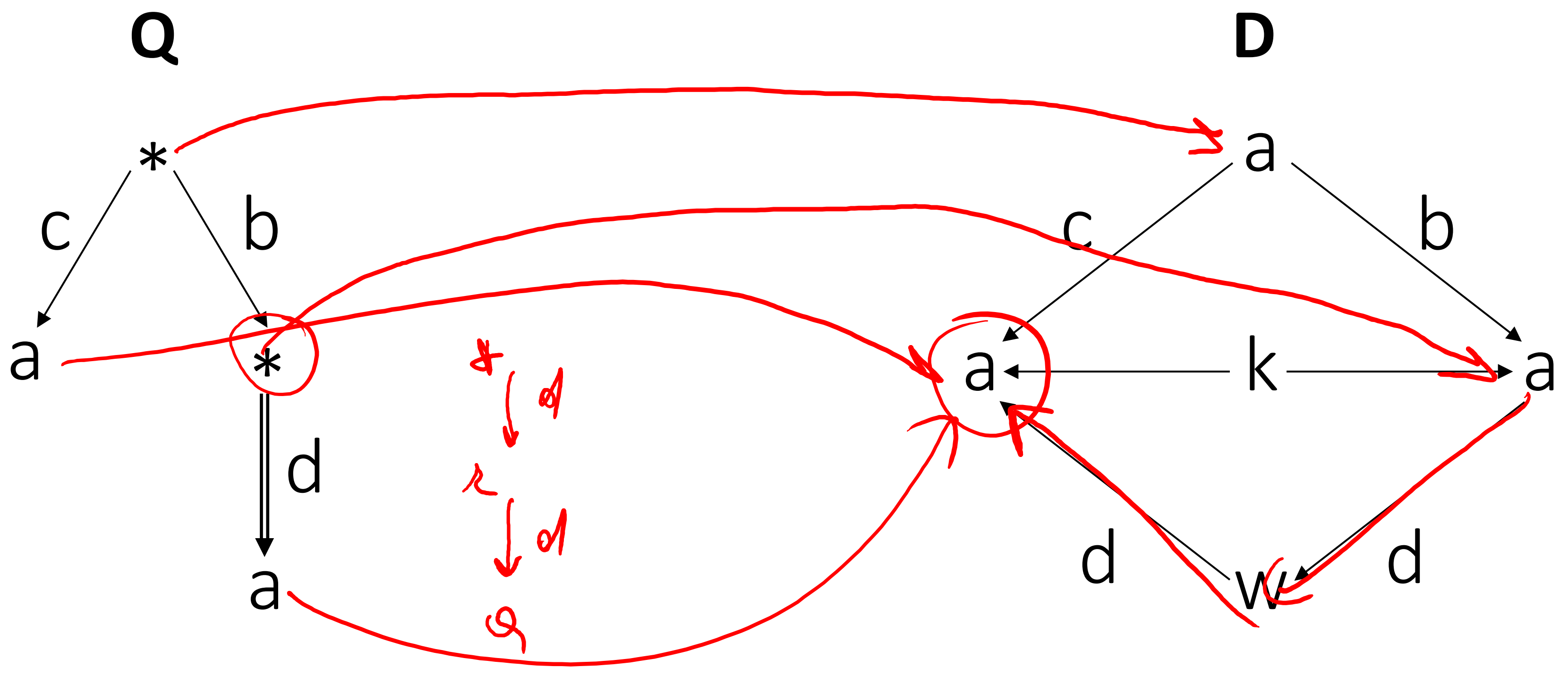
Does the query on the left have a match on in the data on the right (i.e. is there a homomorphism from left to right)?

?

Notice that "a", "b", "c" are labels (not node ids), thus like constants in a query, or like predicates (colored edges)

Example from: "Optimizing Tree Patterns for Querying Graph- and Tree-Structured Data" by Czerwinski, Martens, Niewerth, Parys. SIGMOD record 2017. https://doi.org/10.1145/3093754.3093759

Tree pattern queries



Example from: “Optimizing Tree Patterns for Querying Graph- and Tree-Structured Data” by Czerwinski, Martens, Niewerth, Parys. SIGMOD record 2017. https://doi.org/10.1145/3093754.3093759

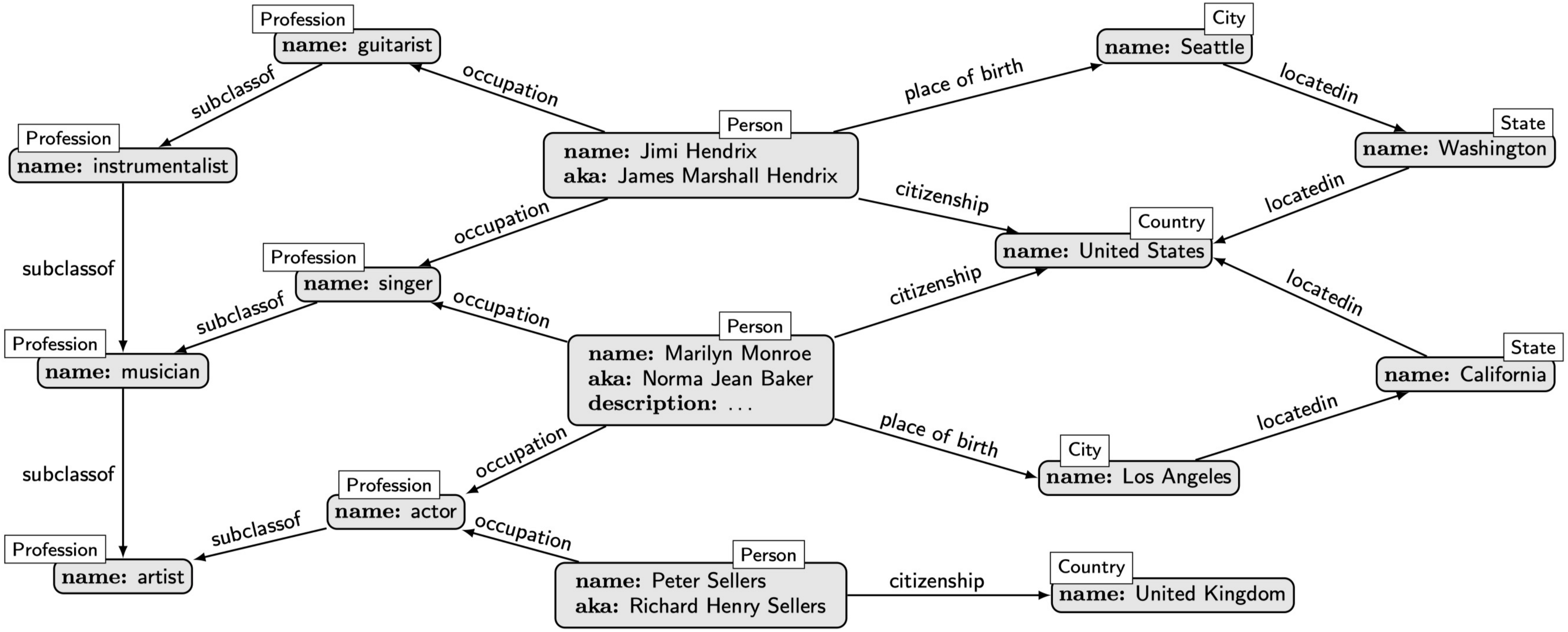


Figure 1: A graph database (as a *property graph*), inspired on a fragment of WikiData

?

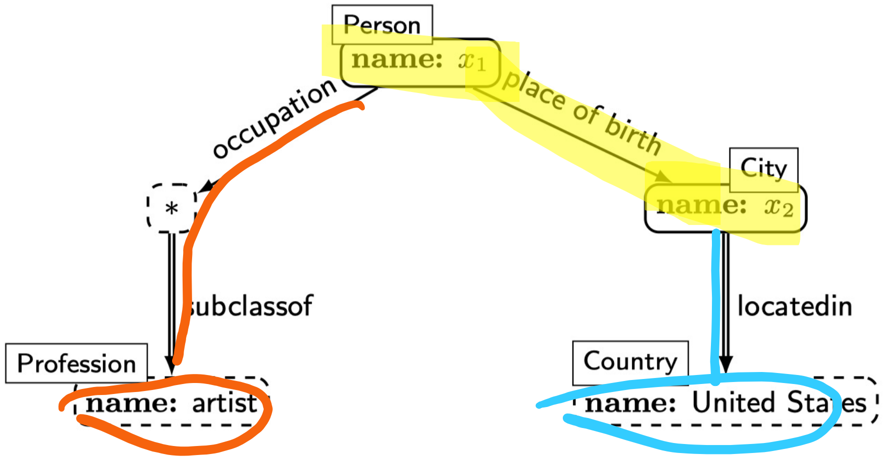


Figure 2: A tree pattern finding the artists who were born in the United States. The query returns the person names and the cities where they were born. (Fully circled nodes are return nodes.)

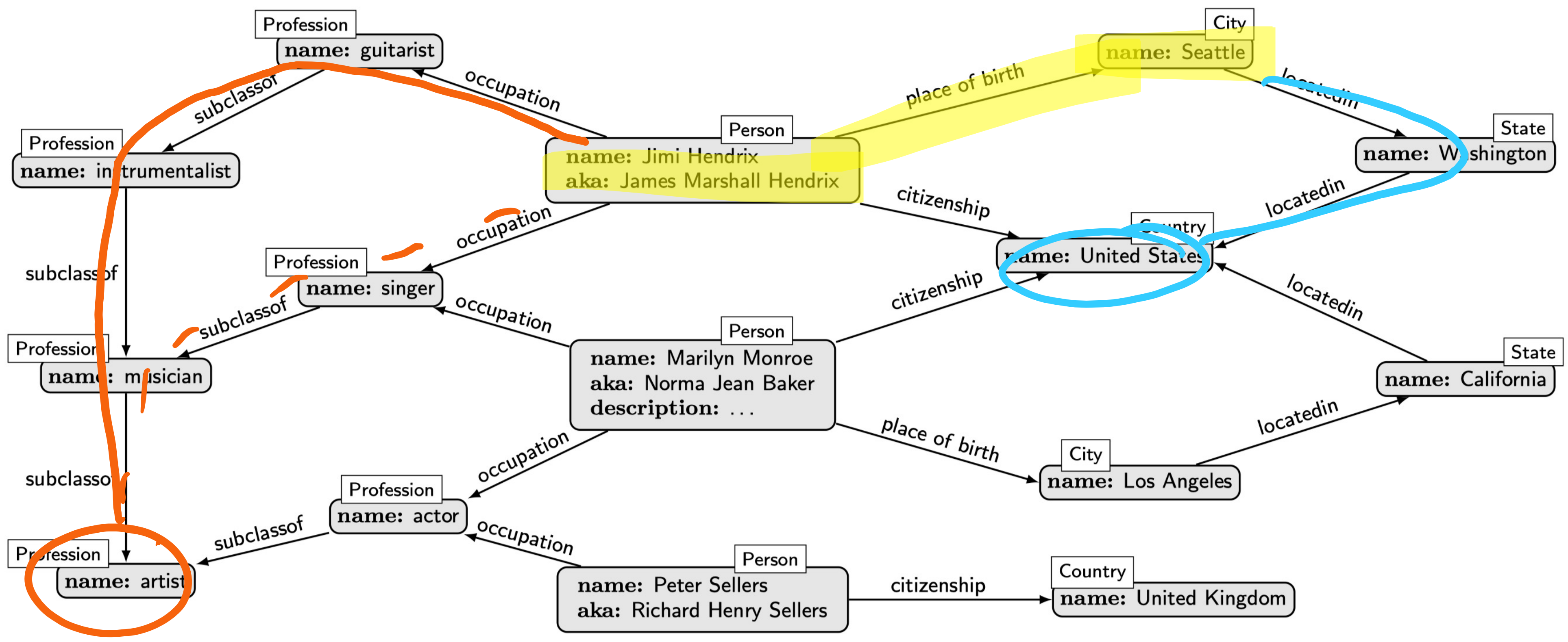


Figure 1: A graph database (as a *property graph*), inspired on a fragment of WikiData

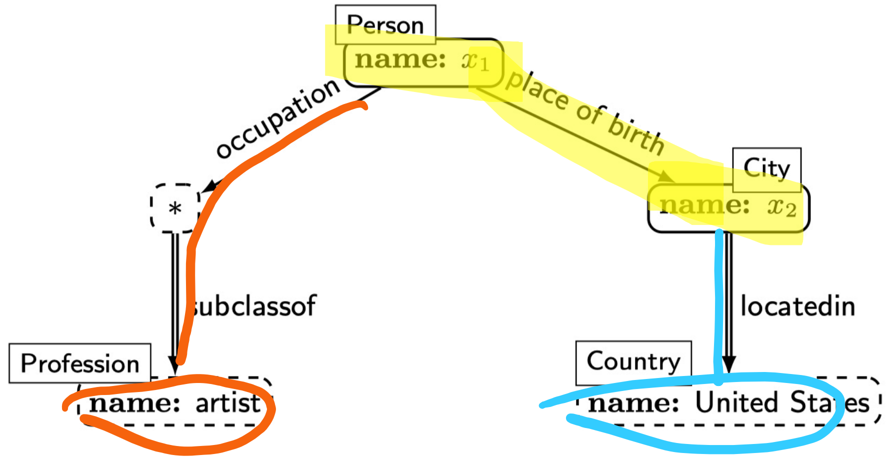


Figure 2: A tree pattern finding the artists who were born in the United States. The query returns the person names and the cities where they were born. (Fully circled nodes are return nodes.)

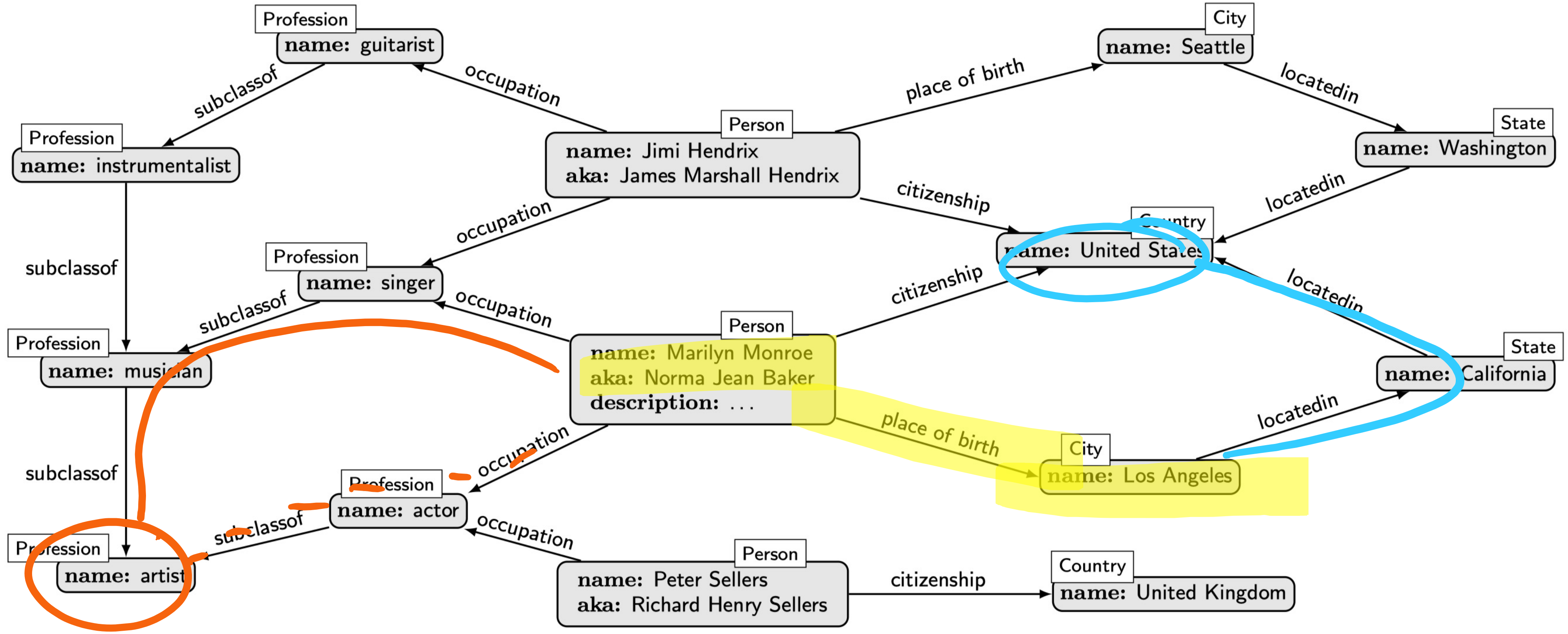


Figure 1: A graph database (as a *property graph*), inspired on a fragment of WikiData

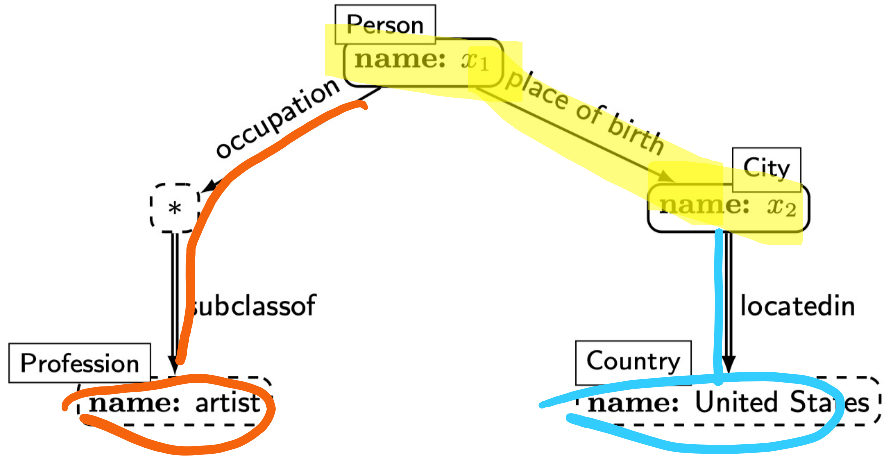
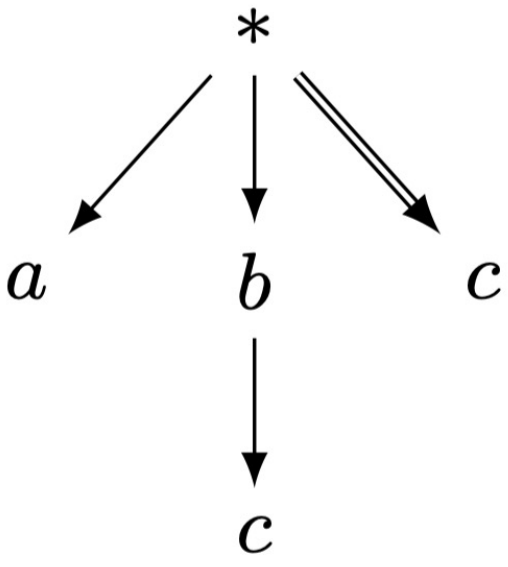
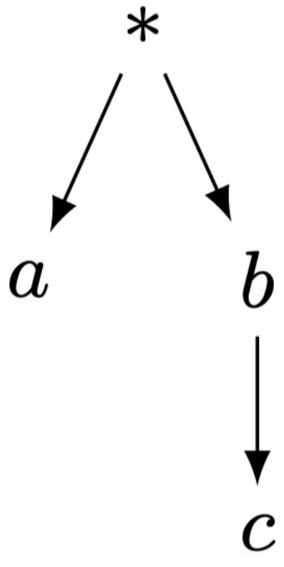


Figure 2: A tree pattern finding the artists who were born in the United States. The query returns the person names and the cities where they were born. (Fully circled nodes are return nodes.)

Optimizing tree patterns



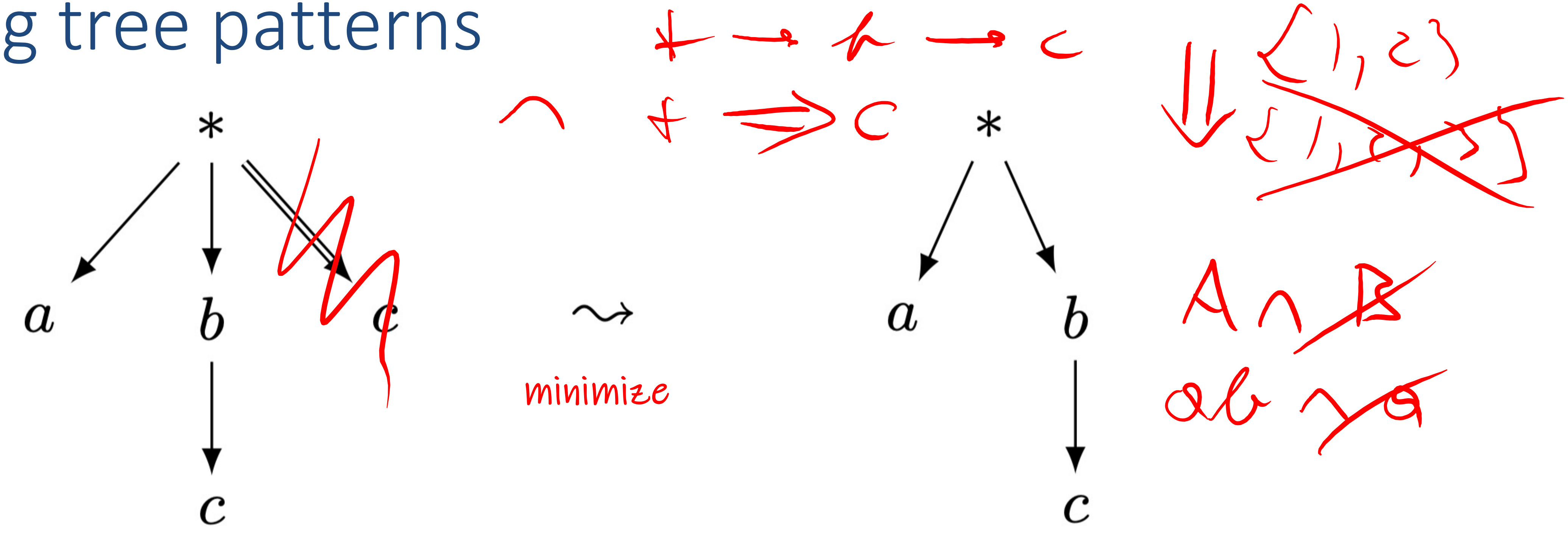


How are those two tree patterns
related to each other?

?

Example from: "Optimizing Tree Patterns for Querying Graph- and Tree-Structured Data" by Czerwinski, Martens, Niewerth, Parys. SIGMOD record 2017. https://doi.org/10.1145/3093754.3093759

Optimizing tree patterns



TREE PATTERN MINIMIZATION	
Given:	A tree pattern p and $k \in \mathbb{N}$
Question:	Is there a tree pattern q, equivalent to p, such that its size is at most k?

Minimality =? Nonredundancy

1.4 History of the Problem

Although the patterns we consider here have been widely studied [14, 24, 36, 15, 22, 1, 9, 4, 32], their minimization problem remained elusive for a long time. The most important previous work for their minimization was done by Kimelfeld and Sagiv [22] and by Flesca, Furfaro, and Masciari [14, 15].

The key challenge was understanding the relationship between *minimality* (M) and *nonredundancy* (NR). Here, a tree pattern is minimal if it has the smallest number of nodes among all equivalent tree patterns. It is nonredundant if none of its leaves (or branches[2]) can be deleted while remaining equivalent. The question was if minimality and nonredundancy are the same ([22, Section 7] and [15, p. 35]):

> $M \stackrel{?}{=} NR$ PROBLEM:
>
> Is a tree pattern minimal
> if and only if it is nonredundant?

Notice that a part of the $M \stackrel{?}{=} NR$ problem is easy to see: a minimal pattern is trivially also nonredundant (that is, $M \subseteq NR$). The opposite direction is much less clear.

If the problem would have a positive answer, it would mean that the simple algorithmic idea summarised in Algorithm 1 correctly minimizes tree patterns. Therefore, the $M \stackrel{?}{=} NR$ problem is a natural question about the design of minimization algorithms for tree patterns.

Algorithm 1 Computing a nonredundant subpattern

Input: A tree pattern p
Output: A nonredundant tree pattern q, equivalent to p

```
while a leaf of p can be removed
                    (remaining equivalent to p) do
    Remove the leaf
end while
return the resulting pattern
```

The M $\stackrel{?}{=}$ NR problem is also a question about complexity. The main source of complexity of the nonredundancy algorithm lies in testing equivalence between a pattern p and a pattern p', which is generally coNP-complete [24]. If M $\stackrel{?}{=}$ NR has a positive answer, then TREE PATTERN MINIMIZATION would also be coNP-complete.

In fact, the problem was claimed to be coNP-complete in 2003 [14, Theorem 2], but the status of the minimization- and the M $\stackrel{?}{=}$ NR problems were re-opened by Kimelfeld and Sagiv [22], who found errors in the proofs. Flesca et al.'s journal paper then proved that M = NR for a limited class of tree patterns, namely those where *every wildcard node has at most one child* [15]. Nevertheless, for tree patterns,

(a) the status of the M $\stackrel{?}{=}$ NR problem and

(b) the complexity of the minimization problem

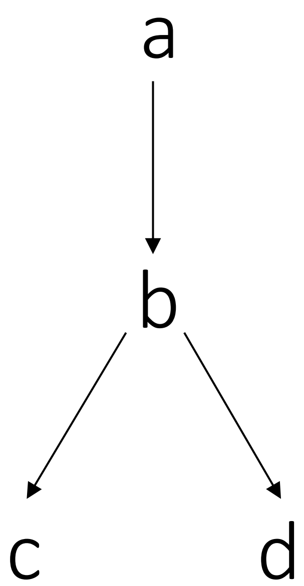
remained open.

Czerwinski, Martens, Niewerth, Parys [PODS 2016}

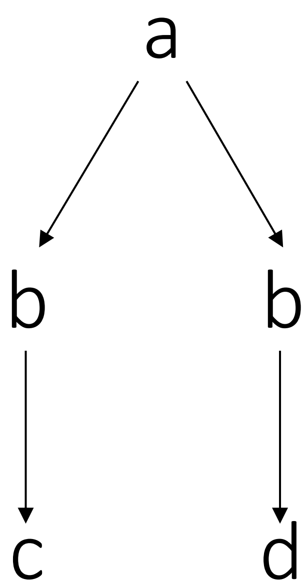
(a) There exists a tree pattern that is nonredundant but not minimal. Therefore, M $\neq$ NR.

(b) TREE PATTERN MINIMIZATION is Σ_2^P-complete. This implies that even the main idea in Algorithm 1 cannot work unless coNP = Σ_2^P.

Tree pattern containment



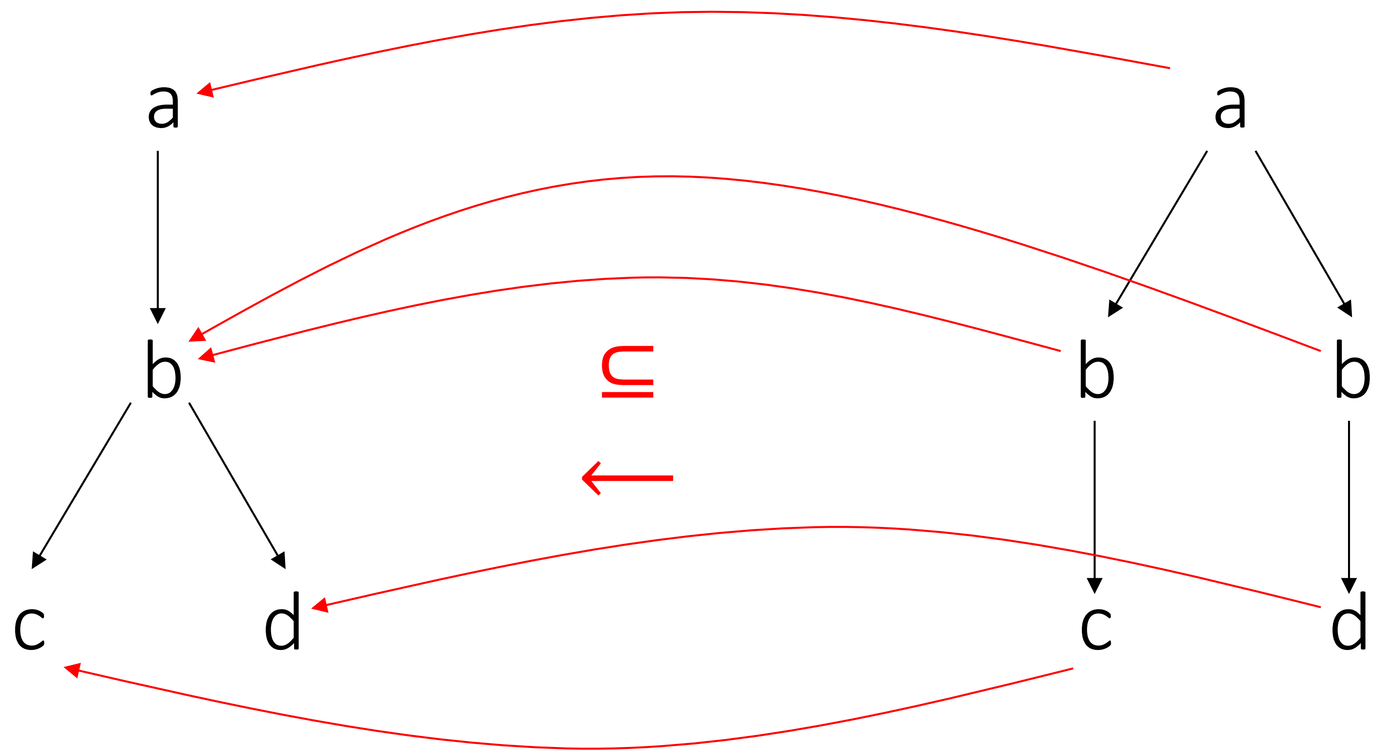
$\subseteq$ or $\supseteq$



?

Example from: "Optimizing Tree Patterns for Querying Graph- and Tree-Structured Data" by Czerwinski, Martens, Niewerth, Parys. SIGMOD record 2017. https://doi.org/10.1145/3093754.3093759

Tree pattern containment



but ⊉!

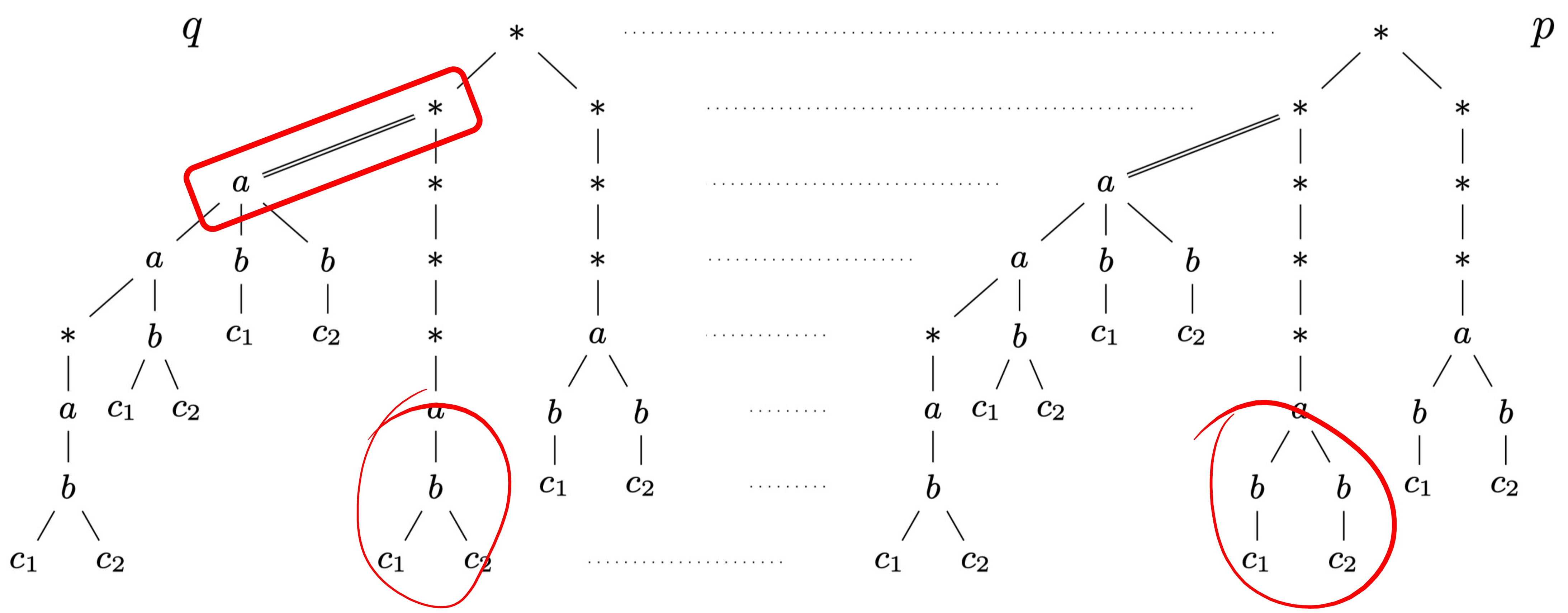
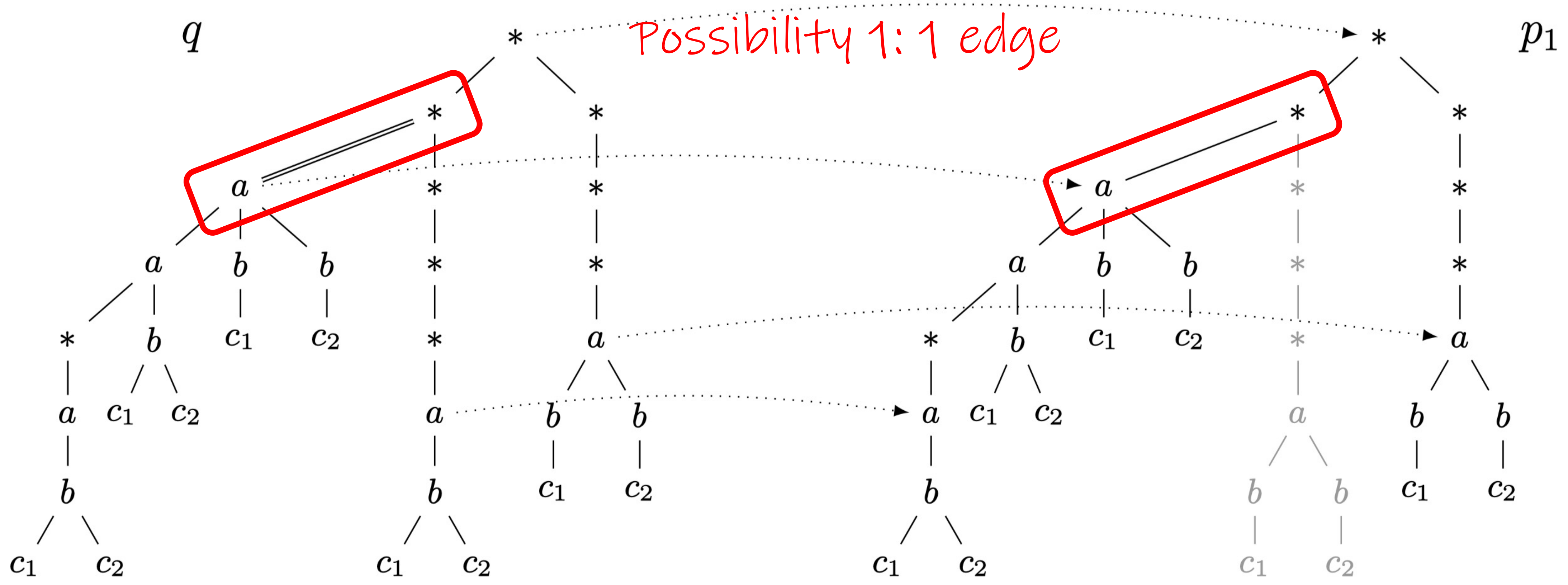


Figure 7: A non-redundant tree pattern p (right) and an equivalent tree pattern q that is smaller (left)

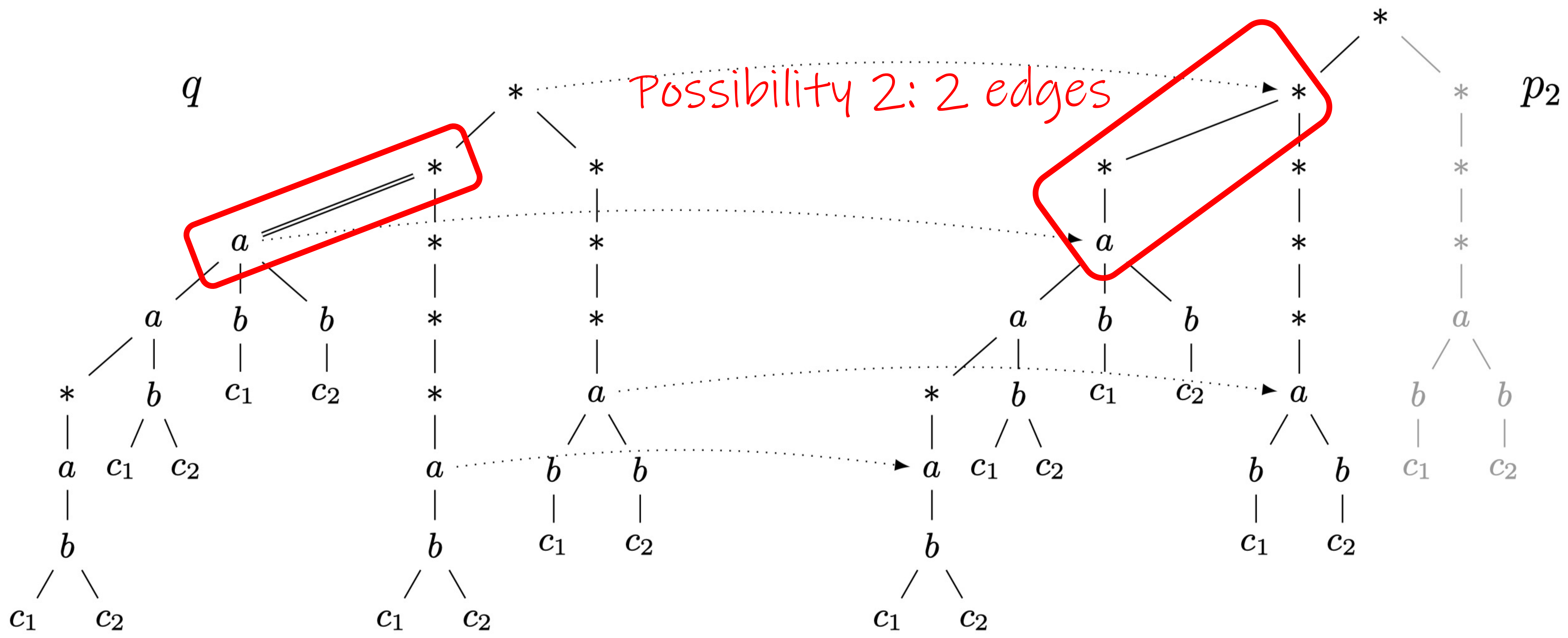
$q \subseteq p$ follows from argument on previous page.

To be shown $q \supseteq p$, then equivalent. Idea: whenever p matches, then also q.

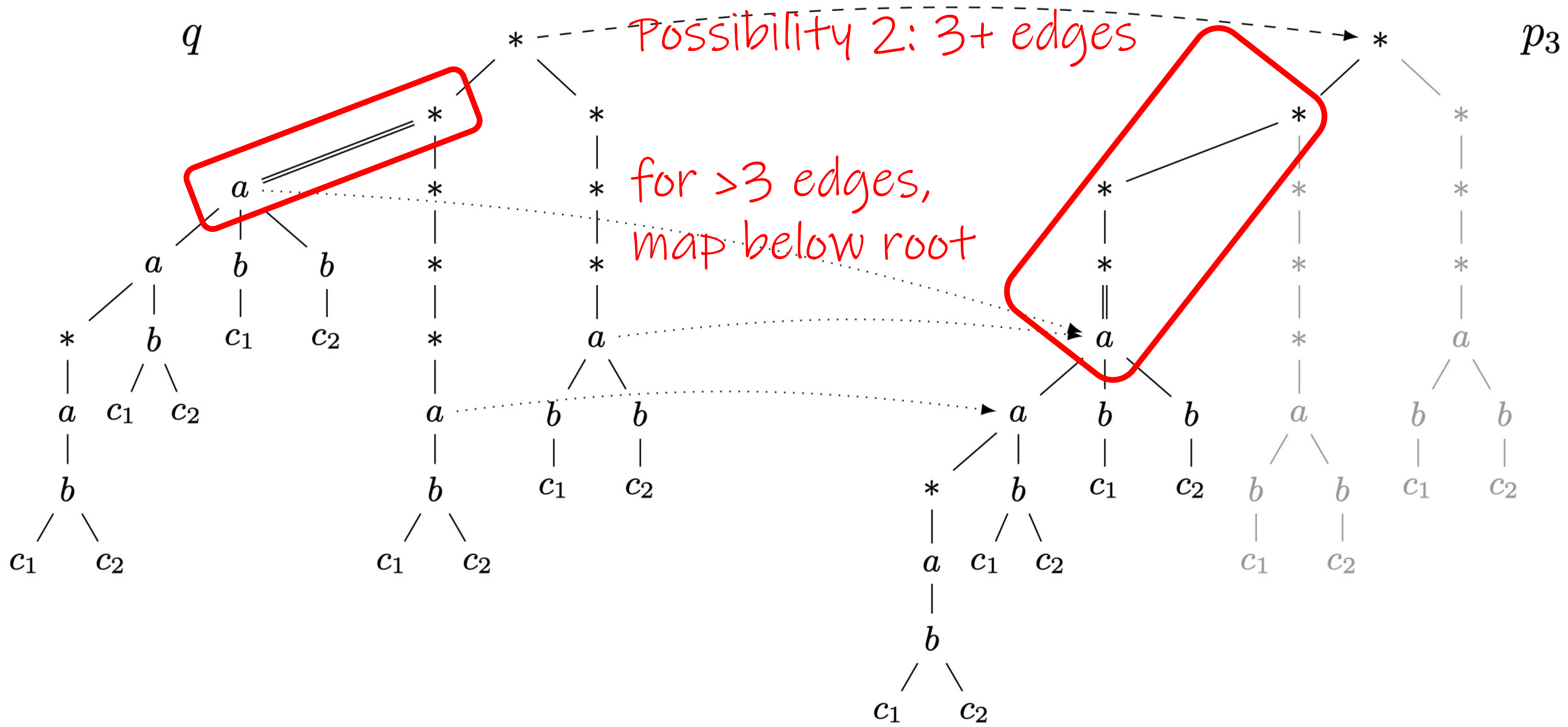
Idea: $a = \star$ can be matched in 3 ways in a graph



(a) How q can be matched if p_1 can be matched



(b) How q can be matched if p_2 can be matched



(c) How q can be matched if p_3 can be matched

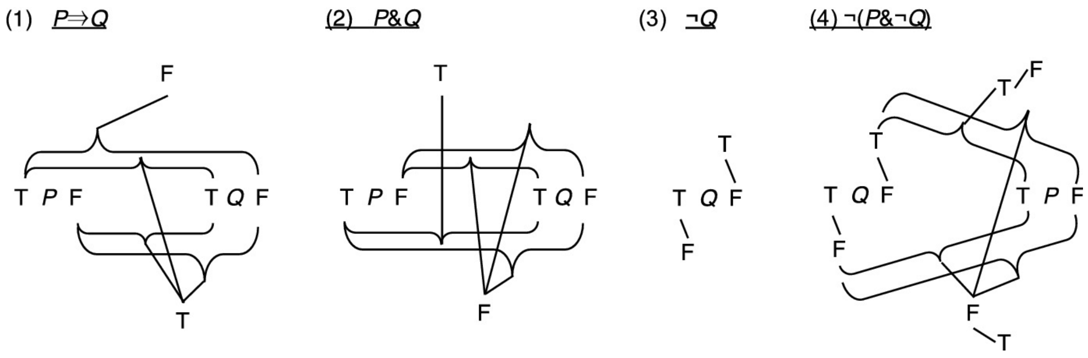
Outline: T2-1/2: Query Evaluation & Query Equivalence

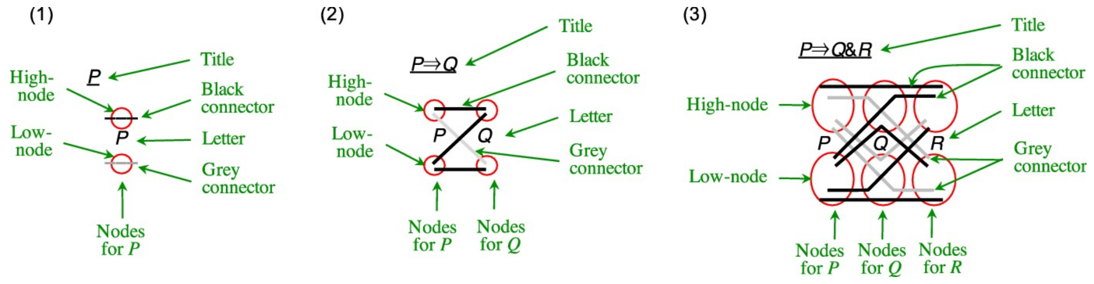
- T2-1: Conjunctive Queries (CQs)
  - CQ equivalence and containment
  - Graph homomorphisms
  - Homomorphism beyond graphs
  - CQ containment
  - CQ minimization
- T2-2: Equivalence Beyond CQs
  - Union of CQs, and inequalities
  - Union of CQs equivalence under bag semantics
  - Tree pattern queries
  - Nested queries

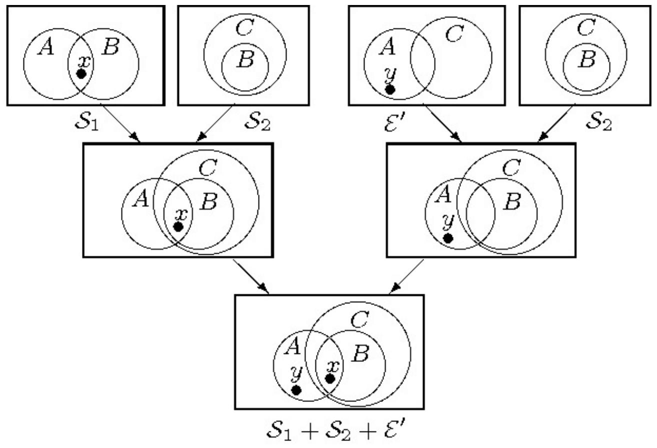
Equivalence of nested queries

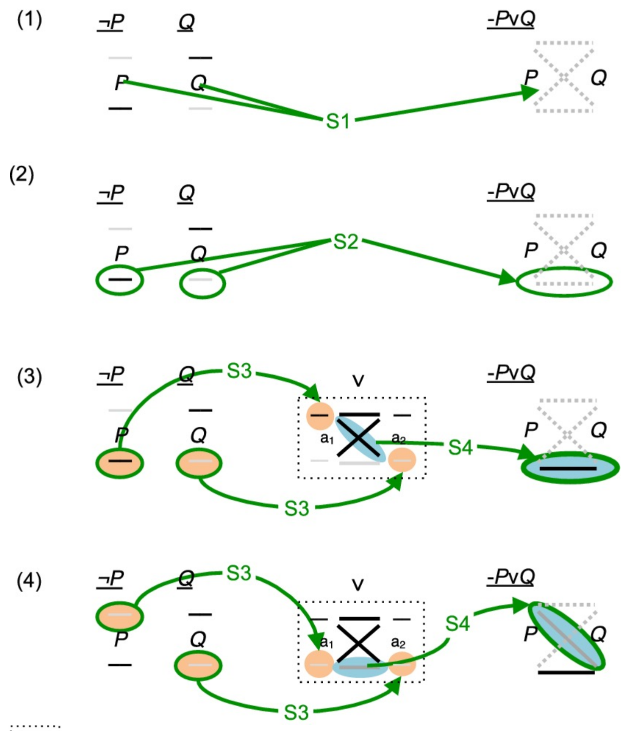
- Query equivalence is one of the foundational questions in database theory (and practice?)
  - touches on logics and decidability
  - what modifications allow tractability
- Lots of work (and open questions) on query equivalence
  - But not so much on nested queries!
- Related to QueryVis project (http://queryvis.com/) and two foundational questions on visual formalism:
  1. When can visual formalism *unambiguously* express logical statements?
  2. When can equivalent logical statements be transformed to each other by a sequence of visual transformations? (*Query equivalence*)

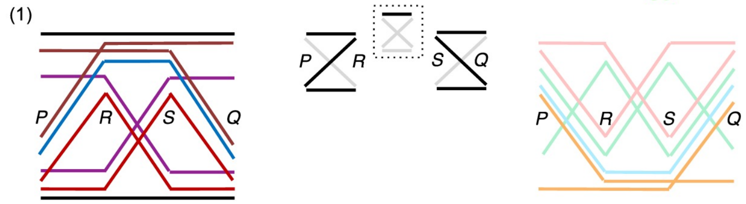
Diagrammatic reasoning systems and their expressiveness

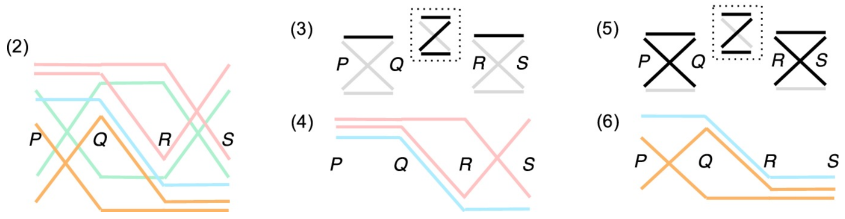




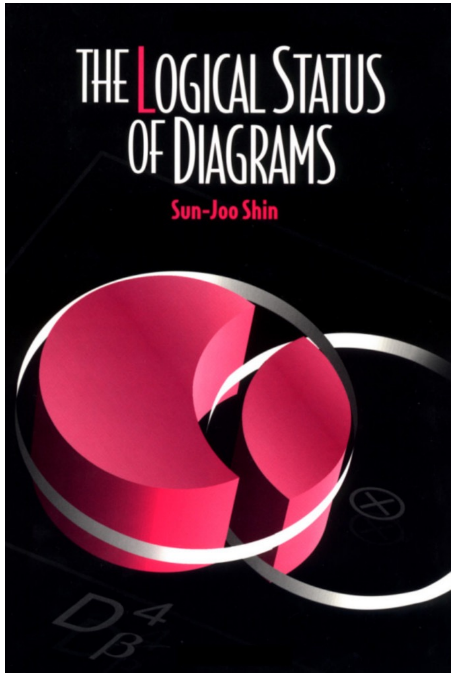








Diagrammatic reasoning systems and their expressiveness



Diagrams are widely used in reasoning about problems in physics, mathematics, and logic, but have traditionally been considered to be only heuristic tools and not valid elements of mathematical proofs. This book challenges this prejudice against visualization in the history of logic and mathematics and provides a formal foundation for work on natural reasoning in a visual mode.

The author presents Venn diagrams as a formal system of representation equipped with its own syntax and semantics and specifies rules of transformation that make this system sound and complete. The system is then extended to the equivalent of a first-order monadic language. The soundness of these diagrammatic systems refutes the contention that graphical representation is misleading in reasoning. The validity of the transformation rules ensures that the correct application of the rules will not lead to fallacies. The book concludes with a discussion of some fundamental differences between graphical systems and linguistic systems.

This groundbreaking work will have important influence on research in logic, philosophy, and knowledge representation.

objects. Conjunctive information is more naturally represented by diagrams than by linguistic formulæ. For example, a single Venn diagram can

Still, not all relations can be viewed as membership or inclusion. Shin has been careful throughout her book to restrict herself to monadic systems. Relations per se (polyadic predicates) are not considered. And while it may be true that the formation of a system (such as Venn-II) that is provably both sound and complete would help mitigate the prejudice

perception. In her discussion of perception she shows that disjunctive information is not representable in *any* system. In doing so she relies on

The logical status of diagrams, Sun-Joo Shin, Cambridge university press 1994. https://doi.org/10.1017/CBO9780511574696
Sun-Joo Shin at Yale: https://philosophy.yale.edu/people/sun-joo-shin

QueryVis

- Motivation: Can we create an automatic diagramming system that:
  - unambiguously visualizes the logical intent of a SQL query (thus no two different queries lead to an "identical" visualization; with "identical" to be formalized correctly)
  - for some important subset of nested queries
  - with visual diagrams that allow us to reason about logical SQL design patterns
- Related:
  - Lot's of interest on conjunctive queries equivalence. Now: For what fragment of nested queries is equivalence decidable (under set semantics)?
- Suggestion:
  - nested queries, with inequalities, without any disjunctions
  - Strict superset of conjunctive queries

Logical SQL Patterns

Logical patterns are the building blocks of most SQL queries.

Patterns are very hard to extract from the SQL text.

A pattern can appear across different database schemas.

Think of queries like:

- Find sailors who reserved all red boats
- Find students who took all art classes
- Find actors who played in all movies by Hitchcock

What does this query return ?

Likes(drinker,beer)

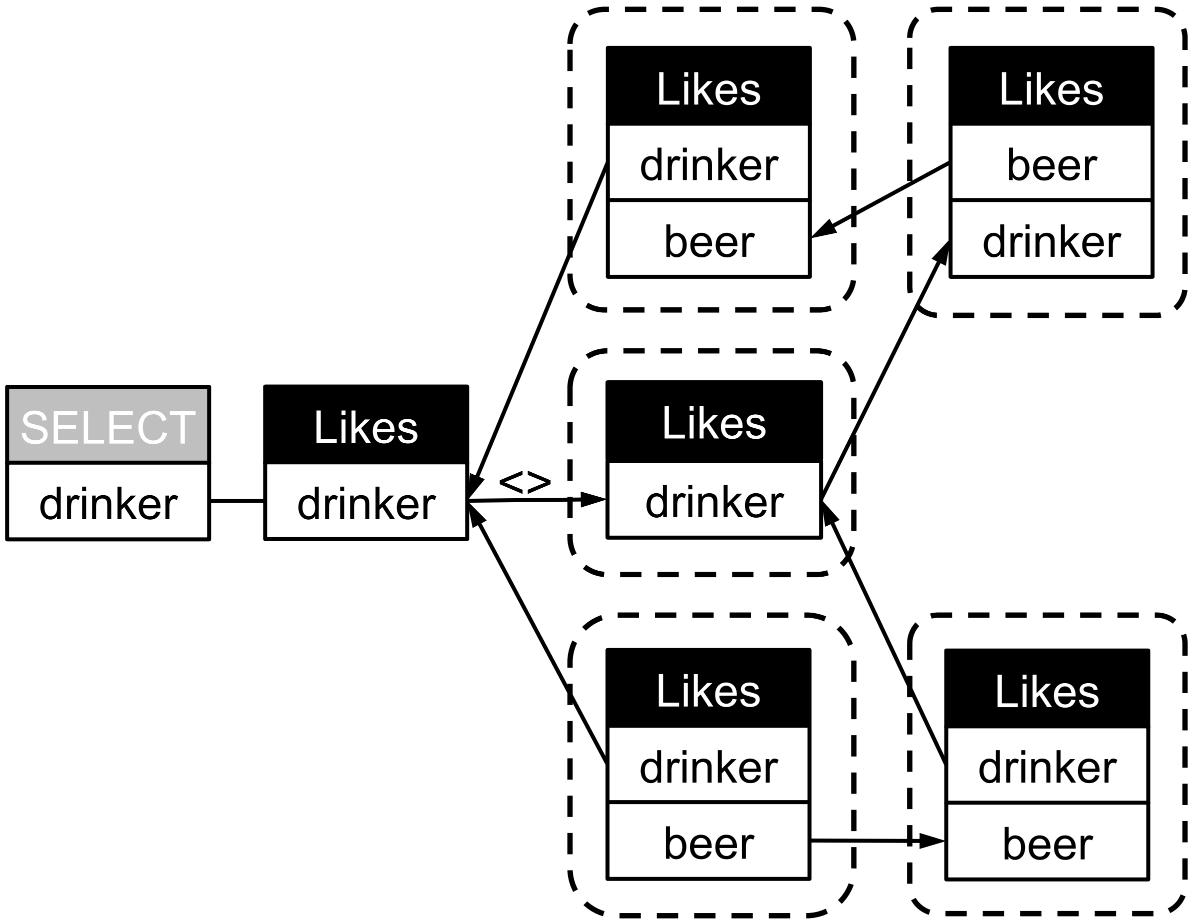
```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
  (SELECT *
  FROM Likes L2
  WHERE L1.drinker <> L2.drinker
  AND not exists
    (SELECT  *
    FROM Likes L3
    WHERE L3.drinker = L2.drinker
    AND not exists
      (SELECT *
      FROM Likes L4
      WHERE L4.drinker = L1.drinker
      AND L4.beer = L3.beer))
  AND not exists
    (SELECT *
    FROM Likes L5
    WHERE L5. drinker = L1. drinker
    AND not exists
      (SELECT *
      FROM Likes L6
      WHERE L6.drinker = L2.drinker
      AND L6.beer= L5.beer)))
```

What does this query return

Likes(drinker,beer)

```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
   (SELECT *
   FROM Likes L2
   WHERE L1.drinker <> L2.drinker
   AND not exists
      (SELECT  *
      FROM Likes L3
      WHERE L3.drinker = L2.drinker
      AND not exists
         (SELECT *
         FROM Likes L4
         WHERE L4.drinker = L1.drinker
         AND L4.beer = L3.beer))
   AND not exists
      (SELECT *
      FROM Likes L5
      WHERE L5. drinker = L1. drinker
      AND not exists
         (SELECT *
         FROM Likes L6
         WHERE L6.drinker = L2.drinker
         AND L6.beer= L5.beer)))
```

QueryVis scoping

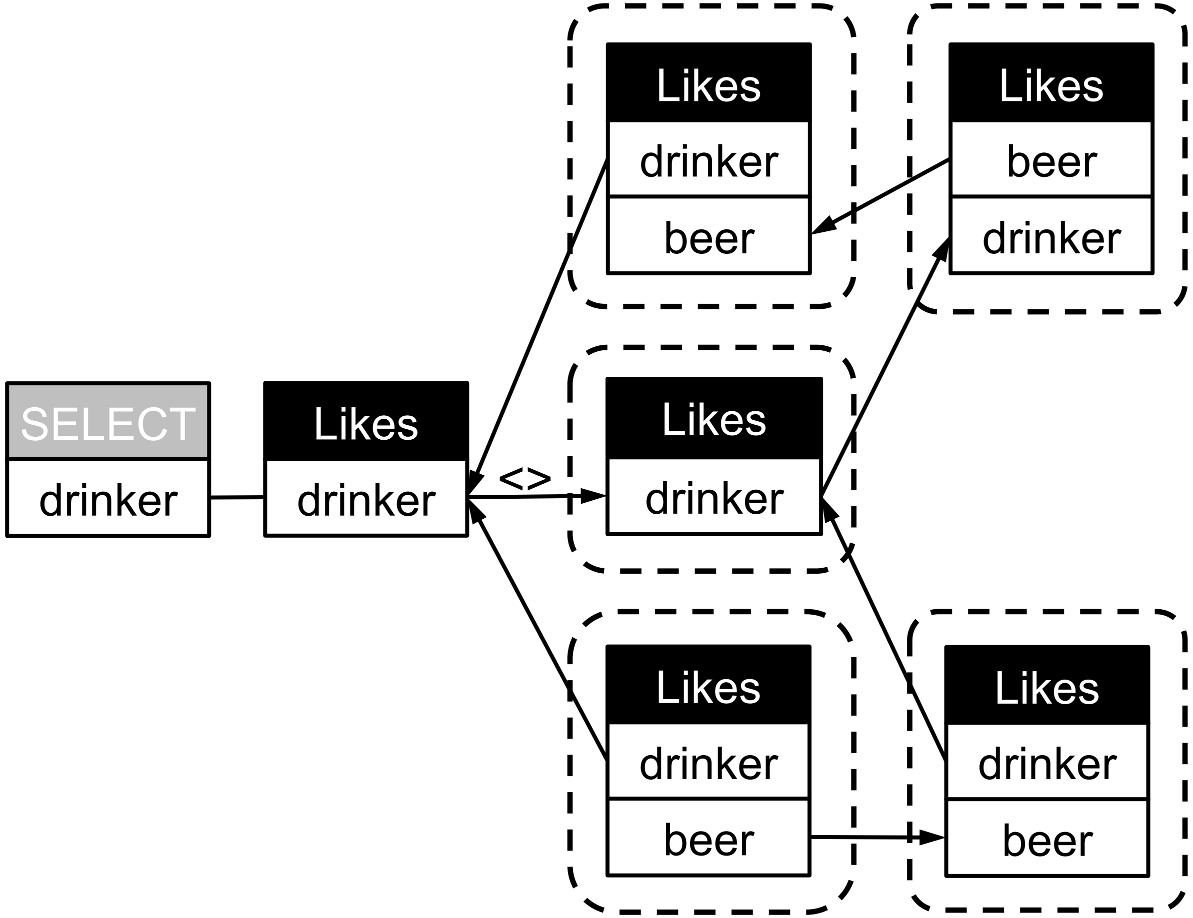


Q: Finder drinkers with a unique beer taste

Likes(drinker,beer)

```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
   (SELECT *
   FROM Likes L2
   WHERE L1.drinker <> L2.drinker
   AND not exists
      (SELECT  *
      FROM Likes L3
      WHERE L3.drinker = L2.drinker
      AND not exists
         (SELECT *
         FROM Likes L4
         WHERE L4.drinker = L1.drinker
         AND L4.beer = L3.beer))
   AND not exists
      (SELECT *
      FROM Likes L5
      WHERE L5. drinker = L1. drinker
      AND not exists
         (SELECT *
         FROM Likes L6
         WHERE L6.drinker = L2.drinker
         AND L6.beer= L5.beer)))
```

QueryVis scoping

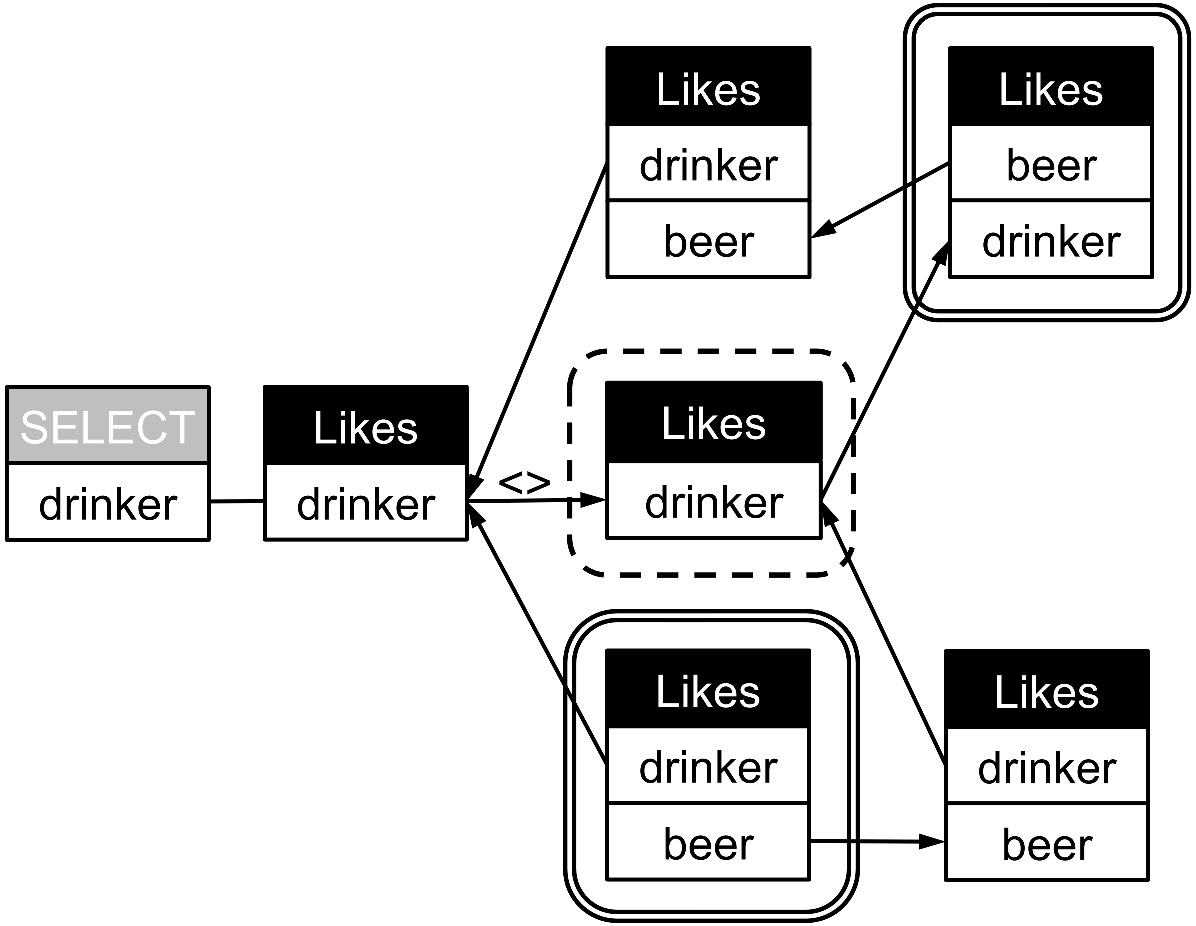


Q: Finder drinkers with a unique beer taste

Likes(drinker,beer)

```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
   (SELECT *
   FROM Likes L2
   WHERE L1.drinker <> L2.drinker
   AND not exists
      (SELECT  *
      FROM Likes L3
      WHERE L3.drinker = L2.drinker
      AND not exists
         (SELECT *
         FROM Likes L4
         WHERE L4.drinker = L1.drinker
         AND L4.beer = L3.beer))
   AND not exists
      (SELECT *
      FROM Likes L5
      WHERE L5. drinker = L1. drinker
      AND not exists
         (SELECT *
         FROM Likes L6
         WHERE L6.drinker = L2.drinker
         AND L6.beer= L5.beer)))
```

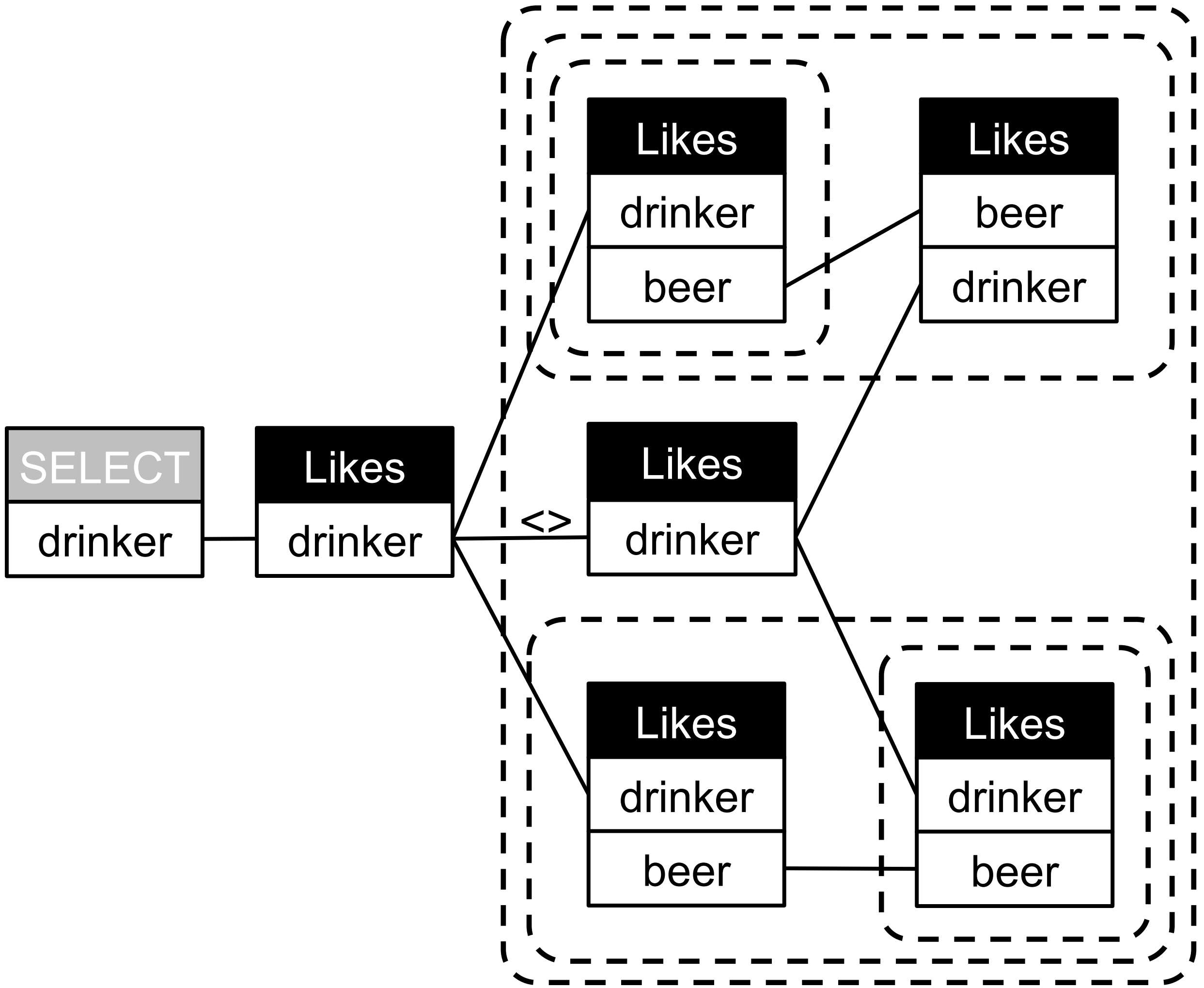
QueryVis scoping



Q: Finder drinkers with a unique beer taste

Likes(drinker,beer)

```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
   (SELECT *
   FROM Likes L2
   WHERE L1.drinker <> L2.drinker
   AND not exists
      (SELECT  *
      FROM Likes L3
      WHERE L3.drinker = L2.drinker
      AND not exists
         (SELECT *
         FROM Likes L4
         WHERE L4.drinker = L1.drinker
         AND L4.beer = L3.beer))
   AND not exists
      (SELECT *
      FROM Likes L5
      WHERE L5. drinker = L1. drinker
      AND not exists
         (SELECT *
         FROM Likes L6
         WHERE L6.drinker = L2.drinker
         AND L6.beer= L5.beer)))
```



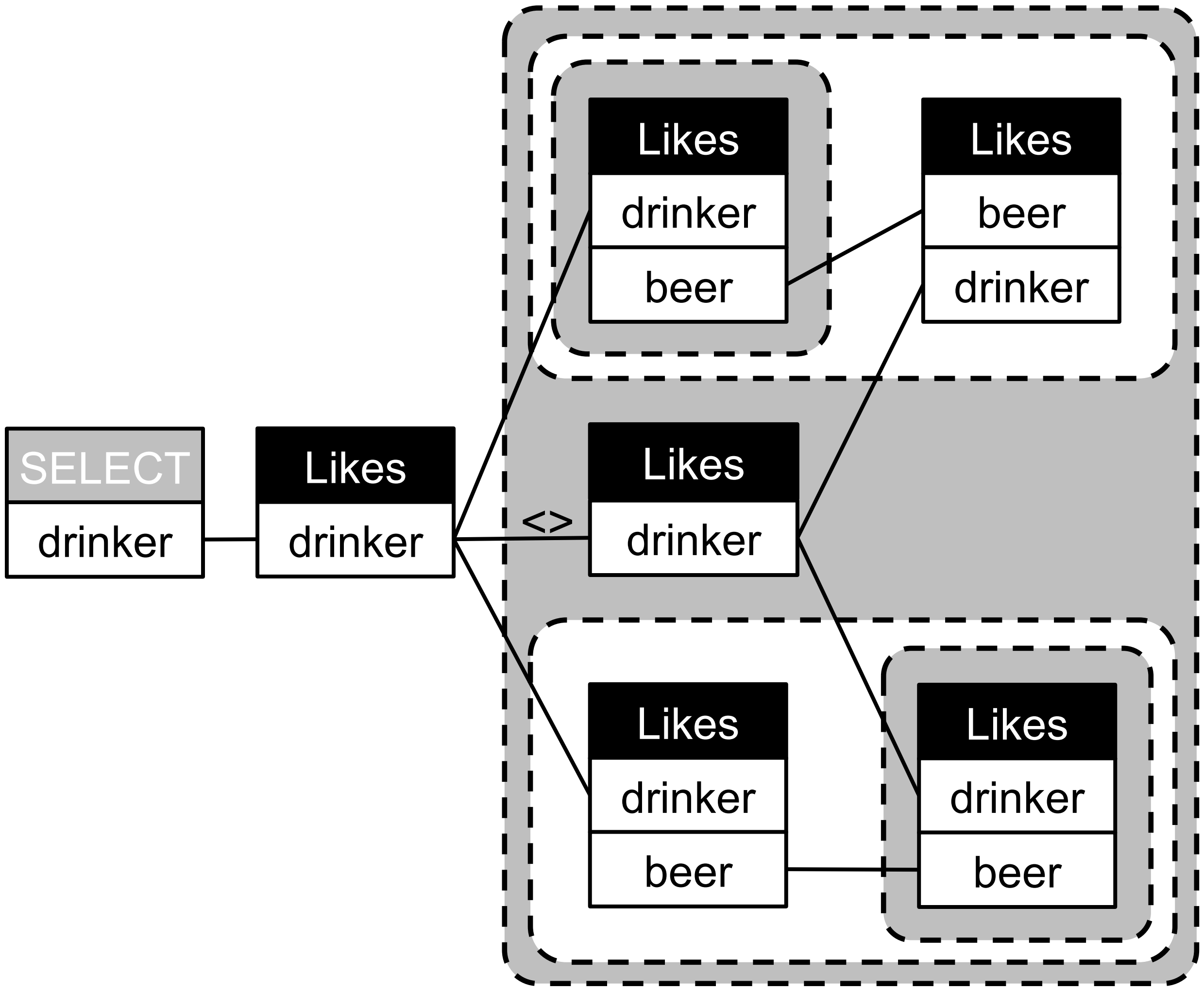
QueryVis scoping

Relational Diagrams scoping (https://relationaldiagrams.com)

Q: Finder drinkers with a unique beer taste

Likes(drinker,beer)

```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
   (SELECT *
   FROM Likes L2
   WHERE L1.drinker <> L2.drinker
   AND not exists
     (SELECT  *
     FROM Likes L3
     WHERE L3.drinker = L2.drinker
     AND not exists
       (SELECT *
       FROM Likes L4
       WHERE L4.drinker = L1.drinker
       AND L4.beer = L3.beer))
   AND not exists
     (SELECT *
     FROM Likes L5
     WHERE L5. drinker = L1. drinker
     AND not exists
       (SELECT *
       FROM Likes L6
       WHERE L6.drinker = L2.drinker
       AND L6.beer= L5.beer)))
```



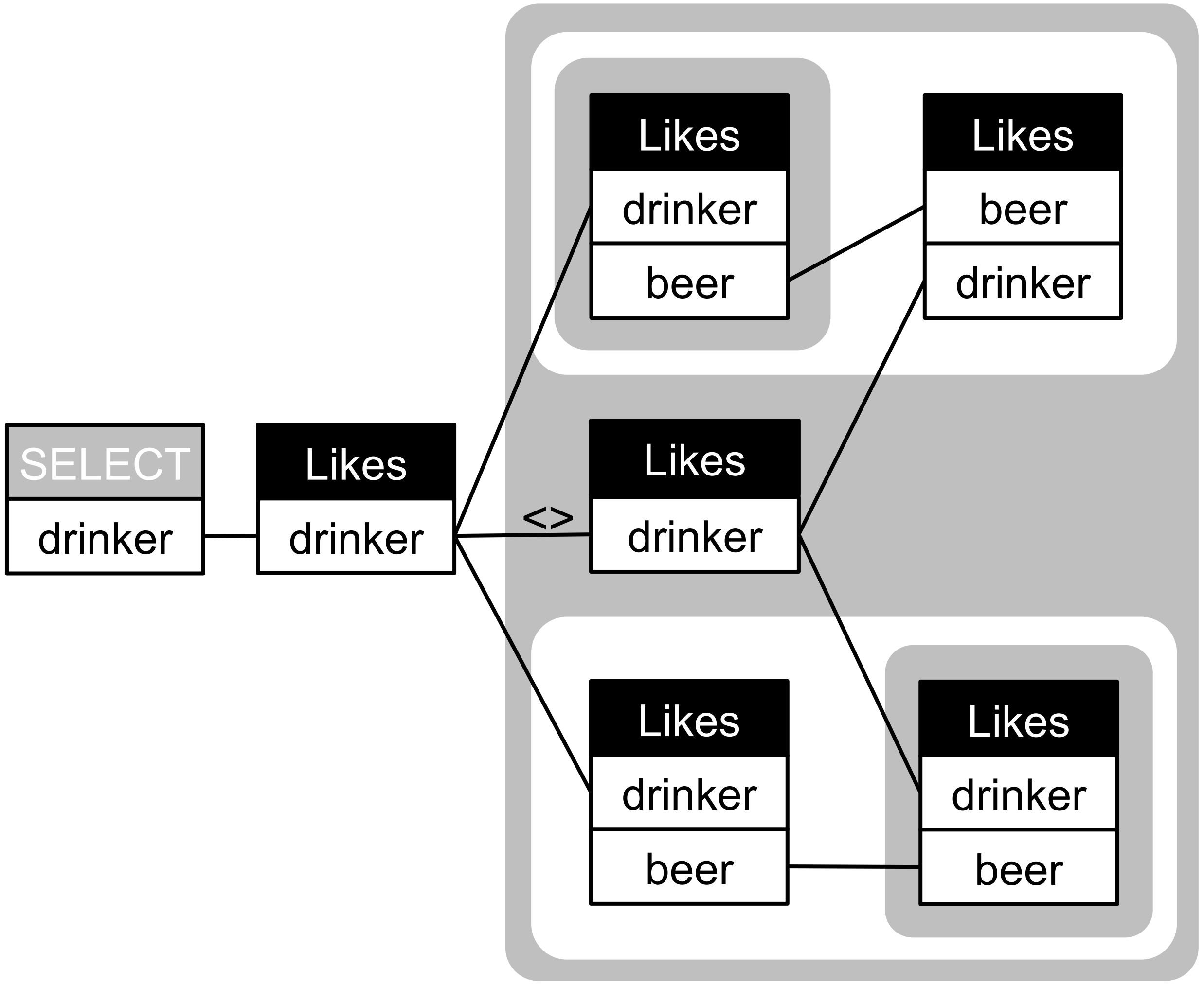
QueryVis scoping

Relational Diagrams scoping (https://relationaldiagrams.com)

Q: Finder drinkers with a unique beer taste

Likes(drinker,beer)

```
SELECT L1.drinker
FROM Likes L1
WHERE not exists
   (SELECT *
   FROM Likes L2
   WHERE L1.drinker <> L2.drinker
   AND not exists
     (SELECT  *
     FROM Likes L3
     WHERE L3.drinker = L2.drinker
     AND not exists
       (SELECT *
       FROM Likes L4
       WHERE L4.drinker = L1.drinker
       AND L4.beer = L3.beer))
   AND not exists
     (SELECT *
     FROM Likes L5
     WHERE L5. drinker = L1. drinker
     AND not exists
       (SELECT *
       FROM Likes L6
       WHERE L6.drinker = L2.drinker
       AND L6.beer= L5.beer)))
```



QueryVis scoping

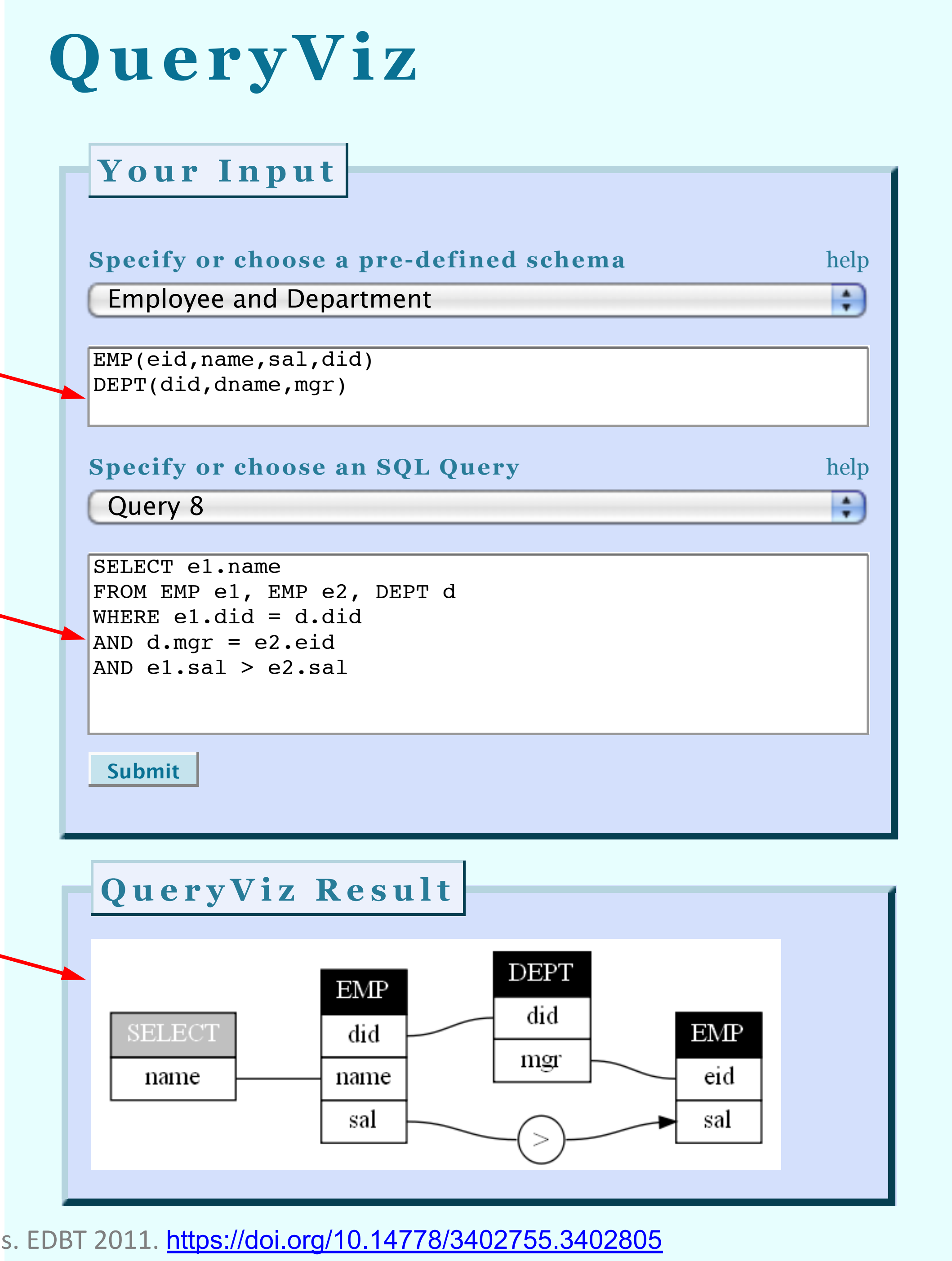
Relational Diagrams scoping (https://relationaldiagrams.com)

https://demo.queryvis.com

Input: Schema

Input Query

Output: Visualization



Danaparamita, G. [EDBT'11]

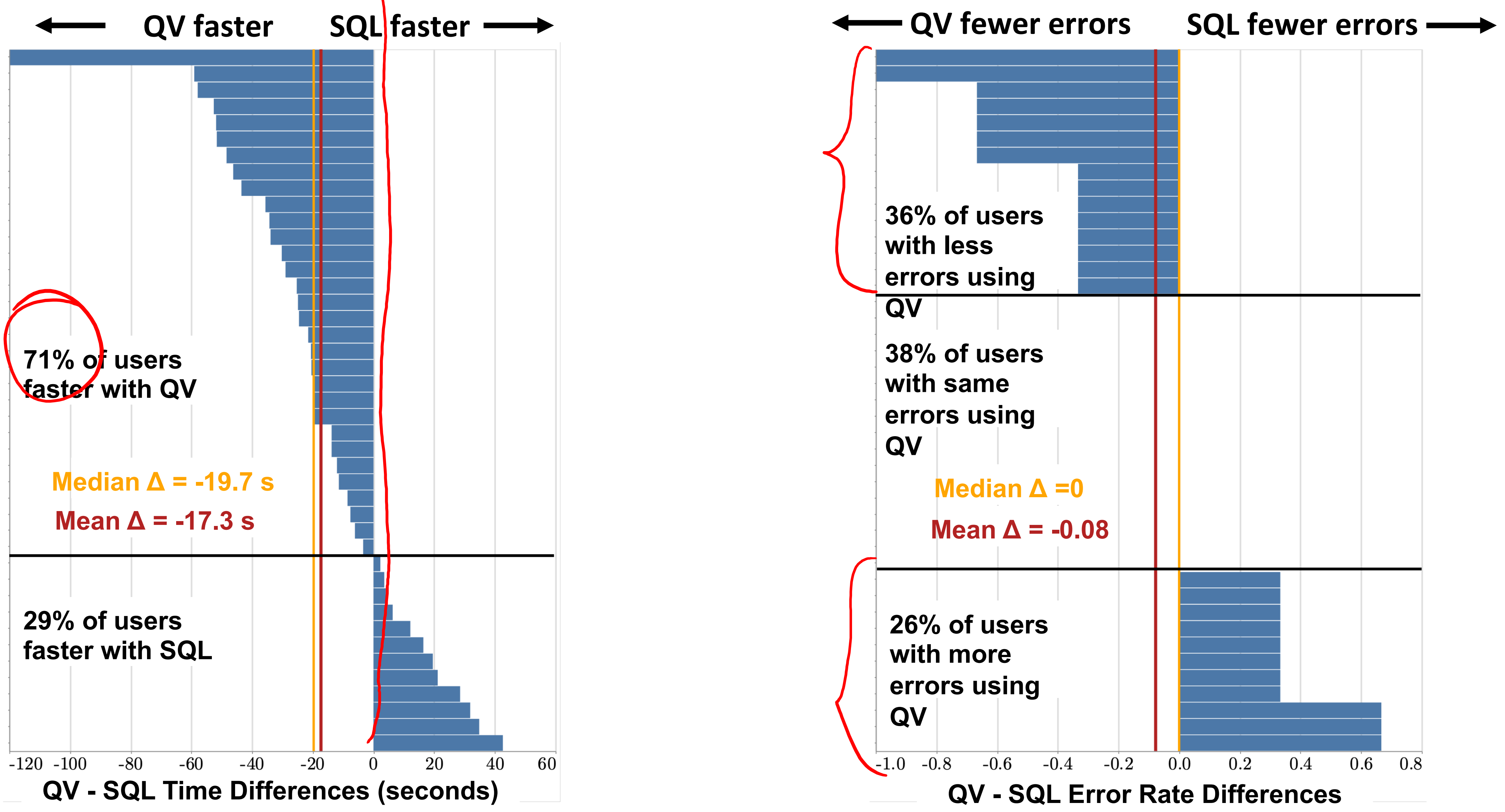
https://queryvis.com/

http://www.youtube.com/watch?v=kVFnQRGAQls

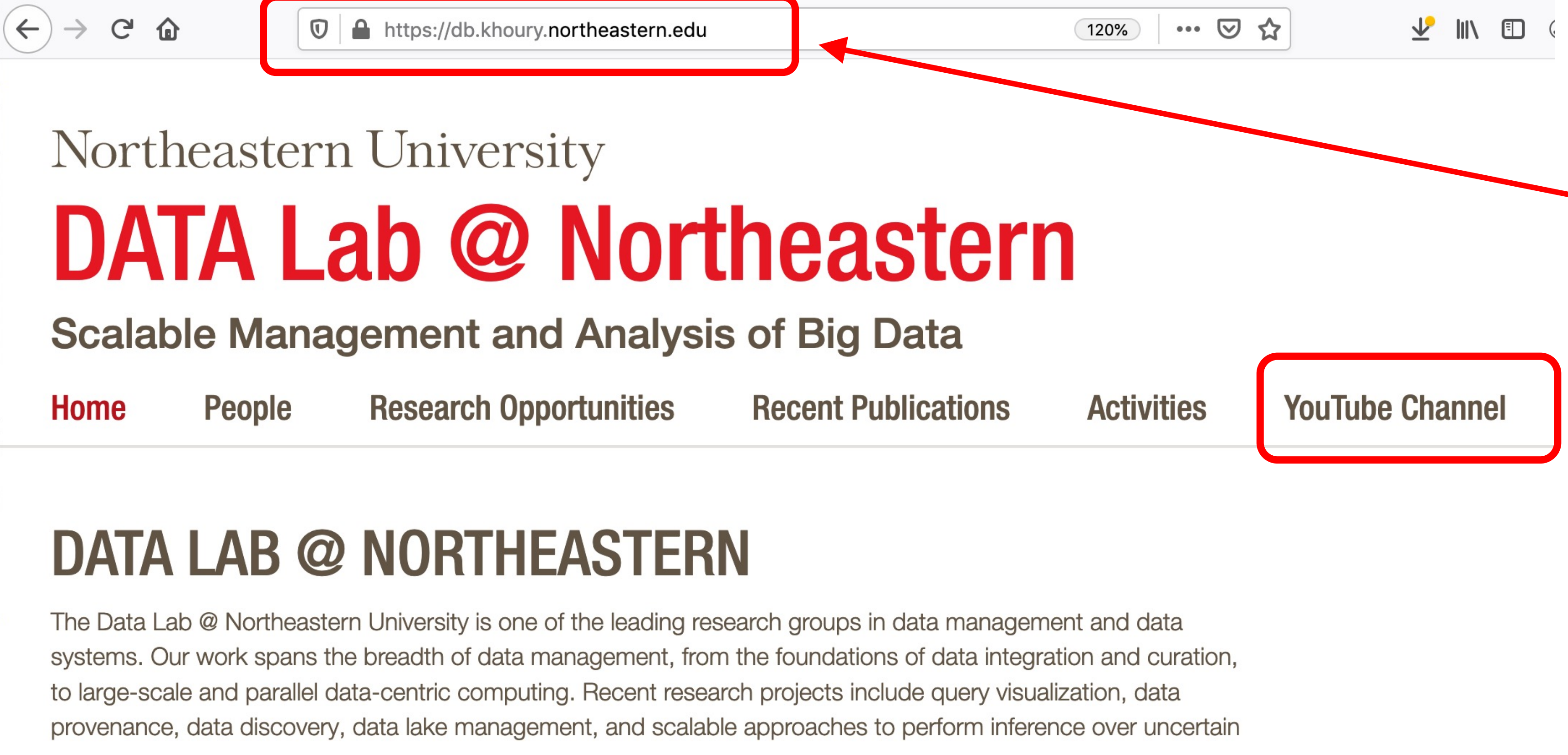
Amazon Turk user study with SQL users

Leventidis+ [SIGMOD'20]

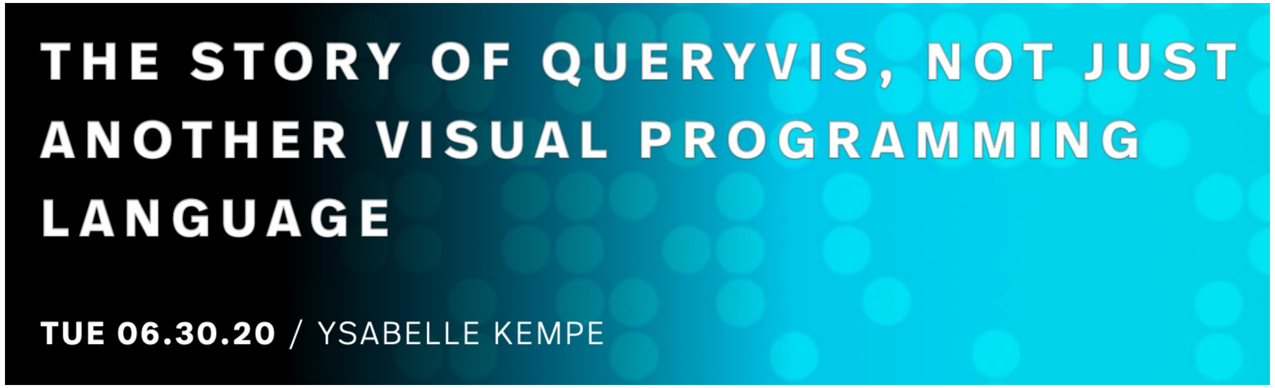
Each bar below corresponds to one participant (42 bars/participants in total)



Source: Leventidis, Zhang, Dunne, Gatterbauer, Jagadish, Riedewald: *QueryVis: Logic-based Diagrams help Users Understand Complicated SQL Queries Faster*. SIGMOD 2020. https://doi.org/10.1145/3318464.3389767



https://queryvis.com



https://www.khoury.northeastern.edu/the-story-of-queryvis-not-just-another-visual-programming-language/

Unique set query: "Find drinkers that like a unique set of beers."

"Return any drinker, s.t. there does not exist any other drinker, s.t. there does not exist any beer liked by that other drinker that is not also liked by the returned drinker and there does not exist any beer liked by the returned drinker that is not also liked by the same other drinker."

Let x be a drinker, and S(x) be the set of liked beers by drinker x.
Find any drinker x, s.t. there does not exist another drinker x ' , x for which:
$S(x') \subseteq S(x)$ *and* $S(x') \supseteq S(x)$

Unique set query: "Find drinkers that like a unique set of beers."

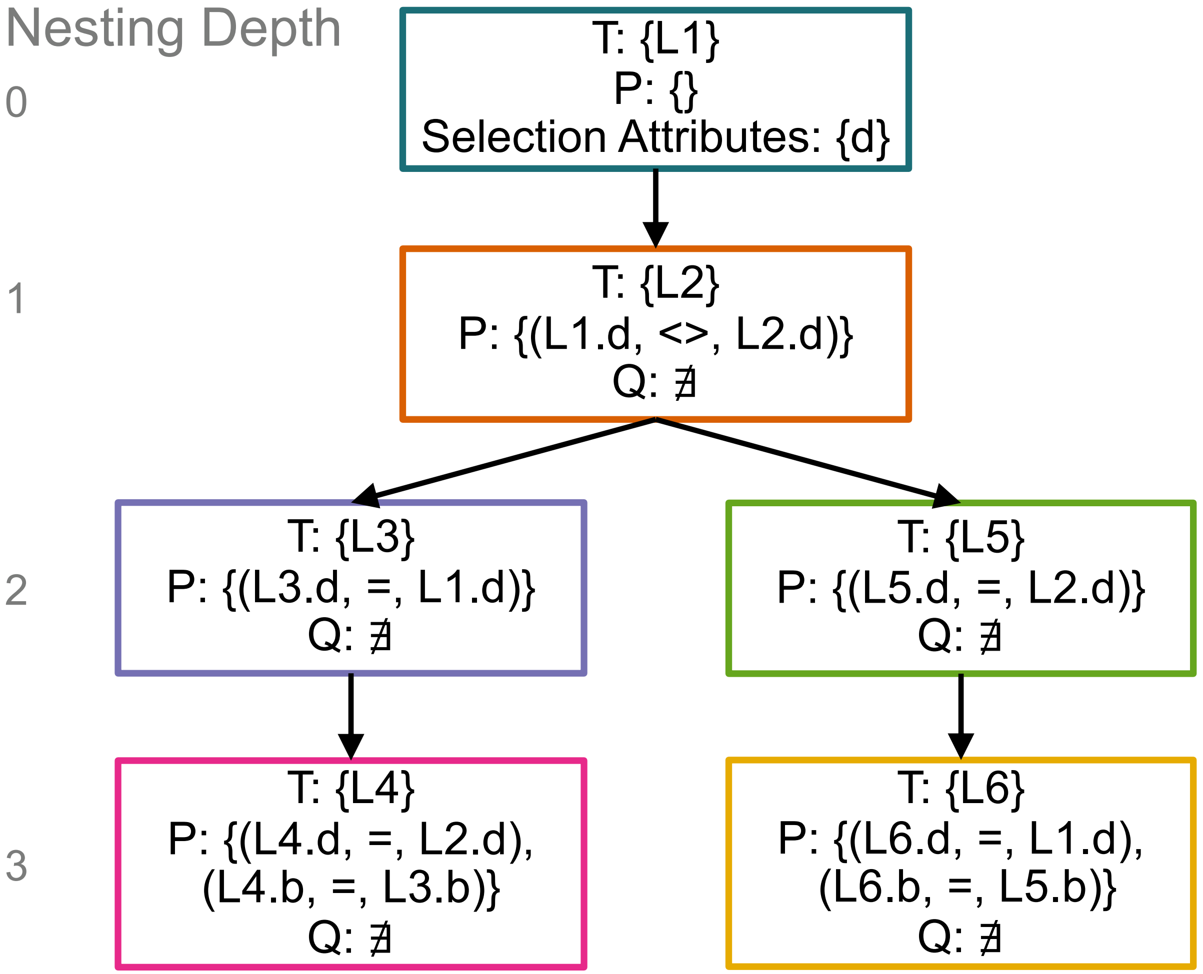
Likes
drinker
beer

Likes
d
b

$$
\begin{aligned}
&\{ L1.d \mid \exists L1 \in Likes \wedge \\
&\quad \nexists L2 \in Likes\, [L2.d <> L1.d \wedge \\
&\qquad \nexists L3 \in Likes\, [L3.d = L1.d \wedge \\
&\qquad\quad \nexists L4 \in Likes\, [L4.d = L2.d \wedge L4.b = L3.b]] \wedge \\
&\qquad \nexists L5 \in Likes\, [L5.d = L2.d \wedge \\
&\qquad\quad \nexists L6 \in Likes\, [L6.d = L1.d \wedge L6.b = L5.b]]]\}
\end{aligned}
$$

Notice how the logic tree portrays the nesting hierarchy shown in the FOL (TRC) representation of the SQL query.

Each node in the LT represents the root of a scope in the FOL representation. The predicates in each node are the predicates in the root of the scope of a given node (thus the predicates which do not use any additionally quantified variables).



Atomic predicate classification

Our simple rule: **every predicate needs to have at least one local table identifier.**

Allowed:
- local op value (local selection pred.)
- local op local (local join pred.)
- local op ancestor (connecting join pred.)

Not allowed:
- ancestor op value (foreign selection pred.)
- ancestor op ancestor (foreign join pred.)

type

scope	**selection p.**	**join p.**
local *(all C are local)*	**C O V**	**C O C**
connecting *(one C is local, another one is foreign*	X	**C O C**
foreign *(all C are foreign)*		

Focus: one single nesting level

- We first restrict ourselves to
  - equi-joins (no inequalities like T.A < T.B)
  - paths (no siblings = every node can have only one nested child)
  - one single nesting level
  - Boolean queries
  - no foreign predicates
  - only binary relations (thus can be represented as graphs)
  - only one single relation R
  - (and as before only conjunctions)
- Given two such queries, what is a generalization of the homomorphism procedure that works for that fragment?

Simplifying notation

Schema: R(A,B)

What will become handy, is a short convenient notation for queries

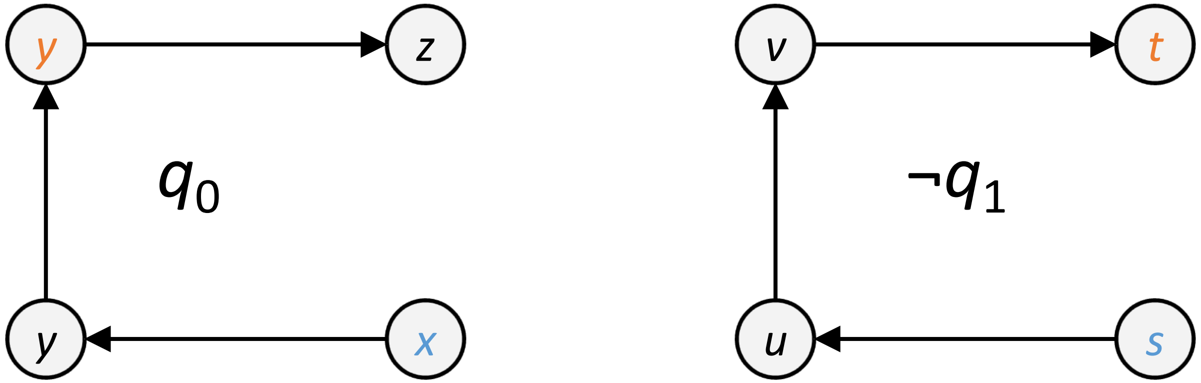
```
SELECT   TRUE
FROM     R R1, R R2, R R3
WHERE    R1.B = R2.A
AND      R2.B = R3.A
NOT EXISTS
        (SELECT *
        FROM    R R4, R R5, R R6
        WHERE   R4.B = R5.A
        AND     R5.B = R6.A
        AND     R4.A = R1.A
        AND     R6.A = R2.B)
```

∃ R1, R2, R3 ∈ R
(R1.B=R2.A ∧ R2.B=R3.A ∧
∄ R4, R5, R6 ∈ R
(R4.B=R5.A ∧ R5.B=R6.A ∧
R4.A=R1.A ∧ R6.A = R2.B)
)

q_0 :- $R(x,y), R(y,z), R(z,w)$

$q_1(s,t)$:- $R(s,u), R(u,v), R(v,t)$, s=x, t=y

q :- $R(x,y), R(y,z), R(z,w), \neg q_1(x,z)$



s=x, t=y

Simplifying notation

Schema: R(A,B)

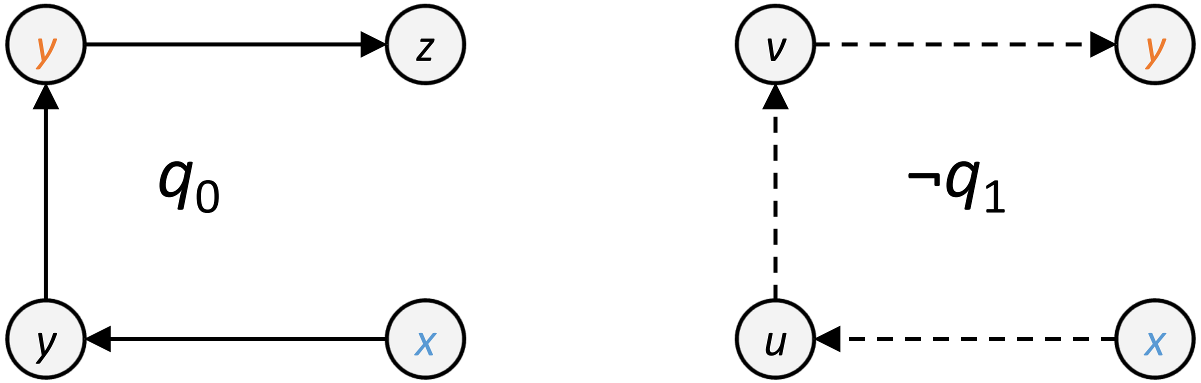
What will become handy, is a short convenient notation for queries

```
SELECT  TRUE
FROM    R R1, R R2, R R3
WHERE   R1.B = R2.A
AND     R2.B = R3.A
NOT EXISTS
        (SELECT  *
        FROM     R R4, R R5, R R6
        WHERE    R4.B = R5.A
        AND      R5.B = R6.A
        AND      R4.A = R1.A
        AND      R6.A = R2.B)
```

q_0 :- $R(x,y), R(y,z), R(z,w)$

$\neg q_1$:- $R(x,u), R(u,v), R(v,y)$

$\exists$ R1, R2, R3 $\in$ R
  (R1.B=R2.A $\wedge$ R2.B=R3.A $\wedge$
    $\nexists$ R4, R5, R6 $\in$ R
      (R4.B=R5.A $\wedge$ R5.B=R6.A $\wedge$
      R4.A=R1.A $\wedge$ R6.A = R2.B)
)



Simplifying notation

Schema: R(A,B)

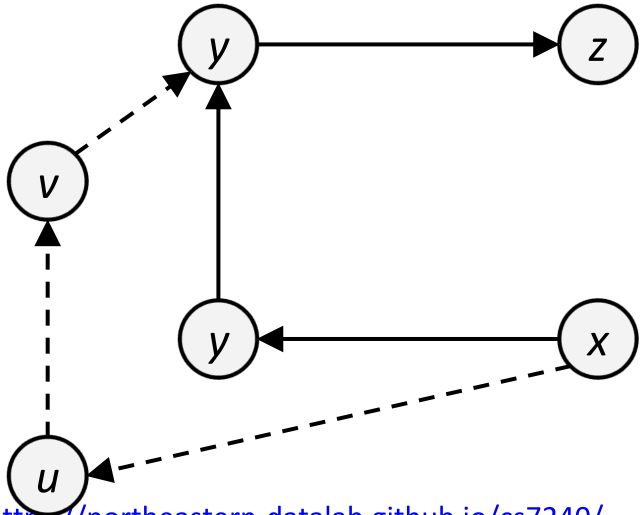
What will become handy, is a short convenient notation for queries

```
SELECT  TRUE
FROM    R R1, R R2, R R3
WHERE   R1.B = R2.A
AND     R2.B = R3.A
NOT EXISTS
        (SELECT  *
        FROM     R R4, R R5, R R6
        WHERE    R4.B = R5.A
        AND      R5.B = R6.A
        AND      R4.A = R1.A
        AND      R6.A = R2.B)
```

$$q_0 \text{ :- } R(x,y), R(y,z), R(z,w)$$

$$\neg q_1 \text{ :- } R(x,u), R(u,v), R(v,y)$$

∃ R1, R2, R3 ∈ R
(R1.B=R2.A ∧ R2.B=R3.A ∧
∄ R4, R5, R6 ∈ R
(R4.B=R5.A ∧ R5.B=R6.A ∧
R4.A=R1.A ∧ R6.A = R2.B)
)



Cartesian product: R'(x,y,z,w)= R(x,y), R(y,z), R(z,w)?
can be expressed in guarded fragment of FOL (with negation)?
But single join already not guarded

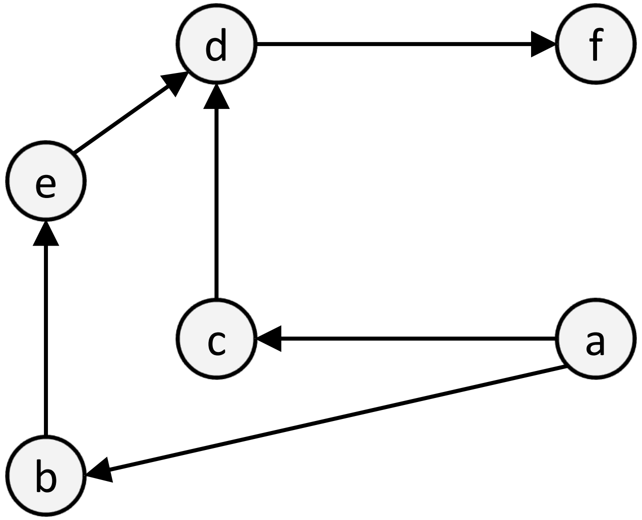
See Barany, Cate, Segoufin, "Guarded negatation", JACM 2015

guardedness

Exercise

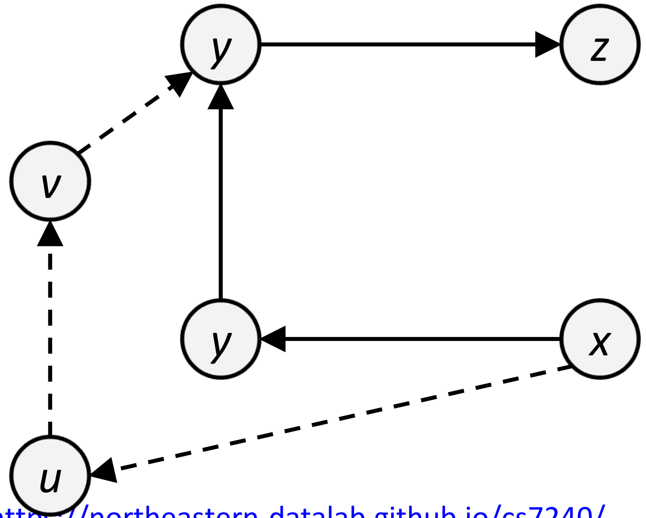
Schema: R(A,B)

Database *D*



Does the query below evaluate to true on above database?

Query *q*

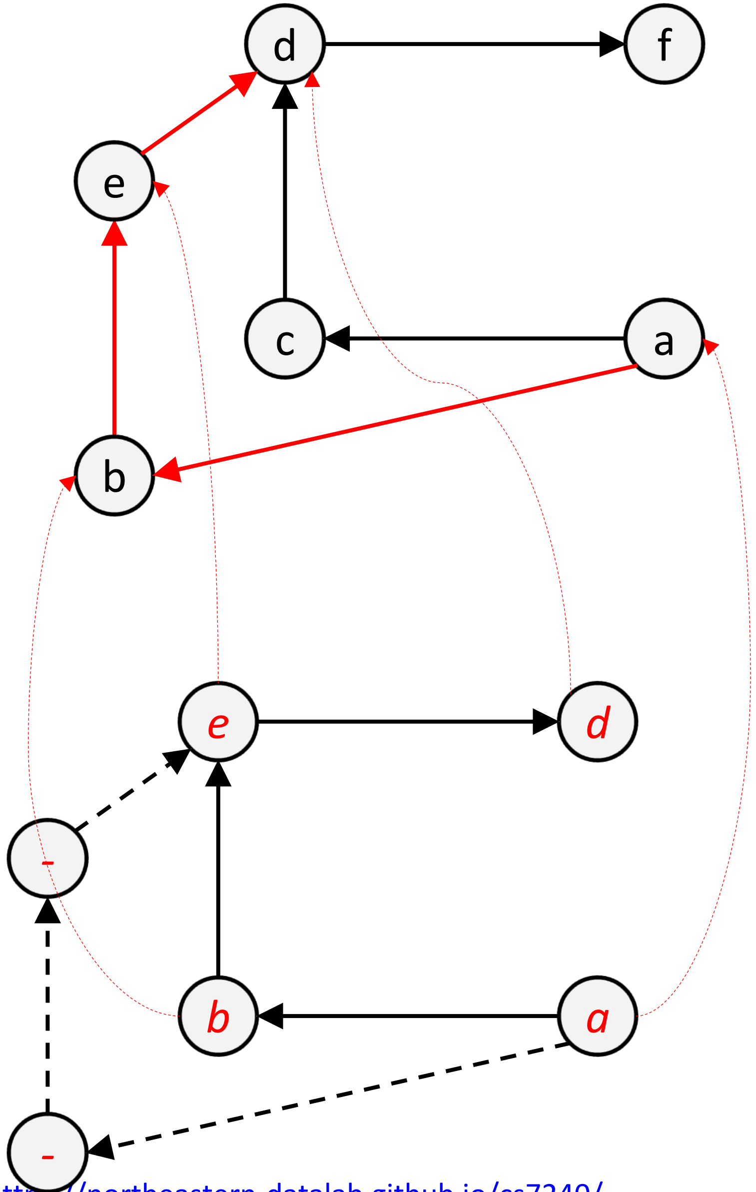


Exercise

Schema: R(A,B)

Database *D*

Query *q*



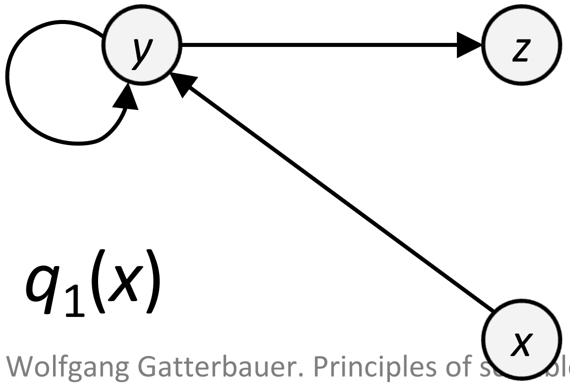
Question

- Find two such nested queries (somehow leveraging the example below) that are equivalent (based on some simple reasoning)
- What is then the *structured* procedure to prove equivalence?

Example

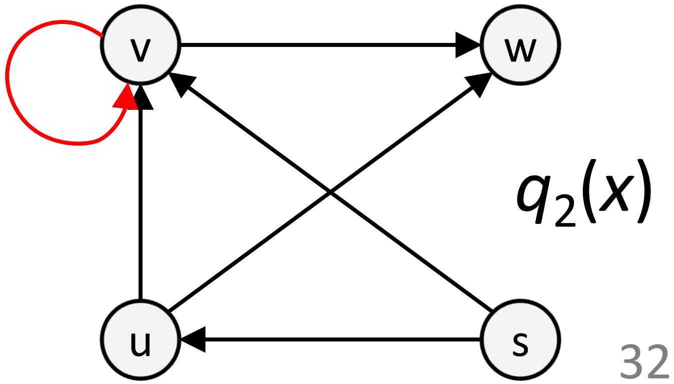
$q_1(x)$:- $R(x,y), R(y,y), R(y,z)$

$q_2(s)$:- $R(s,u), R(u,w), R(s,v), R(u,w), R(u,v)$, $R(v,v)$



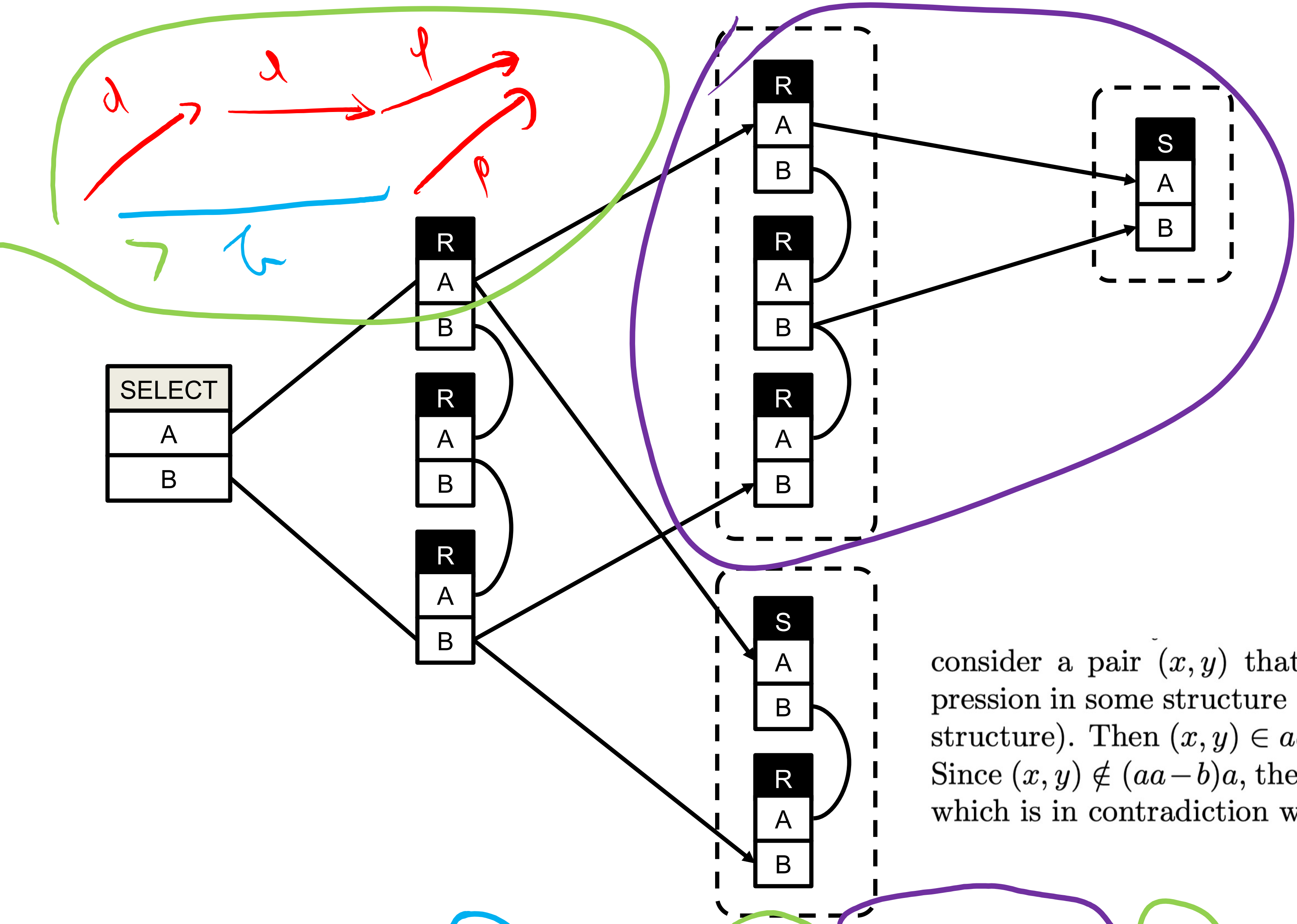
~~$h_{1\to 2}$: {(x,s),(y,v),(z,w)}~~ $q_1 \not\subseteq q_2$

$h_{2\to 1}$: {(s,x),(u,y),(v,y),(w,z)} $q_1 \subseteq q_2$



Undecidability ☹

- Unfortunately, the following problem is already undecidable
  - Consider the class of nested queries with maximal nesting level 2, no disjunctions, our safety restrictions from earlier, set semantics, arbitrary number of siblings
  - Deciding whether any given query is finitely satisfiable is undecidable.
- This follows non-trivially from from following Arxiv paper:
  - "**Undecidability of satisfiability in the algebra of finite binary relations with union, composition, and difference**" by Tony Tan, Jan Van den Bussche, Xiaowang Zhang, Corr 1406.0349. https://arxiv.org/abs/1406.0349



$$= aaa - (aa - b)a - ba$$

$$= aef - (ae - b)f - bf$$

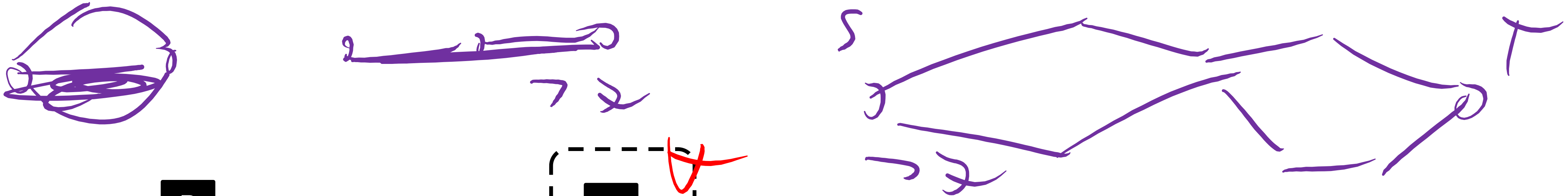
$$= aef - aef \cup bf - bf$$

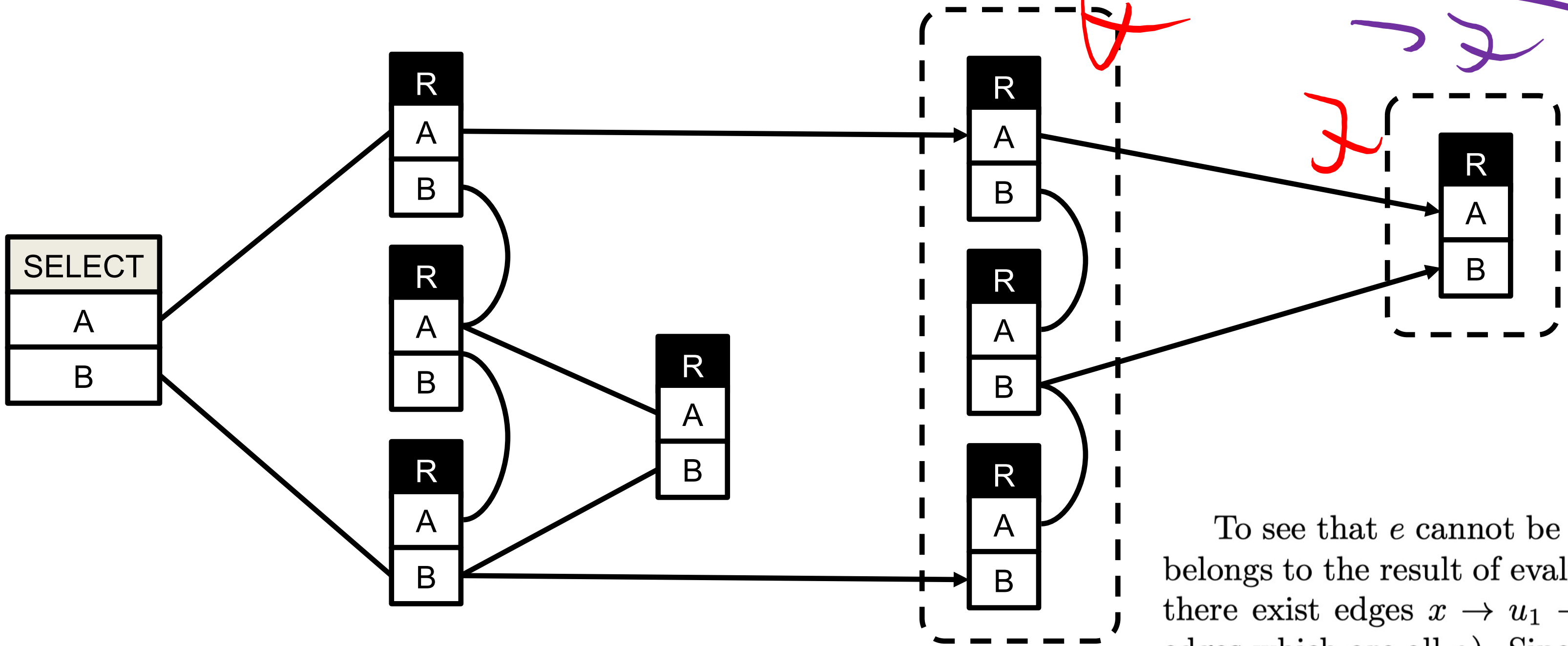
consider a pair (x, y) that would belong to the result of evaluating this expression in some structure (for brevity we are omitting explicit reference to this structure). Then $(x, y) \in aaa$ so there exist a-edges (x, x_1), (x_1, x_2), and (x_2, y). Since $(x, y) \notin (aa - b)a$, the b-edge (x, x_2) must be present. But then $(x, y) \in ba$, which is in contradiction with the last part of the expression. □

$$aaa - ((aa - b)a \cup ba) = aaa - (aa - b)a - ba$$

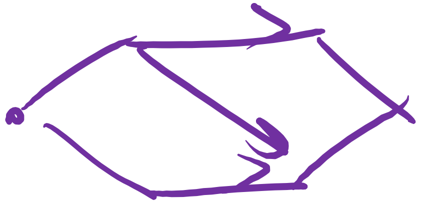
$$X - (Y \cup Z) = X - Y - Z$$

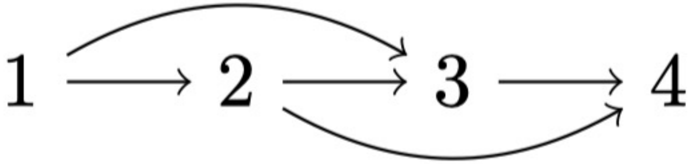
See "Undecidability of satisfiability in the algebra of finite binary relations with union, composition, and difference" by Tan, Van den Bussche, Zhang. https://arxiv.org/abs/1406.0349

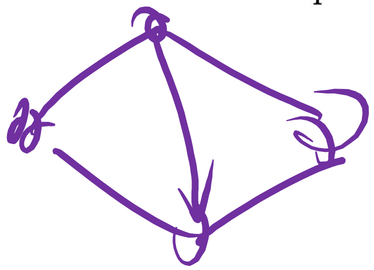




To see that e cannot be satisfied by any series-parallel graph, suppose (x, y) belongs to the result of evaluating e on some structure. Since $(x, y) \in a(a \cap aa)$, there exist edges $x \to u_1 \to u_2 \to y$ and $u_1 \to y$ (we omit the labels on the edges which are all a). Since $(x, y) \notin (aa - a)a$, there must be an edge $x \to u_2$. If at least two of the four elements x, u_1, u_2 and y are identical, the graph contains a cycle and is not series-parallel. If all four elements are distinct, we have a subgraph isomorphic to W above, so the structure is not series-parallel







$a(aa \cap a) - (aa - a)a$

See "Undecidability of satisfiability in the algebra of finite binary relations with union, composition, and difference" by Tan, Van den Bussche, Zhang. https://arxiv.org/abs/1406.0349

Open question

(SIBLINGS) ~ OUTDEGREE

NESTING	1	2	3+
0	CQ	—	—
1			?
2			
3+			