Efficient Approximations of Conjunctive Queries

Pablo Barceló (University of Chile)
Leonid Libkin (University of Edinburgh)
Miguel Romero (University of Chile)

CQ evaluation

- Conjunctive queries (CQs) – the best studied class of database queries
  - capture select-project-join queries

CQ evaluation

- Conjunctive queries (CQs) – the best studied class of database queries
  - capture select-project-join queries
- Complexity of evaluation:
  - data: very low (in AC^0)
  - combined: NP-complete
  - algorithmically: $|\text{database}|^{O(|\text{query}|)}$

CQ evaluation

- Conjunctive queries (CQs) – the best studied class of database queries
  - capture select-project-join queries
- Complexity of evaluation:
  - data: very low (in AC^0)
  - combined: NP-complete
  - algorithmically: $|\text{database}|^{O(|\text{query}|)}$
- Prohibitively expensive for very large databases
  - new applications: scientific databases, social networks, etc may store up to several terabytes of information.

CQ evaluation

- Conjunctive queries (CQs) – the best studied class of database queries
  - capture select-project-join queries
- Complexity of evaluation:
  - data: very low (in AC^0)
  - combined: NP-complete
  - algorithmically: $|\text{database}|^{O(|\text{query}|)}$
- Prohibitively expensive for very large databases
  - new applications: scientific databases, social networks, etc may store up to several terabytes of information.
- What do we do when we cannot find an exact solution?

CQ evaluation

- Conjunctive queries (CQs) – the best studied class of database queries
  - capture select-project-join queries
- Complexity of evaluation:
  - data: very low (in AC^0)
  - combined: NP-complete
  - algorithmically: $|\text{database}|^{O(|\text{query}|)}$
- Prohibitively expensive for very large databases
  - new applications: scientific databases, social networks, etc may store up to several terabytes of information.
- What do we do when we cannot find an exact solution?
- APPROXIMATE!

Approximation idea, by picture

SLOW QUERY Q

Approximation idea, by picture

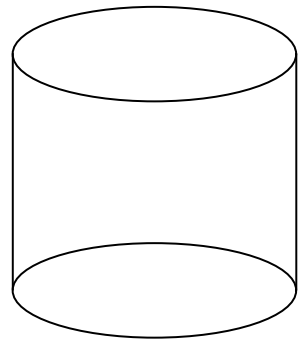
SLOW QUERY Q

FAST QUERY Q'

Approximation idea, by picture

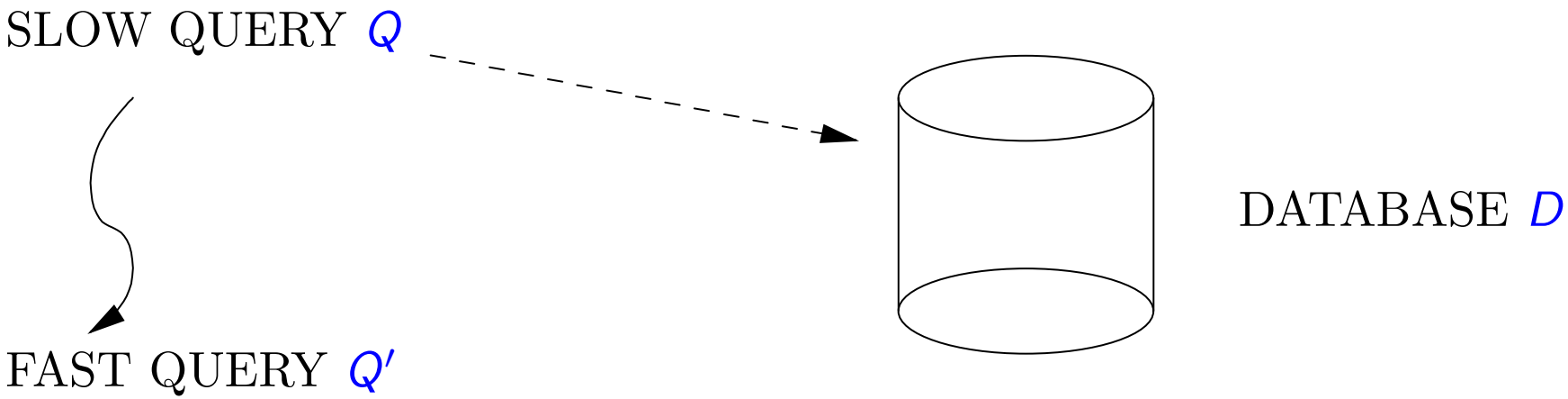
SLOW QUERY Q

FAST QUERY Q'

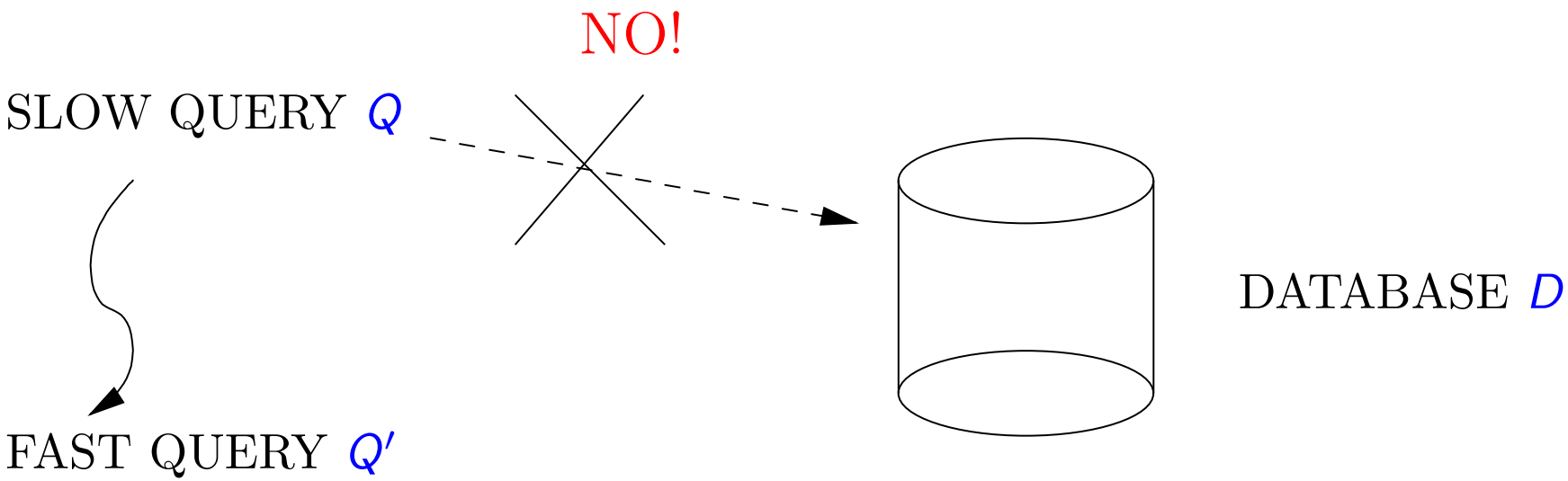


DATABASE D

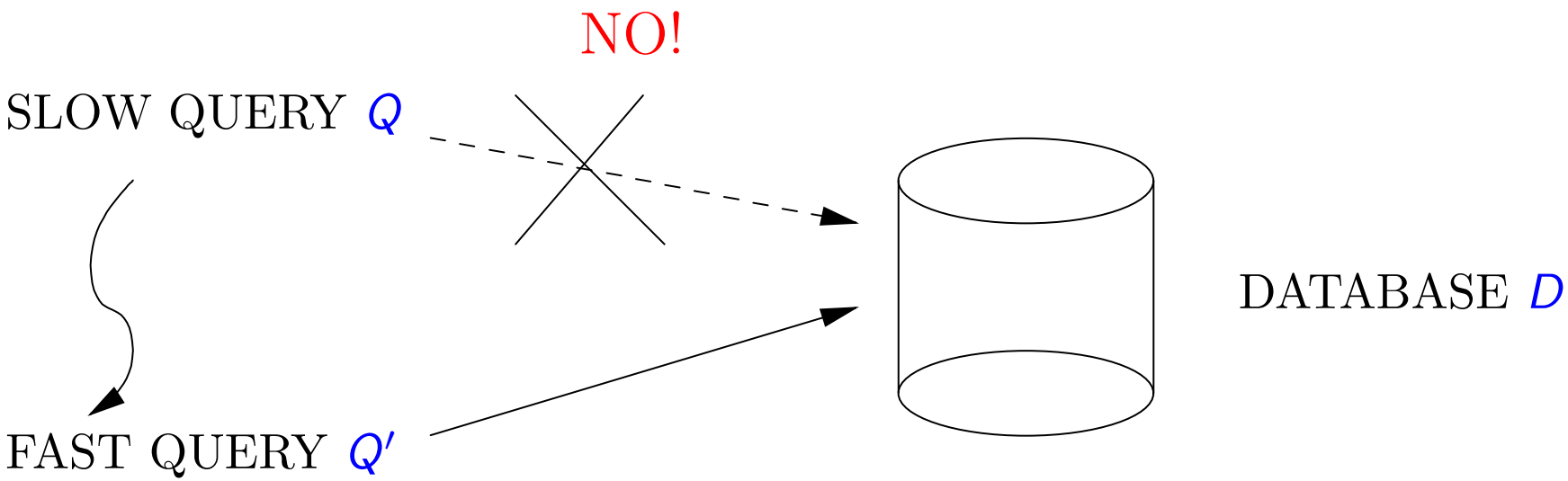
Approximation idea, by picture



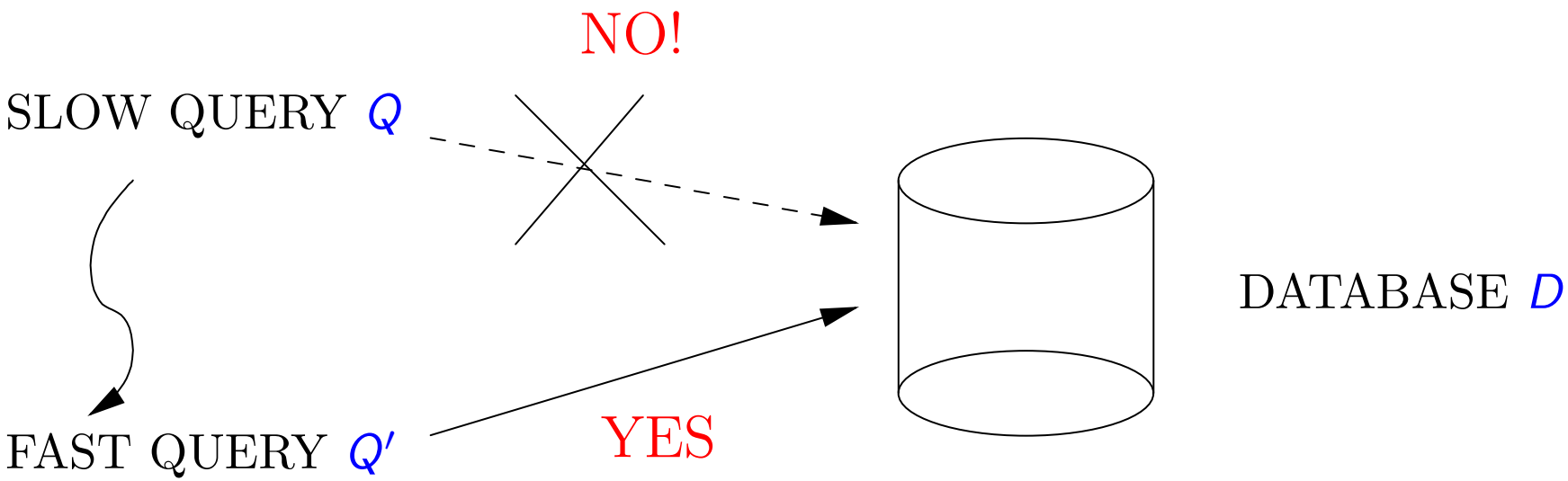
Approximation idea, by picture



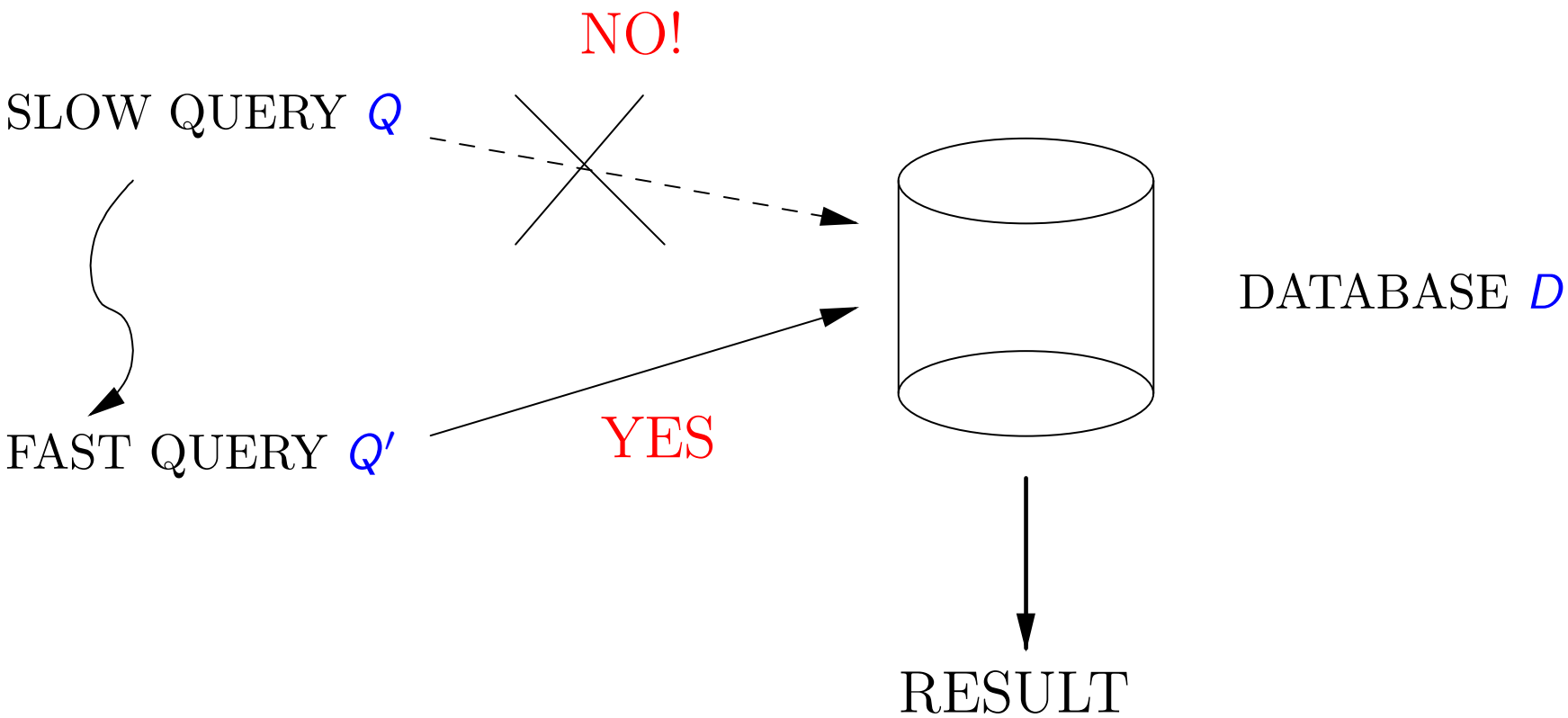
Approximation idea, by picture



Approximation idea, by picture



Approximation idea, by picture



Approximation techniques

The idea is, of course, quite old.

Standard approaches include:

- Statistical methods, e.g. histograms, combined with data mining techniques (Ioannidis ICDT 2003 survey + many others).
  - uses both data and query
- Semantic relaxations based on traditional data management techniques (Fan ICDT 2012 survey).
  - uses only the query (static analysis)

Approximation techniques

The idea is, of course, quite old.

Standard approaches include:

- Statistical methods, e.g. histograms, combined with data mining techniques (Ioannidis ICDT 2003 survey + many others).
  - uses both data and query
- Semantic relaxations based on traditional data management techniques (Fan ICDT 2012 survey).
  - uses only the query (static analysis)

Our choice: static analysis.
Once a query is approximated, it can be run and re-run when updates are applied.

Desiderata for approximations

- They must be fast
- They must be close to the queries they approximate

Next: formulate these two requirements for conjunctive queries.

Conjunctive queries (CQs)

$$\underbrace{Q(\bar{x})}_{\textit{head}} \quad :- \quad \underbrace{S_1(\bar{u}_1), \ldots, S_n(\bar{u}_n)}_{\textit{body}}$$

- atoms $S_i(\bar{u}_i)$: each S_i is a relation symbol, each $\bar{u}_i$ a list of variables.
- variables $\bar{y}$ in the body but not in the head are existentially quantified:
  $Q(\bar{x}) \;=\; \exists \bar{y}\, \big(S_1(\bar{u}_1) \wedge \cdots \wedge S_n(\bar{u}_n)\big)$.

Conjunctive queries (CQs)

$$\underbrace{Q(\bar{x})}_{\textit{head}} \quad :- \quad \underbrace{S_1(\bar{u}_1), \ldots, S_n(\bar{u}_n)}_{\textit{body}}$$

- atoms $S_i(\bar{u}_i)$: each S_i is a relation symbol, each $\bar{u}_i$ a list of variables.
- variables $\bar{y}$ in the body but not in the head are existentially quantified:
  $Q(\bar{x}) \;=\; \exists \bar{y}\, \big(S_1(\bar{u}_1) \wedge \cdots \wedge S_n(\bar{u}_n)\big)$.
- Why CQs?
  - a very important and common class of queries
  - equivalent to select-project-join queries
  - we know a lot about their evaluation.

Tableaux, evaluation, containment

$$Q(\bar{x}) \quad :- \quad S_1(\bar{u}_1), \ldots S_n(\bar{u}_n)$$

Its tableaux is the body of Q viewed as a database:

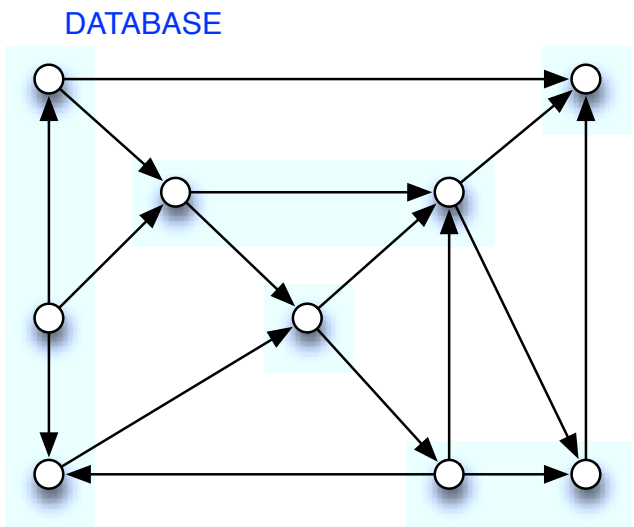
$$T_Q \quad = \quad (\{S_1(\bar{u}_1), \ldots, S_n(\bar{u}_n)\}, \quad \bar{x})$$

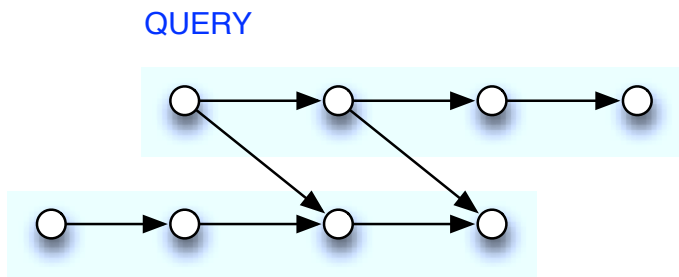
Evaluation and static analysis via tableaux and homomorphisms:

- $\bar{a} \in Q(D) \Leftrightarrow$ there is a homomorphism $T_Q \to (D, \bar{a})$
- $Q \subseteq Q' \Leftrightarrow$ there is a homomorphism $T_{Q'} \to T_Q$

CQ evaluation, graphically

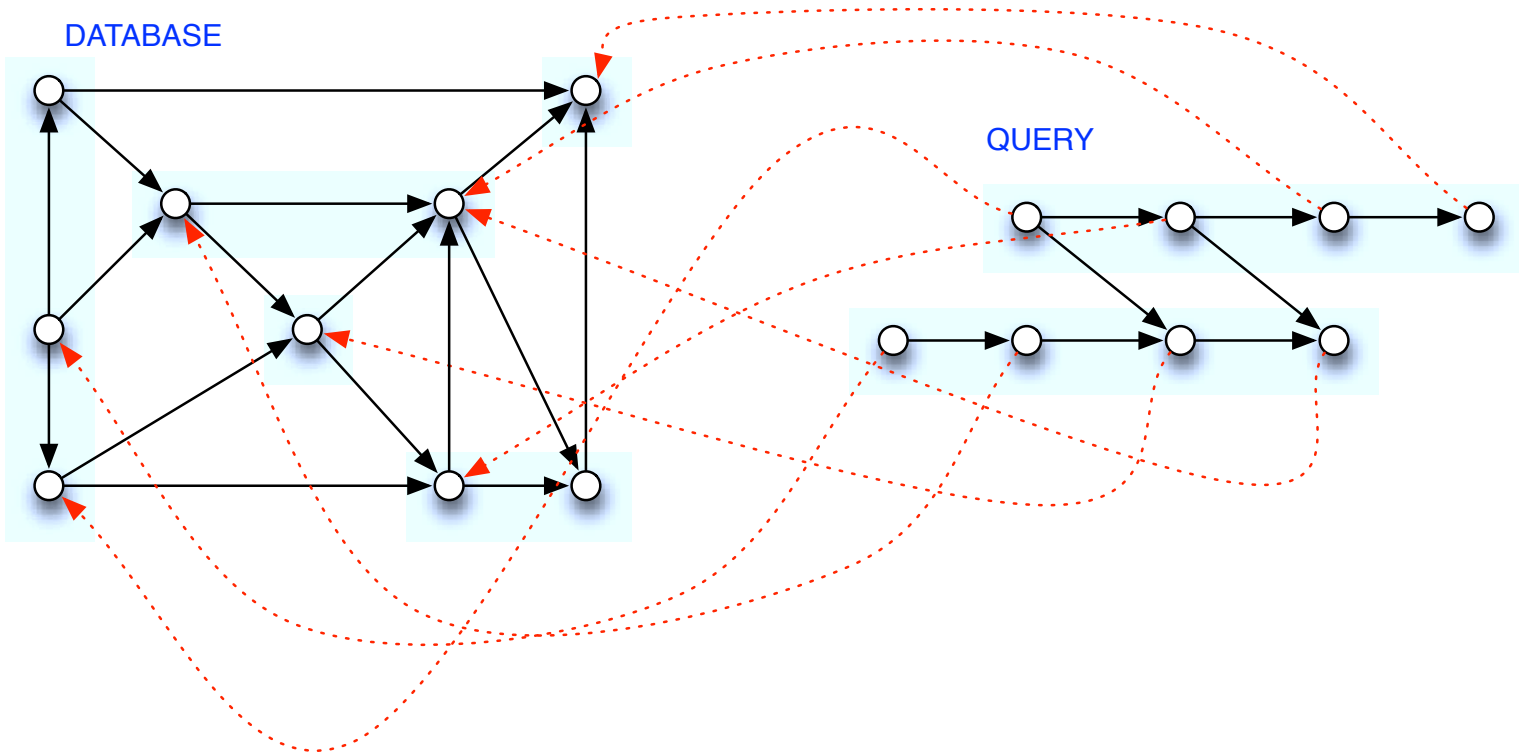
Goal: Find a homomorphism from T_Q into D.





CQ evaluation, graphically

Goal: Find a homomorphism from T_Q into D.

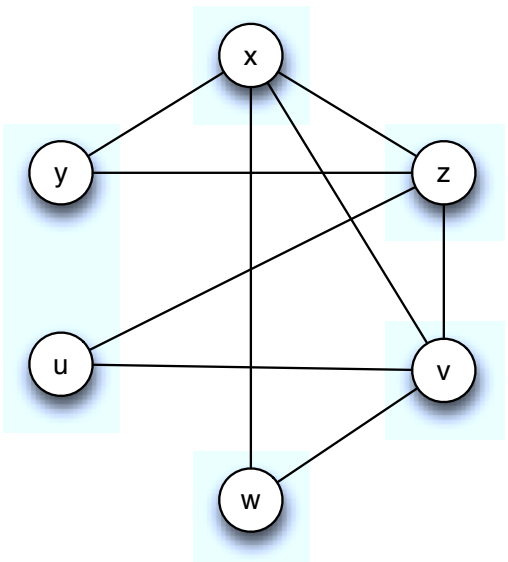


Good classes of CQs

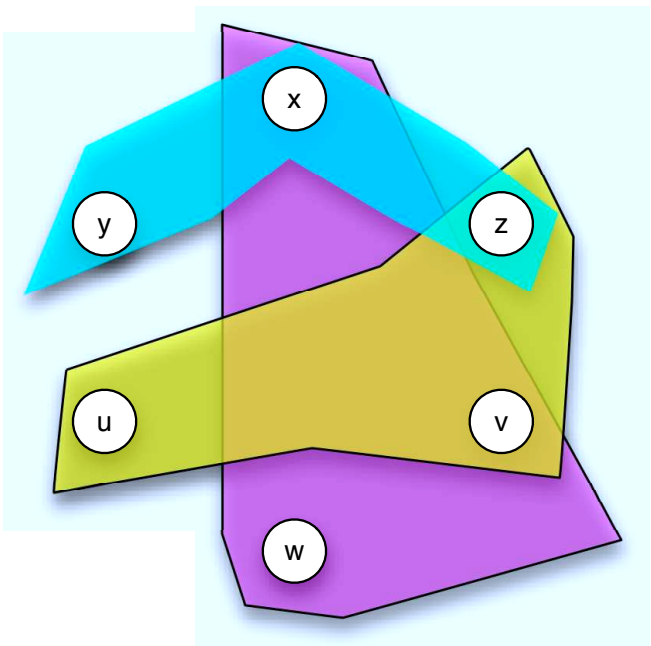
Two kinds: graph-based and hypergraph-based

$$Q \;:-\; R(x,y,z), R(z,u,v), R(v,w,x)$$

graph of the query, $G(Q)$



hypergraph of the query, $H(Q)$



Good classes of CQs – acyclicity and relatives

A general idea of tractable restrictions for CQs: acyclicity
(Yannakakis 1981 – linear-time evaluation)

Extensions:

- for graph-based notions: bounded treewidth
  - (Chekuri, Rajaraman, Kolaitis, Vardi, Grohe, Flum, Segoufin, Schwentick)
- for hypergraph-based notions: bounded hypertree width
  - (Gottlob, Leone, Scarcello)

Good classes of CQs – acyclicity and relatives

A general idea of tractable restrictions for CQs: acyclicity
(Yannakakis 1981 – linear-time evaluation)

Extensions:

- for graph-based notions: bounded treewidth
  - (Chekuri, Rajaraman, Kolaitis, Vardi, Grohe, Flum, Segoufin, Schwentick)
- for hypergraph-based notions: bounded hypertree width
  - (Gottlob, Leone, Scarcello)
- Yannakakis' notion of acyclicity is actually hypertree width 1.

Good classes of CQs – acyclicity and relatives

A general idea of tractable restrictions for CQs: acyclicity
(Yannakakis 1981 – linear-time evaluation)

Extensions:

- for graph-based notions: bounded treewidth
  - (Chekuri, Rajaraman, Kolaitis, Vardi, Grohe, Flum, Segoufin, Schwentick)
- for hypergraph-based notions: bounded hypertree width
  - (Gottlob, Leone, Scarcello)
- Yannakakis' notion of acyclicity is actually hypertree width 1.
- For queries on graphs, treewidth 1 $=$ acyclicity.

Good classes of CQs – acyclicity and relatives

A general idea of tractable restrictions for CQs: acyclicity
(Yannakakis 1981 – linear-time evaluation)

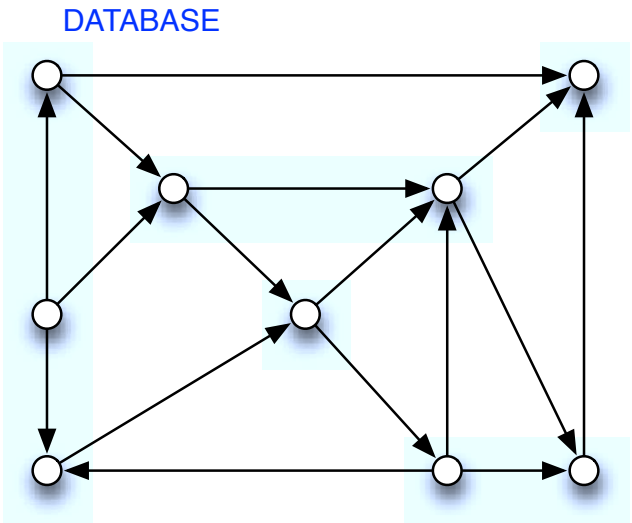
Extensions:

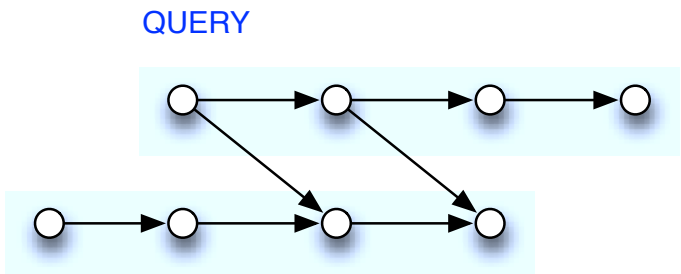
- for graph-based notions: bounded treewidth
  - (Chekuri, Rajaraman, Kolaitis, Vardi, Grohe, Flum, Segoufin, Schwentick)
- for hypergraph-based notions: bounded hypertree width
  - (Gottlob, Leone, Scarcello)
- Yannakakis' notion of acyclicity is actually hypertree width 1.
- For queries on graphs, treewidth 1 $=$ acyclicity.

Revised goal: approximate within tractable classes of CQs

Approximating CQs, graphically

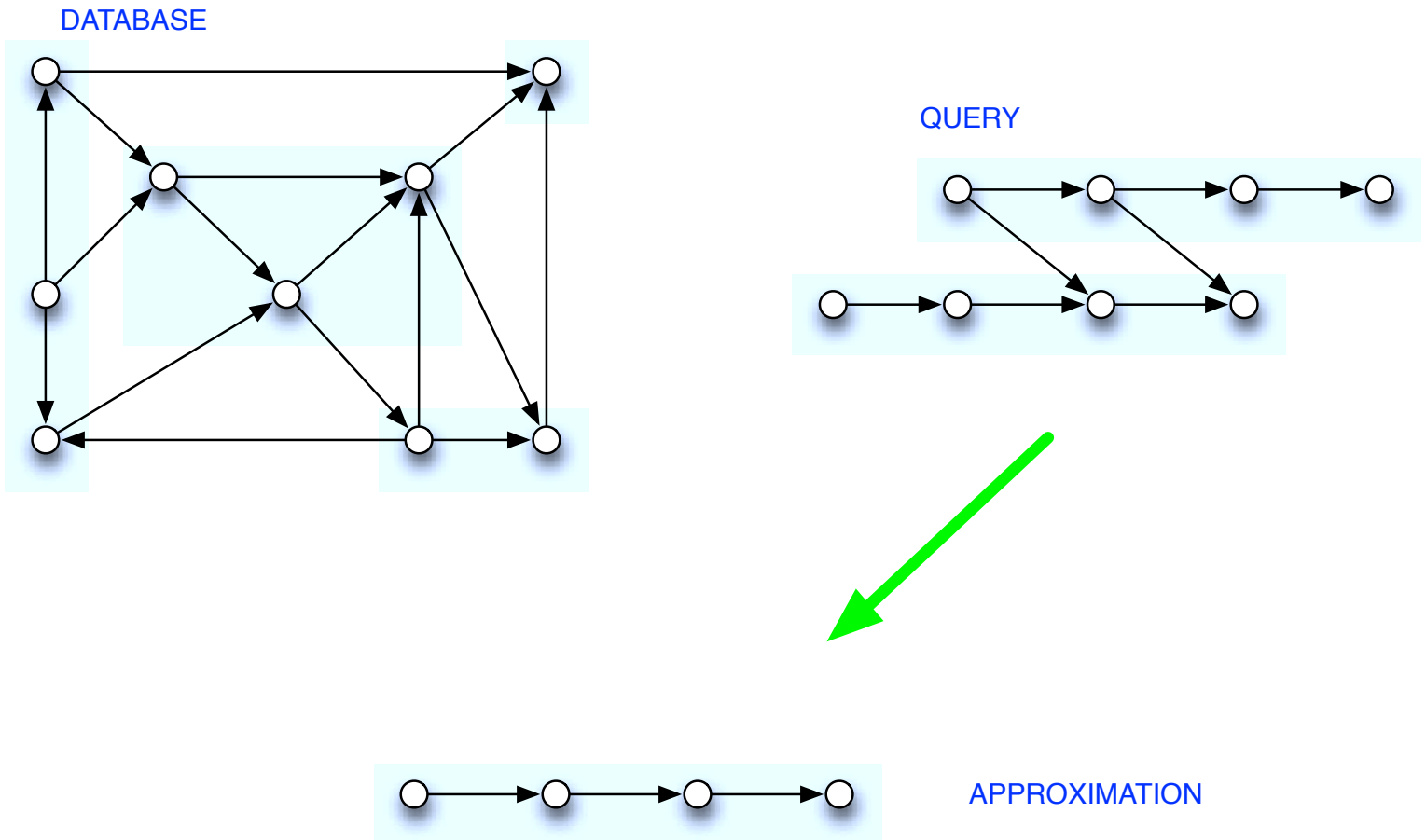
Given Q and D





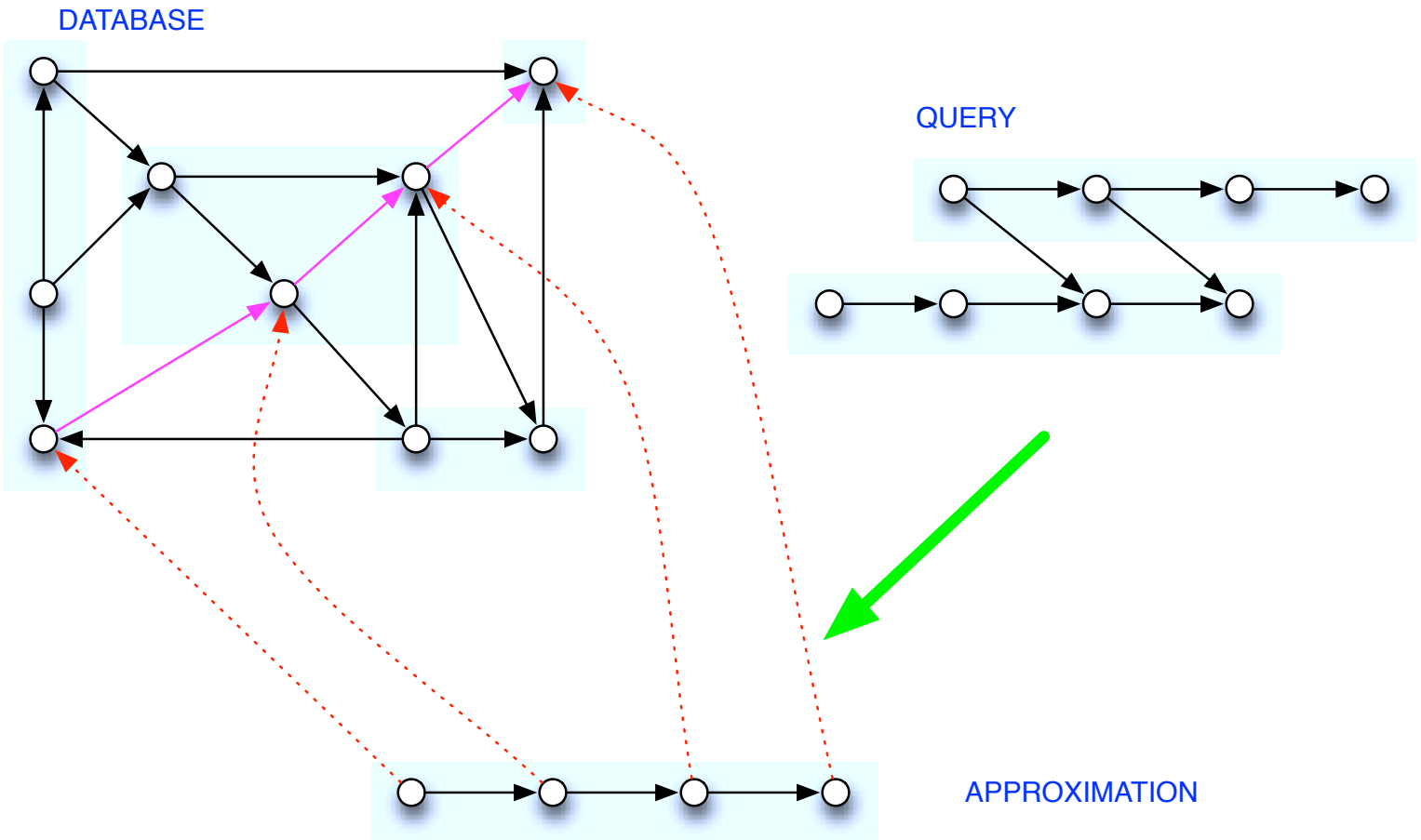
Approximating CQs, graphically

Given Q and D, find a tractable approximation



Approximating CQs, graphically

Given Q and D, find a tractable approximation and evaluate on D



Approximations: complexity analysis

- Evaluate Q on D: $|D|^{O(|Q|)}$

Approximations: complexity analysis

- Evaluate Q on D: $|D|^{O(|Q|)}$
- Evaluate approximation Q' on D:

$$O\big(\text{time to compute } Q' \;+\; |D|^c \cdot p(|Q'|)\big)$$

  - c is constant; p is polynomial

Approximations: complexity analysis

- Evaluate Q on D: $|D|^{O(|Q|)}$
- Evaluate approximation Q' on D:

$$O\big(\text{time to compute } Q' \;+\; |D|^c \cdot p(|Q'|)\big)$$

  - c is constant; p is polynomial
- Desiderata:
  - Q' at most polynomial in Q
  - time to compute Q' at most single-exponential in Q
  - hard to hope for more given the complexity of static analysis for CQs

Approximations: complexity analysis

- Evaluate Q on D: $|D|^{O(|Q|)}$
- Evaluate approximation Q' on D:

$$O\big(\text{time to compute } Q' \;+\; |D|^c \cdot p(|Q'|)\big)$$

  - c is constant; p is polynomial
- Desiderata:
  - Q' at most polynomial in Q
  - time to compute Q' at most single-exponential in Q
  - hard to hope for more given the complexity of static analysis for CQs
- For example, using acyclic approximations we have complexity $2^{O(|Q|\cdot\log|Q|)} + |D| \cdot |Q|^k$
- Much faster than $|D|^{O(|Q|)}$ on large databases.

CQ approximations: Definition

We want to approximate within a class $\mathcal{C}$ of good queries.

$$Q' \sqsubset_Q Q''$$

means: Q'' disagrees with Q less often than Q' disagrees with Q.

CQ approximations: Definition

We want to approximate within a class $\mathcal{C}$ of good queries.

$$Q' \sqsubset_Q Q''$$

means: Q'' disagrees with Q less often than Q' disagrees with Q.

We also want to approximate without producing false results:

if Q' approximates Q then $Q' \subseteq Q$.

CQ approximations: Definition

We want to approximate within a class $\mathcal{C}$ of good queries.

$$Q' \sqsubset_Q Q''$$

means: Q'' disagrees with Q less often than Q' disagrees with Q.

We also want to approximate without producing false results:

$$\text{if } Q' \text{ approximates } Q \text{ then } Q' \subseteq Q.$$

Definition A query $Q' \in \mathcal{C}$ is a $\mathcal{C}$-approximation of Q if:

- $Q' \subseteq Q$; and
- there is no $Q'' \in \mathcal{C}$ such that $Q' \sqsubset_Q Q''$.

The order $\sqsubset_Q$

Boolean version: $Q' \sqsubset_Q Q''$ if

The order $\sqsubset_Q$

Boolean version: $Q' \sqsubseteq_Q Q''$ if

- for every database D,

The order $\sqsubset_Q$

Boolean version: $Q' \sqsubseteq_Q Q''$ if

- for every database D,
- if Q and Q' agree on D (i.e., $D \models Q \leftrightarrow Q'$),

The order $\sqsubset_Q$

Boolean version: $Q' \sqsubseteq_Q Q''$ if

- for every database D,
- if Q and Q' agree on D (i.e., $D \models Q \leftrightarrow Q'$),
- then Q and Q'' agree on D (i.e., $D \models Q \leftrightarrow Q''$).

The order $\sqsubseteq_Q$

Boolean version: $Q' \sqsubseteq_Q Q''$ if

- for every database D,
- if Q and Q' agree on D (i.e., $D \models Q \leftrightarrow Q'$),
- then Q and Q'' agree on D (i.e., $D \models Q \leftrightarrow Q''$).
- General definition: replace D by $D, \bar{a}$, where $\bar{a}$ is a tuple of elements.

The order $\sqsubset_Q$

Boolean version: $Q' \sqsubseteq_Q Q''$ if

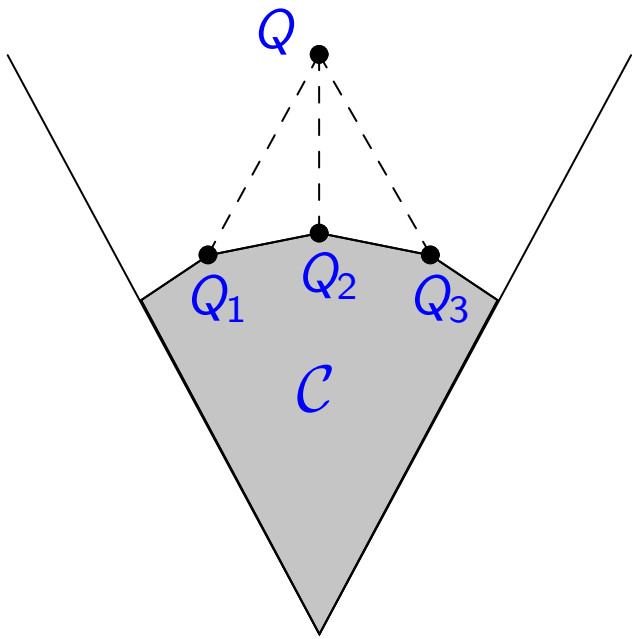
- for every database D,
- if Q and Q' agree on D (i.e., $D \models Q \leftrightarrow Q'$),
- then Q and Q'' agree on D (i.e., $D \models Q \leftrightarrow Q''$).
- General definition: replace D by $D, \bar{a}$, where $\bar{a}$ is a tuple of elements.

$$Q' \sqsubset_Q Q'' \quad \Leftrightarrow \quad Q' \sqsubseteq_Q Q'' \text{ and } Q'' \not\sqsubseteq_Q Q'.$$

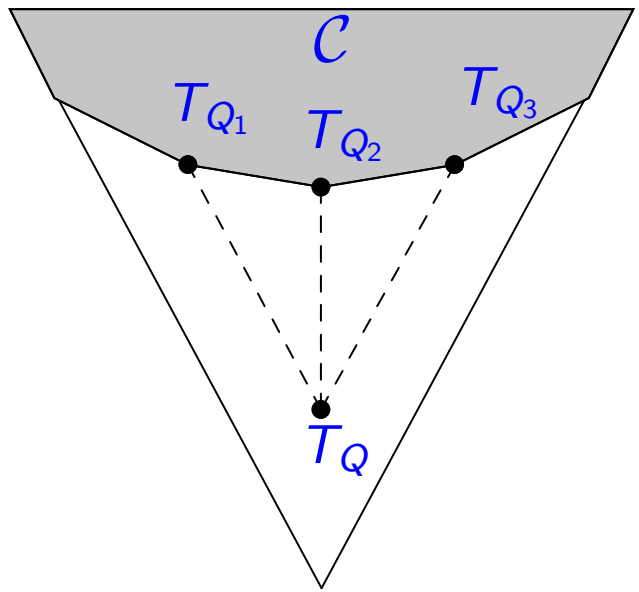
CQ approximations, graphically

Proposition

We can use the familiar containment/homomomorphism (pre)ordering instead of $\sqsubset$*:*



a query view:
based on containment



a tableau view:
based on homomorphism

Existence of approximations: graph-based case

A class $\mathcal{C}$ of queries is closed under substructures if:

$$Q \in \mathcal{C} \text{ and } T_{Q'} \subseteq T_Q \implies Q' \in \mathcal{C}$$

Example: classes of queries of bounded treewidth.

Existence of approximations: graph-based case

A class $\mathcal{C}$ of queries is closed under substructures if:

$$Q \in \mathcal{C} \text{ and } T_{Q'} \subseteq T_Q \implies Q' \in \mathcal{C}$$

Example: classes of queries of bounded treewidth.

A small assumption: the trivial query $Q :- R(x, x, \ldots, x)$ is in $\mathcal{C}$.

Does not hurt at all:

- this query is of treewidth 1 and acyclic, i.e. belongs to all tractable classes;
- it is also contained in every other CQ.

Existence theorem

Let $\mathcal{C}$ be closed under substructures.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*

Existence theorem

Let $\mathcal{C}$ be closed under substructures.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size does not exceed the size of Q.*

Existence theorem

Let $\mathcal{C}$ be closed under substructures.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size does not exceed the size of Q.*
- *An approximation can be found in single-exponential time*
  - $2^{O(n \log n)}$ *to be precise*

Existence theorem

Let $\mathcal{C}$ be closed under substructures.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size does not exceed the size of Q.*
- *An approximation can be found in single-exponential time*
  - $2^{O(n \log n)}$ *to be precise*
- *There are at most exponentially many non-equivalent approximations.*

Example

- Take

$$Q :- R(x, y, z),\ R(y, x, u),\ R(u, z, x)$$

- Its treewidth is 3 – maximum possible for a query with 4 variables; its graph is K_4.
- Has approximations of the smallest possible treewidth, i.e., 1:

$$Q' :- R(x, y, y),\ R(y, x, y),\ R(y, y, x).$$

- Good to reduce treewidth as it determines the exponent in query evaluation.

Bounds on static analysis

In general, exponential bounds are unavoidable for analysis of CQs (the most basic containment problem is NP-complete).

Exponential number of non-equivalent approximations is unavoidable too.

Bounds on static analysis

In general, exponential bounds are unavoidable for analysis of CQs (the most basic containment problem is NP-complete).

Exponential number of non-equivalent approximations is unavoidable too.

Theorem

There is a sequence of conjunctive queries $Q_1, Q_2, \ldots, Q_n, \ldots$ *such that:*

- *the size of* Q_n *grows linearly in* n*, and*
- *each* Q_n *has at least* 2^n *non-equivalent approximations of treewidth* 1*.*

Bounds on static analysis

In general, exponential bounds are unavoidable for analysis of CQs (the most basic containment problem is NP-complete).

Exponential number of non-equivalent approximations is unavoidable too.

Theorem

There is a sequence of conjunctive queries $Q_1, Q_2, \ldots, Q_n, \ldots$ such that:

- *the size of Q_n grows linearly in n, and*
- *each Q_n has at least 2^n non-equivalent approximations of treewidth 1.*

Q_ns can be Boolean CQs on graphs, so the bound applies to acyclic approximations as well.

Complexity of approximations: decision version

Input:	A CQ Q and another CQ $Q' \in \mathcal{C}$.
Question:	is Q' a $\mathcal{C}$-approximation of Q?

Complexity of approximations: decision version

Input:	A CQ Q and another CQ $Q' \in \mathcal{C}$.
Question:	is Q' a $\mathcal{C}$-approximation of Q?

Theorem

Let $\mathcal{C}$ be the class of queries of treewidth k, for any $k \geq 1$. Then this problem is DP-complete, even if both Q and Q' are minimized.

Complexity of approximations: decision version

Input:	A CQ Q and another CQ $Q' \in \mathcal{C}$.
Question:	is Q' a $\mathcal{C}$-approximation of Q?

Theorem

Let $\mathcal{C}$ be the class of queries of treewidth k, for any $k \geq 1$. Then this problem is DP-complete, even if both Q and Q' are minimized.

Remarks:

- Class DP: slightly above NP and coNP (intersection of an NP problem and a coNP problem).

Complexity of approximations: decision version

Input:	A CQ Q and another CQ $Q' \in \mathcal{C}$.
Question:	is Q' a $\mathcal{C}$-approximation of Q?

Theorem

Let $\mathcal{C}$ be the class of queries of treewidth k, for any $k \geq 1$. Then this problem is DP-complete, even if both Q and Q' are minimized.

Remarks:

- Class DP: slightly above NP and coNP (intersection of an NP problem and a coNP problem).
- Hardness proven for Boolean CQs on graphs.

Complexity of approximations: decision version

Input:	A CQ Q and another CQ $Q' \in \mathcal{C}$.
Question:	is Q' a $\mathcal{C}$-approximation of Q?

Theorem

Let $\mathcal{C}$ be the class of queries of treewidth k, for any $k \geq 1$. Then this problem is DP-complete, even if both Q and Q' are minimized.

Remarks:

- Class DP: slightly above NP and coNP (intersection of an NP problem and a coNP problem).
- Hardness proven for Boolean CQs on graphs.
- Nontrivial reduction from the problem whether a graph is 4- but not 3-colorable (Riege, Rothe, 2006).

Hypergraph-based classes: acyclicity

A CQ Q is acyclic if its hypergraph $H(Q)$ is acyclic.

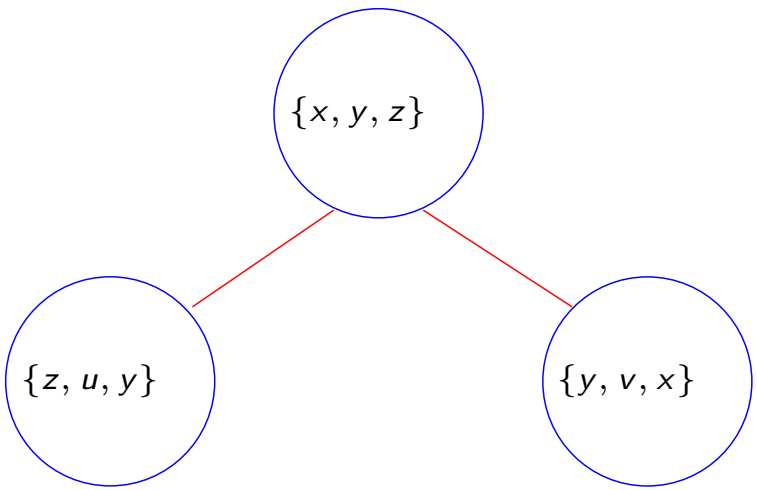
- has a tree decomposition in which ever block is a hyperedge

Hypergraph-based classes: acyclicity

A CQ Q is acyclic if its hypergraph $H(Q)$ is acyclic.

- has a tree decomposition in which ever block is a hyperedge

Example: Decomposition for the query
$Q :- R(x, y, z), R(z, u, y), R(y, v, x)$:



Acyclicity and hypertreewidth

Theorem (Yannakakis 1981)

If Q is an acyclic conjunctive query, then checking whether $\bar{a} \in Q(D)$ can be done in time $O(|D| \cdot |Q|)$.

Acyclicity and hypertreewidth

Theorem (Yannakakis 1981)

If Q is an acyclic conjunctive query, then checking whether $\bar{a} \in Q(D)$ can be done in time $O(|D| \cdot |Q|)$.

Generalization: hypertree width (Gottlob, Leone, Scarcello, 2000).

- Extends the notion of treewidth.
- Acyclicity = hypertree width 1.
- Guaranteed tractable evaluation of CQs:
  - Hypertree width determines the exponent.
  - $O(|D|^c \cdot p(|Q|))$.

Existence of approximation: closure conditions

Problem: previous existence conditions don't work – even acyclic hypergraphs are not closed under taking sub-hypergraphs.

Existence of approximation: closure conditions

Problem: previous existence conditions don't work – even acyclic hypergraphs are not closed under taking sub-hypergraphs.

Instead, we use two new closure conditions:

Existence of approximation: closure conditions

Problem: previous existence conditions don't work – even acyclic hypergraphs are not closed under taking sub-hypergraphs.

Instead, we use two new closure conditions:

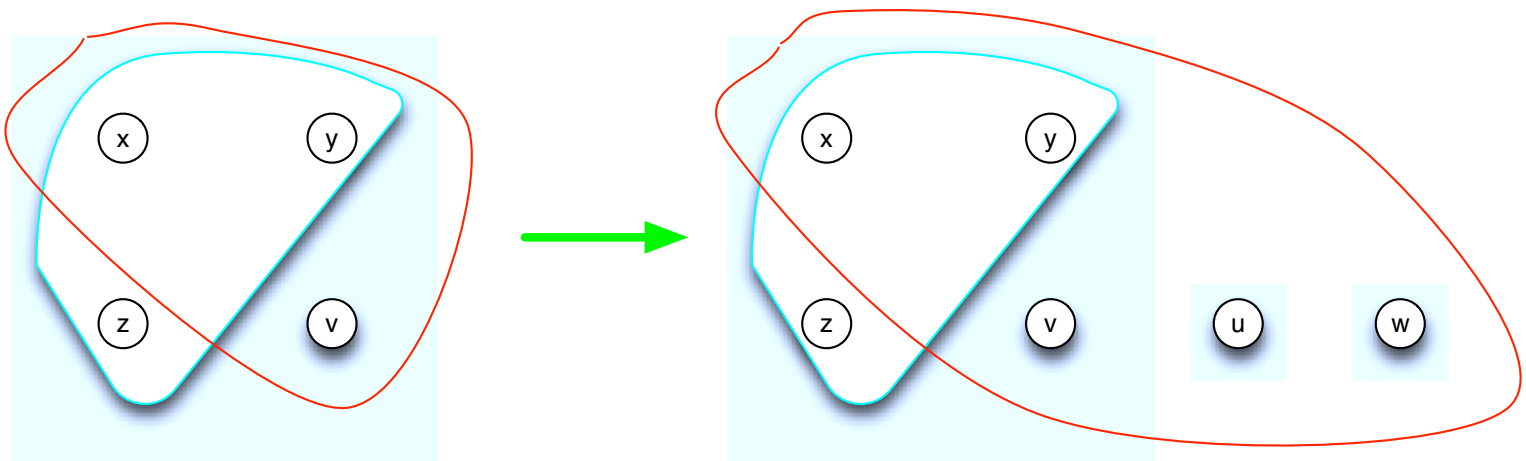
1. Closure under induced sub-hypergraphs;

Existence of approximation: closure conditions

Problem: previous existence conditions don't work – even acyclic hypergraphs are not closed under taking sub-hypergraphs.

Instead, we use two new closure conditions:

1. Closure under induced sub-hypergraphs;
2. Closure under edge extensions:



New closure conditions and hypertreewidth

The closure conditions are good for us:

Proposition

The class of hypergraphs of hypertree width $\leq k$, for every $k \geq 1$, is closed under both induced subhypergraphs and hyperedge extensions.

In particular, the class of acyclic hypergraphs satisfies these closure conditions.

Existence of approximations for hypergraph-based classes

$\mathcal{C}$ – a class of CQs whose hypergraphs are closed under induced subhypergraphs and edge extensions:

- for example, queries of hypertree width $\leq k$, or
- acyclic queries.

Existence of approximations for hypergraph-based classes

$\mathcal{C}$ – a class of CQs whose hypergraphs are closed under induced subhypergraphs and edge extensions:

- for example, queries of hypertree width $\leq k$, or
- acyclic queries.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*

Existence of approximations for hypergraph-based classes

$\mathcal{C}$ – a class of CQs whose hypergraphs are closed under induced subhypergraphs and edge extensions:

- for example, queries of hypertree width $\leq k$, or
- acyclic queries.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size is at most polynomial in the size of Q.*

Existence of approximations for hypergraph-based classes

$\mathcal{C}$ – a class of CQs whose hypergraphs are closed under induced subhypergraphs and edge extensions:

- for example, queries of hypertree width $\leq k$, or
- acyclic queries.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size is at most polynomial in the size of Q.*
- *An approximation can be found in single-exponential time*
  - $2^{O(n \log n)}$ *to be precise*

Existence of approximations for hypergraph-based classes

$\mathcal{C}$ – a class of CQs whose hypergraphs are closed under induced subhypergraphs and edge extensions:

- for example, queries of hypertree width $\leq k$, or
- acyclic queries.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size is at most polynomial in the size of Q.*
- *An approximation can be found in single-exponential time*
  - $2^{O(n \log n)}$ *to be precise*
- *There are at most exponentially many non-equivalent approximations.*

Existence of approximations for hypergraph-based classes

$\mathcal{C}$ – a class of CQs whose hypergraphs are closed under induced subhypergraphs and edge extensions:

- for example, queries of hypertree width $\leq k$, or
- acyclic queries.

Theorem

- *Every conjunctive query Q has a $\mathcal{C}$-approximation.*
- *Each approximation is (equivalent to) a query whose size is at most polynomial in the size of Q.*
- *An approximation can be found in single-exponential time*
  - $2^{O(n \log n)}$ *to be precise*
- *There are at most exponentially many non-equivalent approximations.*

All lower bounds shown for graphs, hence apply here.

Example of acyclic approximations

A cyclic query:

$$Q \text{ :- } R(x,y,z),\ R(z,u,v),\ R(v,w,x)$$

Example of acyclic approximations

A cyclic query:

$$Q \ :- \ R(x,y,z), \ R(z,u,v), \ R(v,w,x)$$

It has several acyclic approximations:

Example of acyclic approximations

A cyclic query:

$$Q \ :- \ R(x,y,z), \ R(z,u,v), \ R(v,w,x)$$

It has several acyclic approximations:

$$Q_1 \ :- \ R(x,y,z), \ R(z,u,y), \ R(y,v,x)$$

Example of acyclic approximations

A cyclic query:

$$Q \ :- \ R(x,y,z),\ R(z,u,v),\ R(v,w,x)$$

It has several acyclic approximations:

$$Q_1 \ :- \ R(x,y,z),\ R(z,u,y),\ R(y,v,x)$$

$$Q_2 \ :- \ R(x,y,z),\ R(z,u,v),\ R(v,w,x),\ R(x,z,v)$$

Example of acyclic approximations

A cyclic query:

$$Q \ :- \ R(x,y,z),\ R(z,u,v),\ R(v,w,x)$$

It has several acyclic approximations:

$$Q_1 \ :- \ R(x,y,z),\ R(z,u,y),\ R(y,v,x)$$

$$Q_2 \ :- \ R(x,y,z),\ R(z,u,v),\ R(v,w,x),\ R(x,z,v)$$

$$Q_3 \ :- \ R(x,y,x)$$

Future work

- Quantitative analysis
  - no fake results like: if Q' is a $\mathcal{C}$-approximation of Q, then with probability 1 they agree on a randomly chosen database (under the uniform distribution)
- More relaxed approximations: do not insist on $Q' \subseteq Q$
- Analyze generalized hypertree width as a means of getting good approximations.