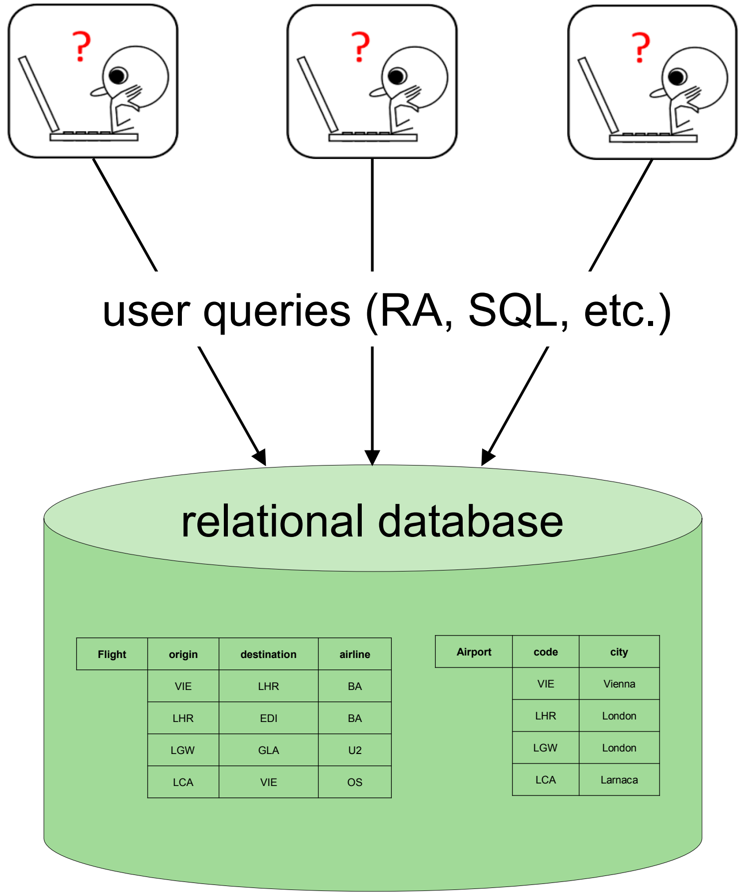


Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca

a standard database system

user queries (RA, SQL, etc.)

…but, we live in the era of big data

Volume
size does mattes
(thousands of TBs of data)

Veracity
data is often
incomplete/inconsistent

Variety
many data formats
(structured, semi-structured, etc.)

Velocity
data often arrives at fast speed
(updates are frequent)

the rest of this course

Volume
size does mattes
(thousands of TBs of data)

Veracity
data is often
incomplete/inconsistent

Variety
many data formats
(structured, semi-structured, etc.)

Velocity
data often arrives at fast speed
(updates are frequent)

Approximation of Conjunctive Queries

A Plausible Approach

…to address the challenges raised by the volume of big data

replace the query with one that is much faster to execute!!!

Minimizing Conjunctive Queries

- Database theory has developed principled methods for optimizing CQs:
  - Find an equivalent CQ with minimal number of atoms (the core)
  - Provides a notion of “true” optimality

Q(x) :- R(x,y), R(x,b), R(a,b), R(u,c), R(u,v), S(a,c,d)

$\{y \rightarrow b\}$

Q(x) :- R(x,b), R(a,b), R(u,c), R(u,v), S(a,c,d)

$\{v \rightarrow c\}$

Q(x) :- R(x,b), R(a,b), R(u,c), S(a,c,d)

minimal query

Minimizing Conjunctive Queries

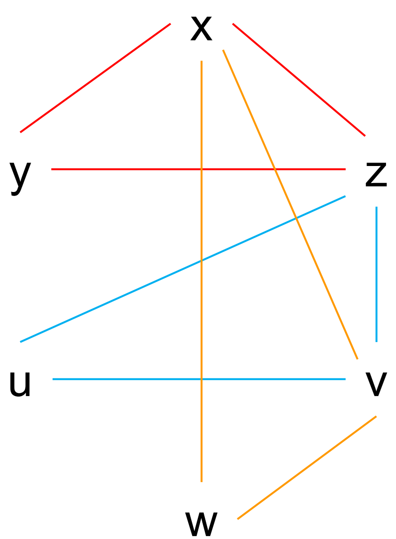
- But, a minimal equivalent CQ might not be easier to evaluate – query evaluation remains NP-hard

- However, we know “good” classes of CQs for which query evaluation is tractable (in combined complexity):
  - Graph-based
  - Hypergraph-based

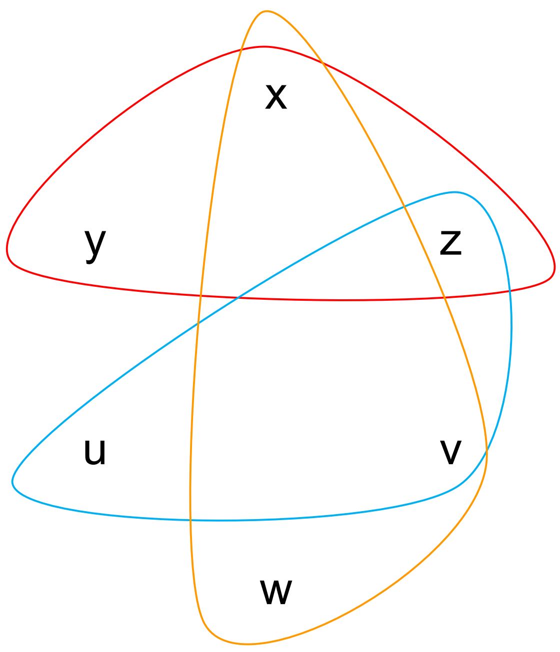
(Hyper)graph of Conjunctive Queries

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

graph of Q - $G(Q)$



hypergraph of Q - $H(Q)$



“Good” Classes of Conjunctive Queries

measures how close a graph is to a tree

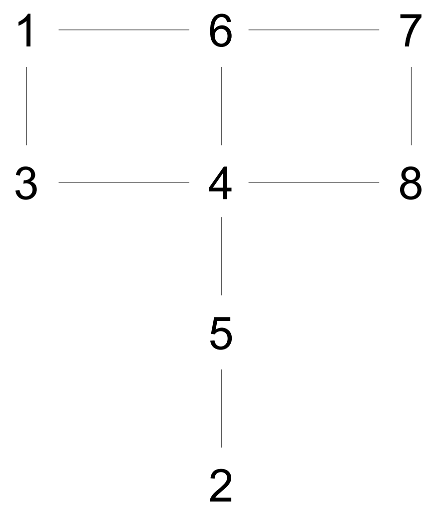
- Graph-based
  - CQs of bounded treewidth – their graph has bounded treewidth

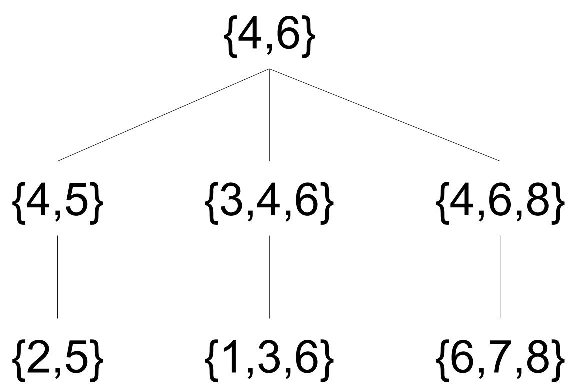
measures how close a hypergraph is to an acyclic one

- Hypergraph-based:
  - CQs of bounded hypertree width – their hypergraph has bounded hypertree width
  - Acyclic CQs – their hypegraph has hypertree width 1

Treewidth of a Graph

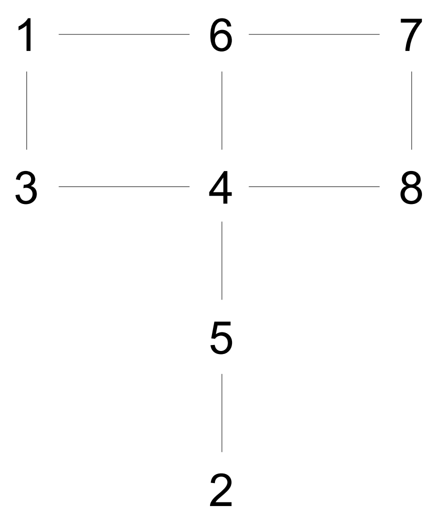
- A tree decomposition of a graph **G** = (V,E) is a labeled tree **T** = (N,F,λ), where λ : N $\rightarrow$ 2^V such that:
  1. For each node u $\in$ V of **G**, there exists n $\in$ N such that u $\in$ λ(n)
  2. For each edge (u,v) $\in$ E, there exists n $\in$ N such that {u,v} $\subseteq$ λ(n)
  3. For each node u $\in$ V of **G**, the set {n $\in$ N | u $\in$ λ(n)} induces a *connected* subtree of **T**

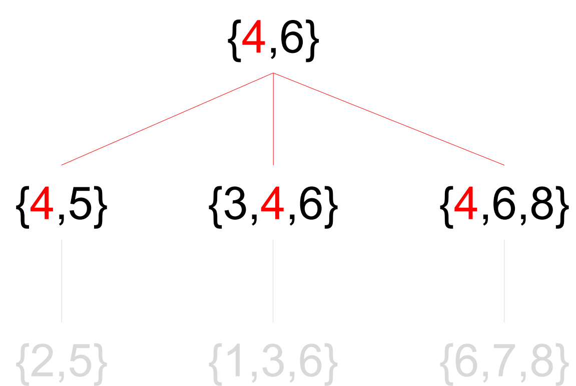




Treewidth of a Graph

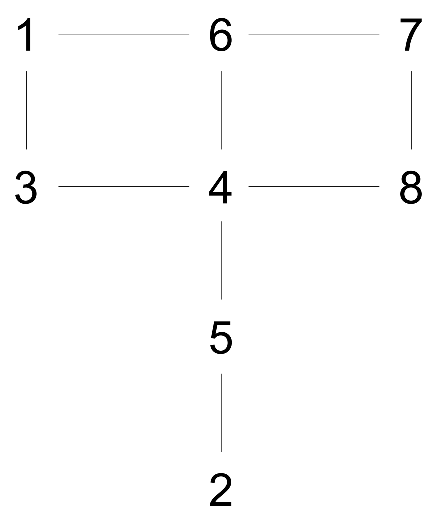
- A tree decomposition of a graph **G** = (V,E) is a labeled tree **T** = (N,F,λ), where $\lambda : N \to 2^V$ such that:
  1. For each node $u \in V$ of **G**, there exists $n \in N$ such that $u \in \lambda(n)$
  2. For each edge $(u,v) \in E$, there exists $n \in N$ such that $\{u,v\} \subseteq \lambda(n)$
  3. For each node $u \in V$ of **G**, the set $\{n \in N \mid u \in \lambda(n)\}$ induces a *connected* subtree of **T**

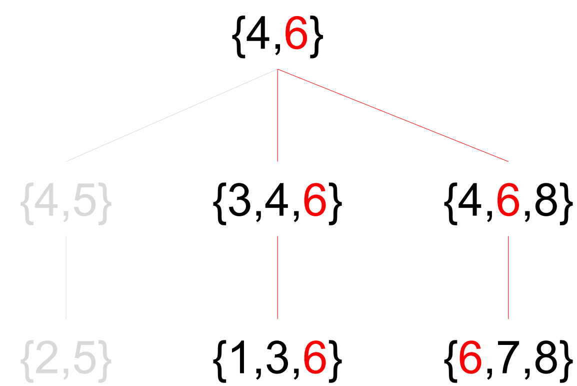




Treewidth of a Graph

- A tree decomposition of a graph **G** = (V,E) is a labeled tree **T** = (N,F,λ), where λ : N → 2^V such that:
  1. For each node u ∈ V of **G**, there exists n ∈ N such that u ∈ λ(n)
  2. For each edge (u,v) ∈ E, there exists n ∈ N such that {u,v} ⊆ λ(n)
  3. For each node u ∈ V of **G**, the set {n ∈ N | u ∈ λ(n)} induces a *connected* subtree of **T**





Treewidth of a Graph

- A tree decomposition of a graph **G** = (V,E) is a labeled tree **T** = (N,F,λ), where λ : N $\rightarrow$ 2^V such that:
  1. For each node u $\in$ V of **G**, there exists n $\in$ N such that u $\in$ λ(n)
  2. For each edge (u,v) $\in$ E, there exists n $\in$ N such that {u,v} $\subseteq$ λ(n)
  3. For each node u $\in$ V of **G**, the set {n $\in$ N | u $\in$ λ(n)} induces a *connected* subtree of **T**

- The width of a tree decomposition **T** = (N,F,λ) is $\max_{n \in N} \{|\lambda(n)| - 1\}$

-1 so that the treewidth of a tree is 1

- The treewidth of **G** is the minimum width over all tree decompositions of **G**

CQs of Bounded Treewidth

Theorem: For a fixed k ≥ 0, BQE($\mathbf{CQTW_k}$) is in PTIME

{$Q \in$ **CQ** | the treewidth of $G(Q)$ is at most k}

Actually, if $G(Q)$ has treewidth k ≥ 0, then Q can be evaluated in time

$O(|D|^k)$ + time to compute a tree decomposition for $G(Q)$ of optimal width,

which is feasible in linear time

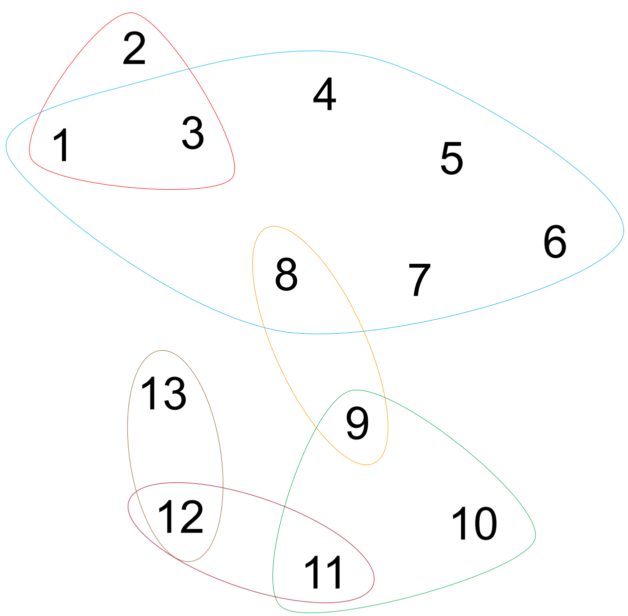
“Good” Classes of Conjunctive Queries

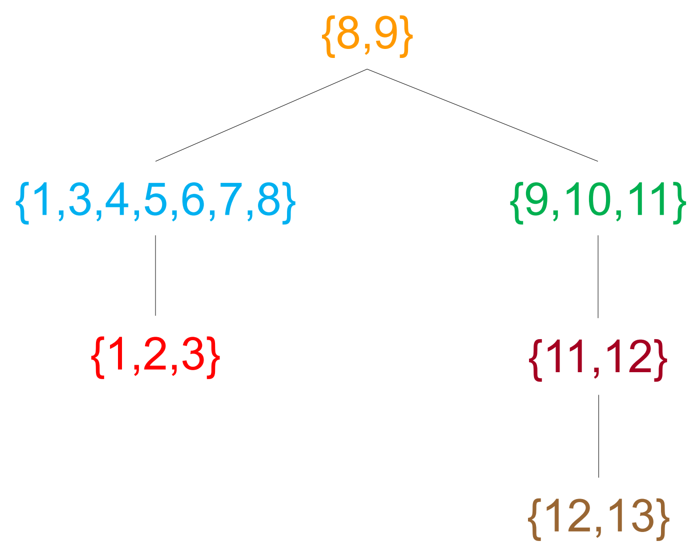
- Graph-based
  - CQs of bounded treewidth – their graph has bounded treewidth
    - Evaluation is feasible in polynomial time

- Hypergraph-based:
  - CQs of bounded hypertree width – their hypergraph has bounded hypertree width
  - Acyclic CQs – their hypegraph has hypertree width 1

Acyclic Hypergraphs

- A join tree of a hypergraph **H** = (V,E) is a labeled tree **T** = (N,F,λ), where λ : N → E such that:
  1. For each hyperedge e ∈ E of **H**, there exists n ∈ N such that e = λ(n)
  2. For each node u ∈ V of **H**, the set {n ∈ N | u ∈ λ(n)} induces a *connected* subtree of **T**

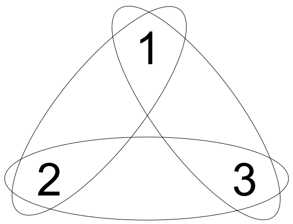




Acyclic Hypergraphs

- A join tree of a hypergraph **H** = (V,E) is a labeled tree **T** = (N,F,λ), where λ : N $\rightarrow$ E such that:
  1. For each hyperedge e $\in$ E of **H**, there exists n $\in$ N such that e = λ(n)
  2. For each node u $\in$ V of **H**, the set {n $\in$ N | u $\in$ λ(n)} induces a *connected* subtree of **T**

- **Definition:** A hypergraph is acyclic if it has a join tree

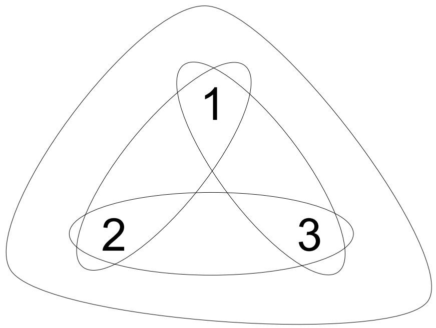


prime example of a cyclic hypergraph

Acyclic Hypergraphs

- A join tree of a hypergraph **H** = (V,E) is a labeled tree **T** = (N,F,λ), where λ : N → E such that:
  1. For each hyperedge e ∈ E of **H**, there exists n ∈ N such that e = λ(n)
  2. For each node u ∈ V of **H**, the set {n ∈ N | u ∈ λ(n)} induces a *connected* subtree of **T**

- **Definition:** A hypergraph is acyclic if it has a join tree



but this is acyclic

Acyclic CQs

Theorem: BQE(**ACQ**) is in PTIME

$\{Q \in \textbf{CQ} \mid H(Q) \text{ is acyclic}\}$

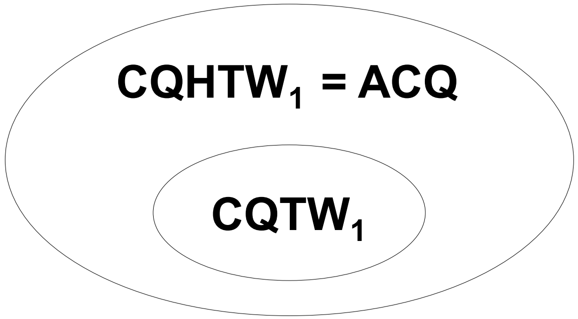
Actually, if $H(Q)$ is acyclic, then Q can be evaluated in time $O(|D| \cdot |Q|)$,

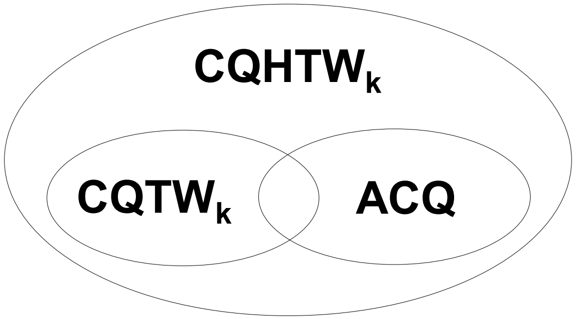
i.e., linear time in the size of D and Q

“Good” Classes of Conjunctive Queries: Recap

- Graph-based
  - CQs of bounded treewidth – their graph has bounded treewidth
    - Evaluation is feasible in polynomial time

- Hypergraph-based:
  - CQs of bounded hypertree width – their hypergraph has bounded hypertree width
    - Evaluation is feasible in polynomial time
  - Acyclic CQs – their hypegraph has hypertree width 1
    - Evaluation is feasible in linear time





Back to Our Goal

Replace a given CQ with one that is much faster to execute

or

Replace a given CQ with one that falls in “good” class of CQs

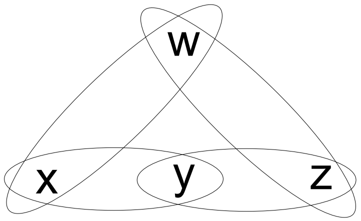
preferably, with an acyclic CQ

since evaluation is in linear time

Semantic Acyclicity

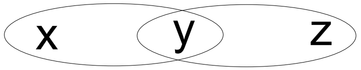
Definition: A CQ Q is semantically acyclic if there exists an acyclic CQ Q' such that $Q \equiv Q'$

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



$\{w \rightarrow y, z \rightarrow y\}$

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



Semantic Acyclicity

Theorem: A CQ Q is semantically acyclic iff its core is acyclic

Theorem: Deciding whether a CQ Q is semantically acyclic is NP-complete

Proof idea (upper bound):

- We can show the following: if Q is semantically acyclic, then there exists an acyclic CQ Q' such that $|Q'| \leq |Q|$ and $Q \equiv Q'$
- Then, we can guess in polynomial time:
  - An acyclic CQ Q' such that $|Q'| \leq |Q|$
  - A mapping h_1 : terms(Q) $\rightarrow$ terms(Q')
  - A mapping h_2 : terms(Q') $\rightarrow$ terms(Q)
- And verify in polynomial time that h_1 is a query homomorphism from Q to Q' (i.e., $Q' \subseteq Q$), and h_2 is a query homomorphism from Q' to Q (i.e., $Q \subseteq Q'$)

Semantic Acyclicity

Theorem: A CQ Q is semantically acyclic iff its core is acyclic

Theorem: Deciding whether a CQ Q is semantically acyclic is NP-complete

But, semantic acyclicity is rather *weak*:

- Not many CQs are semantically acyclic

  $\Rightarrow$ consider acyclic approximations of CQs

- Semantic acyclicity is not an improvement over usual optimization – both approaches are based on the core

  $\Rightarrow$ exploit semantic information in the form of constraints

Acyclic Approximations of CQs

Acyclic Approximations

If our CQ Q is not semantically acyclic, we may target a CQ that is:

1. Easy to evaluate – acyclic
2. Provides sound answers – contained in Q
3. As "informative" as possible – "maximally" contained in Q

Definition: A CQ Q' is an acyclic approximation of Q if:

1. Q' is acyclic
2. $Q' \subseteq Q$
3. There is no acyclic CQ Q'' such that $Q' \subset Q'' \subseteq Q$

Do Acyclic Approximations Exist?

The cyclic CQ

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

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)

Existence, Size and Computation

Theorem: Consider a CQ Q. Then:

1. Q has an acyclic approximation
2. Each acyclic approximation of Q has size polynomial in Q
3. An acyclic approximation of Q can be found in time $2^{O(|Q| \cdot \log |Q|)}$
4. Q has at most exponentially many (non-equivalent) acyclic approximations

Evaluating Acyclic Approximations

- Recall that evaluating Q over D takes time $|D|^{O(|Q|)}$

- Evaluating an acyclic approximation Q' of Q over D takes time

$$\underbrace{2^{O(|Q| \cdot \log |Q|)}}_{\text{time for computing } Q'} \; + \; \underbrace{|D| \cdot |Q|^k}_{\text{time for evaluating } Q'}$$

time for computing Q'

time for evaluating Q'

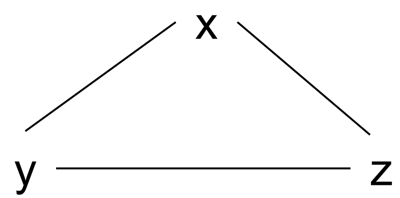
- $|Q'| \leq |Q|^k$
- Evaluation of an acyclic CQ Q_A is feasible in time $O(|D| \cdot |Q_A|)$

- Observe that $2^{O(|Q| \cdot \log |Q|)} + |D| \cdot |Q|^k$ is dominated by $|D| \cdot 2^{O(|Q| \cdot \log |Q|)}$

  $\Rightarrow$ fixed-parameter tractable

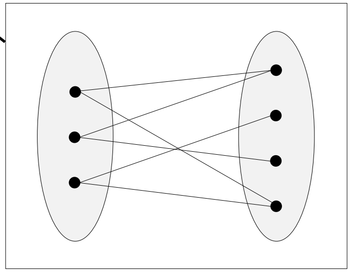
Poor Approximations

Q :- E(x,y), E(y,z), E(z,x)



has only one acyclic approximation, that is, Q' :- E(x,x)

Proposition: Consider a Boolean CQ Q that contains a single binary relation E(.,.). If *G*(Q) is not bipartite, then the only acyclic approximation of Q is Q' :- E(x,x)



Acyclic Approximations: Recap

- Acyclic approximations are useful when the CQ is not semantically acyclic
- Always exist, but are not unique
- Have polynomial size, and can be computed in exponential time
- Can be evaluated “efficiently” (fixed-parameter tractability)
- In some cases, acyclic approximations are not very informative

Back to Semantic Acyclicity

But, semantic acyclicity is rather *weak*:

- Not many CQs are semantically acyclic

  $\Rightarrow$ consider acyclic approximations of CQs ✓

- Semantic acyclicity is not an improvement over usual optimization – both approaches are based on the core

  $\Rightarrow$ exploit semantic information in the form of constraints

Associated Papers

- Pablo Barceló, Leonid Libkin, Miguel Romero: Efficient Approximations of Conjunctive Queries. SIAM J. Comput. 43(3): 1085-1130 (2014)

  Eligible topics include static analysis of approximations

- Pablo Barceló, Miguel Romero, Moshe Y. Vardi: Semantic Acyclicity on Graph Databases. SIAM J. Comput. 45(4): 1339-1376 (2016)

  Semantic acyclicity for CQs

- Hubie Chen, Víctor Dalmau: Beyond Hypertree Width: Decomposition Methods Without Decompositions. CP 2005: 167-181

  Complexity of semantic acyclicity for CQs (in a different context)

- Víctor Dalmau, Phokion G. Kolaitis, Moshe Y. Vardi: Constraint Satisfaction, Bounded Treewidth, and Finite-Variable Logics. CP 2002: 310-326

  Evaluation of semantically acyclic CQ (in a different context)

Associated Papers

- Joerg Flum, Martin Grohe: Fixed-Parameter Tractability, Definability, and Model-Checking. SIAM J. Comput. 31(1): 113-145 (2001)

  A different way of measuring complexity, and its full analysis

- Joerg Flum, Markus Frick, Martin Grohe: Query evaluation via tree- decompositions. Journal of the ACM 49(6): 716-752 (2002)

  Using tree decompositions to get faster query evaluation

- Markus Frick, Martin Grohe: Deciding first-order properties of locally tree-decomposable structures. Journal of the ACM 48(6): 1184-1206 (2001)

  How to improve performance of relational queries on databases with special properties

Associated Papers

- Georg Gottlob, Nicola Leone, Francesco Scarcello: The complexity of acyclic conjunctive queries. Journal of the ACM 48(3):431-498 (2001)

  An in-depth study of acyclicity

- Georg Gottlob, Nicola Leone, Francesco Scarcello: Hypertree Decompositions and Tractable Queries. J. Comput. Syst. Sci. 64(3):579-627 (2002)

  A hierarchy of classes of efficient CQs, the bottom level of which is acyclic queries

- Martin Grohe, Thomas Schwentick, Luc Segoufin: When is the evaluation of conjunctive queries tractable? STOC 2001: 657-666

  Characterizing efficiency of CQs via the notion of bounded treewidth

- Mihalis Yannakakis: Algorithms for Acyclic Database Schemes. VLDB 1981: 82-94

  Notion of acyclicity of CQs and fast evaluation scheme based on it