Semantic Optimization of Conjunctive Queries
Semantic Acyclicity

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

But, semantic acyclicity is rather weak:

- Not many CQs are semantically acyclic
  ⇒ consider acyclic approximations of CQs

- Semantic acyclicity is not an improvement over usual optimization – both approaches are based on the core
  ⇒ exploit semantic information in the form of constraints
Constraints Enrich Semantic Acyclicity

\[ Q \text{ :- } R(x,y), R(y,z), R(z,x) \]

- Assume that \( Q \) will be evaluated over databases that comply with the following set of inclusion dependencies

\[
\begin{align*}
R[1,2] & \subseteq P[1,2] \iff \forall x \forall y \ (R(x,y) \rightarrow \exists z \ P(x,y,z)) \\
P[2,3] & \subseteq R[1,2] \iff \forall x \forall y \forall z \ (P(x,y,z) \rightarrow R(y,z)) \\
P[3,1] & \subseteq R[1,2] \iff \forall x \forall y \forall z \ (P(x,y,z) \rightarrow R(z,x))
\end{align*}
\]

- Then \( Q \) can be replaced by

\[ Q' \text{ :- } R(x,y) \]
Constraints Enrich Semantic Acyclicity

\[ Q \ :- \ R(x,y), R(y,z), R(z,x) \]

- Assume that \( Q \) will be evaluated over databases that comply with the following set of inclusion dependencies

\[
R[1,2] \subseteq P[1,2] \ \equiv \ \forall x \forall y (R(x,y) \rightarrow \exists z P(x,y,z))
\]

\[
P[2,3] \subseteq R[1,2] \ \equiv \ \forall x \forall y \forall z (P(x,y,z) \rightarrow R(y,z))
\]

\[
P[3,1] \subseteq R[1,2] \ \equiv \ \forall x \forall y \forall z (P(x,y,z) \rightarrow R(z,x))
\]

- Moreover, \( Q \) can be replaced by

\[ Q' \ :- \ R(x,y), R(y,z), R(z,x), P(x,y,z) \]
Constraints Enrich Semantic Acyclicity

\[ Q : R(x,y), R(y,z), R(z,x), R(x,z) \]

- Assume that \( Q \) will be evaluated over databases that comply with the following functional dependency

\[ R : \{1\} \rightarrow \{2\} \equiv \forall x \forall y \forall z \ (R(x,y) \land R(x,z) \rightarrow y = z) \]

- Then \( Q \) can be replaced by

\[ Q' : R(x,y), R(y,y), R(y,x) \]
Semantic Acyclicity Under Constraints

(henceforth, by set of constraints we mean a set of inclusion or functional dependencies)

**Definition:** Given a CQ $Q$ and a set of constraints $\Sigma$, we say that $Q$ is **semantically acyclic under** $\Sigma$ if there exists an acyclic CQ $Q'$ such that $Q \equiv_\Sigma Q'$

for every database $D$ that satisfies $\Sigma$, $Q(D) = Q'(D)$

(analogously, we define the notation $Q \subseteq_\Sigma Q'$)

in the above definition, we only care for the databases in the shaded part
Semantic Acyclicity Under Constraints

(henceforth, by set of constraints we mean a set of inclusion or functional dependencies)

**Definition:** Given a CQ $Q$ and a set of constraints $\Sigma$, we say that $Q$ is semantically acyclic under $\Sigma$ if there exists an acyclic CQ $Q'$ such that $Q \equiv_\Sigma Q'$ for every database $D$ that satisfies $\Sigma$, $Q(D) = Q'(D)$

(analogously, we define the notation $Q \subseteq_\Sigma Q'$)

**Two crucial questions:** given a CQ $Q$ and a set $\Sigma$ of constraints

1. Can we decide whether $Q$ is semantically acyclic under $\Sigma$, and what is the exact complexity?
2. Does this help query evaluation?
Semantic Acyclicity Under Constraints

(henceforth, by set of constraints we mean a set of inclusion or functional dependencies)

**Definition:** Given a CQ $Q$ and a set of constraints $\Sigma$, we say that $Q$ is semantically acyclic under $\Sigma$ if there exists an acyclic CQ $Q'$ such that $Q \equiv_\Sigma Q'$ for every database $D$ that satisfies $\Sigma$, $Q(D) = Q'(D)$

(analogously, we define the notation $Q \subseteq_\Sigma Q'$)

**Two crucial questions:** given a CQ $Q$ and a set $\Sigma$ of constraints
1. Can we decide whether $Q$ is semantically acyclic under $\Sigma$, and what is the exact complexity? *First, we need to understand CQ containment under constraints*
2. Does this help query evaluation?
CQ Containment Revisited

\[ Q \subseteq Q' \iff \text{there exists a query homomorphism from } Q' \text{ to } Q \]

\[ \downarrow \uparrow \]

\[ Q \subseteq_\Sigma Q' \]

\[ Q ::= R(x,y), R(y,z), R(z,x) \]

\[ Q' ::= R(x,y), R(y,z), R(z,x), P(x,y,z) \]

\[ \Sigma = \begin{cases} R[1,2] \subseteq P[1,2] \\
P[2,3] \subseteq R[1,2] \\
P[3,1] \subseteq R[1,2] \end{cases} \]

\[ Q \subseteq_\Sigma Q' \quad \text{but} \quad \text{there is no query homomorphism from } Q' \text{ to } Q \]
CQ Containment Revisited

\( Q \subseteq Q' \iff \text{there exists a query homomorphism from } Q' \text{ to } Q \)

\[ \Downarrow \uparrow \]

\( Q \subseteq \Sigma Q' \)

\[
Q : - R(x,y), R(y,z), R(z,x)
\]

\[
Q' : - R(x,y), R(y,y), R(y,x)
\]

\[\Sigma = \begin{cases} R : \{1\} \rightarrow \{2\} \end{cases}\]

\( Q \subseteq \Sigma Q' \) but there is no query homomorphism from \( Q' \) to \( Q \)
Theorem: Let $Q$ and $Q'$ be conjunctive queries, and $\Sigma$ a set of constraints. It holds that: \( Q \subseteq_\Sigma Q' \iff \) there exists a query homomorphism from $Q'$ to $Q_\Sigma$

$a$ CQ that acts as a representative for all the specializations of $Q$ that comply with $\Sigma$

$Q_\Sigma$ can be constructed by applying a well-known algorithm – the chase
The Chase by Example

(inclusion dependencies)

\[
Q(x) \; :\; \rightarrow \; R(x,y)
\]

\[
\Sigma = \left\{ \begin{array}{l}
P[1,2] \subseteq P[2,1]
\end{array} \right\}
\]
The Chase by Example

(inclusion dependencies)

\[ Q(x) :\dash R(x,y) \]

\[ \Sigma = \begin{cases} 
P[1,2] \subseteq P[2,1] 
\end{cases} \]

\[ Q(x) :\dash R(x,y) \]
The Chase by Example

(inclusion dependencies)

\[ Q(x) :\overline{\rightarrow} R(x,y) \]

\[ \Sigma = \begin{cases} 
P[1,2] \subseteq P[2,1] 
\end{cases} \]

\[ Q(x) :\overline{\rightarrow} R(x,y) \]

\[ Q(x) :\overline{\rightarrow} R(x,y), P(y,z) \]
The Chase by Example

(inclusion dependencies)

\[ Q(x) \rightarrow R(x,y) \]

\[ \Sigma = \begin{cases} R[2] \subseteq P[1] \\ P[1,2] \subseteq P[2,1] \end{cases} \]

\[ Q(x) \rightarrow R(x,y) \]

\[ Q(x) \rightarrow R(x,y), P(y,z) \]

\[ Q(x) \rightarrow R(x,y), P(y,z), P(z,y) \]
The Chase by Example

(inclusion dependencies)

\[
Q(x) \iff R(x,y)
\]

\[
\Sigma = \left\{ \right.
\begin{array}{l}
R[2] \subseteq R[1]
\end{array}
\right\}
\]
The Chase by Example

(inclusion dependencies)

\[ Q(x) \trianglerighteq R(x, y) \]

\[ \Sigma = \{ R[2] \subseteq R[1] \} \]

\[ Q(x) \trianglerighteq R(x, y) \]

\[ Q(x) \trianglerighteq R(x, y), R(y, z) \]

\[ Q(x) \trianglerighteq R(x, y), R(y, z), R(z, w) \]

\[ \vdots \]

we need to build an infinite CQ
The Chase by Example

(functional dependencies)

\[ Q(x,y) \to R(x,y), R(y,z), R(x,z) \]

\[ \Sigma = \left\{ R : \{1\} \to \{2\} \right\} \]
The Chase by Example

(functional dependencies)

\[
Q(x,y) \iff R(x,y), R(y,z), R(x,z) \quad \Sigma = \left\{ \begin{array}{c} R : \{1\} \rightarrow \{2\} \end{array} \right\}
\]

\[
Q(x,y) \iff R(x,y), R(y,z), R(x,z)
\]
The Chase by Example

(functional dependencies)

\[ Q(x,y) \ :- \ R(x,y), R(y,z), R(x,z) \]

\[ \Sigma = \{ \begin{array}{c} R : \{1\} \rightarrow \{2\} \end{array} \} \]

\[ Q(x,y) \ :- \ R(x,y), R(y,z), R(x,z) \]

\[ Q(x,y) \ :- \ R(x,y), R(y,y) \]

✓
The Chase by Example

(functional dependencies)

\[ Q(x,y) \quad \text{:-} \quad R(x,a), R(y,z), R(x,b) \]

(a, b are constants)

\[ \Sigma = \left\{ R : \{1\} \rightarrow \{2\} \right\} \]
The Chase by Example

(functional dependencies)

\[ Q(x,y) \ :- \ R(x,a), R(y,z), R(x,b) \]

\( \Sigma = \{ \{1\} \rightarrow \{2\} \) 

(a,b are constants)

the chase fails – constants cannot be unified
the empty query is returned
CQ Containment Under Functional Dependencies

**Theorem:** Let $Q$ and $Q'$ be conjunctive queries, and $\Sigma$ a set of functional dependencies.

It holds that: $Q \subseteq_\Sigma Q' \iff$ there exists a query homomorphism from $Q'$ to $\text{chase}(Q, \Sigma)$

the result of the chase algorithm starting from $Q$ and applying the constraints of $\Sigma$

**Proof hint:** adapt the proof for the homomorphism theorem by exploiting the following:

- The canonical database of $\text{chase}(Q, \Sigma)$ is a finite database that satisfies $\Sigma$
- **Main property of the chase:** there exists a homomorphism that maps the body of $\text{chase}(Q, \Sigma)$ to every $D$ that (i) can be mapped to the body of $Q$, and (ii) satisfies $\Sigma$
CQ Containment Under Inclusion Dependencies

- Things are much more difficult for inclusion dependencies. By following the same approach as for functional dependencies we only show the following:

**Theorem:** Let \( Q \) and \( Q' \) be conjunctive queries, and \( \Sigma \) a set of inclusion dependencies. It holds that: \( Q \subseteq_{\Sigma,\infty} Q' \iff \) there exists a query homomorphism from \( Q' \) to \( \text{chase}(Q,\Sigma) \) for every, possibly infinite, database \( D \) that satisfies \( \Sigma \), \( Q(D) \subseteq Q'(D) \)

- Interestingly, the following highly non-trivial and deep theorem holds:

**Theorem (Finite Controllability):** \( Q \subseteq_\Sigma Q' \iff Q \subseteq_{\Sigma,\infty} Q' \)
CQ Containment Under Constraints

**Theorem:** Let $Q$ and $Q'$ be conjunctive queries, and $\Sigma$ a set of constraints. The problem of deciding whether $Q \subseteq_{\Sigma} Q'$ is

- NP-complete, if $\Sigma$ is a set of functional dependencies
- PSPACE-complete, if $\Sigma$ is a set of inclusion dependencies

**Proof Idea:**

**(NP-membership)** (i) Construct chase$(Q, \Sigma)$ in polynomial time, (ii) guess a substitution $h$, and (iii) verify that $h$ is a query homomorphism from $Q'$ to chase$(Q, \Sigma)$

**(NP-hardness)** Inherited from the constraint-free case

**(PSPACE-membership)** (i) Non-deterministically construct a subquery $Q''$ of chase$(Q, \Sigma)$ with $|Q''| \leq |Q'|$, (ii) guess a substitution $h$, and (iii) verify that $h$ is a query homomorphism from $Q'$ to $Q''$

**(PSPACE-hardness)** Simulate a PSPACE Turing machine
**Definition:** Given a CQ $Q$ and a set of constraints $\Sigma$, we say that $Q$ is semantically acyclic under $\Sigma$ if there exists an acyclic CQ $Q'$ such that $Q \equiv_{\Sigma} Q'$

\[
Q \subseteq_{\Sigma} Q' \quad \text{and} \quad Q' \subseteq_{\Sigma} Q
\]

**Two crucial questions:** given a CQ $Q$ and a set $\Sigma$ of constraints

1. Can we decide whether $Q$ is semantically acyclic under $\Sigma$, and what is the exact complexity? *Now, we have the tools to study this problem*

2. Does this help query evaluation?
Proposition (Small Query Property): Consider a CQ $Q$ and a set $\Sigma$ of inclusion dependencies. If $Q$ is semantically acyclic under $\Sigma$, then there exists an acyclic CQ $Q'$ such that $|Q'| \leq 2 \cdot |Q|$ and $Q \equiv_\Sigma Q'$

Guess-and-check algorithm:
1. Guess an acyclic CQ $Q'$ of size at most $2 \cdot |Q|$
2. Verify that $Q \subseteq_\Sigma Q'$ and $Q' \subseteq_\Sigma Q$

Theorem: Deciding semantic acyclicity under inclusion dependencies is:
- PSPACE-complete in general
- NP-complete for fixed arity (because containment is NP-complete)
Proposition (Small Query Property): Consider a CQ $Q$ and a set $\Sigma$ of functional dependencies over unary and binary relations. If $Q$ is semantically acyclic under $\Sigma$, then there exists an acyclic CQ $Q'$ such that $|Q'| \leq 2 \cdot |Q|$ and $Q \equiv_\Sigma Q'$

Guess-and-check algorithm:

1. Guess an acyclic CQ $Q'$ of size at most $2 \cdot |Q|$
2. Verify that $Q \subseteq_\Sigma Q'$ and $Q' \subseteq_\Sigma Q$

Theorem: Deciding semantic acyclicity under inclusion dependencies is NP-complete
Semantic Acyclicity Under Functional Dependencies

\[ R : \{1\} \rightarrow \{3\} \equiv R(x,y,z,w), R(x,y',z',w') \rightarrow z = z' \]

only one attribute

**Theorem:** Semantic acyclicity under unary functional dependencies (over unconstrained signatures) is NP-complete

**Open Problem:** Deciding semantic acyclicity under arbitrary (or even binary) functional dependencies is a non-trivial open problem
Evaluating Semantically Acyclic CQs

• Recall that evaluating \( Q \) over \( D \) takes time \( |D|^{O(|Q|)} \)

• Evaluating a CQ \( Q \) that is semantically acyclic under \( \Sigma \) over \( D \) takes time

\[
2^{O(|Q| + |\Sigma|)} + O(|D| \cdot |Q|)
\]

- time for computing an acyclic CQ \( Q' \) such that \( |Q'| \leq 2 \cdot |Q| \)
- \( |Q'| \leq 2 \cdot |Q| \)
- \( Q \equiv_{\Sigma} Q' \)
- Evaluation of an acyclic CQ \( Q_A \) is feasible in time \( O(|D| \cdot |Q_A|) \)

• Observe that \( 2^{O(|Q| + |\Sigma|)} + O(|D| \cdot |Q|) \) is dominated by \( O(|D| \cdot 2^{O(|Q| + |\Sigma|)}) \)

\( \Rightarrow \) fixed-parameter tractable
Acyclic Approximations Under Constraints

- There are CQs that are not semantically acyclic even in the presence of constraints.
- The small query properties lead to **acyclic approximations**.

**Theorem:** Consider a CQ $Q$ and a set $\Sigma$ of constraints. There exists an acyclic CQ $Q'$ of size at most $2 \cdot |Q|$ that is maximally contained in $Q$ under $\Sigma$ such that $Q' \subseteq_\Sigma Q$ and there is no acyclic CQ $Q''$ such that $Q'' \subseteq_\Sigma Q$ and $Q' \subseteq_\Sigma Q''$.

- We know that acyclic approximations of polynomial size always exist.
- However, by exploiting the constraints we obtain **more informative** approximations.
Semantic Optimization: Recap

- Constraints enrich semantic acyclicity

- We can decide semantic acyclicity in the presence of inclusion dependencies and functional dependencies over unary and binary relations
  - The underlying tool is CQ containment under constraints

- Semantic acyclicity under functional dependencies is an important open problem

- Semantically acyclic CQs can be evaluated “efficiently” (fixed-parameter tractability)

- For CQs that are not semantically acyclic, even in the presence of constraints, we can always compute (more informative) acyclic approximations
Semantic Acyclicity: Wrap-Up

• Semantic acyclicity is an interesting notion that allows us to replace a CQ with an acyclic one – this significantly improves query evaluation

• 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

  
  Semantic acyclicity under several classes of constraints

- Diego Figueira: Semantically Acyclic Conjunctive Queries under Functional Dependencies. LICS 2016: 847-856
  
  Semantic acyclicity under unary functional dependencies

  
  Containment of CQ under inclusion dependencies via the chase

  
  The paper that introduced the chase algorithm