Querying the Guarded Fragment via Resolution

Sen Zheng, Renate A. Schmidt

University of Manchester, UK

June 6, 2022

Aim: BCQ answering for GF

Aim: BCQ answering for GF

The guarded fragment (GF)

- Equality-free and no function symbol
- Subsumes many description logics and modal logics
- $\forall\overline{y}(G(\overline{x}) \rightarrow \varphi)$ and $\exists\overline{y}(G(\overline{x}) \wedge \varphi)$ if free variables of φ are among $\overline{x}$

Aim: BCQ answering for GF

The guarded fragment (GF)

- Equality-free and no function symbol
- Subsumes many description logics and modal logics
- $\forall \overline{y}(G(\overline{x}) \to \varphi)$ and $\exists \overline{y}(G(\overline{x}) \land \varphi)$ if free variables of φ are among $\overline{x}$

Boolean conjunctive query (BCQ)

- $\exists \overline{x} \varphi(\overline{x})$: a conjunction of atoms containing only constants and variables as arguments
- $\exists x_{1\ldots6}(A(x_1, x_2) \land B(x_1, x_3) \land C(x_3, x_4, x_5) \land D(x_5, x_6))$

Aim: BCQ answering for GF

The guarded fragment (GF)

- Equality-free and no function symbol
- Subsumes many description logics and modal logics
- $\forall \overline{y}(G(\overline{x}) \rightarrow \varphi)$ and $\exists \overline{y}(G(\overline{x}) \wedge \varphi)$ if free variables of φ are among $\overline{x}$

Boolean conjunctive query (BCQ)

- $\exists \overline{x}\varphi(\overline{x})$: a conjunction of atoms containing only constants and variables as arguments
- $\exists x_{1\ldots6}(A(x_1, x_2) \wedge B(x_1, x_3) \wedge C(x_3, x_4, x_5) \wedge D(x_5, x_6))$

BCQ answering for GF

$\Sigma \cup \mathcal{D} \models q$, given Σ in GF, ground atoms $\mathcal{D}$, a BCQ q.

Motivation

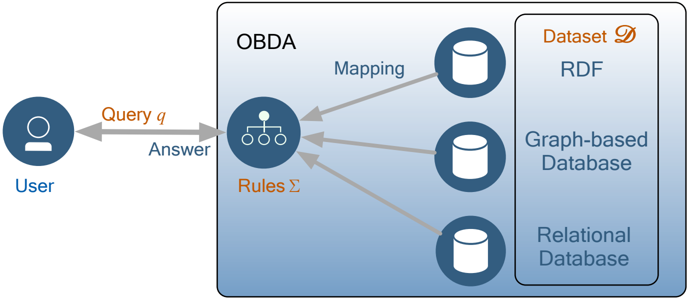
BCQ answering for GF

- Query containment/evaluation/entailment
- Constraint-satisfaction/homomorphism problem
- Ontology-based data access (OBDA) systems

Motivation

BCQ answering for GF

- Query containment/evaluation/entailment
- Constraint-satisfaction/homomorphism problem
- Ontology-based data access (OBDA) systems



Problems of interest

Some known results:

- Resolution decides GF [**?**, **?**]
- Retrieving answers over GF is undecidable [**?**]
  ($\exists y(Axy \wedge Byz)$ derives $\neg Axy \vee \neg Byz \vee qxz$)
- Querying GF is $2\mathrm{EXP}\textsc{time}$-complete
  [Vince Bárány and Georg Gottlob and Martin Otto(2010)]

Problems of interest

Some known results:

- Resolution decides GF [**?**, **?**]
- Retrieving answers over GF is undecidable [**?**]
  ($\exists y(Axy \wedge Byz)$ derives $\neg Axy \vee \neg Byz \vee qxz$)
- Querying GF is $2\mathrm{EXP}\textsc{Time}$-complete
  [Vince Bárány and Georg Gottlob and Martin Otto(2010)]

Problems of insterest:

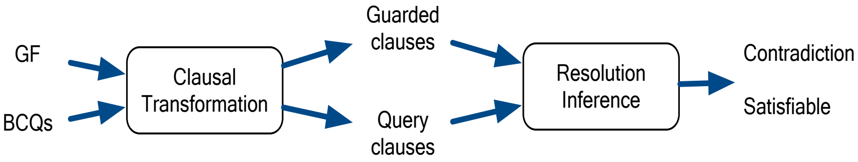
- No practical procedure exists for querying GF
- No practical procedure for BCQ rewriting for GF

Querying GF via resolution

Aim: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup Q \models \perp \qquad (Q \text{ means } \neg q)$

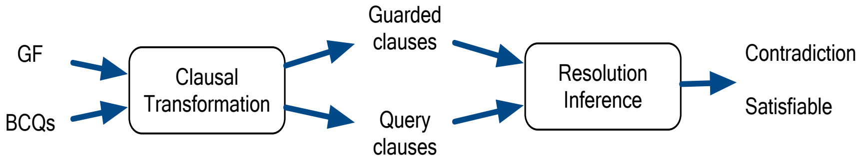
Querying GF via resolution

Aim: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup Q \models \perp \qquad (Q \text{ means } \neg q)$



Querying GF via resolution

Aim: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup Q \models \bot \qquad (Q \text{ means } \neg q)$

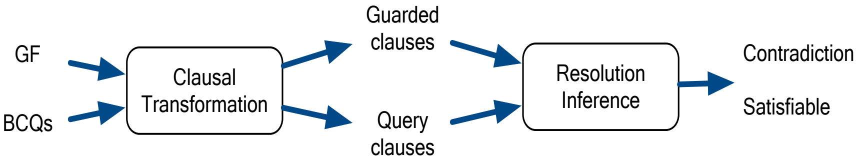


Guarded Clause:

- $D_1(fxy, x) \vee \neg Gxy, \quad \neg D_2(fx, x) \vee \neg Gxy, \quad \neg D_3(fxy, x) \vee \neg Gx$
- Covering and guardedness

Querying GF via resolution

Aim: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup Q \models \bot \qquad (Q \text{ means } \neg q)$



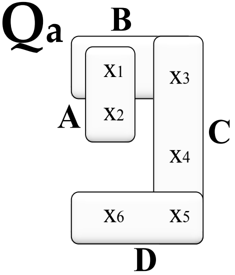
Guarded Clause:

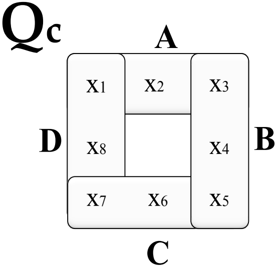
- $D_1(fxy, x) \vee \neg Gxy, \quad \neg D_2(fx, x) \vee \neg Gxy, \quad \neg D_3(fxy, x) \vee \neg Gx$
- Covering and guardedness

Query Clause:

- $\neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$
- Negative and compound-term-free

Handling query clauses





Acyclic query clause:

$$Q_a = \neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$$

Cyclic query clause:

$$Q_c = \neg A(x_1, x_2, x_3) \vee \neg B(x_3, x_4, x_5) \vee \neg C(x_5, x_6, x_7) \vee \neg D(x_7, x_8, x_1)$$

Handling query clauses

The separation rule:

$$\textbf{Sep}: \quad \frac{N \cup \{C \vee D\}}{N \cup \{\neg d_s(\overline{x}) \vee C,\ d_s(\overline{x}) \vee D\}}$$

1. $\mathrm{var}(C) \not\subseteq \mathrm{var}(D)$ and $\mathrm{var}(D) \not\subseteq \mathrm{var}(C)$
2. $\overline{x} = \mathrm{var}(C) \cap \mathrm{var}(D)$
3. d_s is a fresh predicate symbol, as a definer.

- **Replacement rule!**

Handling acyclic query clauses

$Q_a = \neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$

Handling acyclic query clauses

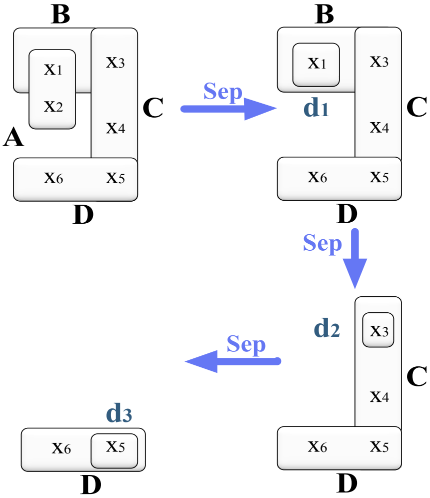
$Q_a = \neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$

Chained variables: x_1, x_3, x_5
Isolated variables: x_2, x_4, x_6

Handling acyclic query clauses

$Q_a = \neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$

Chained variables: x_1, x_3, x_5
Isolated variables: x_2, x_4, x_6



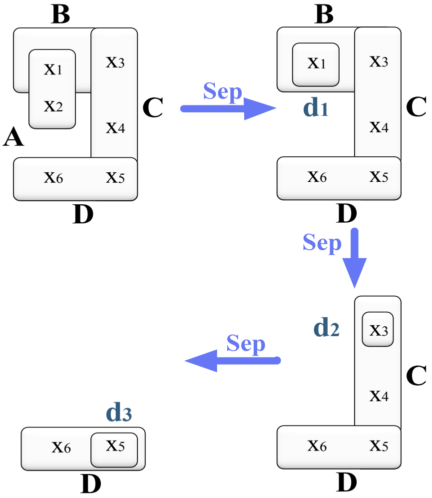
Handling acyclic query clauses

$Q_a = \neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$

Chained variables: x_1, x_3, x_5
Isolated variables: x_2, x_4, x_6

Sep:

- **Remove isolated variables**
- **Replace isolated-variable literals by definers**



Handling acyclic query clauses

$Q_a = \neg A(x_1, x_2) \vee \neg B(x_1, x_3) \vee \neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$

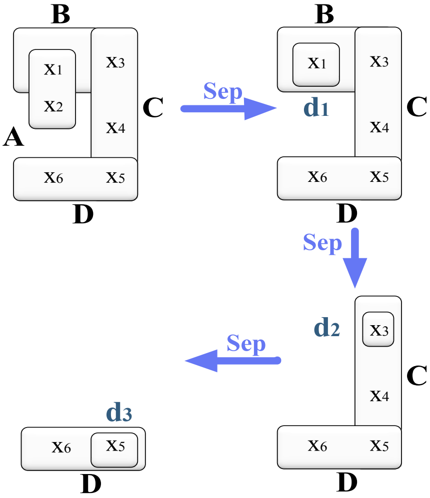
Chained variables: x_1, x_3, x_5
Isolated variables: x_2, x_4, x_6

Sep:

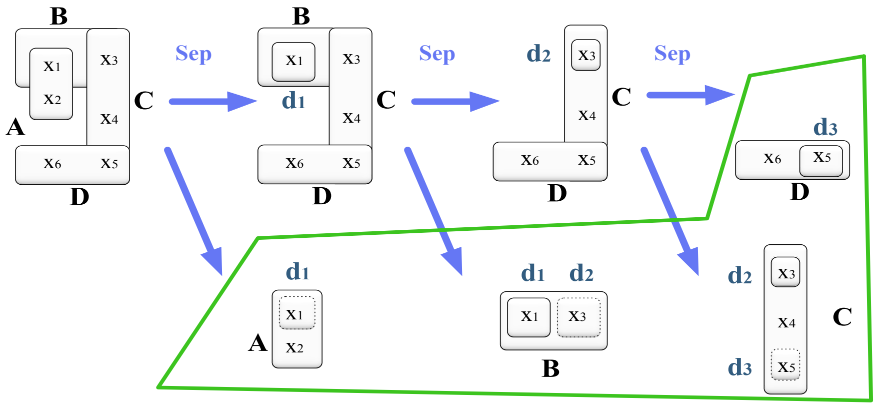
- **Remove isolated variables**
- **Replace isolated-variable literals by definers**

$Q_a \vdash_{Sep}$

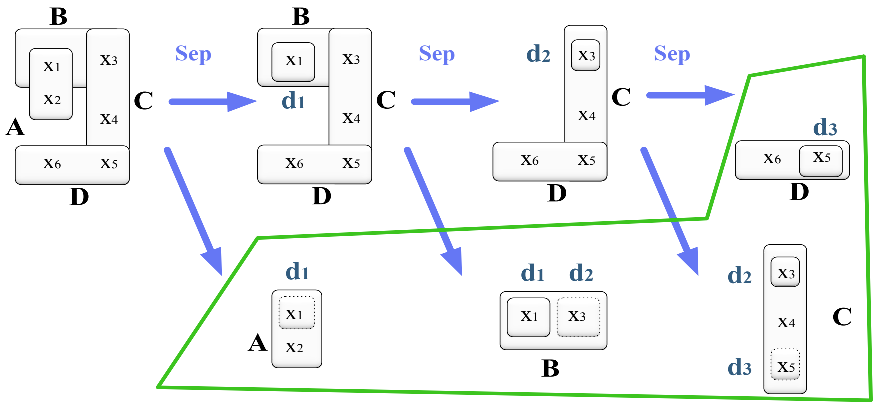
- $\neg A(x_1, x_2) \vee d_1(x_1)$
- $\neg d_1(x_1) \vee \neg B(x_1, x_3) \vee$
  $\neg C(x_3, x_4, x_5) \vee \neg D(x_5, x_6)$
- ...



Handling acyclic query clauses



Handling acyclic query clauses



1. $\neg A(x_1, x_2) \vee d_1(x_1)$
2. $\neg d_1(x_1) \vee \neg B(x_1, x_3) \vee d_2(x_3)$
3. $\neg d_2(x_3) \vee \neg C(x_3, x_4, x_5) \vee d_3(x_5)$
4. $\neg d_3(x_5) \vee \neg D(x_5, x_6)$

Handling cyclic query clauses

$$Q_c = \neg A(x_1, x_2, x_3) \vee \neg B(x_3, x_4, x_5) \vee \neg C(x_5, x_6, x_7) \vee \neg D(x_7, x_8, x_1)$$

Handling cyclic query clauses

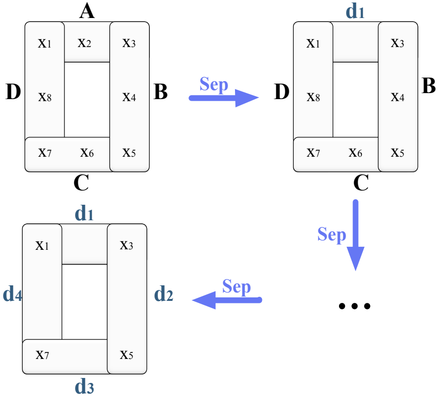
$$Q_c = \neg A(x_1, x_2, x_3) \vee \neg B(x_3, x_4, x_5) \vee \neg C(x_5, x_6, x_7) \vee \neg D(x_7, x_8, x_1)$$

Chained variable: x_1, x_3, x_5, x_7
Isolated variable: x_2, x_4, x_6, x_8

Handling cyclic query clauses

$$Q_c = \neg A(x_1, x_2, x_3) \vee \neg B(x_3, x_4, x_5) \vee \neg C(x_5, x_6, x_7) \vee \neg D(x_7, x_8, x_1)$$

Chained variable: x_1, x_3, x_5, x_7
Isolated variable: x_2, x_4, x_6, x_8



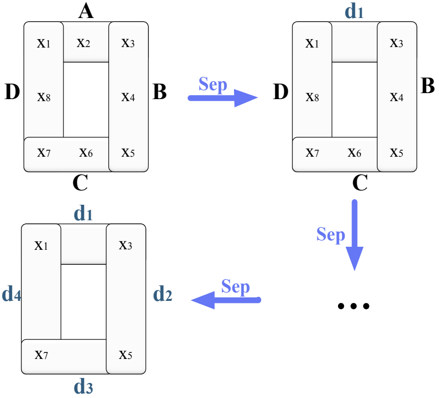
Handling cyclic query clauses

$$Q_c = \neg A(x_1, x_2, x_3) \vee \neg B(x_3, x_4, x_5) \vee \neg C(x_5, x_6, x_7) \vee \neg D(x_7, x_8, x_1)$$

Chained variable: x_1, x_3, x_5, x_7
Isolated variable: x_2, x_4, x_6, x_8

$Q_c \vdash_{Sep}$:

- $\neg A(x_1, x_2, x_3) \vee d_1(x_1, x_3)$
- $\neg d_1(x_1, x_3) \vee \neg B(x_3, x_4, x_5) \vee \neg C(x_5, x_6, x_7) \vee \neg D(x_7, x_8, x_1)$
- ...
- $\neg d_1(x_1, x_3) \vee \neg d_2(x_3, x_5) \vee \neg d_3(x_5, x_7) \vee \neg d_4(x_7, x_1)$



Handling chained-only query clause

$Q = \neg d_1(x_1, x_3) \vee \neg d_2(x_3, x_5) \vee \neg d_3(x_5, x_7) \vee \neg d_4(x_7, x_1)$

- Only chained variables

Handling chained-only query clause

$Q = \neg d_1(x_1, x_3) \vee \neg d_2(x_3, x_5) \vee \neg d_3(x_5, x_7) \vee \neg d_4(x_7, x_1)$

- Only chained variables

TRes + T-Trans

TRes:

- Macro inference rule on Q
- Inspired by 'MAXVAR' [**?**]

T-Trans:

- Structural transformation on **TRes** resolvents
- Obtain guarded clauses and a query clause Q'
- Q' is smaller than Q

Handling chained-only query clause

How to perform inferences to obtain clauses in our class?

- $Q = \neg d_1(x_1, x_3) \vee \neg d_2(x_3, x_5) \vee \neg d_3(x_5, x_7) \vee \neg d_4(x_7, x_1)$
- $C_1 = d_1(x, gxy) \vee \neg G_1xy$ $\quad C_2 = d_2(gxy, x) \vee P(hxy) \vee \neg G_2xy$
- $C_3 = d_3(fx, x) \vee \neg G_3x$ $\quad C_4 = d_4(x, fx) \vee \neg G_4x$

Handling chained-only query clause

How to perform inferences to obtain clauses in our class?

- $Q = \neg d_1(x_1, x_3) \vee \neg d_2(x_3, x_5) \vee \neg d_3(x_5, x_7) \vee \neg d_4(x_7, x_1)$
- $C_1 = d_1(x, gxy) \vee \neg G_1xy \qquad C_2 = d_2(gxy, x) \vee P(hxy) \vee \neg G_2xy$
- $C_3 = d_3(fx, x) \vee \neg G_3x \qquad C_4 = d_4(x, fx) \vee \neg G_4x$

TRes:

- Compute the mgu σ' among Q, $C_{1\ldots4}$:
  $\{x_1/fx,\ x_3/g(fx, y),\ x_5/fx,\ x_7/x\}$
- Perform a 'partial inference' on top variable (x_3) literals in Q
- Resolve Q, C_1 and C_2 using the mgu σ: $\{x_1/x, x_3/gxy, x_5/x\}$
- Compute 'partial conclusion'
  $R = \neg G_1xy \vee \neg G_2xy \vee P(hxy) \vee \neg d_3(x, x_7) \vee \neg d_4(x_7, x)$
- Makes 'maximal selection resolution' redundant

Handling chained-only query clause

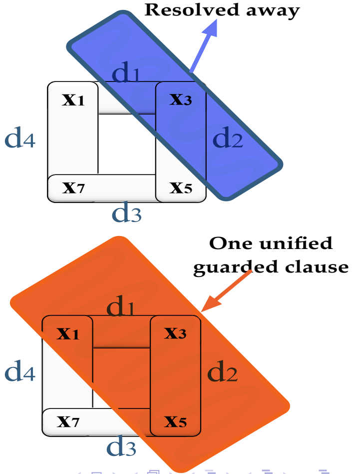
$R = \neg G_1 xy \vee \neg G_2 xy \vee P(hxy) \vee \neg d_3(x, x_7) \vee \neg d_4(x_7, x)$

- Neither a guarded clause nor a query clause.

Handling chained-only query clause

$R = \neg G_1xy \vee \neg G_2xy \vee P(hxy) \vee \neg d_3(x, x_7) \vee \neg d_4(x_7, x)$

- Neither a guarded clause nor a query clause.



Handling chained-only query clause

$R = \neg G_1xy \vee \neg G_2xy \vee P(hxy) \vee \neg d_3(x, x_7) \vee \neg d_4(x_7, x)$

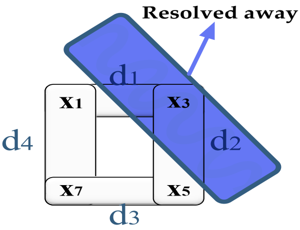
- Neither a guarded clause nor a query clause.

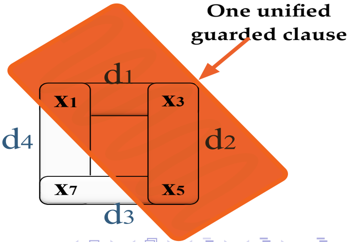
T-Trans represents R by

- Guarded clause $\neg G_1xy \vee \neg G_2xy \vee P(hxy) \vee d_t(x, y)$
- Query clause Q': $\neg d_t(x, y) \vee \neg d_3(x, x_7) \vee \neg d_4(x_7, x)$

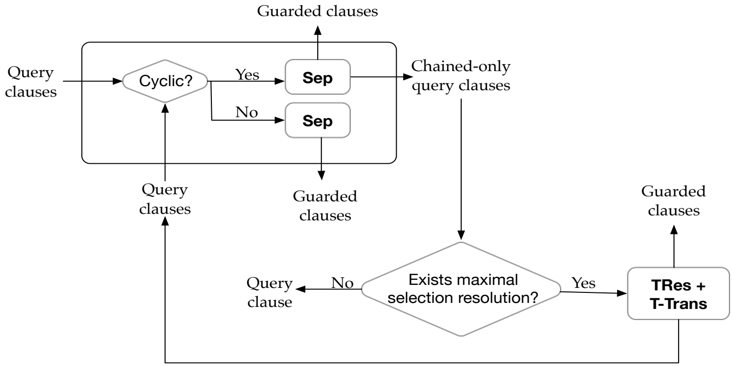
Q':

- Non-cyclic
- Breaks at least one cycle
- Smaller than Q





BCQ rewriting procedure **Q-Rewrite**



- Goal-oriented
- No grounding needed
- Saturate $\{\Sigma \cup \neg q\}$ to obtain Σ_q before considering $\mathcal{D}$
- $\mathcal{D} \cup \Sigma \models q$ reduces to $\mathcal{D} \models \Sigma_q$ **Tractable!** (in data complexity)

Conclusions and future work

- First practical BCQ answering/rewriting procedures over GF
- **Sep** is useful in decoupling non-cyclic chains
- But be careful with infinitely many definers
- ‘Partial inference’ in of the form **TRes** is insteresting

Conclusions and future work

- First practical BCQ answering/rewriting procedures over GF
- **Sep** is useful in decoupling non-cyclic chains
- But be careful with infinitely many definers
- ‘Partial inference’ in of the form **TRes** is insteresting

- Querying the loosely guarded fragment?
- Querying the guarded negation fragment ($\approx$)?
- Deciding fluted logic?
- Implementations and empirical evaluations

Thanks!