Deciding the Loosely Guarded Fragment and Querying Its Horn Fragment Using Resolution

Sen Zheng, Renate A. Schmidt

University of Manchester, UK

September 22, 2020

Aim

1. Deciding LGF

Aim

1. Deciding LGF

The loosely guarded fragment (LGF)

- Subsumes the guarded fragment
- Subsumes description logic $\mathcal{ALCHIO}$
- Generalises standard modal logics $\mathcal{K}, \mathcal{D}, \mathcal{S}4, \mathcal{B}$
- Express the temporal logic operater **until**

Aim

1. Deciding LGF

The loosely guarded fragment (LGF)

- Subsumes the guarded fragment
- Subsumes description logic $\mathcal{ALCHIO}$
- Generalises standard modal logics $\mathcal{K}, \mathcal{D}, \mathcal{S}4, \mathcal{B}$
- Express the temporal logic operater **until**

Deciding LGF

1. $\Sigma \models \bot$, given Σ in LGF

Aim

2. BCQ answering for Horn LGF

Aim

2. BCQ answering for Horn LGF

The Horn loosely guarded fragment (Horn LGF)

- Subsumes the guarded existential rules

Aim

2. BCQ answering for Horn LGF

The Horn loosely guarded fragment (Horn LGF)

- Subsumes the guarded existential rules

Boolean conjunctive query (BCQ)

- Returns yes or no.

Aim

2. BCQ answering for Horn LGF

The Horn loosely guarded fragment (Horn LGF)

- Subsumes the guarded existential rules

Boolean conjunctive query (BCQ)

- Returns yes or no.

BCQ answering for Horn LGF

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

Motivation

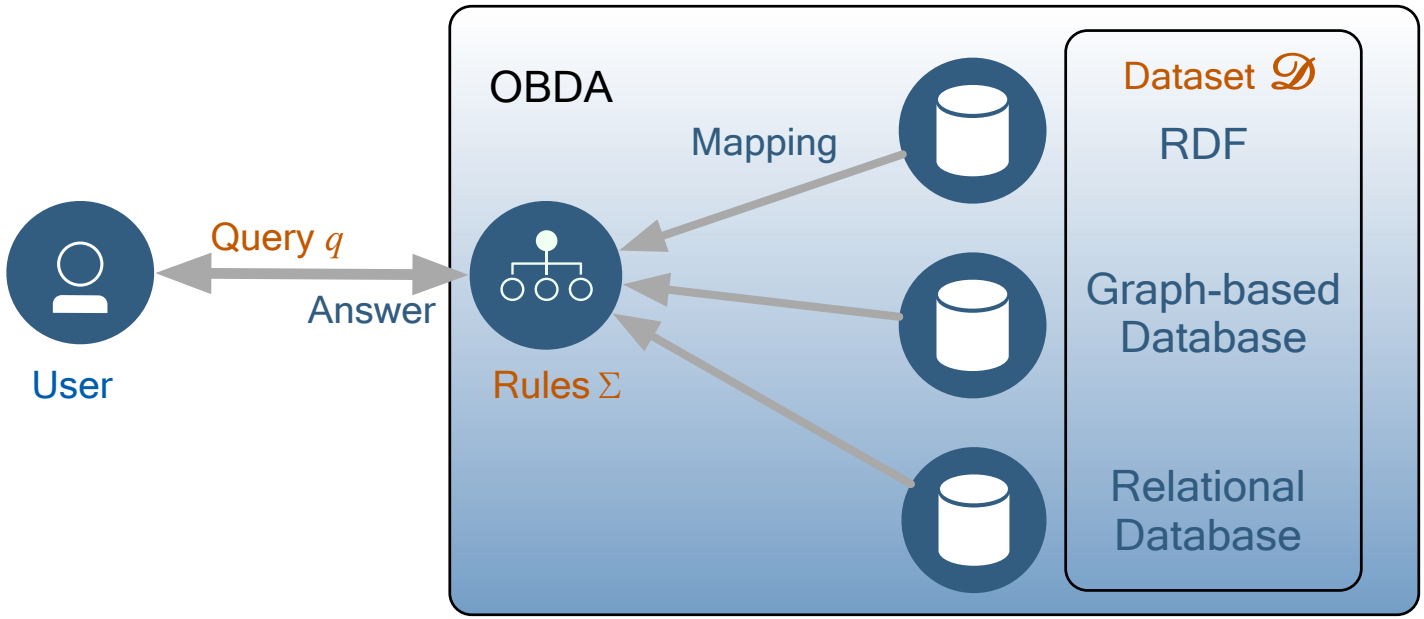
Deciding LGF/BCQ answering for Horn LGF

- Hyper-tree width queries $\mathcal{D} \models q$
- Query containment/evaluation/entailment $\Sigma \cup \mathcal{D} \models q$
- Constraint-satisfaction/homomorphism problem
- Ontology-based data access (OBDA) systems

Motivation

Deciding LGF/BCQ answering for Horn LGF

- Hyper-tree width queries $\mathcal{D} \models q$
- Query containment/evaluation/entailment $\Sigma \cup \mathcal{D} \models q$
- Constraint-satisfaction/homomorphism problem
- Ontology-based data access (OBDA) systems



Problems of interest

Some decidability results:

- First-order logic is undecidable
- Deciding LGF is 2ExpTime-complete
- Querying Horn LGF is 2ExpTime-complete

Problems of interest

Some decidability results:

- First-order logic is undecidable
- Deciding LGF is 2ExpTime-complete
- Querying Horn LGF is 2ExpTime-complete

Problem:

- No practical procedure exists for querying Horn LGF

Problems of interest

Some decidability results:

- First-order logic is undecidable
- Deciding LGF is $2\text{EXP}_{\text{TIME}}$-complete
- Querying Horn LGF is $2\text{EXP}_{\text{TIME}}$-complete

Problem:

- No practical procedure exists for querying Horn LGF

Contributions:

1. A practical procedure for deciding LGF
2. First practical procedure to query Horn LGF
3. First practical procedure for answering star queries and cloud queries over LGF

Resolution decides LGF

Shown by [de Nivelle et al. 2003] and [Ganzinger et al. 1999]:

- Introduce the 'MAXVAR' technique

Problems:

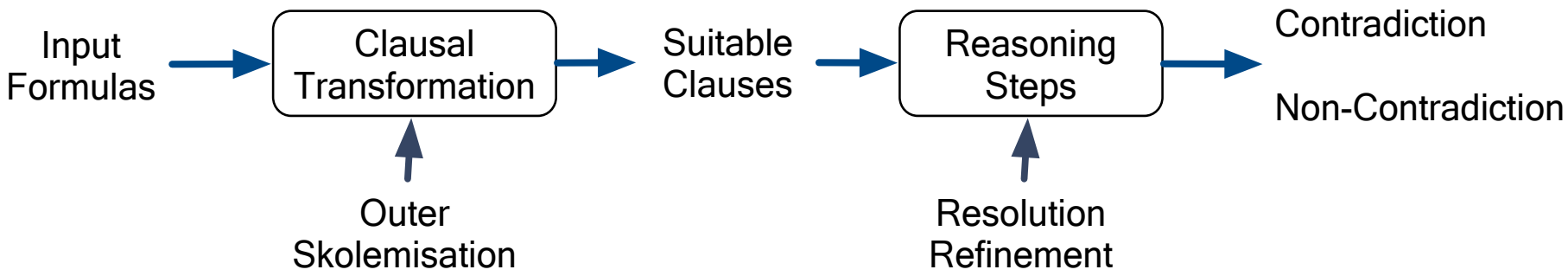
- Complicated
- Not considering queries

Reasoning using resolution

1. Deciding LGF: $\Sigma \models \bot$
2. Querying Horn LGF: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup \neg q \models \bot$

Reasoning using resolution

1. Deciding LGF: $\Sigma \models \bot$
2. Querying Horn LGF: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup \neg q \models \bot$

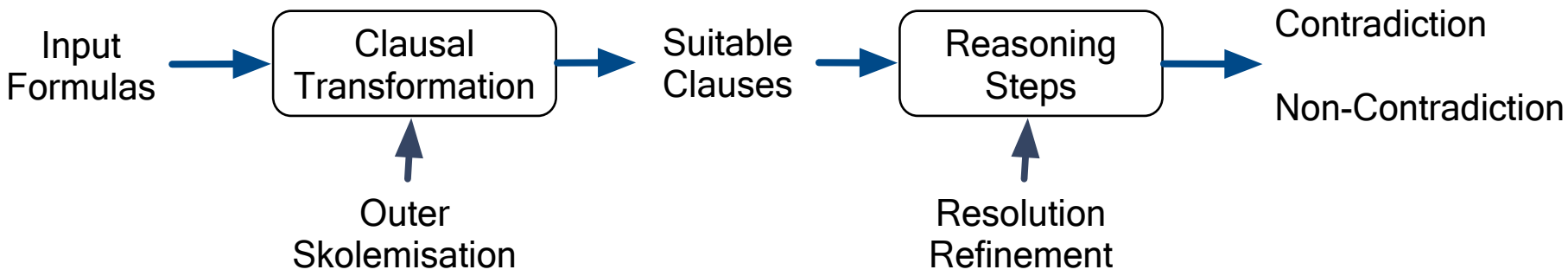


Applying resolution on Σ (or $\Sigma \cup \mathcal{D} \cup \neg q$) derives

- $\bot \Rightarrow$ Unsatisfiable (Yes)
- Saturated clauses $\Rightarrow$ Satisfiable (No)

Reasoning using resolution

1. Deciding LGF: $\Sigma \models \bot$
2. Querying Horn LGF: $\Sigma \cup \mathcal{D} \models q \quad \Leftrightarrow \quad \Sigma \cup \mathcal{D} \cup \neg q \models \bot$



Applying resolution on Σ (or $\Sigma \cup \mathcal{D} \cup \neg q$) derives

- $\bot \Rightarrow$ Unsatisfiable (Yes)
- Saturated clauses $\Rightarrow$ Satisfiable (No)

In finitely many steps!

Deciding LGF using resolution

Challenges:
Deeper terms and wider resolvents

Deciding LGF using resolution

Challenges:
Deeper terms and wider resolvents

Example 1

Given loosely guarded clauses C, C_1, C_2 and C_3:

$$C = \neg A_1 xy \vee \neg A_2 yz \vee \neg A_3 zx \qquad C_1 = A_1(fx, x) \vee D(gx) \vee \neg G_1 x$$
$$C_2 = A_2(fx, fx) \vee \neg G_2 x \qquad C_3 = A_3(x, fx) \vee \neg G_3 x$$

C, C_1, C_2 and C_3 derives $D(g(fx)) \vee \neg G_1(fx) \vee \neg G_2 x \vee \neg G_3(fx)$.

Contribution 1: Deciding LGF

LGF-Res system: ordered resolution with selection using **a special top variable selection refinement**

- A variation of [de Nivelle et al., 2003]
- No need for a specific unification algorithm
- Within the framework of [Bachmair et al., 2001]
- Fully developed definitions and proofs
- Minor corrections

Our methods for deciding LGF

Example 1

Given loosely guarded clauses C, C_1, C_2 and C_3:

$$C = \neg A_1 xy \vee \neg A_2 yz \vee \neg A_3 zx \qquad C_1 = A_1(fx, x) \vee D(gx) \vee \neg G_1 x$$
$$C_2 = A_2(fx, fx) \vee \neg G_2 x \qquad C_3 = A_3(x, fx) \vee \neg G_3 x$$

C, C_1, C_2 and C_3 derives $D(g(fx)) \vee \neg G_1(fx) \vee \neg G_2 x \vee \neg G_3(fx)$.

Our methods for deciding LGF

Example 1

Given loosely guarded clauses C, C_1, C_2 and C_3:

$$C = \neg A_1xy \vee \neg A_2yz \vee \neg A_3zx \qquad C_1 = A_1(fx, x) \vee D(gx) \vee \neg G_1x$$
$$C_2 = A_2(fx, fx) \vee \neg G_2x \qquad C_3 = A_3(x, fx) \vee \neg G_3x$$

C, C_1, C_2 and C_3 derives $D(g(fx)) \vee \neg G_1(fx) \vee \neg G_2x \vee \neg G_3(fx)$.

Top variable resolution: Resolving the potentially deepest literals

1. $\{x/ffx', y/fx', z/fx'\}$ to substitute variables in C
2. Select only $\neg A_1xy$ and $\neg A_3zx$

Our methods for deciding LGF

Example 1

Given loosely guarded clauses C, C_1, C_2 and C_3:

$$C = \neg A_1xy \vee \neg A_2yz \vee \neg A_3zx \qquad C_1 = A_1(fx, x) \vee D(gx) \vee \neg G_1x$$
$$C_2 = A_2(fx, fx) \vee \neg G_2x \qquad C_3 = A_3(x, fx) \vee \neg G_3x$$

C, C_1, C_2 and C_3 derives $D(g(fx)) \vee \neg G_1(fx) \vee \neg G_2x \vee \neg G_3(fx)$.

Top variable resolution: Resolving the potentially deepest literals

1. $\{x/ffx', y/fx', z/fx'\}$ to substitute variables in C
2. Select only $\neg A_1xy$ and $\neg A_3zx$
3. Derive $\neg A_2xx \vee D(gx) \vee \neg G_1x \vee \neg G_3x$

Our methods for deciding LGF

Example 1

Given loosely guarded clauses C, C_1, C_2 and C_3:

$$C = \neg A_1xy \vee \neg A_2yz \vee \neg A_3zx \qquad C_1 = A_1(fx, x) \vee D(gx) \vee \neg G_1x$$
$$C_2 = A_2(fx, fx) \vee \neg G_2x \qquad C_3 = A_3(x, fx) \vee \neg G_3x$$

C, C_1, C_2 and C_3 derives $D(g(fx)) \vee \neg G_1(fx) \vee \neg G_2x \vee \neg G_3(fx)$.

Top variable resolution: Resolving the potentially deepest literals

1. $\{x/ffx', y/fx', z/fx'\}$ to substitute variables in C
2. Select only $\neg A_1xy$ and $\neg A_3zx$
3. Derive $\neg A_2xx \vee D(gx) \vee \neg G_1x \vee \neg G_3x$

No term depth increase!

Contribution 2: Querying (Horn) LGF

Query-Res system: extends *LGF-Res* by **considering queries**

- Compute top variables in query clauses
- Query pair clauses $\Rightarrow$ not limited to LGF
- Top variable resolution $\Rightarrow$ rewriting queries
- Querying Horn LGF
- Star/cloud querying over LGF

Conclusions and Future Work

- A practical procedure to decide LGF
- First practical procedure to query Horn LGF
- First practical procedure to answer star/cloud queries over LGF
- Top variable resolution avoids variable depth increase

Conclusions and Future Work

- A practical procedure to decide LGF
- First practical procedure to query Horn LGF
- First practical procedure to answer star/cloud queries over LGF
- Top variable resolution avoids variable depth increase

- Querying the whole of the (loosely) guarded fragment?
- Querying the guarded negation fragment ($\approx$)?
- Deciding fluted logic?
- Experiments

Thanks!