ASP for Consistent Query Answering

YACINE SAHLI, University of Mons, belgium
JOACHIM SNEESSENS, University of Mons, belgium
MAXIME DANIELS, University of Mons, belgium

Fig. 1

Consistent query answering for inconsistent databases is a running problem...

CCS Concepts: • **Information systems → Database design and models**; **Database query processing**.

Additional Key Words and Phrases: Answer Set Programming, Consistent Query Answering

ACM Reference Format:
Yacine Sahli, Joachim Sneessens, and Maxime Daniels. 2020. ASP for Consistent Query Answering. In *Galway '20: ACM International Conference on Information and Knowledge Management, October 19–23, 2020, Galway, Ireland.* ACM, New York, NY, USA, **??** pages. https://doi.org/10.1145/1122445.1122456

Galway '20, October 19–23, 2020, Galway, Ireland

ACM ISBN 978-1-4503-XXXX-X/18/06...$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

The aim of this article is to present a fair comparison between two methods for solving the problem of CERTAINTY(q). Considering an inconsistent database, a repair is a maximal set of tuples from this database that respects his constraints. The CERTAINTY(q) problem consists in answering the question of knowing if it exists a repair that falsifies the query. Depending on the query, the CERTAINTY(q) problem can be either in first order complexity class, or in NP or co-NP. For the queries that are in first order, we want to compare the efficiency of the generate-and-test method and of the first order rewriting method.

To make a one to one comparison with the results found by Akhil A.Dixit and Phokion G.Kolaitis in their "A SAT-Based System for Consistent Query Answering", we decided to reuse the same FO-rewritable queries they used to prove that the KW-fo rewriting can be more efficient by using ASP instead of SQL.

2 CHOSEN QUERIES

$$
\begin{aligned}
q_1(z) &:= \exists x, y, v, w (R_1(\underline{x}, y, z) \wedge R_2(\underline{y}, v, w)) \\
q_2(z, w) &:= \exists x, y, v (R_1(\underline{x}, y, z) \wedge R_2(\underline{y}, v, w)) \\
q_3(z) &:= \exists x, y, v, u, d (R_1(\underline{x}, y, z) \wedge R_3(\underline{y}, v) \wedge R_2(\underline{v}, u, d)) \\
q_4(z, d) &:= \exists x, y, v, u (R_1(\underline{x}, y, z) \wedge R_3(\underline{y}, v) \wedge R_2(\underline{v}, u, d)) \\
q_5(z) &:= \exists x, y, v, w (R_1(\underline{x}, y, z) \wedge R_4(\underline{y, v}, w)) \\
q_6(z) &:= \exists x, y, x', w, d (R_1(\underline{x}, y, z) \wedge R_2(\underline{x'}, y, w)) \wedge R_5(\underline{x}, y, d) \\
q_7(z) &:= \exists x, y, w, d (R_1(\underline{x}, y, z) \wedge R_2(\underline{y}, x, w) \wedge R_5(\underline{x}, y, d))
\end{aligned}
$$

3 FIRST QUERY

```
certainty(Z):-r1(X,Y,Z),not p0(X,Z),not p1(x).
p0(X,Z):-r1(X,Y,Z1),r1(X,_,Z),not Z=Z1.
p1(X):-r1(X,Y,Z1),not p2(Y).
p2(Y):-r2(Y,V,W).

#show certainty/1.
```

4 SECOND QUERY

```
certainty(W,Z):-r1(X,Y,Z),not p0(Z,X),not q0(W,X), r2(P,Q,W).
p0(Z,X):-r1(X,Y,Z1),not Z1=Z,r1(X,_,Z).
q0(W,X):-r1(X,Y,Z1),not q1(W,Y), r2(P,Q,W).
q1(W,Y):-r2(Y,V,W),not q2(W,Y).
q2(W,Y):-r2(Y,V,W1),not W1=W,r2(Y,_,W).

#show certainty/2.
```

5 FOURTH QUERY

Generate-and-test method. (Does not work yet).

```
1 {rr1(X,Y,Z) : r1(X,Y,Z)} 1 :- r1(X,_,_).
```

```
1 {rr4(X,Y,Z) : r4(X,Y,Z)} 1 :- r4(X,Y,_).

:- rr1(X,Y,Z), rr4(Y,V,W).
```

FO rewriting

```
p(X,Z) :- r1(X,Y,Z2), Z2!=Z, r1(X,Y2,Z).
t(X) :- r1(X,Y,Z), not q(Y).
q(Y) :- r4(Y,V,W).
answer(Z) :- r1(X,Y,Z), not p(X,Z), not t(X).

#show answer/1.
```

6 SEVENTH QUERY

FO rewriting

```
certainty(Z) :- not d1(Z,Y), r2(Y,X,W), r1(X,Y,Z).
d1(Z,Y) :- not d2(Z,Y,X,W), r2(Y,X,W), r1(X,Y,Z).
d2(Z,Y,X,W) :- not d3(Z,Y,X,W), r2(Y,X,W), r1(X,Y,Z).
d3(Z,Y,X,W) :- r2(Y,X,W), not d4(Z,Y,X,W,P,Q), r1(X,P,Q), r1(X,Y,Z).
d4(Z,Y,X,W,P,Q) :- r1(X,P,Q), P=Y, r2(Y,X,W), Q=Z, d5(Z,Y,X,W).
d5(Z,Y,X,W) :- r5(X,Y,D), not d6(Z,Y,X,W), r2(Y,X,W), r1(X,Y,Z).
d6(Z,Y,X,W) :- not d7(Z,Y,X,W,P,D), r2(Y,X,W), r5(X,P,D), r1(X,Y,Z).
d7(Z,Y,X,W,P,D) :- r2(Y,X,W), r5(X,Z_5_0,D), r1(X,Y,Z), P=Y.

#show certainty/1.
```

ACKNOWLEDGMENTS