

# ASP for Consistent Query Answering

JEF WIJSEN, University of Mons, Belgium

YACINE SAHLI, University of Mons, belgium

JOACHIM SNEESESENS, 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:

Jef Wijsen, Yacine Sahli, Joachim Sneesens, 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, 2 pages. <https://doi.org/10.1145/1122445.1122456>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

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

© 2020 Association for Computing Machinery.

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 efficiencies of the generate-and-test method and of the first order rewriting method.

## 2 CHOSEN QUERIES

$$q_1(z) := \exists x, y, v, w (R_1(\underline{x}, y, z) \wedge R_2(\underline{y}, v, w))$$

$$q_2(z, w) := \exists x, y, v (R_1(\underline{x}, y, z) \wedge R_2(\underline{y}, v, w))$$

$$q_3(z) := \exists x, y, v, u, d (R_1(\underline{x}, y, z) \wedge R_3(\underline{y}, v) \wedge R_2(\underline{v}, u, d))$$

$$q_4(z, d) := \exists x, y, v, u (R_1(\underline{x}, y, z) \wedge R_3(\underline{y}, v) \wedge R_2(\underline{v}, u, d))$$

$$q_5(z) := \exists x, y, v, w (R_1(\underline{x}, y, z) \wedge R_4(\underline{y}, v, w))$$

$$q_6(z) := \exists x, y, x', w, d (R_1(\underline{x}, y, z) \wedge R_2(\underline{x'}, y, w)) \wedge R_5(\underline{x}, y, d)$$

$$q_7(z) := \exists x, y, w, d (R_1(\underline{x}, y, z) \wedge R_2(\underline{y}, x, w) \wedge R_5(\underline{x}, y, d))$$

## ACKNOWLEDGMENTS