Beyond Uniform Equivalence between Answer-Set Programs
Relativisation and Projection

Johannes Oetsch

The context

Answer-set programming

- Encoding to logic program
- Theory
- ASP solver
- Models

Problem instance

For example:

Sudoku

1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 9 1
3 4 5 6 7 8 9 1 2
4 5 6 7 8 9 1 2 3
5 6 7 8 9 1 2 3 4
6 7 8 9 1 2 3 4 5
7 8 9 1 2 3 4 5 6
8 9 1 2 3 4 5 6 7
9 1 2 3 4 5 6 7 8

Dedicated equivalence notion for:

Propositional query equivalence

Given two propositional logic programs, check whether they have the same answer sets

- For each input given as set of facts over a specified alphabet
- If parts of the output may be ignored in the comparison of the answer sets (projecting away auxiliary atoms)

Propositional databases

- propositional programs as queries
- do they produce the same result for each database?

Modular programming

- input/output of program parts determined in terms of alphabets
- program parts may use local atoms

Example

Are those two programs query equivalent?

P1

sad — not happy
happy — not sad
confused — sad, happy

P2

happy — sad, happy
sappy — sad, happy
confused — sappy

That is, do they yield the same output for each input set of facts?

Yes!

Provided that:

- the input contains only facts from (sad, happy)
- the auxiliary atom sappy is ignored in the comparison of the output

For example:

sad — happy — input

P1

sad, happy, confused — answer sets

P2

sad, happy, confused — projection to the output atom

Use of an auxiliary atom

Topic

Refined notions of program equivalence

Key observation: usual notion of equivalence is too weak in a nonmonotonic setting — no replacement property like in classical logic!

We have to consider:

- result of a program depends on the program (context) it is embedded in
- often used in practice but should be ignored in the resulting answer sets

Local (auxiliary) atoms:

- important for tasks like:
  - program optimisation
  - verification
  - debugging
  - modular programming

Contributions

Introduction of propositional query equivalence problems

- Instatiation of the framework by Eiter, Tompits, and Woltran [2005]
- Provide model-theoretic characterisations

Characterizations in terms of counterexamples (for two programs)

- in terms of wedges (associated with one program)

Refine complexity results for deciding such problems

- Without projection to the output atoms in the comparison:
  - second level of the polynomial hierarchy

- With projection to the output atoms in the comparison:
  - third level of the polynomial hierarchy

Thus, projection is an additional source of complexity

In this sense:

"Facts do not cease to exist because they are ignored."

Reduction of equivalence problems to quantified Boolean formulas (QBFs)

Equivalence problem holds iff the associated QBF is true

Implementation: reasoning tool ccT

Experiemntally and empirically evaluation

Quantified Boolean formulas:

- extension of classical propositional logic
- allow quantification over atoms
- QBFs provide prototypical problems for each level of the polynomial hierarchy

Overall architecture of the implementation

This work was partially supported by the Austrian Science Fund (FWF) under grant P18019 and P P21698.

Kontakt: oetsch@kr.tuwien.at

www.kr.tuwien.ac.at/research/ccT