Exploiting Support Sets for Answer Set Programs with External Evaluations

Thomas Eiter  Michael Fink  Christoph Redl  Daria Stepanova
{eiter,fink,redl,dasha}@kr.tuwien.ac.at

1. Motivation

- **HEX-programs** extend ASP by external sources
  - DL-program \( \Pi = (Q, P, \alpha) \), ontology + rules (special case of HEX-programs)
  
  \[
  \begin{align*}
  Q = & \{ \text{Driver} \leftarrow \neg \text{Customer} \} \\
  P = & \{ \exists \text{worksIn}(d1, r2) \} \\
  \end{align*}
  \]

- **Evaluation of HEX-programs**: multiple calls to external sources are expensive!

- **Aim of this work**: avoid multiple calls

- **Contributions**:
  - (Non-)ground support sets as optimization means
  - Application examples:
    - DL-programs (DL-Lite_A) and Query Answering (QA) over ASP
    - Implementation in DLVHEX and experiments

2. Support Sets

- **Support Sets** encode partial info about external source
  - **Ground Support Set** for \( a = \text{DL}[\text{worksIn} \cup \neg \text{to}; \neg \text{Customer}] \): \( S = \{ \neg \text{to}(d1, r4) \} \) for all assignments \( A \supseteq S : A \models a \)
  - **Complete Support Family** \( S \) for \( a \): for all \( A \) there is \( S \subseteq S : A \supseteq S \)
  - **Nonground Support Set** \( S(a(X)) \) is of form \( \langle N, \gamma \rangle \), where
    - \( N \): set of signed nonground literals over input predicates of \( a(X) \)
    - \( \gamma \): function, selecting groundings of \( N \), that are support sets for \( a(c) \)
  - \( S_1 = \{ \neg \text{to}(X', X') \} \), \( T \), \( \) where
    - \( T \) returns \( 1 \) for all groundings of \( \neg \text{to}(X', X') \)
  - \( S_2 = \{ \emptyset, \gamma \} \), where
    - \( \gamma : C \times \emptyset \rightarrow \{ 0, 1 \} \) is s.t. \( \gamma(c, 0) = 1 \) iff \( E\text{Driver}(c) \in A \) of \( O \)

3. Using Support Sets

- **Standard HEX-Program Evaluation**:
  - From \( \Pi \) construct \( \hat{\Pi} \) with all external \( a \) substituted by \( e_a \)
  - Add \( e_a \cup ne_a \) to \( \hat{\Pi} \), where \( ne_a \) corresponds to negation of \( e_a \)
  - For each \( A \in AS(\hat{\Pi}) \), check compatibility (i.e. \( Te_a \in A \iff A \models a? \) and minimality (i.e. exclude self-support)

- **New Approach**:
  - **Support Sets in AS Search**: for \( S \in S^+(a) \) (resp. \( S \in S^-(a) \)) adding \( S \cup \{ Fe_a \} \) (resp. \( S \cup \{ Te_a \} \)) to \( \hat{\Pi} \) prunes not compatible \( A \)
  - **Compatibility Check**: with complete support families for all \( a \) of \( \Pi \)
  - Support sets must be small and easily computable!
  - DL-programs over consistent DL-Lite_A ontologies:
    - size is at most 2
    - computation is tractable
  - Also: QA over positive ASP (e.g. subproblems)

4. Benchmark Results

- **Support Sets** encode partial info about external source
  - **Ground Support Set** for \( a = \text{DL}[\text{worksIn} \cup \neg\text{to}; \neg\text{Customer}] \): \( S = \{ \neg\text{to}(d1, r4) \} \) for all assignments \( A \supseteq S : A \models a \)
  - **Complete Support Family** \( S \) for \( a \): for all \( A \) there is \( S \subseteq S : A \supseteq S \)
  - **Nonground Support Set** \( S(a(X)) \) is of form \( \langle N, \gamma \rangle \), where
    - \( N \): set of signed nonground literals over input predicates of \( a(X) \)
    - \( \gamma \): function, selecting groundings of \( N \), that are support sets for \( a(c) \)
  - \( S_1 = \{ \neg\text{to}(X', X') \} \), \( T \), \( \) where
    - \( T \) returns \( 1 \) for all groundings of \( \neg\text{to}(X', X') \)
  - \( S_2 = \{ \emptyset, \gamma \} \), where
    - \( \gamma : C \times \emptyset \rightarrow \{ 0, 1 \} \) is s.t. \( \gamma(c, 0) = 1 \) iff \( E\text{Driver}(c) \in A \) of \( O \)

- **Other benchmarks**:
  - Default rules over university LUBM ontology
  - Graph non-3-colorability problem

5. Conclusion and Outlook

**Results**:
- Support sets are viewed as knowledge compilation
- Experimental results show significant improvements for practical applications: DL-programs over DL-Lite_A and QA over positive ASP

**Future work**:
- Sophisticated algorithms for nogood grounding, coverage checking
- Exploiting info about the program by support sets
- Further optimization techniques

**References**:
- Calvanese, D.; Lembo, D.; Lenznerini, M.; and Rosati, R.
  Tractable reasoning and efficient query answering in description logics: The DL-Lite family.
- Eiter, T.; Fink, M.; Krennwallner, T.; Redl, C.; and Schüller, P.
  Efficient HEX-program evaluation based on unfounded sets.