1. Motivation

Problem encode

ASP Program

ASP Solver extract

Solution(s)

- HEX-programs extend ASP by external sources
- Similar to SMT for SAT, but external source is black box
- Rule bodies may contain external atoms of the form \( \langle g, p \rangle \)
  - \( g \) is an external predicate name
  - \( p = p_1, ..., p_n \) are input predicate names or constants
  - \( e = e_1, ..., e_l \) are output terms

Semantics: Boolean oracle function \( f_{\langle g, p \rangle} \) is true if \( f_{\langle g, p \rangle}(A, p, c) \), w.r.t. assignment \( A \)

Basic evaluation:
1. replace \( \langle g, p \rangle \) by \( e_{\langle g, p \rangle} \), add \( e_{\langle g, p \rangle} \lor ne_{\langle g, p \rangle} \)
2. run CDNL solver (e.g. Clasp)
3. check guess for \( \langle g, p \rangle \) when \( p \) decided
4. learn io-nogoods when evaluating external atoms to avoid wrong guesses

Challenge: External sources cannot guide the solver effectively, they are black boxes evaluated under complete assignments!

Example

Oracle function for checking if size of predicate extension \( \geq n \):
\[
f_{\langle g, p \rangle}(A, p, n) = \begin{cases} T & \text{if } |\{ \langle p(x, y) \rangle \in A \}| \geq n \\ F & \text{otherwise} \end{cases}
\]

HEX-program:
\[
\begin{align*}
vertex(a), vertex(b), \\
a(X, Y) \lor na(X, Y) & \leftarrow vertex(X), vertex(Y), \\
e_{\langle g, p \rangle}(a, 2) & \lor ne_{\langle g, p \rangle}(a, 2)
\end{align*}
\]

A: \{ \langle Fe_{\langle g, p \rangle}(a, 2) \rangle, Ta(a, b), Fa(b, a), Ta(a, a), Fa(b, b) \}
Learn: \{ \langle Fe_{\langle g, p \rangle}(a, 2) \rangle, Ta(a, b), Fa(b, a), Ta(a, a), Fa(b, b) \}

2. Main Contributions

Extension from two-valued to three-valued assignments, enables:
1. Early evaluation of external sources
2. External theory learning producing smaller nogoods
3. Nogood minimization techniques

New techniques applicable by user without expert knowledge

3. Extension to Partial Assignments

Partial assignment over atoms \( A \) is set of signed literals \( Ta, Fa \) and \( Ua \) s.t. for all \( a \in A \) exactly one of \( Ta(a) \in A, Fa(a) \in A \) or \( Ua \in A \) holds.

A three-valued oracle function \( f_{\langle g, p \rangle} \) is a function such that \( f_{\langle g, p \rangle}(A, p, c) \) is true if \( f_{\langle g, p \rangle}(A, p, c) = \{ T, F, U \} \) for a partial assignment \( A \) and all possible values of \( p \) and \( c \).

A three-valued oracle function \( f_{\langle g, p \rangle} \) is assignment-monotonic if \( f_{\langle g, p \rangle}(A, p, c) = X, X \in \{ T, F \} \), implies \( f_{\langle g, p \rangle}(A', p, c) = X \) for all assignments \( A' \supseteq A \).

4. Nogood Learning with Partial Assignments

Nogood learning: Nogood only containing the decided part of a partial assignment learned as soon as oracle function evaluates to \( T \) or \( F \)
Partial nogoods often significantly smaller

Nogood minimization: Given an io-nogood \( N \), its minimized nogoods are

\[ minimize(N) = \{ N' \subseteq N \mid N' \text{ is an io-nogood,} \]
\[ f_{\langle g, p \rangle}(N', p, c) = U \text{ for all } N'' \subseteq N' \}. \]

Nogoods with same input part can be minimized simultaneously

Example

Extension to three-valued oracle function:
\[
f_{\langle g, p \rangle}(arc, n) = \begin{cases} T & \text{if } |\{ \langle arc(X, Y) \in A \} \rangle \geq n \\ U & \text{if } |\{ \langle arc(X, Y), arc(X, Y) \in A \} \rangle \geq n \\ F & \text{otherwise} \end{cases}
\]

External source can already be checked under partial assignment:
\[ A : \{ \langle Fe_{\langle g, p \rangle}(a, 2) \rangle, Ta(a, b), Fa(b, a), Ta(a, a), Fa(b, b) \} \]
Learn: \{ \langle Fe_{\langle g, p \rangle}(a, 2) \rangle, Ta(a, b), Fa(b, a), Ta(a, a), Fa(b, b) \}

5. Empirical Evaluation

Use Divhex implementation with Gringo/Clasp as backends

References