Liberal Safety for Answer Set Programs with External Sources

Thomas Eiter, Michael Fink, Thomas Krennwallner, Christoph Redl

{eiter,fink,tkren,redl}@kr.tuwien.ac.at

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Knowledge-Based Systems Group

July 16, 2013

Motivation

HEX-Programs

- Extend ASP by external sources
- Traditional safety not sufficient due to value invention
- Current notion of strong safety is unnecessarily restrictive

Example

$$\Pi = \left\{ \begin{array}{ll} r_1: t(a). & r_3: s(Y) \leftarrow t(X), \&cat[X,a](Y). \\ r_2: dom(aa). & r_4: t(X) \leftarrow s(X), dom(X). \end{array} \right\}$$

Contribution

- New more liberal safety criteria
- Still guarantee finite groundability
- Based on a modular framework $\Rightarrow$ extensibility of the approach

Liberal Safety: Basic Concepts

Monotone Grounding Operator

$$G_\Pi(\Pi') = \bigcup_{r\in\Pi}\{r\theta \mid \mathbf{A} \subseteq \mathcal{A}(\Pi'), \mathbf{A} \not\models \bot, \mathbf{A} \models B^+(r\theta)\},$$

where $\mathcal{A}(\Pi') = \{\mathbf{T}a, \mathbf{F}a \mid a \in A(\Pi')\} \setminus \{\mathbf{F}a \mid a \leftarrow . \in \Pi\}$
and $r\theta$ is the instance of r under variable substitution $\theta{:}\mathcal{V} \to \mathcal{C}$.

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

Least fixpoint of G_Π:

$$r_1' : \quad s(a). \quad r_2' : dom(ax). \quad r_3' : dom(axx).$$

Liberal Safety: Basic Concepts

Monotone Grounding Operator

$$G_\Pi(\Pi') = \bigcup_{r \in \Pi} \{r\theta \mid \mathbf{A} \subseteq \mathcal{A}(\Pi'), \mathbf{A} \not\models \bot, \mathbf{A} \models B^+(r\theta)\},$$

where $\mathcal{A}(\Pi') = \{\mathbf{T}a, \mathbf{F}a \mid a \in A(\Pi')\} \setminus \{\mathbf{F}a \mid a \leftarrow . \in \Pi\}$
and $r\theta$ is the instance of r under variable substitution $\theta{:}\mathcal{V} \to \mathcal{C}$.

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X, x](Y), dom(Y).$$

Least fixpoint of G_Π:

$$r'_1 : \; s(a). \quad r'_2 : \; dom(ax). \quad r'_3 : \; dom(axx).$$
$$r'_4 : \; s(ax) \leftarrow s(a), \&cat[a, x](ax), dom(ax).$$

Liberal Safety: Basic Concepts

Monotone Grounding Operator

$$G_{\Pi}(\Pi') = \bigcup_{r \in \Pi} \{ r\theta \mid \mathbf{A} \subseteq \mathcal{A}(\Pi'), \mathbf{A} \not\models \bot, \mathbf{A} \models B^{+}(r\theta) \},$$

where $\mathcal{A}(\Pi') = \{\mathbf{T}a, \mathbf{F}a \mid a \in A(\Pi')\} \setminus \{\mathbf{F}a \mid a \leftarrow . \in \Pi\}$
and $r\theta$ is the instance of r under variable substitution $\theta{:}\mathcal{V} \rightarrow \mathcal{C}$.

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X, x](Y), dom(Y).$$

Least fixpoint of G_{Π}:

$$r_1' : s(a). \quad r_2' : dom(ax). \quad r_3' : dom(axx).$$
$$r_4' : s(ax) \leftarrow s(a), \&cat[a, x](ax), dom(ax).$$
$$r_5' : s(axx) \leftarrow s(ax), \&cat[ax, x](axx), dom(axx).$$

Liberal Safety: Basic Concepts

Monotone Grounding Operator

$$G_\Pi(\Pi') = \bigcup_{r\in\Pi}\{r\theta \mid \mathbf{A} \subseteq \mathcal{A}(\Pi'), \mathbf{A} \not\models \bot, \mathbf{A} \models B^+(r\theta)\},$$

where $\mathcal{A}(\Pi') = \{\mathbf{T}a, \mathbf{F}a \mid a \in A(\Pi')\} \setminus \{\mathbf{F}a \mid a \leftarrow . \in \Pi\}$
and $r\theta$ is the instance of r under variable substitution $\theta{:}\mathcal{V} \to \mathcal{C}$.

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

Least fixpoint of G_Π:

$$r'_1 : s(a). \quad r'_2 : dom(ax). \quad r'_3 : dom(axx).$$
$$r'_4 : s(ax) \leftarrow s(a), \&cat[a,x](ax), dom(ax).$$
$$r'_5 : s(axx) \leftarrow s(ax), \&cat[ax,x](axx), dom(axx).$$

Intuition: We call a program safe if this operator produces a finite grounding

Liberal Safety

Two concepts

- A term is bounded if $G_{\Pi}(\Pi')$ contains only finitely many substitutions for it
- An attribute is de-safe if $G_{\Pi}(\Pi')$ contains only finitely many values at this attribute position

Idea

1. Start with empty set of bounded terms B_0 and de-safe attributes S_0
2. For all $n \geq 0$ until B_n and S_n do not change anymore
   a. Identify additional bounded terms $\Rightarrow B_{n+1}$
      (assuming that B_n are bounded and S_n are de-safe)
   b. Identify additional de-safe attributes $\Rightarrow S_{n+1}$
      (assuming that B_{n+1} are bounded and S_n are de-safe)

Liberal Safety

Two concepts

- A term is bounded if $G_{\Pi}(\Pi')$ contains only finitely many substitutions for it
- An attribute is de-safe if $G_{\Pi}(\Pi')$ contains only finitely many values at this attribute position

Idea

1. Start with empty set of bounded terms B_0 and de-safe attributes S_0
2. For all $n \geq 0$ until B_n and S_n do not change anymore
   a. Identify additional bounded terms $\Rightarrow B_{n+1}$ (assuming that B_n are bounded and S_n are de-safe)
   b. Identify additional de-safe attributes $\Rightarrow S_{n+1}$ (assuming that B_{n+1} are bounded and S_n are de-safe)

Identification of bounded terms in Step 2a by term bounding functions (TBFs)
Concrete safety criteria can be plugged in by specific TBF $b(\Pi, r, S, B)$

Liberal Safety

Two concepts

- A term is bounded if $G_\Pi(\Pi')$ contains only finitely many substitutions for it
- An attribute is de-safe if $G_\Pi(\Pi')$ contains only finitely many values at this attribute position

Idea

1. Start with empty set of bounded terms B_0 and de-safe attributes S_0
2. For all $n \geq 0$ until B_n and S_n do not change anymore
   a. Identify additional bounded terms $\Rightarrow B_{n+1}$ (assuming that B_n are bounded and S_n are de-safe)
   b. Identify additional de-safe attributes $\Rightarrow S_{n+1}$ (assuming that B_{n+1} are bounded and S_n are de-safe)

Identification of bounded terms in Step 2a by term bounding functions (TBFs)
Concrete safety criteria can be plugged in by specific TBF $b(\Pi, r, S, B)$
$\Rightarrow$ TBFs are a flexible means that however must fulfill certain conditions

Liberal Safety: Concrete TBF

Definition (Syntactic Term Bounding Function)

$t \in b_{syn}(\Pi, r, S, B)$ iff

(i) t is a constant in r; or

(ii) there is an ordinary atom $q(s_1, \ldots, s_{ar(q)}) \in B^+(r)$ s.t. $t = s_j$, for some $1 \leq j \leq ar(q)$ and $q \restriction j \in S$; or

(iii) for some external atom $\&g[\vec{X}](\vec{Y}) \in B^+(r)$, we have that $t = Y_i$ for some $Y_i \in \vec{Y}$, and for each $X_i \in \vec{X}$,

$$\begin{cases} X_i \in B, & \text{if } \tau(\&g, i) = \mathbf{const}, \\ X_i \restriction 1, \ldots, X_i \restriction ar(X_i) \in S, & \text{if } \tau(\&g, i) = \mathbf{pred}. \end{cases}$$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$
- $\Rightarrow S_1(\Pi) = \{dom \restriction 1, \&cat[X,x]_{r_4} \restriction_{\mathrm{I}} 2\}$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$
- $\Rightarrow S_1(\Pi) = \{dom\restriction 1, \&cat[X,x]_{r_4}\restriction_{\mathsf{l}} 2\}$
- $B_2(r_4, \Pi, b_{syn}) = \{Y\}$, $B_2(r_1, \Pi, b_{syn}) = \{a\}$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$
- $\Rightarrow S_1(\Pi) = \{dom \upharpoonright 1, \&cat[X,x]_{r_4} \upharpoonright_{\mathsf{I}} 2\}$
- $B_2(r_4, \Pi, b_{syn}) = \{Y\}$, $B_2(r_1, \Pi, b_{syn}) = \{a\}$
- $\Rightarrow S_2(\Pi) \supseteq \{s \upharpoonright 1, \&cat[X,x]_{r_4} \upharpoonright_{\mathsf{O}} 1\}$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X, x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$
- $\Rightarrow S_1(\Pi) = \{dom\restriction 1, \&cat[X, x]_{r_4}\restriction_{\mathsf{I}} 2\}$
- $B_2(r_4, \Pi, b_{syn}) = \{Y\}$, $B_2(r_1, \Pi, b_{syn}) = \{a\}$
- $\Rightarrow S_2(\Pi) \supseteq \{s\restriction 1, \&cat[X, x]_{r_4}\restriction_{\mathsf{o}} 1\}$
- $X \in B_3(r_4, \Pi, b_{syn})$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$
- $\Rightarrow S_1(\Pi) = \{dom \restriction 1, \&cat[X,x]_{r_4} \restriction_{\mathrm{i}} 2\}$
- $B_2(r_4, \Pi, b_{syn}) = \{Y\}$, $B_2(r_1, \Pi, b_{syn}) = \{a\}$
- $\Rightarrow S_2(\Pi) \supseteq \{s \restriction 1, \&cat[X,x]_{r_4} \restriction_{\mathrm{o}} 1\}$
- $X \in B_3(r_4, \Pi, b_{syn})$
- $\Rightarrow \&cat[X,x]_{r_4} \restriction_{\mathrm{i}} 1 \in S_3(\Pi)$

Liberal Safety: Concrete TBF

Example

Program Π:

$$r_1 : s(a). \quad r_2 : dom(ax). \quad r_3 : dom(axx).$$
$$r_4 : s(Y) \leftarrow s(X), \&cat[X,x](Y), dom(Y).$$

- $B_1(r_2, \Pi, b_{syn}) = \{ax\}$, $B_1(r_3, \Pi, b_{syn}) = \{axx\}$, $B_1(r_4, \Pi, b_{syn}) = \{x\}$
- $\Rightarrow S_1(\Pi) = \{dom \restriction 1, \&cat[X,x]_{r_4} \restriction_{\mathsf{I}} 2\}$
- $B_2(r_4, \Pi, b_{syn}) = \{Y\}$, $B_2(r_1, \Pi, b_{syn}) = \{a\}$
- $\Rightarrow S_2(\Pi) \supseteq \{s \restriction 1, \&cat[X,x]_{r_4} \restriction_{\mathsf{o}} 1\}$
- $X \in B_3(r_4, \Pi, b_{syn})$
- $\Rightarrow \&cat[X,x]_{r_4} \restriction_{\mathsf{I}} 1 \in S_3(\Pi)$

We also provide a TBF which exploits semantic properties of external sources

Liberal Safety: Results

Modular composition of TBFs:

Proposition

If $b_i(\Pi, r, S, B)$*,* $1 \leq i \leq \ell$*, are TBFs,*
then $b(\Pi, r, S, B) = \bigcup_{1 \leq i \leq \ell} b_i(\Pi, r, S, B)$ *is a TBF.*

Liberal Safety: Results

Modular composition of TBFs:

Proposition

If $b_i(\Pi, r, S, B)$, $1 \leq i \leq \ell$, *are TBFs,*
then $b(\Pi, r, S, B) = \bigcup_{1 \leq i \leq \ell} b_i(\Pi, r, S, B)$ *is a TBF.*

Operator G is a witness for finite groundability:

Proposition

If Π *is a de-safe program, then* $G_{\Pi}^{\infty}(\emptyset)$ *is finite.*

Proposition

Let Π *be a de-safe program. Then* Π *is finitely restrictable and* $G_{\Pi}^{\infty}(\emptyset) \equiv^{pos} \Pi$.

The results hold for any TBF!

Relations to Other Notions of Safety

Using TBF $b_{syn}(\Pi, r, S, B) \cup b_{sem}(\Pi, r, S, B)$, liberal de-safety is strictly more general than many other approaches:

Proposition

Every strongly de-safe [Eiter et al., 2006] program is de-safe.

Proposition

Every VI-restricted program [Calimeri et al., 2007] is de-safe.

Proposition

If Π is ω-restricted [Syrjänen, 2001], then it corresponds to a rewritten program $F(\Pi)$ which is de-safe.

Conclusion

ASP Programs with External Sources

- Ordinary safety not sufficient due to value invention
- Traditional strong safety is unnecessarily restrictive

Liberal Safety Criteria

- Based on term bounding functions (TBFs)
- Allows for easy extensibility of the approach
- We also provide concrete TBFs, which are strictly more liberal than many other approaches

Ongoing and Future Work

- Refine and extend existing TBFs (e.g. exploiting domain-specific properties)
- Define and implement grounding algorithms for the new class of programs

References

Calimeri, F., Cozza, S., and Ianni, G. (2007).
External Sources of Knowledge and Value Invention in Logic Programming.
Annals of Mathematics and Artificial Intelligence, 50(3–4):333–361.

Eiter, T., Ianni, G., Schindlauer, R., and Tompits, H. (2006).
Effective Integration of Declarative Rules with External Evaluations for Semantic-Web Reasoning.
In *3rd European Semantic Web Conference (ESWC'06)*, volume 4011 of *LNCS*, pages 273–287. Springer.

Syrjänen, T. (2001).
Omega-restricted logic programs.
In *6th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'01)*, volume 2173 of *LNCS*, pages 267–279. Springer.

Zantema, H. (1994).
Termination of term rewriting: Interpretation and type elimination.
Journal of Symbolic Computation, 17(1):23–50.