A many-sorted semantics for annotated logic programs

Ekaterina Komendantskaya

Department of Mathematics, University College Cork, Ireland

Information-MFCSIT,
1 August – 6 August, 2006

Outline

Outline

Outline

1. Motivation
2. Many-sorted languages and structures
3. Translation of annotated logic programs into many-sorted
   - Syntactical translation
   - Semantical translation
   - Translation for program clauses
   - Two SLD-resolutions

Outline

Motivation

When studying many-valued logic programming we need to establish:

- a better understanding of its semantics and models;
- a place for many-valued logic programs among other non-classical first-order logic programs;
- explicit description of properties of underlying structure of each program.

Many-sorted languages and structures

Many-sorted logic [Manzano]: signature

- We define a signature $\Sigma = \langle \mathrm{SORT}, \mathrm{RANK} \rangle$:

Many-sorted languages and structures

Many-sorted logic [Manzano]: signature

- We define a signature $\Sigma = \langle \mathrm{SORT}, \mathrm{RANK} \rangle$:
- Take an arbitrary set SORT and the set $S_\omega(\mathrm{SORT})$ of finite sequences of elements of SORT;
- Define a mapping $\mathrm{RANK} : \mathrm{OPER.SYM} \to S_\omega(\mathrm{SORT})$.

We take $\mathrm{SORT}^* = \{0, 1, 2, < 0, \overbrace{1, \ldots, 1}^{n}, 2 >\}$ (for each $n \in \mathbb{N}$). And call the resulting many-sorted language MSL^*.

Function RANK over OPER.SYM:

$$\text{OPER.SYM} = \{\neg, \vee, \wedge, \supset, \doteq, \leq_k, f^n, \vartheta^n, R^n, I^n, \varepsilon\}.$$

Function RANK over OPER.SYM:

$\mathrm{OPER.SYM} = \{\neg, \vee, \wedge, \supset, \doteq, \leq_k, f^n, \vartheta^n, R^n, I^n, \varepsilon\}$.
$\mathrm{RANK}(\neg) = \langle 0, 0\rangle$, $\mathrm{RANK}(\vee) = \mathrm{RANK}(\wedge) = \mathrm{RANK}(\supset) = \langle 0, 0, 0\rangle$;
$\mathrm{RANK}(\doteq) = \mathrm{RANK}(\leq_k) = \langle 0, 2, 2\rangle$.

$\mathrm{RANK}(f^n(x_1, \ldots, x_n)) = \langle 1, \overbrace{1, \ldots, 1}^{n}\rangle$;

$\mathrm{RANK}(\vartheta^n((\mu_1, \nu_1), \ldots, (\mu_n, \nu_n)) = \langle 2, \overbrace{2, \ldots, 2}^{n}\rangle$;

$\mathrm{RANK}(R^n(x_1, \ldots, x_n)) = \langle 0, \overbrace{1, \ldots, 1}^{n}\rangle$.

$\mathrm{RANK}(I^n((x_1, \ldots, x_n), (\mu, \nu))) = \langle 0, \overbrace{1, \ldots, 1}^{n}, 2\rangle$.

$\mathrm{RANK}(\varepsilon_n) = \langle 0, \overbrace{1, \ldots, 1}^{n}, 2, < 0, \overbrace{1, \ldots, 1}^{n}, 2 >\rangle$.

Many-sorted structure

Definition

We define a many-sorted *structure*

$$S = \langle A_1, A_2, A_3^n, f^{A_1}, f^{A_2}, f^{A_3^n} \rangle, (\texttt{for each } n \in \mathbb{N}),$$

where A_1, A_2 and A_3^n are universes for variables of sorts $< 1 >$, $< 2 >$, $< 0, \overbrace{1, \ldots, 1}^{n}, 2 >$; $f^{A_1} \subseteq A_1^k$, $f^{A_2} \subseteq A_2^k$ and $f^{A_3^n} \subseteq (A_3^n)^k$.

We define a many-sorted language $\mathcal{L}$ to consist of symbols from $\mathrm{OPER.SYM}$, quantifier $\exists$ and the set of variables $\mathcal{V} = V_i : i \in \mathrm{SORT} - \{0\}$.

Syntactical translation of atomic annotated formulae

Definition

For atomic annotated formulae we define
$\mathrm{TRANSL}_{BAP \mapsto \mathrm{MSL}^*}(R^n(x_1, \ldots, x_n) : (\mu, \nu)) =$
$\forall I \forall x_1, \ldots, x_n \forall (\mu', \nu')((R^n(x_1, \ldots, x_n) \supset (I^n(x_1, \ldots, x_n) \doteq (\mu', \nu')) \supset$
$((\mu, \nu) \leq_k (\mu', \nu'))).$

Syntactical translation of atomic annotated formulae

Definition

For atomic annotated formulae we define
$\mathrm{TRANSL}_{BAP \mapsto \mathrm{MSL}^*}(R^n(x_1, \ldots, x_n) : (\mu, \nu)) =$
$\forall I \forall x_1, \ldots, x_n \forall (\mu', \nu')((R^n(x_1, \ldots, x_n) \supset (I^n(x_1, \ldots, x_n) \doteq (\mu', \nu')) \supset$
$((\mu, \nu) \leq_k (\mu', \nu'))).$

In fact, $(I^n(x_1, \ldots, x_n) \doteq (\mu', \nu'))$ stands for
$\varepsilon x_1, \ldots x_n, \mu'', \nu'' I((x_1, \ldots, x_n), (\mu'', \nu'')) \wedge (\mu'', \nu'') \doteq (\mu', \nu').$

Translation of complex annotated formulae

$$\mathrm{TRANSL}^*(F_1 \otimes F_2) = \mathrm{TRANSL}^*(F_1) \wedge \mathrm{TRANSL}^*(F_2).$$

$$\mathrm{TRANSL}^*(\Sigma x F) = \exists x \mathrm{TRANSL}^*(F).$$

Important result

Lemma

Let F be an annotated formula of bilattice-based annotated language (BAL). Let Σ^* and S be a signature and a structure of MSL^*, respectively, and $||_I$ be an interpretation for the BAL, then the following holds:

$$|F|_I = \langle 1, 0 \rangle \text{ in } \mathrm{BAL} \iff S(\mathrm{TRANSL}^*(F)) = T \text{ in } \mathrm{MSL}^* \ .$$

Translation for program clauses

For each clause of the form
$R^n(x_1, \ldots, x_n) : (\mu, \nu) \leftarrow L_1 : (\mu_1, \nu_1) \otimes \ldots \otimes L_n : (\mu_n, \nu_n)$ we obtain the following set of many-sorted clauses:

- $(\mu, \nu) \leq_k (\mu', \nu') \leftarrow R^n(x_1, \ldots, x_n) \wedge I(x_1, \ldots, x_n) \doteq (\mu', \nu') \wedge \mathrm{TRANSL}^{**}(L_1 : (\mu_1, \nu_1) \otimes \ldots \otimes L_n : (\mu_n, \nu_n))$;
- $I(x_1, \ldots, x_n) \doteq (\mu', \nu') \leftarrow R^n(x_1, \ldots, x_n) \wedge \mathrm{TRANSL}^{**}(L_1 : (\mu_1, \nu_1) \otimes \ldots \otimes L_n : (\mu_n, \nu_n))$;
- $R^n(x_1, \ldots, x_n) \leftarrow \mathrm{TRANSL}^{**}(L_1 : (\mu_1, \nu_1) \otimes \ldots \otimes L_n : (\mu_n, \nu_n))$.

Some notes on how the translation works

1. For each annotated literal $L_i : (\mu_i, \nu_i) = R_i^m(y_1, \ldots, y_m) : (\mu_i, \nu_i)$ the translation function TRANSL^{**} gives the following result:
$\mathrm{TRANSL}^{**}(R_i^m(y_1, \ldots, y_m) : (\mu_i, \nu_i)) =$
$R_i^m(y_1, \ldots, y_m) \wedge (I_i(y_1, \ldots, y_m) \doteq (\mu_i', \nu_i')) \wedge ((\mu_i, \nu_i) \leq (\mu_i', \nu_i'))$.

Some notes on how the translation works

1. For each annotated literal $L_i : (\mu_i, \nu_i) = R_i^m(y_1, \ldots, y_m) : (\mu_i, \nu_i)$ the translation function TRANSL^{**} gives the following result:
$\mathrm{TRANSL}^{**}(R_i^m(y_1, \ldots, y_m) : (\mu_i, \nu_i)) =$
$R_i^m(y_1, \ldots, y_m) \wedge (I_i(y_1, \ldots, y_m) \doteq (\mu_i', \nu_i')) \wedge ((\mu_i, \nu_i) \leq (\mu_i', \nu_i'))$.
2. We enrich each $\mathrm{TRANSL}^{**}(P)$ with axioms which describe properties of $\doteq$ and $\leq_k$ and we will denote this enriched program as $\mathcal{P}^{**}$.

Axioms for bilattice-based logic programs

- $\forall(x \doteq x \leftarrow)$;
- $\forall(f(x_1, \ldots, x_y) \doteq f(y_1, \ldots, y_n) \leftarrow (x_1 \doteq y_1) \wedge \ldots \wedge (x_1 \doteq y_1))$;
- $\forall(x \leq_k x) \leftarrow$;
- $\forall(x \not\leq_k y) \leftarrow (y \leq_k x) \wedge (x \neq y)$;
- $\forall((x \leq_k z) \leftarrow (x \leq_k y) \wedge (y \leq_k z))$;
- $\forall((x \odot y) \leq_k (x' \odot y') \leftarrow (x \leq_k x') \wedge (y \leq_k y'))$, where $\odot$ stands for any of bilattice operations $\oplus, \otimes$;
- $\forall(x \leq_k (y \oplus z) \leftarrow (x \leq_k y))$;
- $\forall((x \otimes y) \leq_k z \leftarrow (x \leq_k z))$;
- $\forall((0,0) \leq_k x \leftarrow)$.

Main theorem

Theorem

For any bilattice-based annotated logic program P and any annotated formula F, $(P \models F) \Rightarrow (\mathcal{P}^{**} \models_S \mathrm{TRANSL}^{**}(F))$.

Two SLD-resolutions

Theorem

For any bilattice-based annotated logic program P and goal G the following property holds: if $P \cup \{\leftarrow G\}$ has a refutation, than $\mathcal{P}^{**} \cup \mathrm{TRANSL}^{**}(\leftarrow G)$ has a refutation.

Conclusions

1. The algorithm of translation of annotated logic programs into many-sorted works uniformly for most of many-valued logic programs, such as annotation-free logic programs; logic programs with annotations in implication; generalized annotated logic programs.

Conclusions

1. The algorithm of translation of annotated logic programs into many-sorted works uniformly for most of many-valued logic programs, such as annotation-free logic programs; logic programs with annotations in implication; generalized annotated logic programs.
2. The many-sorted structures we use are similar to the structures which Manzano used for translation of second-order logics into many-sorted. And this shows that many-valued logic programs are essentially second-order.

Conclusions

1. The algorithm of translation of annotated logic programs into many-sorted works uniformly for most of many-valued logic programs, such as annotation-free logic programs; logic programs with annotations in implication; generalized annotated logic programs.
2. The many-sorted structures we use are similar to the structures which Manzano used for translation of second-order logics into many-sorted. And this shows that many-valued logic programs are essentially second-order.
3. Translation of (complicated) annotated resolution into many-sorted resolution shows the way of simplifying existing many-valued resolutions and gives resolution procedure for those many-valued logic programs which have not received it yet.

Conclusions

1. The algorithm of translation of annotated logic programs into many-sorted works uniformly for most of many-valued logic programs, such as annotation-free logic programs; logic programs with annotations in implication; generalized annotated logic programs.
2. The many-sorted structures we use are similar to the structures which Manzano used for translation of second-order logics into many-sorted. And this shows that many-valued logic programs are essentially second-order.
3. Translation of (complicated) annotated resolution into many-sorted resolution shows the way of simplifying existing many-valued resolutions and gives resolution procedure for those many-valued logic programs which have not received it yet.
4. Properties of an underlying set/lattice/bilattice (or whatever) of a many-valued logic programs are shown explicitly in its many-sorted translation.

Thank you!