A Logic of Semantic Representations for Shallow Parsing

Alexander Koller
a.koller@ed.ac.uk
University of Edinburgh

Alex Lascarides
alex@inf.ed.ac.uk
University of Edinburgh

Ann Copestake
Ann.Copestake@cl.cam.ac.uk
University of Cambridge

Abstract

la la la

1 Introduction

Representing semantics as a logical form that supports automated inference and model construction is vital for deeper language engineering tasks, such as dialogue systems. These need access to the vast array of semantic information that is captured in conventional logical forms, including (but not limited to) information about semantic scope and predicate argument structure. Hand-crafted grammars yield detailed logical forms (e.g., (Butt et al., 1999; Copestake and Flickinger, 2000)), but deep parsing tends to lack robustness: hand-crafted grammars fail to cover all words and linguistic constructions; and unedited text and speech contains ill-formed phrases, which by design deep grammars do not handle.

Robost language processors that produce a single conventional logical form for a given natural language string are beginning to emerge (e.g., (Bos et al., 2004; Rupp et al., 2000; Wong and Mooney, 2006; Zettlemoyer and Collins, 2005)). But the output of these systems don't relate to any gold standard deep parse as produced by expert grammar developers (for instance, while the training corpus used in (Zettlemoyer and Collins, 2005) features control phenomena in the language strings, their logical forms don't represent it). This makes it hard to judge the logical forms that the models derive from a linguistic perspective; nor can one integrate their output with that of a hand-crafted grammar when desired.

This paper focusses on a particular approach to producing semantic information from robust parsers, exemplified in (Copestake, 2003; Frank, 2004), among others. Their strategy is to utilise semantic underspecification to semi-automatically build semantic components to shallow parsers, so that the output neither over-determines nor under-determines the semantic information that is revealed by the (shallow) syntactic analysis. The semantic formalism used to express this is Robust Minimal Recursion Semantics (RMRS, (Copestake, 2003)); this is an extension of MRS (Copestake et al., 2005) that is designed to be maximally flexible in the type of semantic information that can be left underspecified: it can express partial information about semantic scope, the values of arguments to predicate symbols and/or their argument position, the arity of the predicate symbols and the sorts of arguments they take. We show in Section 2 that all these features are needed when information about lexical subcategorisation or syntactic dependencies is missing—a characteristic feature of shallow parsers. Several researchers have demonstrated that RMRS is a suitable framework on which to semi-automatically construct semantic components to shallow parsers, ranging in depth from POS taggers (refs) to chunk parsers (frank ref) and intermediate parsers (RASP ref).

A major motivation for adopting RMRS over other techniques for robustly deriving logical forms is the promise that it can form the basis for integrating the output of several parsers, and be compared in particular with the output of a hand-crafted grammar. This paper demonstrates the feasibility of this integration for the first time, by introducing a model theory for RMRS, that in turn defines entailment among

RMRS representations. This entailment relation is also characterised syntactically as an extension of solved forms (ref to solved forms). We show that the proof theory and model theory of RMRS that results provides a formal basis for integrating the semantic output of several shallow parsers, for checking the satisfiability of a shallow parse, and for testing its compatibility with a deep parse.

2 Deep and shallow semantic construction

We motivate and describe the main features of RMRS with a (toy) example. Sentence (1a) features a semantic scope ambiguity that, according to most deep grammars, syntax doesn't resolve. The MRS (1b) derived by the English Resource Grammar (ERG, (Copestake and Flickinger, 2000)) reflects this by underspecifying socpe: (i) each predication is labelled (l_1, l_4, etc); (ii) scopal arguments are indicated by holes (e.g., h_2 and h_3 in the quantifier *every*); and (iii) constraints on scope are expressed by the qeq conditions (roughly, $h =_q l$ means that only zero or more quantifiers lie between the scopal argument h and the predication labelled by l in any fully-resolved logical form); and (iv) these scope constraints allow for more than one way of 'plugging' the holes with labels.[1]

(1) a. Every fat cat chased some dog.
b. l_1:_every_q_1(x, h_2, h_3)
l_4:_fat_j_1(e', x)
l_4:_cat_n_1(x)
l_5:_some_n_1(y, h_6, h_7)
l_8:_dog_n_1(y)
l_9:_chased_n_1(e, x, y)
$h_0 =_q l_9, h_2 =_q l_4, h_6 =_q l_9$

(1b) also exhibits MRS' compact description of conjunction through label sharing (see l_4 above), and the ERG's naming convention for predicate symbols: each predicate is constructed from the word lemma, its syntactic category and a sense number, and the leading underscore identifies lexical predicates.

Conceptually, this MRS is a partial description of all its possible (fully specific) interpretations in context. These are expressed in a so-called *base language*, where each predicate symbol in the MRS is a proxy for one or more base-language constructors of the same arity Assuming a very close correspondence between the MRS predicate symbols and the base-language constructors they stand for, the MRS (1b) (partially) describes both the base-language formulae in Figure 1, shown as trees to emphasise that an MRS is a partial description of the formulae's *form*; not its (base-language) entailments. In essence, the labels in the MRS reify the nodes in the tree, and each of these trees encodes one way of equating holes with labels that respects qeq constraints and binding conditions on the variables.

Put trees in here

Figure 1: The Form of alternative base logical forms of (1a).

The MRS (1c) is a notational variant of the RMRS (1d). RMRSs offer a more factorised representation: its predicate symbols are all unary and the other arguments to the (base-language) constructor that it corresponds to are represented by separate binary relations on the unique *anchor* of the predicate symbol $(a_1, a_{41}, \ldots)$ together with variable and label equalities (e.g., $x = x_1$, $l_{41} = l_{42}$).

(1) c. $l_1 : a_1$:_every_q_1(x_1),
RSTR(a_1, h_2), BODY(a_1, h_3)
$l_{41} : a_{41}$:_fat_j_1(e'), ARG1(a_{41}, x_2)
$l_{42} : a_{42}$:_cat_n_1(x_3)
$l_5 : a_5$:_some_q_1(x_4),
RSTR(a_5, h_6), BODY(a_5, h_7)
$l_8 : a_8$:_dog_n_1(x_5)
$l_9 : a_9$:_chase_v_1(e_{spast}),
ARG1(a_9, x_6), ARG2(a_9, x_7)
$h_2 =_q l_{42}, l_{41} = l_{42}, h_6 =_q l_8, h_0 =_q l_9$
$x_1 = x_2, x_2 = x_3, x_4 = x_5,$
$x_3 = x_6, x_5 = x_7$

This factored representation allows one to build semantic components to shallow parsers where lexical or syntactic information is absent. An extreme example would be a POS tagger: one can build its semantic component simply by deriving lexical predicate symbols from the word lemmas and their tags and ensuring that the predication for each word receives a unique label, anchor and argument, as given in (2):

[1] The event variable in the predicate symbol corresponding to the adjective is motivated by copulas—*The cat is fat* arguably has the logical form _the_q_1$(x$, _cat_n_1(x), _fat_j_1$(e, x))$.

(2) a. Every_AT1 fat_JJ cat_NN1 chased_VVD some_DD dog_NN1

b. $l_1 : a_1 :$ _every_q(x),
$l_{41} : a_{41} :$ _fat_a(e'),
$l_{42} : a_{42} :$ _cat_n(x_3)
$l_9 : a_9 :$ _chased_v(e_{past}),
$l_5 : a_5 :$ _some_q(x_4),
$l_8 : a_8 :$ _dog_n(x_5)

Semantic relations, sense tags and the arity of the base-language constructors are missing from (2b) because the POS tagger doesn't reveal information about syntactic constituency, word sense or lexical subcategorisation respectively. But the RMRSs (1c) and (2b) are entirely compatible, the former being more specific than the latter. At least, this holds so long as the base-language constructors that the RMRS predicate symbol _name_t_s corresponds to is a subset of those corresponding to the predicate symbol *_name_t*, for any word lemma *name* and POS tag t (written *_name_t_s* $\sqsubset$ *name_t*).

The values of non-scopal arguments are underspecified through the absence of ARG relations and/or through the absence of equalities and inequalities among the variables $x_1, x_2, \ldots$. The need to underspecify such semantic dependencies is required when the shallow parser doesn't record syntactic dependencies. Indeed, there is no single MRS which is satisfied by both the trees in Figure 2, assuming of course that disjunction is not part of the MRS language. In words, these trees collectively specify the (partial) semantic information that x is either the second argument (i.e., the direct object) or the third argument (i.e., the indirect object) to *give*, but we don't know which. A shallow processor that misses long distance dependencies in (3) cannot discern that the bone is the second argument to *give* in (3a), while the dog is the third argument in (3b).

(3) a. To which dog did Kim give the bone?

b. Which bone did Kim give the dog?

Thus to maximise the semantic contribution of the parser while not overdetermining it, one should express that the argument position of the noun phrases in *give the bone* vs. *give the dog* are underspecified between the second and third position, as shown in Figure 2. This will be expressed in RMRS via the binary relation $ARG_{\{2,3\}}(a, x)$ between the anchor

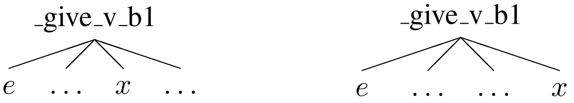

Figure 2: x is a participant in a giving event, but x's role is unknown.

a of the predicate symbol for *give* and the individual x. More generally, $ARG_n(a, x)$ means that x is an argument to the predicate anchored at a, but its argument position is unspecified.

The binary ARG relations also supports underspecifying the *arity* of the base-language constructors, as is required when lexical subcategorisation information is missing from the shallow parser. While a deep parser assumes complete information about lexical subcategorisation in any of its derivations, thereby fixing the arity of the base-language constructors in the fully-specific logical form to be the same as the MRS predicate symbol P that's introduced by the lexeme, the predicate symbols in an RMRS yielded from a shallow parse should, in principle at least, correspond to base language constructors of mixed arity. For instance, the English verb *bake* can be used causatively (*Kim baked the potatoes*), in which case semantically it corresponds to a 3-place predicate symbol *_bake_v_b1* in the base language, or it may be used inchoatively (*the potatoes are baking*), that semantically corresponds to a 2-place predicate *_bake_v_b2*. So the predicate symbol *_bake_v* in the RMRS that's introduced by a POS tagger must be able to denote either of these base-language predicates in the RMRS's interpretation.

3 Robust Minimal Recursion Semantics

3.1 RMRS Syntax

To formalise the basic ideas from Section 2, we start by defining the syntax of the RMRS language. We adopt a syntax in the style of CLLS (ref), in that labels, anchors and holes are all represented as *node variables* $\mathsf{NVar} = \{X, Y, X_1, \ldots\}$—the distinction among them comes from their position in the formulae of an RMRS description (see Definition 3.1). The RMRS vocabulary also includes base variables BVar, consisting of individual variables $\{x, x_1, y, \ldots\}$ and event variables $\{e_1, \ldots\}$ (one can also include supersorts to these, but we forego doing so here). Finally, the vocabulary includes 0-ary predicate symbols $\mathsf{Pred} = \{P, Q, P_1, \ldots\}$.

Definition 3.1. *An* RMRS *is an underspecified description φ consisting of a finite set of atoms of the forms in (4), such that φ satisfies the following three constraints:*

1. *If $X{:}Y{:}P, Z{:}Y{:}Q \in \varphi$, then $X = Z$ and $P = Q$ (i.e., each predication in φ has a unique anchor);*
2. *If $\mathsf{ARG}_S(X, v) \in \varphi$, then $X : Y : P \notin \varphi$ and $ARG_{S'}(Y, X) \notin \varphi$ for any $Y \in \mathsf{NVar}$ and $P \in \mathsf{Pred}$ (i.e., the first argument to an ARG-relation must be an anchor).*
3. *If $X{:}Y{:}P \in \varphi$, then $Y{:}Z{:}Q, Z{:}X{:}Q \notin \varphi$ for all $Z \in \mathsf{NVar}$ or $Q \in \mathsf{Pred}$ (i.e., an anchor cannot be a label, or* vice versa*).*

$$
\text{(4)}\qquad
\begin{array}{rcl}
\varphi & ::= & X{:}Y{:}P \\
 & | & \mathsf{ARG}_S(X, v) \\
 & | & \mathsf{ARG}_S(X, Y) \\
 & | & X =_q Y \\
 & | & v_1 = v_2 \mid v_1 \neq v_2 \\
 & | & X = Y \mid X \neq Y \\
 & | & P \sqsubseteq Q
\end{array}
$$

The subscript S in the ARG relations stands for a subset of $\mathbb{N}$ which is either finite or $\mathbb{N}$ itself. The original RMRS syntax ARG_i is an abbreviation of $\mathsf{ARG}_{\{i\}}$ and ARG_n is an abbreviation of $\mathsf{ARG}_{\mathbb{N}}$. Furthermore, $l : a : P(v)$ in the RMRSs from Section 2 and earlier papers (e.g., (Copestake, 2003)) is a notational variant of $\{X : Y : P, ARG_{\{0\}}(v)\}$ in Definition 3.1.

3.2 Model Theory

The model theory formalises the relationship between an RMRS and the fully specific, alternative logical forms that it describes, as expressed in the base language. As suggested in Section 2, each model τ corresponds to a unique formula of the base language, conceived as a tree (see Figure 1). *Satisfaction* must then give rise to a notion of *truth* and *entailment* that respectively correspond to an RMRS φ being a partial description of τ, and one RMRS φ being more specific than another φ' (in the sense that φ imposes more constraints than φ' on the possible fully-specific logical forms).

Since each model corresponds to a (unique) base language formula, defining models depends on the base-language's vocabulary Σ. More formally, Σ is a sorted signature of base-language constructors f of sorts $s_1 \times \ldots s_n \rightarrow s$, where $s, s_1, \ldots, s_n \in \mathcal{S}$, $\mathcal{S}$ being the sort hierarchy for Σ. We call $n \geq 0$ the *arity* of f, and we write Σ_i and $\Sigma_{\geq i}$ for the constructors in Σ with arity exactly i and at least i respectively.

The models of RMRS are then defined to be finite constructor trees:

Definition 3.2. *A* finite constructor tree *τ is a function $\tau : D \rightarrow \Sigma$ such that D is a* tree domain *(i.e., a subset of $\mathbb{N}^*$ which is closed under prefix and left sibling) and the number of children of each node $u \in D$ is equal to the arity of $\tau(u)$. We define the* sort *$s(u)$ of each node u in τ bottom-up by saying that if u has children $u_1, \ldots, u_n$ and $s(\tau(u)) = s_1 \times \ldots \times s_n \rightarrow s$, then $s(u) = s$ iff $s(u_i) \leq s_i$ for all i. τ is* well-sorted *if all nodes can be assigned sorts in this way. In what follows, we only consider well-sorted finite constructor trees.*

The same tree τ can represent formulas from different-style base languages (e.g., extensional and modal languages, DRT, etc) by leaving the encoding flexible. But for reasons of space, we don't explore this here; for simplicity, we assume a sort hierarchy $\mathcal{S}$ for Σ that is closely related to the sorts of RMRS descriptions themselves. So $\mathcal{S}$ contains the sorts h (for subformulas), x (for individual variables) and e (for event variables). We also assume that Σ contains conjunction constructors $\wedge_i : h \times \ldots h \rightarrow h$ of arity i for every $i \in \mathbb{N}$. We write $D(\tau)$ for the tree domain of a constructor tree τ, and further define the following relations between nodes in a finite constructor tree:

Definition 3.3. *$u \lhd_q^* v$ iff $v = u \cdot 2^k$ for some $k \geq 0$ and all symbols $\tau(u), \tau(u \cdot 2), \ldots, \tau(u \cdot 2^{k-1})$ are quantifiers.*

$u \lhd_\wedge^* v$ iff $u \leq v$ (i.e. it is a prefix), and all symbols on the path from u to v (not including v) are one of the $\wedge_i \in \Sigma$.

The *satisfaction* relation between an RMRS description φ and a finite constructor tree τ is defined in terms of several assignment functions. The need for a node variable assignment function $\alpha : \mathsf{NVar} \rightarrow D(\tau)$ is clear; the constraints on scope defined by the RMRS must be resolvable to τ's specific specific scope information, and for any given τ there may be more than one way to do this. Secondly, to ensure that, for instance, the predications corresponding to *every* and *cat* in the RMRS (2b) partially describe fully-specific logical forms that accurately record the semantic dependency between these lexemes in (1a) as well as partially describing fully-specific logical forms where they are not semantically dependent (e.g., as required when the same predications are produced from the POS tagged sentence *every dog chased some cat*), the relationship of satisfaction should ensure that the variables from BVar in an RMRS description can map in a many-to-one way to variables in $D(\tau)$. So satisfaction must also be defined in terms of a (0-ary) variable assignment function $g : \mathsf{BVar} \rightarrow \Sigma_0$ from RMRS variables to base-language variables that respects the sorts, i.e. $s(g(v)) \leq s(v)$ for all $v \in \mathsf{BVar}$. Finally, satisfaction must be defined in terms of a relation $\sigma \subseteq \mathsf{Pred} \times \Sigma_{\geq 1}$ that maps each RMRS predicate symbol to a set of constructors from Σ.

Definition 3.4. *Satisfaction of atoms is defined as follows:*

$\tau, \alpha, g, \sigma \models X{:}Y{:}P$ *iff*
$\quad\tau(\alpha(Y)) \in \sigma(P)$ *and* $\alpha(X) \lhd^*_{\wedge} \alpha(Y)$
$\tau, \alpha, g, \sigma \models \mathsf{ARG}_S(X, a)$ *iff exists* $i \in S$ *s.t.*
$\quad\alpha(X) \cdot i \in D(\tau)$ *and* $\tau(\alpha(X) \cdot i) = g(a)$
$\tau, \alpha, g, \sigma \models \mathsf{ARG}_S(X, Y)$ *iff exists* $i \in S$ *s.t.*
$\quad\alpha(X) \cdot i \in D(\tau), \alpha(X) \cdot i = \alpha(Y)$ *and*
$\quad s(\alpha(X) \cdot i) = h$
$\tau, \alpha, g, \sigma \models X =_q Y$ *iff* $\alpha(X) \lhd^*_q \alpha(Y)$
$\tau, \alpha, g, \sigma \models X = / \neq Y$ *iff*
$\quad\alpha(X) = / \neq \alpha(Y)$
$\tau, \alpha, g, \sigma \models v_1 = / \neq v_2$ *iff*
$\quad g(v_1) = / \neq g(v_2)$
$\tau, \alpha, g, \sigma \models P \sqsubseteq Q$*iff* $\sigma(P) \subseteq \sigma(Q)$

A 4-tuple τ, α, g, σ satisfies an underspecified representation φ (written $\tau, \alpha, g, \sigma \models \varphi$) iff it satisfies all of its elements.

Satisfaction encapsulates information about semantic scope ambiguities, missing information about semantic dependencies and/or missing information about lexical subcategorisation and lexical senses. For $j = \{1, 2\}$, suppose that $\tau_j, \alpha_j, g_j, \sigma \models \varphi$ and let $\mathsf{v}(\varphi)$ be the vocabulary that features in φ. Then φ exhibits a semantic scope ambiguity if there is a $Y, Y' \in \mathsf{v}(\varphi)$ such that $\alpha_1(Y) \lhd^*_q \alpha_1(Y')$ and $\alpha_2(Y') \lhd^*_q \alpha_2(Y)$. It exhibits missing information about semantic dependencies if $\exists v, v' \in \mathsf{BVar}$ in $\mathsf{v}(\varphi)$ such that $g_1(v) = g_1(v')$ and $g_2(v) \neq g_2(v')$. It exhibits missing lexical subcategorisation information if $\exists Y \in \mathsf{v}(\varphi)$ such that $\tau_1(\alpha_1(Y))$ is a constructor of a different type from $\tau_2(\alpha_2(Y))$. And it exhibits missing lexical sense information if $\tau_1(\alpha_1(Y))$ and $\tau_2(\alpha_2(Y))$ are different base-language constructors, but of the same type.

Truth, validity and entailment can now be defined in terms of satisfiability in the usual way:

Definition 3.5. *Let τ be a model, α a node variable assignment function, g a 0-ary variable assignment function, σ a mapping from* Pred *to the powerset of Σ, and φ, φ'* RMRS *descriptions. Then:*

truth: $\tau \models \varphi$ *iff* $\exists \alpha, g, \sigma$ *such that* $\tau, \alpha, g, \sigma \models \varphi$

validity: $\models \varphi$ *iff* $\forall \tau$, $\tau \models \varphi$.

entailment: $\varphi \models \varphi'$ *iff* $\forall \tau$, *if* $\tau \models \varphi$ *then* $\tau \models \varphi'$.

3.3 Solved Forms

Following the framework of dominance constraints, one can define a fragment of RMRS descriptions, known as *solved forms*, which are guaranteed to be satisfiable. For dominance constraints, solved forms are sets of constraints whose node variables form a forrest under the dominance relation $\lhd^*$. Solved forms for RMRS must impose additional constraints, so as to ensure that all the (underspecified) ARG relations are satisfiable. Indeed, we help to ensure this by substituting assuming that RMRS solved forms have any variable equalities substituted into the ARG relations themselves.

Definition 3.6. *An RMRS φ is* in solved form *iff:*

1. *every variable in φ is either a hole, a label or an anchor (but not two of these);*

2. *φ doesn't contain equality, inequality, and SPEC ($\sqsubseteq$) atoms;*

3. *if* $\mathsf{ARG}_S(Y, i)$ *is in φ, then $|S| = 1$;*

4. *for any label Y and index set S, there are no two atoms* $\mathsf{ARG}_S(Y, i)$ *and* $\mathsf{ARG}_S(Y, i')$ *in φ;*

5. *no label occurs on the right-hand side of two different $\lhd^*_q$ atoms.*

Claim 3.1. *Every RMRS in solved form is satisfiable.*

Proof. by constructing a model τ that contains for each predicate $P \in \varphi$ a member of $\sigma(P)$ with maximum arity, and that satisfies the ARGs and dominances in φ. □

One can now define the solved forms of an RMRS description:

Definition 3.7. *The* syntactic dominance relation *in an* RMRS φ *is the reflexive, transitive closure of the binary relation*

$$\{(X,Y) \mid \varphi \textit{ contains } X \lhd_q^* Y \textit{ or } ARG_S(X,Y) \textit{ for some } S\}$$

We write $D(\varphi)$ for the syntactic dominance relation of φ.

An RMRS φ' is a solved form of *the RMRS φ iff φ' is in solved form and there is a substitution s that maps the node and base variables of φ to the node and base variables of φ' such that*

1. *φ' contains the EP X':Y':P iff there are variables X, Y such that X:Y:P is in φ, $X' = s(X)$, and $Y' = s(Y)$;*
2. *for every atom $\mathsf{ARG}_S(X,i)$ in φ, there is exactly one atom $\mathsf{ARG}_{S'}(X',i')$ in φ with $X' = s(X)$, $i' = s(i)$, and $S' \subseteq S$;*
3. *$D(\varphi') \supseteq s(D(\varphi))$.*

Claim 3.2. *For every model τ of some* RMRS *φ, there is a solved form φ' of φ such that τ is also a model of φ'.*

Proof. will be by constructing a solved form that respects the equalities, ARGs, and dominances in τ. □

Proposition 3.1. *Every* RMRS *φ has only a finite number of solved forms, up to renaming of variables.*

Proof. Up to renaming of variables, there is only a finite number of substitutions on the node and base variables of φ. Let s be such a substitution. This fixes the set of EPs of any solved form of φ that is based on s uniquely. There is only a finite set of choices for the subsets S' in condition 2 of Def. 3.7, and there is only a finite set of choices of new $\lhd_q^*$ atoms that satisfy condition 3. Therefore, the set of solved forms of φ is finite. □

4 RMRS comparison as entailment

As discussed in Sections 1 and 2, RMRS's design aims to make it possible to compare the semantic output of several language processors of varying depth, and even combine them when appropriate. The following theorem demonstrates that RMRS-entailment—defined in terms of models and or solved forms—provides the logical basis for such parser integration:

Theorem 1. *Let φ, φ' be two* RMRS*s. Then $\varphi \models \varphi'$ iff for every solved form S of φ, there is a solved form S' of φ' such that S is an extension of S'.*

This theorem doesn't provide efficient algorithms for entailment checking, and of course this would be required to make parser comparison and integration with RMRS practical. Computing the solved forms of φ and φ' may be exponential, and the complexity of the RMRS validity problem is still unknown. However, the above theorem illustrates that our model theory for RMRS functions in the way it should for parser comparison and integration, and it can thus be used to prove important logical properties of shallow parsing systems that adopt RMRS.

The outline proof of Theorem 1 is as follows:

Proof. "$\Leftarrow$": Let M, α an arbitrary model and variable assignment that satisfy φ. There is a solved form s of φ such that $M, \alpha \models s$. By assumption, this means that there is a solved form s' of φ' such that s is an extension of s'. Therefore, $M, \alpha \models s'$ and hence $M, \alpha \models \varphi'$.

"$\Rightarrow$": Assume that $\varphi \models \varphi'$, and let s be an arbitrary solved form of φ. We need to show that we can remove atoms from s such that s remains an extension of the resulting s' and s' is a solved form of φ'.

We construct s' as the conjunction of all atoms in s that only contain variables occurring in φ' and all atom $X \lhd^* Y$ such that X and Y occur in φ' and $(X,Y) \in D(s)$.

It is obvious that s is an extension of s'. Furthermore, s' is in solved form, because s was a tree already and we didn't introduce new edges. The tricky part is to show that s' is a solved form of φ'. s' contains all labeling atoms of φ' according to Lemma 4.1; a similar argument holds for inequality atoms. Finally, $D(\varphi') \subseteq D(s')$ because $M, \alpha \models s'$

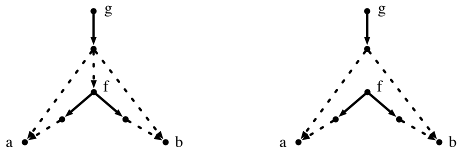

Figure 3: A counterexample to a simpler version of Theorem 1.

and s' is tree-shaped and normal and therefore contains all statements about dominance that are true in M. □

Lemma 4.1. *Under the assumptions of Theorem 1, if $\varphi \models \varphi'$, then φ' only contains labeling atoms that also occur in φ.*

Proof. Let $\varphi \models \varphi'$, but let φ' contain a labeling atom $X{:}f(X_1, \ldots, X_n)$ that doesn't occur in φ. Call this atom A.

Furthermore, let $M, \alpha \models \varphi$ arbitrary. Because $M, \alpha \models \varphi'$, we must have $M(\alpha(X)) = f$.

Now replace the label of $\alpha(X)$ in M by some other symbol g of arity n. Call the resulting model M'. We still have $M', \alpha \models \varphi$ because φ doesn't contain a labeling atom for X. But now $M', \alpha \not\models \varphi'$. This contradicts the entailment assumption. □

It is really necessary to state Theorem 1 in terms of extensions of solved forms. It is *not* true that $\varphi \models \varphi'$ implies that φ is an extension of φ'. A counterexample is shown in Fig. 3, where the right-hand constraint entails the left-hand constraint (they are equivalent), but is not an extension of it.

5 Conclusion

References

Johan Bos, Stephen Clark, Mark Steedman, James Curran, and Julia Hockenmaier. 2004. Wide coverage semantic representations from a CCG parser. In *Proceedings of the International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland.

Miriam Butt, Tracy Holloway King, Maria-Eugenia Ni no, and Frédérique Segond. 1999. *A Grammar Writer's Cookbook.* CSLI Publications.

A. Copestake and D. Flickinger. 2000. An open-source grammar development environment and english grammar using HPSG. In *Proceedings of the Second Conference on Language Resources and Evaluation (*LREC 2000*)*, pages 591–600, Athens.

A. Copestake, D. Flickinger, I. Sag, and C. Pollard. 2005. Minimal recursion semantics: An introduction. *Research on Language and Computation*, 3(2–3):281–332.

A. Copestake. 2003. Report on the design of rmrs. Technical Report EU Deliverable for Project number IST-2001-37836, WP1a, Computer Laboratory, University of Cambridge.

Anette Frank. 2004. Constraint-based rmrs construction from shallow grammars. In *Proceedings of the International Conference in Computational Linguistics (COLING 2004)*, Geneva, Switzerland.

C. J. Rupp, J. Spilker, M. Klarner, and K. Worm. 2000. Combining analyses from various parsers. In W. Wahlster, editor, *Verbmobil: Foundations of Speech to Speech Translation*, pages 311–320. Springer: Berlin.

Y. Wong and R. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the 2006 Human Language Technology Conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL-2006)*.

L. Zettlemoyer and M. Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of UAI.*