Semantic composition with (Robust) Minimal Recursion Semantics

Ann Copestake
Computer Laboratory, University of Cambridge
JJ Thomson Avenue, Cambridge, UK
aac@cl.cam.ac.uk

Abstract

We discuss semantic composition in Minimal Recursion Semantics (MRS) and Robust Minimal Recursion Semantics (RMRS). We demonstrate that a previously defined formal algebra applies to grammar engineering across a much greater range of frameworks than was originally envisaged. We show how this algebra can be adapted to composition in grammar frameworks where a lexicon is not assumed, and how this underlies a practical implementation of semantic construction for the RASP system.

1 Introduction

Minimal Recursion Semantics (MRS: Copestake et al. (2005)) is a flat semantic representation which factors semantics into elementary predications (EPs) and allows for underspecification of scope. It has been widely used, especially for HPSG. Robust Minimal Recursion Semantics (RMRS: Copestake (2003)) is a variant of MRS which takes this factorisation further to allow underspecification of relational information as well. While MRS has generally been used with hand-built HPSG grammars, RMRS is also suitable for use with shallower approaches to analysis, including part-of-speech tagging, noun phrase chunking and stochastic parsers which operate without detailed lexicons. MRSs can be converted into RMRSs: RMRS output from shallower systems is less fully specified than the output from deeper systems, but in principle fully compatible. In our work, the semantics produced by a deep grammar is taken as normative when developing semantic representations from shallower processing. For English, the target semantic representations are those produced by the English Resource Grammar (ERG, Flickinger (2000)). The MRS/RMRS approach has been adopted as a common framework for the DELPH-IN initiative (Deep Linguistic Processing with HPSG: `http://www.delph-in.net`).

An algebra for MRS was defined by Copestake et al. (2001) (henceforth CLF) and forms the starting point for the work reported here.

The aim of CLF was to formalise the notion of semantic composition within grammars expressed in a typed feature structure (TFS) logic. Here, we extend that work to non-lexicalist approaches and also describe how the formal principles of composition used in MRS can be adapted to produce a formalism for RMRS composition. Thus we demonstrate that the algebra applies to grammar engineering across a much wider range of frameworks than was originally envisaged. Besides its theoretical interest, this result has practical benefits when combining multiple processing systems in that it allows compatible semantic representations at a phrasal level as well as at a sentence level.

The next section (§2) describes the most important features of MRS, RMRS and the earlier work on the algebra. We then outline how the algebra can be used for implementing deep non-TFS approaches (§3) and explain how it works with RMRS (§4). This is followed by discussion of the extension to grammars without a detailed lexicon (§5). To briefly illustrate the practical applications, section (§6) outlines how RMRS semantics is constructed from RASP (Robust accurate domain-independent statistical parsing: Briscoe and Carroll (2002)).

2 MRS, RMRS and the algebra

Details of MRS, RMRS and the algebra are given in the cited papers, but we will briefly introduce them here for convenience. Fig. 1 illustrates an MRS from a deep grammar (based on the ERG output, but simplified for expository purposes), an equivalent RMRS and a very underspecified RMRS, derived from a POS tagger.

MRS achieves a flat representation via the use of labels on EPs, thus factoring out scopal relationships. Scope constraints (HCONS) are shown as qeq relationships ($=_q$ equality modulo quantifiers: the

Proceedings of the 5th Workshop on Important Unresolved Matters, pages 73–80,
Ann Arbor, June 2005.

MRS representation:
$l0$: _the_q$(x0, h01, h02), l1$: _fat_j$(x1), l2$: _cat_n$(x2), l3$: _sit_v_1$(e3, x3), l4$: _on_p$(e4, e41, x4)$,
$l5$: _a_q$(x5, h51, h52), l6$: _mat_n_1$(x6)$,
$h01 =_q l1, h51 =_q l6$
$x0 = x1 = x2 = x3, e3 = e41, x4 = x5 = x6, l1 = l2, l3 = l4$
RMRS equivalent to the MRS above:
$l0$: $a0$: _the_q$(x0), l0$: $a0$: RSTR$(h01), l0$: $a0$: BODY$(h02), l1$: $a1$: _fat_j$(x1), l2$: $a2$: _cat_n$(x2)$,
$l3$: $a3$: _sit_v_1$(e3), l3$: $a3$: ARG1$(x31), l4$: $a4$: _on_p$(e4, e41, x4), l4$: $a4$: ARG1$(e41), l4$: $a4$: ARG2$(x4)$,
$l5$: $a5$: _a_q$(x5), l5$: $a5$: RSTR$(h51), l5$: $a5$: BODY$(h52), l6$: $a6$: _mat_n_1$(x6)$,
$h01 =_q l1, h51 =_q l6$
$x0 = x1 = x2 = x3, e3 = e41, x4 = x5 = x6, l1 = l2, l3 = l4$
Highly underspecified RMRS output:
$l0$: $a0$: _the_q$(x0), l1$: $a1$: _fat_j$(x1), l2$: $a2$: _cat_n$(x2), l3$: $a3$: _sit_v$(e3), l4$: $a4$: _on_p$(e4)$,
$l5$: $a5$: _a_q$(x5), l6$: $a6$: _mat_n$(x6)$

Figure 1: MRS and RMRS for *the fat cat sat on a mat*

details are not important to understand this paper). In MRS, implicit conjunction is indicated by equality between labels. For instance, the labels on $l1$: fat(x) and $l2$: cat1(x) are equated. In this figure, we show MRS using explicit equalities (eqs: $=$) rather than coindexation of variables since this corresponds to the formalism used in the algebra.

RMRS uses the same approach to scope but adopts a variant of a neo-Davidsonian representation, where arguments (ARGs) are represented as distinct elements. In the very underspecified RMRS at the bottom of Fig.1, no relational information is known so there are no ARGs. Separating out ARGs from the EPs and allowing them to be omitted permits a straightforward notion of a specificity hierarchy in terms of information content. ARGs may also be underspecified: e.g., ARGn indicates that there is some argument relationship, but it is unknown whether it is an ARG1, ARG2 or ARG3. In the version of RMRS described in this paper, the ARGs are related to the main EPs via an 'anchor' element. An EP and its associated ARGs share a unique anchor. This version of RMRS uses exactly the same mechanism for conjunction as does MRS: the anchor elements are required so that ARGs can still be associated with a single EP even if the label of the EP has been equated with another EP. This is a change from Copestake (2003): the reasons for this proposal are discussed in §4, below. The conjunction information is not available from a POS tagger alone and so is not present in the second RMRS in Fig.1.

The naming convention adopted for the relations (e.g., _sit_v) allows them to be constructed without access to a lexicon. '_v' etc are indications of the coarse-grained sense distinctions which can be inferred from part-of-speech information. Deep grammars can produce finer-grained sense distinctions, indicated by '_1' etc, and there is an implicit hierarchy such that _sit_v_1 is taken as being more specific than _sit_v. However, in what follows, we will use simple relation names for readability. MRS and RMRS both assume an inventory of features on variables which are used to represent tense etc, but these will not be discussed in this paper.

2.1 The MRS algebra

In the algebra introduced by CLF, semantic structures (SEMENTS) for phrases consist of five parts:

1. Hooks: can be thought of as pointers into the relations list. In a full grammar, hooks consist of three parts: a label (l), an index (i) and an external argument (omitted here for simplicity).
2. Slots: structures corresponding to syntactic/semantic unsaturation — they specify how the semantics is combined. A slot in one sign is instantiated by being equated with the hook of another sign. (CLF use the term 'hole' instead of 'slot'.) For the TFS grammars considered in CLF, the slot corresponds to the part of the TFS accessed via a valence feature. The inventory of slot labels given by CLF is SUBJ, SPR, SPEC, COMP1, COMP2, COMP3 and MOD.

3. rels: The bag of EPs.
4. hcons: qeq constraints ($=_q$).
5. eqs: the variable equivalences which are the results of equating slots and hooks.

SEMENTs are: $[l, i]\{slots\}[eps][hcons]\{eqs\}$.

Some rules contribute their own semantics (construction semantics: e.g., compound nouns). However, the MRS approach requires that this can always be treated as equivalent to having an additional daughter in the rule. Thus construction semantics need not be considered separately in the formal algebra, although it does result in some syntactically binary rules being semantically ternary (and so on).

The principles of composition are:

1. A (syntactically specified) slot in one structure (the daughter which corresponds to the **semantic head**) is filled by the hook of the other structure (by adding equalities).
2. The hook of the phrase is the semantic head's hook.
3. The eps of the phrase is equal to appending the eps of the daughters.
4. The eqs of the phrase is equal to appending the eqs of the daughters plus any eqs contributed by the filling of the slot.
5. The slots of the phrase are the unfilled slots of the daughters (although see below).
6. The hcons of the phrase is equal to appending the hcons of the daughters.

Formally, the algebra is defined in terms of a series of binary operations, such as op_{spec}, which each correspond to the instantiation of a particular labelled slot.

Fig. 2 illustrates this. The hook of *cat* instantiates the SPEC slot of *a*, which is the semantic head (though not the syntactic head in the ERG). This leads to the equalities between the variables in the result. Since the SPEC slot has been filled, it is not carried up to the phrase. Thus, abstractly at least, the semantics of the HPSG specifier-head rule corresponds to op_{spec}.[1]

[1]As usual in MRS, in order to allow scope underspecification, the label l4 of the quantifier's hook is not coindexed with any EP.

The MRS algebra was designed to abstract away from the details of the syntax and of the syntax-semantics interface, so that it can be applied to grammars with differing feature geometry. The assumption in CLF is simply that the syntax selects the appropriate *op* and its arguments for each application. i.e., semantic operations are associated with HPSG constructions so that there is a mapping from the daughters of the construction to the arguments of the operation. The algebra does not attempt to completely replicate all aspects of semantic construction: e.g., the way that the features (representing tense and so on) are instantiated on variables is not modelled. However, it does constrain semantic construction compared with the possibilities for TFS semantic compositional in general. For instance, as discussed by CLF, it enforces a strong monotonicity constraint. The algebra also contributes to limiting the possibilities for specification of scope. These properties can be exploited by algorithms that operate on MRS: e.g., generation, scope resolution.

2.2 The MRS algebra and the syntax-semantics interface

CLF did not discuss the syntax-semantics interface in detail, but we do so here for two reasons. Firstly, it is a preliminary for discussing the use of the algebra in frameworks other than HPSG in the following sections. Secondly, as CLF discuss, the constraints that the algebra imposes cannot be fully implemented in a TFS. Thus, for grammar engineering in TFS frameworks, an additional automatic checker is needed to determine whether a grammar meets the algebra's constraints. This requires specification of the syntax-semantics interface so that the checker can extract the slots from the TFSs and determine the slot operation(s) corresponding to a rule.

Unfortunately, CLF are imprecise about the algebra in several respects. One problem is that they gloss over the issue of slot propagation in real grammars. CLF state that for an operation op_x, the slot corresponding to op_x on the semantic head is instantiated and all other slots appear on the result. For instance, the definition of op_{spec} states that for all labels $l \neq spec$: $slot_l(op_{\text{spec}}(a_1, a_2)) = slot_l(a_1) \cup slot_l(a_2)$. However, this is inadequate for real grammars, if a simple correspondence between the slot names and the valence paths in the feature structure

	hook	slots	rels	eqs	hcons
cat :	$[l1, x1]$	$\{\}$	$[l1 : \text{cat}(x1)]$	$\{\}$	$[]$
a :	$[l4, x2]$	$\{[l3, x2]_{spec}\}$	$[l2 : \text{a}(x2, h2, h3)]$	$\{\}$	$[h2 =_q l3]$
a cat :	$[l4, x2]$	$\{\}$	$[l2 : \text{a}(x2, h2, h3), l1 : \text{cat}(x1)]$	$\{l3 = l1, x2 = x1\}$	$[h2 =_q l3]$

Figure 2: Example of the MRS algebra

is assumed. For instance, the passive rule involves coindexing a COMP in the original lexical sign with the SUBJ of the passive (informally, the complement 'becomes' the subject).

There are two ways round this problem. The first is to keep the algebra unchanged, but to assume that, for instance, the subject-head grammar rule corresponds to op_{subj} in the algebra for non-passivized cases and to op_{comp1} for passives of simple transitives and so on. Though possible formally, this is not in accord with the spirit of the approach since selection of the appropriate algebra operation in the syntax-semantics interface would require non-local information. Practically, it also precludes the implementation of an algebra checker, since keeping track of the slot uses would be both complex and grammar-specific. The alternative is to extend the algebra to allow for slot renaming. For instance, $op_{\text{comp1-subj}}$ can be defined so that the COMP1 slot on the daughter is a SUBJ slot on the mother.

1. For all labels $l \neq comp1, l \neq subj$: $slot_l(op_{\text{comp1-subj}}(a)) = slot_l(a)$
2. $slot_{\text{subj}}(op_{\text{comp1-subj}}(a)) = slot_{\text{comp1}}(a)$

This means extending the inventory of operations, but the choice of operation is then locally determinable from the rule (e.g., the passive rule would specify $op_{\text{comp1-subj}}$ to be its operation).

Another issue arises in grammars which allow for optional complements. For instance, one approach to a verb like *eat* is to give it a single lexical entry which corresponds to both transitive and intransitive uses. The complement is marked as optional and the corresponding variable in the semantics is assumed to be discourse bound if there is no syntactic complement in the phrase. Optional complements can be discharged by a construction. This approach is (arguably) appropriate for *eat* because the intransitive use involves an implicit patient (e.g., *I already ate* means *I already ate something*), in contrast to a verb like *kick*. CLF do not discuss optionality but it can be formalised in the algebra in terms of a construction-specified sement which has a hook containing the discourse referent and is otherwise empty. For instance, an optional complement construction corresponds to $op_{\text{comp1}}(a_1, a_2)$ where a_1 is the head (and the only daughter appearing in the TFS for the construction) and a_2 is stipulated by the rule to be $[l, d]\{\}[][]\{\}$, where d is the discourse-bound referent.

3 The algebra in non-lexicalist grammars

CLF motivate the MRS algebra in terms of formalisation of the semantics of constraint-based grammars, such as HPSG, but, as we outline here, it is equally applicable to non-lexicalist frameworks. With a suitable definition of the syntax-semantics interface, the algebra can be used with non-TFS-based grammars. Fig. 3 sketches an example of MRS semantics for a CFG. A syntax-semantic interface component of the rule (shown in the second line of the figure) specifies the ops and their daughters: the IOBJ slot of the verb is instantiated with the first NP's hook and the OBJ slot of the result is instantiated with the hook of the second NP. The idea is extremely similar to the use of the algebra with TFS but note that with the addition of this syntax-semantic interface, the algebra can be used directly to implement semantic composition for a CFG.

This still relies on the assumption that all slots are known for every lexical item: semantically the grammar is lexicalist even though it is not syntactically. In fact this is analogous to semantic composition in GPSG (Gazdar et al., 1985) in that conventional lambda calculus also assumes that the semantic properties are known at the lexical level.

4 RMRS composition with deep grammars

The use of the CLF algebra in RMRS composition with deep lexicalist grammars is reasonably straight-

VP -> Vditrans NP1 NP2

$op_{\text{obj}}(op_{\text{iobj}}(\text{Vditrans}, \text{NP1}), \text{NP2})$

MRSs for application of the rule to *give a cat a rat*.

	hook	slots	rels	eqs
give :	$[l1, e1]$	$\{[l1, x12]_{subj}, [l1, x13]_{obj}, [l1, x14]_{iobj}\}$	$[l1 : \text{give}(e1, x12, x13, x14)]$	$\{\}$
a cat :	$[l4, x2]$	$\{\}$	$[l2 : \text{a}(x2, h2, h3), l1 : \text{cat}(x1)]$	$\{l3 = l1, x2 = x1\}$
a rat :	$[l7, x5]$	$\{\}$	$[l5 : \text{a}(x5, h5, h6), l4 : \text{rat}(x4)]$	$\{l6 = l4, x5 = x4\}$
iobj :	$[l1, e1]$	$\{[l1, x12]_{subj}, [l1, x13]_{obj}\}$	$[l1 : \text{give}(e1, x12, x13, x14), l2 : \text{a}(x2, h2, h3), l1 : \text{cat}(x1)]$	$\{l3 = l1, x2 = x1, l1 = l4, x14 = x2\}$
obj :	$[l1, e1]$	$\{[l1, x12]_{subj}\}$	$[l1 : \text{give}(e1, x12, x13, x14), l2 : \text{a}(x2, h2, h3), l1 : \text{cat}(x1), l5 : \text{a}(x5, h5, h6), l4 : \text{rat}(x4)]$	$\{l3 = l1, x2 = x1, l1 = l4, x14 = x2, l1 = l7, x13 = x5\}$

Figure 3: MRS algebra with a CFG (hcons omitted for clarity)

forward.[2] The differences between MRS and RMRS are that RMRS uses anchors and factors out the ARGs. Thus for RMRS, we need to redefine the notion of a semantic entity from the MRS algebra to add anchors. An RMRS EP thus contains:

1. a handle, which is the label of the EP
2. an anchor (a)
3. a relation
4. up to one argument of the relation

Hooks also include anchors: $\{[l, a, i]\}$ is a hook. Instead of the rels list only containing EPs, such as l1:chase(e,x,y), it contains a mixture of EPs and ARGs, with associated anchors, such as l1:a1:chase(e), l1:a1:ARG1(x), l1:a1:ARG2(y). But formally ARGs are EPs according to the definition above, so this requires no amendment of the algebra. Fig. 4 shows the RMRS version of Fig. 2.

As mentioned above, earlier forms of RMRS used an explicit representation for conjunction: the in-group, or in-g. Reasons to avoid explicit binary conjunction were discussed with respect to MRS by Copestake et al. (2005) and readers are referred to that paper for an explanation: essentially the problem is that the syntactic assumptions influence the semantic representation. e.g., the order of combination of intersective modifiers affects the semantic representation, though it has no effect on denotation. The binary in-g suffers from this problem.

One alternative would be to use an n-ary conjunction symbol. However such representations cannot be constructed compositionally if modification is binary branching as there is no way of incrementally adding the conjuncts. Another option we considered was the use of, possibly redundant, conjunction relations associated with each element which could be combined to produce a flat conjunction. This leads to a spurious in-g in the case where there is no modifier. This looks ugly, but more importantly, does not allow for incremental specialisation, although the demonstration of this would take us too far from the main point of this paper.

We therefore assume a modified version of RMRS which drops in-g symbols but uses anchors instead. This means that RMRS and MRS TFS grammars can be essentially identical apart from lexical types. Furthermore, it turns out that, for composition without a lexicon, an anchor is needed in the hook regardless of the treatment of conjunction (see below).

5 RMRS composition without a lexicon

We now discuss the algebra for grammars which do not have access to subcategorization information and thus are neither syntactically nor semantically lexicalist. We concentrate in particular on composition for the grammar used in the RASP system. RASP consists of a tokenizer, POS tagger, lemmatizer, tag sequence grammar and statistical disambiguator. Of the robust analysers we have looked at, RASP pro-

[2] Current DELPH-IN grammars generally construct MRSs which may be converted into RMRSs. However, RMRS has potential advantages, for instance in allowing more extensive lexical underspecification than is possible with MRS: e.g., (Haugereid, 2004).

	hook	slots	rels	eqs	hcons
cat :	$[l1, a1, x1]$	$\{\}$	$[l1 : a1 : \text{cat}(x1)]$	$\{\}$	$[]$
a :	$[l4, a2, x2]$	$\{[l3, a2, x2]_{spec}\}$	$[l2 : a2 : \text{a}(x2),$ $l2 : a2 : \text{rstr}(h2), l2 : a2 : \text{body}(h2)]$	$\{\}$	$[h2 =_q l3]$
a cat :	$[l4, a4, x2]$	$\{\}$	$[l1 : a1 : \text{cat}(x1), l2 : a2 : \text{a}(x2),$ $l2 : a2 : \text{rstr}(h2), l2 : a2 : \text{body}(h2)$	$\{l3 = l1,$ $x2 = x1]$	$[h2 =_q l3]$

Figure 4: Example of the RMRS algebra.

vides the biggest challenge for the RMRS approach because it provides quite detailed syntactic analyses which are somewhat dissimilar to the ERG: it is an intermediate rather than a shallow processor. The RMRS approach can only be fully successful to the extent that it abstracts away from the differences in syntactic analyses assumed by different systems, so intermediate processors are more difficult to deal with than shallow ones.

Instead of normal lexical entries, RASP uses the POS tags for the words in the input. For the example in Fig. 1, the output of the POS tagging phase is:

the_AT fat_JJ cat_NN1 sit+ed_VVD on_II a_AT1 mat_NN1

The semantics associated with the individual words in the sentence can be derived from a 'lexicon' of POS tags, which defines the EPs. Schematically:

AT	lexrel_q(x)	NN1	lexrel_n(x)
AT1	lexrel_q(x)	VVD	lexrel_v(e_{past})
JJ	lexrel_j(x)	II	lexrel_p(e)

Here, 'lexrel' is a special symbol, which is to be replaced by the individual lemma (with a leading underscore) — e.g., lexrel_v(e_{past}) yields l1:a1:_sit_v(e). Producing the semantics from the tagger output and this lexicon is a simple matter of substitution. All EPs are labelled with unique labels and all variables are different unless repeated in the same lexical entry.

If the analysis were to stop at POS tagging, the semantic composition rules would apply trivially. There are no slots, the hooks are irrelevant and there are no equalities. The composition principle of accumulation of elementary predications holds, so the semantics of the result involves an accumulation of the rels (see the example at the bottom of Fig. 1).

When using the full RASP parser, although we cannot expect to obtain all the details available from deep grammars, we can derive some relational structure. For instance, given a sentence such as *the cat chased the rat*, it should be possible to derive the ARG1 and ARG2 for *chase* by associating the ARG1 with the application of the `S/np_vp` RASP rule (i.e., `S->NP VP`) and the ARG2 with the application of the `V1/v_np` rule. But since there can be no slot information in the lexical structures (at least not for open-class words), it is necessary to modify the lexicalist approach to semantics taken so far.

We assume that both the ARGs and the slots are specified at a phrasal level rather than lexically. As mentioned in §2.1, the MRS algebra allows for rules to contribute semantics as though they were normal phrases. The central idea in the application of the algebra to RASP is to make use of construction semantics in all rules. Fig. 5 illustrates this with the `V1/v_np` rule (the NP has been simplified for clarity) assuming the same sort of syntax-semantics interface specification as shown earlier for the CFG. This is semantically ternary because of the rule semantics. The rule has an ARG2 slot plus a slot R which is instantiated by the verb's hook. In effect, the rule adds a slot to the verb.

It is necessary for the anchor of the argument-taking structure to be visible at all points where arguments may be attached. For instance, in the example above, the anchor of the verb *chase* has to be accessible when the ARG1 relation is introduced. Although generally the anchor will correspond to the anchor of the semantic head daughter, this is not the case if there is a scopal modifier (consider *a cat did not chase a rat*: the ARG1 must be attached to *chase* rather than to *not*). This is illustrated by *not sleep* in Fig. 6. Because *not* is associated with a unique tag in RASP, it can be assigned a slot and an ARG1 directly. The anchor of the result is equated with the label of *sleep* and thus the subject ARG1 can be appropriately attached. So the hook would have to include an anchor even if explicit conjunction were used instead of equating labels.

$$\text{VP -> V NP}$$
$$op_{\text{arg2}}(op_{\text{r}}(\text{rule, V}), \text{NP})$$

chase :	$[l1, a1, e1]$	$\{\}$	$[l1 : a1 : \text{chase}(e1)]$	$\{\}$
rule :	$[l2, a2, e2]$	$\{[l2, a2, e2]_r, [l4, a4, x2]_{arg2}\}$	$[l2 : a2 : \text{ARG2}(x2)]$	$\{\}$
(rule V)/r :	$[l2, a2, e2]$	$\{[l4, x2]_{arg2}\}$	$[l2 : a1 : \text{ARG2}(x2), l1 : a1 : \text{chase}(e1)]$	$\{l1 = l2, e2 = e1\}$
it :	$[l3, a3, x3]$	$\{\}$	$[l3 : a3 : \text{pron}(x3)]$	$\{\}$
chase it :	$[l2, a2, e2]$	$\{\}$	$[l2 : a2 : \text{ARG2}(x2), l1 : a1 : \text{chase}(e1), l3 : a3 : \text{pron}(x3)]$	$\{l1 = l2, e2 = e1, l4 = l3, x2 = x3\}$

Figure 5: RASP-RMRS algebra (hcons omitted)

not :	$[l1, a2, e2]$	$\{[l2, a3, e2]_{mod}\}$	$[l1 : a1 : \text{not}(e2), l1 : a1 : \text{ARG1}(h4)]$	$\{\}$	$[h4 =_q l2]$
sleep :	$[l2, a2, e2]$	$\{\}$	$[l2 : a2 : \text{sleep}(e2)]$	$\{\}$	$[]$
not sleep :	$[l1, a2, e2]$	$\{\}$	$[l1 : a1 : \text{not}(e2), l1 : a1 : \text{ARG1}(h4), l2 : a2 : \text{sleep}(e2)]$	$\{\}$	$[h4 =_q l3]$

Figure 6: RASP-RMRS illustrating the use of the anchor

6 Experiments with RASP-RMRS

In this section, we outline the practical implementation of the algebra for RASP-RMRS. The RASP tag sequence grammar is formally equivalent to a CFG: it uses phrase structure rules augmented with features. As discussed, the algebra requires that ops are specified for each rule application, and the easiest way of achieving this is to associate semantic composition rules with each rule name. Composition operates on the tree output from RASP, e.g.,:

```
(|T/txt-sc1/----|
 (|S/np_vp|
  (|NP/det_n1| |Every:1_AT1|
   (|N1/n| |cat:2_NN1|))
  (|V1/v| |bark+ed:3_VVD|)))
```

Composition operates bottom-up: the semantic structures derived from the tags are combined according to the semantics associated with the rule. The implementation corresponds very directly to the algebra, although the transitive closure of the equalities is computed on the final structure, since nothing requires that it be available earlier.

The notation used to specify semantics associated with the rules incorporates some simplifications to avoid having to explicitly specify the slot and ops. The specification of equalities between variables and components of the individual daughters' hooks is a convenient shorthand for the full algebra.

```
rule V1/v_np
daughters V NP
semhead V
hook [l,a,e] rels {l:a:ARG2(x)}
eqs {x=NP.index,l=V.label,
     a=V.anchor}
```

If no semantic rule is specified corresponding to a rule used in a tree, the rels are simply appended. Semantic composition is thus robust to omissions in the semantic component of the grammar. In fact, semantic rules can be constructed semi-automatically, rather than fully manually, although we do not have space to discuss this in detail here.

There are cases of incompatibility between RASP-RMRS and ERG-RMRS. For example, the ERG treats *it* as expletive in *it rains*: the lexical entry for *rain* specifies an expletive subject (i.e., a semantically empty *it*). RASP makes no such distinction, since it lacks the lexical information and thus the sentence has extraneous relations for the pronoun and an incorrect ARG1 for *rain*. This is an inevitable consequence of the lack of lexical information in RASP. However, from the perspective of the evaluation of the revised algebra, the issue is whether there are any cases where compositional construction of RASP-RMRSs which match ERG-RMRSs is impossible due to the restrictions imposed by the algebra. No such cases have been found.

7 Related work

Bos et al. (2004) and Bos (2005) derive semantic interpretations from a wide-coverage categorial grammar. There are several differences between this and RASP-RMRS, but the most important arise from the differences between CCG and RASP. The CCG parser relies on having detailed subcategorization information (automatically derived from the CCG Bank which was semi-automatically constructed from the Penn Treebank), and thus semantic construction can assume that the arity of the predicate is lexically available. However, because CCG is purely lexicalist, phenomena that we expect to have construction semantics (e.g., compound nouns, larger numbers) have to be dealt with in a post-parsing phase rather than compositionally.

Spreyer and Frank (2005) demonstrate RMRS construction from TIGER dependencies, but do not attempt to match a deep parser output.

8 Conclusion

We have demonstrated that the MRS algebra, originally intended as a formalisation of some aspects of semantic composition in constraint-based grammars, can be extended to RMRS and other types of grammar framework and can be used as the basis of a full implementation of composition. The algebra can thus be used much more widely than originally envisaged and could be exploited by a wide range of parsers. Useful properties concerning monotonicity and scope (see Fuchss et al. (2004)) are thus guaranteed for a range of grammars. Phrasal-level compatibility of RMRS (to the extent that this is syntactically possible) is also an important result. The main practical outcome of this work so far has been a semantic component for the RASP system which produces representations compatible with that of the ERG without compromising RASP speed or robustness. RASP-RMRSs have already been used in systems for question answering, information extraction, email response, creative authoring and ontology extraction (e.g., Uszkoreit et al. (2004), Watson et al. (2003), Herbelot and Copestake (2006)).

Acknowledgements

This work was funded by EPSRC (EP/C010035/1). I am very grateful to the anonymous reviewers for their insightful comments, which sadly I have had to ignore due to the constraints of time and space. I hope to address them in a later paper.

References

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran and Julia Hockenmaier 2004. Wide-Coverage Semantic Representations from a CCG Parser. COLING '04, Geneva.

Johan Bos. 2005. Towards Wide-Coverage Semantic Interpretation. Sixth International Workshop on Computational Semantics IWCS-6. 42–53.

Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. LREC-2002, Las Palmas, Gran Canaria.

Ann Copestake, Alex Lascarides, and Dan Flickinger. 2001. An Algebra for Semantic Construction in Constraint-based Grammars. ACL-01, Toulouse.

Ann Copestake. 2003. Report on the design of RMRS. DeepThought project deliverable.

Ann Copestake, Dan Flickinger, Ivan Sag, and Carl Pollard. 2005. Minimal Recursion Semantics: An introduction. *Research in Language and Computation* 3(2–3), 281–332.

Dan Flickinger 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*,**6:1**,15–28.

Ruth Fuchss, Alexander Koller, Joachim Niehren, and Stefan Thater 2004. Minimal Recursion Semantics as Dominance Constraints: Translation, Evaluation, and Analysis. Proceedings of the 42nd ACL, Barcelona.

Gerald Gazdar, Ewan Klein, Geoffrey Pullum and Ivan Sag 1985. Generalized Phrase Structure Grammar. Basil Blackwell, Oxford

Petter Haugereid 2004. Linking in Constructions. HPSG2004, Leuven.

Aurelie Herbelot and Ann Copestake 2006. Acquiring Ontological Relationships from Wikipedia Using RMRS. ISWC 2006 Workshop on Web Content Mining with Human Language Technologies, Athens, Georgia.

Kathrin Spreyer and Anette Frank. 2005 Projecting RMRS from TIGER Dependencies. HPSG 2005, Lisbon. 354–363.

Hans Uszkoreit, Ulrich Callmeier, Andreas Eisele, Ulrich Schfer, Melanie Siegel, Jakob Uszkoreit. 2004. Hybrid Robust Deep and Shallow Semantic Processing for Creativity Support in Document Production. KONVENS 2004, Vienna, Austria, 209–216.

Rebecca Watson, Judita Preiss and EJ Briscoe. 2003. The Contribution of Domain-independent Robust Pronominal Anaphora Resolution to Open-Domain Question-Answering. Int. Symposium on Reference Resolution and its Application to Question-Answering and Summarisation, Venice.