Inheriting Verb Alternations

Adam Kilgarriff
Longman Dictionaries
Burnt Mill
Harlow, Essex CM20 2JE
England

Abstract

The paper shows how the verbal lexicon can be formalised in a way that captures and exploits generalisations about the alternation behaviour of verb classes. An alternation is a pattern in which a number of words share the same relationship between a pair of senses. The alternations captured are ones where the different senses specify different relationships between syntactic complements and semantic arguments, as between *bake* in "John is baking the cake" and "The cake is baking". The formal language used is DATR. The lexical entries it builds are as specified in HPSG. The complex alternation behaviour shared between families of verbs is elegantly represented in a way that makes generalisations explicit, avoids redundancy, and offers practical benefits to computational lexicographers.

1 Introduction

The paper shows how the verbal lexicon can be formalised in a way that captures and exploits generalisations about the alternation behaviour of verb classes. An alternation is a pattern in which a number of words share the same relationship between a pair of senses. The kinds of alternations to be captured are ones where the different senses specify different relationships between syntactic complements and semantic arguments, as in the relation between *bake* in "John is baking the cake" and "John is baking", or between *melt* in "the chocolate melted" and "Mary melted the chocolate".[1] Given that compactness and non-redundancy are a desideratum of theoretical descriptions, the different usage-types for *bake* and *wipe* should not require us to introduce different primitives into the lexicon. Moreover, as the alternations are shared with other verbs, they should be described at some general node in a hierarchically organised lexicon, and inherited.

DATR is a formal language in which the such relationships and generalisations can be simply stated.

Much has been written about verb alternations and their syntactic corollaries. Here we do not add to the evidence or construct new theory, but simply formalise other people's accounts: those of [Atkins *et al.*, 1986] and [Levin and Rappoport Hovav, 1991]. The first investigates the range of alternations between transitive and intransitive forms of verbs. The second, titled *Wiping the Slate Clean*, explores the relations between meaning and subcategorisation possibilities for 'wipe' verbs, 'clean' verbs, and related groupings. The language used is DATR, a default inheritance formalism designed for lexical representation. We follow Levin and Rappoport Hovav in taking a distinct subcategorisation frame as defining a distinct word sense, and also in working with commonsense verb classes such as 'cooking verbs', since classes such as this serve to predict the alternations a verb participates in with some accuracy.

An important constraint is that the lexical entries are of a kind specified by a grammar formalism, so can be used for parsing and semantic interpretation. The formalism chosen in this paper is HPSG [Pollard

*I would like to thank Gerald Gazdar and Roger Evans for their many valuable comments, and SERC for the grant under which the work was undertaken.

[1] The morphosyntactic distinctions between, for example, *bake* and *is baking* are not addressed here. Extensive DATR treatments of morphology are provided in various papers in [Evans and Gazdar, 1990].

and Sag, 1987].

Below we present detailed formal accounts for alternations involving cooking verbs and physical-process verbs. After motivating the DATR treatment and considering related work, we describe how verb entries appear in HPSG, then represent alternations as mappings between HPSG lexical entries, then introduce the main constructs of DATR and define a translation from HPSG notation to DATR. Finally we build a DATR inheritance network which represents the alternate verb forms by inference, without the lexicographer having to explicitly say anything about them.

The analysis presented in this paper is a part of a larger lexicon fragment which describes a further five alternations relating seven verb classes and formalises much of the structure described in both articles. The complete fragment, illustrated in Fig. 1. is presented in full in [Kilgarriff, 1992].

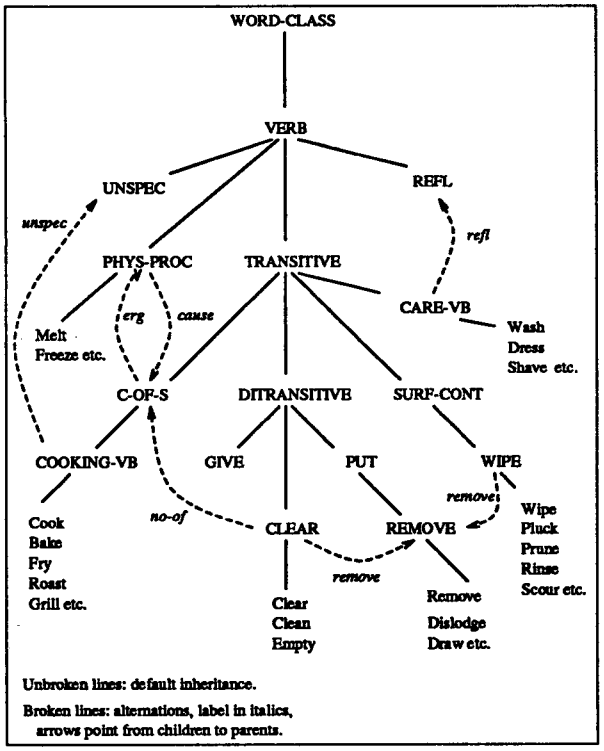

Figure 1: Verb taxonomy

1.1 Why DATR?

As 'lexicalism' — the doctrine that the bulk of the information about the behaviour of words should be located in the lexicon — has become popular in computational and theoretical linguistics, so formalisms for expressing lexical information have been developed. The syntax, semantics and morphology of most words is shared with that of many others, so the first desideratum for any such formalism is to provide a mechanism for stating information just once, in such a way that it is defined for large numbers of words. Inheritance networks serve this purpose. If words are arranged into a taxonomy or some other form of network, then a fact which applies to a class of words can be stated at a nonterminal node in the network and inherited by the words to which it applies. Work in knowledge representation has addressed questions of different kinds of network, and the kinds of machinery needed to retrieve inherited information, in detail (see, e.g., [Brachman and Levesque, 1985]).

The next requirement is that exceptions and subregularities can be expressed. It must be possible to describe concisely the situation where a word or class of words are members of some superclass, and share the regular characteristics of the superclass in most respects, but have different values for some feature or cluster of features. Several lexical representation formalisms addressing these desiderata have been proposed, e.g. DATR [Evans and Gazdar 1989a, 1989b, 1990]; LRL [Copestake, 1992]; [Russell *et al.* 1991]. The work described here uses DATR.

DATR has certain desirable formal and computational properties. It is a formal language with a declarative semantics. Retrieving values for queries involves no search. Multiple inheritance specifications are always orthogonal, so a word may inherit from more than one place, but any fact about that word has the place it is to be inherited from uniquely specified. The problem of different ancestors providing contradictory values, often associated with multiple default inheritance, is thereby avoided, yet the kinds of generalisation most often associated with the lexicon can still be simply stated. To date it has been used to express syntactic, morphological, phonological and a limited amount of semantic lexical information [Evans and Gazdar, 1990; Cahill and Evans, 1990; Gibbon, 1990; Cahill, 1993]. Verb alternations have not previously received a DATR treatment.

1.2 Related work

The work described here is at the meeting-point of lexical representation languages (as discussed above), lexical semantics (as in Atkins et al. and Levin and Rappoport Hovav; see also [Levin, 1991]) and formal accounts of alternations (see particularly [Dowty, 1979]).

Recent work which aims to bring these three threads together in relation to the lexical representation of nouns includes [Briscoe *et al.*, 1990; Pustejovsky, 1991; Copestake and Briscoe, 1991; Kilgarriff, 1993 forthcoming; Kilgarriff and Gazdar, 1993 forthcoming]. (The latter two are companion papers to this, also using DATR in similar ways.) A paper addressing verbs is [Sanfilippo and Poznanski, 1992].

This covers some of the same alternations as this

paper, and has similar goals. The formalism it uses is LRL, the typed default unification formalism of [Copestake, 1992]. Unlike DATR, this is both a lexical representation language and a grammar formalism. Whereas, in this paper, we represent the lexicon in DATR and then construct HPSG lexical entries, Sanfilippo and Poznanski need deal with only one formalism. This has a *prima facie* advantage but also a cost: the formalism must do two jobs. DATR is designed specifically for one, and offers more flexibility in the representation of exceptions and subregularities. In LRL, multiple default inheritance is restricted to the cases where there is no clash, with the condition enforced by a checking procedure, in contrast to DATR where the orthogonal nature of inheritance required by the syntax means that the problem does not arise. Also, LRL default inheritance must operate within the constraints of a type hierarchy, and the formalism requires two kinds of inheritance, default and non-default. In DATR, inheritance is not constrained by a type hierarchy, and inheritance, default or otherwise, invokes a single mechanism.

2 An HPSG-style lexicon

The alternations to be addressed in detail here are the ones relating the transitive, which we treat as the base form, to the *ergative* ("The cake baked") and to the *unspecified object* ("John baked").

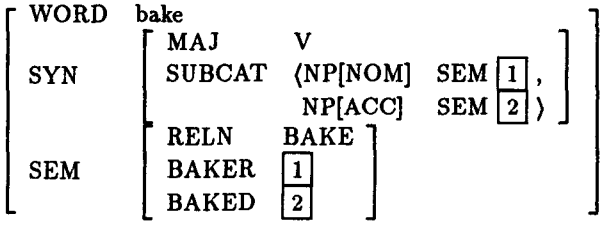

Figure 2: AVM for transitive *bake*.

Fig. 2 shows a simplified version of the HPSG lexical entry for transitive *bake*, in attribute-value matrix (AVM) notation. NP abbreviations and angle-bracket list notation, where a comma separates list elements and there is no separator between the conjuncts of a feature-structure within a list, is as in [Pollard and Sag, 1987]. The boxed variables indicate the roles the semantic arguments play in the syntactic structure.

For ergative *bake*, the same BAKE relation holds as in the base form, but now between an unspecified BAKER and a BAKED which is the subject of the sentence. The unspecified role filler is not 'bound' to a complement (i.e. any item on the SUBCAT list) but is existentially quantified (EX-Q). The ergative form is intransitive so has only one item on its SUBCAT list and the SEM of that item unifies with the BAKED, so the AVM for ergative *bake* will be as in Fig. 3. For unspecified-object *bake* in "John was

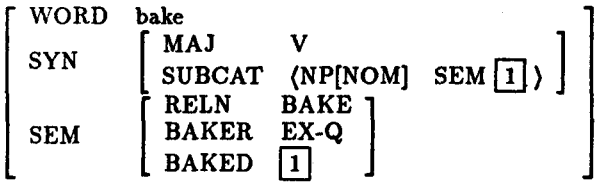

Figure 3: AVM for ergative *bake*.

baking", the subject is matched to the BAKER and it is the BAKED which is unspecified, so existentially quantified, as in Fig. 4.

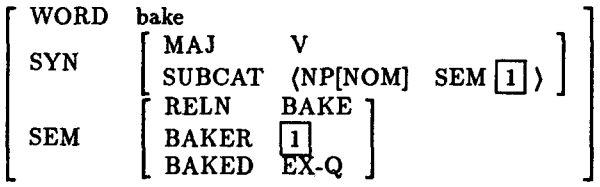

Figure 4: AVM for unspecified-object *bake*.

For *bake* and other cooking verbs, we are able to represent the extended senses directly in terms of the same predicate that applied in the base sense. We now move on to a case where this does not hold.

For *melt*, the intransitive ("The ice melted") is basic and the transitive ("Maria melted the ice") is extended, and it is not possible to define the extended sense directly in terms of the base. The transitive can be paraphrased using *cause*, "Maria caused the ice to melt" and we call the alternation 'causative'. It is clearly closely related to the ergative, and it would be possible to treat the transitive form as basic, with the ergative alternation applying. That route has not been followed for two reasons. Firstly, *melt* is a member of a class of physical-process verbs, also including *evaporate, freeze, dissolve, sublime* and *coalesce*. They all clearly have intransitive senses. They all might, in the right setting, be used transitively, but in cases such as *coalesce* the transitive is not a standard use and it would patently be inappropriate for it to be treated as a base form. If we are to stand by the intuition that these verbs form a class, and all participate in the same alternation, then all must have an intransitive base form.

Secondly, transitive *melt* introduces an aspect of meaning, call it CAUSE, which is not in any sense present in the intransitive. For *bake*, CAUSE is already a component of the meaning, whether or not the verb is being used ergatively. A default entailment of CAUSE is that its first argument, the CAUSER, has proto-agent properties [Dowty, 1991]. If intransitive *melt* were treated like ergative *bake*,

CAUSE would be a component of the meaning of intransitive *melt*. Its semantics would have an existentially quantified MELTER argument, which would be a CAUSER and which we would expect to have agent-like properties. In ergative uses of *bake*, the baking scenario still includes an agent who is doing the baking and fills the BAKER role, even though they are not mentioned. (We concern ourselves here only with cooking *bake*, not "The stones baked in the sun" and other usage-types where *bake* is behaving as a physical process verb.) In "The ice melted" there is usually no agent involved. While it might always be possible to assign a filler to the MELTER slot, perhaps "the hot temperature" or "the warm climate", they do not fit readily into the agent, CAUSER role. So we do not treat causatives as ergatives.

A standard analysis of causatives after [Dowty, 1979] as presented by [Chierchia and McConnell-Ginet, 1990, chapter 8], is

$$\lambda y \lambda x MELT/2(x, y) = \lambda y \lambda x CAUSE(x, MELT/1(y)).$$

The semantics of the causative has the predicate CAUSE, with MELT/1 re-appearing as its second argument. In addition to intransitive *melt* as shown in Fig. 5 we have causative *melt* as shown in Fig. 6. (The relation between lambda expressions and feature structures is discussed in [Moore, 1989; Kilgarriff, 1992].)

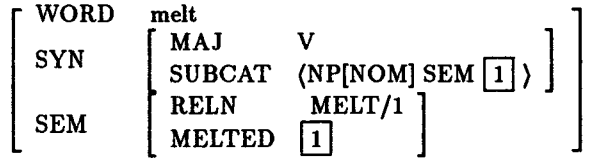

Figure 5: AVM for intransitive *melt*.

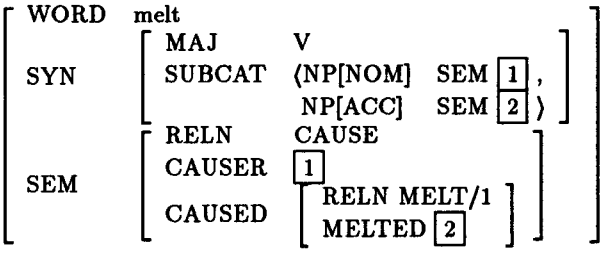

Figure 6: AVM for causative *melt*.

3 DATR: a gentle introduction

A simple DATR equation has, on its lhs, a node and a path, and, on its rhs, either a value:

```
Node1:<a b c> == value.
```

or an inheritance specification. Nodes start with capital letters, paths are sequences enclosed in angle-brackets, anything on the rhs that is not a node or a path is a value. The primary operation on a DATR description is the evaluation of a query, that is, the determination of a value associated with a given path at a given node. Where a value is not given directly, it may be inherited by following a trail: the inheritance specification on the rhs at step *n* becomes the lhs for step *n+1*. The specifications may state both node and path, node only or path only. They may also be local or global. Where they are local, the unstated node or path is as it was on the lhs, so if we have the node:

```
Node1: <a> == Node2: <x>
       <b> == Node3
       <c> == <y>.
```

then

```
Node1: <a> inherits from Node2: <x>
Node1: <b> inherits from Node3: <b>
Node1: <c> inherits from Node1: <y>.
```

(Where a number of node-path specifications for a given node are stated together, the node need not be re-iterated. The full stop is delimiter for either a single equation or such a cluster of equations.)

Where inheritance specifications are global, with the node or path on the rhs in double quotes:

```
Node4: <a> == "Node5"
       <b> == "<z>".
```

then the 'global context' node or path is picked up to complete the specification. For the purposes of this paper, the global context node and path are the initial query node and path.

When there is no lhs to exactly match a node-path pair to be evaluated, the mechanism which gives rise to DATR's nonmonotonicity comes into play. This is the 'longest leading subpath' principle. The node-path pair inherits according to the equation at the node which matches the longest leading subpath. Thus, with `Node1` as defined above,

```
Node1:<a ax ay> inherits from Node2:<x ax ay>
Node1:<b bx by> inherits from Node3:<b bx by>
Node1:<c cx cy> inherits from Node1:<y cx cy>
```

If there were any more specific paths defined at `Node1`, for

```
<a ax>,
<a ax ay>,
<b bx>, etc.,
```

then these inheritances would be overridden. Note that the match must be with the longest leading subpath. In this fragment, the queries

```
Node1: <d>,
Node1: <ax a>, and
Node1: <>
```

all fail to match and are undefined. (The other queries may also be undefined, if the trail of inheritance specifications terminates without reaching a value at some later stage, but they are not found to be undefined at this stage.)

Two particular cases of inheritance used in the paper are:

```
Node5: <>  == Node6
       <e> == Node7:<>.
```

In the first, the leading subpath to be matched is null, so this is a default of defaults: no queries will terminate at this point, since any query which does not make a more specific match will match this line and get passed on from **Node5** to **Node6**, path unchanged. This is the simplest form of inheritance, usually used to specify the basic taxonomy in a DATR theory. In the second, path element **e** is 'chopped' from the beginning of the path, so:

Node5:<e ex ey> inherits from **Node7:<ex ey>**.

4 Translations into DATR

Now we move on from describing the alternations, and describing the inheritance formalism, to representing the alternations within the formalism. The DATR translation is straightforward: AVMs can be rewritten as sets of equations which then become sets of DATR equations. DATR paths must be associated with nodes, so a node for the paths to be located at is introduced. FIRST and REST have been shortened to **fi** and **re**. DATR is not a unification formalism, and all the theory will do in relation to re-entrancies will be to mark them with matched pairs of variables, here **v1**, **v2** etc., to be interpreted as re-entrant pairs outside DATR. We introduce the feature **binding** for the variables to be the value of.[2] In order that generalisations covering BAKERs, COOKERs and FRYERs can be stated, we replace verb-specific names such as BAKER for slots on a semantic **args** list. (This does not represent a change in the semantics: the first member of the argument list of the **bake** predicate will continue to be the BAKER whatever lexical entry it occurs in. It simply allows us to express generalisations.) We use **pred** for the predicate rather than RELN. Following these changes, the (simplified) DATR lexical entry for transitive *bake* is:

```
Bake:<word> = bake
     <syn maj> = v
     <syn subcat fi sem binding> = v1
     <syn subcat re fi sem binding> = v2
     <syn subcat re re> = nil
     <sem pred> = bake
     <sem args fi binding> = v1
     <sem args re  fi binding> = v2
     <sem args fi binding> = nil.
```

[2]The feature also makes it possible to use the fact that a semantic argument has an existential-quantification (**ex-q**) binding to override the default that it is bound to a complement.

5 An inheritance hierarchy

The next task is to place the verbs in a hierarchy so generalisations need stating only once. DATR allows different kinds of information to be inherited from different places, and also allows generalisations to be overridden by either idiosyncratic facts or subregularities. The hierarchy is illustrated in Fig. 1. At the top of the tree is **WORD-CLASS**, then **VERB**, from where all verbs inherit. They all have a subject, and by default this unifies with the first item on the **args** list. There will be no call for an **INTRANSITIVE** node because all the positive information that might be stated there is true of all verbs so can be stated at the **VERB** node, and the negative information that intransitive verbs do not have direct objects is expressed by the termination of the subcat list after its first item at **VERB** (via **ARG** and **NIL**; see below). **TRANSITIVE** inherits from **VERB**, adding the default binding between second complement and second argument.

```
VERB: <> == WORD-CLASS
    <syn maj> == verb
    <syn subcat fi sem binding> == v1
    <sem args fi binding> == v1.
TRANSITIVE: <> == VERB
    <syn subcat re fi sem binding> == v2
    <sem args re fi binding> == v2.
```

List termination involves a measure of ingenuity, in order that **nil** is the value of **<syn subcat re>** and **<sem args re>** at **VERB** and **<syn subcat re re>** and **<sem args re re>** at **TRANSITIVE**, but nowhere else:[3]

```
VERB: <sem args> == ARG: <>
    <syn subcat> == COMP:<>.
    <syn subcat fi syn case> == nom
    <sem args fi semfeats> == AGENT:<>.
TRANSITIVE: <syn subcat re> == COMP:<>
    <sem args re> == ARG:<>.
ARG:<fi semfeats> == PATIENT:<>
    <re> == NIL:<>.
COMP:<fi syn> == NP:<>
    <re> == NIL:<>.
NIL:<> == nil
    <fi> == UNDEF
    <re> == UNDEF.
```

The **COMP** and **ARG** nodes provide a location for default information about syntactic complements and semantic arguments. Complements are, by default, accusative noun phrases. Following [Dowty, 1991], we have a default expectation that subjects will have 'proto-agent' semantic features and objects, 'proto-patient' ones. The role of Dowty's approach in this analysis is that it gives us a way of marking the difference between agents and patients which says more

[3]This treatment is due to Roger Evans.

than simply using the labels 'agent' and 'patient', and has the potential for subtler distinctions, with different subsets of proto-agent and proto-patient features applying to subjects and objects of different verb classes. **AGENT** and **PATIENT** set up the expected values for four of the characteristics Dowty discusses.

```
NP:<maj> == n
   <case> == acc.
AGENT:<volition> == yes
    <sentient> == yes.
PATIENT:<changes-state> == yes
    <causally-affected> == yes.
```

The default accusative case and proto-patient semantic features must be overridden in the case of the subject:

```
VERB:<syn subcat fi syn case> == nom
     <sem args fi semfeats> == AGENT:<>.
```

To this skeleton, we add some smaller classes based on meanings. Once we introduce them we can start expressing generalisations about alternation behaviour. To distinguish alternate forms from base forms, we introduce the **alt** prefix. To request information about a non-base form, we start the query path with **alt** x, where x is a label identifying the alternation under consideration. We adopt a convention whereby all-upper-case nodenames are used for nodes for classes of words, such as cooking verbs, while lexical nodes have only initial letters capitalised.

```
Bake:<> == COOKING-VB
     <word> == bake
     <sem pred> == bake.
COOKING-VB:<> == C-OF-S
    <sem args re fi semfeats edible> == yes.
C-OF-S:<> == TRANSITIVE
    <alt erg> == PHYS-PROC:<>
    <alt erg sem> == "<sem>"
    <alt erg sem args fi binding>  == ex-q
    <alt erg sem args re fi binding> == v1.
```

Bake is a cooking verb, and cooking verbs are, in the base case, transitive change-of-state verbs. Thus **Bake** inherits, by default, from **COOKING-VB** which inherits from **C-OF-S** (for 'change of state') and then from **TRANSITIVE**, so acquiring the default specifications for semantic features for its subject and object, and the re-entrancies between subject and first argument, and object and second argument. The DATR fragment now represents all the information in the DATR lexical entry for *bake* presented above, and case and proto-agent and proto-patient specifications in addition.

The first generalisation about alternations that we wish to capture is that change-of-state transitives such as *bake* undergo the ergative alternation to become change-of-state intransitives, or 'physical process' verbs. We access the lexical entries for the ergative forms of verbs with DATR queries with the path prefix **alt erg**, which work as follows. The semantics of the ergative will be the same predicate-argument structure as the base form, and this is implemented in the third line of the **C-OF-S** node which tells us, with the double-quotes, to inherit the ergative's semantics from the semantics of the node for the base form of the verb. The two further specifications for ergatives are that the first argument is existentially quantified, and the second unifies with the first complement via **v1**.

In all other matters, as the second line of the **C-OF-S** node tells us, the ergative form is diverted to inherit from a node for physical-process intransitives:

```
PHYS-PROC:<> ==  VERB
    <sem args fi semfeats> == PATIENT:<>.
```

The first semantic argument of a physical-process intransitive has proto-patient semantic features and otherwise inherits from **VERB**. This is a case where the default – that first semantic arguments (realised as subjects in the base case) have proto-agent features – has been overridden, but the reader will note that this has been entirely straightforward to express in DATR.

We now have almost all the information needed to build the lexical entry for ergative *bake*. One item we do not yet have is the intuitively obvious fact that the **word** for the alternate form is the **word** for the original. This is true by definition for all alternate forms. All alternate forms will eventually have their **alt** x prefix (or prefixes) stripped and inherit from **WORD-CLASS** at the top of the tree. So we add the following line:

```
WORD-CLASS:<word> == "<word>".
```

Now all alternate forms will inherit their **word** from the **word** at the global context node, which will always be the node for the base form.

Many cooking verbs undergo the 'unspecified object' alternation, for which we shall use the label **unspec**. All information relating to this form is gathered at an **UNSPEC** node:

```
UNSPEC:<>   ==  VERB
    <sem> == "<sem>"
    <sem args re fi binding>  == ex-q.
```

This simply states that the form is a standard intransitive, with the semantics of the base form except that the second argument is existentially quantified. Cooking verbs with **alt unspec** prefixes are diverted here by the addition of:

```
COOKING-VB:<alt unspec>  == UNSPEC:<>.
```

Now we move on to *melt*, a physical-process verb with a causative form. The ergative alternation led from **C-OF-S** to **PHYS-PROC**. This makes a similar journey in the opposite direction, from **PHYS-PROC** to **CAUSE** and then **TRANSITIVE**. The alternation label is **cause**.

```
Melt:<> == PHYS-PROC
    <sem pred> == melt
    <word> == melt.

PHYS-PROC:<> ==  VERB
     <alt cause> == CAUSE:<>
     <alt cause sem args re fi> == "<sem>"
     <alt cause sem args re fi
               args fi binding> == v2.

CAUSE:<>   ==  TRANSITIVE
    <sem pred> == cause.
```

Causative *melt*, with the **alt cause** prefix, is a regular verb of causing, and inherits its syntax and most of its semantics including the predicate **cause/2** from **CAUSE**. Its first argument will have the usual characteristics of a CAUSER, and its second, the predicate-argument structure of the base form of the verb. As the predicate **melt** is now identified as the second argument of **cause**, the item that melts is identified as the first argument of the second argument of the causative form of the verb, and it is this which is re-entrant with the second item on the subcat list, as specified in the final line of **PHYS-PROC**.

The reward for this superstructure is that lexical entries can now be very concise. By adding a three-line entry, e.g.,

```
Bake: <>   == COOKING-VB
    <word> == bake
    <sem pred> == bake.
```

to the lexicon, we make available, for cooking verbs such as *bake*, a set of eighteen specifications for the base form, and fifteen each for the ergative and unspecified-object, and for physical process verbs, fifteen for the base and eighteen for the causative, all complete with case, subcategorisation, proto-agent, proto-patient and re-entrancy specifications, as below:

```
Bake: <lexical> = true.
Bake: <word> = bake.
Bake: <syn maj> = verb.
Bake: <syn subcat fi syn maj> = n.
Bake: <syn subcat fi syn case> = nom.
Bake: <syn subcat fi sem binding> = v1.
Bake: <syn subcat re fi syn maj> = n.
Bake: <syn subcat re fi syn case> = acc.
Bake: <syn subcat re fi sem binding> = v2.
Bake: <syn subcat re re> = nil.
Bake: <sem pred> = bake /2.
Bake: <sem args fi binding> = v1.
Bake: <sem args fi semfeats volition> = yes.
Bake: <sem args fi semfeats sentient> = yes.
Bake: <sem args re fi binding> = v2.
Bake: <sem args re fi semfeats
                      changes-state> = yes.
Bake: <sem args re fi semfeats
                  causally-affected> = yes.
Bake: <sem args re re> = nil.
```

6 Summary and discussion

First, HPSG-style verbal lexical entries, and the mappings between them corresponding to alternations, were described. But at this stage, the generalisations were not captured. So then these entries were translated into DATR, and arranged into a taxonomy so an alternation only needed expressing once, at a non-terminal node from which the verbs to which it applied would inherit. Information about syntax, semantics, and patterns of polysemy was concisely expressed in a manner both theoretically and computationally appealing.

The lexicon fragment described in detail is part of a larger fragment which also formalises the relations holding between transitives and intransitives of 'care' verbs such as *wash*, where the intransitive means the same as the reflexive; between transitive, intransitive, and two ditransitive forms of the 'clear' verbs ("clear the desk"; "the skies cleared"; "clear the desk of papers"; "clear the papers off the desk"); and between transitive and ditransitive forms of 'wipe' verbs ("wipe the shelf"; "wipe the dust off the shelf"). The complete fragment thus covers a number of the common transitivity alternations of English.

The paper aims to present both a study of lexical structure and an approach to practical lexicography. On the latter score, the ideal to which the paper contributes sees the lexicographer only ever needing to explicitly enter information that is idiosyncratic to a word and inheritance specifications, as everything that is predictable about a word's behaviour will be inferred. Maintaining consistency in the lexical representation, and updating and revising it, will also be quicker if a generalisation is located just at one place in the lexicon rather than at every word to which it applies.

Transitivity alternations defy classification as either syntactic or semantic phenomena. They are clearly both. The generalisations are associated with semantic classes of verbs, and have both syntactic and semantic consequences. The verb taxonomy of Fig. 1 may be used for conveying specifically linguistic information, as explored in this paper, but also potentially forms part of an encyclopedic knowledge base, with knowledge about any type of cooking held at the **COOKING-VB** node and knowledge specifically about frying and baking at the **Fry** and **Bake** nodes. It might be argued that this is to confuse two different kinds of information, but, as illustrated in this paper and argued in [Kilgarriff, 1992], the lexicon of English holds both the syntax and semantics of lexical items. The approach offered here indicates how linguistic and encyclopedic generalisations may be attached to the same taxonomic structure.

[Boguraev and Levin, 1990] show that an expressively adequate model for the lexicon must incorporate productive rules so that novel but rulebound uses of words can be captured. Thus "the her-

ring soused" is interpretable by any English speaker who has come across *soused herring*, but intransitive *souse* will not be added by any lexicographer to any dictionary: it is most unlikely that any corpus will provide any evidence for the form, and if it did, it would be of insufficient frequency to justify explicit treatment. The ergative form of *souse* must therefore be in the lexicon implicitly. Its availability to speakers and hearers of English can be inferred from knowledge of the kind of verb which *souse* is and the kinds of processes, or alternations, that verbs of that class can undergo. The DATR analysis demonstrates how such implicit availability of verb forms can be formalised.

6.1 Further work

A further question that the question of productivity invites is this: how are we to represent which verbs undergo which alternations? First, we might wish to develop devices within DATR or a related formalism for identifying which alternations apply where, and two such mechanisms are presented in [Kilgarriff, 1992]. But as we look closer, and consider the difficulty of placing many verbs in a semantic class, or the role of metaphor, analogy, and simple familiarity in determining which alternations are applicable in a given context of language-use, so the idea of a yes/no answer to questions of the form, "does this verb undergo this alternation?" loses plausibility.

This reasoning applies also to verb classes. The analysis offers an account of verb behaviour which is premised on verb classes, but their only justification has been by appeal to commonsense and an ill-defined notion of their ability to predict which alternations a verb participates in. Nothing has been said about how the classes might be identified, or how decisions regarding where a verb should be placed might be made.

The questions, "what class does a verb belong to?", "what are the relative frequencies of the different patterns it occurs in?", and "is this pattern grammatical?" are intimately connected. Alternation behaviour is a major source of evidence as to how a verb should be classified, and grammaticality judgements are premised upon the patterns a competent speaker has frequently encountered in their experience of the language. The further development of computational lexical semantics of the kind described in this paper requires foundational work on the relation of corpus-based statistical findings to formal knowledge representation.

References

[Atkins *et al.*, 1986] B. T. S. Atkins, Judy Kegl, and Beth Levin. Explicit and implicit information in dictionaries. In *Advances in Lexicography: Proc. Second Ann. Conf. of the UW Centre for the New OED*, pages 45–65, Waterloo, Canada, 1986.

[Boguraev and Levin, 1990] Branimir K. Boguraev and Beth Levin. Models for lexical knowledge bases. In *Electronic Text Research: Proc. Sixth Ann. Conf. of the UW Centre for the New OED*, pages 65–78, Waterloo, Canada, 1990.

[Brachman and Levesque, 1985] Ronald J. Brachman and Hector J. Levesque, editors. *Readings in Knowledge Representation.* Morgan Kaufmann, Los Altos, California, 1985.

[Briscoe *et al.*, 1990] Edward J. Briscoe, Ann A. Copestake, and Branimir K. Boguraev. Enjoy the paper: Lexical semantics via lexicology. In *COLING 90*, volume 2, pages 42–47, Helsinki, 1990.

[Cahill and Evans, 1990] Lynne J. Cahill and Roger Evans. An application of DATR: The TIC lexicon. In *Proc. ECAI-90*, pages 120–125, 1990.

[Cahill, 1993] Lynne J. Cahill. Morphonology in the lexicon. In *ACL Proceedings, 6th European Conference*, Utrecht, 1993.

[Chierchia and McConnell-Ginet, 1990] Gennaro Chierchia and Sally McConnell-Ginet. *Meaning and Grammar. An Introduction to Semantics.* MIT Press, Cambridge, Mass., 1990.

[Copestake and Briscoe, 1991] Ann A. Copestake and Edward J. Briscoe. Lexical operations in a unification-based framework. In James Pustejovsky and Sabine Bergler, editors, *Lexical semantics and knowledge representation: ACL SIGLEX Workshop*, Berkeley, California, 1991.

[Copestake, 1992] Ann Copestake. The ACQUILEX LKB: representation issues in semi-automatic acquisition of large lexicons. In *Proc. Third Conf. on Applied Natural Language Processing*, pages 88–95, Trento, Italy, 1992. Association of Computational Linguistics.

[Dowty, 1979] David R. Dowty. *Word Meaning in Montague Semantics.* Reidel, Dordrecht, 1979.

[Dowty, 1991] David R. Dowty. Thematic proto-roles and argument selection. *Language*, 67(3):547–619, 1991.

[Evans and Gazdar, 1989a] Roger Evans and Gerald Gazdar. Inference in DATR. In *ACL Proceedings, 4th European Conference*, pages 1–9, Manchester, 1989.

[Evans and Gazdar, 1989b] Roger Evans and Gerald Gazdar. The semantics of DATR. In Anthony G. Cohn, editor, *Proc. Seventh Conference of the AISB*, pages 79–87, Falmer, Sussex, 1989.

[Evans and Gazdar, 1990] Roger Evans and Gerald Gazdar. The DATR papers. Technical Report CSRP 139, School of Cognitive and Computing Sciences, University of Sussex, Falmer, Sussex, 1990.

[Gibbon, 1990] Dafydd Gibbon. Prosodic association by template inheritance. In *Proc. Workshop*

on Inheritance in Natural Language Processing, pages 65–81, Tilburg, 1990. ITK.

[Kilgarriff and Gazdar, 1993 forthcoming] Adam Kilgarriff and Gerald Gazdar. Polysemous relations. In Frank R. Palmer, editor, *Festschrift for Sir John Lyons*. CUP, Cambridge, England, 1993, forthcoming.

[Kilgarriff, 1992] Adam Kilgarriff. *Polysemy*. PhD thesis, University of Sussex, CSRP 261, School of Cognitive and Computing Sciences, 1992.

[Kilgarriff, 1993 forthcoming] Adam Kilgarriff. Inheriting polysemy. In Patrick St. Dizier and Evelyne Viegas, editors, *Computational Lexical Semantics*. CUP, 1993, forthcoming.

[Levin and Rappoport Hovav, 1991] Beth Levin and Malka Rappoport Hovav. Wiping the slate clean: A lexical semantic exploration. *Cognition*, 41:123–151, 1991.

[Levin, 1991] Beth Levin. Building a lexicon: The contribution of linguistics. *International Journal of Lexicography*, 4(3):205–226, 1991.

[Moore, 1989] Robert C. Moore. Unification-based semantic interpretation. In *ACL Proceedings, 27th Annual Meeting*, pages 33–41, Vancouver, 1989.

[Pollard and Sag, 1987] Carl Pollard and Ivan A. Sag. *An Information-Based Approach to Syntax and Semantics. Volume 1: Fundamentals*. CSLI Lecture Notes, No. 13. Chicago University Press, Chicago, 1987.

[Pustejovsky, 1991] James Pustejovsky. The generative lexicon. *Computational Linguistics*, 17(4):409–441, 1991.

[Russell *et al.*, 1991] Graham Russell, Afzal Ballim, John Carroll, and Susan Armstrong-Warwick. A practical approach to multiple default inheritance for unification-based lexicons. In Edward J. Briscoe, Ann A. Copestake, and Valeria de Paiva, editors, *Proc. ACQUILEX workshop on default inheritance in the lexicon*, Tech. report 238, University of Cambridge Computer Laboratory, 1991.

[Sanfilippo and Poznanski, 1992] Antonio Sanfilippo and Victor Poznanski. The acquisition of lexical knowledge from combined machine-readable dictionary sources. In *Proc. Third Conf. on Applied Natural Language Processing*, pages 80–87, Trento, Italy, 1992. Association of Computational Linguistics.