FAME: a Functional Annotation Meta-scheme for multi–modal and multi–lingual Parsing Evaluation

Alessandro Lenci, Simonetta Montemagni, Vito Pirrelli, Claudia Soria
Istituto di Linguistica Computazionale – CNR
via della Faggiola 32
Pisa, 56126, ITALY
{lenci,simo,vito,soria}@ilc.pi.cnr.it

Abstract

The paper describes FAME, a functional annotation meta–scheme for comparison and evaluation of existing syntactic annotation schemes, intended to be used as a flexible yardstick in multi–lingual and multi–modal parser evaluation campaigns. We show that FAME complies with a variety of non–trivial methodological requirements, and has the potential for being effectively used as an "interlingua" between different syntactic representation formats.

1 Introduction

Broad coverage parsing evaluation has received growing attention in the NLP community. In particular, comparative, quantitative evaluation of parsing systems has acquired a crucial role in technology assessment. In this context, it is important that evaluation be relatively independent of, or easily parametrizable relative to the following three dimensions of variation among parsing systems:

- **theoretical assumptions**: compared systems may be based on different theoretical frameworks;
- **multi–linguality**: parsers are often optimally designed to deal with a particular language or family of languages;
- **multi–modality**: systems tend to be specialized for dealing with a specific type of input, i.e. written or spoken language.

As to the first point, it is important that alternative annotation schemes be evaluated (i) on the basis of the linguistic information they are intended to provide, and (ii) in terms of the utility of this information with respect to a particular task. Moreover, multi–linguality and multi–modality are crucial parameters for evaluating the robustness and portability of a given parser, with a view to the growing need for embedding NLP systems into multi–modal and multi–medial applications.

An essential aspect of every evaluation campaign is the specification of an annotation scheme into which the output of the participant systems is converted and on whose basis the system performance is eventually evaluated. A suitable annotation scheme must satisfy some requirements. First of all, it should be able to represent the information relevant to a certain evaluation task in a way which is naturally conducive to quantitative evaluation. Secondly, it should easily be mappable onto different system outputs, and flexible enough to deal with multilingual phenomena and with the specific nature of both written and spoken language.

The aim of this paper is to illustrate FAME, a Functional Annotation Meta–scheme for Evaluation. We will show that it complies with the above mentioned requirements, and lends itself to effectively being used in comparative evaluation campaigns of parsing systems. There are two main features of FAME that will receive particular emphasis here: it is **functional** and it is a **meta–scheme**. We claim that these two features are essential for meeting the specific requirements of comparative parsing evaluation, while tackling issues of multi–linguality and multi–modality in a principled fashion.

2 FAME: Basics

What we intend to offer here is not yet another off–the–shelf annotation scheme, but rather a formal framework for comparison and evaluation of existing annotation practices at the level of linguistic analysis traditionally known as "functional". Hereafter, this framework will be referred to as an annotation "meta–scheme".

2.1 Why functional evaluation

The choice of evaluating parsing systems at the functional level is largely motivated on the basis of a number of practical concerns. We contend that information about how functional relations are actually instantiated in a text is important for the following reasons:

- it is linguistically valuable, both as an end in itself and as an intermediate linguistic resource; in fact, it is sufficiently close to semantic representations to be used as an intermediate stage of analysis in systems requiring full text understanding capabilities;

- it is likely to become a more and more heavily used information asset in its own right for NLP applications: a shift of emphasis from purely pattern matching methods operating on n–word windows to functional information about word pairs has recently been witnessed both in the context of information retrieval/filtering systems (Grefenstette, 1994) and for the purposes of word sense disambiguation (see the last SENSEVAL and ROMANSEVAL evaluation campaigns);
- it is comparatively easy and "fair" to evaluate since it overcomes some of the shortcomings of constituency–based evaluation (Carroll and Briscoe, 1996; Carroll et al., 1998; Sampson, 1998; Lin, 1998);
- it represents a very informative "lowest common ground" of a variety of different syntactic annotation schemes (Lin, 1998);
- it is naturally multi–lingual, as functional relations probably represent the most significant level of syntactic analysis at which cross–language comparability makes sense;
- it permits joint evaluation of systems dealing with both spoken and written language. Spoken data are typically fraught with cases of disfluency, anacoluthon, syntactic incompleteness and any sort of non-canonical syntactic structure (Antoine, 1995): the level of functional analysis naturally reflects a somewhat standardized representation, which abstracts away from the surface realization of syntactic units in a sentence, thus being relatively independent of, and unconcerned with disfluency phenomena and phrase partials (Klein et al., 1998);
- it is "lexical" enough in character to make provision for *partial* and *focused* annotation: since a functional relation always involves two lexical heads at a time, as opposed to complex hierarchies of embedded constituents, it is comparatively easy to evaluate an annotated text only relative to a subset of the actually occurring headwords, e.g. those carrying a critical information weight for the intended task and/or specific domain.

2.2 Why an annotation meta–scheme

FAME is designed to meet the following desiderata:

- provide not only a measure of coverage but also of the utility of the covered information as opposed to missing information;
- make explicit, through annotation, information which is otherwise only indirectly derivable from the parsed text;
- factor out linguistically independent (but possibly correlated) primitive dimensions of functional information.

All these requirements serve the main purpose of making evaluation open to both annotation–dependent and task–dependent parameterization. This is felt important since the definition of closeness to a standard, and the utility of an analysis that is less–than–perfect along some dimension can vary from task to task, and, perhaps more crucially, from annotation scheme to annotation scheme.

The basic idea underpinning the design of the annotation meta-scheme is that information about how functional relations are actually instantiated in context can be factored out into linguistically independent levels. In many cases, this can in fact be *redundant*, as information at one level can be logically presupposed by a piece of information encoded at another level: for example, "nominative case" is often (but not always) a unique indicator of "subjecthood", and the same holds for grammatical agreement. Yet, there is a general consensus that redundancy should not be a primary concern in the design of a standard representation, as syntactic schemes often differ from each other in the *way* levels of information are mutually implied, rather than in the *intrinsic nature* of these levels (Sanfilippo et al., 1996). By assuming that all levels are, in a sense, primitive, rather than some of them being derivative of others, one provides considerable leeway for radically different definitions of functional relations to be cast into a common, albeit redundant, core of required information. We will return to this point in section 3 of the paper.

To be more concrete, a binary functional relationship can be represented formally as consisting of the following types of information:

i. the unordered terms of the relationship (i.e. the linguistic units in text which enter a given functional relationship): example `(give, Mary)`;

ii. the order relationship between the terms considered, conveying information about the head and the dependent: example `<give, Mary>`;

iii. the type of relationship involved: example, the functional relation of the pair `(give, Mary)` in the sentence *John gave the book to Mary* is "indirect object";

iv. morpho–syntactic features associated with the dependent and the head; e.g. the dependent in the pair `(give, Mary)` is "non-clausal";

v. the predicate–argument status of the terms involved: for example `give(John, book, Mary)` in *John gave the book to Mary.*

Most available tag taxonomies for functional annotation (such as those provided by, e.g., Karlsson's Constraint Grammar (Karlsson et al., 1995), or the SPARKLE annotation scheme (Carroll et al., 1996), to mention but two of them) typically collapse the levels above into one level only, for reasons ranging from a theoretical bias towards a maximally economic description of the phenomena in question or a particular view of the way syntactic phenomena are mutually implied from a logical standpoint, to choices chiefly motivated by the intended application. A typical example of this is the tag `xcomp` in the SPARKLE scheme, which (following LFG) covers ***all subcategorized open predicates***: namely, traditional predicative complements (whether subject or object predicative), and unsaturated clausal complements, such as embedded infinitival and participial clauses (as opposed to, e.g., *that*–clauses). In Constraint Grammar, predicative nominal and adjectival phrases are tagged as "subject complement" or "object complement", while, say, controlled infinitive clauses, as in *Mary wants to read*, are marked functionally as an "object" of the main verb. Any context-free attempt to map SPARKLE `xcomp` onto a Constraint Grammar tag, would inevitably be one–to–many and not necessarily information–preserving. Clearly, both these aspects make it very hard to provide any sort of fair baseline for comparing a SPARKLE annotated text against the same text tagged with Constraint Grammar labels.

The design of a meta–scheme is intended to tackle these difficulties by spelling out the levels of information commonly collapsed into each tag. More concretely, SPARKLE `xcomp(want,leave)`, for the sentence *She wants to leave*, appears to convey two sorts of information: (a) that *leave* is a complement of *want*, (b) that *leave* is an open predicate. Both pieces of information can be evaluated independently against levels i, ii, iii and v above.

Surely, a translation into FAME is not guaranteed to always be *information preserving*. For example, `xcomp(want,leave)` can also be interpreted as conveying information about the intended functional control of *leave*, given some (lexical) information about the main verb *want*, and some (contextual) information concerning the absence of a direct object in the sentence considered. However, this sort of context–sensitive translation would involve a more or less complete reprocessing of the entire output representation.[1] In our view, a partial context–free translation into FAME represents a sort of realistic compromise between a fairly uninformative one–to–many mapping and the complete translation of the information conveyed by one scheme into another format.

2.3 Information layers in FAME

To date, FAME covers levels i–iv only. The building blocks of the proposed annotation scheme are functional relations, where a functional relation is an asymmetric binary relation between a word called HEAD and another word called DEPENDENT. We assume only relations holding between lexical or full words. Therefore, we exclude functional relations involving grammatical elements such as determiners, auxiliaries, complementizers, prepositions, etc. The information concerning these elements is conveyed through features, as described below in section 2.3.3.

Each functional relation is expressed as follows:

```
dep_type (lex_head.<head_features>,
          dependent.<dep_features>)
```

`Dep_type` specifies the relationship holding between the lexical head (`lex_head`) and its dependent (`dependent`). The head and the dependent of the relation are further specified through a (possibly empty) list of valued features (respectively `head_features` and `dep_features`), which complement functional information.

2.3.1 The hierarchy of functions

`Dep_types` are hierarchically structured to make provision for underspecified representations of highly ambiguous functional analyses (see further below). The hierarchy of relations is given in figure 1 below.

In the hierarchy, the function `subj` (for "subject") is opposed to other grammatical relations by being assigned a higher prominence in the taxonomy, as customary in contemporary grammar theories (e.g. HPSG, GB). Moreover, modifiers and arguments are subsumed under the same `comp` node (mnemonic for complement), allowing for the possibility of leaving underspecified the distinction between an adjunct and a subcategorised argument in those cases where the distinction is difficult to draw in practice. In turn, the node `arg` (for argument) is split into `pred`, subsuming all and only classical predicative complements, and `non-pred`, further specified into `dobj`

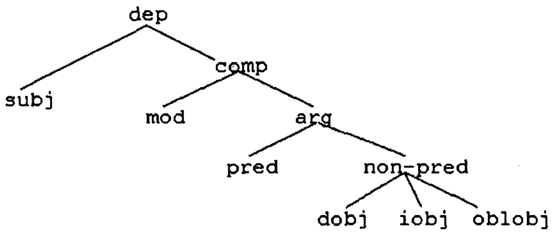

Figure 1: Hierarchy of functional relations

[1] In fact, the SPARKLE annotation scheme annotates control information explicitly, as illustrated later in the paper: the point here is simply that this information cannot be derived *directly* from `xcomp(want,leave)`.

(for direct objects), iobj (for indirect objects) and oblobj (for oblique arguments).

The hierarchy of figure 2.3.1 is a revision of the SPARKLE functional hierarchy (Carroll et al., 1996), in the light of the methodological points raised in section 2.2. The main point of departure can be found under the node comp, which, in SPARKLE, dominates the nodes obj and clausal, thus reflecting a view of predicative complements as small clauses, to be assimilated with other unsaturated clausal constructions such as infinitival and participial clauses. This is in clear conflict with another grammatical tradition that marks clausal complements with the functional relations also assigned to non clausal complements, when the latter appear to be in a parallel distribution with the former, as in *I accept his position* and *I accept that he leaves*, where both *his position* and *that he leaves* are tagged as objects (Karlsson et al., 1995). This is a typical example of how functions may differ due to a difference in the levels of the linguistic information taken to be criterial for tag assignment. As we will see in more detail in section 2.3.2, the FAME hierarchy circumvents the problem by assigning all non–subject clausal complements the tag arg, which subsumes both traditional predicatives (pred) and non clausal arguments (non-pred), thus granting sentential complements a kind of ambivalent (underspecified) functional status.

2.3.2 The typology of functions

In what follows we sketchily define each functional relation; examples are provided for non generic nodes of the hierarchy only.

dep(head,dependent) is the most generic relation between a head and a dependent, subsuming the distinction between a subject and a complement.

subj(head,dependent) is the relation between a verb predicate and its subject:

subj(arrive,John) *John arrived in Paris*
subj(employ,IBM) *IBM employed 10 C programmers*
subj(employ,Paul) *Paul was employed by IBM*

Subj refers to the superficial subject of a verb, regardless of the latter being used in the active or passive voice. Moreover, it can also be used to mark subject control relations and, possibly, raising to object/subject phenomena, as exemplified below:

subj(leave,John) *John promised Mary to leave*
subj(leave,Mary) *John ordered Mary to leave*
subj(be,her) *John believes her to be intelligent*
subj(be,John) *John seems to be intelligent*

Also clausal subjects are marked as subj:

subj(mean,leave) *that Mary left meant she was sick*
subj(require,win) *to win the America's Cup requires heaps of cash*

comp(head,dependent) is the most generic relation between a head and a complement, whether a modifier or a subcategorized argument.

mod(head,dependent) holds between a head and its modifier, whether clausal or non–clausal; e.g.

mod(flag,red) *a red flag*
mod(walk,slowly) *walk slowly*
mod(walk,John) *walk with John*
mod(Picasso,painter) *Picasso the painter*
mod(walk,talk) *walk while talking*

Mod is also used to encode the relation between an event noun (including deverbal nouns) and its participants, and the relation between a head and a semantic argument which is syntactically realised as a modifier (as in the passive construction), e.g.:

mod(destruction,city) *the destruction of the city*
mod(kill,Brutus) *he was killed by Brutus*

arg(head,dependent) is the most generic relation between a head and a subcategorized argument; besides functional underspecification, it is used to tag the syntactic relation between a verbal head and a non–subject clausal argument (see section 2.3.1 above):

arg(say,accept) *He said that he will accept the job*

pred(head,dependent) is the relation which holds between a head and a predicative complement, be it subject or object predicative, e.g.

pred(be,intelligent) *John is intelligent*
pred(consider,genius) *John considers Mary a genius*

nonpred(head,dependent) is the relation which holds between a head and a non predicative complement.

dobj(head,dependent) is the relation between a predicate and its direct object (always non–clausal), e.g.:

dobj(read,book) *John read many books*

iobj(head,dependent) is the relation between a predicate and the indirect object, i.e. the complement expressing the recipient or beneficiary of the action expressed by the verb, e.g.

iobj(speak,Mary) *John speaks to Mary*
iobj(give,Mary) *John gave Mary the contract*
iobj(give,Mary) *John gave the contract to Mary*

oblobj(head,dependent) is the relation between a predicate and a non-direct non clausal complement, e.g.

oblobj(live,Rome) *John lives in Rome*

oblobj(inform,run) *John informed me of his run*

In order to represent conjunctions and disjunctions, FAME avails itself of the two symmetric relations conj and disj, lying outside the dependency hierarchy. Consider, for instance, the FAME representation of the following sentence, containing a conjoined subject:

John and Mary arrived
subj(arrive,John)
subj(arrive,Mary)
conj(John,Mary)

The FAME representation of the sentence *John or Mary arrived* differs from the previous one only in the type of relation linking *John* and *Mary*: namely, disj(John,Mary).

2.3.3 Feature specification

In FAME, a crucial role is played by the features associated with both elements of the relation.
Dep(endent)_features are as follows:

- Intro(ducer): it refers to the grammatical word (a preposition, a conjunction etc.) which possibly introduces the dependent in a given functional relation, e.g.

  iobj (give, Mary.<intro=''to''>) *give to Mary*

  arg(say,accept.<intro=''that''>) *Paul said that he accepts his offer*

- Case: it encodes the case of the dependent, e.g.

  iobj (dare, gli.<case=DAT>) *dargli* 'give to him'

- Synt_real: it refers to a broad classification of the syntactic realization of a given dependent, with respect to its being clausal or non–clausal, or with respect to the type of clausal structure (i.e. whether it is an open function or a closed function). Possible values of this feature are:

  - x: a subcategorized argument or modifier containing an empty argument position which must be controlled by a constituent outside it, e.g.

    arg(decide,leave.<synt_real=x>) *John decided to leave*

  - c: a subcategorized argument or modifier which requires no control by a constituent outside it, e.g.

    arg(say, leave.<synt_real=c>) *John said he left*

  - nc: a non–clausal argument or modifier, e.g.

    dobj(eat,pizza.<synt_real=nc>) *John ate a pizza*

Head_features are as follows:

- Diath: it specifies the diathesis of a verbal head, e.g.

  subj(employ.<diath=passive>, Paul) *Paul was employed by IBM*

  subj(employ.<diath=active>, IBM) *IBM employed Paul*

- Person: it specifies the person of a verbal head, e.g.

  subj(eat.<person=3>, he) *he eats a pizza*

- Number: it specifies the number of a verbal head. e.g.

  subj(eat.<number=sing>, he) *he eats a pizza*

- Gender: it specifies the gender of a head, e.g.

  subj(arrivare.<gender=fem>, Maria) *Maria è arrivata* 'Maria has come'

3 FAME at work

Theory-neutrality Theory-neutrality is an often emphasised requirement for reference annotation schemata to be used in evaluation campaigns (see GRACE, (Adda et al., 1998)). The problem with theory neutrality in this context is that, although some agreement can be found on a set of basic *labels*, problems arise as soon as the *definition* of these labels comes in. For example, the definition of "subject" as a noun constituent marked with nominative case is not entirely satisfactory, since a system might want to analyse the accusative pronoun in *John believes her to be intelligent* as the subject of the verb heading the embedded infinitival clause (as customary in some linguistic analyses of this type of complements). Even agreement, often invoked as a criterial property for subject identification, may be equally tricky and too theory–loaded for purposes of parser comparison and evaluation.

The approach of FAME to this bunch of issues is to separate the repertoire of functional relation types (labels), from the set of morpho-syntactic features associated with the head and dependent, as shown in the examples below:

subj(be,she.<case=accusative>) *John believes her to be intelligent* `subj(be,she.<case=nominative>)` *She seems to be intelligent*

By doing this way, emphasis is shifted from theory-neutrality (an almost unattainable goal) to *modularity* of representation: a functional representation is articulated into different information levels, each factoring out different but possibly interrelated linguistic facets of functional annotation.

Intertranslatability A comparative evaluation campaign has to take into account that participant systems may include parsers based on rather different approaches to syntax (e.g. dependency-based, constituency-based, HPSG-like, LFG-like, etc.) and applied to different languages and test corpora. For a comparative evaluation to be possible, it is therefore necessary to take into account the specificity of a system, while at the same time guaranteeing the feasibility and effectiveness of a mapping of the system output format onto the reference annotation scheme. It is important to bear in mind at this stage that:

- most broad-coverage parsers are constituency-based;
- the largest syntactic databases (treebanks) use constituency-based representations.

It is then crucial to make it sure that constituency-based representations, or any other variants thereof, be mappable onto the functional reference annotation meta-scheme. The same point is convincingly argued for by Lin (1998), who also provides an algorithm for mapping a constituency-based representation onto a dependency-based format. To show that the requirement of intertranslatability is satisfied by FAME, we consider here four different analyses for the sentence *John tried to open the window* together with their translation equivalent in the FAME format:

1. ANLT Parser (Briscoe & Carroll, 1995) - traditional PSG representation:

   ```
   (Tp
    (V2 (N2 (N1 (NO John_NP1)))
     (V1 (VO tried_VVD)
      (V1 (VO to_TO)
       (V1 (VO open_VVO)
        (N2 (DT the_AT)(N1 (NO window_NN1))
     )))))).
   ```

   FAME equivalent:

   ```
   subj(try,John)
   arg(try,open.<introducer="to">)
   dobj(open,window)
   ```

2. Fast Partial Parser (Grefenstette, 1994):

   ```
   SUBJ(try,John)
   DOBJ(open,window)
   SUBJ(open,John)
   MODIF(open,try).
   ```

   FAME equivalent:

   ```
   subj(try,John)
   dobj(open,window)
   subj(open,John)
   mod(open,try)
   ```

3. Finite State Constraint Grammar Parser (Karlsson et al., 1995):

   ```
   John N SUBJ
   tried V MVMAINC^
   to INFMARK open V_INF MV OBJ^
   the DET window NOBJ.
   ```

   FAME equivalent:

   ```
   subj(try,John)
   arg(try,open.<introducer="to",
                 synt_real=x>)
   dobj(open,window)
   ```

4. PENN Predicate Argument structure (Marcus et al., 1994):

   ```
   want(try(John,open(John, window))).
   ```

   FAME equivalent:

   ```
   subj(try,John)
   arg(try,open)
   subj(open,John)
   dobj(open,window)
   ```

Let us suppose now that the reference analysis for the evaluation of the same sentence in FAME is as follows:

```
subj(try,John)
arg(try,open.<introducer="to",synt_real=x>)
subj(open,John)
dobj(open,window)
```

Notice that this representation differs from the output of the ANLT Parser and of the Finite State Constraint Grammar Parser mainly because they both give no explicit indication of the control relationship between the verb in the infinitive clause and the matrix subject. This information is marked in the output of both the Fast Partial Parser and the PENN predicate–argument tagging. Note further that the Fast Partial Parser gives a different interpretation of the infinitival complement, which is marked as being modified by *try*, rather than being interpreted as a direct object of *try*.

FAME does justice to these subtle differences as follows. First, it should be reminded that the FAME equivalents given above are in fact shorthand representations. Full representations are distributed over four levels, and precision and recall are to be gauged jointly relative to all such levels. To be concrete, let us first show a full version of the FAME standard representation for the sentence *John tried to open the window* (cf. Section 2.2):

i. `(try,John)`

```
  ii. <try,John>
 iii. subj
   i. (try,open)
  ii. <try,open>
 iii. arg
  iv. open.<introducer="to",synt_real=x>
   i. (open,John)
  ii. <open,John>
 iii. subj
   i. (open,window)
  ii. <open,window>
 iii. dobj
```

Note first that information about the unsaturated clausal complement *to open* is separately encoded as synt_real=x in the standard representation. The failure to explicitly annotate this piece of information incurred by ANLT and the Constraint Grammar Parser will then be penalised in terms of *recall*, but would eventually not affect ***precision***. By the same token, the subject control relation between *John* and *open* is recalled only by the Fast Partial Parser and PENN, and left untagged in the remaining schemes, thus lowering recall. The somewhat unorthodox functional dependency between *try* and *open* proposed by the Fast Partial Parser will receive the following full–blown FAME translation:

```
mod
(try,open)
<open,try>
```

When compared with the standard representation, this translation is a hit at the level of identification of the unordered dependency pair (try,open), although both the order of elements in the pair (<open,try>) and their functional dependency (mod) fail to match the standard. On this specific dependency, thus, recall will be $\frac{1}{3}$. As a more charitable alternative to this evaluation, it can be suggested that the difference between the FAME standard and the Fast Partial Parser output is the consequence of theory internal assumptions concerning the analysis of subject-control structures, and that this difference should eventually be leveled out in the translation into FAME. This may yield a fairer evaluation, but has the drawback, in our view, of obscuring an important difference between the two representations.

Evaluation of dialogue systems Dialogue management systems have to be able to deal with both syntactic and semantic information at the same time. These two levels of information are usually dealt with separately for reasons of higher ease of representation, and ease of change, updating and adaptation to different domains and different languages. Nonetheless, the formalisms used for syntax and semantics must have a certain degree of similarity and some additional knowledge about the relationships between syntax and semantics is necessary. An example is provided by what has been done in the ESPRIT SUNDIAL project (Peckam, 1991), where Syntax is defined using a dependency grammar augmented with morphological agreement rules; Semantics is declared through case frames (Fillmore, 1968; Fillmore, 1985) using a conceptual graph formalism. An additional bulk of knowledge, called mapping knowledge, specifies possible links between the symbols of the dependency grammar and the concepts of case frames. In this way syntactic and semantic controls are performed at the same time, avoiding the generation of parse trees that must afterwards be validated semantically. The FAME meta–scheme fits in comparatively well with this approach to parsing, as (a) functional annotation is readily translatable into dependency–like tags, and (b) the scheme makes provision for integration of syntactic and semantic information.

Furthermore, the lexical character of FAME functional analysis as a dependency between specific headwords, makes annotation at the functional level compatible with score driven, middle-out parsing algorithms, whereby parsing may "jump" from one place to another of the sentence, beginning, for example, with the best-scored word, expanding it with adjacent words in accordance with the language model (Giachin, 1997). Scoring can be a function of the reliability of speech recognition in the word lattice, so that the parser can start off from the most-reliably recognized word(s). Alternatively, higher scores can be assigned to the most relevant content words in the dialogue, given a specific domain/task at hand, thus reducing the complexity space of parses.

Use of underspecification FAME hierarchical organization of functional relations makes it possible to resort to underspecified tags for notoriously hard cases of functional disambiguation. For example, both *Gianni* and *Mario* can be subject or object in the Italian sentence *Mario, non l'ha ancora visto, Gianni*, which can mean both 'Mario has not seen Gianni yet' and 'Gianni has not seen Mario yet'. In this case, the parser could leave the ambiguity unresolved by using the underspecified functional relation dep, e.g. dep(vedere,Mario) and dep(vedere,Gianni). Similarly, the underspecified relation comp comes in handy for those cases where it is difficult to draw a line between adjuncts and subcategorized elements. This is a crucial issue if one considers the wide range of variability in the subcategorization information contained by the lexical resources used by participant systems. Given the sentence *John pushed the cart to the station*, for example, a comp relation is compatible both with an analysis where *to the station* is tagged as a modifier, and with an analysis which considers it an argument. We already considered (section 2.3.1) the issue of tagging sentential complements as arg, as a way to circumvent the theoretical issue of whether the functional relations of clauses should be defined on the basis of their predicative status, or, alternatively, of their syntactic distribution.

To sum up, underspecification thus guarantees a more flexible and balanced evaluation of the system outputs, especially relative to those constructions whose syntactic analysis is controversial.

4 Conclusion and developments

The suggestion of using a functional meta–scheme as a fair basis for parsing evaluation rests on the idea that parsing systems must be assessed for what they are intended to provide, not for how well they meet the requisites of other annotation schemes. Still, it makes a lot of sense to compare the amount of information provided by different parsers by casting this information into a common format. The distributed information structure of FAME is conducive to an incremental evaluation procedure, which ranges from a base evaluation level (involving sheer identification of the terms of a syntactic relationship and/or their order), to more refined levels, including morpho–syntactic information, dependency type, and ultimately predicate–argument structure. The evaluation of a text annotated for functional information can then be conceived of as a function of estimating precision and recall for each of the independent evaluation levels envisaged. Evaluation results obtained for the different levels can eventually be combined together or, for particular purposes, assessed in their own right (e.g. for IR applications the basic evaluation level could be sufficient). We are considering the possibility of extending FAME through addition of still further levels of lingustic information.

References

Adda, G., Mariani, J., Lecomte, J., Paroubek, P. and M. Rajman. 1998. The GRACE French Part–of–speech tagging evaluation task. In *Proceedings of the First International Conference on Language Resources and Evaluation*, pages 443–441, Granada, Spain.

Antoine, J. Y. 1995. Conception de Dessin et CHM. In K. Zreik et J. Caelen. *Le communicationnel pour concevoir*, pages 161–184. Europia, Paris, France.

Briscoe, Ted and John Carroll. 1995. Developing and evaluating a probabilistic LR parser of part–of–speech and punctuation labels. In *Proceedings of the Fourth ACL/SIGPARSE International Workshop on parsing technologies*, pages 48–58, Prague, Czech Republic.

Carroll, John, Briscoe, Ted, Calzolari, Nicoletta, Federici, Stefano, Montemagni, Simonetta, Pirrelli, Vito, Grefenstette, Gregory, Sanfilippo, Antonio, Carroll, Glenn and Mats Rooth. 1996. Specification of Phrasal Parsing. *SPARKLE Deliverable 1.*

Carroll, John and Ted Briscoe. 1996. Apportioning development effort in a probabilistic LR parsing system through evaluation. In *Proceedings of the ACL/SIGDAT Conference on Empirical Methods in Natural Language Processing*, pages 92–100.

Carroll, John, Briscoe, Ted and Antonio Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proceedings of the First International Conference on Language Resources and Evaluation*, pages 447–454, Granada, Spain.

Fillmore, C. J.. 1968. The case for case. In E. Bach and R. Harms (Eds.), *Universals in Linguistic Theory*, pages 1–88, Holt, Rinehart & Winston, New York, USA.

Fillmore, C. J.. 1985. Frames and the semantics of understanding. *Quaderni di Semantica*, 6:222-255.

Giachin, E. and S. McGlashan . 1997. Spoken Language Dialogue Systems. In Steve Young and Gerrit Bloothooft (Eds.)*Corpus-Based Methods in Language and Speech Processing*, pages 69–117, Kluwer, Dordrecht, The Netherlands.

Grefenstette, Greg. 1994. *Explorations in Automatic Thesaurus Discovery.* Kluwer, Dordrecht, The Netherlands.

Hudson, Richard. 1984. *Word Grammar.* Blackwell, Oxford, England.

Karlsson, Fred, Voutilainen, Atro, Heikkilä, Juha and Arto Anttila. 1995. *Constraint Grammar: A Language–Independent System for Parsing Unrestricted Text.* de Gruyter, Berlin, Germany.

Klein, M., Bernsen, N. O., Davies, S. , Dybkjaer, L. , Garrido, J. , Kasch, H. , Mengel, A., Pirrelli, V., Poesio, M., Quazza, S. and C. Soria. 1998. Supported Coding Schemes. MATE Technical Report D1.1.

Lin, Dekang. 1998. A dependency based method for evaluating broad-coverage parsers. *Natural Language Engineering*, 4(2):97–114.

Marcus, Mitchell, Kim, Grace, Marcinkiewicz, Mary Ann, MacIntyre, Robert, Bies, Ann, Ferguson, Mark, Katz, Karen and Britta Schasberger. 1994. The Penn Treebank: annotating predicate argument structure. *Proceedings of DARPA 1994.*

Peckam, J.. 1991. Speech understanding and dialogue over the telephone, An overview of the ESPRIT SUNDIAL project. In *Proceedings DARPA Speech and Natural Language Workshop.*

Sampson, Geoffrey. 1998. A proposal for improving the measurement of parse accuracy. Unpublished manuscript.

Sanfilippo, A., Barnett, R., Calzolari, N., Flores, S., Hellwig, P., Kahrel, P., Leech, G., Melero, M., Montemagni, S., Odijk, J., Pirrelli, V., Teufel, S., Villegas, M. and Zaysser, L. 1996. Subcategorization Standards. Report of the EAGLES Lexicon/Syntax Group. *SHARP Laboratories of Europe, Oxford (Regno Unito).*