An Aboutness-based Dependency Parser for Dutch

Cornelis H.A. Koster
Computing Science Institute,
University of Nijmegen,
The Netherlands

June 30, 2011

Abstract

In this note we describe the rationale behind Dupira, a new Dependency Parser for Dutch which is being developed at the University of Nijmegen. Dupira is a rule-based parser, generated by means of the AGFL parser generator from the Dupira grammar, lexicon and fact tables. By means of transductions which are specified in the grammar (and can be modified), the parser transduces sentences to dependency graphs.

Dupira was developed for practical applications in Information Retrieval and for Information Systems needing a Natural Language interface. Its intended users are computer scientists and computer professionals rather than linguists.

We shall first describe the aboutness-based dependency model of Dupira. Then we show by means of examples the transduction of clauses and phrases. We report the availability of Dupira Version 0.9 in the public domain, give some preliminary results about its performance, and discuss its potential applications.

1 Introduction

Dupira (the Dutch Parser for IR Applications) is a new Dependency Parser for Dutch, which has been developed in the context of the Text Mining project TM4IP. It is the outcome of many years of collaboration between the departments of Linguistics and Informatics of the Radboud University Nijmegen. It was developed in less than two years, based on the lexical resources of the Amazon parser (Coppen 2002).

Dupira is intended for applications in Information Retrieval (IR) rather than in Linguistics and for that reason has the following properties:

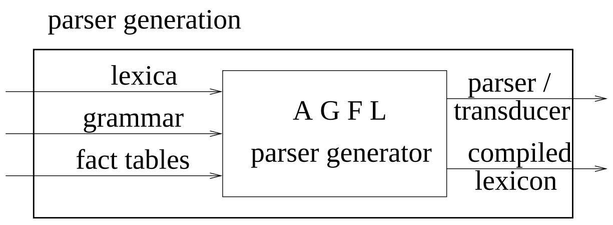
- the dependency model of Dupira expresses the *aboutness* (see section 2.3) of a sentence rather than describing its complete syntactic structure;
- Dupira resolves the thematic role of noun phrases (see 2);
- it extracts *dependency triples* from the text (see 2.2), which can be used as high-accuracy terms for text categorization and full-text search;
- to enhance recall, it performs certain aboutness-preserving normalizing transformations, including *de-passivization* and *de-topicalization* (see 2.7);
- it uses *subcategorization preferences* to resolve where possible the attachment of Preposition Phrases and embedded clauses (for the case of verb transitivity, see 3.2);
- therefore it is highly suitable for extracting *factoids* (see 2.6) from running text;
- it is highly *robust*, both lexically and syntactically (see 1.2).

The present document serves to explain, primarily to computer linguists and computer scientists, the rationale behind the Dupira parser and the aboutness-based model.

1.1 About Dupira

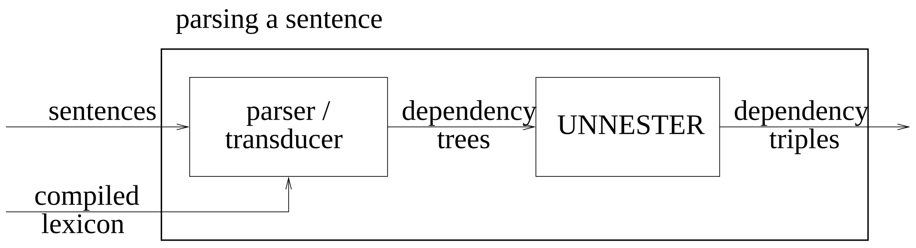
Dupira is a rule-based parser/transducer, which is generated by means of the AGFL parser generator[1] from the Dupira grammar (a weighted attribute grammar in the AGFL formalism (Koster 1992)) and from appropriate lexica and fact tables:

[1]`www.cs.ru.nl/agfl`



By means of *transductions* which are specified in the grammar (and can be modified), the parser transduces sentences to dependency trees.

The generated parser/transducer (or just parser, for short) is a Top-Down Chart parser, using the Best-Only heuristic (Koster et al. 2007). It is accompanied by a compiled lexicon.



The Dupira parser runs under both UNIX and MSWindows.

1.2 Robustness

A parser for IR applications should be robust in many senses. Technically, it should not go into uncontrolled behaviour for any input, not even when that input is machine generated. But in particular, it should possess lexical robustness and syntactic robustness.

1.2.1 lexical robustness

The lexicon of the parser, which is based on the lexicon of Amazon (Coppen 2002) has a high coverage (see **??**). In addition, the grammar also contains mechanisms for "guessing" the Part-of-speech and features of word forms that are not in the lexicon.

1.2.2 syntactic robustness

The parser must be able to accept sentences whose syntax is only marginally correct (colloquialisms, dialects, common errors) and to salvage recognizable parts from input that is syntactically incorrect. The parser should not pedantically demarcate the fine line between sentences that are correct and those that are incorrect, but

1. for each correct input the parser should give the right analysis
2. for foreseeable errors of input it should give the best analysis possible
3. for other incorrect input it should make an intelligent guess about the intended interpretation.

The key to this desirable behaviour is *over-generation*. Over-generation is not a large source of analysis errors, because unwarranted generalizations will simply not appear in the input. Correct input will be interpreted correctly, and incorrect input is incorrect anyway...

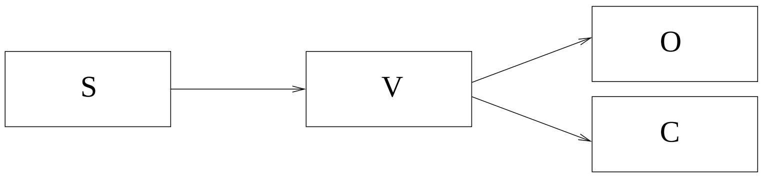
1.3 The rest of this paper

In the next sections we describe the aboutness-based dependency model and then, mostly by examples, the relation between Dutch sentences and the dependency trees generated from them. Finally we discuss the present status of Dupira and its potential applications.

2 Dependency graphs, trees and triples

The Dupira parser serves to transduce a Dutch sentence to its most probable dependency graph. By a *dependency graph* we mean a directed acyclic graph whose nodes are marked with words and whose arcs are marked with relators. Each arc is directed from one node (its *head*) to another (its *tail*).

The Dupira parser constructs for every sentence consisting of a subject S, verb V, object O and other complements C a dependency graph of (schematically, without the relators) the following form:



As an example, in the sentence de man eet zijn soep met een lepel (the man eats his soup with a spoon), de man is the subject, eet the verb, zijn soep the object and met een lepel a (prepositional) complement.

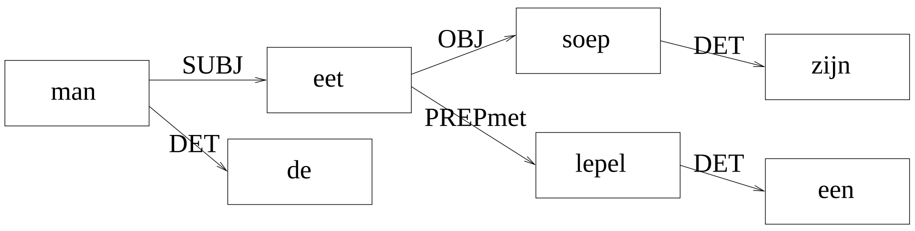
2.1 Dependency trees

In the output of the parser/transducer this dependency graph is represented by a *dependency tree*, one of the spanning trees of the graph, in the following linear notation:

```
[[man<DET de]<SUBJ[eet<OBJ[soep<DET zijn]<PREPmet[lepel<DET een]]]
```

In the examples of dependency trees and graphs, we will lemmatize the verb forms. The words in capital letters, prefixed by $<$ or $>$, are *relators*, and the prefix indicates their *direction*. The relator PREP carries the preposition met as a parameter (`<PREPmet`).

This dependency tree denotes the graph (this time including the relators)



There are many other trees representing the same dependency graph, because the order of the arcs is free and the direction of a relator can be inverted, e.g.

```
[[man<DET de]<SUBJ[eten<PREPmet[lepel<DET een]<OBJ[soep<DET zijn]]]
[[eten>SUBJ[man<DET de]<OBJ[soep<DET zijn]<PREPmet[lepel<DET een]]]
[[soep<DET zijn]>OBJ[eten>SUBJ[man<DET de]<PREPmet[lepel<DET een]]]
[[lepel<DET een]>PREPmet[eten>SUBJ[man<DET de]<OBJ[soep<DET zijn]]]
```

Any node of the graph may be taken as the root of a dependency tree for that graph.

2.2 Dependency triples

A dependency graph (or tree) can be *unnested* into a set of dependency triples, one for each arc in the graph. By a *dependency triple* we mean a triple of the form $<$head,relator,tail$>$. From the above graph (or from any of the above trees) the following triples can be obtained by unnesting:

```
[eten,OBJ,soep]
[eten,PREPmet,lepel]
[lepel,DET,een]
[man,DET,de]
[man,SUBJ,eten]
[soep,DET,zijn]
```

A dependency graph can be considered as an equivalence class of dependency trees, all consisting of the same dependency triples.

Those dependency triples come close to a semantical representation of the original sentence, and are suitable for many forms of further processing. However, they are derived by purely syntactic means.

2.3 Aboutness in IR

The state-of-the-art in Information Retrieval is to a large degree not based on semantics and logics, but rather on *aboutness* combined with statistics (see (Bruza and Huibers 1994) for the IR view on the notion of aboutness).

It has been shown in text classification experiments (Arampatzis et al. 2000) that in the bag-of-words document representation the words from the open categories (nouns, verbs, and to a lesser extent adjectives and adverbs) carry the aboutness of a text, the words from the closed categories are rightly considered as *stop words*. Similarly, it has been shown in (Koster and Beney 2009) that in text classification using the bag-of-triples representation those dependency triples *whose head and tail are both from an open category* carry the aboutness of the text, any other triples can be discarded as stop words.

2.4 The aboutness-based dependency model

The following table gives one example triple for each of the aboutness-carrying dependency relations:

subject relation	`[ik,SUBJ,verwacht]`
object relation	`[verwacht,OBJ,storm]`
predicate relation	`[Harry Mulisch,PRED,schrijver]`
attribute relation (adj.)	`[parser,ATTR,hybride]`
attribute relation (noun)	`[parser,ATTR,core]`
prepos relation (noun)	`[storm,PREPaan,kust]`
prepos relation (verb)	`[ga,PREPop,bezoek]`
prepos relation (adj.)	`[verwant,PREPmet,zuster]`
modifier relation (verb)	`[ga,MOD,graag]`
modifier relation (adj.)	`[grote,MOD,erg]`
modifier relation (adv.)	`[graag,MOD,niet]`

The prepositions like aan (on) are attached as parameters to the PREP relators.

The aboutness-based dependency model (ABDM) is different from traditional linguistically-motivated dependency models which set out to describe the relations between *all* of words in the sentence, including punctuation, such as Minipar (Lin 1998) and Link Grammar (Sleator and Temperley 1995); but it is close to the "collapsed" output version of the Stanford Dependency Parser (Marneffe and Manning 2009). It is meant for applications in IR, which require a representation of the aboutness of the text, rather than a detailed syntactic analysis.

The aboutness of most modifier triples is dubious, and even the negation could be elided as being a stop word. Pronouns are members of a closed class, but they serve an important role as heads of certain NP's, therefore we include them among the nouns and employ a generous notion of 'content word'.

2.5 Other dependency relations

The pure ABDM model (only relations between content words) abstracts from many aspects of the sentence which are relevant to its semantics and pragmatics: time, mood, modality, definiteness, the emphasis indicated by topicalization. Pure ABDM ignores discourse relations (which connect not words but whole clauses).

For linguistic applications, and for applications such as Information Extraction, text paraphrasis or translation, much more detailed dependency models will be appropriate than the one described here. Of course, the Dupira parser has to recognize the complete synactic structure of each sentence anyway, and its transduction may be modified to produce also other relations involving non-content words.

In some of the examples we will also give the DET relations, for which only the head is a content word, the tail being taken from a (small and closed) collection of function words (determiners). We are still experimenting with the production and use of other relations.

2.6 Factoids

The above dependency relations serve to express only the *factual* content of the sentence, without further frills like the time, mood and modality of the verbs. The linear word order is lost in the graph representation, as well as the topicalization and the argumentation structure (for as far as it is expressed by conjunctions). Aboutness-based dependency graphs are well-suited to represent *factoids*, simple sentences expressing a (purported) fact. A typical factoid pattern is 'who did what to whom with what', in which relevant phrases are filled in for the 'who', 'whom' and 'what'. The schematical sentence structure described in 2.1 corresponds to a factoid.

2.7 Normalizing transformations

In the transduction, Dupira applies certain aboutness-preserving normalizing transformations in order to map sentences with the same aboutness onto the same dependency graph.

The *syntactical normalization* is expressed in the grammar underlying our parser, using compositional transduction: every construct in the grammar is described along with its transduction to the output format (in our case dependency graphs with content words as heads and tails).

- elements which do not contribute to the aboutness of the text are elided: articles and determiners, quantifiers, auxiliary verbs, conjunctions - which is much like applying a stop list;
- topicalization and other variations in word order are eliminated;
- embedded constructions such as relative clauses and participial constructions are expanded into additional basic sentences (yielding additional SUBJ and OBJ relations);
- a preposition linking two content words is treated as a parameter to the PREP relation, bringing the content words storm and kust together into one triple `[storm,PREPaan,kust]`, rather than in two triples like

  ```
  [storm, PREP,aan] [aan,TARGET,kust]
  ```

  in which the preposition (a non-content word) occurs both as a head and as a tail;
- one of the most effective normalizing transformations is *de-passivation*: transforming a passive sentence into an active sentence with the same aboutness (see 3.4). By this transformation, the English phrase renal damage caused by Aspirin is considered equivalent to Aspirin may cause renal damage.
- *Morphological normalization* by lemmatization is (optionally) applied to the nouns, verbs and adjectives occurring in the resulting triples.

Such normalizing transformations serve to improve the recall in IR applications, while surrendering little or no precision.

3 Phrases and their transduction

In this section, we will describe by examples the transduction that Dupira gives for the major constituents of Dutch sentences.

3.1 Noun phrases

The Noun Phrase (NP) typically has a noun as its head and some number of modifiers attached to that head. As described in 3.5, it may also have multiple (coordinated) heads.

In the examples we will also show some DET, QUANT and TEMP relations, which do not belong to the aboutness-based model. Only the verbs are lemmatized. We do not show the (optional) position numbers of heads and tails, which serve to distinguish different occurrences of the same word in the sentence.

- The NP may contain pre-modifying determiners, quantifiers and adjectives:

  de mooiste acht mannen heb ik gezien
  the beautifullest eight men have I seen
  `{[zien>SUBJ ik<OBJ[mannen<QUANT acht<ATTRmooiste<DET de]]}`

- an adjective phrase (AP) may occur as a post-modifier:

  aan **een man afkerig van roem** vertrouw ik het toe
  to **a man averse of celebrity** entrust I it ( )
  ```
  {[toevertrouwen>SUBJ ik<OBJ het<PREPaan[man<ATTR[afkerig<PREPvan
  roem]<DET een]}
  ```

- a preposition phrase (PP) may occur as a post-modifier:

  de man met de zeis komt voorbij
  the man with the scythe comes by
  `{[man<PREPmet[zeis<DET de]<DET de]<SUBJ voorbijkomen}`

- a relative phrase (RP) may occur as a post-modifier:

  ik zag **de man die jij bedoelt**
  I saw **the man that you mean**
  `{ik<SUBJ[zien<OBJ[[man<DET de]>OBJ[bedoelen>SUBJ jij]]]}`

3.2 Verb phrases

In transducing a verb phrase, the infinitive of the main verb is taken as its head, and any auxiliary verb or copula is eliminated (but it may be transduced as a modifier to the main verb).

We give just one example for each of the verb transitivities, using the basic SVOC order. Of course, the same transitivities may be used in the other word orders; furthermore the complements of a verb may be permuted in intricate ways. In each example the verb complements are shown in boldface, including verb particles.

- no – no object

  de boot komt **morgen aan**
  the boat comes **tomorrow on**
  `{[boot<DET de]<SUBJ[aankomen<TEMP morgen]}`

  The element morgen is a temporal adverb, acting as a modifier to the verb, for which we use the relation TEMP (it is not the noun morgen).

- do – direct object

  ik geloof **daar niets van**
  I believe **there nothing of**
  `{ik<SUBJ[geloven<OBJ niets<PREPvan daar]}`

  Notice the split PP, a peculiar feature of Dutch.

- io – indirect object

  zijn benadering beviel **mij niet**
  his approach pleased **me not**
  `{[benadering<DET zijn]<SUBJ[bevallen<PREPaan mij<MOD niet]}`

  The indirect object may be realized by an NP or a PP with aan (to); it is transduced to the latter.

- pred – predicate (a predicate may result in an PRED or ATTR relation)

  hij blijkt **een sukkel**
  he appears **a sucker**
  `hij<SUBJ[blijken<PRED[sukkel<DET een]]`
  zijn gezicht werd **helemaal groen**
  his face became **wholly green**
  `{[gezicht<DET zijn]<SUBJ[worden<ATTR[groen<MOD helemaal]]}`

- io+pred

  dat lied komt **mij bekend** voor
  that song comes **me known** for
  `{[lied<DET dat]<SUBJ[voorkomen<ATTR bekend<PREPaan mij]}`

- io+do

  ik kan **hem niets** weigeren
  I can **him nothing** refuse
  `{ik<SUBJ[weigeren<PREPaan hem<OBJ niets]}`

Because there is no case marker for NP's, objects and short indirect objects often can not be distinguished, an unfailing source of ambiguity.

- inf – (bare) infinitive

  ik wil **graag helpen**
  I want **please help**
  `{ik<SUBJ[helpen<OBJ HET<MOD graag]}`

  The marker HET (it) indicates the dropped object of helpen.

- te inf – infinitive with 'te' (to)

  ik vermeed **hem aan te kijken**
  I avoided **him at to look**
  `{ik<SUBJ[vermijden<SUBJ[aankijken<OBJ hem]]}`

  The subject of the main verb is also the subject of the infinitive.

The other transitivities are described in the next section. They have the property that additional dependency triples may be extracted.

3.3 Capturing additional dependency triples

For the two following transitivities, we use semantic knowledge to capture additional SUBJ triples.

- do+inf – object raising

  wij helpen **jullie de fietsen dragen**
  we help **you the bicycles carry**
  `{wij<SUBJ[helpen<OBJ[jullie<SUBJ[dragen<OBJ[fietsen<DET de]]]]}`

- do+te+inf

  dat helpt **jou om deze terugslag te verwerken**
  that helps **you for this setback to process**
  `{dat<SUBJ[helpen<OBJ[jou<SUBJ[verwerken<OBJ[terugslag<DET deze]]]]}`

- do+pred

  daarom noem ik **mij katholiek**
  therefore call I **me catholic**
  `{[[noemen>SUBJ ik<OBJ[mij<ATTR[katholiek]]]<PREPom daar]}`

  Notice that the attribute is not attached to the verb but to the object, whose properties it describes.

3.4 Passive and impersonal

Passive sentences are transduced to the corresponding active sentence: the syntactic subject becomes the object, and the agent a subject. An empty agent is transduced to an impersonal MEN (ONE).

ik ben gekomen (active perfective)
I am come
`{[ik]<SUBJ[komen]}`
ik ben genomen (passive perfective)
I am taken
`{[ik]>OBJ[nemen>SUBJ MEN ]}`
ik kan gezien worden door de mensen
I can seen become by the people
`{[ik]>OBJ[zien>SUBJ[mensen<DET de]]}`

Impersonal sentences look in Dutch like passive sentences, but without subject.

er wordt gesproken
there is spoken
`{[spreken>SUBJ MEN ]<MOD er}`

3.5 Coordination

Several constructions can be coordinated, using comma's or certain delimiters. It is e.g. possible to have more subjects for one clause:

de regering en de oppositie weigerden toe te geven
the government and the opposition refused to yield

Many more constructs can be coordinated: sentences, nouns, adjectives, verbs, adverbs, et cetera. In the dependency tree produced, the coordination is indicated by an *or-operator* `|` between the heads:

```
{[[regering<DET de]|[oppositie<DET de]]<SUBJ[weigeren]<SUBJ[toegeven<OBJ HET]}
```

resulting in the triples

```
[regering,SUBJ,weigeren] [regering,SUBJ,toegeven] [regering,DET,de]
[oppositie,SUBJ,weigeren] [oppositie,SUBJ,toegeven] [oppositie,DET,de]
[toegeven,OBJ,HET]
```

The example shows clearly that aboutness is not the same as semantics: the text is *about* a government yielding (something unspecified), even though the semantics makes clear it did not yield. In the same way, the famous sentence "this is not a pipe" is *about* a pipe, in spite of the (semantical) negation.

4 Status of Dupira

Although older versions of Dupira have been used by others, the current version is the first one that we make available publicly. It is reasonably stable, reasonably fast and reasonably complete. For the parser/transducer to be suitable for serious applications, it must have enough coverage, accuracy and speed, and the dependency model must be suitable for the application.

4.1 Lexical coverage

The lexical coverage of Dupira as measured on a part of the old NRC corpus is presently about 98.6 % (total number of word tokens 865342, out-of-lexicon word tokens 12252)

It is easy to add more lexical material, but this feels like a bottomless pit. In Dutch, as in some other languages, new nouns can be freely invented, by simply glueing other nouns together. Something similar applies to other open word categories. Therefore Dupira makes use of *robust guessers* for nouns, verbs and adjectives.

Since other Dutch parsers have to cope with the same problem, some authority managing an up-to-date and freely downloadable lexicon based on very large corpora and including Part-Of-Speech information would be welcome. Until then, each new corpus will have to be filtered for new words.

4.2 Accuracy

We must apologize that we have not yet had the opportunity to make a serious evaluation. Dupira is still in an experimental stage. Because it goes beyond syntax analysis and performs many aboutness-preserving transformations, no gold standard is available for measuring its accuracy. We have to construct our own standards. Presently, a student is building a script to convert Alpino triples to Dupira triples, which might help

We present here some very initial results obtained on the regression standard (1069 sentences, 7052 words, resulting in 4874 triples) which was used in the development of Dupira. It contains one or more examples for each rule in the grammar, as well as examples for some constructs not yet implemented.

It should be realized that *this standard is not representative for the language in general*, but for the most important constructs that we wanted to cover. It consists of short, syntactically correct but relatively complicated sentences. Lower sentences will probably have a lower accuracy.

The accuracy of the parser was measured on the regression standard using the technique described by Lin (1998), as the precision, recall and F1 found when comparing the triples generated by the parser from the sentences in the standard with the triples in that standard. The following table summarizes the result, and gives also a breakdown according to the various relations.

relator	total	common	missed	wrong	precision	recall	F1
all	4874	3605	670	599	0.858	0.843	0.850
SUBJ	1505	1197	177	131	0.901	0.871	0.886
OBJ	835	596	126	113	0.841	0.825	0.833
PREP	817	521	150	146	0.781	0.776	0.779
DET	635	570	38	27	0.955	0.938	0.946
MOD	531	378	66	87	0.813	0.851	0.832
ATTR	326	210	75	41	0.837	0.737	0.784
PRED	154	73	33	48	0.603	0.689	0.643
TEMP	71	60	5	6	0.909	0.923	0.916

This experiment shows the accuracy obtained when considering only the first, most highly ranked analysis. This is how the parser is generally meant to be used.

We have also measured the accuracy obtained when forcing the parser to make use of the triples in the standard as an oracle. The *angelic* oracle forces it to produce for each sentence, for as far as this is possible according to the grammar, the triples from the standard. In this case, the accuracy obtained was 90.5%. It is not 100% because of errors in the grammar (ongoing work), incompleteness of the grammar (the standard contains also examples of constructions that have not yet been implemented) and errors in the standard (for some constructions it is by no means obvious what should be produced). We are working to increase this angelic accuracy.

The *demonic* oracle forces it to avoid, for as far as this is possible according to the grammar, the triples from the standard. In this case, the accuracy was 50.6%. About half of the triples are unavoidable.

It is important to realize that Dupira is not a probabilistic parser, and that therefore it was not “trained” on the regression set. It is a rule based parser, which makes use of the subcategorization information provided by the Amazon lexicon.

In case of ambiguity, penalties given in the grammar or derived from lexical frequencies are used to choose the best (the most probable) analysis. For non-ambiguous sentences, no such information is necessary. As the accuracy shows, the accuracy of Dupira is already quite reasonable, but with the aid of some disambiguation device accuracy could be further improved by up to 4.6% (and more if the grammar is improved).

4.3 Speed

Parsing the NRC corpus (883251 words) took 4263 seconds, about 200 words per second. Unnesting the resulting trees took about 35 seconds, resulting in 597838 dependency triples.

5 Potential applications of Dupira

The aboutness-based dependency model (ABDM) of Dupira tries to express the aboutness of the sentence, but abstracts from most of its syntactical, semantical and pragmatical aspects. For many applications, an extension of the model with further relations is needed. But even a strict ABDM parser has many applications:

- **Document Classification** – it was shown that adding triples to the bag of words representation significantly improve the classification accuracy on patent abstracts (Koster and Beney 2009) using the aboutness-based parsers AEGIR for English and FR4IR for French

- **Text Mining** – our first aboutness-based dependency grammar EP4IR, the predecessor of AEGIR. was used in the prototype of the Literature Search Engine PHASAR/MEDLINE (Koster et al. 2006)

- **natural-language interfaces for Information Systems** – EP4IR was also used successfully for dialogue parsing (Hindriks et al. 2007)

Other promising application areas are:

- Factoid Extraction – our present work using Dupira in a small Google-sponsored project

- Question Answering was shown to benefit from using dependency triples (cf. (Bouma 2005))

- sentence summarization and sentence fusion (cf. (Filippova and Strube 2008))

- documentation support and controlled language.

6 Conclusion

Dupira is the second dependency parser for Dutch to become available in the public domain. The first useful parser for Dutch was Alpino which has been available for over a decade. Alpino is more suitable for linguistic applications than Dupira. Dupira however was developed specifically for certain applications in Information Retrieval. The two parsers are not easily comparable, because they are oriented towards different tasks. They work in very different ways and have different strengths and weaknesses, which suggests that for some applications it might be useful to combine them. And the Dutch language is important enough to deserve more than one parser.

The Dupira parser is based on the notion of *aboutness*, and is therefore most suited for parsing factual text. Rather than producing a detailed syntactical description of the structure of the sentence, it produces dependency triples which are close to a semantical representation. It is still under development, first enlarging its syntactical coverage and then raising its accuracy by adding a disambiguation device. But in its present state, it is already suitable for practical applications in Information Retrieval and Information Analysis.

6.1 Availability

The present version 0.9 of the Dupira Parser is released as-it-is into the public domain[2] for the benefit of researchers and practitioners in Information Retrieval and Linguistics. Extended and corrected versions will appear when possible. We hope in particular that other researchers will join in applying, evaluating and improving Dupira, and will contribute to making Dupira a high quality resource for the common benefit.

References

Arampatzis, A., Th.P. Van der Weide, C.H.A. Koster, and P. van Bommel (2000), An evaluation of linguistically-motivated indexing schemes, *Proceedings of BCS-IRSG 2000 Colloquium on IR Research*, Sidney Sussex College, Cambridge, England.

Bouma, G. (2005), Question answering for dutch using dependency relations, *In: Working Notes for the CLEF 2005 Workshop*, Springer.

Bruza, P. and T.W.C. Huibers (1994), Investigating aboutness axioms using information fields, *Proceedings SIGIR 94*, pp. 112–121.

Coppen, P.A. (2002), Het geheim van de oude dame. de nijmeegse parser amazon., *Nederlandse Taalkunde, 7:4*.

Filippova, K. and M. Strube (2008), Sentence fusion via dependency graph compression, *EMNLP*, pp. 177–185.

Hindriks, K.V., S. Hoppenbrouwers, C.M. Jonker, and D. Tykhonov (2007), Automatic issue extraction from a focused dialogue, *Proceedings NLDB 2007*, Springer LNCS 4592, pp. 204–217.

Koster, C.H.A. (1992), Affix Grammars for Natural Languages, *Attribute Grammars, Applications and Systems*, Springer LNCS 545, pp. 469–484.

Koster, C.H.A. and J.G. Beney (2009), Phrase-based document categorization revisited, *Proceedings CIKM 2009*, pp. 49–55.

Koster, C.H.A., M. Seutter, and O. Seibert (2006), The phasar search engine, *Proceedings NLDB 2006*, Springer SLNC 3999, pp. 141–152.

Koster, C.H.A., M. Seutter, and O. Seibert (2007), Parsing the medline corpus, *Proceedings RANLP 2007*, pp. 325–329.

Lin, D. (1998), Dependency-based evaluation of minipar.

Marneffe, M.C. De and C.D. Manning (2009), The stanford typed dependencies representation, pp. 1–8.

Sleator, D.D. and D. Temperley (1995), Parsing english with a link grammar.

[2] `www.cs.ru.nl/agfl/`