Evaluating a Robust Parser for Italian

R. Basili, M.T. Pazienza, F.M. Zanzotto

Dipartimento di Informatica, Sistemi e Produzione,
Universita' di Roma Tor Vergata (ITALY)
{basili,pazienza, zanzotto}@info.utovrm.it

Abstract

Evaluation of performance of a NLP system is crucially related to the evaluation of the core parsing component. Large scale evaluation is thus required to fully measure the quality of the underlying grammatical description and the robustness and speed of the parser. The evaluation of performances requires the availability of large sets of sentences where syntactic information (possibly on a theory free basis) has been manually annotated. Since existing corpora are annotated on a constituency oriented basis (e.g. Penn TreeBank for English), the evaluation of parsing systems based on different paradigms requires a mapping between the different grammatical representations. Furthermore, the parse description (usually as a set of trees and/or forests) is unsuited for some grammatical information (i.e. dependencies or clause boundaries) that is usually extracted and recognized in robust parsers (e.g. chunker or link grammar parsers). In this paper the experience in the evaluation of a robust parser for the Italian language is described. The evaluation framework (i.e. development of specific test data, performance indexes and results) will be defined and discussed. Aspects and outcomes related to systematic criteria of creation and annotations of large test data sets will be presented.

1. Introduction

Evaluation of the performances of a NLP system is crucially related to the evaluation of the core parsing component. Real scale systems are tightly committed to significant sets of sentences (thousands to millions). Large scale evaluation is thus required to fully measure the quality of the underlying grammatical description and the robustness and speed of the parser. Parsing systems are expected to produce on sentences a variety of grammatical information: recognition and typing of constituents, determination of grammatical relations and other additional features depending upon the underlying linguistic theory. The evaluation of performances requires the availability of large sets of sentences where syntactic information (possibly on a theory free basis) has been manually annotated.

Since existing corpora are annotated on a constituency oriented basis (e.g. Penn TreeBank for English (Marcus et al. 1993)), the evaluation of parsing systems based on different paradigms requires a mapping between the different grammatical representations.

Furthermore, the parse description (usually as a set of trees and/or forests) is unsuited for some grammatical information (i.e. dependencies or clause boundaries) that is usually extracted and recognized in robust parsers (e.g. chunker or link grammar parsers).

This has significant consequences on the selection of performance indexes. Some measure m (depending upon the set of available annotations) may be unsuited to capture the nature of grammatical phenomena that are output by a given parser. Sometime a given measure m is even inapplicable. For example, it is not always possible to detect some grammatical dependencies (e.g. the dependence between a constituent and the subject of a subordinate clause) over a parsing annotation scheme based on a set of constituents (e.g. $SBAR$ in the Penn Treebank). On the contrary a (possibly robust) parser extracting grammatical relations between textual units (e.g. a link grammar parser, (Grinberg et al. 1995)) is tightly committed to such information. Any evaluation of the latter making use of such input data is incomplete and may result in a wrong measure. Relaxing the evaluation procedure (e.g. by eliminating unlucky cases) is not always possible for sake of consistency and completeness (coverage). Some indexes may thus be too penalizing or optimistic with respect to some crucial phenomena.

It is also difficult to compare performance of different parsing methods making use of different knowledge sources (e.g. lexicalized vs. not lexicalized approaches). The kind of dependence of a parser on specific lexical information should be captured by specific performance indexes.

In this paper the experience in the evaluation of a robust parser for the Italian language is described. **Chaos** is a Chunk-Oriented Analysis System for the syntactic processing of Italian texts, and it is described in (Zanzotto,1997). Section 2 discusses the general parsing framework of Chaos.

Problematic issues in the performance evaluation experiments have been:

- the absence of extensive test data for the Italian language.
- the suitable information expected to be found in annotation with respect to the specific nature of $Chaos$.
- portability to sublanguages

The evaluation framework has been originally defined according to some pre-existing annotated corpora. A source corpus of Italian sentences (described in (Basili et al.,1997)) has been used as reference information. These data were generated by validating parse trees produced by a DCG parser. The development of specific test data for Chaos has been carried out. Methods and results are discussed in Section 3. Performance indexes on Chaos output have been measured. Results and methods are defined and discussed in Section 4.

2. A parsing system for Italian language

The system whose performances are under investigation is Chaos, a chunk-oriented system for syntactic analysis of Italian language, described in (Zanzotto, 1997). The system is the result of the integration of two basic principles: stratification and lexicalization.

2.1. Chaos: a stratified and lexicalized parsing system

The proposed method is triggered by the basic assumption that it is possible to define an interesting intermediate level between words and sentences. Usually a sentence is thought as a sequence of words, even if, in its written version, it is a sequence of characters. This because subsequences of characters are considered unambiguously as units, even when they show different behaviors in different sentences. The idea is to build up a machinery, computationally simple as a tokenizer, able to group words of a sentence in functionally justified larger entities. For example, given the following NL fragment

coordinare le fasi successive di lavoro …,
** coordinate the activities following of work …,*
i.e. *coordinate the following work activities …,*

any potential syntactic interpretation unambiguously group words as follows:

```
[coordinare][le fasi][successive][di lavoro]
```

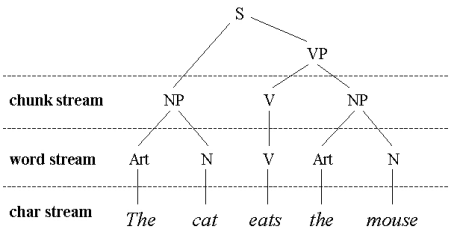

Figure 1: Stratification in a simple derivation tree

The grouping of words has been introduced as a level of sentence interpretation in (Abney,1996), where groups are called $chunks$. The chunk level of interpretation is thus the intermediate level between words and sentences, as shown in the simpler example of fig. 1.

In (Federici et al.,1996), the idea of chunking is successfully applied to Italian.

The *stratification* proposed in Chaos is mainly inspired by the specific role played by verbs. Verbs play a crucial role in the recognition of clauses as well as syntactic relations established between content words in a sentence. Ambiguity is controlled via grammatical rules related to the verb

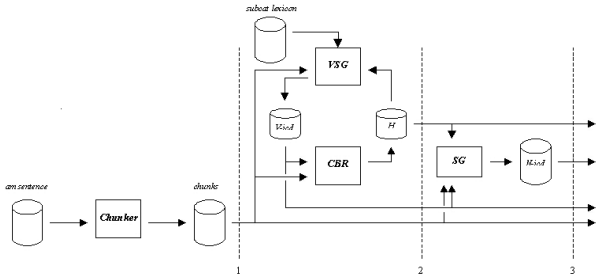

Figure 2: CHAOS: the functional architecture

functional class and specific to each verb. Main grammatical rules for verbs used in Chaos are the verbal subcategorization frames. The constraints they impose are integrated in the recognizer of clauses and guide the determination of clause maximal and minimal boundaries.

Finally, syntactic representation of the analyzed text, is completed by recognition of other set of *dependencies between chunks* (e.g. post nominal prepositional phrases) by a technique already proposed in the SSA parser (Basili et al.,1992,1994): in this phase dependencies between chunks within the discovered clauses (i.e. infra-clause dependencies) are extracted from the source text.

In Fig. 2 the overall architecture of the system Chaos is depicted. *am sentences* are tokenized and morphologically annotated sentences, given as input to the Chunker. Chunks are used as input to the *Clause Boundary Recognition* module (CBR) aiming to recognize clauses and structure them in a hierarchy (see H is Fig. 2). The recognition of clauses is integrated with a special purpose parser (Verb Shallow Recognizer, VSG) aiming to detect relations between a verb and members of its subcategorization pattern (i.e. its arguments). The interaction between the CBR and VSG provides a combined recognition of the clause hierarchy and the set of argumental dependencies of verbs. The *Shallow recognizer* (SG) is finally triggered by Chunks, the hierarchy H and the already extracted argumental relations (verbal icd's).

2.2. The derived grammatical information

The final grammatical sentence representation is ensured by *planar graphs* (i.e. graph with no crossing links) whose nodes are chunks and oriented edges are inter-chunk dependencies (icd).The planar graph representation of grammatical information is discussed in (Grinberg et al., 95). Icds are classified according to their degree of ambiguity. Icds obtained by VSG (i.e. argumental ones) are considered unambiguous. The ones obtained by SG may be ambiguous: they are assigned to plausibility scores (as described in (Basili et al.,1992,1994)) inversely proportional to their degree of ambiguity. Furthermore, the structure of embedded clauses is inferred and expressed into a hierarchical structure (see Appendix for an extended example).

3. Development of test data for the Italian language

The test set has been built in two ways: manual annotation of sentences and automatic translation of pre-existing annotated corpora. The result is a test set of nearly 130 annotated sentences belonging to different corpora (e.g. ECRAN made of financial news, ENEA on environment and Legal, a collection of V.A.T. laws). The manual annotation of sentences allows to express information in the suitable way. Since Chaos expresses grammatical information through planar graphs whose nodes are the chunks representing the sentence and edges are the icds (e.g. inter-chunk dependencies), the annotation of the 20 sentences extracted from the ECRAN Corpus and the 30 from ENEA followed the same schemata: chunks and inter-chunk dependencies (icd) are fully compiled in explicit annotations. The equivalence between parser and manual annotations allowed to estimate performances through the usual $Recall$ and $Precision$ indexes.

On the other hand, as existing syntactically annotated language resources were available (i.e. the 80 sentences from the Legal corpus), experiments on them were also run. The syntactic information of the 80 sentences was represented via constituency based parse trees. The main grammatical mismatches related to:

- mismatch between constituents in the parse trees and chunks
- mismatch between dominance relations between constituents and inter-chunk dependencies
- mismatch between dominance relations between constituents and infra-chunk dependencies (e.g. complex nominal forms)

In order to get a uniform measure (i.e. matches of the different annotations) parse trees as well the Chaos planar graphs have been compiled into a word oriented planar graph annotation. In the resulting graphs the following information can be found:

- nodes representing words
- edges as relations between word couples

This latter annotation, as already stated, is inspired by the idea that each word in the sentence is grammatically related to one other word, e.g. its head, and no word, except the null markers introduced, escapes this rule. Thus, the natural operation for the translation of the trees is to operate on the underlying grammar by making explicit use of the rule heads. The following simple rewriting algorithm has been applied to parse trees.

Tree flattener(tree,graph,head)
begin
father := father(tree);
sons := Sons(tree);
if $sons = \emptyset$
then begin graph :=(father,$\emptyset$); head := father; **end**
else
begin
for each $son_i \in sons$
do Tree flattener(son_i,graph(son_i),head(son_i));
let $father \rightarrow father(son_1)...father(son_n)$ be
the rule of the grammar where $father(son_h)$
is the marked head;
$edges_{rule}$:=
$\{(head(son_h), head(son_i)) \quad | \quad h \neq i, son_i \in sons\}$;
nodes :=$\bigcup_{son_i \in sons} nodes(graph(son_i))$;
edges :=$\bigcup_{son_i \in sons} edges(graph(son_i))) \cup edges_{rule}$;
graph :=(nodes,edges);
end
end

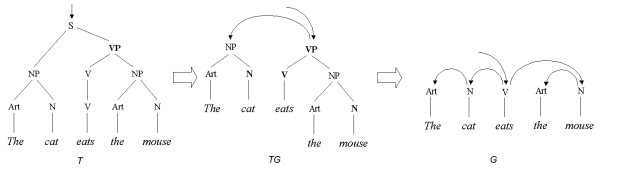

Figure 3: Tree Flattener: an example

Fig. 3 shows how the algorithm operates on the simple sentence of fig. 1: the tree T is transformed in the corresponding graph G via TG. The underlying grammar where heads of rules are emphasized in bold is the following:
$S \rightarrow NP$**VP**
$VP \rightarrow$ **V**NP
$NP \rightarrow Art$**N**

The translation of the Chaos planar graphs to the word planar graphs is immediate: for each chunk its head has been retained in the target graph while each icd is mapped into a corresponding edge between the related couple of chunk heads.

4. Performance Evaluation of Chaos

As an exact match exists between the set of correct grammatical information ($graph_{correct}$) (i.e. correct parser trees mapped into graphs) and the set of those recognized by Chaos ($graph_{Chaos}$), different evaluations are possible. That is, different sets of interesting phenomena can be captured. The whole information contained in a tree is a bias for the focus of the experiments but several measures are possible. We concentrated in those dependencies that are relevant to application tasks like Information Extraction. For example the recognition of significant events (i.e. management successions) are crucially dependent on the ability of the parsing

system to capture verbal dependencies (e.g. argumental PPs or temporal expressions).

So specific measures for classes of icd have been carried out in order to evaluate either an intrinsic ability of the grammar to capture those dependencies, as well as the extrinsic character of the whole parser (i.e. capability to focus on application related aspects). As

$$graph_{correct} = (nodes_{correct}, edges_{correct})$$

and

$$graph_{Chaos} = (nodes_{Chaos}, edges_{Chaos})$$

are the representation of the sentence S, since $nodes_{correct}$ = $nodes_{Chaos}$, the following definition is possible:

$$Recall = \frac{|edges_{correct} \cap edges_{Chaos}|}{|edges_{correct}|}$$
$$Precision = \frac{|edges_{correct} \cap edges_{Chaos}|}{|edges_{Chaos}|}$$

The only drawback of this definition is that all the information described by the graph is equivalently represented. The difference existing between inter- and infra-chunk dependencies is not stressed. The evaluation experiments gave the results reported in Tab. 1). Recall and precision have been estimated, separately, for each annotated test set (Legal, ENEA and ECRAN). Values of recall and precision have been investigated in different grammatical levels (see fig. 2), e.g. (1) Chunk level, (2) Verbal level, (3) unambiguous links and ambiguous ones. Note that, as the system increases the grammatical coverage on input sentences (i.e. higher recall), precision decreases. Nevertheless, the measured performances are significantly good on the set of covered phenomena (about 80% coverage, with a corresponding 82% precision). Table 1 also suggests slightly different behaviors on different domains, although similar performances are obtained.

Table 1: Performance figures on the three corpora

	Legal		**ENEA**		**ECRAN**	
#words	1,460		1,149		494	
#sentences	80		56		20	
	Rec	Prec	Rec	Prec	Rec	Prec
Chunk	0.39	1	0.41	1	0.37	1
Verbal	0.55	0.95	0.60	0.99	0.64	0.99
Certain	0.73	0.92	0.76	0.95	0.77	0.94
Final	0.80	0.82	0.85	0.83	0.81	0.85

5. Conclusions

In this paper the experience in the evaluation of a robust parser for the Italian language has been described. The evaluation framework (i.e. development of specific test data, performance indexes and results) has been discussed. Critical issues concerning availability of large test data sets for the Italian language, the problems in using usual (i.e. constituency based) annotation information as a test golden standard and the dependence of the measures on the underlying grammatical output of a robust parser have been stressed as critical aspects of the evaluation process. The outcomes of systematic experiments in annotating large test data sets of Italian texts and in measuring several classes of phenomena have been reported. The experience demonstrated that any annotation schema for Italian test data made exclusively of constituency based information is at least unsuited for large scale evaluation of robust parsing methods and systems. An algorithm applied to parse trees to compile a dependency oriented form of annotation is described. A specific data structure (planar graph) has been here adopted for tests. A better valuation process has been thus defined and first measurements show significant expressiveness of the underlying system behavior. Different measures over different linguistic phenomena (verbal vs. nominal inter chunk dependencies) suggest the suitable changes to be made to the underlying grammatical description.

6. References

(Abney, 96) Steven Abney, Part-of-speech tagging and partial parsing in: S.Young and G.Bloothooft, Corpus-based methods in language and speech, Kluwer academic publishers, Dordrecht, 1996

(Basili et al., 92) Roberto Basili, Maria Teresa Pazienza and Paola Velardi A shallow syntactic analyser to extract word association from corpora, Literary and linguistic computing, 1992, vol.7, n.2, 114-124

(Basili et al., 94) Roberto Basili, Maria Teresa Pazienza and Paola Velardi, A (not-so) shallow parser for collocational analysis, Proceedings of COLING '94, Kyoto, Japan, 1994.

(Basili et al.,97) Roberto Basili, Marie-Helene Candito, Maria Teresa Pazienza and Paola Velardi, Evaluating the information gain of probabilty-based PP-disambiguation methods, New Methods in Language Processing, UCL Press 1997.

(Carroll and Briscoe 96) John Carrol and Ted Briscoe, Robust parsing - a brief overview, Workshop on robust parsing ESSLLI 1996.

(Federici et al., 96) Stefano Federici, Simonetta Montemagni and Vito Pirrelli, Shallow parsing and text parsing: a view in underspecification in syntax, Workshop on robust parsing, ESSLLI 1996

(Goodman,1996) Joshua Goodman, Parsing Algorithms and Metrics, Proceedings of 34th Annual Meeting of ACL, Santa Cruz, CA, 24-27 June, 1996.

(Grinberg et al., 95) Dennis Grinberg, John Lafferty and Daniel Sleator, A robust parsing algorithm for link grammars, Proceedings of the fourth international workshop on parsing technologies, Prague, 1995

(Marcus et al. 93) Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz: Building a Large Annotated Corpus of English: The Penn Treebank, in Computational Linguistics, Volume 19, Number 2 (June 1993), pp. 313–330 (Special Issue on Using Large Corpora)

(Zanzotto,1997), Fabio Massimo Zanzotto, Una Metodologia Stratificata per la Analisi del Linguaggio Naturale: il sistema CHAOS, Thesis Dissertation, Faculty of Engineering, University of Roma, Tor Vergata, 1997.

7. Appendix 1 : Parsing Results for a Complex Sentence

Sentence:

Supponendo di avere a disposizione un certo budget economico relativo ad un intervento di risanamento ambientale, ACE s.p.a. intende programmare e coordinare le fasi successive di lavoro che permetteranno, infine, di elaborare un progetto esecutivo d'intervento.

Assuming to have at disposal a certain budget level for an environmental recovery action, ACE spa, intends to prepare the necessary plan to coordinate the following work activities, which will end in the completion of the operational implementation project.

7.1. Chunks

```
[supponendo /VerGer/1][di avere /VerInf/2]
[a disposizione /Prep/3]
[uncerto budget /Nom/4][economico /Agg/5]
[relativo /Agg/6][ad un intervento/Prep/7]
[di risanamento/Prep/8][ambientale /Agg/9]
[, /CongCo/10][ACE s.p.a. Nom/11]
[intende /VerFin/12][programmare/VerInf/13]
[e /CongCo/14][coordinare /VerInf/15]
[le fasi /Nom/16][successive /Agg/17]
[di lavoro /Prep/18][che /NomRel/19]
[permetteranno /VerFin/20][, /CongCo/21]
[infine /Avv/22][, /CongCo/23]
[di elaborare /VerInf/24]
[un progetto /Nom/25][esecutivo /Agg/26]
[d' intervento /Prep/27] [. /CongCo/28]
```

7.2. Clause Hierarchy

```
clause(head(NO_ONE),boundaries(0,28),
 [clause(head(12),boundaries(0,28),
  [clause(head(13),boundaries(13,28),
   [clause(head(15),boundaries(15,28),
    [clause(head(20),boundaries(19,28),
     [clause(head(24),boundaries(24,28),
      [])
     ])
    ])
   ]),
   clause(head(1),boundaries(1,6),
    [clause(head(2),boundaries(2,6),
     [])
    ])
  ])
 ])
```

7.3. Verbal dependencies

```
link(0,12,MainClause).
link(2,4,G_V_Ogg).
link(12,11,G_V_Sogg).
link(12,13,G_V_Fras).
link(15,16,G_V_Ogg).
link(19,20,Clausole).
link(20,24,G_V_Fras).
link(24,25,G_V_Ogg).
```

7.4. Unambiguous ICD

```
link(4,5,G_Nom_Agg,plaus('1.000')).
link(4,6,G_Nom_Agg,plaus('1.000')).
link(16,17,G_Nom_Agg,plaus('1.000')).
link(25,26,G_Nom_Agg,plaus('1.000')).
```

7.5. Ambiguous ICD

```
link(6,7,G_Agg_Prep,plaus('0.500')).
link(4,7,G_Nom_Prep,plaus('0.500')).
link(17,18,G_Agg_Prep,plaus('0.500')).
link(16,18,G_Nom_Prep,plaus('0.500')).
link(20,22,G_Ver_Avv,plaus('0.500')).
link(24,22,G_Ver_Avv,plaus('0.500')).
link(25,27,G_Nom_Prep,plaus('0.500')).
link(26,27,G_Agg_Prep,plaus('0.500')).
```