Summary of the Ph.D. Thesis

Automatic Syntactic Parsing of the Hungarian Language Appliyng Rule-based Machine Learning Methods

András Hócza

Advisor: *Tibor Gyimóthy, PhD*

Doctoral School in Mathematics and Computer Science of the Faculty of Science of the University of Szeged

Szeged, 2008

Introduction

The booklet summarizes the scientific results of the author of the Ph.D. dissertation entitled „*A magyar nyelv automatikus szintaktikai elemzése szabályalapú gépi tanulási technikák alkalmazásával*”. The dissertation concentrates on syntactic parsing of Hungarian texts. The author applied rule-based machine learning methods using an annotated corpus to build models for syntactic parsers. The rule-based approach stores the collected information in readable format for human readers, and allow experts to extend the syntactic database with their knowledge.

Syntactic parsing of natural languages

Describing a phenomenon of natural languages is a great challenge for computational linguistics, especially the Hungarian language, that is customarily defined as an agglutinative, free word order language with a rich morphology. These properties make a full analysis of it difficult, compared to Indo-European languages. The grammar formalisms are meta-languages that are made to describe rules and features of natural languages. We can specify the following constraints for these formalisms:

- ***Linguistic suitability***: a measure for meta-languages to characterize their describing ability according to the principles formulated by linguists.
- ***Computational effectiveness***: a measure for meta-languages to characterize time and memory complexity of a realized solution.

The generative grammars seemed promising possibilities at the time of their introduction, because these grammars can be parsed with effective algorithms, especially in case of applying regular and context free grammars. However counterexamples appeared soon showing that this grammar classes are not suitable to describe certain phenomenon of natural languages. Such counterexample the self-embed structures, that can not be described by regular grammars, and the description of cross-dependencies can not be solved by context free grammars.

Today the generative approach have been changed such a linguistic theories and formalisms which concentrate on more precise description of a natural language phenomenon instead of generating languages. If we apply context free grammars to handle agreement and subcategorization we have to insert a lot of new rule and at the end of this process we get a very big grammar. Similar problem the handling of free word order. Describing dependency, especially far dependency, which can not be solved by context free grammars, because the rules of these grammars can cover only neighboring words. If we want to choose the best of ambiguous syntactic structures we have to extend rules with probability the values and estimate probability of syntax trees. Finally, ignoring lexical and structural dependencies during the automatic parsing can results such an unused or unlikely structures which we can exclude according to sense of text.

With tree patterns introduced by the author we can recognize complex syntactic structures with help of description given by regular expressions. Let us suppose we have a tree (Fig. 1) and de description of words and syntactic labels includes cases (e.g. nominative, accusative).

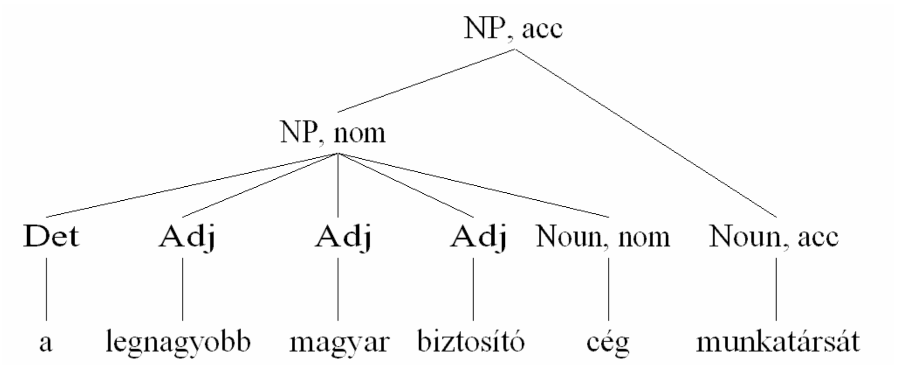

Figure 1. A complex noun phrase

We can execute a lot of transformation with the word groups covered by this tree, we can erase, insert, reorder and exchange words. In this process we get similar word groups and we can find out which word position can be variable without changing the structure of the tree. Other similar cases can be the following:

a$_{\{\text{Det}\}}$ legnagyobb$_{\{\text{Adj}\}}$ biztosító$_{\{\text{Adj}\}}$ cég$_{\{\text{Noun,nom}\}}$ munkatársát$_{\{\text{Noun,acc}\}}$

a$_{\{\text{Det}\}}$ 2$_{\{\text{Num}\}}$ legnagyobb$_{\{\text{Adj}\}}$ biztosító$_{\{\text{Adj}\}}$ cég$_{\{\text{Noun,nom}\}}$ munkatársát$_{\{\text{Noun,acc}\}}$

a$_{\{\text{Det}\}}$ 2$_{\{\text{Num}\}}$ cég$_{\{\text{Noun,nom}\}}$ munkatársát$_{\{\text{Noun,acc}\}}$

az$_{\{\text{Det}\}}$ első$_{\{\text{Num}\}}$ 2$_{\{\text{Num}\}}$ cég$_{\{\text{Noun,nom}\}}$ munkatársát$_{\{\text{Noun,acc}\}}$

We can cover the enumerated cases with one tree pattern (Fig. 2), which moreover generalize cases covering each not enumerated similar word groups.

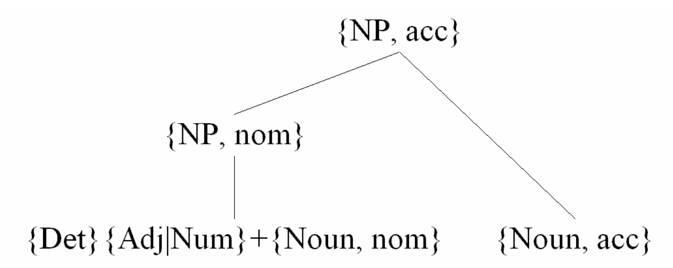

Figure 2. A tree pattern that covers similar structures

If we apply complex structures in syntactic parsing, this model will expectedly contains more items (rules) than a context free which grammar produced with the same corpus. The tree pattern description compensates this growing with containing similar structures in one pattern. The flexibility of description allow building in elements of other formalisms, so various phenomenon of natural languages can be handled. We apply a modified version of ***chart parser*** ([Kaplan73], [Kay86]) to syntactic parsing of text with tree patterns.

Applying machine learning to create grammar

An efficient solution for natural language problems might be the application of machine learning methods, but it requires a large number of training and test examples of annotated phrases. If we have an annotated corpus we can collect the examples of a certain phenomenon. The set of examples contain (x_i, y_i) pairs, where x_i is the description of an object or event and y_i is the category or decision. In a ***classification*** problems the examples have discrete y_i values. It is a ***supervised training*** if the y_i values are known for each x_i (e.g. it can be extracted from an annotated corpus), and the task of machine learning to seek a suitable f function, where $f(x_i) = y_i$. In this case we suppose that the f function will be suitable to determine y values for unseen cases, this principle is called ***inductive learning***. When the classification problem has two classes (true or false) we call it ***concept learning***, in this case we have positive or negative examples depending on the decision of examples true or false.

The author developed the ***RGLearn*** pattern learning algorithm. The input of algorithm is a set of positive and negative examples collected from annotated corpus depend on the example has proper or improper coverage. The output of algorithm is a set of generalized patterns which collective precision is maximized, consequently it covers the most positive and the least negative examples. This algorithm was applied to learn disambiguation model for a rule-based POS-tagger [Kuba04], and to learn syntactic tree patterns ([Hócza04a], [Hócza06a]).

The initial step of specialization generates all possible new tree patterns by extending generalized tree patterns with exactly one attribute from the covered positive examples. The next steps of specialization extend the set of tree patterns with all possible new tree patterns by a combination of each pair of tree patterns. The combination of two tree patterns means the union of their lexical attributes. To avoid the exponential growth of a tree pattern set weak tree patterns are excluded by applying error statistics on positive and negative examples. The following score of a given tree pattern is used as the target for maximization:

$$score = \lambda_1 * (pos-neg) / pos + \lambda_2 * (pos-neg) / (pos+neg)$$

where *pos* is the number of covered positive examples, *neg* is the number of covered negative examples and $\lambda_1 + \lambda_2 = 1$.

There are other possibilities of applying machine learning methods in building models for syntactic parsing. The POS tagger used for disambiguation of Part-of-Speech can be applied for predicting boundaries of word groups as well. For example the task of predicting boundaries of noun phrases (NP) can be defined the following way: classify word positions with 5 label using the information of their context. These labels are: begin of NP (B), inside of NP (I), end of NP (E), one-word NP (BE), outside of NP (O). This is a tagging problem and can be solved by HMM-tagger [Charniak93], or it is a classifying problem and we can apply a supervised machine learning algorithm (e.g. C4.5 [Quinlan93]), or we can improve results of individual methods by using weighted system combination, and the weights of voters are optimized on annotated corpus. We can utilize a word group boundary prediction method in various ***shallow parsing*** tasks, for example segmentation of sentence to smaller parts, or recognizing basic phrases (base-NP, top-NP).

In order to create a probability model we can assign probabilities to patterns. If we have annotated corpus, we can estimate a pattern probability from its normalized occurrence. I was proved [Prescher03], that this way gives the ***maximum likelihood*** estimation of the given annotated corpus. Without annotated corpus we can use the ***Inside-Outside algorithm*** [Baker79] for probability estimation.

The content of pattern set can be different if we introduce parameters for the complex process of model making. We can search the best parameter settings evaluating models in a smaller part of the annotated corpus as well. An applicable optimization algorithm for this task the ***simulated annealing*** [Aarts89]. We can improve our results with system combination of different methods. If we use weights for these methods we can also optimize setting on the annotated corpus.

Tree pattern based complex syntactic parsing method

The author developed an automatic syntactic parsing method collecting the solutions of problem parts in a complex parameter-driven system.

The model building start with collecting syntactic structures examples from the annotated corpus with help of tree shape types. Typical tree shape types are the ***self-embed*** and ***string*** structures that give constraints for properties of syntactic structures collected from annotated corpus. The conditions of tree shape types compose a coherent logical system and these types drive the part tree collection process, and we can disassemble any syntax tree with help of them. We can see a short example of processing a syntax tree in Fig. 3.

Example sentence:

[$_{\text{CP}}$ [$_{\text{NP}}$ [$_{\text{NP}}$ Mihály$_{\text{Noun}}$] és$_{\text{Conj}}$ [$_{\text{NP}}$ az$_{\text{Det}}$ ügyvéd$_{\text{Noun}}$]] [$_{\text{VP}}$ felkereste$_{\text{Verb}}$]
[$_{\text{NP}}$ a$_{\text{Det}}$ [$_{\text{ADJP}}$ budapesti$_{\text{Adj}}$] egyesület$_{\text{Noun}}$] elnökét$_{\text{Noun}}$] .$_{\text{Punct}}$]

Extracted tree parts:

string: [$_{\text{NP}}$ [$_{\text{NP}}$ Mihály$_{\text{Noun}}$] és$_{\text{Conj}}$ [$_{\text{NP}}$ az$_{\text{Det}}$ ügyvéd$_{\text{Noun}}$]]
self-embed: [$_{\text{VP}}$ felkereste$_{\text{Verb}}$]
self-embed: [$_{\text{NP}}$ [$_{\text{NP}}$ a$_{\text{Det}}$ [$_{\text{ADJP}}$ budapesti$_{\text{Adj}}$] egyesület$_{\text{Noun}}$] elnökét$_{\text{Noun}}$]

The structure of the sentence after substituting extracted tree parts:

[$_{\text{CP}}$ NP VP NP .$_{\text{Punct}}$]

Extracted tree parts:

self-embed: [$_{\text{CP}}$ NP VP NP .$_{\text{Punct}}$]

Figure 3. The process of extracting tree parts with help of tree shape types.

Since we can collect a huge set of tree parts from entire corpus, it may cause a serious technical problem processing these examples together, therefore the examples are grouped together according to their most general forms. The tree pattern learning is performed with the RGLearn algorithm group by group.

The learned set of tree patterns is used by a modified version of ***chart parser*** ([Kaplan73], [Kay86]). This mean only some small changes in the original algorithm. We are using bottom-up strategy during parsing. In the parse tree the part trees that are recognized with tree patterns are individual object, therefore we do not fit other tree patterns to the inside node of part trees, we only use the root (Fig. 4). This concept cutes down the running time of syntactic parsing.

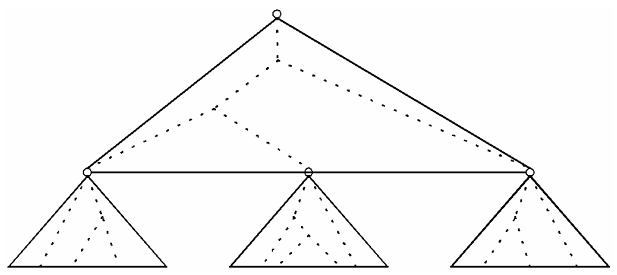

Figure 4. A bottom-up building method of parsing with tree patterns.

The performance of syntactic parsing is measured with three scores known as PARSEVAL metrics [Black91]. The ***precision*** is a percentage of detected phrases that are correct, and the ***recall*** is a percentage of phrases in the data that is found via the parser. The weighted harmonic mean of precision and recall, the traditional ***F-measure*** or balanced F-score, that it has generally $F_{\beta=1}$ setting and it is computed from precision and recall with the same weight. These scores express the goodness of parse tree instead of its error.

The application of an evaluation method allow us to feedback our experiences and improving results. In order to make optimization we added parameters to our complex tree pattern learning methods. We made a frame procedure from the simulated annealing algorithm and we optimize the parameters of model generating on F-measure of parse trees.

Syntactic parsing of Hungarian texts

The author prepared and evaluated various type of syntactic parsers on Hungarian texts using annotated texts of Szeged Treebank [Csendes05], and applied his syntactic parsers in natural language tasks like Information Extraction and Machine Translation.

Hungarian is customarily defined as an agglutinative, free word order language with a rich morphology. These properties make its full analysis difficult compared to Indo-European languages. Unambiguous marks for the automatic recognition of phrase boundaries do not exist. For example the right bound of noun phrases could be the nouns as a head, but there is a possibility of replacement of noun phrase heads with its modifiers:

[$_{NP}$ Péter] [$_{NP}$ a régi könyvet] olvassa , [$_{NP}$ Mari] pedig **[**$_{NP}$ az **újat]** .

Determining the left bound of noun phrases is harder than the head, because it could be a determinant element. However, in certain cases the determinant can be omitted:

[$_{\text{NP}}$ *Péter*] [$_{\text{NP}}$ ~~*(egy)*~~ *könyvet*] *olvas* .

[$_{\text{NP}}$ *Péter*] *olvassa* [$_{\text{NP}}$ *a könyvet*] .

Another problem is the high morphological and syntactic diversity of the Hungarian language. Many words with same stem have up to 100 word forms. The (almost) free word order significantly raises the number of possible patterns and schemas, and this decreases the effectiveness of statistical machine learning methods applied. Especially the realization of inflections is a problem because the linguistic information that is stored in word order in English are expressed with endings in Hungarian.

In order to perform well and learn from the various *Natural Language Processing* (NLP) tasks and achieve a sufficient standard of *Information Extraction* (IE), an adequately large corpus had to be collected to serve as the training database. A relatively large corpus of Hungarian texts of various types was collected, and later called the Szeged Treebank [Csendes05]. It has six topic areas of roughly 200 thousand words each, meaning a text database of some 1.2 million words. The treebank contains about 82,000 POS-tagged and full syntactic parsed sentences. The Hungarian version of the internationally acknowledged MSD (Morpho-Syntactic Description) schema [Erjavec97] was used for the encoding of the words. The MSD encoding schema can store morphological information about part-of-speech determined attributes on up to 17 positions. About 1800 different MSD labels are employed in the annotated corpus. The syntactic tag-set used in the corpus has a correlation with many other internationally accepted syntactic tag-sets.

The first version of Szeged Treebank allowed us to build models for NP recognition parsers. NP recognition is the process of determining whether sequences of words can be grouped together with nouns, and as a part of the field of *Shallow Parsing* is rich enough to support a number of large-scale natural language processing applications including *Information Extraction*, *Information Retrieval*, *Text Summarisation*, and a variety of text-mining operations. The author developed a shallow parser [Hócza04a] and it was applied and evaluated on general texts and short business news.

In the shallow parsing version of complex method we are concentrating on easier problems (e.g. *base-NP* or *top-NP chunking*), applying heuristics to raise the efficiency and speed of the parser algorithm. For example if we analyze the results of NP boundary tagger, a sentence can be segmented to smaller parts, and we have to only recognize tree structures inside the boundaries.

The author apply his shallow parsing method in an *Information Extraction* (IE) system [Hócza03b]. The IE system that was made by the author and his colleagues connects their results of various NLP tasks that had been developed as a toolchain. The input of this IE system is a plane text and the output is a structured database containing the extracted information. During the IE process the syntactic and semantic features of a sentence are determined with a pipeline of NLP modules, and this consist of tokenization, sentence segmentation, morpho-syntactic analysis, part-of-speech tagging, shallow syntactic parsing, recognizing semantic frames, storing extracted information in a structured database. The IE process was applied on short business news taken from Szeged Treebank.

In many aspects the full syntactic parsing is a harder task than shallow parsing. There are more syntactic labels and syntactic structures are more deeper and complex than structures of shallow parsing. There are additional problems in full syntactic parsing like the VP of Hungarian language. Due to free word order the components of a verb group can be rearranged to a lot of order, and in addition the sentence part of a verb group is not always continuous. Therefore it is not possible to describe this phenomenon of Hungarian language with context free rules.

The author developed a full tree pattern based syntactic parser [Hócza06a] evaluated his method on general texts and short business news taken from Szeged Treebank 2.0. The author and his colleagues effectively improve the recognition accuracy of the syntactic parser using the Boosting algorithm [Hócza05a]. In [Hócza05b] author and his colleagues reported their effort in making a database from Szeged Treebank 2.0 to evaluate syntactic parsers in this domain, and they proposed using this database to compare Hungarian syntactic parsers that had been made so far and new parsers in the future.

Machine Translation (MT) is the application of computers to the translation of texts from one natural language to another. The practical reason for attempting this is that in many fields people have to read documents and have to communicate in languages they do not know and a good quality MT system could provide a quick solution for this problem. Today's state of the art in MT has been defined by *Statistical Machine Translation* (SMT) systems. The author extended an existing syntax-driven SMT system, the GenPar, building in the Hungarian-English as a new machine translation language pair [Hócza06b]. In order to examine effects of various preprocessing steps (POS-tagging, lemmatization and syntactic parsing) on system performance more prototypes had been made. The manually POS-tagged and syntactically parsed Hungarian and English texts needed for preprocessing was derived from the Szeged Treebank 2.0. The author used his tree pattern based method to parse Hungarian sentences, and the preprocessor modules for English texts was given in the GenPar original prototypes. Parallel sentences were selected from the Hunglish Corpus [Varga05], a sentence-aligned Hungarian-English parallel corpus of about 54.2 m words in 2.07 m sentence pairs. The corpus was manually collected in eight topic areas and was aligned with automatic methods. The evaluation was performed on 5k training and 500 test sentence pairs selected from the Hunglish Corpus.

Results

The present thesis summarizes the results obtained by the author in the past couple of years. The results can be separated into two different groups, we can read about theoretical constructions and practical applications. The theoretical results are the following:

I/1. The author developed a new formalism named tree patterns [Hócza04a], that identify larger syntactic structures inside a sentence. With a single tree pattern we can describe several similar structures as a variation of a syntactic object. This formalism give us a powerful tool to parsing inflective and free word order languages like Hungarian.

I/2. The author implemented the RGLearn general pattern learning algorithm [Hócza04a]. The algorithm searches the optimal pattern between generalized and specialized form, for example in case of tree patterns the algorithm looks for the set of tree patterns that achieve the best result evaluating it on a test corpus.

I/3. The author implemented a tree pattern based chart parser that can be applied with bottom-up building strategy [Hócza04a].

I/4. The author worked out the complex method of tree pattern based syntactic parsing building together the individual modules: extracting syntactic structures from annotated corpus, learning tree patterns, parsing with tree patterns, evaluating results of the parser, feedback the information of performance to optimize the model [Hócza04a].

The practical applications from the second group of results, these are:

II/1. The author implemented a rule-based POS-tagger applying the RGLearn algorithm to learn context sensitive patterns for disambiguation. This method was compared with other methods made by colleagues of the author [Kuba04].

II/2. The author applied his complex tree pattern based method to learn and recognize noun phrases of Hungarian texts significantly outperforming previous results [Hócza04a].

II/3. The shallow parser for noun phrase recognition was built in an information extraction toolchain made by the author and his colleagues [Hócza03b]. This IE system was applied on Hungarian short business news.

II/4. The complex tree pattern based method was applied on full syntactic parsing of Hungarian texts [Hócza05b], [Hócza06a].

II/5. The results of full syntactic parsing method was improved by the author and his colleagues using the Boosting algorithm [Hócza05a].

II/6. The full syntactic parser was built in a statistical machine translation system named GenPar as a part of a new Hungarian-English extension [Hócza06b].

References

[Aarts89] E. H. L. Aarts, E., Korst, J. (1989): Simulated Annealing and Boltzmann Machines, *John Wiley & Sons,* New York

[Baker79] Baker, James K. (1979): Trainable grammars for speech recognition, in *Proceedings of the Spring Conference of the Acoustical Society of America*, pp. 547–550.

[Black91] E. Black, S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini and T. Strzalkowski (1991): A procedure for quantitatively comparing the syntactic coverage of English grammars, in *Proceedings of the DARPA Speech and Natural Language Workshop*, pp. 306-311.

[Charniak93] Charniak, E (1993): Statistical Language Learning, *MIT Press*, Cambridge, Massachusetts

[Csendes05] Csendes, D., Csirik, J., Gyimóthy, T., Kocsor, A. (2005): The Szeged Treebank, in *Proceedings of the 8th International Conference on Text, Speech and Dialogue, TSD 2005,* Karlovy Vary, pp. 123-131

[Erjavec97] Erjavec, T. and Monachini, M., ed. (1997): Specification and Notation for Lexicon Encoding, *Copernicus project 106 "MULTEXT-EAST"*, Work Package WP1 – Task 1.1 Deliverable D1.1F.

[Hócza03b] Hócza, A., Alexin, Z., Csendes, D., Csirik, J., Gyimóthy, T. (2003): Application of ILP methods in different natural language processing phases for information extraction from Hungarian texts, in *Proceedings of the Kalmár Workshop on Logic and Computer Science,* Szeged, pp. 107-116.

[Hócza04a] Hócza, A. (2004): Noun Phrase Recognition with Tree Patterns, in *Acta Cybernetica*, Szeged, Volume 16, Issue 4, pp. 611-623

[Hócza05a] Hócza, A., Felföldi, L., Kocsor, A. (2005): Learning Syntactic Patterns Using Boosting and Other Classifier Combination Schemas, in *V. Matousek et al. (Eds.): Proceedings of the 8th International Conference on Text, Speech and Dialogue, TSD 2005*, TSD 2005, Karlovy Vary, Czech Republic, LNAI 3658, pp. 69-76

[Hócza05b] Hócza, A., Kovács, K., Kocsor, A. (2005): Szintaktikai elemzők eredményeinek összehasonlítása, *MSZNY 2005 konferenciakiadványa*, Szeged, 277-284 oldal

[Hócza06a] Hócza, A. (2006): Learning Tree Patterns for Syntactic Parsing, in *Acta Cybernetica*, Szeged, Volume 17, Issue 3, pp. 647 - 659

[Hócza06b] Hócza, A., Kocsor, A. (2006): Hungarian-English machine translation using GenPar, in *Proceedings of the 9th International Conference on Text, Speech and Dialogue, TSD 2006,* Brno, Czech Republic, September 11-15, pp. 87-94

[Kuba04] Kuba, A., Hócza, A., and Csirik, J. (2004): POS Tagging of Hungarian with Combined Statistical and Rule-Based Methods, in *Proceedings of the 7th International Conference on Text, Speech and Dialogue TSD 2004,* Brno, Czech Republic, September 8-11, pp. 113-120

[Kaplan73] R. M. Kaplan (1973). A general syntactic processor. In Rustin, R. (Ed.), *Natural Language Processing*, pp. 193-241. Algorithmics Press, New York.

[Kay86] Martin Kay. (1986). Algorithm schemata and data structures in syntactic processing. In Readings in natural language processing, pp. 35-70. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Prescher03] Prescher, D. (2003): A Tutorial on the Expectation-Maximization Algorithm Including Maximum-Likelihood Estimation and EM Training of Probabilistic Context-Free Grammars, *Presented at the 15th European Summer School in Logic, Language, and Information (ESSLLI 2003).*

[Quinlan93] Quinlan, J. R. (1993): C4.5: Programs for Machine Learning, *Morgan Kaufmann Publisher*.