# Approaching the Chinese Word Segmentation Problem with CHR Grammars

Henning Christiansen and Bo Li

Research group PLIS: Programming, Logic and Intelligent Systems
Department of Communication, Business and Information Technologies
Roskilde University, P.O.Box 260, DK-4000 Roskilde, Denmark
E-mail: {henning, bol}@ruc.dk

**Abstract.** Written Chinese text does not include separators between words, as do European languages using space characters, and this creates the Chinese Word Segmentation Problem: given a text in Chinese, divide it in a correct way into segments corresponding to words. Correctness means how a competent Chinese language user would do this. CHR Grammars (CHRG) is an implemented grammar system that allows highly flexible bottom-up analyses using rule-based constraint solving techniques. We demonstrate how different approaches to the problem can be expressed in CHRG in a highly concise way, and how different principles can complement each other in this paradigm. We do not claim to have provided any improvement with methods currently in use, our aims are a) to forward a way for further experimentation with solutions to the problem, and b) to show how CHRG gives rise to succinct and executable specifications of such methods. We present here some preliminary and promising experiments tested on simple examples.

## 1 Introduction

Chinese text is written without explicit separation between the different words and it is up to the reader to make this separation; however, periods are unambiguously delineated using the special character "。" which serves no other purpose. The Chinese language presents the same collection of problems as any other language with respect to automatic analysis, including syntactic (lexical, structural) and semantic ambiguities, unknown words, etc. Compared with European languages, written Chinese exposes further problematic issues, most notably the lack of word separators which is further emphasized by that fact that most single characters, seen in isolation, may form a word, and quite many pairs of two characters also form words. As in other languages, analysis is also made difficult by the fact that certain parts may be left out of a sentence; as opposed to most European languages, even the verb may be left out when obvious from the context, though mostly verbs corresponding to "have" or "be". Also, Chinese has almost no inflectional markers. These facts make it even more difficult to use syntactic constraints or cues to guide or validate a given segmentation.

To appear in *Proceedings of the 6th International Workshop on Constraints and Language Processing*
A workshop at CONTEXT '11: The 7th International and Interdisciplinary Conference on Modeling and Using Context 2011. Karlsruhe, Germany, September 27, 2011

The recent textbook [1] contains a good introduction to these difficulties also for non-Chinese speakers.

Good solutions to the Chinese Word Segmentation Problem, henceforth referred to as CWSP, are evidently in demand for different kinds of digital processing of Chinese text, ranging from simple type-setting (breaking text into lines), over web search engines (text indexing and query processing), to (front-ends for) different sorts of deep analysis. As Chinese is one of the most used languages on the Internet,[1] the problem has attracted much research attention; proceedings of the CIPS-SIGHAN Joint Conferences on Chinese Language Processing since 2002 and previous workshops provide a good overview of the history and state of the art of such methods [2]. Lexicon-based approaches, complemented by different heuristics and statistical-based methods are dominating; see, e.g., [1, 3] for overview. Controlled competitions between different Chinese word segmentation systems have been arranged together with the CIPS-SIGHAN conferences. The report from the 2010 competition [4] shows precision and recall figures up to around 0.95 for tests on selected corpora. This seems quite impressive and indicate only little room for improvement; on the other hand it is not obvious that the reported results will hold on arbitrary unseen (types of) text.

Some general systems for Internet search such as Google[2] and Baidu[3] use their own word segmentation algorithms which are not publicly available; [3] provides some tests and discussion of these approaches.

CHR Grammars [5] is a grammar formalism and implemented system that adds a grammar notation layer on top of the programming language of Constraint Handling Rules [6, 7], CHR, analogous to the way Definite Clause Grammars [8] are added on top of Prolog. CHR itself was introduced in early 1990es as a rule-based, logical programming language for writing constraint solvers for traditional constraint domains such as integer or real numbers in a declarative way, but has turned out to be a quite general and versatile forward-chaining reasoner suited for a variety of applications, including language processing through CHRG. CHR, often in the shape of CHRG, have been used for a variety of language processing tasks until now, but to our knowledge, not to Chinese before; see, e.g., [9–15]. In this paper we present the first, early experiments in approaching CWSP using CHRG, and our experience so far is very promising in that different principles for (partly) solving CWSP can be expressed in very elegant ways. This should be seen in contrast to the difficulties and comprehensive amount of detailed programming that would be necessary if similar experiments and prototype implementations were made using a traditional programming language such as Java or C.

In the following, we start giving a brief overview of CHRG in section 2. Section 3 shows how a lexicon can be represented in a CHRG and demonstrates it for

[1] Chinese is currently the secondly most used language on the Internet after English, and, based on the current growth rate, it may very well be the most used in a few years; consult, e.g., http://www.internetworldstats.com/stats7.htm.

[2] http://www.google.com

[3] http://www.baidu.com; the biggest Chinese web search engine.

a rudimentary CWSP method based on the so-called maximum match principle, and 4 shows how two simple CHRG rules can split a text into smaller portions, called maximum ambiguous segments, which can be analyzed separately. In section 5, we discuss further principles that we would like to experiment with in CHRG, and section 6 gives a short summary and a conclusion.

## 2 Brief Overview of CHR Grammars

We assume the terminology and basic concepts of CHR and Prolog to be known, but the following introduction of CHR Grammars may also provide sufficient insight to readers with a broader background. These grammars are implemented as an additional layer of syntax on top of a Prolog-based CHR system and are automatically compiled into CHR rules when loaded. The CHRG system and a comprehensive Users' Guide are available [16].

The basic idea is that grammar symbols (terminals and nonterminals) are represented as constraints, decorated with integer numbers, that refer to positions in the text and in this way maintain the usual sequential order. Rules work bottom up: when certain patterns of grammar symbols are observed in the store, a given rule may apply and add new grammar symbols. Consider the following example of a grammar given as its full source text.

```
:- chrg_symbols noun/0, verb/0, sentence/0.
[dogs] ::> noun.
[cats] ::> noun.
[hate] ::> verb.
noun, verb, noun ::> sentence.
end_of_CHRG_source.
```

Given the query

```
?- parse([dogs,hate,cats])
```

constraints corresponding to the three terminal symbols will be entered; the three "lexical" rules will apply and add new grammar symbols representing the recognition of two `noun`s and a `verb` in a suitable order such that the last rule can apply and report the recognition of a `sentence`. The answer is given as the final constraint store which will include the following constraint; notice that the positions (or boundaries) in the string, that were invisible in the grammar, are shown.

```
sentence(0,3).
```

The rules shown above are *propagation* rules, which add new grammar symbols to the existing ones; when the arrow in a rule is replaced by `<:>`, the rule becomes a *simplification* rule which will remove the symbols matched on the letfhand side; there is also a form of rules, called *simpagations*, that allow to remove only some of the matched symbols which will be shown below.

Notice that when simplification rules are used, the actual result of a parsing process may depend on the procedural semantics of the underlying CHR system (which rules are applied when), and a knowledge about this is needed for those who want to exploit the full power of CHRGs. However, these properties imply a natural handling of ambiguity: when propagation rules are used, all possible analysis are generated in the same constraint store, while simplification rules may be applied for pruning or sorting out among different solutions.

In some cases, it may be relevant to order the application of rules into phases such that firstly all rules of one kind apply as much as possible, and then a next sort of rules is allowed to apply. This can be done by embedding non-grammatical constraints in the head of grammar rule, declared as ordinary CHR constraints. We can illustrate this principle by a modification of the sample grammar above.

```
...
:- chr_constraint phase2/0.
{phase2}, noun, verb, noun ::> sentence.
```

Notice the special syntax with curly brackets, which is inspired by Definite Clause Grammars [8]. This means that, in this rule, the constraint `phase2` does not depend on positions in the text, but must be present for the rule to apply. The query for analysis should then be changed as follows.

```
?- parse([dogs,hate,cats]), phase2.
```

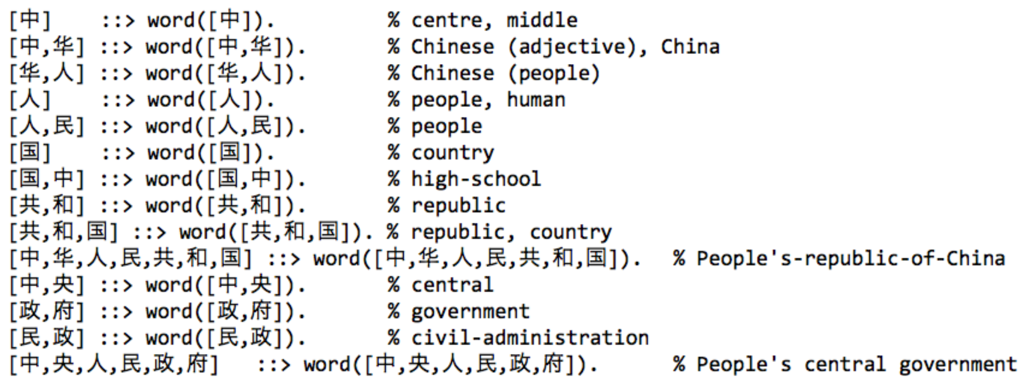
This means that first, the lexical rules will apply as long as possible (as they are not conditioned by the constraint `phase2`), and when they are finished, the `sentence` rule is allow to be tried. In this particular example, this technique will in fact not change the result, but we give examples below where it is essential.

CHRG rules allow an extensive collection of patterns on the lefthand side for how grammar symbols can be matched in the store: context-sensive matching, parallel matching, gaps, etc.; these facilities will be explained below when they are used in our examples. As in a Definite Clause Grammar [8], grammar symbols may be extended with additional arguments that may store arbitrary information of syntactic and semantic kinds.

CHRG runs under SICStus Prolog 4 [17] and SWI Prolog [18], which are both capable of handling UNICODE characters, so Chinese characters and text can be represented directly.

## 3 Representing Lexicon in a CHR Grammar; Lexicon-Based Methods

The simplest way to represent a lexicon in a CHRG is a by sequence of short rules as those we indicated as lexical rules above. For our first experiments with CWSP, we represent a lexicon classifying words only. The following example rules provide the lexicon for examples to follow.



```
[中]    ::> word([中]).         % centre, middle
[中,华] ::> word([中,华]).      % Chinese (adjective), China
[华,人] ::> word([华,人]).      % Chinese (people)
[人]    ::> word([人]).         % people, human
[人,民] ::> word([人,民]).      % people
[国]    ::> word([国]).         % country
[国,中] ::> word([国,中]).      % high-school
[共,和] ::> word([共,和]).      % republic
[共,和,国] ::> word([共,和,国]). % republic, country
[中,华,人,民,共,和,国] ::> word([中,华,人,民,共,和,国]).   % People's-republic-of-China
[中,央] ::> word([中,央]).      % central
[政,府] ::> word([政,府]).      % government
[民,政] ::> word([民,政]).      % civil-administration
[中,央,人,民,政,府]   ::> word([中,央,人,民,政,府]).     % People's central government
```

Notice that the grammar contains two rather large words that look like compounds, but which will be included in any dictionary as words as they are known and fixed terms with fixed meanings.

The `word` grammar symbol may of course be extended with syntactic tags, but for now we will do with the simplest form as shown.

## A simple Lexicon-Based Method for CWSP, Maximum Matching

A first naive idea for CWSP may be to generate all possible words from the input, followed by an assembly of all possible segmentations that happens to include the entire input, and then a final phase selecting a best segmentation according to some criteria. Obviously, this is of exponential or worse computational complexity and thus more efficient heuristics have been developed. One such heuristics is the maximum matching method, which has been used in both forward and backward versions; here we show the forward method (see [1] for background). The sentence is scanned from left to right, always picking the longest possible word; then the process continues this way until the entire string has been processed.[4]

Three CHRG rules are sufficient to implement this principle. The first one, which needs some explanation, will remove occurrences of words that are proper prefixes of other word occurrences.

```
!word(_) $$ word(_), ... <:> true.
```

The “`$$`” operator is CHRG’s notation for parallel match: the rule applies whenever both of the indicated patterns match grammatical constraints in the store for the same range of positions (i.e., substring). The symbol “`...`” refers to a *gap* that may match any number of positions in the string, from zero and upwards, independently of whatever grammar symbols might be assiciated with those positions.[5] In other words, the pattern “`word(_), ...`” matches any substring that starts with a word. So when this is matched in parallel with a single

[4] In theory, this may fail to capture the entire string, but as most single characters can represent a word, this will be extremely rare.

[5] In fact, gaps themselves are not implemented by matching, but affect how its neighbouring grammar symbols are matched, putting restrictions on their word bound-

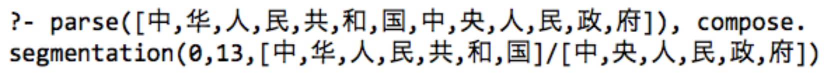
word, it can apply in exactly those cases where two words occur, one being a (not necessarily proper) prefix of the other. Finally, the exclamation mark in front of the first `word` indicates, in a rule having the `<:>` arrow (which otherwise signifies simplification), a simpagation rule. The meaning is that here only grammar symbols and constraints marked with “`!`” are kept in the store. The `true` on the righthand side stands for nothing, meaning that no new constraints or grammar symbols are added.

So when a string is entered, this rule will apply as many times as possible, each time a lexicon rule adds a new word, and thus keeping only longest words.

In a second phase, we compose a segmentation from left to right, starting from the word starting after position 0. The first rule applies an optional notation in “`:(0,_)`”, which makes the word boundaries explicit, here used to indicate that this rule only applies for a leftmost word. The `compose` constraint is used as described above to control that these rules cannot be applied before all short words have been removed by the rule above.

```
{!compose}, word(W):(0,_) <:> segmentation(W).
{!compose}, segmentation(Ws), word(W) <:> segmentation(Ws/W).
```

Assuming the lexicon given above, we can query this program as follows, shown also with the answer found (with constraints removed that are not important for our discussion).



```
?- parse([中,华,人,民,共,和,国,中,央,人,民,政,府]), compose.
segmentation(0,13,[中,华,人,民,共,和,国]/[中,央,人,民,政,府])
```

Here the method actually produces the right segmentation, meaning “The Central People’s Government of the People’s Republic of China”; the “of” being implicit in the Chinese text. Notice that there is actually a word spanning over the split, namely the word for high-school. This example showed also the advantage of combining the maximum match principle with having common terms or idioms represented as entries in the lexicon.

We can show another example that demonstrates how maximum matching easily can go wrong, with a suggestion for a repair. We extend the lexicon with the following rules.

aries. In the example shown, it must hold that $r_1 \geq r_2$ for the rule to apply where $r_1$ and $r_2$ designate the right boundary of the first, resp., the second `word` in the rule head.

```
[明,确]    ::> word([明,确]).    % definitude, clearly
[确,实]    ::> word([确,实]).    % really, actually
[实,在]    ::> word([实,在]).    % honest, actually
[考,虑]    ::> word([考,虑]).    % consider
[将,来]    ::> word([将,来]).    % future
[将,来,的] ::> word([将,来,的]). % future-related
[李]       ::> word([李]).       % Li (family name)
[李,子]    ::> word([李,子]).    % plum
[明]       ::> word([明]).       % bright, Ming (given name)
[将]       ::> word([将]).       % will
[在]       ::> word([在]).       % at
[来]       ::> word([来]).       % come
[事]       ::> word([事]).       % thing
```

The sample sentence we want to check is "李明确实在考虑将来的事", which can be translated into English as "Li Ming is really considering the future things" corresponding to the correct segmentation

```
[李,明]/[确,实]/[在]/[考,虑]/[将,来,的]/[事].
```

Querying the maximum matching program as shown above for this sentence gives the segmentation

```
[李]/[明,确]/[实,在]/[考,虑]/[将,来,的]/[事]
```

that does not give sense to a Chinese reader. The problem is that the first two characters are not seen as a unit, but the first character is taken as a single word and thus the second and third second character are taken as a word, and so on. In the middle of the sentence, the program accidentally gets on the right track again and gets the remaining words right. Due to the high frequency of two-character words in Chinese, it is easy to produce quite long sentences where one wrong step in the beginning makes everything go wrong for the maximum matching method.

If instead, in the example above, the two characters for the personal name Li Ming are treated as one unit, everything would go right. This could suggest that a specialized algorithm for identifying personal names might be useful as an auxiliary for CWSP, as has been suggested among others by [19]. We can simulate such a facility by adding a rule for this specific name as follows.

```
[李,明] ::> word([李,明]).  % Li Ming (person name)
```

Finally, we mention that combinations of forward and backward maximum segmentation have been used, and in those regions where the two disagree, more advanced methods are applied; see, e.g., [20].

## 4 Identifying Maximum Ambiguous Segments

Another principle that may be used in algorithms for CWSP is to run a first phase, identifying the maximum ambiguous segments of a text. We have distilled the principle from a variety of methods that apply similar principles; we have not been able to trace it back to a single source, but [1] may be consulted for an overview and [2] for detailed contributions.

An *ambiguous segment* is defined as a contiguous segment $s$ in which

– any two contiguous characters are part of a word,
– there are at least two words that overlap, and
– the first and last character are each part of a word entirely within $s$.

For example, if *abcd* and *def* are words, then the substring *abcdef* will form an ambiguous segments, but not necessarily *cdef* or *abcdefg*. An ambiguous segment is *maximal*, a MAS, whenever it cannot be expanded in any direction to form a larger ambiguous segment. For example, if *abc*, *cde*, *def*, *defg* are words, then the substring *abcdefg* may form a maximal ambiguous segment.

In other words, if no unknown words occur in a text, the splits between the MASs will be definitive. Except in construed cases, the length of the MASs are reasonable, which means that we can apply more expensive methods subsequently within each MAS, perhaps even with exponential methods that enumerate and evaluate all possible segmentations.

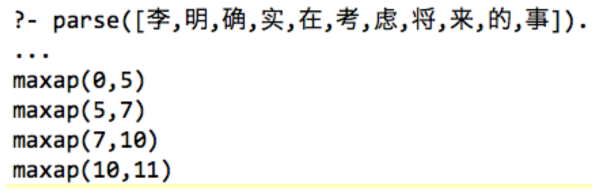
Identifying these MASs can be done by very few CHR rules. For simplicity, we introduce a grammar symbol `maxap` which covers MASs as well as single words that can only be recognized as such. Assuming a lexicon defined as shown above, which identifies any possible word in the text, the following two CHRG rules and an additional Prolog predicate are sufficient to identify the `maxap`s.

```
word(W) ::> maxap.
maxap:R1, ... $$ ..., maxap:R2 <:> overlap(R1,R2) | maxap.

overlap((A1,B1),(A2,B2)):- A1 < B2, A2 < B1.
```

The second rule uses the auxiliary Prolog predicate `overlap` as a guard. The presence of a guard, between the arrow and the vertical bar, means that the rule can only apply in those cases where the guard is true. The `overlap` predicate tests, as its name indicates, whether the two segments in the string occupied by the two input `maxap`s do overlap. This grammar rule will gradually put together ambiguous segments and, via repeated applications, merge together so only maximum ones remain.

We can test this program for the previous example, “Li Ming is really ...” as follows.



```
?- parse([李,明,确,实,在,考,虑,将,来,的,事]).
...
maxap(0,5)
maxap(5,7)
maxap(7,10)
maxap(10,11)
```

This corresponds to splitting the sequence into the substrings

李明确实在，考虑，将来的，事，

which then can be separately analyzed.

## 5 Further extensions

Our main sources on CWSP research [1, 2] report also statistically based methods of different sorts, possibly combining with part-of-speech tagging. While part-of-speech tagging is straightforward to add via the lexicon rules, CHRG is currently not supported by machine-learning techniques to produce useful statistics. However, it is straightforward to integrate probabilities and other weighting schemes in a CHRG: each constituent has an associated weight, and when a rule applies, it calculates a new weight for the compound. Additional rules can be added that prune partial segmentations of low weight. Comprehensive statistics concerning ambiguity phenomena in Chinese text is reported by [21], which appears to be very useful for further research into CWSP.

Furthermore, we can extend the CHRG rules with particular knowledge about the Chinese language. For example, the sign "的" (pronounced "de") normally serves as a marker that converts a preceding noun into an adjective; in fact, most adjectives are constructed in this way from nouns which often have no direct equivalent in European languages, e.g., adjective "red" is constructed from a noun for "red things". Thus, what comes before "的" should preferably be a noun.[6] There are of course lots of such small pieces of knowledge that can be employed and should be employed, and we may hope that the modular rule-based nature of CHR can make it possible to add such principles in an incremental way, one by one.

We did not yet approach the out-of vocabulary (OOV) problem, but one direction to follow is to identify specific patterns in the way that a CHRG-based analyzer attempts to solve the problem with a limited dictionary, including which grammatical constraints that might be broken in this process. OOV words are often proper names and it is obvious that a module for recognizing proper names should be included. We have already referred to [19] that suggests an approach to recognize person names, and [1] lists several characteristics than may be applied in identifying also place names, transcription of foreign names, etc. We may also refer to an interesting approach to OOV in CWSP that incorporate web searches [22]. In [3] a method is suggested that involves web searches to evaluate alternative suggestions for segmentations which also may improve performance in case of OOV.

## 6 Conclusion

We have presented the first steps of experimentations using CHR Grammars to approach the Chinese Word Segmentation Problem. This problem has a high

[6] There are few additional usages of "的" (where it is pronounced "di"), but these are in special words that are expected to be included in any Chinese dictionary.

demand for efficient and precise solutions due to the high presence of the Chinese language on the Internet, as well as for Chinese language processing in general. It is also an interesting test case for the versatility of CHR Grammars. We have demonstrated very concise and succinct contributions to solutions in terms of CHR Grammar rules, which we take as a preliminary evidence for the suitability of CHR Grammars as an experimental platform for the Chinese Word Segmentation Problem.

We do not believe scaling to be a big issue for our approach:

– Periods in Chinese are always delimited in an unambiguous way, which puts an upper limit to the length of the texts that need to be treated as a hole.
– The straightforward lexicon-as-grammar-rules approach that we have applied here, which is perfect for small prototypes, does not scale well to full dictionaries. However, it is easy to get around this problem using an external dictionary, so that a text is entered into the CHR Grammar system as a character sequence (as shown here) together with constraints that represent all possible word occurrences in the text.

Then CHR Grammars' flexibility may be utilized to handle lots of special cases, perhaps ordered into layered phases, as demonstrated in our examples. An important next step is to incorporate methods for handling OOV words.

*Acknowledgements.* This work is supported by the project "Logic-statistic modelling and analysis of biological sequence data" funded by the NABIIT program under the Danish Strategic Research Council.

## References

1. Wong, K.F., Li, W., Xu, R., Zhang, Z.S.: Introduction to Chinese Natural Language Processing. Morgan and Claypool publishers (2010)
2. ACL Anthology: A Digital Archive of Research Papers in Computational Linguistics: Special Interest Group on Chinese Language Processing (SIGHAN) Webarchive with all articles of CIPS-SIGHAN Joint Conference on Chinese Language Processing 2010, Proceedings of the $n$th SIGHAN Workshop on Chinese Language Processing, $n = 1, \ldots, 6$, 2002–2010, Second Chinese Language Processing Workshop. 2000. http://www.aclweb.org/anthology/sighan.html. *Link checked July 2011.*
3. Li, B.: Research on Chinese Word Segmentation and proposals for improvement. Master's thesis, Roskilde University, Computer Science Studies, Roskilde, Denmark (2011)
4. Zhao, H., Liu, Q.: The CIPS-SIGHAN CLP 2010 Chinese Word Segmentation Bakeoff. In: Proceedings of the Joint Conference on Chinese Language Processing, Association for Computational Linguistics (2010) 199–209
5. Christiansen, H.: CHR Grammars. Int'l Journal on Theory and Practice of Logic Programming **5**(4-5) (2005) 467–501
6. Frühwirth, T.W.: Theory and practice of Constraint Handling Rules. Journal of Logic Programming **37**(1-3) (1998) 95–138

7. Frühwirth, T.: Constraint Handling Rules. Cambridge University Press (August 2009)
8. Pereira, F.C.N., Warren, D.H.D.: Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. Artificial Intelligence **13**(3) (1980) 231–278
9. Christiansen, H., Dahl, V.: Logic grammars for diagnosis and repair. International Journal on Artificial Intelligence Tools **12**(3) (2003) 227–248
10. Bavarian, M., Dahl, V.: Constraint based methods for biological sequence analysis. Journal of Universal Computing Science **12**(11) (2006) 1500–1520
11. Dahl, V., Voll, K.D.: Concept formation rules: An executable cognitive model of knowledge construction. In Sharp, B., ed.: NLUCS, INSTICC Press (2004) 28–36
12. Dahl, V., Blache, P.: Extracting selected phrases through constraint satisfaction. In: Constraint Solving and Language Processing; Proceedings of the 2nd International Workshop. Volume 104 of Datalogiske skrifter, Roskilde University. (2005) 3–17
13. Christiansen, H., Have, C.T., Tveitane, K.: From use cases to UML class diagrams using logic grammars and constraints. In: RANLP '07: Proc. Intl. Conf. Recent Adv. Nat. Lang. Processing. (September 2007) 128–132
14. Dahl, V., Gu, B.: A CHRG analysis of ambiguity in biological texts. In: CSLP '07: Proc. 4th Intl. Workshop on Constraints and Language Processing. Volume 113 of Computer Science Research Report. (August 2007) 53–64 Extended Abstract.
15. Hecksher, T., Nielsen, S.T., Pigeon, A.: A CHRG model of the ancient Egyptian grammar. Unpublished student project report, Roskilde University, Denmark (December 2002)
16. Christiansen, H.: CHR Grammar web site; released 2002. http://www.ruc.dk/~henning/chrg (2002)
17. Swedish Institute of Computer Science: SICStus Prolog Website (checked 2012) http://www.sics.se/isl/sicstuswww/site/index.html.
18. SWI Prolog Organization: SWI Prolog Website (checked 2012) http://www.swi-prolog.org/.
19. Chen, Y., Jin, P., Li, W., Huang, C.R.: The Chinese persons name disambiguation evaluation: Exploration of personal name disambiguation in Chinese news. In: CIPS-SIGHAN Joint Conference on Chinese Language Processing 2010. (2010) Online proceedings, http://aclweb.org/anthology/W/W10/W10-4152.pdf.
20. Zhai, F.W., He, F.W., Zuo, W.L.: Chinese word segmentation based on dictionary and statistics. Journal of Chinese Computer Systems, **9**(1) (2009) (No page numbers given)
21. Qiao, W., Sun, M., Menzel, W.: Statistical properties of overlapping ambiguities in Chinese word segmentation and a strategy for their disambiguation. In Sojka, P., Horák, A., Kopecek, I., Pala, K., eds.: TSD. Volume 5246 of Lecture Notes in Computer Science., Springer (2008) 177–186
22. Qiao, W., Sun, M.: Incorporate web search technology to solve out-of-vocabulary words in Chinese word segmentation. In: Proceedings of 11th Pacific Asia Conference on Language, Information and Computation (PACLIC'2009). (2009) 454–463