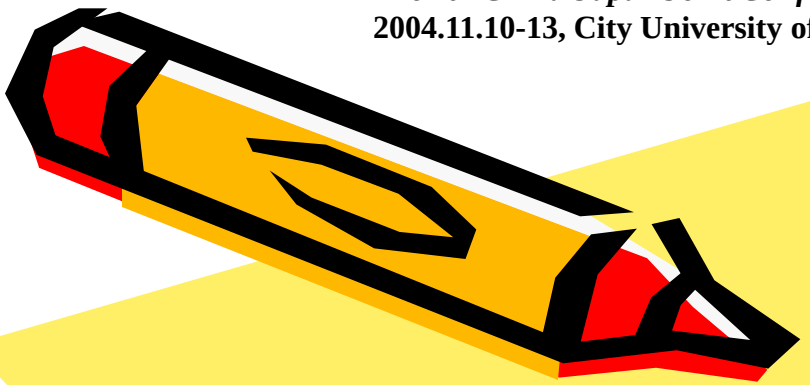
The 4th China-Japan Joint Conference to Promote Cooperation in Natural Language Processing (CJNLP-04), 2004.11.10-13, City University of Hong Kong.



An Integrated Chinese Grammar Development Environment

Zhan Weidong*, Liu Qun, Chang Baobao, Zhang Qinlong, Wu Yonghua

* Dept. of Chinese Language & Literature, Peking University
Beijing, 100871, PR.China

Email: zwd@pku.edu.cn
Personal Homepage: http://ccl.pku.edu.cn/doubtfire/

Outline of Talk

1) Introduction: Why we need ICGDE?

2) What does ICGDE consist of?

3) What is the characteristics of ICGDE?

4) What have been done with ICGDE

5) Future works

1 Introduction

- There are many applications which require a sentence parser as a core or fundamental component. For instance, MT, Human-computer interface, IE, IR, etc..

- Whatever the applications make use of shallow parsing techniques or rely on deep analysis, precise large-coverage grammars, lexicons and annotated corpus of natural languages are always needed to build up.

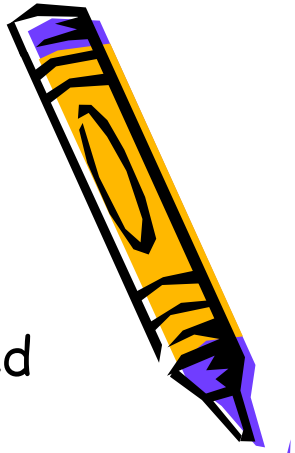
- Developing such so-called linguistic knowledge base(LKB, for short) is a very time-consuming and difficult activity. The following factors are often needed to taken into consideration: coverage, granularity, efficiency, etc..

- We need a LKB development environment that can process various forms of linguistic knowledge effectively and integrate them into an organized system to support applications of NLP and linguistic research and teaching.

Overseas Researches on Development of Linguistic Resource

- INTEX: A Linguistic Development Environment
  （http://www.nyu.edu/pages/linguistics/intex/）
- The LinGO (Linguistic Grammars Online) project
  （http://lingo.stanford.edu）
- Language Resources & Evaluation Conference
  （http://www.lrec-conf.org/）
- Xerox Linguistics Environment Project
  （http://www2.parc.com/istl/groups/nltt/xle/）
- XTAG System
  （http://www.cis.upenn.edu/~xtag/release-8.31.98-html/node12.html）

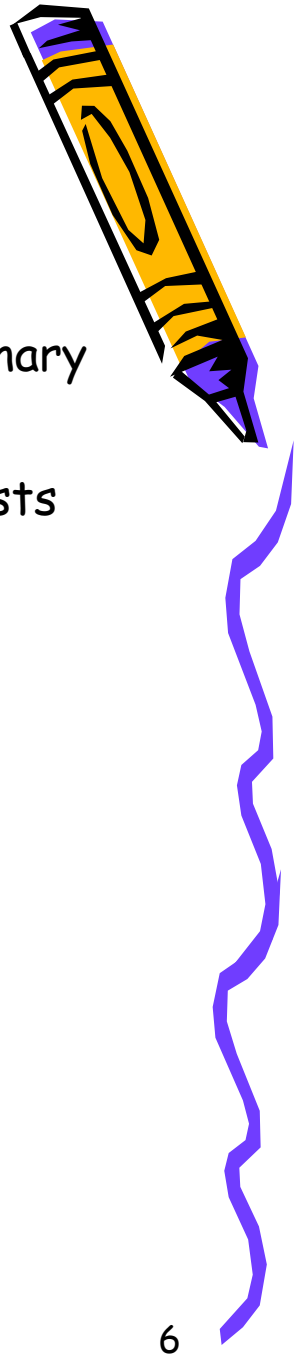
Our Ongoing Projects



Our research work is now supported by several state-funded projects which are listed below.

- 2002.1 -- 2006.12 Construction Rules of Chinese Sentence And Development Environment of Chinese Parsing System, funded by Ministry of Education of China (Project No.200110)
- 2002.1 -- 2004.12 Study on Annotation Specification of Chinese Phrases and Sentence Patterns, funded by Ministry of Education of China (Project No.YB105-49)
- 2002.12 -- 2005.12 Computing Platform for Research on Contemporary Chinese and Information Processing, funded by "211 Project" of The National Tenth Five-year Plan

Research Collaboration

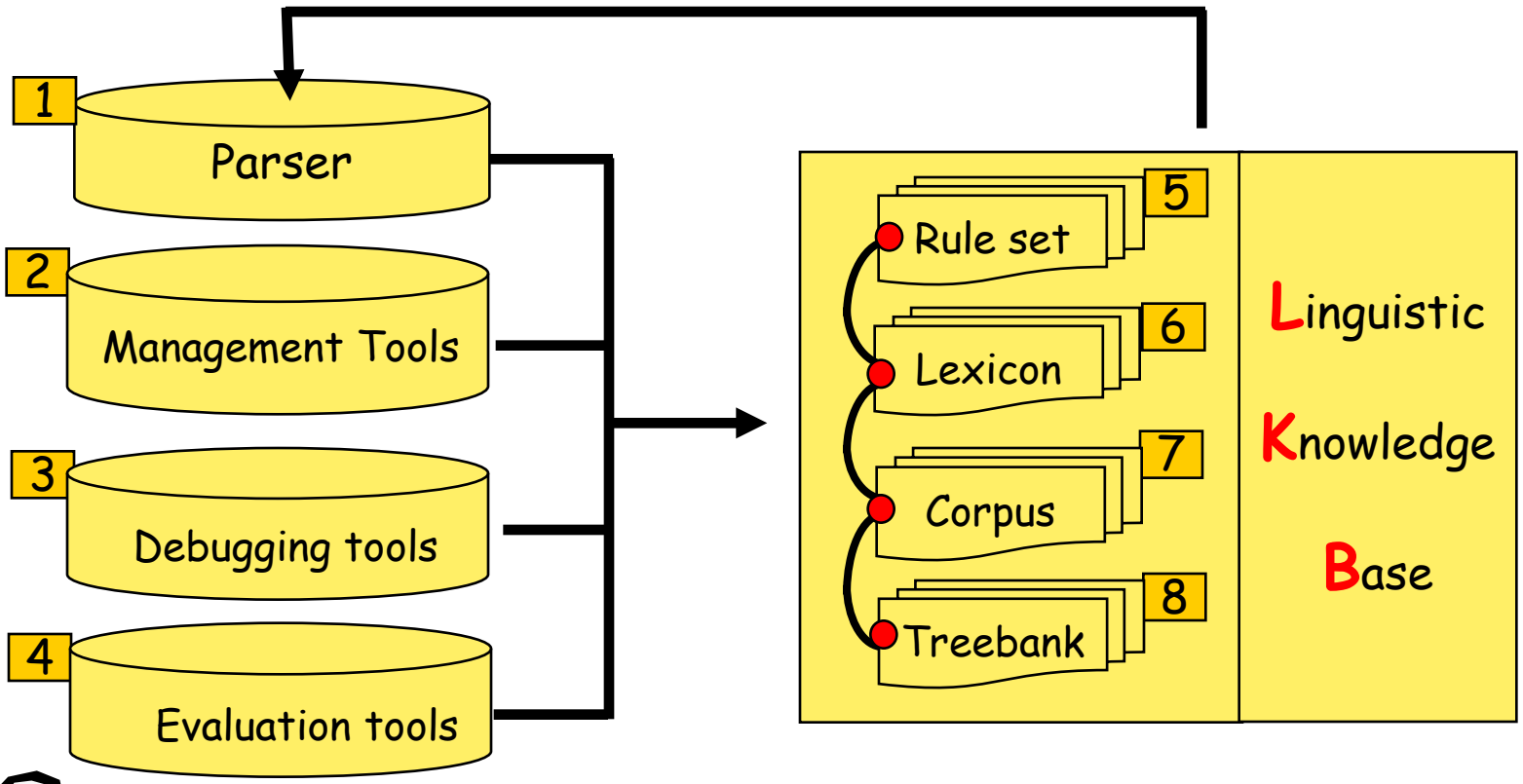


Computing-oriented research on natural language needs interdisciplinary collaboration more than traditional language research works.

Our research team is composed of linguists and computational linguists from four institutes, including:

- Department of Chinese Language and Literature, Peking University
- Center for Chinese Linguistics, Peking University
- Institute of Computing Technology, Chinese Academy of Sciences
- Institute of Computational Linguistics, Peking University

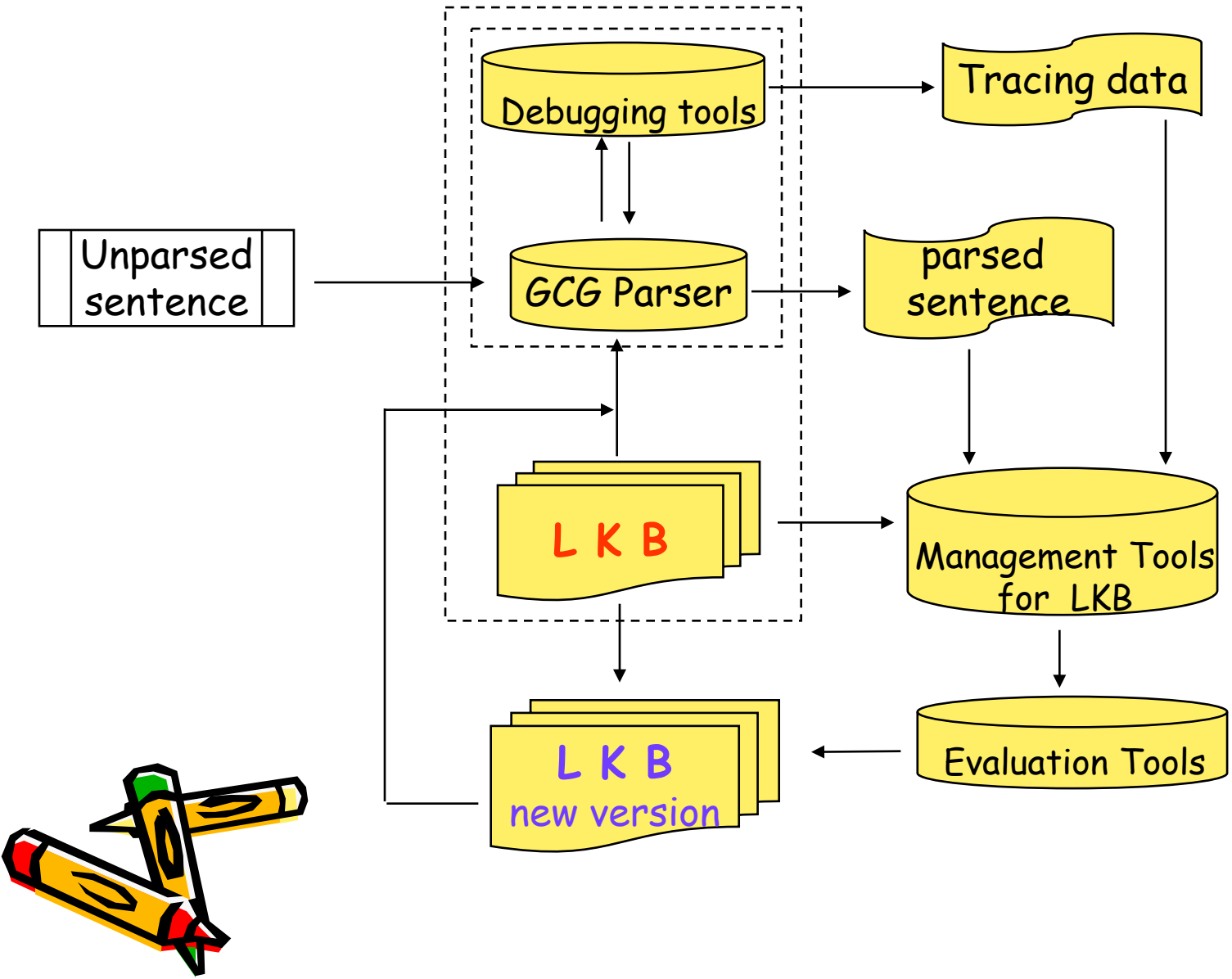
2 What does ICGDE consists of



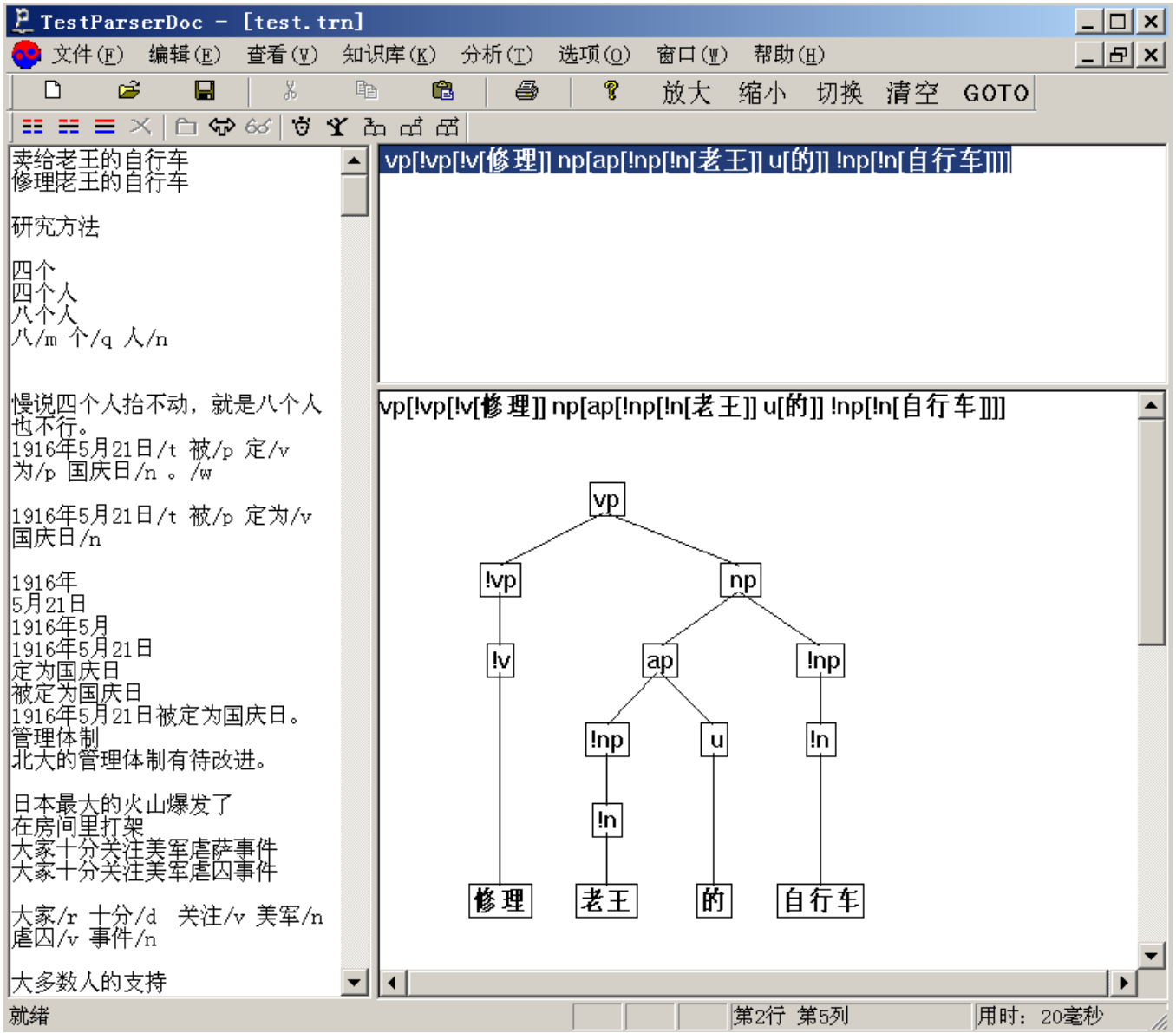
1,2,3,5 will be discussed detailedly in the following sections.

4 is under construction. 6,7,8 are shown in appendix.

Workflow of ICGDE



The main interface of GCG parser



3 What is the characteristics of ICGDE?

In this section, we are going to talk about the highlighted features of the three main modules in ICGDE

1) A Chinese Parser with fully constraint-based grammar formalism (GCG parser)
2) Tools for linguistic rule writing and debugging
3) Tools for building and using Chinese Treebank

3.1 A Chinese Parser with fully constraint-based grammar formalism

About the parser①

- Non-deterministic Bottom-up Chart parser
- Towards a multi-engine approach
- Configurability and Customization

About the linguistic knowledge representation

- CFG rule enhanced with marking the head of a phrase(like HPSG formalism) and the right hand side of a production rule is allowed to be composed of non-terminal symbols and trees(like Tree Adjoining Grammar formalism)
- Each rule is decorated with feature structure-based unification
- Built-in Functions and operators for enhancing the capability of construction rules
- Score mechanism for disambiguation of competing rules

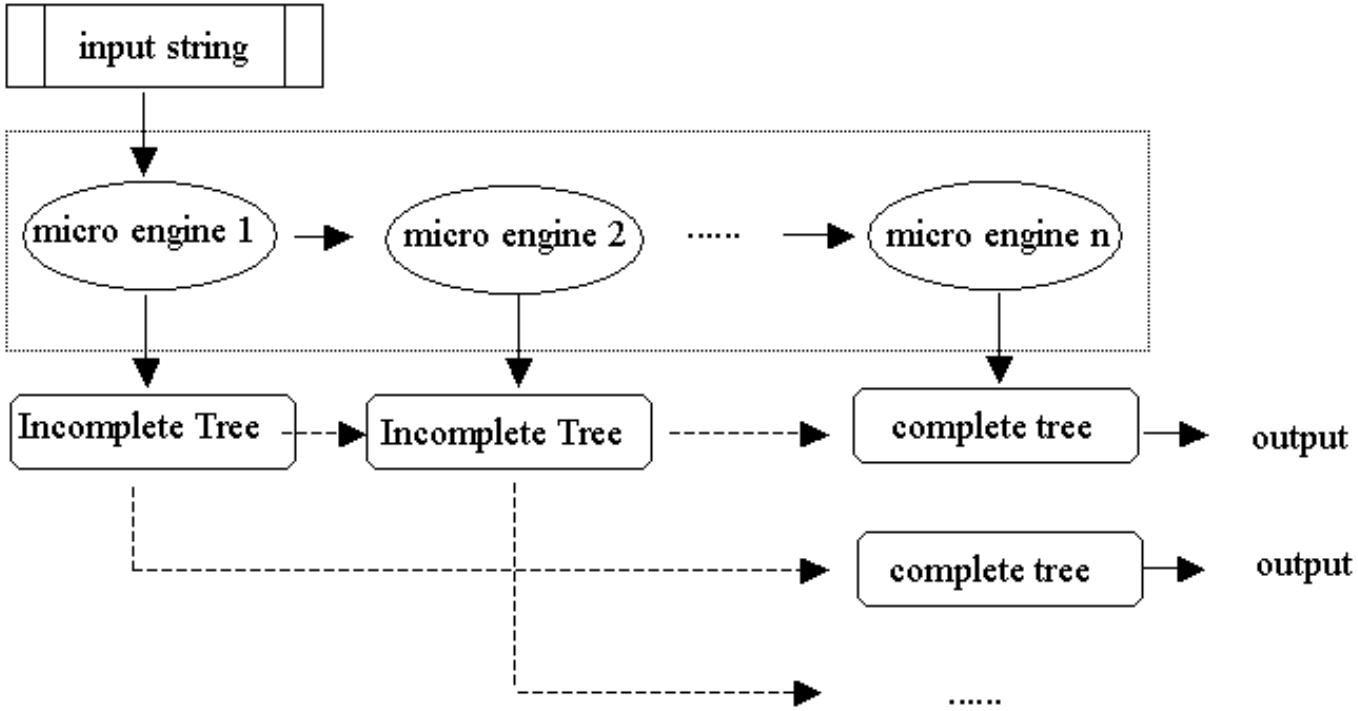
footnote:① The main architecture and the core module of the parser is designed and developed by Dr. Liu Qun

Multi-engine architecture of the parser

- The parser consists of various micro-engines which share a same **chart data structure** while parsing. Each engine uses its own linguistic resource to make contribution to add new nodes with score into the chart. It is allowed to use different engine to do the same job. For example, we can use different Chinese word segmenter and POS-tagger in our parser. The nodes which are derived from different engines and cover the same segment of input string will be selected according to their score.

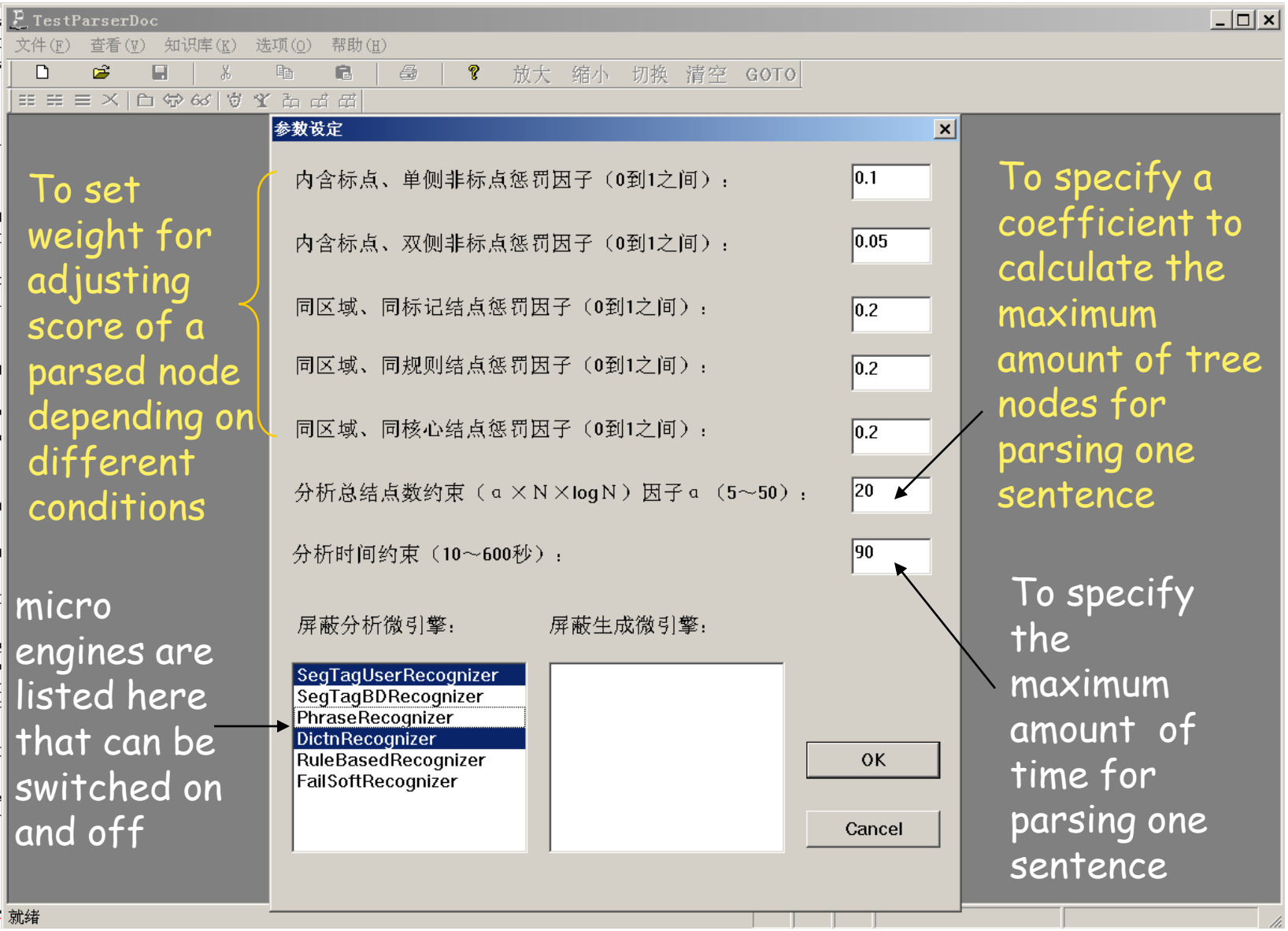
- Each micro-engine is bound to its own linguistic knowledge base, which can be various form and on the different linguistic level, such as lexicon, rule set, corpus, probability parameters, etc..

Multi-engine architecture of the parser



A pipelined multi-engine approach to Chinese parsing

Configurability and Customization



To set weight for adjusting score of a parsed node depending on different conditions

micro engines are listed here that can be switched on and off

To specify a coefficient to calculate the maximum amount of tree nodes for parsing one sentence

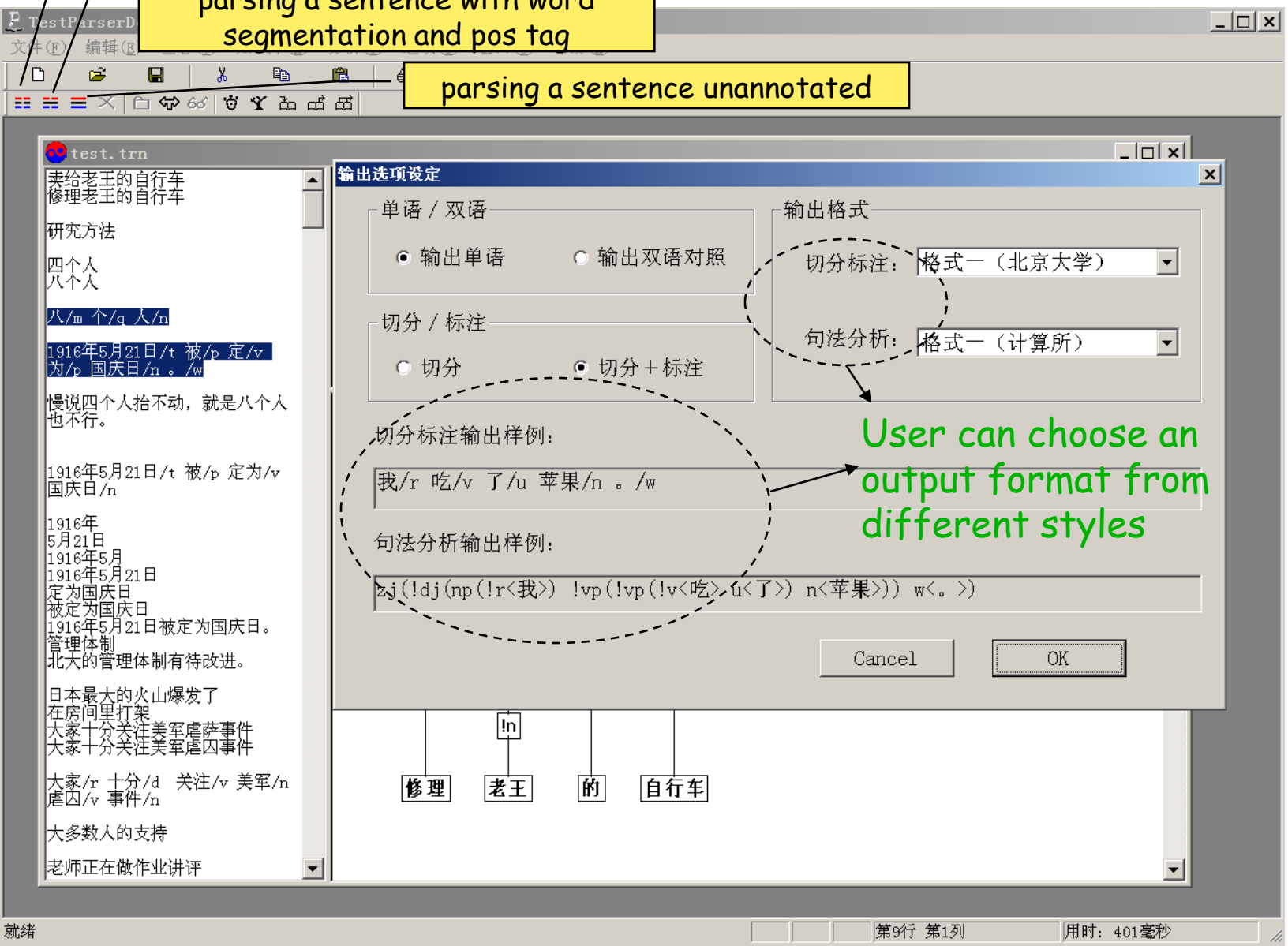
To specify the maximum amount of time for parsing one sentence

Configurability and Customization(cont.)

segmentation and pos tagging

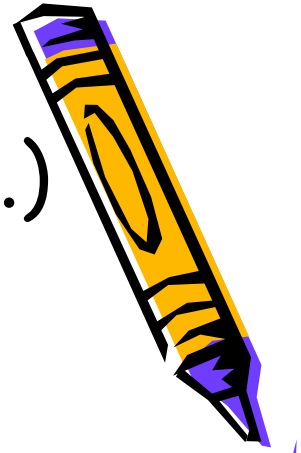
parsing a sentence with word segmentation and pos tag

parsing a sentence unannotated



User can choose an output format from different styles

Configurability and Customization (cont.)



- Each micro-engine can be switched on and off before parsing a sentence
- User can put a new engine into the parser if necessary, for example, name entity recorgnizer and shallow parsing engine may be needed to add for information extraction system.
- User can set up the time limitation for parsing a sentence
- User can set up the maximum amount of tree nodes created by parsing one sentence as a limitation
- User can set weight for adjusting score of a parsed node in the different cases. For example, if a node contains punctuation, its score should be reduced slightly
- User can choose different input and output format

Enhanced CFG production rule

Global rule -- different coverage-- Local rule

pp->!p np

ap->!ap c ap

np->mp !np

pp -> !p<从> tp v<起|开始>

tp -> mcp !q<点> mcp q<分>

ap -> m<一> q<年|天> p<比> m<一> q<年|天> !ap

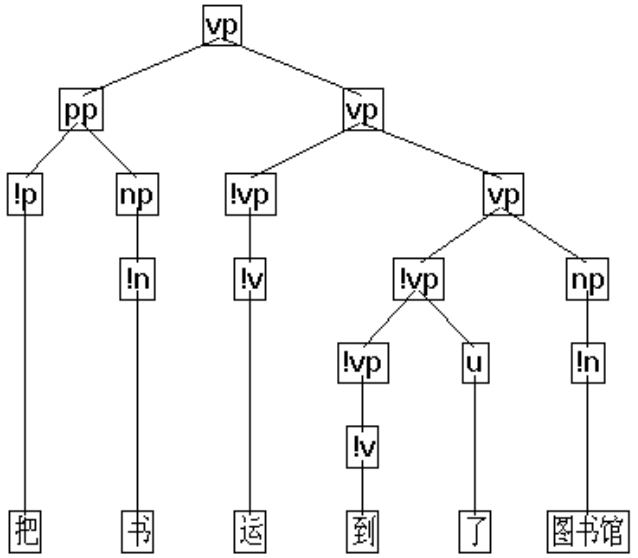
A non terminal symbol in the left hand side of a rule (root node of a tree) can be rewrite as more than two symbols(branches) in the right hand side

… If one or more nodes in the right hand side are specified with terminal symbol,i.e. word, this kind of rule is called "local rule" for its locality.

vp -> pp(!p<把> np) !vp(!vp vp(!vp<<到>> np))

so called "mildly context sensitive grammar formalism" like TAG-style grammar

the right hand side of a rule can contain one or more trees instead of just nodes



carried the books to library

CFG production rule with constraints

start symbol of a rule

stands for root node

stands for the first node appears in the right hand side of the rule

cfg rewriting rule

```
&& {npdz11} np->vp !n  :: $.内部结构=粘合定中,$.定语=%vp,$.中心语=%n,
   %vp.后名=是,%n.前动=是,%vp.内部结构=单词|联合,
   IF %vp.内部结构=单词 THEN %vp.兼类=~n ENDIF,
   IF %vp.内部结构=单词,%vp.音节=2, %n.音节=2 THEN
#Score(5) ENDIF,
   IF %vp.内部结构=单词,%vp.音节=1, %n.音节=2 THEN
#Score(-5) ENDIF,
   IF #GetLength(%vp) > 4 FALSE
```

Rule-ID

delimiter

a sequence of constraints that are separated by comma

symbol is always followed by a built-in function

Basically, each constraint is independent from each other. But it will be easier to read if the constraints can be written in good order.

CFG production rule with constraints(cont.)

There are four types of constraints listed here one after another.

1) The first type of constraint is used to initialize the value of some required features of a rule. For example, the equation "$.内部结构=粘合定中" means that the internal relation of this construction is Modifier-Head.

2) The second type of constraint is used to describe that the constitutes of a phrase should satisfy certain conditions. For example, the equation "%vp.后名=是" means that the property "后名" of the first vp in the phrase should has a value "yes". If the value of a verb is "no", the condition is unmatched and then the verb is rejected by this rule

CFG production rule with constraints(cont.)

3) The third type of constraint is also used to describe the constraints like the second one. But the form is different. Like programming language designed for computer, we can use the sentence of "IF ... FALSE", "If ... Then ... Endif", "IF ... then ... else ... Endif", etc., to describe compound conditions.

4) The fourth type of constraint contains built-in functions and operators that can be used to enhance the capability of writing constraints for rules. The usage of built-in functions will be illustrated in the next slides.

Built-in Functions and Operators

All built-in functions can be classified into the following four types according to the type of value returned by a function.

Bool Function

- MatchPattern
- Score

String Function

- SubString

Int Function

- NumOfChild
- GetLength
- FindWord

Node Function

- GetMostRightNode
- GetMostLeftNode
- GetLeafNode

Operator: > < >= <= == !=

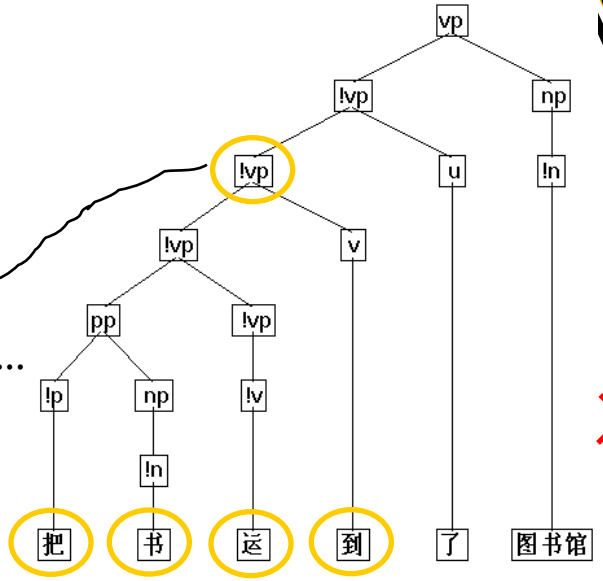
Usage Illustration of Built-in Functions

ex.1 GetLength()

The top right parsing tree is not correct.

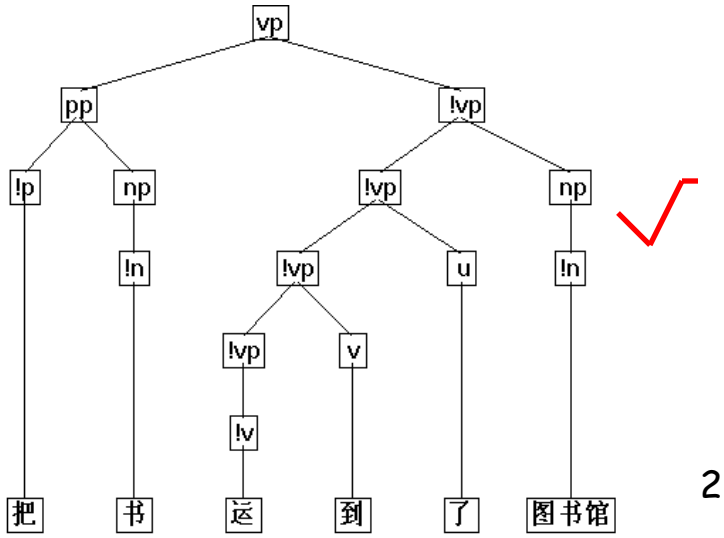
In order to avoid it, we can make use of the function GetLength() in the following rule:

{vp1} vp -> !vp u :: ..., IF #GetLength(%vp) >=4 FALSE, ...



carried the books to library

The verb phrase(vp) "把 + 书 + 运 + 到 " consists of four words. So the function GetLength took this vp as argument will return the integer value 4, which satisfy the unification condition "#GetLength(%vp) >=4 ". The rule vp1, therefore, is rejected while parsing the phrase. And then the expected result is produced. See the bottom right parsing tree.

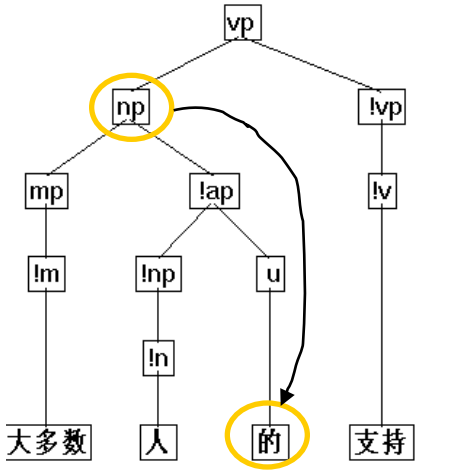


Usage Illustration of Built-in Functions

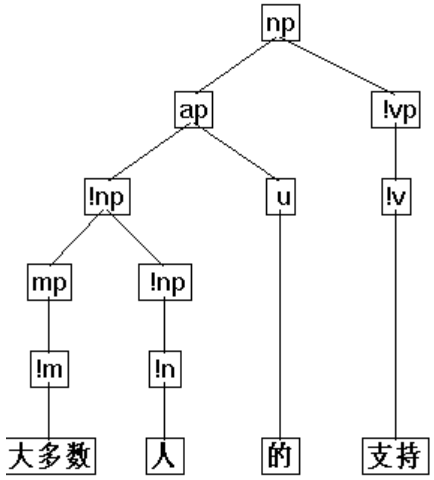
ex.2 GetLeafNode()

The function GetLeafNode takes two arguments. The first argument is a tree node, the second argument is the index of a leaf node which is projected by the tree node that the first argument denotes. For instance, in order to eliminate the incorrect parsing tree(shown top right), we can make use of this function in writing the following rule:

{vp2} vp -> np !vp :: ..., IF %GetLeafNode(%np,-1).原形=的 FALSE, ...



most people DE support
support by most people



The second argument "-1" is interpreted as the most right leaf node of the first argument, i.e. np. The above constraint means that a np ended with "的" can NOT be positioned before a vp as its adverbial modifier. In the above rule, GetLeafNode returns the leaf node "的", whose feature "原形" (lexmeme) has the value "的". So the unification will succeed.

Score mechanism

- Using a simple score mechanism to make fine granularity

  set different score thresholds for individual rules or different lexicons depending on their credibility

for instance:

1) basic lexicon vs. extended lexicon
2) global rule vs. local rule
3) two global rules with same construction but different frequency

Score mechanism (cont.)

Local rule

&& {vpdao1} vp -> pp(!p<把> np) !vp(!vp vp(!vp<<到>> np))

```
vp -2.10694{vpdao1}
====pp -0.694124{pp3}
    ====p<把> 0{}
    ====np -0.00097704{np00}
        ====n<书> 0{}
====vp -1.40497{vpyundao}
    ====vp -0.00097704{vp00}
        ====v<运> 0{}
    ====vp -0.71085{vpsb1}
        ====vp -0.0167254{vp1}
            ====vp -0.00097704{vp00}
                ====v<到> 0{}
            ====u<了> 0{}
        ====np -0.00097704{np00}
            ====n<图书馆> 0{}
```

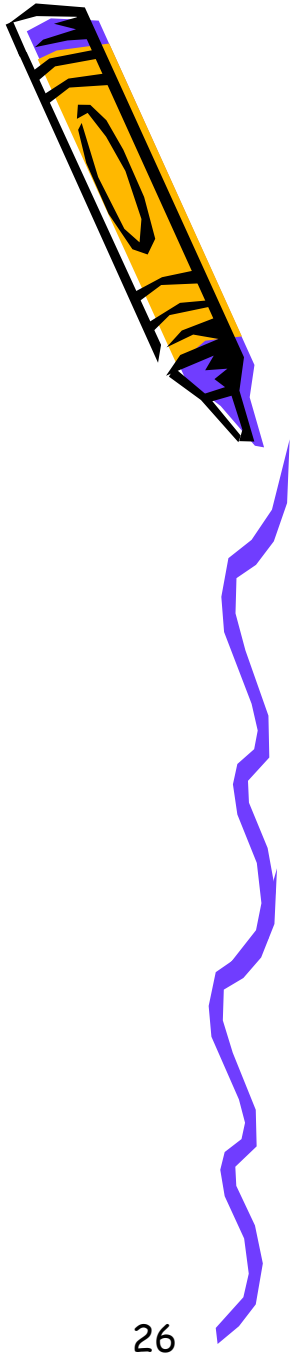
Global rule

&& {vpzz1} vp->pp !vp

```
vp -2.79127{vpzz1}
====pp -0.694124{pp3}
    ====p<把> 0{}
    ====np -0.00097704{np00}
        ====n<书> 0{}
====vp -1.404{vpsb1}
    ====vp -0.709873{vp1}
        ====vp -0.694124{vpsbu1}
            ====vp -0.00097704{vp00}
                ====v<运> 0{}
            ====v<到> 0{}
        ====u<了> 0{}
    ====np -0.00097704{np00}
        ====n<图书馆> 0{}
```

A node generated by local rule is assumed to have higher score than the node with same label that is generated by global rule. In the case of above example, the left tree will be ranked higher than the right one and be selected as the final result to output.

3.2 Tools for linguistic rule writing and debugging



- Management tools for linguistic Knowledge base
- Tracing the parsing process

Management Tools for Linguistic Knowledge Base

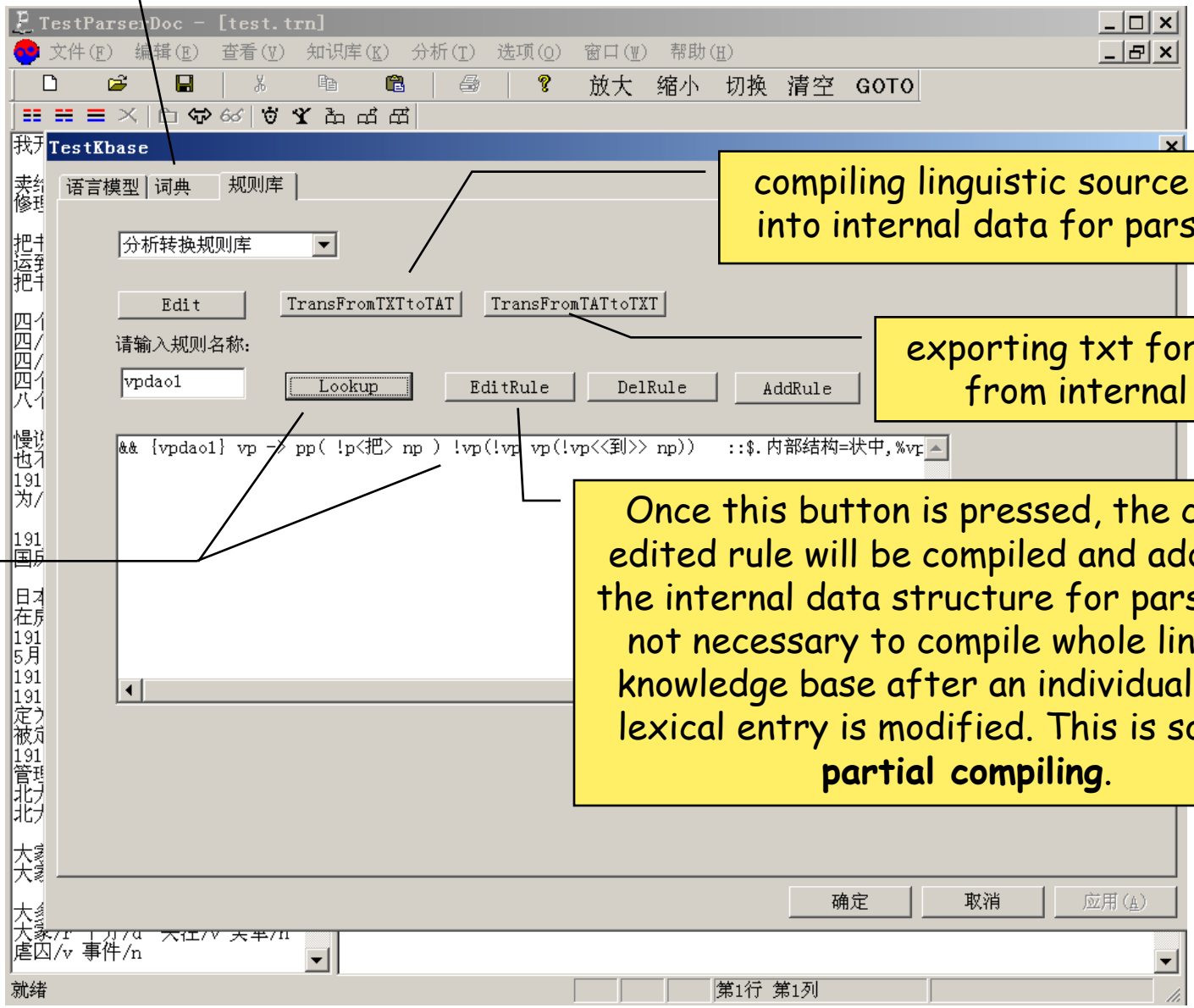
- Grammar checking for linguistic knowledge base
- The interface for linguistic knowledge base

  edit, add, delete, search, etc..

- Compiling linguistic source files to object files which are used in parsing indeed.

Management Interface of LKB

Cascading style menu



compiling linguistic source file into internal data for parsing,

exporting txt format file from internal data

Handy editing of a rule searched by it's title

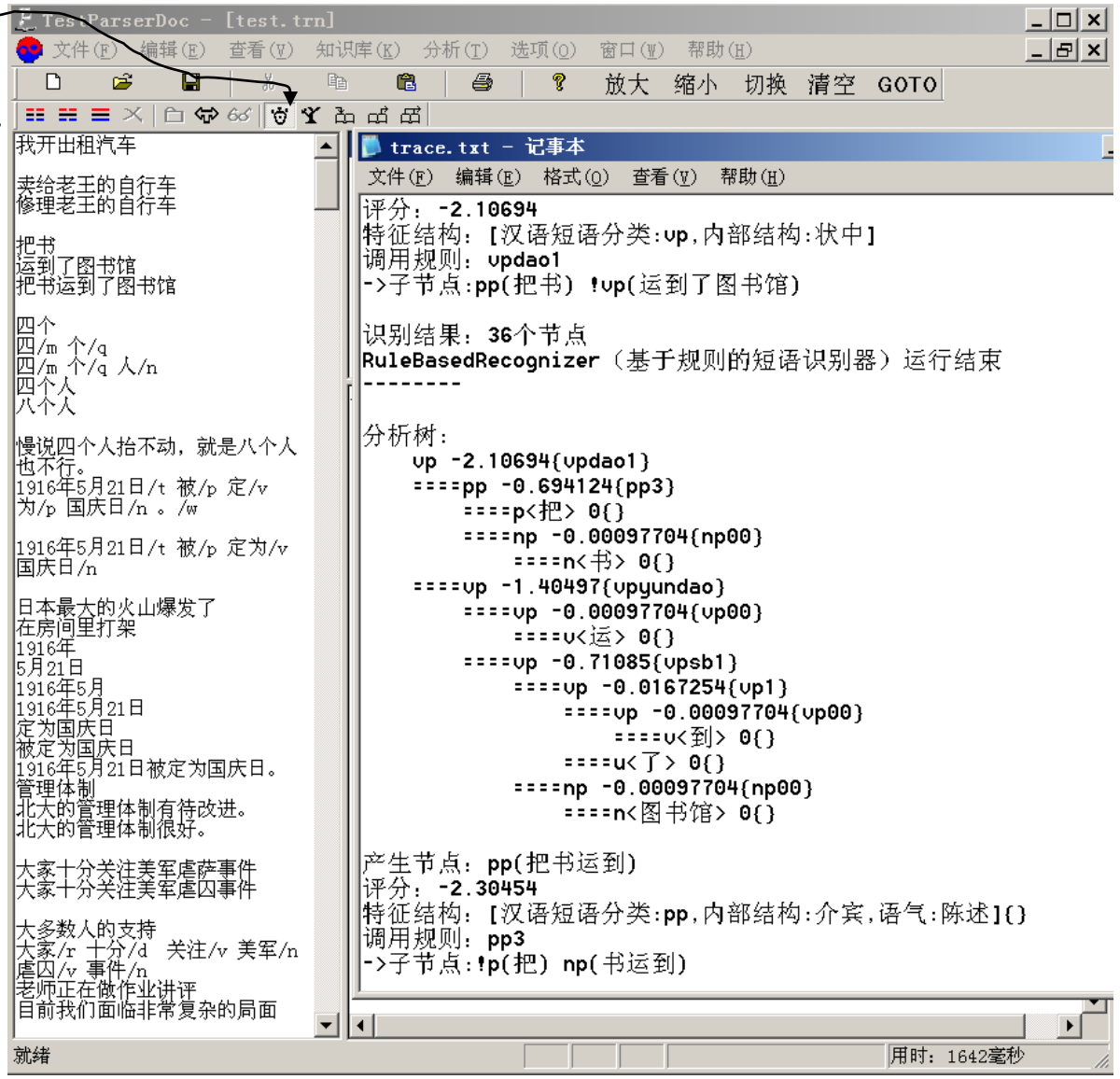
Once this button is pressed, the current edited rule will be compiled and added into the internal data structure for parsing. It's not necessary to compile whole linguistic knowledge base after an individual rule or lexical entry is modified. This is so-called **partial compiling**.

Tracing

There is a button to trace the execution of the parser. It can be switched on and off before start of parsing. Tracing can give more insight on what the parsing behaves.

All detailed tracing information is saved in a text file which contains the following descriptions.

- The micro-engines are called in parsing
- The feature structure and the score of each node
- The rules are employed in parsing
- The parsed trees



We have planned to add the function of setting breakpoint for enforcing the debugger. But it's not available now.

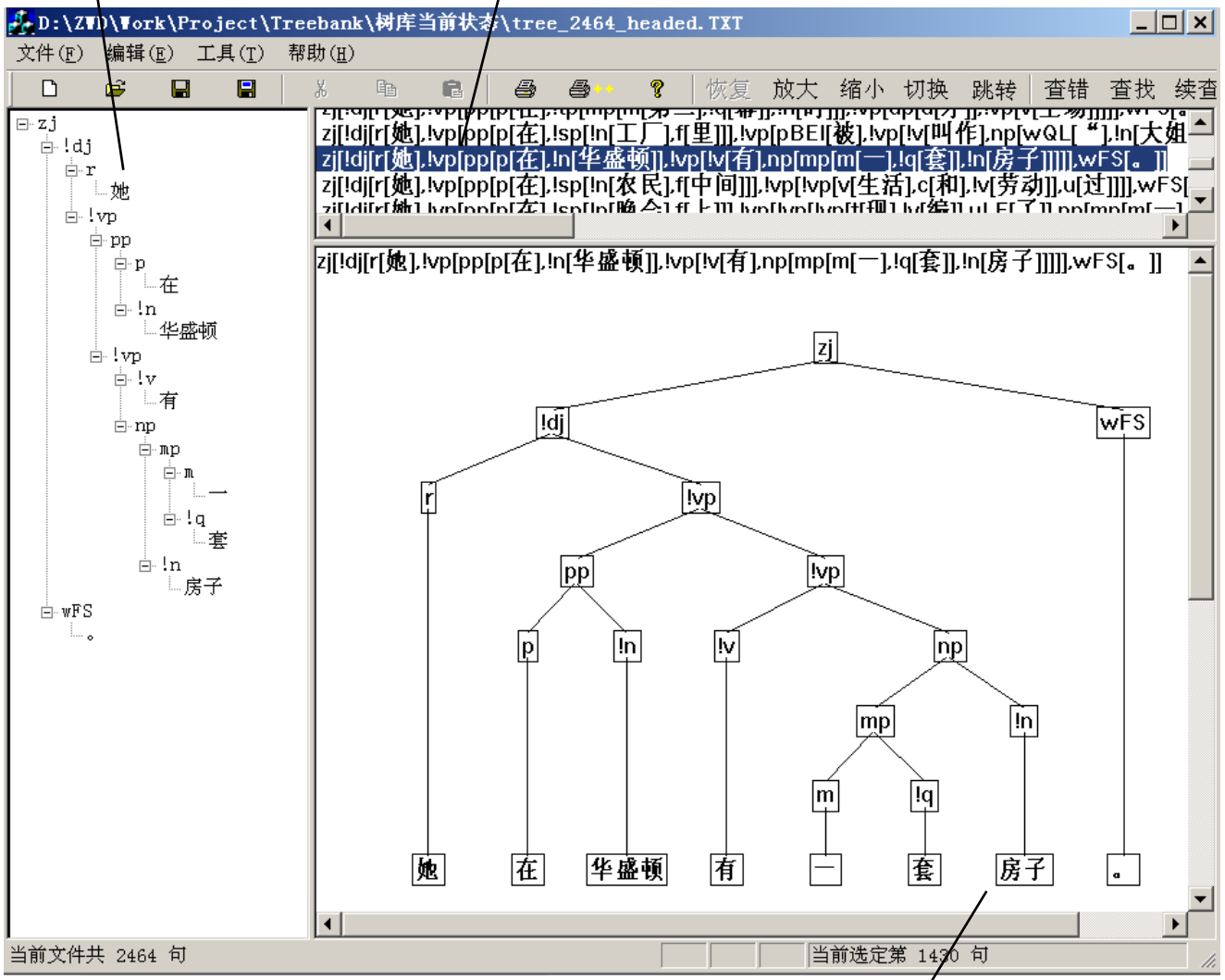
3.3 Tools for building and using Chinese Treebank

- Tree Editor
- Rule Extraction
- Search in Treebank
- Subtree replacement

Tree Editor

a list of trees inputted from the output of the GCG parser

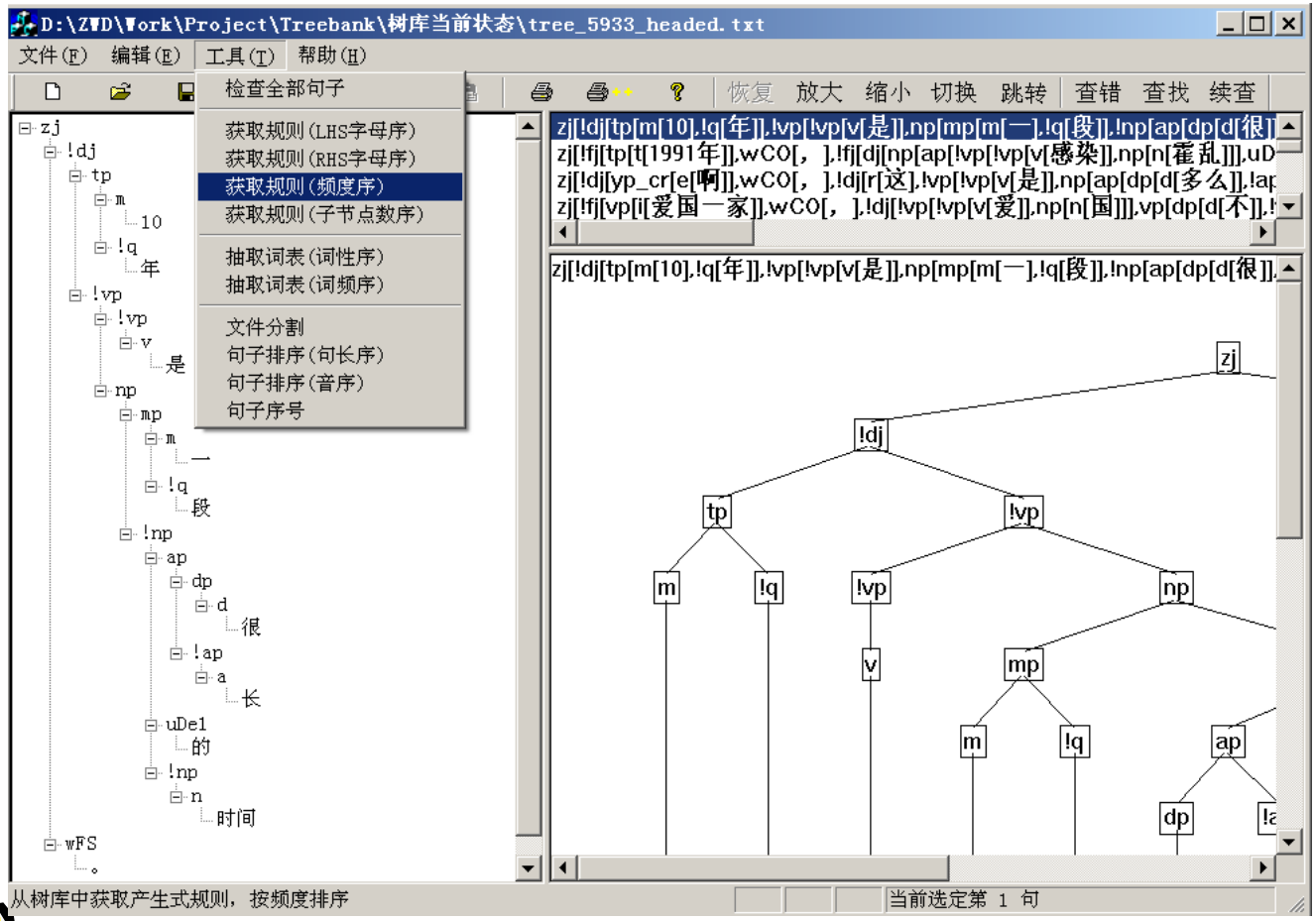
tree editing area, in which user can edit manually the label of a tree node, add a new child node to a selected tree node, combine two leaf nodes, drag a selected tree node to be the child node of another target tree node, and so on.



tree viewing area, in which a parsed tree is shown as user-friendly graphic inverted tree.

Rule extraction from a treebank

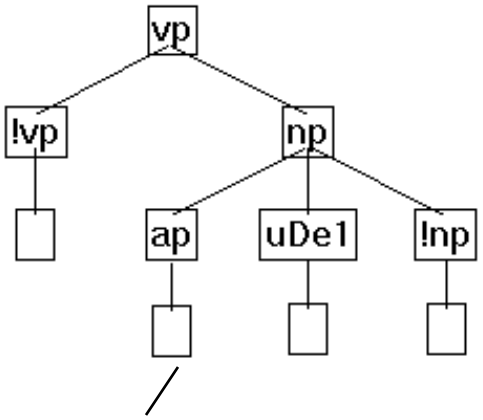
It can increase the coverage of rule set effectively to extract rules from a manually corrected treebank.



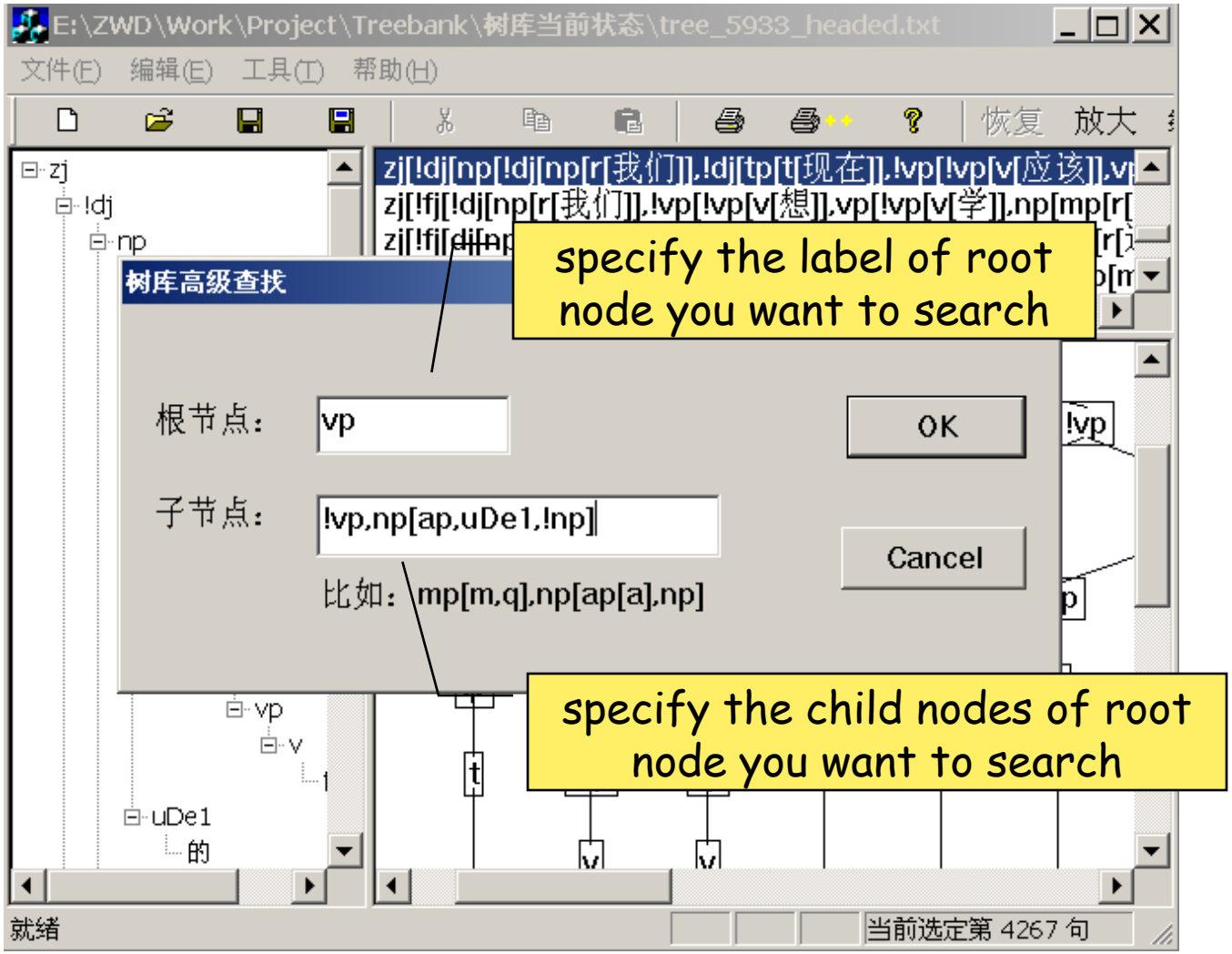
The rules extracted can be sorted by different criteria, such as the label of root node, the right hand side symbols of a rule, frequency of rule, number of child nodes of a root node

Search in treebank

The tree you want to search

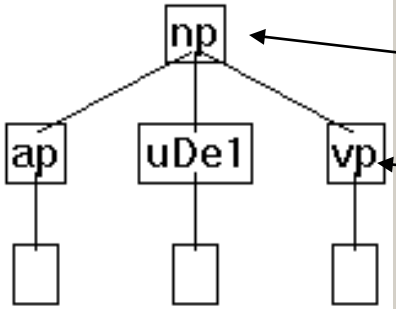


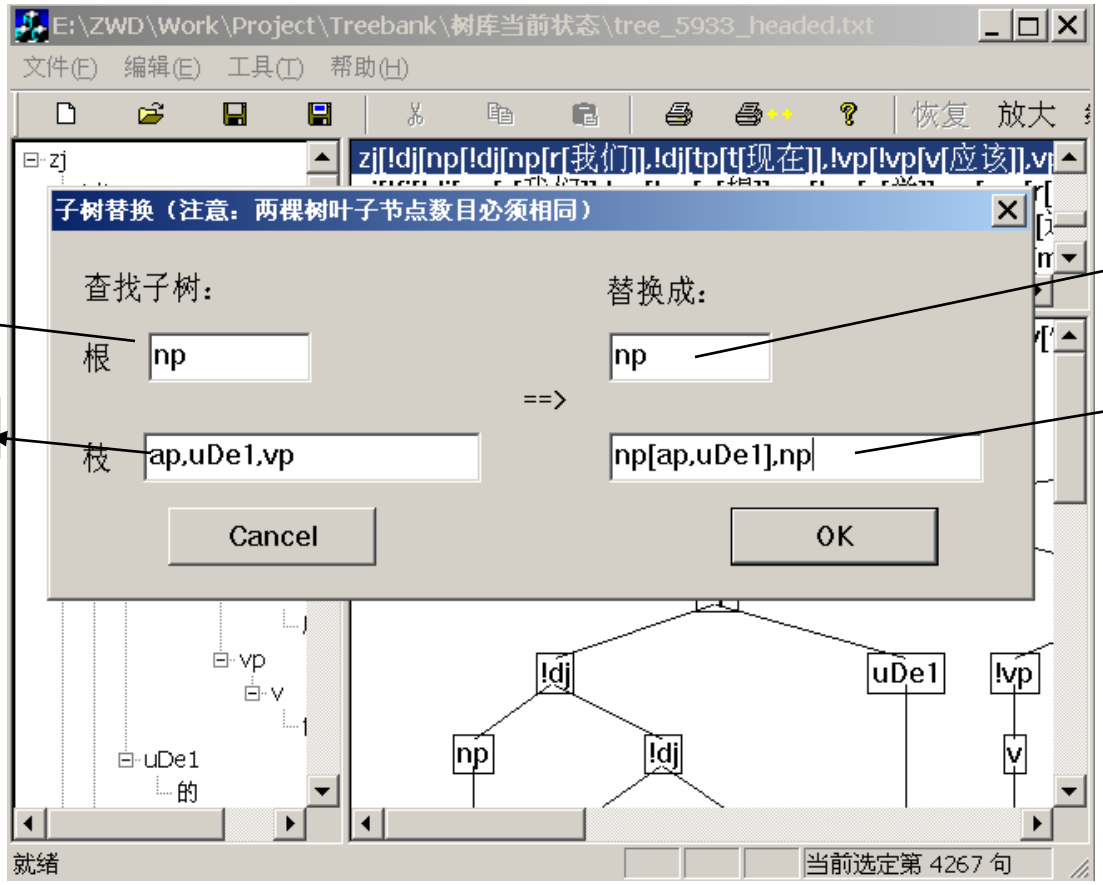
Note: The white square can match any tree node



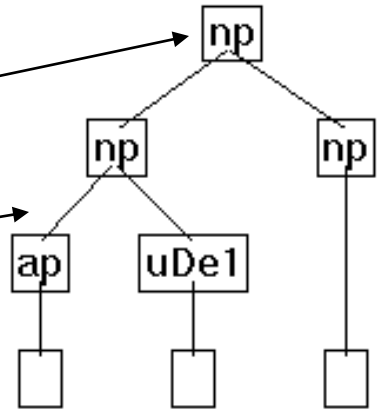
Subtree replacement

find the tree below that is embedded in larger trees





replace the found tree(s) with the tree below



You can type * as both root node and leaf node that matches any label for a tree node when you input a tree for searching and replacing.

4 Experiences with ICGDE

- Statistics of current linguistic knowledge database
- Precision and efficiency of the parser
- Statistics of current PKU Chinese treebank

4.1 Statistics of current linguistic knowledge database

- Core lexicon: more than 43,000 entries with rich syntactic and semantic information in attribute-value format
- Appended lexicon: more than 200,000 entries with just part-of-speech tag. Entries in this lexicon are almost compound word or multi-words unit.
- Phrase database: more than 30,000 entries which are annotated with part-of-speech of words that constructs a phrase and the functional category and internal structure of the phrase
- Rule set: more than 900 rules, including 330 global rules and 577 local rules

4.2 Precision and efficiency of the parser

- The test set contains 500 sentences, 5000 Chinese characters (4028 words).
- The average sentence length is 10 characters per sentence (or 8 words per sentence).
- The computer used for testing is configured with 2.60GHz Pentium 4 processor, 512MB SDRAM

Large-scale Extended Lexicon engine	Phrase lexicon engine	Rule-based engine	Parsing time (ms)	Bracketing Recall	Bracketing Precision	Complete match	Average crossing	No crossing	2 or less crossing	Tagging accuracy
+	+	+	151678	67.07%	74.18%	6.37%	1.27	49.06%	79.03%	70.37%
+	-	+	114966	67.00%	74.14%	6.39%	1.27	48.87%	78.95%	70.37%
-	+	+	66345	65.06%	73.34%	5.87%	1.27	47.20%	80.80%	69.98%
-	-	+	63842	65.06%	73.34%	5.87%	1.27	47.20%	80.80%	69.98%

The above statistics is produced by Dr. Chang Baobao who used the bracket scoring program developed by Dr. Satoshi Sekine. The program is free for downloading online.

4.3 Statistics of current PKU Chinese treebank

- There are 14,736 bracketed sentences that were corrected manually in the treebank.
  9.628 words on average each sentence

- The treebank contains 12,685 word types and 141,879 word tokens now.

- 2477 rules can be extracted from the treebank

5 Future works

The ICGDE and the linguistic knowledge base described above can continue to be actively developed.

Our planned research is focused in the following areas:

- improving ability of current debugging tool, including to set breakpoint, record more tracing data while parsing, etc..
- extending the current rule set and lexicon through acquisition from corpora
- enhancing functionality of current management tool for linguistic knowledge base, especially focusing on environment of collaborative grammar coding
- developing evaluation technology for integration of existing linguistic resources

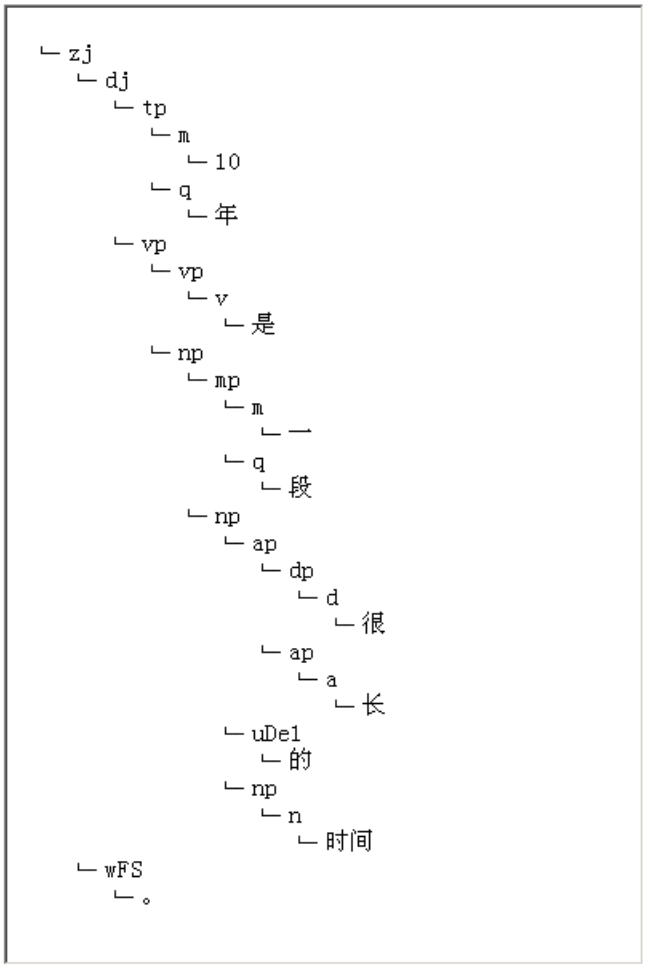
Appendix: Resources Online

- PKU Chinese Treebank Online
  http://ccl.pku.edu.cn/doubtfire/projects/chinesesentencestructure/

- Online Chinese Semantic Lexicon
  http://ccl.pku.edu.cn/ccl_sem_dict/

- Online Chinese Balanced Corpus with search engine support
  http://ccl.pku.edu.cn/ccl_corpus/jsearch/

北大中文树库（Chinese Treebank）工程



浏览第 1 页/(共17页,423句)

后页 尾页

1) 10年是一段很长的时间。
2) 1991年，感染霍乱的病人多达3026人，其中440人死亡。
3) 啊，这是多么美妙的前景！
4) 爱国一家，爱国不分先后。
5) 安葬他的地方很美。
6) 安装灯的人是我的同学。
7) 安娜自己打开了门。
8) 八成他不来了。
9) 八点时，他正在吃早饭。
10) 八减三得五。
11) 八月份物价将上涨。
12) 把东西清理干净
13) 把花盆搬到外面去。
14) 把画挂到墙上！
15) 把火拨一拨
16) 把课文再读一遍！
17) 把龙头打开
18) 把那本书扔给我
19) 把那本杂志扔给我
20) 把你的名字告诉我
21) 把你的衣服挂起来
22) 把瓶子放在桌上。
23) 把其余的吃的留到明天。
24) 把全部情况告诉她
25) 把身子探出窗外是很危险的。

后页 尾页

Online Chinese Semantic Lexicon (CCL, PKU)

Entries can be accessed for browsing, editing, searching via Internet. Authorized users can add new entries and delete entries.

hierarchical semantic tree for Chinese nouns

Online Concordance Service for Chinese Corpus

- Both modern and classical Chinese are contained in the balanced corpus that exceeds **100 million Chinese characters** now.
- Support both single term query and complex multi-term query for searching which can meet most requirements for language studying.

Reference

- Ann, Copestake & Flickinger Dan (2000), An Open Source Grammar Development Environment And Broad-coverage English Grammar Using HPSG, In Proceedings of LREC 2000 (The 2nd International Conference on Language Resoruce & Evaluation), Zappeion Megaron, Greece, May 31 – June 2, 2000.
- Blevins, James(2003), Feature-based Grammar, In Borsley, R.D. & Borjars, K. eds., Non-transformational Syntax, Oxford: Blackwell, to be published in 2005.
- Blache, Philippe, Marie-Laure Guĭnot, Tristan van Rullen (2003), A Corpus-based Technique for Grammar Development, In Proceedings of The Shallow Processing of Large Corpora Workshop (SProLaC 2003), Lancaster University (UK), 27 March, 2003.
- Borsley, Robert D., 1996, *Modern Phrase Structure Grammar*, No. 11 in Blackwell textbooks in Linguistics, Blackwell Publishers Inc..
- Chen,Feng-yi, et al. 1999, Sinica Treebank, Computational Linguistics and Chinese Language Processing,4(2):183-204
- Chen, Keh-Jiann & Yu-Ming Hsieh, 2002, Chinese Treebanks and Grammar Extraction, CJNLP'2002, Peking University, 2002.10.30-11.2
- Erbach, Gregor (1991), A Flexible Parser for a Linguistic Development Environment, In O. Herzog & C.-R. Rollinger eds., Text Understanding in LILOG, Springer, 1991, pp. 74-87

Reference

- Heinecke, Johannes, Jurgen Kunze, Wolfgang Menzel, and Ingo Schroder (1998), Eliminative parsing with graded constraints. In Proceedings of 17th International Conference on Computational Linguistics, 36th Annual Meeting of the ACL, Coling-ACL '98, Montreal, Canada, 1998.
- Knight,Kevin,1989, Unification: A Multidisciplinary Survey, ACM Computing Surverys, Vol.21, No.1.
- Sag, Ivan A. & Thomas Wasow, 1999, *Syntactic Theory: A Formal Introduction*, CSLI Publications, Stanford, California.
- Schmid, Helmut (1999), YAP: Parsing and Disambiguation With Feature-Based Grammar. PhD thesis, Institute of Maschinelle Sprachverarbeitung, University Stuttgart, Germany, 1999.
- Suzuki, Hisami (2002), A Development Environment for Large-scale Multi-lingual Parsing Systems, In Workshop on Grammar Engineering and Evaluation (Post-conference workshop in conjunction with COLING-2002, Taipei, Sept. 1, 2002).
- Uszkoreit, Hans(2002), New Chances for Deep Linguistic Processing, Coling2002,Taipei.
- Volk, Martin & Dirk Richarz (1997), Experiences with the GTU Grammar Development Environment, ACL workshop on Environments for Grammar Development, 1997, Madrid, Spain.
- Xu Ruifeng, et al., 2004, The Construction of A Chinese Shallow Treebank, ACL2004 SIGHAN Workshop, July 21-16, 2004. Barcelona, Spain.

詹卫东（2000），《面向中文信息处理的现代汉语短语结构规则研究》，清华大学出版社，广西科学技术出版社。

Thank you
for your attention

Welcome to

http://ccl.pku.edu.cn