Parsing Chinese with an Almost-Context-Free Grammar

Xuanyin Xia and **Dekai Wu**
HKUST
Department of Computer Science
University of Science and Technology
Clear Water Bay, Hong Kong
{samxia,dekai}@cs.ust.hk

Abstract

We describe a novel parsing strategy we are employing for Chinese. We believe progress in Chinese parsing technology has been slowed by the excessive ambiguity that typically arises in pure context-free grammars. This problem has inspired a modified formalism that enhances our ability to write and maintain robust large grammars, by constraining productions with left/right contexts and/or nonterminal functions. Parsing is somewhat more expensive than for pure context-free parsing, but is still efficient by both theoretical and empirical analyses. Encouraging experimental results with our current grammar are described.

Introduction

Chinese NLP is still greatly impeded by the relative scarcity of resources that have already become commonplace for English and other European languages. A strategy we are pursuing is the use of automatic methods to aid in the acquisition of such resources (4, 5, 6). However, we are also selectively engineering certain resources by hand, for both comparison and applications purposes. One such tool that we have been developing is a general-purpose bracketer for unrestricted Chinese text. In this paper we describe an approach to parsing that has evolved as a result of the problems we have encountered in making the transition from English to Chinese processing.

We have found that the primary obstacle has been the syntactic flexibility of Chinese, coupled with an absence of explicit marking by morphological inflections. In particular, compounding is extremely flexible in Chinese, allowing both verb and noun constituents to be arbitrarily mixed. This creates extraordinary difficulties for grammar writers, since robust rules for such compound forms tend to also accept many undesirable forms. This creates too many possible parses per sentence. We employ probabilistic grammars, so that it is possible to choose the Viterbi (most probable) parse, but probabilities alone do not compensate sufficiently for the inadequacy of structural constraints.

There are two usual routes: either (1) keep the context-free basis but introduce finer-grained categories, or (2) move to context-sensitive grammars. The former strategy includes feature-based grammars with weak unification. One disadvantage of this approach is that some features can become obscure and cumbersome. Moreover, the expressive power remains restricted to that of a CFG, so certain constraints simply cannot be expressed. Thus, many systems opt for some variety of context-sensitive grammar. However, it is easy for parsing complexity in such systems to become impractical.

We describe an approach that is not quite context-free, but still admits acceptably fast Earley-style parsing. A benefit of this approach is that the form of rules is natural and simple to write. We have found this approach to be very effective for constraining the types of ambiguities that arise from the compounding flexibility in Chinese.

In the remainder of this paper, we first describe our grammar framework in Sections 2–4. The parsing strategy is then described in Section 5, followed by current experimental results in Section 6.

The Grammar Framework

We have made two extensions to the form of standard context-free grammars:

1. Right-hand-side contexts
2. Nonterminal functions

We would like to note at the outset that from the formal language standpoint, the complications introduced by the form of our production rules have so far hindered theoretical analyses of the formal expressiveness characteristics of this grammar. Because of the nature of the constraints, it is unclear how the expressiveness relates to, for example, the more powerful unification-based grammars that are widespread for English.

At the same time, however, we will show that the natural format of the rules has greatly facilitated the writing of robust large grammars. Also, an efficient Earley-style parser can be constructed as discussed below for grammars of this form. For our applications, we therefore feel the effectiveness of the grammar form compensates for the theoretical complications.

We now describe the extensions, but first define some notation used throughout the paper. A traditional *context-free grammar* (CFG) is a four-tuple $G = (N, \Sigma, P, S)$, where N is a finite set of *nonterminal* symbols, Σ is a finite set of *terminal* symbols such that $N \bigcap \Sigma = \emptyset$, p is a finite set of *productions* and $S \in N$ is a special designated *start symbol*. Productions in P are denoted by symbol P_r, $1 \le r \le |P|$, and have the form $D_r \to Z_{r,1} Z_{r,2} \cdots Z_{r,\pi_r}$, $\pi_r \neq 0$, where $D_r \in N$ and $Z_{r,j} \in N \bigcup \Sigma$, $1 \le j \le \pi_r$.

Right-hand-side contexts

We introduce *right-hand-side contexts* to improve rule applicability decisions for complex compounding phenomena. The difficulty that ordinary CFGs have with complex compounding phenomena can be seen from the following example grammar fragment:

1. $RelPh \to NP \;\; vn$ 的(de)
2. $Nom \to NP \;\; vn$ 的(de)
3. $NP \to \;\; Nom$
4. $NP \to \;\; RelPh \;\; NP$
5. $NP \to \;\; NP \;\; NP$

Here, *RelPh* is a relative phrase, *Nom* is a nominalization (similar to a gerund), *vn* is lexical verb category requiring an *NP* argument, and 的(de) is a genitive particle.

The sequence

(1) a. 警務處 提供 的 答覆
b. jǐngwùchù tígōng de dáfù
c. police provide — answer
d. the answer provided by police

can be parsed either by

$$[[[\text{警務處}]_{NP}[\text{提供}]_{vn}\text{的}]_{RelPh}[\text{答覆}]_{NP}]_{NP}$$

or by

$$[[[[\text{警務處}]_{NP}[\text{提供}]_{vn}\text{的}]_{Nom}]_{NP}[\text{答覆}]_{NP}]_{NP}$$

However the latter parse is not linguistically meaningful, and is rather an artifact of the overly general noun compounding rule 5. The problem is that it becomes quite cumbersome in a pure CFG to specify accurately which types of noun phrases are permitted to compound, and this usually leads to excessive proliferation of features and/or nonterminal categories.

Instead, the approach described here augments the CFG rules with a restricted set of contextual applicability conditions. A production in our extended formalism may have left and/or right context, and is denoted as $P_r = \{L\} Z_{r,1} Z_{r,2} \cdots Z_{r,\pi_r} \{R\}$, where $L, R \in (N \bigcup \Sigma)^*$ and the left context condition L and the right context condition R are of a form described below. These context conditions help cut the parser's search space by eliminating many possible parse trees, increasing both parsing speed and accuracy. Though ambiguities remain, the smaller number of parses per sentence makes it more likely that most-probable parsing can pick out the correct parse.

Nonterminal functions

In addition, a second extension is the introduction of a variety of *nonterminal functions* that may be attached to any nonterminal or terminal symbol.[1] These functions are de-

[1] The term *nonterminal functions* was chosen for mnemonic purposes; it is actually a misnomer since they can be applied to terminal symbols as well.

signed to facilitate natural expression of conditions for reducing ambiguities. Some of the functions are simply notational sugar for standard CFGs, while others are context-sensitive extensions. These functions are list in the following sections. By convention, we will use a and b for symbols that can be either terminals or nonterminals, c for terminal symbols only, d for the semantic domain of a terminal, and i for an integer index.

The not function

The *not* function is denoted as **/!b**, which means any constituent not labeled b. Note that this feature must not be used with rules that can cause circular derivations of the type $A \Rightarrow^* A$, since this would lead to a logical contradiction.

In the previous example, if we change rule 2 to

$$Nom \rightarrow NP \quad vn \text{ 的 } \{/!NP\}$$

the new right condition $/!NP$ prevents rule 2 from being used within cases such as rule 5, where the immediately following constituent is an NP. This causes the the correct parse to be chosen:

$$[[[\text{警務處}]_{NP}[\text{提供}]_{vn}]_{RelPh}[\text{答覆}]_{NP}]_{NP}$$

We have only found this function useful for left and right contexts, rather than the main body of production right-hand-sides.

The excluded-category function

The *excluded-category* function is denoted as **a/!b** that means a constituent labeled a, which moreover cannot be labeled as b. Again, not to be used with rules that can cause circular derivations.

The main purpose of the excluded-category function is to improve robustness when the grammar coverage inadequacies prevent a full parse tree from being found. In such cases, our parser will instead return a partial parse tree, as discussed further in Section 5. The excluded-category function can help improve the chances of choosing the correct rules within the partial parse tree.

For example, consider its use with the verb phrase construction

$$\text{把} \; NP \quad verb \quad (Obj)$$

which is known as the 把(bǎ)-construction. If the verb has part of speech vn, then it is monotransitive and only one object is needed to form a VP, but if the verb is a ditransitive vnn, then a second object is needed to form the VP.

An example of the monotransitive case is[2]

(2) a. 把 飯 吃 了
b. bǎ fàn chī le
c. — food eat —
d. have eaten the food

while an example of the ditransitive case is

(3) a. 把 飯 送 人 了
b. bǎ fàn sòng rén le
c. — food give somebody —
d. give food to somebody

The former phrase can be correctly parsed by the monotransitive rule

$$VP \rightarrow \text{把} \; NP \quad vn$$

Suppose that the parser is unable to find any full parse tree for some sentence that includes the latter phrase. The above monotransitive rule would still be considered by the parser, since it is performing partial parsing, and this rule matches the subsequence 把 飯 送. In fact this is not the correct rule for the ditransitive phrase—the VP is not 把 飯 送 but rather 把 飯 送 人 了—but we would not be able to distinguish the monotransitive and ditransitive cases 把 飯 吃 and 把 飯 送, because both 吃 and 送 can have part of speech vn. Thus the monotransitive subparse might incorrectly be chosen for the partial parse output (whether this happens depends rather arbitrarily on the possible subparses found over the rest of the sentence).

The key to eliminating the incorrect possibility altogether is that only 送 can also have the part of speech vnn. We refine the rule with our *excluded-category function*:

$$VP \rightarrow \text{把} \quad NP \quad vn/!vnn$$

[2] For this and all subsequent examples, (a) is the Chinese written form, (b) is its pronunciation, (c) is its word gloss ('—' means there is no directly corresponding word in English), and (d) is its approximate English translation.

The monotransitive phrase can still be parsed by this new rule since 吃 cannot have the part of speech *vnn*:[3]

$$[把[飯]_{NP}[吃]vn]_{VP}了.$$

But because 送 can be labeled as either *vn* or *vnn*, it does not match *vn/!vnn*, and therefore the rule cannot be applied to the ditransitive phrase. This leaves the ditransitive production

$$VP \rightarrow 把 \quad NP \quad vnn \quad NP$$

as the only possibility, forcing the correct sub-parse to be chosen here. In a sense, this function allows a measure of redundancy in the grammar specification and thereby improves robustness.

The substring-linking function

The *substring-linking* function is denoted **a/i**. This is used to remember the string that was matched to a constituent a, so that the string can be compared to a subsequent appearance of a/i in the same production. In general, we may have several occurrences of the same nonterminal, and it is occasionally useful to be able to constrain those occurrences to match exactly the same string.

One important use of substring-linking in Chinese is for reduplicative patterns. Another use can be seen in the following two sentences:

(4) a. 他 做 不 做 這 件 事
b. tā zuò bù zuò zhè jiàn shì
c. he do not do this — thing
d. will he do this thing

(5) a. 他 做 不 到 這 件 事
b. tā zuò bù dào zhè jiàn shì
c. he do not do this — thing
d. he cannot do this thing

Let us consider two sequences 做 不 做 and 做 不 到 in (4) and (5) respectively, where 做 and 到 can both be labeled as *vn*, but they have a different role. The former indicates a question, and the latter a negative declaration; clearly the parses must differentiate these two cases.

If the only rule in the grammar to handle these examples is

[3]The 了 character is an aspect particle.

$$question\_verb \rightarrow vn \text{ 不 } vn$$

then the two sequences will be parsed identically. However, with the substring-linking function we can refine the rule to

$$question\_verb \rightarrow vn/1 \text{ 不 } vn/1$$

Now the first *vn/1* is defined as 做 in both cases when the first 做 is parsed. For the first sequence, the second 做 matches the second *vn/1* when it is compared to the earlier-defined value of *vn/1*. Because the substrings match, the first sequence can be parsed by this rule as

$$[[做]vn不[做]vn]_{question\_verb}$$

In contrast, for the second sequence, when 到 is compared with the defined value of *vn/1* — 做 — they are different, and therefore the second sequence cannot be parsed by the rule.

In this example, the defined value of a nonterminal is only one word. However, in the general case it can be an arbitrarily long string of words spanned by a nonterminal (*vn1* in this example).

The semantic-domain function

The *semantic-domain* function is denoted by **c/&d** and designates a terminal c whose semantic domain is restricted to d. This is an ordinary feature, that we use in conjunction with the BDC dictionary which defines semantic domains.

Given two sentences,

(6) a. 在 廣東省 的 投資
b. zài guǎngdōngshěng de tóuzī
c. in Guangdong — investment
d. the investment in Guangdong Province

(7) a. 在 小張 的 家
b. zài xiǎozhāng de jiā
c. in XiaoZhang — house
d. in XiaoZhang's house

they have the same surface structure

$$在 \ NP \ 的 \ NP$$

but they are quite different. In (6), 在 廣東省 is the modifier of 投資. In (7), 小張 is a modifier of 家, and they together form a *NP* as the object of 在.

It is very hard to distinguish these two cases in general. With traditional CFGs, this is problematic because both 廣東省 and 小張 have the part of speech *np*, and both 投資 and 家 have part of speech *nc*. We can do a somewhat better job by using the domain knowledge supplied by a dictionary with semantic classes.

The difference between the two phrases is that although 廣東省 and 家 are both location nouns, not all *NP*s following a 在 can be formed into locative phrase—only if the head noun of the *NP* is a location noun can it can be parsed as a locative phrase. (6) is parsed as

$$[[[\text{在}[\text{廣東省}]_{NP}]_{LocPh}\text{的}]_{ModPh}[\text{投資}]_{NP}]_{NP}$$

because 在 廣東省 is a locative phrase, where LocPh stands for locative phrase, and ModPh stands for modifier phrase. But in (7), the entire phrase 在 小張 的 家 forms a locative phrase, and is parsed as

$$[\text{在}[[[\text{小張}]_{NP}\text{的}]_{ModPh}[\text{家}]_{NP}]_{NP}]_{LocPh}$$

The key point here is how to define a location noun. We have rules

$$location\_noun \rightarrow np/\&GE$$

and

$$location\_noun \rightarrow nc/\&GE$$

where GE is the abbreviation of geology. Because the domain of 廣東省 is GE, it is parsed as a *location_noun*, and together with the leader 在 is parsed as a locative phrase. But 小張 cannot be parsed as a locative phrase with the leader 在 since its domain is not GE; instead it is parsed as the modifier of 家, at which point the parser will further check whether 在 plus 小張 的 家 can be parsed as a locative phrase.

The has-subconstituent function

This function is denoted as **a/@b**, which means a constituent labeled *a* with any descendant of category *b*, where *a* is a nonterminal and *b* can be either a terminal or a nonterminal. In other words, this matches an internal node labeled *a*, which has a subtree with root labeled *b*.

Consider the two sentences

(8) a. 他 學 了 兩 個 星期
   b. tā xué le liǎng gè xīngqí
   c. he learn — two — week
   d. he has learned it for two weeks

(9) a. 他 學 了 兩 篇 課文
   b. tā xué le liǎng piān kèwén
   c. he learn — two — lesson
   d. he has learned two lessons

In Sentence (8), 兩 個 星期 is the complement of 學, while in Sentence (9), 兩 篇 課文 is the object of 學. However, both *NP*s 兩 個 星期 and 兩 篇 課文 superficially have the same structure, and the parser may assign Sentence 8 the wrong parse tree

$$[[\text{他}]_{NP}[[\text{學}]_{vn}\text{了}[[\text{兩 個}]_{ClPh}[\text{星期}]_{NP}]_{NP}]_{VP}]_{clause}$$

instead of the correct one

$$[[\text{他}]_{NP}[[\text{學}]_{vn}\text{了}[[[[\text{兩 個}]_{ClPh}[[\text{星期}]_{time\_particle}]_{NP}]_{NP}]_{TP}]_{Comp}]_{VP}]_{clause}$$

where $ClPh$ stands for classifier phrase, TP stands for time phrase, and Comp stands for the complement of a verb.

The difference between them lies in that 星期 is a time particle, and therefore is parsed with its classifier 兩 個 as a time phrase, whereas 課文 is a general noun, and is parsed with its classifier 兩 篇 as a general *NP*.

With the rule

$$time\_phrase \rightarrow NP/@time\_particle$$

we can parse 兩 個 星期 as a time phrase, and since it is a time phrase, it will be parsed as the complement of 學. But becase 兩 篇 課文 is a just general *NP*, it can not be parsed with this rule, and it will serve only as the object of 學.

Earley Parsing

We use a generalization of the Earley algorithm (3, 2) to parse grammars of our form. Although the time complexity rises compared to the Earley algorithm, it remains polynomial in the worst case.

Algorithm

The key to modifying the Earley algorithm to handle the left and right context conditions is that our rules can be rewritten into a full form which includes all symbols including the contexts, plus indices indicating the left and/or right context boundaries. For example, let $A \rightarrow \{L\}\ B\ \{R\}$ and $C \rightarrow D\ E\ \{R\}$ be two production rules. They are rewritten respectively as $A \rightarrow L\ \ B\ \ R$, start = 2, len = 1 and $C \rightarrow D\ \ E\ \ R$, start = 1, len = 2. Once this transformation has been made, the machinery from the Earley algorithm carries over remarkably smoothly.

The main loop of the parsing algorithm employs the following schema.

1. Pop the first entry from the agenda; call the popped entry c.
2. If c is already in chart, go to 1.
3. Add c to chart.
4. For all rules whose left corner is b, call *match(b, c)*. If the return value is 1, add an initial edge e for that rule to chart; for all the chart entries (subtrees) c' beginning at end(e)+1, if g is the active symbol in the RHS (right-hand-side) of e and *match(g, c')* returns 1, then call $extend(e, c')$.
5. If the edge e is finished, add an entry to the agenda.
6. For all edges d, if g is the active symbol in the RHS of d and *match(g, c)* returns 1, then call *extend(d, c)* and add the resulting edge.
7. Go to 1.

$extend(e, c)$: (extends an edge e with the chart entry (subtree) c)

1. Create a new edge e'.
2. Set start(e') to start(e).
3. Set end(e') to end(e).
4. Set rule(e') to rule(e) with $\cdot$ moved beyond c.
5. If the edge e' is finished (i.e., a subtree) then add e' to the agenda, else for all chart subtrees c' beginning at end(e')+1, if g is the active symbol in the RHS of e' and *match(g, c')* returns 1, call $extend(e', c')$.

$match(g, c)$: (checks whether a subtree c can be matched by a symbol g)

1. If c's category does not equal to g's category, return 0.
2. Check whether g's associated functions are satisfied by c —
   (a) If g has the form **a/!b** or **/!b**, check all the entries in the chart that span the same range as c, returning 0 if any have category b.
   (b) If g has the form **a/i**, if a/i is not defined, link it to c and return 1. Otherwise, compare c with the defined value of a/i; if not the same, return 0.
   (c) If g has the form **c/&d**, if the semantic domain of c is not d, return 0.
   (d) If g has the form **a/@b**, check all the nodes of the subtree c; if no node of category b is found, return 0.
3. Return 1.

The difference from standard Earley parsing (aside from the rule transformation mentioned above) lies is in *match*. To check whether an entry matches the left corner of a rule or whether an edge can be extended by an entry, we need to check not only that the category of the constituent is matched, but also that the attached function if any is satisfied.

Recall that our application for the parsing algorithm is as the first stage of a robust bracketer. We therefore use an extension of this parsing approach that permits partial parsing. In this version, if the sentence cannot be parsed, a minimum-size subset of subtrees that cover the entire sentence is produced.

In the following, we will use an example sentence to demonstrate how the algorithm works. The sentence and the grammar we use here are oversimplified, but show how a right context is handled.

The sentence to be parsed is

(10) a. 他 買 的 衣服
b. tā mǎi de yīfú
c. he buy – clothes
d. the clothes bought by him

and the grammar is

1. $NP \rightarrow\ \ pron$
2. $NP \rightarrow\ \ nc$
3. $RelPh \rightarrow\ \ NP\ \ vn$ 的 $\{NP\}$
4. $NP \rightarrow\ \ RelPh\ \ NP$

5. $pron \rightarrow$ 他

6. $nc \rightarrow$ 衣服

7. $vn \rightarrow$ 買

The first portion of the parsing for this example is identical to standard Earley parsing. We pop the first the entry from the agenda, 他 , and since it is not already there we add it to the chart. The only initial edge to be added is

pron → 他 ·

Since this edge is finished, we add it to the agenda.

Next we pop *pron* from agenda, create an initial edge

$NP \rightarrow \ pron \ \cdot$

and find it is also finished, and so add the *NP* to the agenda.

Again we pop *NP* from the agenda, and create the initial edge

$RelPh \rightarrow \ NP \ \cdot \ vn$ 的 $\{NP\}$

We find this edge cannot be extended by any entry and is not finished, so we go to step 1 and pop the next entry 買 from the agenda.

We continue this step until we pop 衣服 from the agenda, and add *nc* and later *NP* to the agenda. Up to this point, all we are doing is standard Earley parsing.

Now we pop *NP* which spans 衣服 from the agenda, and find that the edge

$RelPh \rightarrow \ NP \ vn$ 的 $\cdot \ \{NP\}$

can be extended by this entry. We find the extended edge is finished, so we add the *RelPh* to the agenda, then pop it, creating a new edge

$NP \rightarrow \ RelPh \ \cdot \ NP$

An entry (subtree) *NP* which spans 衣服 is already in the chart when the last edge is created. Thus the last edge can be extended, creating a finished edge, so we have created an subtree *NP* that spans the whole sentence. Since there is now a nonterminal that spans the whole sentence, we can write down a parse tree of the sentence in a subscripted bracket form as

$[[[[\text{他}]_{pron}]_{NP}[\text{買}]_{vn}\text{的}]_{RelPh}[[\text{衣服}]_{nc}]_{NP}]_{NP}$

We do not yet have a tight upper-bound for this parsing algorithm in the worst case. Clearly the algorithm will be more time consuming than for CFGs because the *match* procedure will need to check not only the categories of the constituents, but also their associated functions, and this check will not take constant time as for CFGs.

But though the algorithm is clearly worse than CFG in the worst case, in practice, the complexity in practice will depend heavily on particular sentences and the grammar. The number and type of context conditions used in the grammar, and the kind of nonterminal functions, will greatly affect the efficiency of parsing. Thus empirical performance is the true judge, and our experience as described next has been quite encouraging.

Results

We are currently developing a robust grammar of this form for the Chinese bracketing application. Although the number of rules is changing daily, the evaluation was performed on a version of the grammar containing 948 rules. The lexicon used was the BDC dictionary containing approximately 100,000 entries with 33 part of speech categories (1).

To evaluate our progress, we have evaluated precision on a previously unseen sample of 250 sentences drawn from our corpus, which contains Hong Kong legislative proceedings. The sentences were randomly selected in various length ranges of 4–10, 11–20, 21–30, 31–40, and 41–50 words, such that each of the five ranges contained 50 sentences. All those sentences were segmented by hand, though we will use an automatic segmenter in the future. We evaluated three factors:

1. The percentage of labeled words. A word is unlabeled if it can not form deeper structure with at least one other word. Unlabeled words often indicate inadequacies with lexicon coverage rather than the grammar.
2. *Weighted constituent precision*, i.e., the percentage of incorrectly identified syntactic constituents. A constituent is judged to be correct only if both its bracketing and its syntactic label are correct.

   Because we don't give a single parse tree if there is for a sentence at the current stage, we uniformly weight the precision over all the parse trees for the sentence. Therefore this measure is a kind of weighted precision (6).

0: (final (clause (clause (advph (sadv 不過) ,) (clause (nounph (nounph (noun (pron 我)) (noun (nc 希望)))) (verbph (zaiph 在 (nounph (modph (relph (nounph (noun (np 警務處))) (vppart (vn (vadv 即將) (vn 提供))) 的)) (nounph (modph (aa (vi1 詳盡))) (nounph (noun (nc 答覆))))) (locat_part 中))))) (punc ,) (clause (verbph (vn (auxvb (aux 會)) (vn 包括)) (nounph (assocph (nounph (d 這) (nounph (noun (nc 方面)))) 的) (nounph (noun (nc 資料))))))) 。)

0: (final (clause (clause (advph (sadv 不過) ,) (clause (nounph (nounph (noun (pron 我)) (noun (nc 希望)))) (verbph (zaiph 在 (nounph (modph (relph (nounph (noun (np 警務處))) (vppart (vn (vadv 即將) (vn 提供))) 的)) (nounph (modph (aa (vi1 詳盡))) (nounph (noun (nc 答覆))))) (locat_part 中))))) (punc ,) (clause (verbph (vn (auxvb (aux 會)) (vn 包括)) (nounph (assocph (nounph (d 這) (nounph (noun (nc 方面)))) 的) (nounph (noun (nc 資料))))))) 。)

0: (final (clause (clause (advph (sadv 不過) ,) (clause (nounph (d 任何) (nounph (noun (nc 人)))) (cjs 若) (verbph (vn (vadv 真正) (vn 有)) (nounph (noun (nc 困難)))))) (punc ,) (clause (verbph (verbph (vn (vadv 則) (vn (auxvb (aux 可以)) (vn 獲得))) (nounph (noun (nc 豁免)))) (verbph (vn 繳) (nounph (noun (nc 費))))))) 。)

0: (final (clause (nounph (noun (nc 市民))) (verbph (vs (vadv 又) (vs (vadv 是否) (vs 覺得))) (clause (clause (nounph (clph (d 這) (cl 個)) (nounph (noun (nc 制度)))) (verbph (verbph (vn (auxvb (aux 可)) (vn 用於)) (nounph (noun (np 立法局)))) (verbph (vi1 選舉)))) 呢))) ?)

(nounph (nounph (noun (nc 主席)) (noun (nc 先生)))) , (clause (nounph (pron 我)) (verbph (vi1 (neg 不) (vi1 知道)))) (verbph (covph (p 由) (nounph (pron 我))) (verbph (vn 回答) (nounph (clph (d 這) (cl 個)) (nounph (noun (nc 問題)))))) , (clause (verbph (vi1 (vadv 是否) (vi1 (vadv 最) (vi1 為 (vi1 適當)))))) 。

(nounph (nounph (noun (nc 主席)) (noun (nc 先生)))) , (clause (advph (sadv 最後)) (clause (nounph (pron 我)) (verbph (vv 要) (verbph (covph (p 藉) (nounph (modph (relph (nounph (d 此) (nounph (noun (nc 機會)))) (vppart (verbph (vn 感謝) (nounph (noun (np 專案小組)) (noun (nc 成員)))) (vn (vadv 所) (vn 作))) 的)) (nounph (modph (aa (a 一切))) (nounph (noun (nc 努力))))) (punc ,)) (verbph (vnv 使) (nounph (d 這) (nounph (nounph (noun (nc 條例)) (noun (nc 草案))))) (verbph (vi1 (vadv 妥善) (vi1 恰當)))))))) 。

(nounph (assocph (nounph (q 一些) (noun (nc 重要))) 的) (nounph (noun (nc 服務)))) , (advph (sadv 例如)) (nounph (nounph (nounph (noun (nc 精神)) (noun (nc 健康)))) (cjw 和) (nounph (nounph (noun (nc 老人)) (noun (nc 護理))))) , (clause (verbph (vn (vadv 就) (vn (auxvb (aux 未 能)) (vn 達到))) (nounph (assocph (nounph (nounph (nounph (noun (pron 我們)) (noun (nc 心目)))) (locat_part 中)) 的) (nounph (noun (nc 水準)))))) 。

0: (final (clause (clause (clause (nounph (q 一些) (noun (nc 意見書))) (verbph (vn 提出) (nounph (modph (relph (vppart (vn 涉及)) (nounph (noun (nc 條例)) (noun (nc 草案))) 的)) (nounph (nounph (nounph (noun (nc 法律)) (noun (nc 技術))) (noun (nc 問題))))))) (punc ,) (clause (verbph (vs 令) (clause (nounph (pron 我們)) (verbph (covph (p 對) (nounph (d 這) (nounph (noun (nc 方面))))) (verbph (vn (vadv 再) (vn 作)) (nounph (noun (nc 考慮))))))))) (punc ,) (clause (verbph (vi1 (vadv 實在) (vi1 (vadv 非常) (vi1 有用)))))) 。)

(advph (sadv 一般來說) ,) (clause (nounph (noun (np 這))) (verbph (vi2 可))) (verbph (covph (p 為) (nounph (modph (aa (a 本))) (nounph (nounph (noun (nc 港)) (noun (nc 經濟)))))) (verbph (vn 帶來) (nounph (modph (attrph (aa (vi1 龐大)) 的)) (nounph (noun (nc 利益)))))) , 即使 (nounph (d 這些) (nounph (noun (nc 利益)))) 不 (clause (nounph (noun (nc 能))) (verbph (covph (p 以) (nounph (noun (nc 數字)))) (verbph (vi1 衡量)))) 。

Figure 1: Examples of parse output (see text).

(clause (nounph (nounph (noun (nc 土地)) (noun (nc 發展))) (noun (nc 公司))) (verbph (vi2 (neg 不) (vi2 (auxvb (aux 能)) (vi2 犧牲))))) (nounph (assocph (nounph (nounph (noun (nc 大多數)) (noun (np 香港))) (noun (nc 人))) 的) (nounph (noun (nc 利益)))) , (clause (verbph (covph (p 以) (nounph (modph (relph (vppart (vn 高於)) (nounph (modph (aa (vi1 公平))) (nounph (noun (nc 合理)))) 的)) (nounph (noun (nc 價格))))) (verbph (vv 去) (verbph (vn 收購) (nounph (noun (nc 物業))))))) 的 。

(clause (clause (clause (nounph (noun (np 香港))) (verbph (is 是) (nounph (clph (q 一) (cl 個)) (nounph (modph (aa (vi1 大))) (nounph (noun (nc 都會))))))) (punc ,) (clause (verbph (vv 要) (verbph (vn (vadv 繼續) (vn 取得)) (nounph (modph (aa (vi1 卓越))) (nounph (noun (nc 成就)))))))) (punc ,) (clause (verbph (vi2 (vadv 就) (vi2 需要))))) (clause (nounph (nounph (noun (np 本身))) (cjw 及) (nounph (noun (np 國際間)))) (verbph (vn 有) (nounph (modph (relph (vppart (vn 穩定)) 的)) (nounph (nounph (noun (nc 政治)) (noun (nc 環境))))))) , (clause (nounph (modph (aa (a 國際))) (nounph (noun (nc 人士)))) (verbph (verbph (vi2 (vadv 亦) (vi2 (vadv 自然) (vi2 關注)))) (cjw 及) (verbph (vn (vadv 密切) (vn 注視)) (nounph (assocph (nounph (noun (nc 本地)) (noun (nc 事態))) 的) (nounph (noun (nc 發展))))))) 。

(clause (clause (clause (nounph (noun (np 本人))) (verbph (vnv 促請) (nounph (noun (nc 政府))) (verbph (vv 嘗試) (verbph (covph (p 從) (nounph (nounph (noun (nc 另一個)) (noun (nc 角度))))) (verbph (vv 去) (verbph (vn 探討) (nounph (noun (nc 問題))))))))) (punc ,) (clause (verbph (advph (sadv 即)) (verbph (vn 修改) (nounph (noun (nc 規例))))))) (punc ,) (clause (verbph (vnv 使) (nounph (noun (nc 法庭))) (verbph (covph (p 把) (nounph (noun (nc 被告)))) (verbph (vi2 定罪)))))) (nounph (noun (nc 後))) , (clause (nounph (noun (nc 才))) (verbph (vn 引用) (nounph (noun (nc 被告))))) (nounph (modph (attrph (aa (a 過往)) 的)) (nounph (modph (aa (vi1 詳細))) (nounph (noun (nc 紀錄))))) , (clause (verbph (verbph (vn 作為) (nounph (noun (nc 適當)))) (verbph (vn 判) (nounph (assocph (nounph (noun (nc 刑))) 的) (nounph (noun (nc 依據))))))) 。

(clause (cjs 由於) (clause (nounph (noun (np 這些))) (verbph (is (vadv 都) (is 是)) (nounph (modph (aa (vi1 正常))) (nounph (nounph (noun (nc 工務)) (noun (nc 計劃)))))))) 以外 的 (nounph (noun (nc 工程))) , (clause (nounph (modph (aa (vi1 (vadv 因此) (vi1 有)))) (nounph (noun (nc 需要)))) (verbph (locph (locph (zaiph 在 (nounph (modph (relph (vppart (vn (neg 不) (vn 影響))) (nounph (d 其他) (nounph (modph (aa (vi1 正常))) (nounph (nounph (noun (nc 工務)) (noun (nc 工程)))))) 的)) (nounph (noun (nc 情況)))) (locat_part 下))) (punc ,)) (verbph (covph (p 從) (nounph (d 其他) (nounph (noun (nc 地方))))) (verbph (vn 謀求) (nounph (modph (attrph (aa (a 額外)) 的)) (nounph (nounph (noun (nc 人力)) (noun (nc 資源))))))))) 。

(clause (clause (clause (clause (cjs 如果) (clause (nounph (noun (nc 當局))) (verbph (vv 繼續) (verbph (vn 實施) (nounph (modph (aa (a 現行))) (nounph (noun (nc 政策)))))))) (punc ,) (clause (nounph (modph (aa (vi1 (neg 不) (vi1 對)))) (nounph (noun (np 工業界)))) (verbph (verbph (vn 予以) (nounph (noun (nc 直接)))) (verbph (vi2 資助))))) (punc ,) (clause (verbph (vn (vadv 就) (vn (auxvb (aux 應)) (vn 建立))) (nounph (clph (q 一) (cl 個)) (nounph (modph (attrph (aa (vi1 良好)) 的)) (nounph (noun (nc 交通網)))))))) (punc ,) (clause (verbph (covph (p 為) (nounph (nounph (noun (nc 本地))) (cjw 及) (nounph (nounph (nounph (noun (nc 海外)) (noun (np 工業界))) (noun (nc 人士)))))) (verbph (vn 創造) (nounph (modph (attrph (aa (vi1 良好)) 的)) (nounph (noun (nc 投資)))))))) (nounph (noun (nc 環境))) , (clause (verbph (covph (p 讓) (nounph (noun (np 香港)))) (verbph (vv (auxvb (aux 可以)) (vv 保持)) (verbph (vi2 競爭))))) (nounph (noun (nc 能力))) 。

Figure 2: Examples of parse output (cont'd).

length of sentence	4-10	11-20	21-30	31-40	41-50
% words labeled	83.10	99.61	95.67	94.82	95.45
% correct constituents	85.41	83.57	81.23	80.20	78.85
run time per sentence (secs.)	2.03	3.54	9.00	5.08	37.50

Table 1: Evaluation results.

In the future, we will give a single most probable parse tree for a sentence if it can be parsed. Note that the precision in this case is likely to be lower bounded by the weighted precision reported here, since we currently assign equal weight to all parses, even if they are improbable.

3. The average run time per sentence.

Results are shown in Table 1. We have unfortunately found it impossible to perform comparison evaluations against other systems, due to the unavailability of Chinese parsers in general. However, we believe these performance levels to be quite competitive and promising.

Meaningful baseline evaluations are currently difficult to design for Chinese parsing because of the unavailability of comparison standards. Examples of the Chinese output still give by far the most important indication of parsing quality. Some representative examples are shown in Figures 2 and 2. The parser produces two kinds of outputs. If no complete parse tree is found for the input sentence, a partial parse is returned; such examples are shown without a number preceding the parse. Otherwise, the first complete parse tree is shown, preceded by the number 0 (indicating that it was the first alternative produced).

Conclusion

We have described an extension to context-free grammars that admits a practical parsing algorithm. We have found the notation and the increased expressiveness to be well-suited for writing large robust grammars for Chinese, particularly for handling compounding phenomena without incurring the level of parsing ambiguity common to pure context-free grammars. Experiments show promising performance on Chinese sentences.

With regard to the theme of this conference, we are clearly emphasizing representation over algorithms. We have developed a new representation that neatly captures the domain characteristics, and in our experience, greatly improves the coverage and accuracy of our bracketer. Algorithms follow naturally as a consequence of the representational features. It will be interesting to explore the relationships between our grammar and other context-sensitive grammar formalisms, a topic we are currently pursuing.

References

[1] BDC. *The BDC Chinese-English Electronic Dictionary (version 2.0).* Behavior Design Corporation, 1992.

[2] Eugene Charniak. *Statistical Language Learning.* MIT Press, Cambridge, MA, 1993.

[3] Jay Earley. An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery*, 13(2):94–102, 1970.

[4] Dekai Wu. An algorithm for simultaneously bracketing parallel texts by aligning words. In *Proceedings of the 33rd Annual Conference of the Association for Computational Linguistics*, pages 244–251, Cambridge, Massachusetts, June 1995.

[5] Dekai Wu. Trainable coarse bilingual grammars for parallel text bracketing. In *Proceedings of the Third Annual Workshop on Very Large Corpora*, pages 69–81, Cambridge, Massachusetts, June 1995.

[6] Dekai Wu and Xuanyin Xia. Large-scale automatic extraction of an English-Chinese lexicon. *Machine Translation*, 9(3–4):285–313, 1995.