

Towards GP Sentence Parsing of V+P+CP/NP Structure

A Perspective of Computational Linguistics

Du Jiali

School of Foreign Studies
Nanjing University
Nanjing, China

Yu Pingfang

Faculty of Chinese Language and Culture
Guangdong University of Foreign Studies
Guangzhou, China

Abstract—Computational linguistics can provide an effective perspective to explain the partial ambiguity during machine translation. The structure of V+Pron+CP/NP has the ambiguous potential to bring Garden Path effect. If Tell+Pron+NP structure has considerable higher observed frequencies than Tell+Pron+CP structure, the former is regarded as the preferred structure and has much lower confusion quotient. It is possible for the grammatical unpreferred Tell+Pron+CP structure to replace the ungrammatical preferred Tell+Pron+NP, which results in the processing breakdown. The syntactic details of GP processing can be presented by the computational technologies. Computational linguistics is proved to be effective to explore the Garden Path phenomenon.

Keywords—artificial intelligence; computational linguistics; machine learning; local ambiguity; garden path sentences

I. INTRODUCTION

A Garden Path (GP) sentence can produce an effect of processing breakdown. It refers to the special local ambiguous structure which always makes a reader misled down to the garden path. The GP sentence is grammatically correct and the ultimate result of processing is different from the initial incorrect interpretation which is considered to be the most likely one at first. The ups and downs of processing may lure the reader into a dead one. With the appearance of garden path effect, the initially built-up structure is replaced with the improved structure whose processing is involved in the breakdown and backtracking. Connectionist psycholinguistics focus on psychology of language, and connectionist models are considered to play an important role in GP processing.[1]

The parsing of GP sentences is complex during the processing of natural language. Information-based decision-making provides a helpful information method to analyze the complex structure, i.e. GP sentence, by describing the phenomenon of information use and by explaining why an information use occurs as it does. The relative effectiveness can be obtained by using multi-term phrases and POS tagged terms. [2] Knowledge Base System is effective for parsing of GP sentence. Large-scale discriminative two-phase learning algorithms, which ensure the reliable estimation and prevent overfitting, can be used to learn parameters in models with extremely sparse features.[3]Lexical information and knowledge representation can improve the efficiency of parsing. The effective device for imposing structure on lexical

information is that of inheritance, both at the object (lexical items) and meta (lexical concepts) levels of lexicon. Lexical semantics theory can utilize a knowledge representation framework to offer a richer, more expressive vocabulary for lexical information, which obviously brings the advancement of knowledge base system.[4]Writing strategy training can be meaningfully provided by artificial intelligence tutoring system which is effective in assessing essay quality and guiding feedback to students with the help of processing algorithms,the development of which must consider a broad array of linguistic, rhetorical, and contextual features.[5]

Machine translation systems need the effectively parsing of GP sentences. MT systems against semantic frame based MT evaluation metrics and objective function can benefit from the semantic knowledge and produce more adequate output. Machine translation evaluation systems in the metrics task can be used to measure the similarity of the system output translations and the reference translations on word sequences.

Statistical machine translation (SMT) performance can be affected by the small amounts of parallel data with the result of both low accuracy (incorrect translations and feature scores) and low coverage (high out-of-vocabulary rates). The bilingual lexicon induction techniques are helpful for learning new translations from comparable corpora, thus improving the coverage. The model's feature space with translation scores can be estimated over comparable corpora, which brings the improvement of accuracy. [6]Extant Statistical Machine Translation systems embed multiple layers of heuristics and encompass very large numbers of numerical parameters. The phrase-based translation model can be used to decrease the difficulty of analyzing output translations and the various causes of errors.[7]

Based on natural language processing systems, computational linguistics skills and cognitive analysis, related system can be established to efficiently parse GP sentences.

This discussion comprises two parts. The first part will discuss the Shon and Moon's system for machine learning which is helpful for GP sentence parsing. The second part will parse the GP sentence from multi-perspectives. Context-free grammar, recursive transition network, CYK algorithm, and statistical analysis will be introduced to analyze the processing breakdown of GP sentence.

II. THE SYSTEM FOR MACHINE LEARNING

The peculiarities of linguistic cognition are necessary to be analyzed in the machine learning system. ML should have the ability to process the linguistic data originating from natural languages. For example, the evidence shows that it is hard for children to recover from misinterpretations of temporarily ambiguous phrases. They are reluctant to use late-arriving syntactic evidence to override earlier verb-based cues to structure, and late-developing cognitive control abilities mediate the recovery from GP sentences.[8] World-knowledge about actions designated by verbs and syntactic proficiency is reflected in on-line processing of sentence structure.[9]

Automated knowledge acquisition is an essential aspect involved in machine learning. For example, the automated induction of models and the extraction of interesting patterns from empirical data are concerned in the ML domain. Fuzzy set theory is proved to be helpful for machine learning, data mining, and related fields.[10]

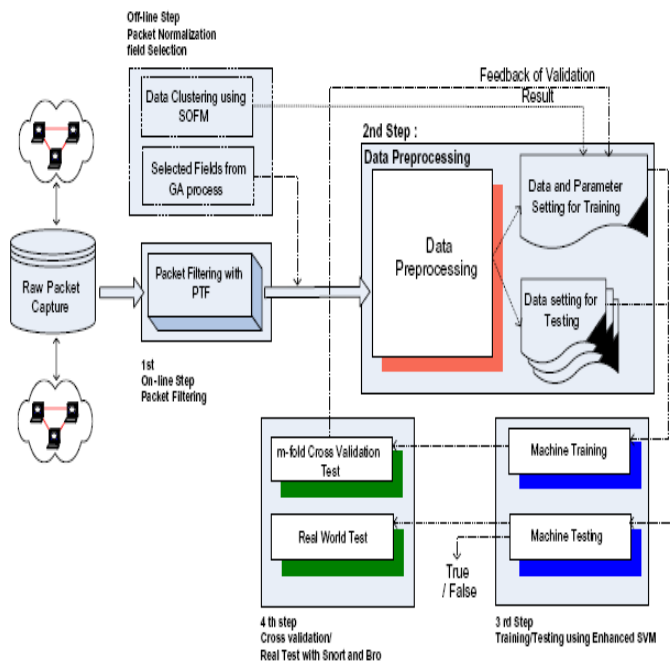


Fig. 1. A Model of Machine Learning Framework

In Fig.1 created by Shon and Moon [11], we can see the system comprises many steps, i.e. on-line processing, off-line processing, feedback of validation, the Enhanced SVM machine learning approaches, cross validation.

On-line processing is the first step, which focuses on a real-time traffic filtering using PTF after the parsing of raw packet capture. PTF is used to drop malformed packets, which can provide efficient work for packet preprocessing and reduce the number of potential attacks of the raw traffic. Off-line processing is involved in data clustering using SOFM and a packet field selection using GA. Both procedures work comfortably before the overall framework. GA chooses the appropriate fields and the filtered packets in real-time select the related fields by the natural evolutionary process. SOFM-based

packet clustering can make a lot of packet profiles for SVM learning which can bring more appropriate training data.

In the feedback of validation, the filtered-packets are used as the preprocessing of high detection performance. The relationships between the packets are considered to charge SVM inputs with temporal characteristics. The SVM comprises soft margin SVM (supervised method) and one-class SVM (unsupervised method). The Enhanced SVM machine learning approach, including machine training and testing, inherits the high performance of soft margin SVM and the novelty detection capability of one-class SVM. Cross validation and real test with Snort and Bro is the final step which entails an m-fold cross validation test and real world test.

Shon and Moon's model provides us a general idea about machine learning. An effective system has to be a flexible and adaptable one. That means any effective system should be the product of theoretical and practical application. The improvement of theoretical analysis can improve the system effectively. Therefore, some good algorithms and parsing skills should be involved in improving machine learning system.

From the perspective of theoretical analysis, machine learning depends on the integration of computational technologies and linguistic background. The harmonious development of linguistics and computer science can bring the advancement of machine learning. The parsing of complex sentences, i.e. garden path sentences, will be clearly shown below to present the procedures of machine learning and linguistic cognition. The integration of context-free grammar, recursive transition network, well-formed substring table and CYK algorithm can be used as an effective method of computational linguistics for application.

III. MULTI-PARSING OF GP SENTENCE

The processing of GP sentences needs the effective involvement of formal language, e.g. context free grammar, and the algorithms, e.g. recursive transition network and CYK algorithm.

A. CFG-Based Processing of GP Sentence

Context free grammar generally comprises nonterminal symbol (V) and terminal symbol (w), and all the production rules belong to the structure of "V→w". Sometimes, "w" can refer to the nonterminal strings or empty strings. The situation in which the production rules are available regardless of the context of a nonterminal is called context free. The single "V" can be replaced by "w" according to the production rules until nearly all the rules are used for the processing. If the ultimate nonterminal symbol "S" can be replaced by the terminal symbols on the right and all the strings have been parsed successfully, the processing is acceptable. If part of the strings are left by system without being parsed, processing has to return to the crossroad to choose the alternative which can lead to the full parsing of all the strings. The backtracking obviously brings the processing breakdown which is the particular effect of GP sentence.

Example 1: She told me a little white lie will come back to haunt me.

She told me a little white lie will come back to haunt me.
 G={Vn, Vt, S, P}
 Vn={S, NP, VP, N, V, Pron, Det, Adj, IP, Aux, Adv, SC}
 Vt={she, told, me, a little, white, lie, will, come, back, to, haunt}
 S=S

- P:
- a. S→NP VP
 - b. NP→N
 - c. VP→V NP NP
 - d. NP→Pron
 - e. NP→Det NP
 - f. NP→Adj N
 - g. VP→V NP IP
 - h. VP→Aux V
 - i. VP→VP Adv
 - j. VP→VP IP
 - k. IP→NP VP
 - l. VP→VP SC
 - m. SC→Aux VP
 - n. VP→V NP
 - o. Det→{a little}
 - p. Pron→{she, me}
 - q. N→{lie}
 - r. V→{told, haunt, come}
 - s. Adj→{white}
 - t. Adv→{back}
 - u. Aux→{will, to}

According to the CFG above, we can find the grammar is defined by the four-parameters “Vn, Vt, S, P” and sub-language defined by “G”. “Vn” means a finite set of non-terminal symbols; “Vt”, a finite set of terminal symbols; “S”, start symbol; “P”, productions of grammar which shows the relationships between non-terminal and terminal symbols. All the rules are available during the processing. Please see the parsing by means of the grammar.

She told me a little white lie will come back to haunt me	Rules
Pron told me a little white lie will come back to haunt me	p
NP told me a little white lie will come back to haunt me	d
NP V me a little white lie will come back to haunt me	r
NP V Prop a little white lie will come back to haunt me	p
NP V NP a little white lie will come back to haunt me	d
NP V NP Det white lie will come back to haunt me	o
NP V NP Det Adj lie will come back to haunt me	s
NP V NP Det Adj N will come back to haunt me	q
NP V NP Det NP will come back to haunt me	f
NP V NP NP will come back to haunt me	e
NP VP will come back to haunt me	c
S will come back to haunt me	a
?	

BREAKDOWN AND BACKTRACKING

TABLE I. THE STATISTICAL ANALYSIS OF “TELL+ME” TYPE

Type	Observed	Expected	Deviation	D2	D2/E
IP	24	136	-112	12544	92.24
NP	148	136	12	144	1.06
Total	272	272			93.30

In the BNCweb database (<http://bncweb.lancs.ac.uk/>), we can obtain the statistical information about “tell+me+any article” type in which both “[that) any article+NP]IP” and

“[any article+NP]NP” are acceptable. The corpus shows that “Your query ‘tell me’ returned 5284 hits in 1213 different texts (98,313,429 words [4,048 texts]; frequency: 53.75 instances per million words), sorted on position +1 with tag-restriction any article (272 hits)”. According to the corpus, the statistical number of “[that) any article+NP]IP” type is 24, and the other NP type is 148.

According to the nonparametric statistics, if df=1 and $X^2=93.30$, then $p<.05$, which means the significant differences between the types. Since the type of NP has much higher observed frequency than the other, NP type is considered to be the prototype by means of which system automatically parses the strings involved. If the prototypical NP type is rejected by system during the processing, the optional IP type will be started, resulting in the GP effect. Please see the successful processing in which IP type is accepted successfully.

NP V NP	NP will come back to haunt me	e
NP V NP	NP Aux come back to haunt me	u
NP V NP	NP AuxV back to haunt me	r
NP V NP	NP VP back to haunt me	h
NP V NP	NP VP Adv to haunt me	t
NP V NP	NP VP to haunt me	i
NP V NP	NP VP Aux haunt me	u
NP V NP	NP VP AuxV me	r
NP V NP	NP VP AuxV Prop	p
NP V NP	NP VP AuxV NP	d
NP V NP	NP VP AuxVP	n
NP V NP	NP VP SC	m
NP V NP	NP VP	l
NP V NP	IP	k
NP VP		g
S		a

SUCCESS

In the diagram, we can find the fact that “a little white lie” is considered NP used as the subject of IP, by which the GP sentence is successfully parsed. Besides the CFG and the tree diagram, Recursive Transition Network (RTN) is helpful for the parsing and processing of GP sentence.

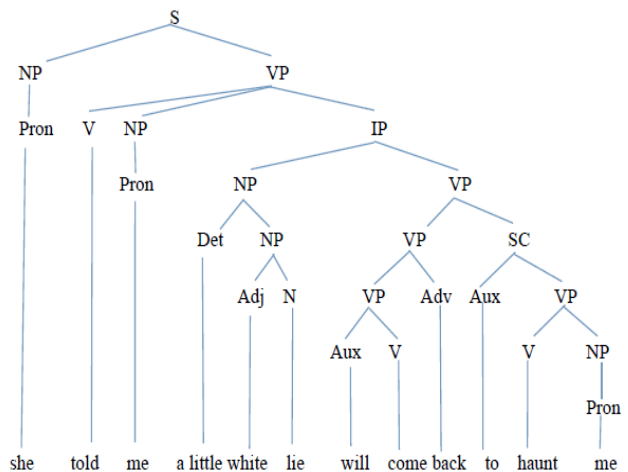


Fig. 2. Tree diagram of example 1

B. RTN-Based Processing of GP Sentence

An RTN usually comprises one S net and some subnets created for the recursive transition of different sub-strings. In

Example 1, the local ambiguity potential of structure asks the network effective to parse the sentence. That means both NP subnet and VP subnet should have convincing description of what function “a little white lie” should be, or can be. Please see RTN of Example 1.

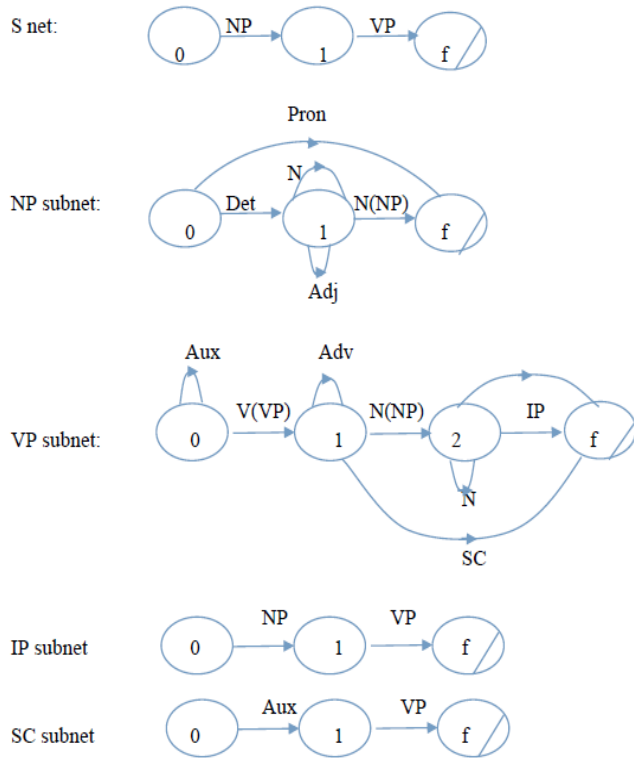


Fig. 3. The RTN of example 1

In Fig. 3, the NP subnet and VP subnet are complexer than the other nets. According to the Table 1, the structure of “She told me a little white lie” is parsed as follows.

```
(ROOT
(S
(NP (PRP She))
(VP (VBD told)
(S
(NP (PRP me))
(NP (DT a) (JJ little) (JJ white) (NN lie))))))
```

The processing above correspondingly brings the backtracking. Please see the details of breakdown processed by means of RTN algorithm.

In the processing, we can see that not all sub-strings are parsed successfully, which means the failure of processing. According to Figure 3, both the structures of “[she]NP [[told]VP [me]NP [a little white lie]NP]VP]S...” and “[she]NP [[told]V [me]NP [a little white lie ...]IP]VP]S” are partially acceptable during the processing, thus resulting in the local ambiguity at the point of “a little white lie”.

```
She told me a little white lie will come back to haunt me
<S/0, She told me a little white lie will come back to haunt me, >
<NP/0, She told me a little white lie will come back to haunt me, S/1: >
<NP/f, told me a little white lie will come back to haunt me, S/1: >
<VP/0, told me a little white lie will come back to haunt me, S/f: >
<VP/1, me a little white lie will come back to haunt me, S/f: >
<NP/0, me a little white lie will come back to haunt me, VP/1: S/f: >
<NP/1, a little white lie will come back to haunt me, VP/1: S/f: >
<NP/f, a little white lie will come back to haunt me, VP/1: S/f: >
<NP/0, a little white lie will come back to haunt me, VP/2: S/f: >
<NP/1, white lie will come back to haunt me, VP/2: S/f: >
<NP/f, lie will come back to haunt me, VP/2: S/f: >
<NP/f, will come back to haunt me, VP/2: S/f: >
<VP/f, will come back to haunt me, S/f: >
<S/f, will come back to haunt me, >
<, will come back to haunt me, >
?
```

BREAKDOWN AND BACKTRACKING

Since the first structure with high statistical frequency is proved to be illegal above, the alternative structure is started subsequently. The shift obviously brings the cognitive overburden and effect of garden path. The effective processing of unpreferred structure is shown as follows.

```
<NP/f, a little white lie will come back to haunt me, VP/1: S/f: >
<NP/f, a little white lie will come back to haunt me, VP/1: S/f: >
<IP/0, a little white lie will come back to haunt me, VP/2: S/f: >
<IP/1, a little white lie will come back to haunt me, VP/2: S/f: >
<NP/0, a little white lie will come back to haunt me, IP/1: VP/2: S/f: >
<NP/1, white lie will come back to haunt me, IP/1: VP/2: S/f: >
<NP/1, lie will come back to haunt me, IP/1: VP/2: S/f: >
<NP/f, will come back to haunt me, IP/1: VP/2: S/f: >
<VP/0, will come back to haunt me, IP/f: VP/2: S/f: >
<VP/1, come back to haunt me, IP/f: VP/2: S/f: >
<VP/1, back to haunt me, IP/f: VP/2: S/f: >
<VP/2, to haunt me, IP/f: VP/2: S/f: >
<SC/0, to haunt me, VP/f: IP/f: VP/2: S/f: >
<SC/1, haunt me, VP/f: IP/f: VP/2: S/f: >
<SC/1, haunt me, VP/f: IP/f: VP/2: S/f: >
<VP/0, haunt me, SC/f: VP/f: IP/f: VP/2: S/f: >
<VP/1, me, SC/f: VP/f: IP/f: VP/2: S/f: >
<NP/0, me, VP/2: SC/f: VP/f: IP/f: VP/2: S/f: >
<NP/f, , VP/2: SC/f: VP/f: IP/f: VP/2: S/f: >
<VP/f, ,SC/f: VP/f: IP/f: VP/2: S/f: >
<SC/f, ,VP/f: IP/f: VP/2: S/f: >
<VP/f, ,IP/f: VP/2: S/f: >
<IP/f, ,VP/2: S/f: >
<VP/f, , S/f: >
<S/f, , >
<, , >
SUCCESS
```

In the processing above, all the sub-strings are included and parsed comprehensively. The procedures reflect the fact that the GP sentence is grammatical correct despite of processing breakdown and backtracking. More details of the decoding algorithm can be analyzed in CYK.

C. CYK-Based Processing of GP Sentence

CYK algorithm is effective for the processing of GP sentence. It can clearly show the procedures of backtracking and present the possibility that computational linguistic knowledge functions as an efficient method providing the helpful hints to alleviate the effect of garden path.

In Table 2, we can see all the procedure of processing breakdown. At the point (7, 6), the CFG rule of “NP->Adj N” is started and the sub-string “white lie” is parsed. At the point (7, 3), another CFG rule of “NP->Det NP” is accepted and the sub-string “a little white lie” is rewritten into another NP. At the point(7, 1), the rules of “VP->V NP NP” and “NP->Pron” are available. The sub-string of “told me a little white lie” is parsed as VP. At the point(7, 0), the rules of “NP->Pron” and “S->NP VP” are provided. The sub-strings of “she told me a little white lie” are processed clearly. The original processing result is obtained. With the appearance of other sub-strings of “will come back to haunt me” which needs a NP as a SUBJECT, system has to re-analyze the original result. Thus, GP effect comes into being. Please see the non-well-formed sub-string table which is created on the basis of Table 2.

TABLE II. THE BREAKDOWN MATRIX OF EXAMPLE 1

	1	2	3	4	5	6	7	8	9	10	11	12	13
0	(P)	()	()	()	()	()	(S)	()	()	()	()	()	(?)
1		(V)	()	()	()	(VP)	()	()	()	()	()	()	()
2			(P)	()	()	()	()	()	()	()	()	()	()
3				(D)	(D)	()	(NP)	()	()	()	()	()	()
4					(A)	()	()	()	()	()	()	()	()
5						(A)	(NP)	()	()	()	()	()	()
6							(N)	()	()	()	()	()	()
7								(A)	(VP)	(VP)	()	()	(VP)
8									(V)	()	()	()	()
9										(A)	()	()	()
10											(A)	()	(SC)
11												(V)	(VP)
12													(P)

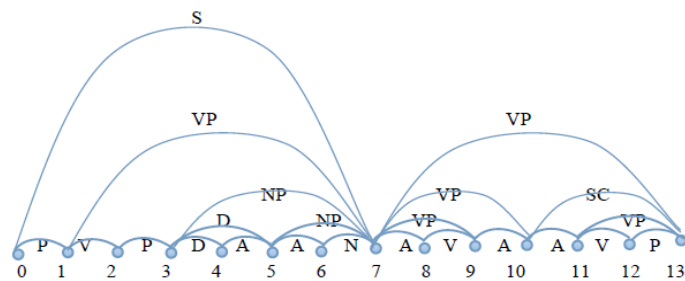


Fig. 4. Non-well-formed sub-string table of breakdown

In Fig. 4, the turning point is 7 in which the processing is cut into S+VP. The result is as same as the situation in Table 2 in which S is obtained at the point(7, 0) and VP is got at the point(13, 7). Since no effective CFG rule used to rewrite S+VP, the sub-string table is not well-formed, meaning the failure of processing. If system can backtrack to the turning point and apply the rule “VP->V NP IP” rather than “VP->V NP NP”, the whole processing procedures can be attractively presented in Table 3.

TABLE III. THE PROCESSING MATRIX OF EXAMPLE 1

	1	2	3	4	5	6	7	8	9	10	11	12	13
0	(P)	()	()	()	()	()	()	()	()	()	()	()	(S)
1		(V)	()	()	()	()	()	()	()	()	()	()	(VP)
2			(P)	()	()	()	()	()	()	()	()	()	()
3				(D)	(D)	()	(NP)	()	()	()	()	()	(IP)
4					(A)	()	()	()	()	()	()	()	()
5						(A)	(NP)	()	()	()	()	()	()
6							(N)	()	()	()	()	()	()
7								(A)	(VP)	(VP)	()	()	(VP)
8									(V)	()	()	()	()
9										(A)	()	()	()
10											(A)	()	(SC)
11												(V)	(VP)
													(P)

In Table 3, the CFG rules, e.g. “S->P(NP) VP”, “VP->V P(N) IP”, “IP->NP VP”, “VP->VP SC” etc. are available and the sentence can be effectively parsed. The well-formed sub-string table created from Table 3 can be shown in the Figure 5. The procedures of parsing reflect the fact that all the strings are successfully processed, and correspondingly, the algorithm used for the system to improve the efficiency can be presented as follows.

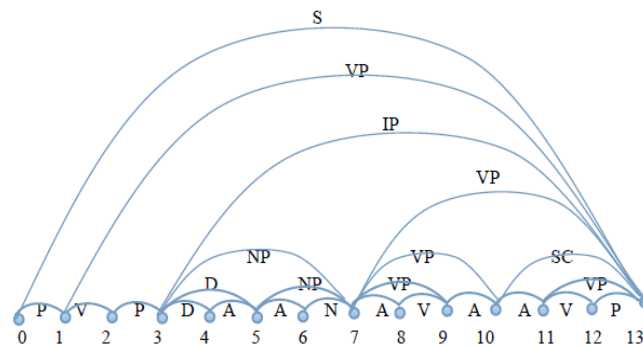


Fig. 5. Well-formed sub-string table of parsing

In the algorithm, two parameters (i & j) are provided to present the procedures. The parameter j is used to construct the chart structure and the parameter i is used to built the grammatical structure.

The first step is to construct the chart structure. The sentence has the string length of 13, i.e. 13 words are involved

in the processing. The duration of j is 1-13. Then the first word “she” can be drawn by the chart (0, 1); the second word “told”, by the chart(1,2); the third word “me”, by the chart(2,3);...the last word, by the chart(12, 13).

```

n:=13
for j: =1 to string length(13)
    lexical_chart_fill(j-1, j)
    for i: j-2 down to 0
        syntactic_chart_fill(i, j)
Fill the field (j-1, j) in the chart with the word j which belongs to the preterminal
category.
chart(j-1, j):={X | X→ wordj ∈ P}
j-1=0, j=1, chart(0, 1):={She}
j-1=1, j=2, chart(1, 2):={told}
j-1=2, j=3, chart(2, 3):={me}
j-1=3, j=4, chart(3, 4):={a}
j-1=4, j=5, chart(4, 5):={little}
j-1=5, j=6, chart(5, 6):={white}
j-1=6, j=7, chart(6, 7):={lie}
j-1=7, j=8, chart(7, 8):={will}
j-1=8, j=9, chart(8, 9):={come}
j-1=9, j=10, chart(9, 10):={back}
j-1=10, j=11, chart(10, 11):={to}
j-1=11, j=12, chart(11, 12):={haunt}
j-1=12, j=13, chart(12, 13):={me}

```

The second step is the construction of syntactic grammar. Since at least two basic charts can constitute one basic syntactic sector, the value duration of i is $j-2 \geq 0$. Thus, the created syntactic rules of symbols cover the string from i to j , and the algorithm with i and j can be obtained. The basic construction is “syntactic_chart_fill(i, j)”. Please see the algorithm presentation.

In the algorithm, the parameter k input between i and j is introduced to show the dynamic forms of analysis ($k:=i+1$ to $j-1$). With the help of i, j , and k , the processing algorithm can be smoothly run, and all the processing procedures based on well-formed sub-string table can be expressed clearly.

```

for i: =0 to 13
    chart(i, j)= {
        A | A→BC ∈ P
           i<k <j
           B ∈ chart(i, k)
           C ∈ chart(k, j)
    }
    chart(i, j):={}
    for k:= i+1 to j-1
        for every A→ BC ∈ P
            if B ∈ chart(i, k) and C ∈ chart(k, j) then
                chart(i, j):=chart(i, j) ∪ {A}
    If S ∈ chart(0,n) then accept else reject.
The processing procedures are shown below.
chart(j-1, j):={X | X→ wordj ∈ P}
j: =1 to string length
i: = j-2 down to 0
k:= i+1 to j-1

```

Since the procedures are complex and long, we choose one of the processing procedures ($j=13, 11 \geq i \geq 0, 12 \geq k \geq 1$) to show how the algorithm works to parse the sentence.

```

j=13, chart(12, 13):={me}
j=13, i=0, k=1, chart(0,1) ∪ chart(1, 13):={P} ∪ {VP}={S}
j=13, i=0, k=2, chart(0, 2) ∪ chart(2, 13):={} ∪ {}={}
j=13, i=0, k=3, chart(0,3) ∪ chart(3,13 ):={} ∪ {IP}={}
j=13, i=0, k=4, chart(0,4) ∪ chart(4, 13):={} ∪ {}={}
j=13, i=0, k=5, chart(0, 5) ∪ chart(5, 13):={} ∪ {}={}
j=13, i=0, k=6, chart(0,6) ∪ chart(6, 13):={} ∪ {}={}
j=13, i=0, k=7, chart(0, 7) ∪ chart(7, 13):={} ∪ {VP}={}
j=13, i=0, k=8, chart(0, 8) ∪ chart(8, 13):={} ∪ {}={}
j=13, i=0, k=9, chart(0, 9) ∪ chart(9,13 ):={} ∪ {}={}
j=13, i=0, k=10, chart(0,10) ∪ chart(10, 13):={} ∪ {SC}={}
j=13, i=0, k=11, chart(0, 11) ∪ chart(11, 13):={} ∪ {VP}={}
j=13, i=0, k=12, chart(0, 12) ∪ chart(12, 13):={} ∪ {P}={}
j=13, i=1, k=2, chart(1, 2) ∪ chart(2, 13):={V} ∪ {}={}
j=13, i=1, k=3, chart(1,3) ∪ chart(3, 13):={VP} ∪ {IP}={VP}
j=13, i=1, k=4, chart(1, 4) ∪ chart(4, 13):={} ∪ {}={}
j=13, i=1, k=5, chart(1, 5) ∪ chart(5, 13):={} ∪ {}={}
j=13, i=1, k=6, chart(1, 6) ∪ chart(6, 13):={} ∪ {}={}
j=13, i=1, k=7, chart(1, 7) ∪ chart(7,13 ):={} ∪ {VP}={}
j=13, i=1, k=8, chart(1, 8) ∪ chart(8, 13):={} ∪ {}={}
j=13, i=1, k=9, chart(1,9) ∪ chart(9, 13):={} ∪ {}={}
j=13, i=1, k=10, chart(1, 10) ∪ chart(10, 13):={} ∪ {SC}={}
j=13, i=1, k=11, chart(1, 11) ∪ chart(11, 13):={} ∪ {VP}={}
j=13, i=1, k=12, chart(1, 12) ∪ chart(12, 13):={} ∪ {P}={}
j=13, i=2, k=3, chart(2,3) ∪ chart(3, 13):={P} ∪ {}={}
j=13, i=2, k=4, chart(2,4) ∪ chart(4, 13):={} ∪ {}={}
j=13, i=2, k=5, chart(2,5) ∪ chart(5, 13):={} ∪ {}={}
j=13, i=2, k=6, chart(2, 6) ∪ chart(6, 13):={} ∪ {}={}
j=13, i=2, k=7, chart(2, 7) ∪ chart(7, 13):={} ∪ {VP}={}
j=13, i=2, k=8, chart(2, 8) ∪ chart(8, 13):={} ∪ {}={}
j=13, i=2, k=9, chart(2, 9) ∪ chart(9, 13):={} ∪ {}={}
j=13, i=2, k=10, chart(2, 10) ∪ chart(10,13 ):={} ∪ {SC}={}
j=13, i=2, k=11, chart(2, 11) ∪ chart(11, 13):={} ∪ {VP}={}
j=13, i=2, k=12, chart(2, 12) ∪ chart(12, 13):={} ∪ {P}={}
j=13, i=3, k=4, chart(3, 4) ∪ chart(4, 13):={D} ∪ {}={}
j=13, i=3, k=5, chart(3, 5) ∪ chart(5,13 ):=={D} ∪ {}={}
j=13, i=3, k=6, chart(3, 6) ∪ chart(6, 13):={} ∪ {}={}
j=13, i=3, k=7, chart(3, 7) ∪ chart(7, 13):={NP} ∪ {VP}={IP}
j=13, i=3, k=8, chart(3, 8) ∪ chart(8, 13):={} ∪ {}={}
j=13, i=3, k=9, chart(3,9) ∪ chart(9, 13):={} ∪ {}={}
j=13, i=3, k=10, chart(3, 10) ∪ chart(10, 13):={} ∪ {SC}={}
j=13, i=3, k=11, chart(3, 11) ∪ chart(11, 13):={} ∪ {VP}={}

```

- j=13, i=3, k=12, chart(3, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=4, k=5, chart(4, 5) ∪ chart(5, 13) := {A} ∪ {} = {}
 - j=13, i=4, k=6, chart(4, 6) ∪ chart(6, 13) := {} ∪ {} = {}
 - j=13, i=4, k=7, chart(4, 7) ∪ chart(7, 13) := {} ∪ {VP} = {}
 - j=13, i=4, k=8, chart(4, 8) ∪ chart(8, 13) := {} ∪ {} = {}
 - j=13, i=4, k=9, chart(4, 9) ∪ chart(9, 13) := {} ∪ {} = {}
 - j=13, i=4, k=10, chart(4, 10) ∪ chart(10, 13) := {} ∪ {SC} = {}
 - j=13, i=4, k=11, chart(4, 11) ∪ chart(11, 13) := {} ∪ {VP} = {}
 - j=13, i=4, k=12, chart(4, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=5, k=6, chart(5, 6) ∪ chart(6, 13) := {A} ∪ {} = {}
 - j=13, i=5, k=7, chart(5, 7) ∪ chart(7, 13) := {NP} ∪ {VP} = {}
 - j=13, i=5, k=8, chart(5, 8) ∪ chart(8, 13) := {} ∪ {} = {}
 - j=13, i=5, k=9, chart(5, 9) ∪ chart(9, 13) := {} ∪ {} = {}
 - j=13, i=5, k=10, chart(5, 10) ∪ chart(10, 13) := {} ∪ {SC} = {}
 - j=13, i=5, k=11, chart(5, 11) ∪ chart(11, 13) := {} ∪ {VP} = {}
 - j=13, i=5, k=12, chart(5, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=6, k=7, chart(6, 7) ∪ chart(7, 13) := {N} ∪ {VP} = {}
 - j=13, i=6, k=8, chart(6, 8) ∪ chart(8, 13) := {} ∪ {} = {}
 - j=13, i=6, k=9, chart(6, 9) ∪ chart(9, 13) := {} ∪ {} = {}
 - j=13, i=6, k=10, chart(6, 10) ∪ chart(10, 13) := {} ∪ {SC} = {}
 - j=13, i=6, k=11, chart(6, 11) ∪ chart(11, 13) := {} ∪ {VP} = {}
 - j=13, i=6, k=12, chart(6, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=7, k=8, chart(7, 8) ∪ chart(8, 13) := {A} ∪ {} = {}
 - j=13, i=7, k=9, chart(7, 9) ∪ chart(9, 13) := {VP} ∪ {} = {}
 - j=13, i=7, k=10, chart(7, 10) ∪ chart(10, 13) := {VP} ∪ {SC} = {VP}
 - j=13, i=7, k=11, chart(7, 11) ∪ chart(11, 13) := {} ∪ {VP} = {}
 - j=13, i=7, k=12, chart(7, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=8, k=9, chart(8, 9) ∪ chart(9, 13) := {V} ∪ {} = {}
 - j=13, i=8, k=10, chart(8, 10) ∪ chart(10, 13) := {} ∪ {SC} = {}
 - j=13, i=8, k=11, chart(8, 11) ∪ chart(11, 13) := {} ∪ {VP} = {}
 - j=13, i=8, k=12, chart(8, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=9, k=10, chart(9, 10) ∪ chart(10, 13) := {A} ∪ {SC} = {}
 - j=13, i=9, k=11, chart(9, 11) ∪ chart(11, 13) := {} ∪ {VP} = {}
 - j=13, i=9, k=12, chart(9, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=10, k=11, chart(10, 11) ∪ chart(11, 13) := {A} ∪ {VP} = {SC}
 - j=13, i=10, k=12, chart(10, 12) ∪ chart(12, 13) := {} ∪ {P} = {}
 - j=13, i=11, k=12, chart(11, 12) ∪ chart(12, 13) := {V} ∪ {P} = {VP}
- SUCCESS

From the procedures above, we can see the details of the processing which includes 78 steps if j=13. When i=0, k expands from 1 to 12(12 steps); when i=1, k expands from 2 to 12 (11 steps); when i=2, k expands from 3 to 12 (10 steps)..When i=11, k is 12(1steps). All the steps involved can be obtained as follows: 12+11+10+9...+4+3+2+1=78. The calculations of steps are shown in Table 4.

TABLE IV. THE PROCESSING PROCEDURES (J=13)

j=13	i=0	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12
j=13	i=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12	
j=13	i=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12		
j=13	i=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12			
j=13	i=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12				
j=13	i=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12					
j=13	i=6	k=7	k=8	k=9	k=10	k=11	k=12						
j=13	i=7	k=8	k=9	k=10	k=11	k=12							
j=13	i=8	k=9	k=10	k=11	k=12								
j=13	i=9	k=10	k=11	k=12									
j=13	i=10	k=11	k=12										
j=13	i=11	k=12											

D. The Statistical Analysis of GP Effect

Effect of garden path can be calculated based on the statistical analysis. The formula created to analyze the confusion quotient is as follows:

$$V_{cq} = \frac{\sum_{i=1}^n (1 - \frac{O_i - E_i}{E_i})}{n} = \frac{1}{n} \sum_{i=1}^n (2 - \frac{O_i}{E_i})$$

According to the details of parameters, Vcq is used to show the confusion degree and cq means ‘‘confusion quotient’’. The letter O is the abbreviation of ‘‘observed frequencies’’ during the processing. The letter E is the abbreviation of ‘‘expected frequencies’’. The letter n is the abbreviation of ‘‘number’’, meaning the numbers of peculiarities. The letter i means the unit of peculiarity. The complexer the structure of garden path sentence is, the higher the confusion quotient is. The value of 1 is the axis of confusion quotient. The GP effect appears when $2 \geq V_{cq} \geq 1$. If the value of Vcq is less than 1, whole ambiguity appears rather than the partial ambiguity which is the result of GP effect.[12]

TABLE V. THE CONFUSION QUOTIENT OF CP/NP AMBIGUITY

Model	O	E	O/E	Vcq
CP	24	136	0.18	1.82
NP	148	136	1.09	0.91
Total	272	272		

In Table 5, we can see the CP model has the value (1.82) and NP model has the value (0.91), which means the NP structure has less confusion quotient and therefore is considered the prototype. If the preferred NP model is replaced by the unpreferred or higher confusion model, the processing breakdown appears and GP phenomenon takes effect. The statistical analysis of GP effect reflects the fact that the ungrammatical preferred structure is always parsed firstly, which brings the backtracking during the processing. The grammatical unpreferred structure is proved to be the replacement of the initial processing. The chronological start of partial ambiguous structures is the potential reason of processing breakdown. The parsing can be shown by the online Stanford Parser which focuses more on statistics. The parsing is as follows:

She/PRP	told/VBD
me/PRP	a/DT
little/JJ	white/JJ
lie/NN	will/MD
come/VB	back/RP
to/TO	haunt/VB
me/PRP	/.
(ROOT	
(S	
(NP (PRP She))	
(VP (VBD told)	
(NP (PRP me))	
(SBAR	
(S	
(NP (DT a) (JJ little) (JJ white) (NN lie))	
(VP (MD will)	

ACKNOWLEDGMENT

This research is supported in part by China National Social Science Fund (12FYY019) and China Postdoctoral Fund (2015M570424)

REFERENCES

- [1] M. H. Christiansen and N. Chater, "Connectionist natural language processing: The state of the art," *Cognitive Science*, 1999, 23(10), pp. 417-437.
- [2] R. M. Losee, "Natural language processing in support of decision-making: Phrases and part-of-speech tagging," *Information Processing & Management*, 2001, 37(11), pp. 769-787.
- [3] J. Flanigan, C. Dyer, and J. Carbonell, "Large-Scale Discriminative Training for Statistical Machine Translation Using Held-Out Line Search," *Proceedings of NAACL-HLT*, 2013, pp. 248-258.
- [4] J. Pustejovsky and B. Boguraev, "Lexical knowledge representation and natural language processing," *Artificial Intelligence*, 1993, 63(10), pp. 193-223.
- [5] D. S. McNamara, S. A. Crossley, and R. Roscoe, "Natural language processing in an intelligent writing strategy tutoring system," *Behavior Research Methods*, 2013, 45(2), pp. 499-515.
- [6] A. Irvine and C. Callison-Burch, "Combining bilingual and comparable corpora for low resource machine translation," *Proceedings of the Eighth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, 2013, August, pp. 262-270.
- [7] G. Wisniewski and F. Yvon, "Oracle decoding as a new way to analyze phrase-based machine translation," *Machine Translation*, 2013, 27(2), pp. 115-138.
- [8] Y. Choi and J. C. Trueswell, "Children's (in)ability to recover from garden paths in a verb-final language: Evidence for developing control in sentence processing," *Journal of Experimental Child Psychology*, 2010, 106(5), pp. 41-61.
- [9] E. Malaia, R. B. Wilbur, and C. Weber-Fox, "ERP evidence for telicity effects on syntactic processing in garden-path sentences," *Brain and Language*, 2009, 108(3), pp. 145-158.
- [10] E. Hüllermeier, "Fuzzy sets in machine learning and data mining," *Applied Soft Computing*, 2011, 11(2), pp. 1493-1505.
- [11] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Information Sciences*, 2007, 177(18), pp. 3799-3821.
- [12] J. L. Du, *The Asymmetric Information Compensation Hypothesis: Research on confusion Quotient in Garden Path Model*, Doctoral Dissertation for Communication University of China, 2013.
- [13] J. L. Du and P.F.Yu, "A computational linguistic approach to natural language processing with applications to garden path sentences analysis," *International Journal of Advanced Computer Science & Applications*, 2012, 3(9), pp.61-75.
- [14] J. L. Du and P.F.Yu, "Predicting Garden Path Sentences Based on Natural Language Understanding System," *International Journal of Advanced Computer Science & Applications*, 2012, 3(11), pp.1-7.
- [15] J. L. Du, P.F.Yu, and M. L. Li, "Machine Learning from Garden Path Sentences: The Application of computational Linguistics," *International Journal of Emerging Technologies in Learning*, 2014, 9(6), pp. 58-62.
- [16] Z. G. Shoostari, and S. Shahri. "Down the Garden Path: An Effective Kind of EFL Grammar Instruction," *Procedia Social and Behavioral Sciences*, 2014, 98(6), pp.1777-1784.
- [17] T. J. Slattery, et al. "Lingering Misinterpretations of Garden Path Sentences Arise from Competing Syntactic Representations," *Journal of Memory & Language*, 2013, 69(2), pp.104-120.
- [18] Y. Choi, "Children's (In)Ability to recover from garden paths in a verb-final language: evidence for developing control in sentence processing," *Journal of Experimental Child Psychology*, 2010, 106(1), pp. 41-61.

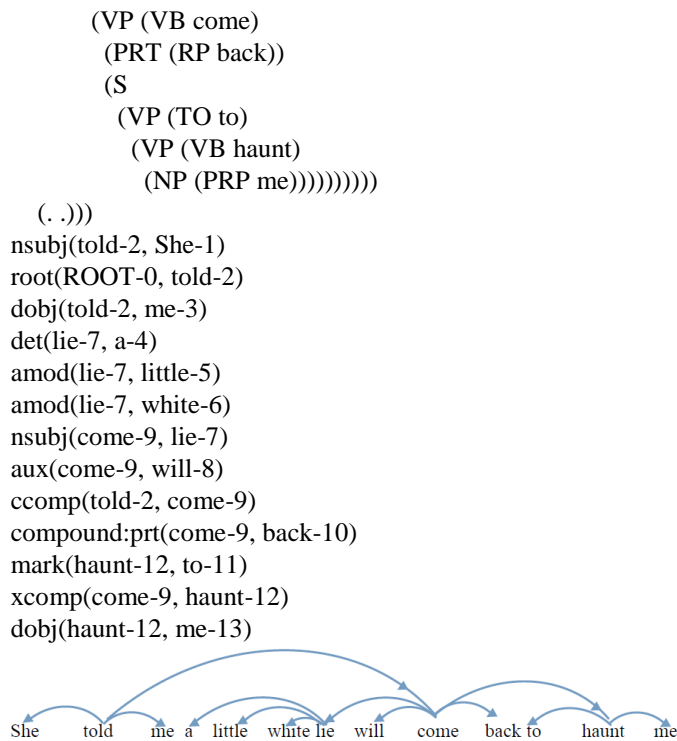


Fig. 6. Universal dependencies of GP sentence from Stanford Parser

From the parsing above, we can learn that Stanford Parser has the potential to deal with some kind of local ambiguity, e.g. tell+CP/NP, based on the Probabilistic Context Free Grammar (PCFG). And the dependency grammar is proved to be useful to semantically analyze the complex sentence. Please see the universal dependencies provided by the parser.

IV. CONCLUSION

The ambiguous structure of V+Pron+CP/NP has the potential to result in the processing breakdown. If V+Pron+NP structure has high observed frequency and low confusion quotient, the structure is considered the preferred one which is parsed initially and the parsing is ungrammatical. The replacement of ungrammatical preferred structure of V+Pron+NP by the grammatical unpreferred structure of V+Pron+CP can bring the GP effect. The methods of computational linguistics are proved to be effective to explore the phenomenon.

GP effect has been discussed by scholars from different perspectives [13-18], and this local ambiguity phenomenon with processing breakdown and backtracking deserves serious attention, especially for natural language processing. This paper discusses the effect theoretically and draws a conclusion that an effective system needs the help of multiple disciplines, including computational linguistics, cognitive linguistics, psychological linguistics, computer science etc. How to balance the needs of rule-based and statistics-based methods is a new challenge for researchers of computational linguistics in future.