Performance of Memoized- Most- Likelihood Parsing in Disambiguation Process

Maya Ingle M Chandwani

Abstract

In this paper, we first present a memoized parsing method for reducing the efforts of computation in parsing the strings/ sentences of a formal' natural language. We then discuss the statistical parsing that extracts the maximum/ most likelihood parse amongst the several parses of a string/ sentence in formal and natural domain as the most appropriate representative in disambiguation process. We integrate the statistical and memoized parsing together to achieve an efficient parsing technique. This integrated approach allows us to obtain the memoized-most-likelihood parse. Memoized-most-likelihood parse has an additional performance strength in the sense that it is highly useful further in parsing semantics.

Keywords : Natural Language Processing, Disambiguation, Statistical Parsing, Character Recognition

0. Introduction

Ambiguity and efficiency have always been the two important issues related to the parsing process as well as disambiguation process. There may exist a large number of possible derivation tree structures for a text of any language (formal or natural) and may require a large searching space. Probabilistic/ statistical techniques have been widely used to draw a maximum likelihood parse (or most likelihood parse) [1][2] whereas memoization technique in parsing allows effectively the scanning and understanding of a derivation tree structure using sub-tree criteria with a certain amount of efficiency [3][4]. The process of computing memoized-most-likelihood parse based on memoized probabilistic parsing technique has been presented in this paper. The memoized-most-likelihood parse helps us in drawing the most appropriate semantics of a formal language text (i.e. strings structure) as well as a natural language text (i.e. the sentences of English language) at the time of parsing itself in efficient manner thereby providing the ease in disambiguation process.

First, we discuss the memoization employed in parsing the strings of formal language and the sentences of natural language in Section 2. Section 3 describes the probabilistic parsing used to select the most likelihood parse amongst the several ones in formal and natural language domain both. The performance of memoized-most-likelihood parse is presented in Section 4. Finally, we conclude briefly in Section 5.

1. Memoized Parsing

Memoization is one of the techniques employed parsing algorithms to speed-up the parsing process. It reduces the re-computation efforts in producing the left parse (or right parse) of the strings (or sentences) of formal languages (or natural languages) using CKY-parsing algorithm. The effectiveness of memoization in parsing the strings of formal language contributes more significant as compared to the parsing of the natural language sentences. Memoization technique in CKY-algorithm (henceforth memoized CKY-algorithm) reduces the re-computation of sub-parses in parsing the sentences of natural language in which the repeated occurrences of same phrase structure exist. Also, it plays a significant role in parsing the strings of formal language particularly when the repeated sub-strings occur in an input string [5]. The bottom-up approach has been used to construct the recognition matrix and the top-down approach to produce the left parse of an input string in the algorithm [6].

1.1 Memoization in Formal Languages

The parses of the strings contain distinct as well as non-distinct sub-trees. We describe the performance of memoized-CKY-algorithm in brief considering both of these cases in this section.

Case I: Distinct sub-trees : The string *w*=abaabaaba of length n=9 is recognized by the grammar G_1 with *T*={*a, b*} and *N*={*S, A, B*} as shown in Table 1 (a), whose left parse tree is built-up and shown in Figure 1 using CKY-top-down parsing. Main algorithm executes the function *lookup-parse*(0, 9, *S*) such that the parse lists produced are either stored in lookup table *tab* or looked up from look-up table. The process of execution continues until all the recursive functions involved during parsing are executed and their corresponding *tab* values become non-null. Table 2 shows the actual visualization of execution of algorithm for parsing this string with parameters of current lookup-parse, status of *tab*, the current parse list at each stage and the parameters of lookup-parse functions which are used in current function recursively. The "looked-up" counts (entries in the status of tab column) compute the optimal degree of memorization which in this case is 5.

Case II: Non-distinct sub-trees : There exists three non-distinct sub-trees of order 1, 2 and 4 respectively in the parse tree of a string *w*=aaaaaaaa generated by some grammar G as shown in Figure 2. Using the algorithm, these non-distinct sub-trees are maintained separately in the look-up table as shown in Table 3 producing a parse tree with optimal degree of memoization 3.

1.2 Memoization in Natural Languages

The Memoized-CKY-algorithm performs effectively for parsing of sentences of natural language in which repeated occurrences of same phrase structure exist [7]. The effectiveness of an algorithm has been explained by considering a specific domain of sentences of a natural language grammar G_2 with production rules for parsing as follows:

Table 1(a) Grammar G_1

1SAB 2SBB 3AAB 4ACC 5Aa 6BBB 7Bb 8BCA 9Cb 10CBA 11CAA

Table 1(b) Rule probabilities of G_1

1	2	3	4	5	6	7	8	9	10	11
0.5743	0.4257	0.1062	0.2017	0.6921	0.0201	0.4848	0.4951	0.6217	0.1212	0.2571

Table 1(c) Relative probabilities of strings recognized by G_1

length of string=2

sno.	string	parses	parse list	relative prob
1	ab	1	1,5,7,	1.000000
2	bb	1	2,7,7,	1.000000

length of string=3

Sno.	String	Parses	Parse list	Relative prob
1	aba	1	1,5,8,9,5,	1.000000
2	abb	2	1,5,6,7,7, 1,3,5,7,7,	0.159507 0.840493
3	bab	1	2,8,9,5,7,	1.000000
4	bba	1	2,7,8,9,5,	1.000000
5	bbb	3	2,7,6,7,7, 2,6,7,7,7, 1,4,9,9,7,	0.041282 0.041282 0.917437

length of string=4

Sno.	String	Parses	Parse list	Relative prob
1	aaaa	1	1,5,8,11,5,5,5,	1.000000
2	aaab	1	2,8,11,5,5,5,7,	1.000000
3	aabb	1	1,4,11,5,5,9,7,	1.000000
4	abaa	1	1,5,8,10,7,5,5,	1.000000
5	abab	2	1,5,8,9,3,5,7, 1,3,5,8,9,5,7,	0.500000 0.500000
6	abba	2	1,5,6,7,8,9,5 ,5,	0.840493
7	abbb	3	1,5,6,7,6,7,7, 1,5,8,9,4,9,9, 1,3,3,5,7,7,7,	0.001825 0.947492 0.050682
8	baaa	1	2,7,8,11,5,5,5,	1.000000
9	baab	2	1,4,9,11,5,5,7, 2,8,10,7,5,5,7,	0.599351 0.400649
10	baba	1	2,8,9,5,8,9,5,	1.000000
11	babb	3	2,8,9,5,6,7,7, 2,6,8,9,5,7,7, 1,4,10,7,5,9,7,	0.188418 0.188418 0.623165
12	bbaa	1	2,7,8,10,7,5,5,	1.000000
13	bbab	3	2,7,8,9,3,5,7, 2,6,7,8,9,5,7, 1,4,9,10,7,5,7,	0.550224 0.104420 0.345356
14	bbba	3	2,7,6,7,8,9,5, 2,6,7,7,8,9,5, 1,4,9,9,8,9,5,	0.041282 0.041282 0.917437
15	bbbb	4	2,7,6,7,6,7,7, 2,7,8,9,4,9,9, 2,6,6,7,7,7,7, 1,3,4,9,9,7,7,	0.001567 0.813367 0.001567 0.183499

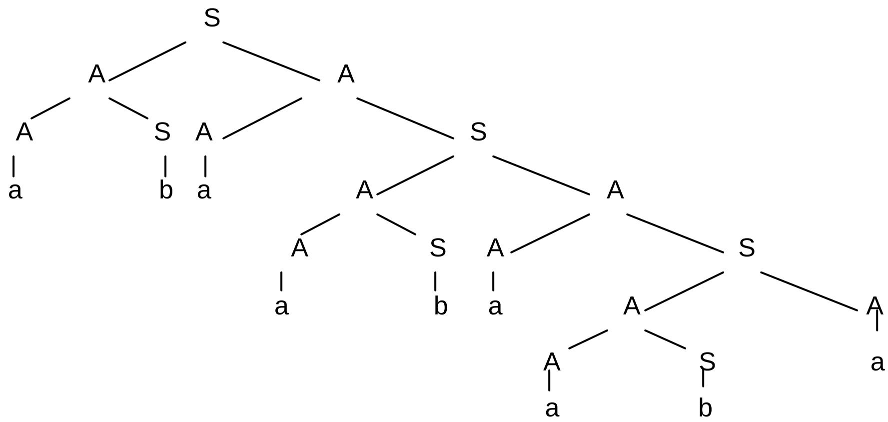


Fig. 1 Parse tree of a string w=abaabaaba showing effect of Memoization

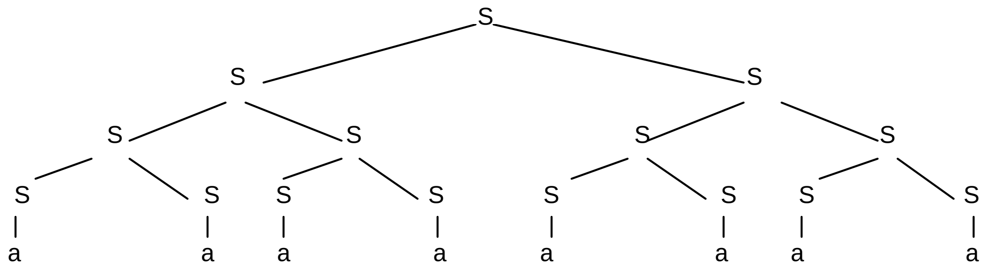


Fig. 2 Parse tree of a string w=aaaaaaaa with non-distinct terminating sub-trees

Table 2: Execution of Memoized CKY-Parsing (Using Distinct sub-trees)

Current Function	**Tab values**	**Current parse list (pt)**	**Functions used in Current Function**
(0, 9, S)	Null	{2}	(0, 2, A), (2, 9, A)
(0, 2, A)	Null	{2, 4}	(0, 1, A), (1, 2, S)
(0, 1, A)	{6}	{2, 4, 6}	—
(1, 2, S)	{3}	{2, 4, 6, 3}	—
(0, 2, A)	{4,6,3}	,,	—
(2, 9, A)	Null	{2, 4, 6, 3, 4}	(2, 3, A), (3, 9, S)
(2, 3, A)	Looked up	{2,4,6,3,4,6}	—
(3, 9, S)	Null	{2, 4, 6, 3, 4, 6, 2}	(3, 5, A), (5, 9, A)
(3, 5, A)	Looked up	{2, 4, 6, 3, 4, 6, 2, 4, 6, 3}	—
(5, 9, A)	Null	{2,4,6,3,4,6,2,4,6,3,4}	(5, 6, A), (6, 9, S)
(5, 6, A)	Looked up	{2, 4, 6, 3, 4, 6, 2, 4, 6, 3, 4, 6}	—

(6, 9, S)	Null	{2, 4, 6, 3, 4, 6, 2, 4, 6, 3, 4, 6, 2}	(6, 8, A), (8, 9, A)
(6, 8, A)	Looked up	{2, 4, 6, 3, 4, 6, 2, 4, 6, 3, 4, 6, 2, 4, 6, 3}	—
(8, 9, A)	Looked up	{2, 4, 6, 3, 4, 6, 2, 4, 6, 3, 4, 6, 2, 4,6,3,6}	—
(6, 9, S)	Non-null	,,	—
(5, 9, A)	Non-null	,,	—
(3, 9, S)	Non-null	,,	—
(2, 9, A)	Non-null	,,	—
(0, 9, S)	Non-null	,,	—

Table 3: Execution of Memoized CKY-Parsing for a string w=aaaaaaaa

(Using Non-distinct sub-trees)

Current Function	Tab Values	Functions Used in Current Function
(0, 8, S)	Null	(0, 4, S) (4, 8, S)
(0, 4, S)	Null	(0, 2, S) (2, 4, S)
(0, 2, S)	Null	(0, 1, S) (1, 2, S)
(0, 1, S)	Non-null	—-
(1, 2, S)	Looked-up*****	—-
(0, 2, S)	Non-null	—-
(2 , 4, S)	Looked-up******	—-
(0, 4, S)	Non-null	—-
(4, 8, S)	Looked-up*******	—-

***** Parse tree of order 1 is looked up from look-up table.

****** Parse tree of order 2 is looked up from look-up table.

******* Parse tree of order 4 is looked up from look-up table.

Table 4: Functions with their equivalent looked-up functions

No.	Current functions	Looked-up functions
1	(2, 4, NP)	(5, 7, NP), (8, 10, NP), (11, 13, NP)
2	(2, 3, *det)	(5, 6, *det), (8, 9, *det) etc.
3	(3, 4, *n)	(6, 7, *n) (9, 10, *n) (12, 13, *n)

Grammar G_2:

1. S®NP.VP
2. S®S.PP

8. *n ®l| telescope| man| hill| garden
9. *v®saw| cut| read

3. NP ®NP.PP
4. NP ®*det.*n
5. NP ®*n
6. PP ®*prep.NP
7. VP ®*v.NP
10. *det®a| the| an
11. *prep®with| on| from

A large number of procedures have been encountered with various parameters during top-down recursive in CKY-parsing algorithm as the sentence "*I saw a man with a telescope in the garden from the hill*" contains the three prepositional phrases as "*with a telescope*", "*in the garden*" and "*from the hill*". Some of the procedures share their output at lexical level and above also, thereby avoid the re-computation efforts as shown in Table 4.

2. Statistical Parsing

The statistical and structural information may be integrated together to rank the various parses of a string of a formal language as well. To derive all possible parses of a sentence/ string is not a difficult problem but it is crucial to rank these parses according to some criteria [8]. Probabilistic or statistical parsing has been widely used to resolve the various kinds of ambiguity. The sentence "*send for us the timely news*" seems to be ambiguous when "*us*" is interpreted as United States. In these situations, it is possible to compute the probabilities of various rules of grammar and to select the most likelihood parse amongst the various parses.

There exist three types of probabilistic grammars namely probabilistic context-free grammar, probabilistic context-sensitive grammar and probabilistic transformational grammar. Further, three types of probabilistic weighting of CFG are possible, they are namely, Suppes type weighting, Salomma type weighting and probabilistic CFG with derivation weighting (d_w grammar) [9]. We have used Suppes type weighting in probabilistic context-free grammar G_1 with *N*={*S, A, B, C*}; *T*={a, b}; augmented by rule probabilities (as listed in parentheses) in a set of production rules as follows:

1. Block	**2. Block**
1. S®®AB (0.50)	*3. A®®CC (0.33)*
2. S®®BB (0.50)	*4. A®®AB (0.33)*
	5. A®®a (0.34)
3. Block	**4. Block**
6. B®®BB (0.33)	*9. C®®b (0.34)*
7. B®®b (0.34)	*10. C®®BA (0.33)*
8. B®®CA (0.33)	*11. C®®AA (0.33)*

The probability of a sentence becomes negligible as its length increases. At the same time, a sentence may possess a finite number of parses. Therefore, the relative probabilities among ambiguous derivation trees are used. This measure gives the likelihood of each derivation amongst all possible derivations. A derivation with highest relative probability denotes the most appropriate parse of a string**.** An algorithm "Estimate" computes the rule probabilities and estimates the resultant probabilities by considering all possible parses (left and right parses both) of the strings of various lengths thereby producing the most correct and maximum likelihood parse of an ambiguous string [5].

2.1 Case Studies

The grammar for formal language G_1 with its rule probabilities is listed in Table 1(a) and Table 1(b) whereas the grammar for natural language G_2 given in Table 5(a) With the help of these probabilities, the language ambiguity has been quantified and pointed out the most likelihood parse of an ambiguous sentence/ string of natural/ formal language thereby improving the performance of top-down parsing technique. We discuss these cases in the following section.

Case I : Parsing in natural language domain : Using an algorithm "*Estimate*" the unambiguous sentences possess relative probability as unity whereas for the ambiguous sentences, the relative probabilities may be less than unity. These probabilities have been shown in Table 5(b). The values of relative probabilities signify the different interpretations each. The sentence "*I saw a man with a telescope in the garden*" is recognized by the grammar but it possesses the structural ambiguity [10]. There exists five distinct parses as shown in Table 5(b). The most likelihood parse possess the highest relative probability i.e. 0.990032. A parse tree of the sentence is shown in Figure 3(a) and its interpretation is given by Venn diagram as shown in Figure 3(b). The ambiguity increases sufficiently if we augment more prepositional phrases in the above sentence.

Case II : Parsing in formal language domain : The resultant rule probabilities for each of the grammatical rules and lexical rules of various grammars computed by an algorithm "*Estimate*" uses both left well as right parses. The resultant probability represents the probability that a L.H.S non-terminal is used in formation of any string *w* recognized by grammar G. The resultant rule probabilities of various rules of grammar G_1 have been shown in Table 6. Thus, resultant probability of rule 1 in G_1 i.e. *S®AB* is 0.5743 meaning that the sequence *AB* forms a sentence *S* with a probability 0.5743. Similarly, Table 7 shows some of ambiguous strings of length 3, 4 and 5 recognized by the grammar G_1 with their corresponding relative probabilities. It is observed that the increase in length of string causes increase in number of parses. A string *w*= abbbb of length n=5 recognized by the grammar G_1 and has five parses with relative probabilities 0.000164, 0.448343, 0.085086, 0.023982 and 0.442425 respectively as shown in Table 7. The relative probability of first parse is all most zero whereas the second parse possesses the relative probability as the highest amongst all other parses. Thus, the probabilistic parsing method owns its usefulness not only in sentence/ string disambiguation but also in improving the performance of top-down parsing when used for obtaining the hints for reordering the rules according to the rule probabilities.

3. Memoized - Most - Likelihood Parsing

The statistical and memorized parsing may be integrated together to produce memoized likelihood parse of a string/ sentence of a formal/ natural language. The performance of such parsing has been investigated and proved to be the best in this section.

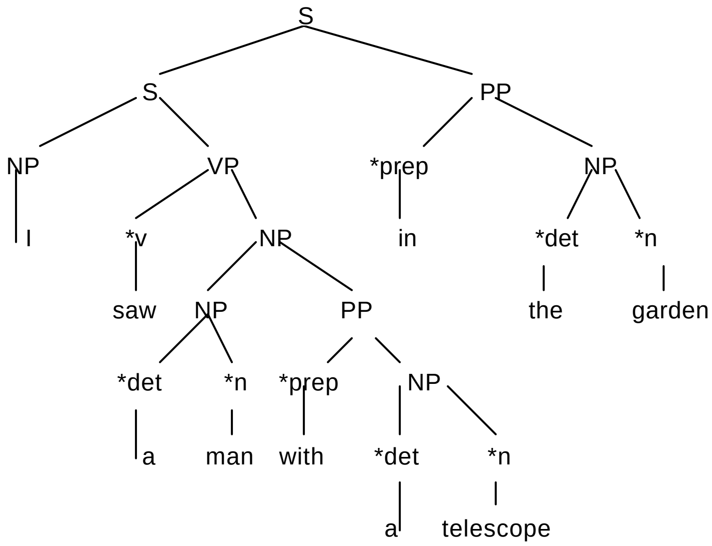
Some structures of sentences in the language or sub-strings of a string in formal language occur frequently. And out of them only few sub-structures need to be computed during parsing whereas other are looked-up thereby saving the computational efforts. Similarly, there may exist the number of parses of a string/ sentence of a language. And out of them, the most likelihood parse has to be selected as it is the best structural representation of that sentence. While parsing in natural language domain, it has been found that the structural ambiguity is possessed in the sentence "*I saw a man with a telescope in the garden*" recognized by the grammar shown in Table 5(a) having five distinct parses as shown in Table 5(b). We consider the best structural representation of this sentence as the most likelihood parse with the highest relative probability as 0.990032. Instead of top-down CKY parsing, a Memoized-CKY-parsing algorithm is used with probabilistic grammars. In the maximum-likelihood parse as shown in Figure 3(a), there exist three distinct terminating sub-parse trees for NP-phrases. Since the algorithm produces a left parse of a sentence, first NP-phrase will be computed while others are looked-up having the optimal degree of memoization thereby speeding-up the procedure of parsing with best output. Similarly, the effectiveness of Memoized-CKY-parsing using probabilistic grammars throws the attention towards the memorized-most-likelihood parse of a string *w*= abbbb recognized by the grammar G_1. Consider the second parse of this string with relative probability 0.442425 as shown in Table 7. During parsing, it stores some of the results of *lookup-parse* with specific parameters whereas some are looked up maintaining the optimal degree of memoization. . Also, it has been observed that amongst the several parses, more than one parse may have the maximum relative probability. In such a situation specially, Memoized-most-likelihood parsing technique helps us in selecting the most appropriate parse on the basis of optimal degree of memoization otherwise we choose a parse arbitrarily along with an additional performance strength. Thus, the memoized parsing in conjunction with probabilistic parsing concept establishes its improved performance in sentence/ string disambiguation in the sense that it is highly useful further in parsing semantics.

Table 5(a) Grammar G_2

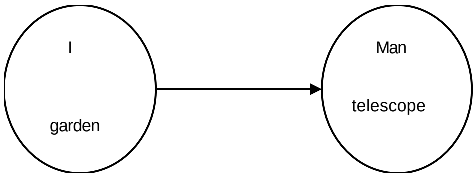
1SNPVP 2SSPP 3NPNPPP 4NP*det*n 5 NP*n 6PP*prepNP 7VP*v NP 8*nI 9*n man 10*ntelescope 11*ngarden 12*nhill 13*vsaw 14*deta 15*detthe 16*prepwith 17*prepin 18*prepon

Table 2(b) Relative probabilities of sentences recognized by Grammar G

I saw a man	
1,5,8,7,13,4,14,9,	1.000000
I saw a man with a telescope	
1,5,7,13,3,4,14,9,6,16,4,14,10,	0.316812
2,1,5,8,7,13,4,14,9,6,16,4,14,10	0.683188
I saw a man with a telescope on the hill	
1,5,7,13,3,4,14,9,6,16,3,4,14,10,6,17,4,15,11,	0.000025
2,1,5,8,7,4,13,4,14,9,6,16,3,4,14,10,6,17,4,15,11,	0.004959
2,1,5,8,7,13,3,4,14,9,6,16,4,14,10,6,17,4,15,11,	0.004959
1,5,8,7,13,3,3,4,14,9,6,16,4,14,10,6,17,4,15,11,	0.000025
2,2,1,5,8,7,13,4,14,6,16,4,14,10,6,17,4,15,11,	0.990032
I saw a man with a telescope on the hill in the garden	
1,5,8,7,13,3,4,14,9,6,16,3,4,14,10,6,17,3,4,15,11,6,18,4,15,12,	0.046872
2,1,5,8,7,13,4,14,9,6,16,3,4,14,10,6,17,3,4,15,11,6,18,4,15,12,	0.101078
2,1,5,8,7,13,3,4,14,9,6,16,4,14,10,6,17,3,4,15,11,6,18,4,15,12,	0.101078
2,1,5,8,7,13,3,4,14,9,6,16,3,4,14,10,6,17,4,15,11,6,18,4,15,12,	0.101078
1,5,8,7,13,3,3,4,14,9,6,16,4,14,10,6,17,3,4,15,11,6,18,4,15,12,	0.046872
1,5,8,7,13,3,3,4,14,9,6,16,3,4,14,10,6,17,4,15,11,6,18,4,15,12,	0.046872
2,1,5,8,7,13,4,14,9,6,16,3,3,4,14,10,6,17,4,15,11,6,18,4,15,12,	0.101078
2,2,1,5,8,7,13,3,4,14,9,6,16,4,14,10,6,17,4,15,11,6,18,4,15,12,	0.217969
2,1,5,8,7,13,3,3,4,14,9,6,16,4,14,10,6,17,4,15,11,6,18,4,15,12,	0.101078
2,2,2,1,5,8,7,13,4,14,9,6,16,4,14,10,6,17,4,15,11,6,18,4,15,12,	0.136026



(a) Most likelihood parse of "*I saw a man with a telescope in the garden*"



(b) A Venn diagram of above parse **Figure - 3**

Table 6 Rule probabilities of grammar G_1

RuleNo.	**ResultantParse**
1	0.5743
2	0.4257
3	0.1062
4	0.2017
5	0.6921
6	0.0201
7	0.4848
8	0.4951
9	0.6217
10	0.1212
11	0.2571

Table 7 Relative prob. of some ambiguous parses of strings recognized by grammar G_1

Length	String	No. of Parses	Parse list	Relative probability
3	abb	2	1,5,6,7,7	0.159507
			1,3,5,7,7	0.840493
	bbb	3	2,7,6,7,7	0.041282
			2,6,7,7,7	0.041282
			1,4,9,9,7	0.917437
4	abab	2	1,5,8,9,3,5,7	0.500000
			1,3,5,8,9,5,7	0.500000
	abbb	3	1,5,6,7,6,7,7	0.001825
			1,5,8,9,4,9,9	0.947492
			1,3,3,5,7,7,7	0.050682
	bbbb	4	2,7,6,7,6,7,7	0.001567
			2,7,8,9,4,9,9	0.813367
			2,6,6,7,7,7,7	0.001567
			1,3,4,9,9,7,7	0.183499
5	aaaab	2	1,5,8,11,5,5,3,5,7	0.503322
			1,4,11,5,5,11,5,5,7	0.496678
	abaab	3	1,5,8,9,4,11,5,5,9	0.720507
			1,3,5,8,10,7,5,5,7	0.108660
			2,8,11,3,5,7,5,5,7	0.170833
	abbab	4	1,5,6,7,8,9,3,5,7	0.022739
			1,5,8,9,4,10,7,5,9	0.333353
			1,3,3,5,7,8,9,5,7	0.119819
			2,8,11,5,4,9,9,5,7	0.524090
	abbbb	5	1,5,6,7,6,7,6,7,7	0.000164
			1,5,8,9,3,4,9,9,7	0.448343
			1,5,6,7,8,9,4,9,9	0.085086
			1,3,3,3,5,7,7,7,7	0.023982
			1,4,11,5,4,9,9,9,7	0.442425

4. Conclusion

Memoized-CKY-parsing with probabilistic parsing plays an effective role not only in disambiguation process but it also provides the most appropriate structural representation of the sentence at the same time. As compared to the performance of most-likelihood parse, definitely memoized-most-likelihood parse has an additional performance strength in the sense that it is highly useful further in parsing semantics.

5. References

1. Fujisaki, T., Jelinek, F. C., Black, E. and Niehino, T. A., “Probabilistic Parsing Method for Sentence Disambiguation”, International parsing Workshop, pp. 85-94, 1989.

2. Corazza, A., “Parsing Strategies for the Integration of Two Stochastic Context-Free Grammars”, IWPT 2003, 8 th International Workshop of Parsing Technologies, 23-25 April 2003.

3. Foth, K., Menzel, W., “Subtree Parsing to speed up Deep Analysis”, IWPT 2003, 8 th International Workshop of Parsing Technologies, 23-25 April 2003.

4. Coremen, T. H., Lieserson, C. E., Rivest, R. L., Introduction to Algorithms, Prentice Hall of India, 1999.

5. Ingle, M., Ph.D Thesis on “Computing Research Investigations in Natural Language Processing”, DAVV, Indore, Dec 2002.

6. Harrison, M. A., Introduction to Formal Language Theory, Addison-Wisley Pub., 1978.

7. Ingle, M. and Chandwani, M. “Memoized CKY-algorithm for Natural Languages,” Journal of Institute of Engineers (India), Vol. 83, pp. 12-15, 2002.

8. Charniak, E., “Statistical Parsing with a Context-free Grammar and Word Statistics”, American Association for Artificial Intelligence, pp. 589-603, 1997.

9. Klein, W. and Dittmar, N., Developing Grammars : The acquisition of German syntax by foreign workers, Springer-Verlag, Berlin H, New York, 1979.

10. Ingle, M. and Chandwani, M., “The Algorithm “Disambiguate” for Resolving Structural Ambiguity”,37 th National Convention 2002 of CSI, Harnessing and Managing Knowledge, IISC, Banglore, 2002.

About Authors

Dr. (Mrs.) M Ingle is System Analyst & Reader in School of Computer Science, D.A.V.V., Indore, Madhya Pradesh.
E-mail : maya_ingle@rediffmail.com