The Implementation Of Left Corner Parsing In Indonesian Language

Dewi Soyusiawaty, Andri Pranolo
{dewi.soyusiawaty@tif.uad.ac.id[1], andri.pranolo@tif.uad.ac.id[2]}

Informatics Department, Universitas Ahmad Dahlan, Yogyakarta, Indonesia[1,2]

Abstract. Indonesian language is a national language spoken not only by Indonesian citizens but also foreign citizens when they are in Indonesia. The communication is carried out through a lot of media both written and orally. Computer based translator applications now not only rely on written input but also spoken one. Whereas, it's not sure whether the inputted sentence is grammatically correct or not. When the inputted sentence is incorrect, it will influence the result given by the translator application. One method to know whether the input is correct or not is by checking the sentence pattern. This research explains left corner parsing applied on sentence pattern tracking. Left corner parsing is a combination of top down and bottom up parsing by taking top down prediction and bottom up tracking. From the two tracking techniques, the sentence will be parsed to be token per token and then the grammar rule which is previously determined is searched. With the tracking and prediction, a parser to parse the sentence will be gotten, so the sentence pattern will be identified. This research discusses and makes an auxiliary tool which function is to identify the pattern of Indonesian language sentences. As a result, it can be identified whether the pattern of the inputted sentence is correct or not based on the stored grammar rule in the application.

Keywords: Parsing, Left Corner, Indonesian Language.

1 Introduction

2.1 Problem

Now days translator software can helps people to translate from one to another language in a relative short time for a relative huge number of words. However, the fact shows that translation result from the software is not yet reliable enough since most of them only translate based on the words from the source language. To produce a good translation in the target language, we can't rely on it because there are many factors influencing the quality of a translation such as the grammar structure of a language which is not always the same. The structure of the inputted sentence which is grammatically incorrect will affect the translation result [1].

As the Indonesian language instruction effort and pilot study to develop translator application applying parsing concept chosen to produce information whether inputted sentence pattern is grammatically correct or not, so an application to check the sentences can be developed. Parsing algorithm used is left corner parsing. This method is a combination of bottom up and top down parsing method. Generally this is bottom up parsing method, but a top down prediction is applied in this method [2].

ICSTI 2019, September 20, Yogyakarta, Indonesia
Copyright © 2020 EAI
DOI 10.4108/eai.20-9-2019.2292090

Context Free Grammar

Context Free Grammar (CFG) is used to explain the syntax of a language. Grammar is a form of human's natural language used in the daily communication. Every grammar rule has a syntactic structure that can be described by a grammar.

The components of CFG:

1. A group of token namely terminal
2. A group of non-terminal
3. A group of production rules
4. First symbol

In the CFG it needs to understand the following terms related to grammar in order to differentiate one symbol to another.

1. The following symbols are terminal
   a. Lowercase letter, in the first alphabet letters such as a, b, or c
   b. Operator symbols such as @, #, -, etc
   c. Number 0, 1, 2, etc
2. The following symbols are non-terminal
   Capital letter, from alphabet, A, B and C
   The first symbol
3. A lowercase letter of the end of alphabet such as u, v,...z represents a group of terminal
4. A lowercase Greek letter such as α, β, μ as an example represents a group of grammar symbol
5. If no another statement, the left part of production is the first symbol [3].

CFG is a language rule consisting of two parts. It left side consists of one non-terminal symbol. By using this grammar rule, the analysis result from a sentence forms a tree structure. This kind of tree is called parse tree. The example of CFG rule as following:

1. sentence → np, vp
2. np → noun, det
3. np → noun
4. vp → verb, np
5. vp → vp, pp
6. pp → prep, np
7. det → [itu]
8. noun → [anak]
9. noun → [anjing]
10. noun → [tongkat]
11. verb → [memukul]
12. prep → [dengan]

If the sentence on the CFG above is stated as a beginner symbol, the sentence 'anak itu memukul anjing dengan tongkat' will be analyzed by using grammar rules above by forming a beginner symbol (sentence) and then it will form a tree as following:

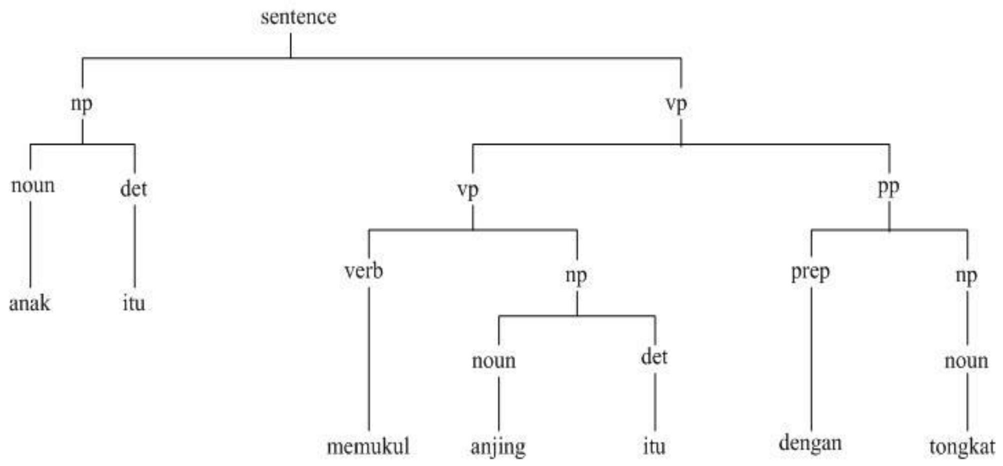


Fig. 1. Tree Structure

In the sentence structure analysis or parsing, there are two main methods which are top down and bottom up parsing method. In the top down parsing, the tree structure formation is done from top to down. Therefore, if the grammar rule uses CFG rule, the tree structure formation is done by forming non-terminal symbol on the left side of CFG rule. It is then continuously transformed to the right side until the terminal symbol is found. In the bottom up parsing, the transformation happens in opposite; the right side of CFG rule is transformed to the left side. Therefore, in this bottom up parsing method, the tree structure formation starts from its terminal symbol [4].

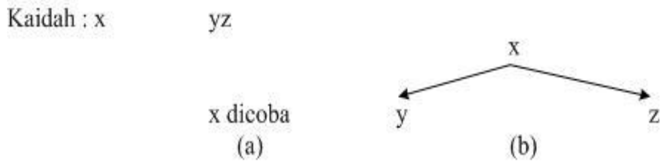


Fig. 2. Top Down Parsing

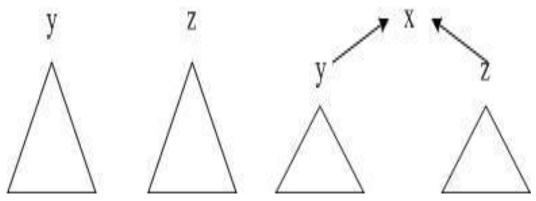


Fig. 3. Bottom Up Parsing

The writing of grammar in the natural language with CFG usually consists of ambiguity. For instance the writing of CFG rules on the (d) and (e) above. To analyze all possibilities written in the CFG rules, a back tracking mechanism is necessary. By using this mechanism, if in the analysis process some possibilities are found, one of them is chosen to be analyzed further until the analysis can’t be continued further. At the moment, back tracking occurs and other possibilities are tracked. Therefore, by the existence of back tracking it is possible to form more than one analysis or parse tree [5].

2.1 Left Corner Parsing

This method is a combination between bottom up and top down parsing. Generally this is bottom up parsing method, but a top down prediction is applied in this method.

As in the bottom up parsing, an analysis starts with the first word input from the sentence. The suitable CFG rule is then searched, but in this method, the right side of CFG rule does not need to completely match. If only the first part matches, so that rule is used. Then the rest words become the new searching target. In this method, the tree structure formation occurs one by one part. When the searching target which is a prediction is successful, so the new tree structure is formed consisting of successful trees. The following example in the figure 4 will explain it.

The searching area using this method is wider; it means that the tracking scope is also wider, so the possibility of the successful searching is bigger. As a result, the possibility of back tracking to occur is fewer. Even if it occurs, it's only in the closer area.

A more definite tracking can be done by giving information when prediction is done to form a certain phrase. As known that verb is an element to form vp, preposition is an element to form pp and np, which is element to form bigger phrase such as sentence, etc. By giving clear elements to form a phrase, the possibility of phrase formation failure can be identified as early as possible, so the useless tracking can be limited [5].

2.1 Indonesian Language Grammar

The sentence constructor structure is physically the clause. The clause is a lingual unit consisting of minimal subject (S) and predicate (P). The other clause constructor elements are object (O), complement/pelengkap (Comp/Pel) and adverb/keterangan (Adv/K). The following Table 1 is the clause structure pattern in Indonesian language that is commonly used [6].

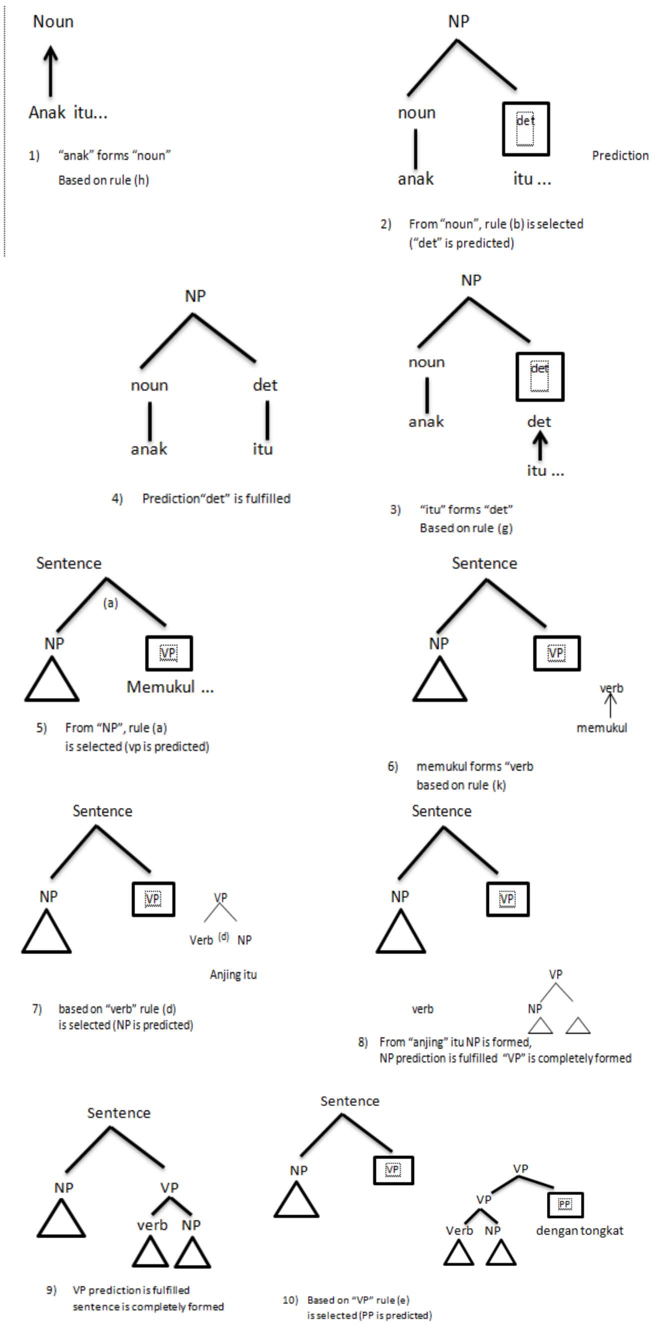


Fig. 4. Left Corner Parsing

Table 1. Clause Structure

No	Pattern Clause	Example of sentences
1	S-P	Dia belajar
2	S-P-O	Adik makan roti
3	S-P-Pel	Aku belajar menari
4	S-P-O1-O2	Kakek membelikan adik sepeda baru
5	S-P-O-K	Ia menendang bola ke atas atap rumah
6	S-P-Pel-K	Aku berenang gaya katak di Umbul Tirto kemarin
7	S-P-O1-O2-K	Kakek membelikan adik sepeda baru kemarin

A phrase is a grammatical unit consisting of two or more words which do not cross the clause function. The following Table 2 shows the type of phrase, pattern and the example. From the clause pattern and the phrase type, the basic sentence pattern can be combined. It consists of the functions which are filled by the phrase that consists of the word categories. The sentence structure can be shown in the Table 4 with the following compositions [7] [6]:

Table 2. Clause Pattern

No	**Pattern**				
1	S	P			
	N	N			
		V			
		Bil			
		Ket			
2	S	P	O		
	N	V	N		
3	S	P	Pel		
	N	V	N		
			BIL		
			V		
4	S	P	K		
	N	N	K		
		V	FD		
		BIL			
5	S	P	O	K	
	N	V	N	K	
6	S	P	O1	02	
	N	V	N	N	
7	S	P	O1	O2	K
	N	V	N	N	KET

2 Discussion

From the structure data of the clause, phrase and clause patterns, then the hierarchy of grammar can be arranged as follows:

2.1. CFG

Formally, a grammar consists of four elements:

1. T terminal finite set consisting of all symbols used to state the sentences in Indonesian language

   !"#$ %&‘()*+,-./ 0123456789 : ;<=> ?@ ABCDEFGHIJKLMNOPQRSTUVW
   XYZ[\]^_`abcdefghijklmnopqrstuvwxyz{¦}~¢£¤¥¦§¨©ª«¬®°±^{23}´µ• (1)

2. N Non-terminal finite set is a symbol to symbolize a sentence, phrase and word group. Table 3 consists of the list of the sentence symbol from the above clause structure, phrase and word group.

Table 3. Non Terminal (N)

No	Non Terminal	Symbol
1	Sentence/Kalimat	Kal
2	Subjek	S
3	Predicate	P
4	Object	O
5	Complement/Pelengkap	PEL
11	Nomina	N
12	Verb	V
13	Adjective	Adj
14	Adverb	Ket
15	Number/Bilangan	BIL
16	Question/Tanya	Tanya
17	Conjunction	Hub
18	Preposition/Kata Depan	Depan
19	etc	

3. P Production Rule Finite Set. The production rule in this research is a combination of clause pattern and the phrase

Table 4. Production Rule

No	Production Rule
1	<Kal> = <S><P>
2	<Kal> = <S><P><O>
3	<Kal>= <S><P><PEL>
4	<Kal>=<S><P><O1><O2>
5	<Kal>= <S><P><O><K>
6	<Kal>=<S><P><PEL><K>
7	<Kal>=<S><P><O1><O2><K>
8	<S>= <N>
9	<P>=<N> | <V> |<Bil>|<Ket>
10	<O>=<N>
11	<K>=<KET>
12	<PEL>=<N> | <BIL>
13	<N>={aku, dia, saya, kamu, bakso...}
14	<V> = {makan, minum, pergi ....}
15	<Ket> = {kemarin, ...}
16	<Bil> = {satu, dua .....}

4. The S ε N first symbol

<Kal>

2.2. General Application on Application

This application is a sentence pattern identification application in Indonesian language using *Left-Corner Parsing.*

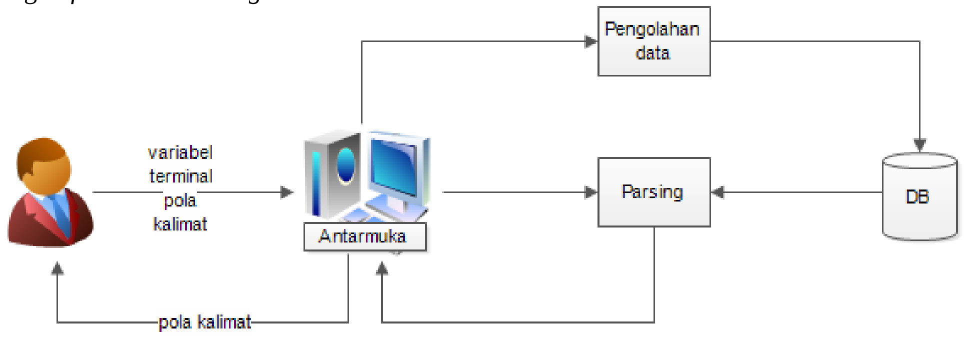


Fig. 5. Application Architecture

The Needs on Application

This application consists of four need module:

1. Variable Module: this module is used to process token variable data
2. Terminal Module: This module is used to manage the token terminal data
3. Pattern Module: This module is used to manage sentence pattern data
4. Parsing Module: This module is the main module used to process sentence pattern identification parsing

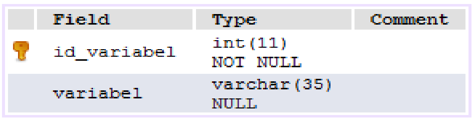
2.3. Application Description Model

Application description model includes the data model and process specification

Data Model

Table of Variable. This is used to store the variable data

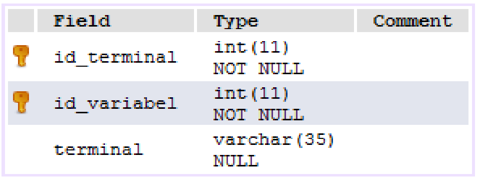
Table 5. Variable Table Structure



	Field	Type	Comment
🔑	id_variabel	int(11) NOT NULL	
	variabel	varchar(35) NULL	

Table of Terminal. This is used to store the terminal data

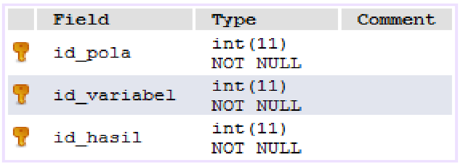
Table 6 : Terminal Table Structure



	Field	Type	Comment
🔑	id_terminal	int(11) NOT NULL	
🔑	id_variabel	int(11) NOT NULL	
	terminal	varchar(35) NULL	

Table of Pattern. This is used to store the pattern data.

Table 7. Pattern Table Structure



	Field	Type	Comment
🔑	id_pola	int(11) NOT NULL	
🔑	id_variabel	int(11) NOT NULL	
🔑	id_hasil	int(11) NOT NULL	

Table of Relation

The relation among the tables is shown in the following picture.

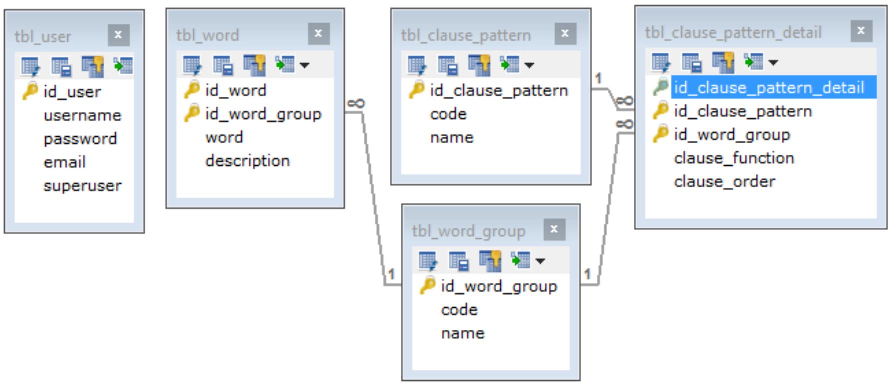


Fig. 6. Table of Relation

The data managed includes:

1. Words, a list of Indonesian vocabulary
2. Part of speech is part of speech of each word.
3. Clause Pattern is sentence patter structure as stated in CFG in Table 4. The samples of the data are SP, SPO, SPK, SPPel, etc.
4. Detail of Clause Pattern, is the detail of clause pattern previously stated. The sample of SP pattern can be formed by some combinations of part of speech. From the table 4, it can be formed sentences with some series combinations of part of speech, which are:
5. <Kal> → <S> <P>

<S>	→	<N>
<P>	→	<N> | <V> |<Bil>|<Ket>
<Kal>	→	<N><N> *SP1
<Kal>	→	<N><V> *SP2
<Kal>	→	<N><Bil> *SP3
<Kal>	→	<N><Ket> *SP4

SP1, SP2, SP3 and SP4 are some clause patterns which can be produced for SP clause pattern. On the SP1, a sentence consisting of two words with the first word is N and the second is also N. Next, a table of clause pattern detail is formed which consists of clause pattern id, part of speech id, clause function id, and its appearance sequence in the clause pattern.

Process Specification. The flow of how function run in the application shown in flowchart in the following figure 7

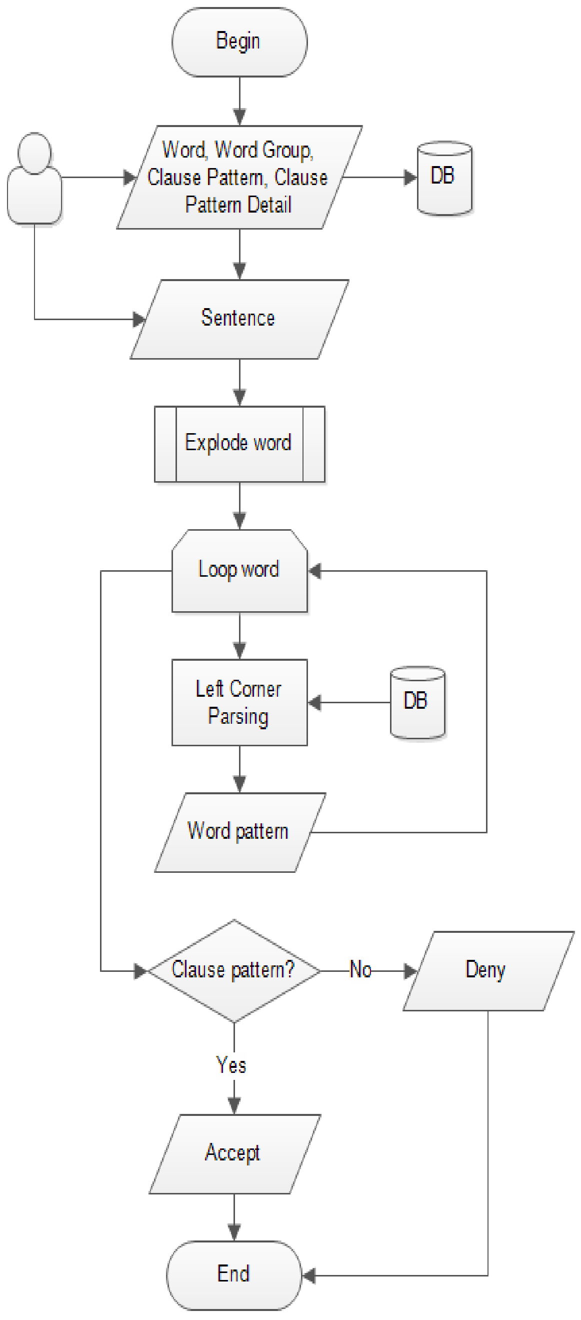


Fig 7. Flowchart

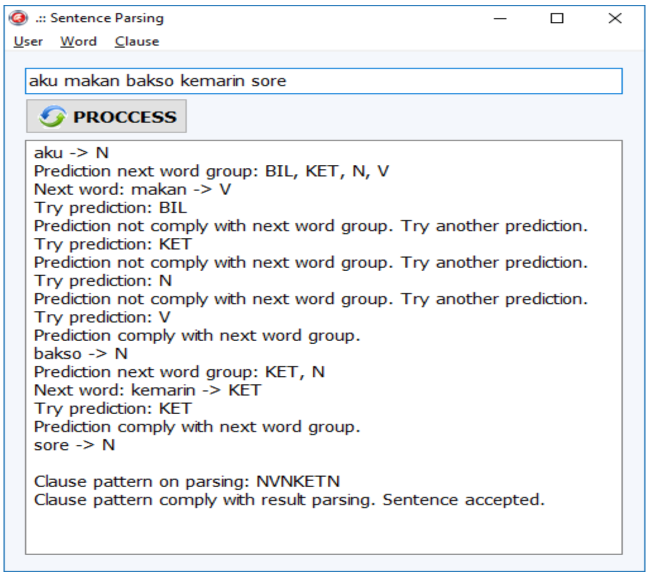
Pseudo Code Left Corner Parsing. The following pseudo code explains the logic of how the application runs.

```
Begin
input sentence //inputting a sentence to parse
explode sentence into word //breaking the sentence into some words (token)
    word_count = word count in sentence //initializing of the number of words to parse
    clause = null //initializing the clause from parsing

for i = 1 to word_count //repeating words from the first to the last (as the number of word obtained)
    get word //taking the word based on its organization in the sentence
    check word group //checking the word group from the taken words above
    label 1: //position to start backtrack
        predict next word group //predicting the next word group (referring to the previous
        parsing result)
        get next word //taking the next word from the sentence
        group //checking whether the next word group (as the clause pattern rule) matches the word
        group predicted by the parser
        join word group //if it matches, combine the word group obtained with the previous word
        groups
        clause = join word group //inputting  the result of word group combination into the clause
        variable
        else //if it doesn't match
        backtrack: go to label 1 //Conduct the backtrack process starting from the instruction line in
        the label 1 position
        end if //word group checking ends
    i = i + 2 //increment variabel i -> to the next word
end for //each word repeating ends
if clause = clause pattern //checking whether the clause from parsing matches the definite
clause pattern rule?
    sentence accept //if it matches, the sentence is accepted (matches the clause pattern)
else //if it doesn't match
    sentence deny //the sentence is denied (it doesn't match the clause pattern)
end if //The sentence clause pattern checking  ends
End
```

Fig. 8. Pseudo Code Left Corner Parsing

Program Implementation.The following figure 9 shows the application display for the sentence based on the rule stored.



```
aku makan bakso kemarin sore
```

```
aku -> N
Prediction next word group: BIL, KET, N, V
Next word: makan -> V
Try prediction: BIL
Prediction not comply with next word group. Try another prediction.
Try prediction: KET
Prediction not comply with next word group. Try another prediction.
Try prediction: N
Prediction not comply with next word group. Try another prediction.
Try prediction: V
Prediction comply with next word group.
bakso -> N
Prediction next word group: KET, N
Next word: kemarin -> KET
Try prediction: KET
Prediction comply with next word group.
sore -> N

Clause pattern on parsing: NVNKETN
Clause pattern comply with result parsing. Sentence accepted.
```

Fig. 9. Application Display for the sentence which matches the clause

Figure 10 shows the manual process of left corner parsing based on stored rules for the sentence which matches the clause.

Figure 11 shows the application display for the sentence which does not match the clause.

Figure 12 shows the manual process of left corner parsing based on stored rules for the sentence does not matches the clause.

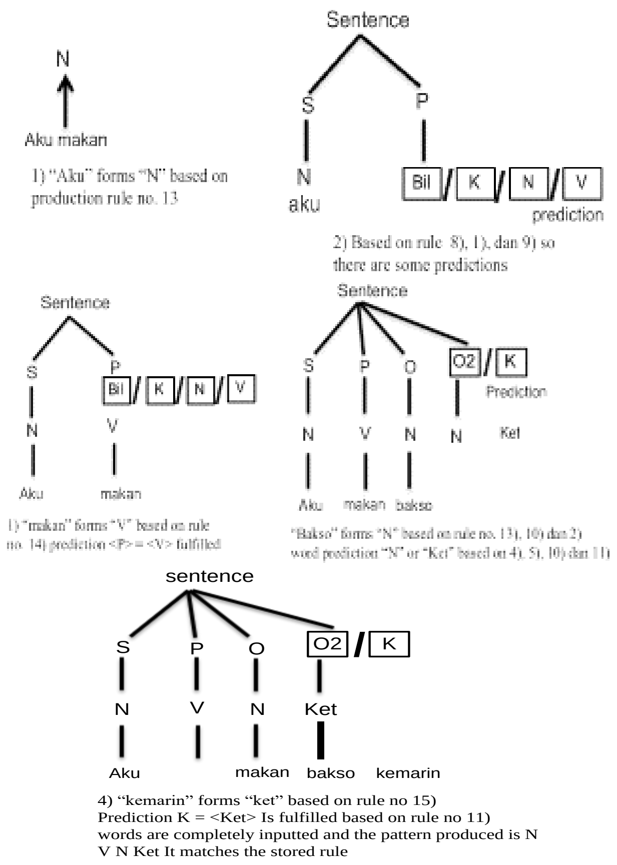
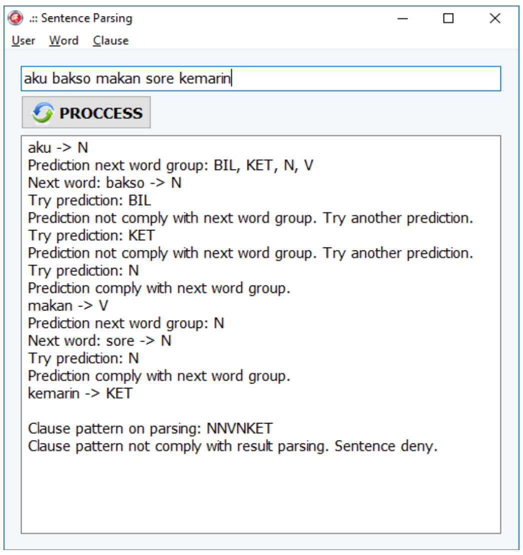

Sentence
N
Aku makan
1) "Aku" forms "N" based on production rule no. 13
S
P
N
aku
Bil
K
N
V
prediction
2) Based on rule 8), 1), dan 9) so there are some predictions
Sentence
S
P
Bil
K
N
V
N
V
Aku
makan
1) "makan" forms "V" based on rule no. 14) prediction <P> = <V> fulfilled
Sentence
S
P
O
O2
K
Prediction
N
V
N
N
Ket
Aku
makan
bakso
"Bakso" forms "N" based on rule no. 13), 10) dan 2) word prediction "N" or "Ket" based on 4), 5), 10) dan 11)
sentence
S
P
O
O2
K
N
V
N
Ket
Aku
makan
bakso
kemarin
4) "kemarin" forms "ket" based on rule no 15)
Prediction K = <Ket> Is fulfilled based on rule no 11)
words are completely inputted and the pattern produced is N V N Ket It matches the stored rule

Fig. 10. Manual Process of Left Corner Parsing for the sentence which matches the clause



```
aku -> N
Prediction next word group: BIL, KET, N, V
Next word: bakso -> N
Try prediction: BIL
Prediction not comply with next word group. Try another prediction.
Try prediction: KET
Prediction not comply with next word group. Try another prediction.
Try prediction: N
Prediction comply with next word group.
makan -> V
Prediction next word group: N
Next word: sore -> N
Try prediction: N
Prediction comply with next word group.
kemarin -> KET

Clause pattern on parsing: NNVNKET
Clause pattern not comply with result parsing. Sentence deny.
```

Fig. 11. Application Display for the sentence which does not match the clause

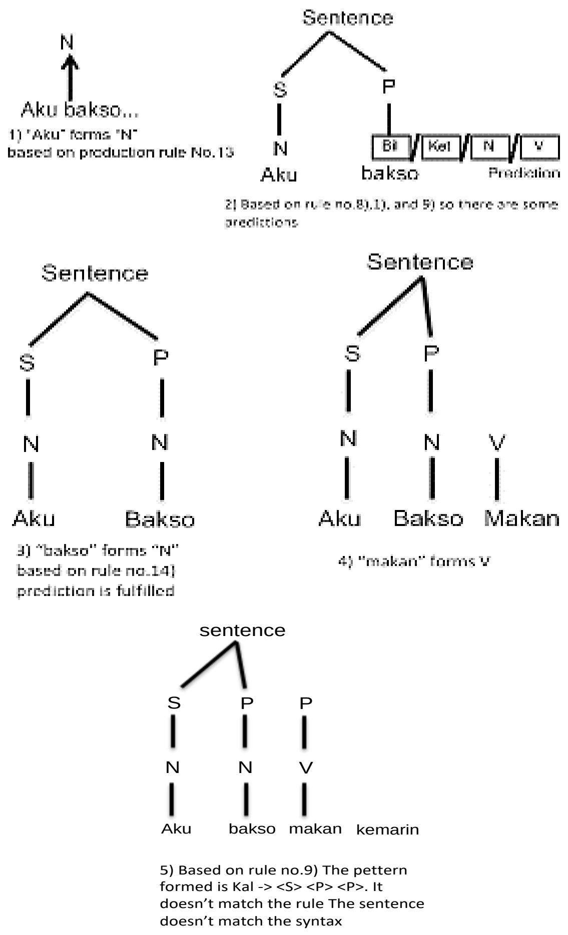

N
Aku bakso...
1) "Aku" forms "N" based on production rule No.13
Sentence
S
P
N
Aku
Bil
Kel
N
V
bakso
Prediction
2) Based on rule no.8),1), and 9) so there are some predictions
Sentence
S
P
N
N
Aku
Bakso
3) "bakso" forms "N" based on rule no.14) prediction is fulfilled
Sentence
S
P
N
N
V
Aku
Bakso
Makan
4) "makan" forms V
sentence
S
P
P
N
N
V
Aku
bakso
makan
kemarin
5) Based on rule no.9) The pettern formed is Kal -> <S> <P> <P>. It doesn't match the rule The sentence doesn't match the syntax

Fig. 12. Manual Process of Left Corner Parsing for the sentence which matches the clause

3 Conclusion & Suggestion

3.1. Conclusion

1. Left corner parsing can be implemented in sentence of Indonesian language.
2. The first word influences the derivation of the next sentence.
3. The number of production rules affect many types of patterns are recognized.

3.2. Suggestion

1. It can be developed for other language parsing by matching the pattern of the language clause.

2. It can be developed rules of parsing the broader pattern of clauses such as those involving the phrase

Acknowledgement. The author would like to thank the Kemenristekdikti Indonesia which has provided research funding through research grants Penelitian Dosen Pemula 2015.

References

[1] B. Patrut, “Syntactic Analytic Based on Morphological Characteristic Features of the Romanian Language.” .

[2] M. Abdurrohman, S. Hadi, and D. Rohidin, “Grammar Checking in English language Using Left Corner Parsing Algorithm,” in Proceeding of National Scientific Seminar on Computer and Intelligent system (KOMMIT2006).

[3] D. J. and J. H. Martin, Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistic and Speech Recognition. Prentice Hall Series In Artificial Intelligence, 1999.

[4] Puslit.petra.ac.id, “James Suciadi, Analysis Study on Parsing Method and Semantic Interpretation on Natural Language Processing,.” [Online]. Available: http://puslit.petra.ac.id/journals/informatics.

[5] D. W. Patterson, Introduction to Artificial Intelligence and Expert System. Prentice Hall International, Inc, 1990.

[6] S. . Tjiptadi, Bambang, Drs., Negoro, Summary of Indonesian language Grammar,. Yudhistira, 1983.

[7] Ramlan, Part of Speech. Yogyakarta: Andi Offset, 1985.