ZHU-TAKAOKA, *LEVENSHTEIN* AND CYK (*COCKE-YOUNGER-KASAMI*) ALGORITHM APPLICATION IN THE CASE OF CORRECTING WORDS AND STRUCTURE OF THE SENTENCE IN BAHASA INDONESIA

Aep Saepudin[1], Adam Mukharil Bachtiar[2]

[1,2] Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-114 Bandung
E-mail: aeps405@gmail.com[1], adam@email.unikom.ac.id[2]

ABSTRACT

Language is communication the most complete and effective way to convey ideas, messages, intentions, feelings and opinions to others. A grammar must be scientifically proven by every person, in every place and at every time. Therefore, an author must pay attention to writing and sentence structure in a language because writing and sentence structure will be influential in the process of delivery and receipt of a message. This is the underlying need for research on repair error writing words and an examination of the structure of the sentence.
The algorithm used in the correcting words and sentence structure improvements are Zhu-Takaoka algorithms, Levenshtein and cyk (Cocke-Younger-Kasami). Algorithm Zhu-Takaoka is one string matching algorithm. In the process of the search for the string of Zhu-Takaokas algorithm there are several stages, that is preprocessing, right-to-left scan, bad character rule and good-suffix rule. Levenshtein algorithm is an algorithm used to search for the least string operation to transform the string into another string. Cyk algorithm is an algorithm that can recognize whether a string can be considered as a sentence or not. From those three algorithms, will do the implementation, so knowing the extent to which Zhu-Takaoka, Levenshtain and cyk algorithm can provide recommendations for improvements and correcting word and structure of a sentence.
Based on the results of tests performed, The result of the implementation of Zhu-Takaoka, Levenshtein and cyk algorithm in case of correcting word errors and structural improvements in Indonesian language stating that the test results obtained word improvements at 76, 4%. As well as the testing of structures obtained a value of 84, 21%. It can be concluded that the three algorithm is good enough in correcting words and correcting structures of a sentence.

Keywords : Language, Algorithm, Zhu-Takaoka, CYK, Levenshtein, Repair World, Repair Structure.

1. INTRODUCTION

1.1 Background

Language is communication the most complete and effective way to convey ideas, messages, intentions, feelings and opinions to others [1]. A grammar must be scientifically proven by every person, in every place and at every time [2].

Therefore, an author must pay attention to writing and sentence structure in a language because writing and sentence structure will be influential in the process of delivery and receipt of a message. This is the underlying need for research on repair error writing words and an examination of the structure of the sentence.

Based on the research of Alinda and evita Kania dewi on "comparison of Reverse Algorithm Algorithm with Colusi Zhu-takaoka in Matching strings. Algorithm of Zhu-Takaoka and Reverse Colusi has as many as 100% accuracy in matching strings, then Reverse Colussi Algorithm takes an average of preprocessing for 15 ms and the average time a matching pattern for 0.13 ms Algorithm while Zhu Takaoka takes an average of 5.43 during preprocessing ms and average time matching pattern for 0.05 ms. It can be concluded that the development of the second algorithm Boyer Moore, Takaoka Zhu algorithm is faster on the process of matching strings [3] Zhu-takaoka, the algorithm can only match the right word can not improve so that the required algorithm that can improve the levenshtein algorithm.

Based on the research of Ni Made Muni Adiyani, I Wayan Santiyasa, Agus Muliantara on "implementation of the Levenshtein Distance Algorithm and Empirical Method to display the Suggestion Fixes Typos Documents speak in Indonesia", there is a way to help overcome the problems on typos of words [4]. Because this research is still limited to checking typos rather than checking the structure of the sentence, then the required algorithm for checking the structure of the sentence.

Based on the research of Sri Susanti on "Comparative analysis of the algorithm for LCP (Left-Corner-Parsing) and CYK Algorithms (Cocke-

Younger-Kasami) to examine patterns of Sentences Baku Indonesian Language", there is a way to recognize sentence structure by using LCP and CYK algorithms. From the results of testing conducted CYK Algorithm has an accuracy of 65% with the rate of error of 0, 35 and LCP Algorithms have an accuracy of 60% with the rate of error is 0, 40 [5]. So it can be inferred that the CYK Algorithm better recognize sentence structure.

Based on the above issues, this research will be done implementation algorithm of Zhu-takaoka, Levenshtein and CYK to correct the mistake of writing the words and sentence structure improvement of language Indonesia on the abstract thesis. Algorithm of Zhu-Takoaka used for matching words, Levenshtein algorithm used to Repair Word and correct sentence structure and CYK algorithm used to identify sentence structure.

1.2 Problem Identification

Based on the background which has been described, then the outline of the research problem was how to implement algorithms Zhu-Takaoka, Levenshtein and CYK for error writing words and sentence structure improvements in the language of Indonesia.

1.3 Purpose and Objective

As for the intent of this research is to apply the algorithm Zhu-Takoaka, Levenshtein and CYK on simulator for repair error writing words and sentence structure.

The purpose of this research is to know the algorithm Zhu-Takaoka, Levenshtain and CYK may give recommendations repair error writing words and sentence structure in the language of Indonesia on the abstract thesis.

1.4 Limitations

Restriction problems on algorithms Zhu-Takaoka, Levenshtein and CYK in case of repair the mistakes of writing words is as follows.

1. Checking limited to just the language of Indonesia.

2 Dictionary based on the POS Tag Indonesia.

3 documents reviewed in the form of the input file with the extension .doc or docx.

4 Checking the sentence just simple sentences and sentences.

5 Checking the Word does not include recognizing words ambiguous.

6 the test Data used is document abstract thesis of elib Unikom.

7 programming language used is C #.

1.5 *Text Mining*

Text mining which is also referred to as Text Data Mining (TDM) or Knowledge Discovery in Text (KDT), generally refers to the process of extracting information from text documents is not structured (unstructured). Text mining can be defined as the discovery of new information and is not known in advance [6].

The main purpose of text mining (text mining) is to support the process of knowledge discovery in large document collections. In principle, the text mining is a multidisciplinary science, involving information retrieval (IR), text analysis, information extraction (IE), categorization, clustering, visualization, database technology, natural language processing (NLP), machine learning, and data mining. It can also be argued that the text mining is a form of artificial intelligence applications (artificial intelligence/AI).

1.6 Syntactic

he syntax of the Setup and discuss the arrangement of words into larger units, called syntactic units, i.e. words, phrases, clauses, sentences, and discourse [7].

1.6.1 Phrase

The phrase is a unit of syntax that is composed of two or more words, which in clause occupied syntactic functions [7]. The phrases are divided into several groups:

1. Noun Phrase Koordinatif
   This phrase can be composed of two words noun category which is a partner and relational antonyms.
2. Noun Phrase Subordinatif
   This phrase can be composed of noun + noun (N + N), noun + verb (N + V), noun + ajektifa (N + A), adverb + noun (Adv + N), noun + adverb (N + Adv), noun + numeralia (N + Num), noun + numeralia (Num + N), and noun + demonstratifa (N + Dem).
3. Verb Phrases Koordinatif
   This phrase can be composed of:

a. two word verb categories is a member of relational antonyms and grammatical meanings of ' merge ' so that it can be inserted in between the word ' and '.

b. two word verb categories that are members of a field has a meaning and grammatical meaning of the ' merge ' so that it can be inserted in between the word ' and '.

4. Verb Phrases Subordinatif

This phrase can be composed of verb + adverb (Adv + V), verb + adverb (V + Adv), verb + noun (V + N), verb + ajektifa (V + A).

5. The phrase Ajektifa Koordinatif

This phrase can be composed of:

a. two words categorized ajektifa who is a member of relational antonyms and grammatical meanings of ' choice ', so that it can be inserted in between the two words ' or '.

b. two categories ajektifa words that are members of a couple are synonymous, and grammatical meaning ' very '.

c. two word ajektifa meaning categories in line does not contradict and grammatical meanings of ' set ' so that it can be inserted in between the word ' and '.

d. two categories of words ajektifa meaning not in line (opposed) and has the meaning of ' reverse ' so that between the two different chemical element should be inserted the word ' but '.

6. The phrase Ajektifa Subordinatif

The phrase ajektifa subordinatif compiled with the structure of A + N, A + A, A + V, Adv + A, and A + Adv.

7. The phrase Preposisional

Preposisional phrases are phrases that function as the function description in a clause. The phrase preposisional is open koordinatif or subordinatif phrase, but rather a eksosentrik phase. So, in this phrase there is no element of core and additional elements. Both elemental is one unified whole. Preposisional phrases composed of words categorized prepositions and Word or phrase noun categories.

1.6.2 Clause

A clause is a unit of syntax that is predictive. That is, in units of construction or that there is a predicate, if there is no unit in the predicate, then it is not a unit called a clause. The position of the predicate is very important, because the types and categories of predicate that is what determines the presence of a function of the subject (S), the function of the object (O), complementary functions, etc. [7]. Based on categories that fill the P function can be differentiated into:

1. Clause Verbs (KV)

Semantically there are three types of verbs, i.e. verbs action verbs, verbs, and incident circumstances.

a. Clauses Verbs of action

Targeted action verb clause is not berpelengkap can be composed of a verb berkomponen meaning to (+) and (+ target), so the klausanya function syntax has the S, P, O in this component of the meaning of V that fills the function of P must be in line with the meaning of components belonging to the functions of the S and O.

Targeted action clause berpelengkap may be composed of a verb berkomponen meaning (+ action), (+ target), and (+ complement), so klausanya has a function of S, P, O, and the Mop. In this case of course components meaning owned P must be in line with the meaning of components that belong to other functions.

Action verbs clauses not targeted can be composed of a verb has a meaning component (+) and (-target), so that klausanya only have the function of S and P functions. In this case of course components meaning owned P must be in line with the meaning of components owned S.

b. Clause Verb Events

Clause verb events can be reconstructed from the predicate verb has a meaning component (+ Genesis). In terms of syntactic functions that must be present is a function of S and P functions. The functions of the S form of nouns that are experienced as mentioned by the predicate.

Verbs Clauses Circumstances c.

Verbs clauses circumstances can be composed of predicate verbs that have a meaning component (+ State). In this case the function syntax that appears is merely a function of S and P functions. The functions of the S form of nouns that are experiencing such circumstances mentioned by the predicate.

2. Noun Clause

Noun clause only has a mandatory function of S and P this noun Clause. can be composed from function S in the form of a word or phrase and noun category P in the form of a word or phrase noun categories. Noun clauses, among others, can be arranged if:

a. Noun that fills the function of S is a kind of (specific) of the nominal charger function P (generic).

b. Noun that fills the function of S has a name on a nominal charger function P.

c. the Nominal charger function P is a profession (term, employment) for the nominal charger function S.

d. Noun charger function P is a relation to nominal function of the charger S.

e. Noun charger function S distinctive traits or characteristics mentioned by noun charger function S.

3. Clause Ajektifa

Clause ajektifa has a mandatory function of S and p. ajektifa Clause can be composed from function S categories N and P categories a. ajektifa Clauses can be drafted if:

a. the function P categories ajektifa have the meaning component (+ physical state).

b. Function P categories ajektifa have the meaning component (+ inner nature).

c. Function P categories ajektifa have the meaning component (+ inner feelings).

4. Clause Preposisional

Clause preposisional clause is a function P is filled by his phrase preposisional.

5. Clause Numeral

The clause is a clause that the numeral function P is filled by his phrase numeral..

1.6.3 Sentences

The sentence is a syntactic Unit assembled from its constituent base, which usually come with a conjunction of clauses, when needed, as well as accompanied by final intonation. Final intonation can be denoted by comma, exclamation mark, or question mark [7].

1. Simple sentences

A simple sentence is a sentence that is formed from a simple basic clause or clauses, i.e. clauses that sintaksisnya only functions filled by a simple

word or a phrase. Example: dad reading a newspaper.

2. Broad Sentence

A sentence usually make for much more information on it and referred to with the sentence. Example: a night with brother secretly reading comics in the room.

1.7 *Chomsky Normal Form* (CNF)

CNF is one normal form is very useful for CFG. The shape of the CNF can be made from a CFG that has undergone simplification by removing useless, production units, and empty (an empty set denoted by ε) [8]. An example CFG that will be visible on the oversimplified rules (1).

$S \rightarrow bC \mid aB \mid AB$
$A \rightarrow bAC \mid a$
$B \rightarrow aCB \mid b$
$C \rightarrow cBA \mid c$ (1)

The results of the production rules can be seen on the rules (2).

$S \rightarrow V_1C$
$S \rightarrow V_2B$
$S \rightarrow AB$
$A \rightarrow V_1V_3$
$A \rightarrow V_2$
$B \rightarrow V_2V_4$
$B \rightarrow V_3$
$C \rightarrow V_5V_6$
$C \rightarrow V_3$
$V_1 \rightarrow b$
$V_2 \rightarrow a$
$V_3 \rightarrow AC$
$V_4 \rightarrow CB$
$V_5 \rightarrow c$
$V_6 \rightarrow BA$ (2)

1.8 POS Tag

The POST (Part-of-Speech) Tag is a designation of class words, such as nouns, verbs, adjectives, etc. POS Tagging is an activity annotate each word/token with the value of the part-of-speech tags accordingly. POST this Tag value information is fundamental to the purposes of processing human language [9]. HEADING Tags are used as an aid in identifying token/tag each word taken from the post Tag Indonesia. POST Tags will identify which words including nouns, verbs, adjectives, adverbs, prepositions, conjunctions, pronouns, words and numbers. POST this Tag uses a Rule-Based approach based on the grammar rules of Indonesia. First of all, the post Tag will do the tokenisasi of the text using English-Indonesia. The next words that belong to the type of closed-class word processing. Then every ambiguous word processed using the rules defined for the class to find the right words.

Table 1 Production rules

Name of The Group Word	Converted Into
Noun / Kata Benda (N)	NN, NND, NNP,
	PRP, atau WH
Adjective / Kata Sifat (A)	JJ
Verb / Kata Kerja (V)	VB
Adverb / Kata Keterangan (Adv)	RB
Numeralia (Num)	OD dan CD
Preposisi (P)	IN
Demonstrative (Dem)	PR
Konjungsi koordinatif	CC
Artikula	DT
Verba bantuan	MD
Kata ingkar	NEG
Partikel	RP
Konjungsi subordinatif	SC
Simbol	SYM
Tanda baca	Z

2. CONTENTS

2.1 Architecture System

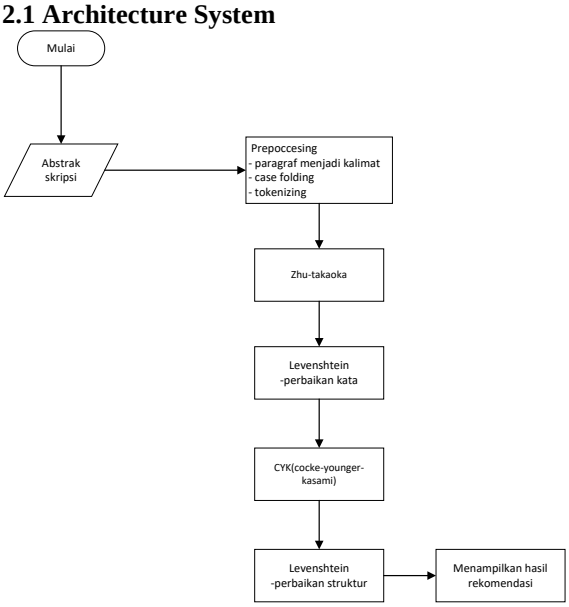


figure 1 Architecture System

2.1 Analysis Data Input

The input data for this form of simulator abstract the thesis with extension .doc or docx.

2.2 Analysis Preprocessing

How to used in studying a text data, is to first determine the features that represent each word for every feature that exists on the document. Before determining the features that represent a preprocessing stage is required, which is done in General in the text mining on document [6], namely solving paragraph into sentences, case folding, and tokenizing.

2.4.1 Process Breakdown Paragraf Into Sentence

Table 2 Process Breakdown of the Paragraf Into sentence

Document	Result Breakdown of the Paragraf Into sentence	
Teknologi permainan dalam	No	Sentence
	1	Teknologi

Document	Result Breakdown of the Paragraf Into sentence	
kemajuan dari sisi desain nenarik. Teknologi pada objek krena desain baik satu kemajuan.		permainan dalam kemajuan dari sisi desain nenarik.
	2	Teknologi pada objek krena desain baik satu kemajuan.

2.2.2 Case Folding

Table 3 *Case folding*

Before	After
Teknologi permainan dalam kemajuan dari sisi desain nenarik.	teknologi permainan dalam kemajuan dari sisi desain nenarik.
Teknologi pada objek krena desain baik satu kemajuan.	teknologi pada objek krena desain baik satu kemajuan.

2.2.3 Tokenizing

Table 4 *Tokenizing*

Sentence	Character
teknologi permainan dalam kemajuan dari sisi desain nenarik.	– teknologi – permainan – dalam – kemajuan – dari – sisi – desain – nenarik – .
teknologi pada objek krena desain baik satu kemajuan.	– teknologi – pada – objek – krena – desain – baik – satu – kemajuan – .

2.3 Analysis Word Search Zhu-Takaoka Algorithm

Algorithm of Zhu-Takaoka is one of string matching algorithm (String Matching) which is a development of the Boyer-Moore algorithm. This algorithm was published by Zhu Rui Feng and Tadao Takaoka in 1986 [3]. In his research, Zhu and string matching algorithm calls Takaoka as BM "Algorithm (the Boyer-Moore Algorithm) because it is a string matching algorithm is a modification of the Boyer-Moore algorithm, that is made by R.S. Boyer and Moore J.S. Algorithm Zhu-Takaoka and Boyer-Moore algorithm has the same characteristics in the process of a search string, i.e. There is a Preprocessing stage, Right-to-left scan, Bad character rule and Good-suffix rule [3].

2.3.1 Create Table ZtBC

To simplify the process of finding a table ztBC the first step is to find the value of OH, i.e. by identifying any displacement, ztBC table before charging.

Table 5 First OH

Index	0	1	2	1
Character	d	a	r	i
Value OH	?	?	?	?

To populate the value of OH, the first step is to do the counting in every character in the pattern. Counting starts from the 2nd character far right (Index = Length (pattern)-2) from the pattern. For the first step of displacement are filled with numbers 1.

Table 6 Enumeration first

begin : *Index = length(pattern) – 2* = 4 – 2 = 2				
Index	0	1	2	3
Character	d	a	r	i
Move = 1				

Compare each Compare every character against character tattooed on table ztBC before, if the character is not found in the table of tattooed, then add those characters into the ztBC and value of OH (Occurrence Heuristic) filled with a number of characters who are being moved in comparison. To the first comparison, compare the first character with Null.

Table 7 The first comparison

Compare d with *null*		Hasil	
ztBC(awal)		ztBC(akhir)	
Karakter	null	Karakter	r
Nilai OH	null	Nilai OH	1

Table 8 Result final value OH

Index	0	1	2	3
Character	d	a	r	i
Value OH	3	2	1	4

2.3.2 Create table ZtGS

After getting the final result on the chart ztBC, then continued with the make table suffix/suffix from the pattern.

Table 9 Early ZtGS

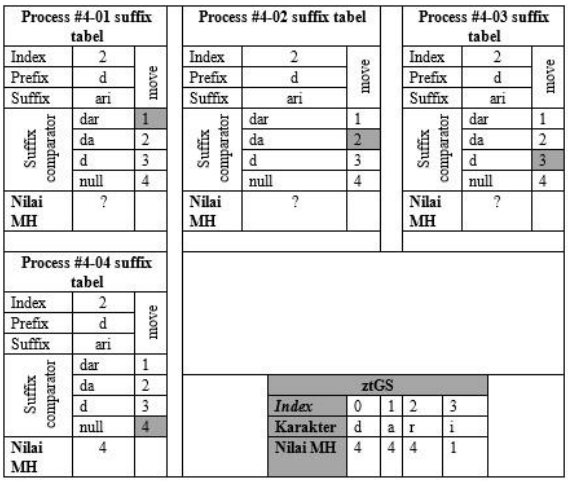
ztGS				
Index	0	1	2	3
Character	d	a	r	i
Value MH	?	?	?	?

Table 10 The first process ZtGS

Process #1-01 suffix tabel		
Index	3	Move
Prefix	i	
Suffix	null	
Suffix comparator	dar	1
	da	2
	d	3
	null	4
Nilai MH		

ztGS				
Index	0	1	2	3
Karakter	d	a	r	i
Nilai MH	?	?	?	1

Table 11 End ZtGS



2.3.3 Word Search

Source word : baikdalamdaridesainkarenakerenkemajuanmenariobjekpadapermainansatusisiteknologi

Pattern : dari

Verification 1

Index	1	2	3	4	5	6	7	8	9
Source word	b	a	i	k	d	a	l	a	m
Pattern	d	a	r	i					

From the results of matching, it can be seen that the final character of the source string that is searched for the characters i align with the character k. that is, in the 1st match happens mismatch, then do shift as far as four characters. Value shift of 4 is obtained from the ztBC table in which two characters dicocokannya to 12th are not the same. Then check on ztBC table row i and column k then it is worth 4.

Verification 5

Index	17	18	19	20	21	22	23
Source word	d	a	r	i	d	e	s
Pattern	d	a	r	i			

From the results of the 5th match, it can be seen that the final character is sought i.e. is parallel as to the character of the character i i in the dictionary. This means that on the 5th match took place a match, then matching is done against the character before the character a, r, a, d on the source Word, compare with parallel characters happen to match. The algorithm then marks the location of the discovery and to continue matching use shift. then the next shift of value taken from table ztGS with the d character then shift as far as 4 characters.

2.4 Repaird Word by Levenshtein Algorithm

Levenshtein algorithm is a search algorithm number of the found string difference by Vladimir Levenshtein, Russia, a scientist in 1965. This algorithm is widely used in various fields, such as a search engine, spelling checker (spell checking), the introduction of talk (speech recognition), the pronunciation of the dialect, DNA analysis, and others. Two pieces of string comparison to find out the value of the edit distance. The Edit distance is a measurement (metrics) that are generated through the calculation of the amount of the difference in the two strings [4].

Each word is considered misrepresentation will be compared with each word in the library that was made later searched the edit distance value is the smallest with Levenshtein algorithm.

Examples of words “nenarik” compared with “menarik”.

Table 12 Comparison “nenarik” end “menarik”

		n	e	n	a	r	i	k
	0	1	2	3	4	5	6	7
m	1	1	2	3	4	5	6	7
e	2	2	1	2	3	4	5	6
n	3	2	2	1	2	3	4	5
a	4	3	3	2	1	2	3	4
r	5	4	4	3	2	1	2	3
i	6	5	5	4	3	2	1	2
k	7	6	6	5	4	3	2	1

Examples of words “nenarik” compared with “menari”.

Table 13 Comparison “nenarik” dan “menari”

		n	e	n	a	r	i	k
	0	1	2	3	4	5	6	7
m	1	1	2	3	4	5	6	7
e	2	2	1	2	3	4	5	6
n	3	2	2	1	2	3	4	5
a	4	3	3	2	1	2	3	4
r	5	4	4	3	2	1	2	3
i	6	5	5	4	3	2	1	2

Table 14 Result comparison word

Word	Suggestions For Improvement	Value edit distance
nenarik	menari	2
	menarik	1

2.5 Algorithm of Cocke-Younger-Kasami (CYK)

CYK algorithm is an algorithm for parsing and membership (membership) for context-free grammar. The requirement for using this algorithm is grammar must be in the form of the CNF [5]. The purpose of this algorithm is to demonstrate whether a string can be received from a grammar.

The first step examination pattern of CYK algorithm with sentence is doing the charging line of two-dimensional CYK algorithm table by looking at the variables in the two blocks, then views which produces both variable variables. The next block is filled with the same steps in accordance with the blocks will be filled. Charging on the second block

is done by taking the variables of the second and third columns.

Table 15 Charging block

Sentence	Block	Tag 1	Tag 2	Production rules
Ke 1	[1,1] & [1,2]	NN	NN	FNN
	[1,2] & [1,3]	NN	IN	-
	[1,3] & [1,4]	IN	NN	FP
	[1,4] & [1,5]	NN	IN	-
	[1,5] & [1,6]	IN	NN	FP
	[1,6] & [1,7]	NN	NN	FNN
	[1,7] & [1,8]	NN	VB	FNV
	[1,8] & [1,9]	VB	Z	-

As already described the explanation about the CYK algorithm steps then the final result of the examination of the pattern with the Indonesia language sentence CYK algorithm.

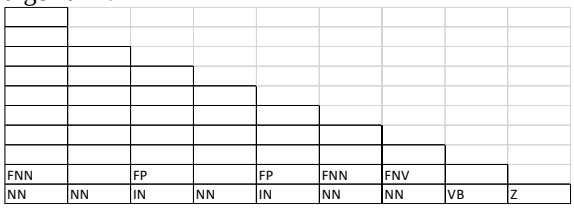


Figure 2 Result of the Final Examination

2.5 Levenshtein Algorithm

Any structure deemed erroneous would be compared to any existing structure on a library that is already made and then look for the smallest edit distance value with the Levenshtein algorithm.

An example of the structure of the SC NN NN NN IN JJ CD NN Z compared with "NN IN the SC NN NN NN CD Z", "NN NN NN NN CD Z SC" and "NN IN the SC NN NN NN JJ IN Z"

Table 16 Result comparison structure

Comparison	Structure	Edit distance
1	NN IN NN SC NN CD NN Z	6
2	NN NN SC NN CD NN Z	9
3	NN IN NN SC NN IN NN JJ Z	9

3. TEST

3.1 Accurasy Test

On this test used as many as 5 abstract with a total of 92 words and incorrect sentence structure 24 were considered incorrect. The documents tested is the document that is already checked by the Grammarian. The correct sentence structure is stored in a document to compare the results with the system. Recommendations for improvement are taken from the repaired structure. In the early stages is carried out using the word checking algorithms zhu-takaoka. This algorithm will recognize the words in the document.

3.1.1 Test Word

This test aims to see how many words that can be improved. The first stage of the pencarain word in the dictionary library using an algorithm of Zhu-Takaoka. Further testing is done by comparing a word wrong with library dictionary. This stage is also a test of the extent to which the Levenshtein algorithm is able to find the appropriate words. The value of the edit distance used is the smallest value of the comparison.

Table 17 Test result word

No	Name Document	Number of Word Error	Total number of word appropiate
1	abstrakskripsi01	13	9
2	abstrakskripsi02	12	10
3	abstrakskripsi03	10	6
4	abstrakskripsi04	23	20
5	abstrakskripsi05	31	23
Total number		89	68

From the test results there are 89 words got improvement. There are 21 words that can't be fixed. The number of words that got the corresponding improvement exist 68. The value of testing the success of repair Word can be measured in different ways:

$$\frac{\text{Total number of word appropiate}}{\text{Number of Word Error}} \; x\,100$$

$$\frac{68}{89}\; x\,100 = 76,4\%$$

The test results obtained a value of 76, 4% of the Levenshtein algorithm for testing in a fix said.

3.1.2 Test Structure of the Sentence

This test aims to see how much the sentence structure can be given recommendations. The first step to recognize the tags of each production rule and recognize the word of each sentence by using the algorithm of CYK. Further testing is done with the wrong sentence structure benchmarking with the library structure. This stage is also a test of the extent to which the Levenshtein algorithm is able to find the appropriate sentence structure. The value of the edit distance used is the smallest value of comparative sentence structure. Testing an improved sentence structure can be seen in Appendix E.

Tabel 18 Amount of a given sentence recomendation

No	Name Document	Number of structure that considered error	Total number of structure appropiate
1	abstrakskripsi01	3	3
2	abstrakskripsi02	4	1
3	abstrakskripsi03	2	2
4	abstrakskripsi04	7	7
5	abstrakskripsi05	3	3
Total number		19	16

In table 18 can be seen that the amount of the appropriate sentence on the sentence pattern with review by grammarian totaled 16 sentences from 19 erroneous sentence. The value of his success can be measured in different ways:

$$\frac{\text{Total number of structure appropiate}}{\text{Number of structure that considered error}} x\ 100$$

$$\frac{16}{19} x\ 100 = 84,21\%$$

The test results obtained a value of 84, 21% of the Levenshtein algorithm for testing in a match pattern of sentence used to repair recommendations.

4. PENUTUP

4.1 CONCLUSION

Based on research on the application of the algorithm of Zhu-Takaoka, levenshtein and CYK. Testing managed to find out that Zhu-Takaoka algorithms can search for a Word with good, CYK algorithm can recognise sentence structure with both the rules of production and the Levenshtein algorithm is also very good in fixing the words and sentence structure. The results of this research indicate that repair Word 76.4% of value obtained and the sentence structure test obtained value of 84, 21%. This value is already good enough in fixing the sentence structure and Word that ill-advised but there is still a lack of research on the implementation of the Levenshtein algorithm to determine this since the improvement of the value of the smallest edit distance, cannot recognize sentence structure which should be in the snap or split, cannot recognize words that should be merged or separated and dictionary POST Tags used are still a lot of words that are not recognized.

4.2 Suggestions

There are some suggestions from this study in order to be better in the future, including:

1. Can use approximate string matching algorithm on a better Repair Word and sentence structure improvements.
2. Can use a better algorithm on inspection of the structure of the sentence.
3. Use the POST Tags another more word list because it is still a lot of words that don't exist, to increase improvement recommendations said.

5. REFERENCES

[1] Walija, “Bahasa Indonesia dalam Perbincangan,” Jakarta, IKIP Muhammadiyah Jakarta Press, 1996.

[2] P. D. Iswara, “Penelitian Variasi Pola Kalimat,” 24 Februari 2009. [Online]. Available: http://iswara.staf.upi.edu/2009/02/24/penelitian-variasi-pola-kalimat.. [Diakses 19 September 2016].

[3] Ramdani. P.P, “Analisis Perbandingan Performansi Algoritma Zhu-Takaoka dan Algoritma Karp-Rabin pada Pencarian Kata di Rumah Baca Buku Sunda,” 2013. [Online]. Available: http ://elib.unikom.ac.id.

[4] N. Made muni Adriyani, I. Wayan santiyasa dan A. Muliantara, Implementasi Algoritma Levenshtein Distance dan Metode Empiris untuk Menampilkan Saran Perbaikan Kesalahan Pengetikan Dokumen Berbahasa Indonesia, pp. 1-7.

[5] Susanti, Sri, “Analisis Perbandingan Algoritma LCP(Left-Corner-Parsing) dan Algoritma CYK (Cocke-Younger-Kasami) untuk Memeriksa Pola Kalimat Baku Bahasa Indonesia,” Ilmu Komputer dan Informatika (KOMPUTA), 2015.

[6] F. Muhammad, I. Veri, N. Muhammad, “Implementasi Algoritma Dynamic Programming untuk Koreksi Kesalahan Ejaan Kata Pada Search Engine,” 2015.

[7] A. Chaer, Sintaksis Bahasa Indonesia (Pendekatan Proses), Jakarta: Rineka Cipta, 2009.

[8] F. Utdirartatmo, Teori Bahasa dan Automata, Yogyakarta: Graha Ilmu, 2005.

[9] A. Dinakaramani, F. Rashel, A. Luthfi dan R. Manurung, “POS Tag Indonesia,” 20 October 2014. [Online]. Available: http://bahasa.cs.ui.ac.id/postag/corpus. [Diakses 20 March 2015].