Time Complexity of Knuth Morris Algorithm and Rejang Algorithm in Rejang-Indonesian Translator

Sastya Hendri Wibowo[1*], Rozali Toyib[2], Yulia Darnita[3], Satria Abadi[4]

[1,2,3]*Informatics Engineering Study Program, Muhammadiyah University of Bengkulu, Indonesia*

[4]*Computing and Meta-Technology, Universiti Pendidikan Sultan Idris, Malaysia*

*corr_author: sastiahendriwibowo@gmail.com

Abstract - Among the pattern-matching algorithms is the Knuth-Morris algorithm. In order to minimize the number of comparisons required and, in the worst scenario, achieve an ideal O(n+m) running time, the Knuth-Morris search algorithm skips unneeded comparisons. Every character in the text and every character in the pattern must be checked at least once by the pattern-matching algorithm. The Knuth-Morris algorithm's primary goal is to preprocess the pattern string P in order to determine the failure function f, which displays P's precise shift, so that earlier comparisons can be reused. In order to extract the fundamental word of the attached sentence, words containing affixes are separated using the Rejang stemming method. The purpose of this research is to determine the time complexity of the Rejang method and the Knuth-Morris algorithm based on affix groups. The Rapid Application Development (RAD) approach, which entails planning, designing, building, and implementing, is used during the research stages. The research results have produced efficient and effective Knuth Morris algorithm and Rejang algorithm, where efficiency is indicated by the algorithm time complexity of O (log n), and effectiveness is indicated by the accuracy results of 99% against testing 6000 affixed words.

Keywords: Knuth Morris algorithm, Rejang algorithm, time complexity

I. INTRODUCTION

Indonesia is rich in culture, one of which is regional Speechs. Regional Speechs in Indonesia differ in structure and vernacular, such as the Minang Speech in West Sumatra, the Batak Speech in North Sumatra, the Sundanese Speech in West Java, and the Rejang Speech in Bengkulu. Regional Speechs oral in Indonesia, including the Rejang Speech oral in Bengkulu, the Sundanese Speech oral in West Java, the Batak Speech oral in North Sumatra, and the Minang Speech oral in West Sumatra, differ in structure and Vernacular. Every province undoubtedly has a printed regional Speech dictionary, albeit not all of them have been created as digital dictionaries [1], [2]. In the current era of computer technology, it is essential to build a digital dictionary that contains regional Speech vocabulary, an application for translating words and sentences from one regional Speech into Indonesian and vice versa, making it simpler for people to understand the meaning of the words and sentences and facilitating communication between speakers of different regional Speechs, The Rejang Speech is one of the regional tongues orals in Indonesia. About 500,000 native speakers of the Rejang Speech speak it as a regional Speech in Rejang Lebong District and the neighboring regions of Bengkulu Province and South Sumatra Province. The Malay or Indonesian Speech family includes the Rejang Speech. In addition to its structured structure, the Rejang Speech uses the Kaganga script as its writing system. Several researchers have conducted research on the Rejang Speech from a morphological or linguistic perspective, such as a) S. Nafsin et al. in 1981 on the Structure and Syntax of the Rejang Speech, b) Saleh Y. in 1988 on the Morphological System of the Rejang Speech, c) R. Atrizai in 1994 on the Syntax of the Coastal Vernacular of Rejang Speech, d) SF Wibowo in 2016 on Segmental Phonemes and Their Distribution in the Rejang Vernacular. In addition, research has also been conducted on the development of algorithms or combinations of other algorithms in the Rejang Speech, such as: a) The Journal of Advanced Research in Dynamical and Control Systems published a paper by Sastya Hendri Wibowo in 2019 titled "Development of Stemming Algorithm for Rejang Speech." The system was initially utilized to find root words, b) In the Journal of Computational and Conceptual Nanoscience, Sastya Hendri Wibowo's 2019 paper Spelling Checker of Words in Rejang Speech Using the N-Gram and Euclidean Distance Methods describes how to combine the N-Gram and Euclidean Distance methods to check spelling in Rejang Speech words, c) Sastya Hendri Wibowo in 2022 Time Complexity in Rejang Speech Stemming published in the Infotel Journal regarding the calculation of the complexity of the Rejang stemming algorithm, and d) In 2023, Sastya Hendri Wibowo submitted a paper in the Journal of Information System Research (JOSH) on NLP-Based Information Systems for Preserving Local

Culture in Bengkulu Province. The paper tested the Rejang stemming method in information systems [3]-[6]. All of these studies are still in the testing or development stage of the method and have not been implemented in the form of an application system that is widely used and utilized by the community, especially in Bengkulu province, this has an impact on the lack of public interest in getting to know, learn and preserve the Rejang Speech. One of the causes is the lack of access for the community to information about the Rejang Speech.

Based on the above problems, the problem-solving approach is to create an application system that can provide access to the public to understand the Rejang Speech such as a) Rejang Digital Dictionary, which functions as a database containing all Rejang vocabulary, b) Rejang to Indonesian Speech translator, which functions to understand the meaning of Rejang vocabulary, c) Information about the Rejang Speech that will display the history of the Rejang Speech, and the Kaganga Rejang letters. The latest system designed is based on Artificial Intelligence (AI) using the Rejang algorithm and the Knuth Morris algorithm for the Rejang-Indonesian Speech translator, meaning that the system will combine two algorithms, namely the Rejang stemming algorithm and the Knuth Morris algorithm which function to truncate words with affixes and match patterns in sentences, as well as the addition of chatbot features and voice recognition as support which is part of Artificial Intelligence (AI). The general objectives of this study are: a) preserving the local cultural wisdom of the Rejang Speech, b) helping the community to easily know, learn, and understand the Rejang Speech, c) creating a Rejang-Indonesian Speech translator application system by combining the Rejang stemming algorithm and the Knuth Morris algorithm based on AI. Specifically, to obtain the time complexity of the Knuth-Morris and Rejang algorithms.

II. METHOD

A. Research Steps

This research uses the Rapid Application Development (RAD) method, which is intended for the short term according to the system or application being developed. The Rapid Application Development (RAD) flow is shown in Fig. 1. The steps are as follows [7]-[10]:

a) Planning: Identifying problems in the community through community leaders in Rejang Lebong district. Field surveys were conducted in the Rejang Lebong district where the majority of people use the Rejang Speech as a Speech of communication. The team involved in this stage is the Head, Members, and Rejang traditional leaders

b) Design: The system design is made in the form of a flowchart and interface. There are three algorithms used at this stage, namely the Rejang Stemming algorithm which functions to separate or chop words that are combined into basic words, the Knuth Morris algorithm which functions to check grammatical patterns contained in a word, and the Natural Speech algorithm Processing used for chatbot features and voice recognition. The team involved in this stage is the Chair and Members

c) Construction: The design that has been made is then implemented into a programming Speech that aims to create a system in the form of an application, namely an interface and database. The team involved in this stage is the Chairperson and Members

d) Implementation: The system design that has been written in the coding is then tested and analyzed, to determine the success of the system that has been created. Testing is carried out on the algorithm and interface using Rejang Speech variables or data. The team involved in this stage is the Chairman, Members, and Rejang Traditional Figures.

The Knuth-Morris and Rejang algorithms are used in this study. Entering a word or sentence in the Rejang Speech and then verifying it in the database is the first step in translating it into Indonesian. The next step is string matching, which matches strings based on terms found in the Rejang database, if the word or sentence is present in the database. The Rejang Stemming algorithm is used to cut words containing prefixes (men-, ke-, be-, ne-, te-, se-) and inserts (-em-, -en-), suffix (-ke), and combined (be- + ke-, te- + ke-, se- + ke-, ke- + -ke) based on Rejang Speech structure before translating words from the Rejang Speech into Indonesian or vice versa. Rejang Speech structure is used to determine suffixes (-ke) and combinations (be- + ke-, te- + ke-, se- + ke-, ke- + -ke) using the Rejang Stemming algorithm. A list of Rejang Speech affix words is shown in Table 2. Utilizing the Knuth Morris method, the definition of a word is displayed after the process of looking up the word in the Rejang digital dictionary. To ascertain a sentence's structure, Knuth Morris will examine the grammar of sentences or input data that is given as a text string. The Knuth-Morris technique yields a response indicating whether the text string in question is a member of the Speech used to explain a specific sentence. Additionally, Knuth Morris is used to detect grammatical faults in sentences and determine whether or not they make sense. The 6000 words in the traditional Rejang Speech dictionary, which was compiled by Rejang cultural specialists in Bengkulu, provide the data used as a digital dictionary for the Rejang Speech.

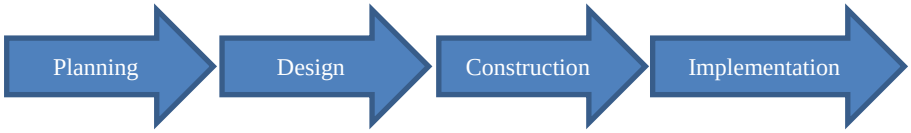

Fig. 1 Rapid application development (RAD) method

B. Performance Analysis of Rejang Algorithm

The Rejang Stemming Algorithm in this study is an algorithm developed from the Searching process to look up root words in the digital dictionary of the Rejang Speech [11][12]. The Binary Searching algorithm, with a temporal complexity of O (log n), was employed in this investigation. The number of root words in the Rejang Speech digital dictionary is represented by n. Checking the prefix, inserting the data, checking the suffix, checking the union, and checking the third procedure make up the main procedure. The searching procedure is used by every path in each sub-routine. Thus, Maximum [4 O (log n), O (log n), O (log n), O (log n)] = 4 O (log n) = O (log n) is the maximum value of the time complexity of the deletion method 1, 2, 3, or 4. This is the time complexity of the main procedure. Fig. 2(a) shows that the prefix deletion technique has a difficulty of 4 (log n), while Fig. 2(b) shows that the complexity of the main deletion procedure is = O (log n).

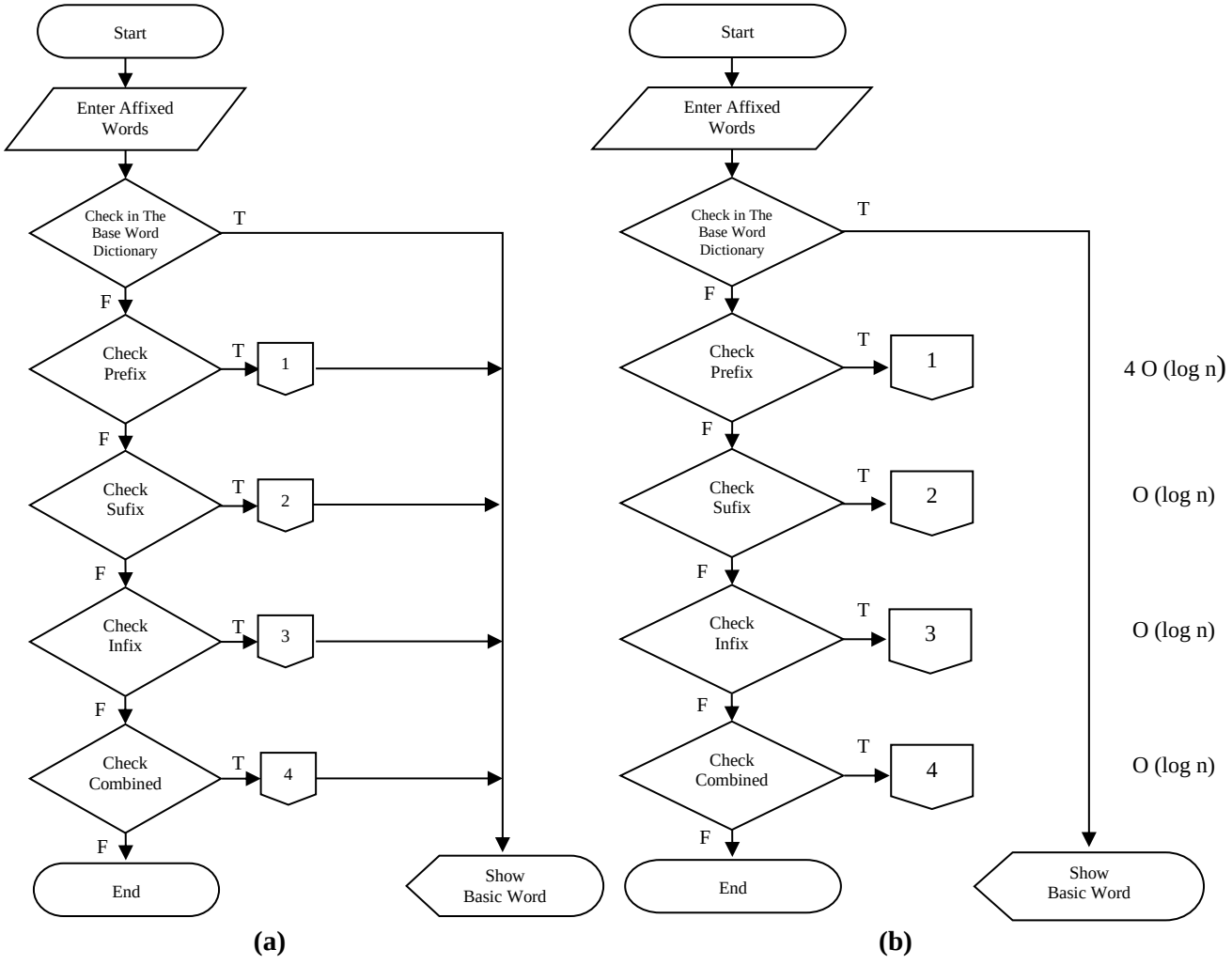

Fig. 2 (a) The complexity of the main procedure of deletion is = O (log n), (b) The complexity of the Prefix removal procedure is 4 (log n)

The study of the time complexity computation of the deletion procedures 1, 2, 3, and 4—which are explained below—provides the difficulty of the algorithm (main procedure) of 4 O ((log n). The procedures 1A (check prefix men), 1B (check prefix ke), 1C (check prefix be), 1D (check prefix te), and 1F (check prefix ke) make up Subprocedure 1 (Check prefix). Every path within each sub-process employs the searching procedure. Thus, process 1's time complexity is equal to the total of the MAXIMUM value of the time complexity of procedures 1A, 1B, 1C, 1D, 1E, and 1F, or Maximum [4 O (log n), O (log n), O (log n), O (log n)] = 4 O (log n). The analysis of the time complexity calculation of the elimination methods 1, 2, 3, and 4—which are explained below—provides the complexity of the algorithm (main process) of 4 O ((log n).

Procedures 1A, 1B, 1C, 1D, and 1F are the components of subprocedure 1 (check prefix), which is the check prefix men, check prefix be, check prefix te, and check prefix to. Every path within each sub-process employs the searching procedure. Consequently, procedure 1's time complexity is equal to the total of the time complexity of the Maximum value of the time complexity of procedures 1A, 1B, 1C, 1D, 1E, and 1F, or Maximum [4 O (log n), O (log n), O (log n), O (log n), O (log n)] = 4 O (log n). Fig. 3 illustrates that the prefix removal process has a 4 (log n) complexity.

Procedures 2A (check insertion em) and 2B (check insertion en) make up subprocedure 2 (Check insertion). Every path within each sub-process employs the searching procedure. Thus, process 2's time complexity is equal to the maximum of 2A and 2B's time complexity, or Maximum [O (log n), O (log n)] = O (log n). The insertion-deletion procedure's complexity (log n) is shown in Figure 4(a), and the -Ke Suffix Procedure's time complexity (log n) is shown in Fig. 4(b).

The procedures for the Vowel Prefix and Consonant Prefix are part of Sub-procedure 3 (Check prefix ke). All of the paths in each sub-process use a search procedure. Thus, process 3's temporal complexity stems solely from that of the Vowel Prefix or Consonant Prefix Removal procedures. Therefore, process 3's temporal complexity is O (log n).

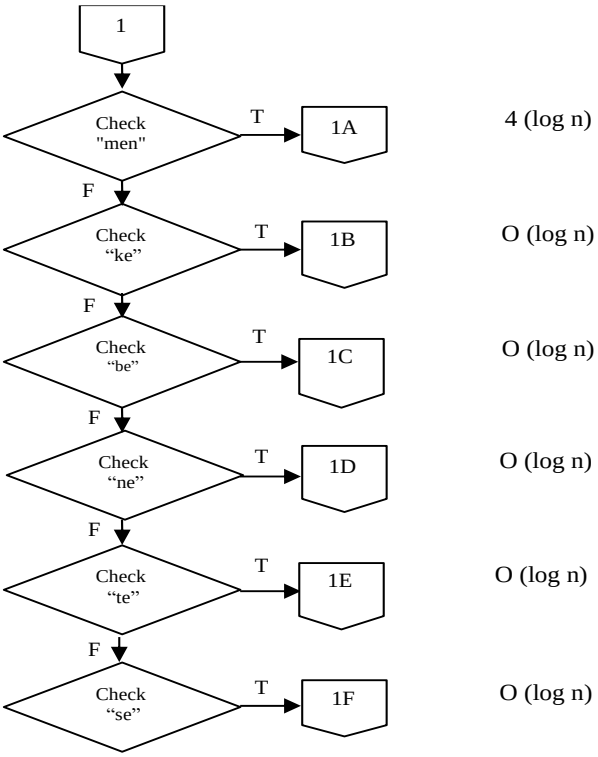

Fig. 3 Complexity of the deletion procedure the prefix is 4 (log n)

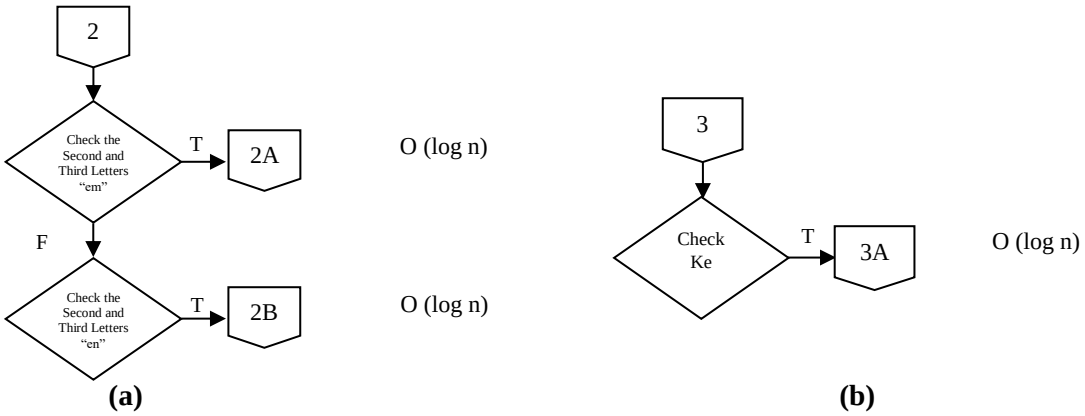

Fig. 4 (a) The complexity of the insertion-deletion procedure is (log n), (b) The Time Complexity of the -Ke Suffix Procedure is 1 (log n)

C. Performance Analysis of Knuth Morris Algorithm

The Knuth-Morris Algorithm is a pattern-matching algorithm used in this investigation. In order to minimize comparisons and, in the worst scenario, achieve an ideal O(n+m) running time, the Knuth-Morris search method removes unneeded comparisons. Every character in the text and every character in the pattern must be checked at least once by the pattern-matching algorithm. The Knuth-Morris algorithm's primary concept is to preprocess the pattern string P in order to compute the failure function f, which displays P's precise shift and allows the program to reuse previously conducted comparisons [13]-[15]. The Binary matching algorithm, whose temporal complexity is O (log n), was employed in this work to match patterns. The number of root words in the Rejang Speech digital dictionary is represented by n. Procedures 1 (prefix matching), 2 (insertion matching), 3 (suffix matching), and 4 (combination matching) make up the primary procedure. Every path within each sub-process employs the matching procedure. Then, the maximum value of the time complexity of the deletion operations 1, 2, 3, or 4—that is, Maximum [4 O (log n), O (log n), O (log n), O (log n)] = 4 O (log n) = O (log n—is the time complexity of the main procedure. In Fig. 5(a), the prefix removal procedure's complexity is expressed as = O (log n), whereas in Fig. 5(b), it is expressed as 4 (log n).

The study of the time complexity computation of the deletion procedures 1, 2, 3, and 4—which are explained below—provides the difficulty of the algorithm (main procedure) of 4 O ((log n). Prefix matching is the first subprocedure, and it includes procedures 1A (prefix matching men), 1B (prefix matching ke), 1C (prefix matching be), 1D (prefix matching te), and 1F (prefix matching ke). Every path makes use of the matching procedure in each sub-process. Thus, procedure 1's time complexity is equal to the total of the Maximum value of the time complexity of procedures 1A, 1B, 1C, 1D, 1E, and 1F, or Maximum [4 O (log n), O (log n), O (log n), O (log n)] = 4 O (log n). Procedures 2A (em insertion matching) and 2B (en insertion matching) make up subprocedure 2 (insertion matching). Every path within each sub-process employs the matching procedure. Thus, process 2's time complexity is equal to the maximum of 2A and 2B's time complexity, or Maximum [O (log n), O (log n)] = O (log n). The insertion-deletion procedure's complexity (log n) is shown in Figure 6(a), while the -Ke Suffix Procedure's time complexity (log n) is shown in Fig. 6(b).

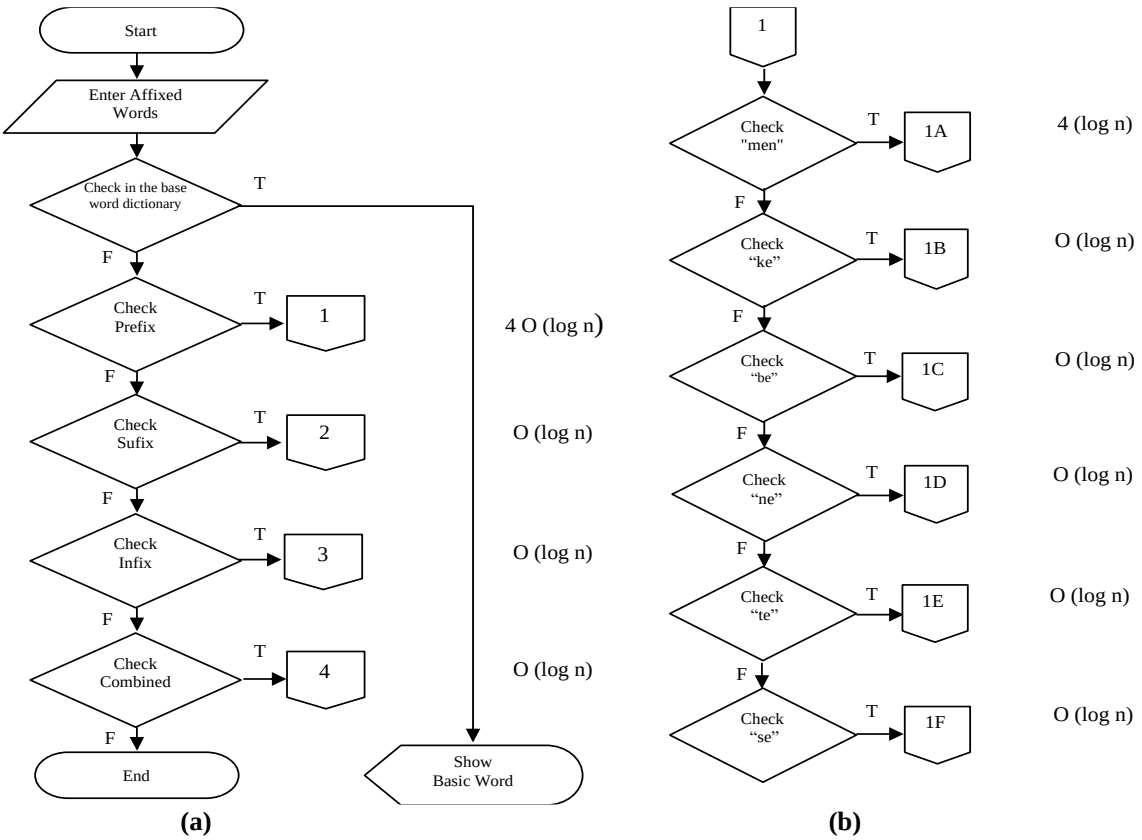

Fig. 5 (a) The complexity of the main procedure of deletion is = O (log n), (b) the complexity of the Prefix removal procedure is 4 (log n)

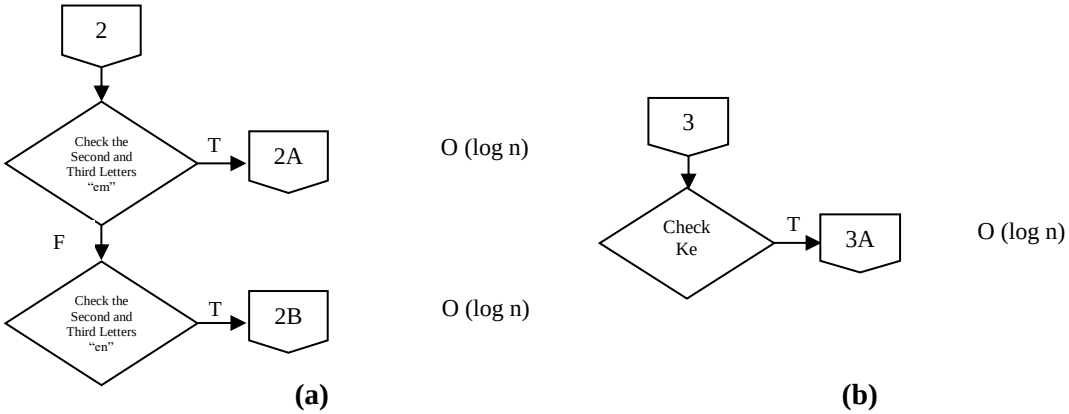

Fig. 6 (a) The complexity of the insertion-deletion procedure is (log n), (b) the time complexity of the -Ke suffix procedure is 1 (log n)

The Vowel Prefix procedure and the Consonant Prefix procedure make up Subprocedure 3 (Matching prefix to). Every path in a sub-process employs a matching procedure. Therefore, process 3's time difficulty solely stems from the Vowel Prefix or Consonant Prefix Removal technique's time complexity. Procedure 3's temporal complexity is therefore O (log n). Procedures 4A (Joint Matching Beke), 4B (Joint Matching Teke), 4C (Joint Matching Seke), and 4D (Joint Matching Keke) make up Sub-Procedure 4 (Joint Matching). Every path within each sub-process employs the matching procedure. The highest possible number of the time complexity of 4A, 4B, 4C, and 4D, or Maximal [O (log n), O (log n), O (log n), O (log n)] = O(log n), is the time complexity of operation 4. The level of complexity of the merging deletion procedure is shown in Fig. 7 as O (log n).

III. RESULT AND DISCUSSION

A. Testing Affixed Words

There are 6000 fundamental terms in the digital Rejang Speech dictionary. This study's algorithm testing was doneon 6,000 common words. Table I lists some simple word instances that were chosen for testing.

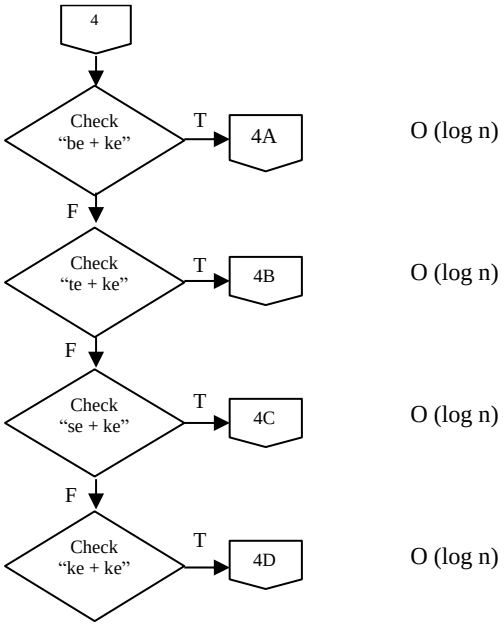

Fig. 7 Complexity of merge deletion procedure is O (log n)

TABLE 1
EXAMPLES OF BASIC REJANG SPEECH WORDS USED FOR TESTING

No.	Group Basic Words	Example
1.	Basic words starting with a vowel	Asen, Abau, Es, Enteng, Inat, Idau, Oak, Oloa, Okoa, Ulok, Ulek, Uku Biyoa, Baleit, Cao, Coa, Dang, Das, Gelembung, Guak, Has, Han, Has, Jas, Jer, Jin, Jok
2.	Basic words starting with consonants	Lucut, Kecek, Keing, Lawen, Le, Leak, Makiak, Malapak, Manat, Nakut, Naliak, Naling, Panes, Pangar, Patang, Ragam, Renyeng, Retek, Segit, Sekalit, Segan, Tegulek, Tekanak, Tekeung, Wakaf, Wakea, Waktau, Yau, Yung
3.	Basic words starting with the letter "K"	Kacea, Kaak, Kaang, Kabar

The Rejang-Indonesian Speech translator was used to test the Rejang and Knuth Morris method using 6000 basic words or affixes, which are basic words that begin with a vowel, a consonant letter, or the letter "k." Table 2 shows the total number of words containing affixes for each group. Table II displays the total number of words containing affixes for each group.

B. Basic Word Test Results

Table III displays the findings of evaluating a number of fundamental terms using the Rejang and Knuth Morris algorithm. The results of testing 6000 words are shown in Table IV, with success = 5975 and failure = 25.

IV. CONCLUSION

Eighteen groups/types of smallest affixes were identified in the Rejang Speech based on the results of the morphological study of the affix addition process. These were then used as conditional statements in the Rejang and Knuth Morris stemming method. It is known that the Rejang and Knuth Morris stemming algorithm sequentially removes affixes by first eliminating prefixes, then inserts, then suffixes, and lastly combinations. This knowledge stems from the results of examining and analyzing the structure of the process of adding affixes. The Rejang and Knuth Morris stemming algorithm's time complexity was successfully tested in this study, and the algorithm's usefulness was shown by its O(log n) time complexity and 99% accuracy rate when evaluating 6000 attached words.

TABLE II
NUMBER OF WORDS PER GROUP BASED ON BASIC WORDS STARTING WITH VOWELS, CONSONANTS AND 'K'

No.	Affixed Word Groups	Number of words starting with a vowel	The number of words that begin with a consonant	The number of words that begin with the letter "K"
1.	Prefix *M-*	0	600	0
2.	Prefix Me-	0	600	0
3.	Prefix *Men-*	0	600	0
4.	Prefix *Meng-*	0	600	0
5.	Prefix *Menge-*	0	600	0
6.	Prefix *Meng-*	600	0	0
7.	Prefix *Ke-*	0	0	600
8.	Prefix *Be-*	600	0	0
9.	Prefix *Ne-*	600	0	0
10.	Prefix *Te-*	0	600	0
11.	Prefix *Se-*	0	600	0
12.	Prefix *-em-*	0	600	0
13.	Prefix *-en-*	0	600	0
14.	Suffix *-ke*	130	370	0
15.	Combined Prefix *Be* + *Ke*	130	370	0
16.	Starting with a combination of *Te* + *Ke*	130	370	0
17.	Starting with a combination of *Se* + *Ke*	130	370	0
18.	Starting with a combination of *Ke* + Berakhiran *Ke*	130	370	0

TABLE III
TEST RESULTS FOR SEVERAL WORDS USING THE REJANG AND KNUTH MORRIS ALGORITHMS

No.	Affixed Words	Words being tested	Results	Identify Success	Failed to Identify
1.	Prefix M-, followed by a root word using a consonant first	Mdecit	Decit	√	-
		Mgusuk	Gusuk	√	-
2.	Starting with Me-, followed by a root word using a consonant first	Megusuk	Gusuk	√	-
		Menyengak	Sengak	√	-
3.	Starting with Men-, followed by a root word using a consonant first	Mengusuk	Gusuk	√	-
		Menanem	Tanem	√	-
4.	Starting with Meng-, followed by a root word using a consonant first	Mengusuk	Gusuk	√	-
		Mengacau	Kacau	√	-
5.	Starting with Meng-, followed by a root word using a consonant first	Mengekaba	Kaba	√	-
		mengekabeu	Kabeu	√	-
6.	Beginning with Meng-, followed by a root word Starting with the vowels	Mengike	Ike	√	-
		Mengokos	Okos	√	-
7.	Starting with Ke-, followed by a root word Beginning with a vowel	Mengike	Ike	√	-
		Mengokos	Okos	√	-
8.	Beginning with Be-, followed by a root word Beginning with a vowel	Mengike	Ike	√	-
		Mengokos	Okos	√	-
9.	Beginning with N- or Ne-, followed by a root word Beginning with a vowel	Neoloa	Oloa	√	-
		Neisut	Isut	√	-
10.	Beginning with Te- or T-, followed by a root word Beginning with a vowel	Tgamit	Gamit	√	-
		Tkecek	Kecek	√	-
11.	Starting with Se-, followed by a root word Starting with a consonant	Segamit	Gamit	√	-
		Sekecek	Kecek	√	-
12.	Infix-em-, which in the root word begins with a consonant	Wemen	Wen	-	√
		Cemco	Ceco	-	√
13.	Infix -en-, which in the root word starts with a consonant	Denko	Deko	-	√
		Cenco	Ceco	-	√
14.	-ke, suffix, which starts with a root word begins with a vowel or consonant	Itungke	Itungke	√	-
		Okoske	Okoske	√	-
15.	A root word beginning with a vowel or consonant, followed by the combination be- and ke-	Bekeibo	Ibo	√	-
		Bekeotos	Otos	√	-
16.	combination of the words "te-" and "ke-," followed by a vowel- or consonant-based root word	Tekeicang	Icang	√	-
		Tekeuleak	Uleak	√	-
17.	Se-and-the combination is followed by a vowel- or consonant-based root word.	Sekekicok	Kicok	√	-
		Sekelipet	Lipet	√	-
18.	a string that begins and ends with- and is inserted as a fundamental word that begins with a vowel or consonant	Kelemke	Kelemke	-	√
		Keosorke	Osor	-	√

TABLE IV
RESULTS OF 6000 WORD TEST (SUCCESS= 5975; FAIL= 25)

No.	Word Count	Number of Words Successfully Identified	Number of Failed Words Identified
1.	500 words Suffix M-, followed by a root word starting with a consonant	500	0
2.	500 words affixed with Me-, followed by a root word starting with a consonant	500	0
3.	500 words Affixed with Men-, followed by a root word starting with a consonant	500	0
4.	500 words Affixed with Meng-, followed by a root word beginning with a consonant	500	0
5.	500 words Affixed with Meng-, followed by a root word beginning with a consonant	500	0
6.	500 words Affixed with Meng-, followed by a root word beginning with a vowel	500	0
7.	500 words Suffixed to-, followed by a root word beginning with a vowel	500	0
8.	500 words Suffix Be-, followed by a root word beginning with a vowel	500	0
9.	500 words Suffix N- or Ne-, followed by a root word Beginning with a vowel	500	0
10.	500 words Suffix Te- or T-, followed by a root word Beginning with a vowel	500	0
11.	500 words Suffix Se-, followed by a root word that begins with a consonant	500	0
12.	500 words with the insertion of -em-, followed by a root word that begins with a consonant	497	3
13.	500 words affixed with -en-, followed by a root word that begins with a consonant	493	7
14.	500 words affixed with the suffix -ke, which begins with a root word begins with a vowel or consonant	500	0
15.	500 words starting with be- and ke-, followed by a root word starting with a vowel or consonant	500	0
16.	500 words starting with te- and ke-, followed by a root word starting with a vowel or consonant	500	0
17.	500 words starting with a combination of se- and th-, followed by a root word starting with a vowel or consonant	500	0
18.	500 words with suffixes beginning with - and ending with -ke, which are inserted with basic words starting with vowels or consonants	483	15
		Total = 5975 (99 %)	Total = 25 (1 %)

REFERENCES

[1] S. Wibowo, B. Soerowirdjo, Ernastuti, and A. Tarigan, "Development of stemming algorithm for Rejang Speech stemmer based on rejang Speech structure," *J. Adv. Res. Dyn. Control Syst.*, vol. 11, no. 5 Special Issue, pp. 1858–1870, 2019.

[2] S. H. Wibowo, B. Soerowirdjo, Ernastuti, and A. Tarigan, "Spelling checker of words in Rejang Speech using the n-gram and Euclidean distance methods," *J. Comput. Theor. Nanosci.*, vol. 16, no. 12, pp. 5384–5395, 2019, doi: 10.1166/jctn.2019.8607.

[3] S. H. Wibowo, "Sistem Informasi Bahasa Rejang Berbasis Natural Speech Processing (NLP) Untuk Pelestarian Budaya Lokal," vol. 5, no. 2, pp. 426–433, 2024, doi: 10.47065/josh.v5i2.4270.

[4] S. H. Wibowo, R. Toyib, M. Muntahanah, and Y. Darnita, "Time complexity in Rejang Speech stemming," *J. Infotel*, vol. 14, no. 3, pp. 174–179, 2022, doi: 10.20895/infotel.v14i3.764.

[5] R. Sovia, S. Defit, Yuhandri, and Sulastri, "Development of natural Speech processing on structure-based Minangkabau Speech stemming algorithm," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 31, no. 1, pp. 542–552, 2023, doi: 10.11591/ijeecs.v31.i1.pp542-552.

[6] A. Sinaga, Adiwijaya, and H. Nugroho, "Development of word-based text compression algorithm for Indonesian Speech document," *2015 3rd Int. Conf. Inf. Commun. Technol. ICoICT 2015*, pp. 450–454, 2015, doi: 10.1109/ICoICT.2015.7231466.

[7] P. Beynon-Davies, C. Came, H. Mackay, and D. Tudhope, "Rapid application development (Rad): An empirical review," *Eur. J. Inf. Syst.*, vol. 8, no. 3, pp. 211–232, 1999, doi: 10.1057/palgrave.ejis.3000325.

[8] V. Kralev and R. Kraleva, "Methods and tools for rapid application development," *Proc. III Int. Sci. Pract. Conf. "Methodology Mod. Res. (March 29, 2017, Dubai, UAE)*, vol. 1, no. 4 (20), pp. 21–24, 2017.

[9] N. M. N. Daud, N. A. A. A. Bakar, and H. M. Rusli, "Implementing Rapid Application Development (RAD) methodology in developing practical training application system," *Proc. 2010 Int. Symp. Inf. Technol. - Syst. Dev. Appl. Knowl. Soc. ITSim'10*, vol. 3, pp. 1664–1667, 2010, doi: 10.1109/ITSIM.2010.5561634.

[10] D. Tudhope, P. Beynon-Davies, H. Mackay, and R. Slack, "Time and representational devices in rapid application development," *Interact. Comput.*, vol. 13, no. 4, pp. 447–466, 2001, doi: 10.1016/S0953-5438(00)00050-3.

[11] R. Setiawan, A. Kurniawan, W. Budiharto, I. H. Kartowisastro, and H. Prabowo, "Flexible affix classification for stemming Indonesian Speech," *2016 13th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol. ECTI-CON 2016*, 2016, doi: 10.1109/ECTICon.2016.7561257.

[12] W. B. Demilie, "Implemented Stemming Algorithms for Information Retrieval Applications," *J. Inf. Eng. Appl.*, vol. 10, no. 3, pp. 1–6, 2020, doi: 10.7176/jiea/10-3-01.

[13] R. D. Djati Pramono and S. Lorena, "Implementasi algoritma knuth-morris-pratt dalam aplikasi untuk penerjemahan idiom bahasa inggris 1),2)," pp. 1–6, 2015.

[14] O. Saputra, "Penerapan Algoritma Knuth Morris Pratt dalam Aplikasi Penerjemah Teks," no. 13510072, 2013.

[15] B. L. Pramudita, "Implementasi Algoritma Knuth Morris Pratt pada Alat Penerjemah Suara," no. 13511042.