 ISSN:2302-8149
e-ISSN:2540-9719

Application of the Finite State Automata (FSA) Method in Indonesian Stemming using the Nazief & Adriani Algorithm

[1]Lady Agustin Fitriana*,[2]Ali Mustopa, [3]Muhammad Rifqi Firdaus, [4]Rizka Dahlia
[1,3,4]Sistem Informasi, Teknik dan Informatika, Universitas Bina Sarana Informatika
[2]Teknik Informatika, Teknik dan Informatika, Universitas Bina Sarana Informatika
Jl. Kramat Raya, Senen, Jakarta Pusat, Jakarta, Indonesia
*e-mail: Lady.lag@bsi.ac.id

(***received*:** 20 Maret 2024, ***revised*:** 24 April 2024, ***accepted*:** 4 Mei 2024)

Abstract

Language is a communication tool commonly used in everyday life. Each country has a different language with predetermined rules. For instance, in the Indonesian language, there are approximately 35 official affixes mentioned in the Big Indonesian Dictionary. These affixes include prefixes (prefixes), infixes (insertions), suffixes (suffixes), and confixes (a combination of prefixes and suffixes). In Information Retrieval, there is a stemming process, which is the process of converting a word form into a base word or the process of transforming variant words into their base form. The theory of language and automata is the foundation of the computer science field that provides the basis for ideas and models of computer systems. In the implementation of the research, several stages were carried out, such as explaining the Nazief & Adriani stemming algorithm, finite state automata, creating pseudocode, and testing using a web-based system, resulting in affixed words becoming the correct base words with 20 affixed words. The results obtained from reading this web-based system, the base word "cinta" (love) used as a test yielded accurate results in accordance with the concept of the Nazief & Adriani stemming algorithm. There are some weaknesses in stemming from suffixes, and the solution is to perform stemming from the prefix position (Prefix).

Keywords: *Finite State Automata, Stemming, Indonesian, Nazief & Adriani Algorithm*

1 Introduction

Language is a very effective and reliable communication tool when living together in society. Language is not only a communication tool, but also a medium to convey opinions and arguments to others. Therefore, language plays an important social role in communicating with society as a whole [1]. Every country has a diverse language with its own set of rules, including the Indonesian language which is known for its richness in affixes. There are about 35 official affixes mentioned in the Big Indonesian dictionary. These are prefixes, infixes, suffixes and confixes (a combination of prefix and suffix)[2]. Information Retrieval (IR) is a process or system to identify, search, and retrieve relevant information from a collection of documents or sources. In this context, IR aims to present search results that meet user needs, separate relevant and irrelevant documents, and improve the efficiency of accessing information. In Information Retrieval, there is a stemming process, which is the process of converting a formed word into a base word.[3]. Stemming is a process at the preprocessing stage that recognizes root words by combining and completing variations of each word. In the context of language processing, stemming aims to simplify words to their basic form to improve the efficiency of word processing and information retrieval. This process is suitable for handling word changes that may occur in text documents, thus improving the accuracy and effectiveness of text analysis and information retrieval systems[4].

The confix-stripping algorithm designed by Jelita Asian is an improvement of the Nazief & Adriani algorithm in stemming the Indonesian language. The resulting update involves a formula in forming a word consisting of [[[DP+]DP+]DP+] root-word [[+DS][+PP][+P]]. An explanation of this formula can be found in reference[5].

 ISSN:2302-8149
e-ISSN:2540-9719

1. DP = Derivation Prefix (Prefix)
2. RW = Root Word
3. DS = Derivation Sufix (Suffix)
4. PP = Posessive Pronoun
5. P = Particle

Many studies using stemming algorithms are applied to Text Mining, especially sentiment analysis or categorization. For example, research conducted by Wahyu Hidayat et al on the effect of Nazief & Adriani Stemming on the Ratcliff/Obershelp algorithm in identifying the level of similarity between slang words and formal words. This study aims to determine the similarity between slang words and formal words so that it can determine the similarity of slang presentations. The datasets used come from Kaggle and Github with the results of the temming research from the Nazief & Adriani algorithm affecting the similarity of slang words and formal words with the most frequent distribution in the value range of 0.8-0.89 (80% - 89.99%) in three different tests with two types of datasets and affecting the level of string similarity precisely on the dataset from Kaggle at a similarity value of 1 (100%), which previously amounted to 40 data to 927 data. The dataset from Github also increased to a similarity value of 1 (100%), from having no data to 409 data[6].

Similar research was also revealed by Anton Yudhana et al who applied the Nazief Adriani Stemmer Algorithm in detecting data errors in Indonesian. This research shows in three pre-processing stages, namely tokenization, letter reduction, and filtering. After the pre-processing stage is complete, the system will call each word for the stemming process. The stemming results will be compared with the base words available in the database. If it does not match, then the word will be highlighted and considered as a wrong word. The first finding is that Nazief Adriani algorithm can detect word errors up to 100%. The second finding is that Nazief Adriani algorithm also detects non-word errors, with a detection accuracy of 97.464%[7]. Furthermore, previous research conducted by Ardiles Sinaga et al. aims to analyze the comparison between two stemming algorithms, namely the Nazief Adriani algorithm and the Arifin Setiono algorithm, to measure the performance of each algorithm through testing on 30 Indonesian text documents. The test results show that the performance of the Nazief Adriani algorithm is superior to the Arifin Setiono algorithm, with the highest average accuracy value of 97.73% and an average stemming process time of 20.17 seconds. On the other hand, Arifin Setiono algorithm has an average accuracy value of 94.37%, with an average stemming process time of 23.32 seconds[8].

In previous studies on the development of the Nazief & Adriani algorithm, no one has applied the Finite State Automata method. Stemming using Finite State Automata (FSA) is an approach in natural language processing to remove affixes from words by using a finite state automata model. In the context of stemming, FSA is used to represent rules and transitions between states that represent steps in the affix removal process[9]. This process involves the creation of automata that model the rules of the language to recognize and convert words into their base form. Each state in the FSA represents a stage in the stemming process, and the transitions between states describe the steps taken to remove affixes. By using FSA in stemming, language rules related to word formation can be interpreted into automata, which can then be executed to generate base words. This approach helps in automating the affix removal process, improves efficiency, and provides an organized structure in the morphological analysis of the language[10].

Modeling Indonesian stemming with Nazief & Adriani algorithm described in the form of Finite State Automata is expected to provide convenience for the people of Indonesia. This research aims to analyze words that have affixes with the base word 'Love' by applying the Nazief and Adriani stemming algorithm using the Finite State Automata (FSA) model. The word made as research material as many as 20 words from the basic word 'Love'.

 ISSN:2302-8149
e-ISSN:2540-9719

2 Literature Review

Research related to steamming using the Finite State Automata technique was conducted by Ni'mah, Suryaningrum and Arifin[10] Stemming is the process of getting the base word by separating its affixes. Research on Indonesian Stemming has produced two types, namely without a dictionary and using a dictionary. Stemming without a dictionary is sometimes inaccurate, while using a dictionary takes a long time and cannot remove affixes in compound words. This study proposes a new dictionary-less algorithm that uses the Finite-State Automata method. The test results show an average accuracy of 66% with a speed of 0.0051 seconds and is able to remove affixes in compound words.

Other research using the Finite State Automata technique was conducted by Mulyana, Suhendra, Ernastuti, Bheta Agus, and Bheta Agus[11] This research aims to develop a new algorithm called UG18 Stemmer for Indonesian language, focusing on overcoming over-stemming and under-stemming that commonly occur in existing stemming algorithms. Using a morphophonemic approach, the algorithm replaces the list of deletion rules with morphophonemic-based elimination rules. Evaluation shows that the UG18 Stemmer has a lower error rate than the NECS Stemmer, with a reduction of over-stemming from 1.48% to 0.12% and complete elimination of under-stemming. In addition, this algorithm improves the processing speed in document similarity measurement by 45.47% compared to ECS Stemmer.

Further research conducted by Rianto, Mutiara, Wibowo and Santosa[12] Stemming is used to retrieve information by tracing prefixed words back to their roots. However, in non-formal Indonesian, existing stemming methods are still limited. This study introduces a new stemming method and aims to improve the accuracy of the text classification model. Using the Support Vector Machine algorithm, a text classification model is developed and evaluated using 550 datasets in Indonesian. Findings: The new stemming method resulted in higher text classification model accuracy than the existing methods, with scores of 0.85 and 0.73, respectively. Similar research was also revealed by Dyah, Ida, and Arifin [13] who conducted Stemming to determine the basic word in Indonesian. This research compares three stemming algorithms, namely Sastrawi Algorithm, Nazief & Adriani Algorithm and Arifin Setiono Algorithm using a sample of 900 affixes. Each word is identified using the algorithm, then the results are compared with the Big Indonesian Dictionary (KBBI) to check the accuracy. The comparison results show that the Sastrawi algorithm produces the best stemming, where 95.2% of the tested affix words can be returned to the base word .

Another research was conducted by Afian, Aris and Rahmat [14] who compared seven Indonesian stemming algorithms and one English stemming algorithm namely Nazief, Arifin, Fadillah, Asian, Enhanced Confix Stripping (ECS), Arifiyanti, and Porter. The data used were 2,734 tweets collected from PLN's official Twitter account. The main objective of this study is to analyze the correlation between stemming accuracy and information retrieval performance in Indonesian text language. The results show that the correlation found in previous studies does not apply to Indonesian. In addition, the proposed algorithm proved to be the best for Indonesian text processing purposes, with a weighted score value of 0.648. The next research was conducted by Merta, Agus, and Ardwi who applied Nazief and Adriani algorithm for stemming Balinese text. In this research, a special stemming algorithm for Balinese language has been developed called Bastal algorithm. Bastal algorithm is a modification of the proven Nazief & Adriani algorithm. The research method used is research and development using the waterfall model. The Bastal algorithm has been successfully implemented in a Balinese translator application that can be downloaded and used on Android smartphones. The test results show that this application works very well, with an overall accuracy rate of 96.15%. In addition, this application also received positive feedback from users[15]. The next research conducted by AAn, IGL Putra, Dedy and Dodik who applied the Nazief & Adriani stemming algorithm with the Cosine Similarity method for Telegram Chatbot Integrated with E-services. In this study it is used to minimize the level of similarity of common questions asked by users, so that using Chatbot can help questions automatically. With the help of the Nazief & Adriani algorithm for stemming and the cosine similarity method to find the level of similarity between user questions and FAQs available in the database, this research will implement a chatbot using the Telegram Messenger service, so this method is considered effective in answering questions automatically[16].

 ISSN:2302-8149
e-ISSN:2540-9719

3 Research Method

In this study there are stages in the design of Indonesian stemming using the Finite State Automata method starting from an understanding of the Indonesian stemming algorithm using Nazief & Adriani to the last stage, namely testing (deployment) The following is an image of the steps of the research steps Figure 1:

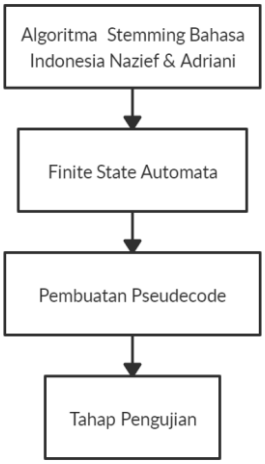

Figure 1. Research Stages of Indonesian Stemming Design Using FSA[17]

A. Nazief & Adriani Indonesian stemming algorithm

The Nazief & Adriani algorithm created by Bobby Nazief and Mirna Adriani will be applied to applications that have the following stages:

1. For the first step, input data in the form of words (in the application in the form of strings).
2. Check the database.
3. Remove Particles (P) ("-lah", "-kah", or "-pun").
4. Remove Possesive Pronouns (PP) ("-my", "-mu", or "-his").
5. Remove Derivation Suffixes (DS) ("-i", "-an" or "-kan").
6. Remove Derivation Prefix (DP).
7. Check the database for data validation.
8. Perform Recoding.
9. If all steps have been completed but not successful, the initial word is assumed to be the root word (RW). Process completed

B. Finite State automata

Finite State Automata will describe the model/behavior in a system. FSA is also useful in stemming in the Nazief & Adriani algorithm. FSA has 2 types of automata, namely Deterministic Finite Automata (DFA) and Non Deterministic Finite Automata (NFA). DFA is an automata that cannot be in more than one state at the same time. While NFA is an automata that has more than one status at the same time. In this research, the form that will be used is NFA. The description of the FSA is as follows Figure 2 :

 ISSN:2302-8149
e-ISSN:2540-9719

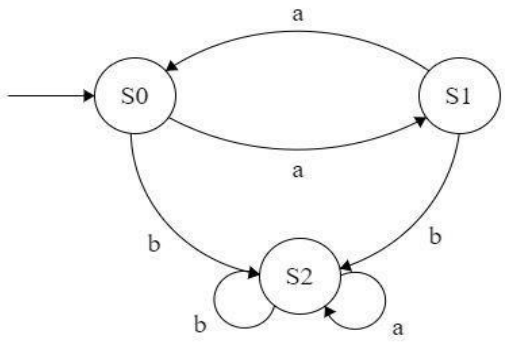

Figure 2. Diagram State FSA

1. The circle represents the state.
2. A circle with an unlabeled arrow at the beginning represents the initial state.
3. A double circle represents the final state.
4. An arrow indicates a state transfer.
5. The label on the circle indicates the state name.

C. Pseudocode

Pseudocode is a coding stage with a language that is easy for humans to understand. The form is almost the same as coding in a program only not as complete as coding. Pseudocode is useful in making it easier for a programmer to create a program because it already has a logic flow that has been described in pseudocode.

Pseudocode usually consists of the following 3 things:

1. Program Declaration
   The program declaration is written with the structure:
   `Program <ProgramName>`
2. Dictionary
   Dictionary is a variable declaration that will be used in the program, and the writing format is `<VariabelName> : <type_data>`
3. Algorithm Description
   Declares the course of the algorithm

D. Testing (Deployment)

In this testing stage, the step that will be taken is to test the Nazief & Adriani stemming algorithm application based on Finite State Automata (FSA) that has been made. This application is implemented on the web with testing done by inputting a word with affixes into the application. The application will process the word by removing affixes, namely at the stopword stage, so that the results obtained are basic words. The basic word will be displayed along with the word that was previously inputted.

4 Results and Analysis

A. Algoritma Nazief & Adriani

It is used as the root algorithm for Indonesian text and has a higher percentage of accuracy than other algorithms. This algorithm is very necessary and important in the IR process of Indonesian documents.

In general, root words in Indonesian consist of combinations:

DP + DP + ROOT WORD + DS + PP

Keterangan:

 ISSN:2302-8149
e-ISSN:2540-9719

DP: derivational prefix
DS: derivational suffix
PP: prossesive pronoun
P: particle
RW: root word
The table below shows the process of stemming. The following is the process of the word truncation stage, on affixed words can be seen from the explanation in the Table 1 Below:

Table 1. Steps of Nazief and Adriani Algorithm

No	Step	Process	Root Word	Stemming	Suffix
1	1	Cek Database	Mencintai	-	Mencintai
	2	Buang Partikel-nya	Mencintai	Mencinta	Mencinta
	3	Buang Partikel –kan	Mencinta	Cinta	Cinta
2	1	Cek Database	Mencintainya	-	Mencintainya
	2	Buang Partikel-nya	Mencintainya	Mencintai	Mencintai
	3	Buang Partikel –kan	Mencintai	Mencinta	Mencinta
	4	Buang Partikel –kan	Mencinta	Cinta	Cinta
3	1	Cek Database	Mencinta	-	Mencinta
	2	Buang Partikel-nya	Mencinta	Cinta	Cinta
4	1	Cek Database	kecintaannya	-	kecintaannya
	2	Buang Partikel-nya	kecintaannya	kecintaan	kecintaan
	3	Buang Partikel –kan	Kecintaan	Kecinta	Kecinta
	4	Buang Partikel –kan	Kecinta	Cinta	Cinta
5	1	Cek Database	Percintaannya	-	Percintaannya
	2	Buang Partikel-nya	Percintaannya	Percintaan	Percintaan
	3	Buang Partikel –kan	Percintaan	Percinta	Percinta
	4	Buang Partikel –kan	Percinta	Cinta	Cinta

Keterangan:
1. Men-cinta-i
   - Derivation prefiks = Me
   - Root word = Cinta
   - Derivation sufiks = I
2. Men-cinta-i-nya
   - Derivation prefiks = Me
   - Root word = Cinta
   - Derivation sufiks = I
   - Posessive Pronoun= nya
3. Men-cinta

 ISSN:2302-8149
e-ISSN:2540-9719

- Derivation prefiks = Me
- Root word = Cinta

4. Ke-cinta-an-nya
   - Derivation prefiks = Ke
   - Root word = Cinta
   - Derivation sufiks = an
   - Posessive Pronoun= nya
5. Per-cinta-an-nya
   - Derivation prefiks = Per
   - Root word = Cinta
   - Derivation sufiks = an
   - Posessive Pronoun= nya

B. Indonesia Stemming Design Using FSA

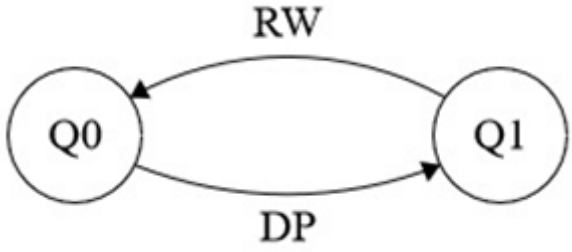

Figure 3. NFA Automata Machine

From the Figure 3, it can be explained that the stemming process consists of state Q0 is RW (root wood) or basic word while Q1 is DP (derivational prefix) or prefix. For example, the prefix uses ber + the base word (DP) using the word love so that it becomes the word love.

The process for removing affixes as in the picture starts from state Q0 to state Q1 then returns from state Q1 to Q0 which means that the affix has been successfully removed and has returned to the base word.

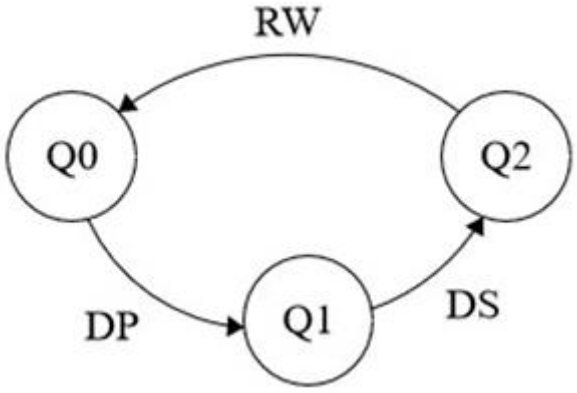

Figure 4. NFA Automata Machine

From the Figure 4 can be explained that the stemming process consists of state Q0 as RW (root wood) or base word, state Q1 as DP (derivational prefix) or prefix, and state Q2 as DS (derivational suffix) or suffix. For example, the prefix ter-(DP) + root word (RW) uses the word love + suffix an (DS) to become the word beloved.

 ISSN:2302-8149
e-ISSN:2540-9719

The process for removing affixes as in the picture starts from state Q0 to state Q1, namely DP then from state Q1 to state Q2, namely DS and from state Q2 back again to Q1 so that it returns to the base word (RW). In other words, the stemming process starts from removing the initial affix DP then the final affix (DS) so that it leaves the base word (RW).

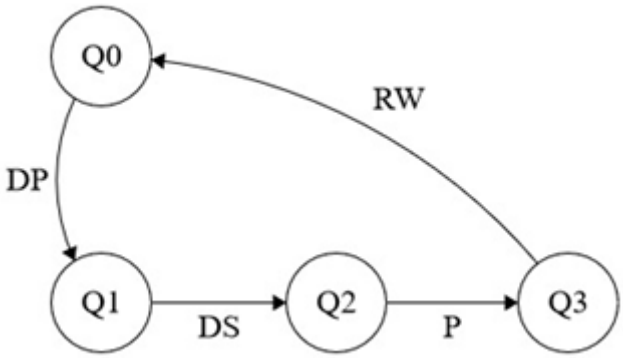

Figure 5. NFA Automata Machine

From the Figure 5, it can be explained that the mapping process consists of state Q0 as RW (root wood) or base word, state Q1 as DP (derivational prefix) or prefix, state Q2 as DS (derivational suffix) or suffix, state Q3 is P in the form of suffix and state Q4 as PP (prossesive pronoun). For example, the prefix ber (DP) + the root word (RW) cinta + lah (P) so that it becomes the word bercintalah.

The process for removing affixes as in the picture starts from state Q0 to state Q1, namely DP then from state Q1 to state Q2, namely DS and from state Q2 to state Q3, namely P and back again from state Q3 to state Q1 so as to re-form the base word (RW). In other words, the stemming process starts from removing the initial affix DP then the final affix (DS) and the final affix P so that it leaves the base word (RW).

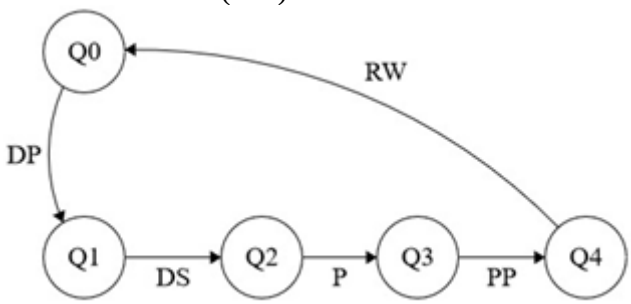

Figure 6. NFA Automata Machine

From the Figure 6, it can be explained that the mapping process consists of state Q0 as RW (root wood) or base word, state Q1 as DP (derivational prefix) or prefix, state Q2 as DS (derivational suffix) or suffix and state Q3 which is P in the form of suffix and state Q4 as PP. For example, the prefix ter (DP) + the root word (RW) cinta + lah so that it becomes the word tercintakanlah.

The process for removing affixes as in the picture starts from state Q0 to state Q1, namely DP, then from state Q1 to state Q2, namely DS and from state Q2 to state Q3, namely P, then from state Q3 to state Q4 and back again to state Q1 so as to re-form the base word (RW). In other words, the stemming process starts from removing the initial affix DP then the final affix (DS), the final affix P, the final affix PP so that it leaves the base word (RW).

 ISSN:2302-8149
e-ISSN:2540-9719

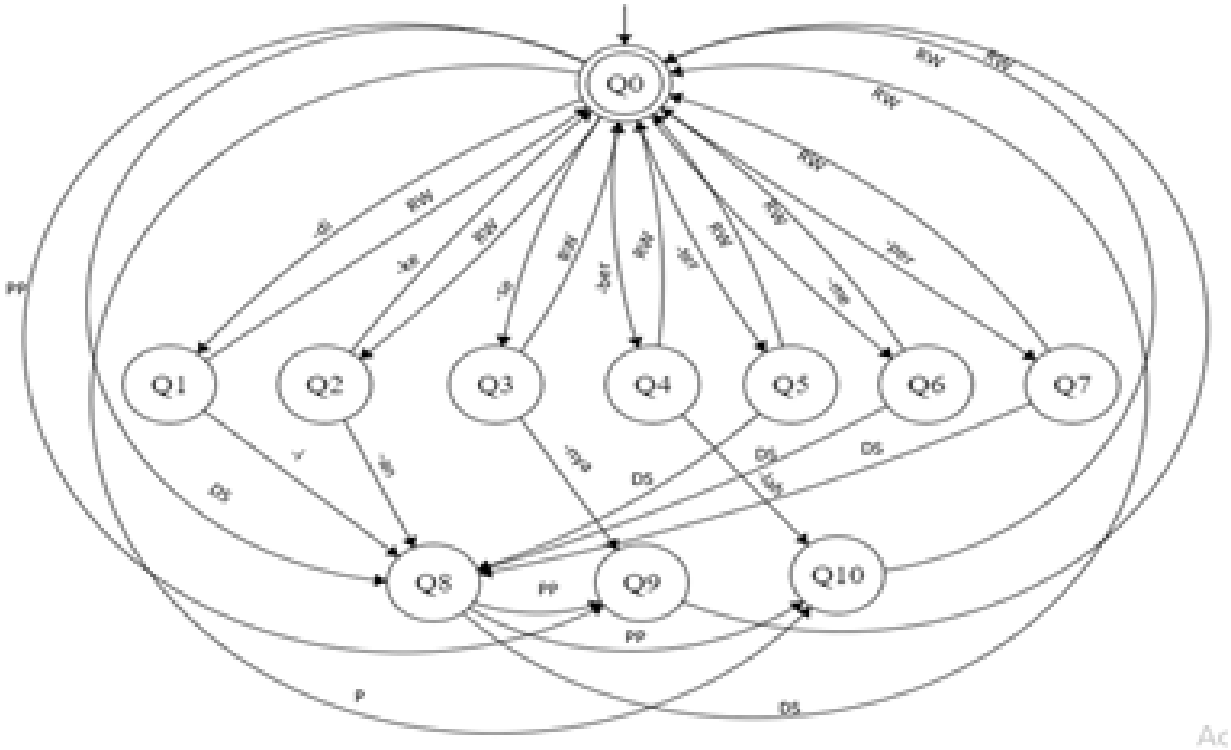

Figure 7. NFA Automata Machine

From the Figure 7, it can be explained that the mapping process consists of state:
Q0 = RW (Root Wood)
Q1 = DP (Di) Q2 = DP (Ke)
Q3 = DP (Se)
Q4 = DP (Ber)
Q5 = DP (Ter)
Q6 = DP (Me)
Q7 = DP (Pe)
Q8 = DS (i, kan, an)
Q9 = PP (ku, mu, nya)
Q10 = P (kah, lah)

For Example:
Cinta (RW) Q0
Dicinta (Di + RW) Q1-Q0
Dicintai (Di + RW + DS) Q1-Q8-Q0
Mencintai (Me + RW + DS) Q6-Q8-Q0
Bercinta (Ber + RW) Q4-Q0
Cintailah (RW + DS + P) Q8-Q10-
Mencintainya (Me + RW + DS + PP) Q6-Q8-Q9-Q0
Mencinta (Me + RW) Q6-Q0
Bercintalah (Ber + RW + P) Q4-Q10-Q0
Kecintaannya (Ke + RW + DS + PP) Q2-Q8-Q9-Q0
Percintaan (Per + RW + DS) Q7-Q8-Q0
Percintaannya (Per + RW + DS + PP) Q7-Q8-Q9-Q0
Cintainya (RW + DS + PP) Q8-Q9-Q0
Cintaku (RW + PP) Q9-Q0
Cintai (RW + DS) Q8-Q0

 ISSN:2302-8149
e-ISSN:2540-9719

Cintakah (RW + P) Q10-Q0
Tercinta (Ter + RW) Q5-Q0
Kecintaan (Ke + RW + DS) Q2-Q8-Q0
Kecintaannyalah (Ke + RW + DS + PP + P) Q2-Q8-Q9-Q10-Q0
Dicintainya (Di + RW + DS + PP) Q1-Q8-Q9-Q0
Secintanya (Se + RW + PP) Q3-Q9-Q0

In the Figure 7, the mapping process starts from each word that has been given affixes will be removed by removing the initial affix until the final affix until it returns to the base word.

C. Pseudecode Development

Program Stemming

```
Sub Modul Cek_Kata
    length($kata) > 3
    cek $kata
End

Sub Modul dissallowed_combination
    be + $kata + i
    se + $kata + (i|kan)
    di + $kata + an
    me + $kata + an
    ke + $kata + (i|kan)
End

Sub Modul hapus_derifation_suffixes1
    $kata = ganti ("i "," ") di $kata
    $kata = ganti ("kan "," ") di $kata
    $kata = ganti ("an "," ") di $kata
End

Sub Modul Hapus_infleksional_sufiks
    $kata= ganti("lah "," ") pada $kata;
    $kata= ganti("kah "," ") pada $kata;
End

Sub Modul Hapus_sufiks
    $kata= ganti("nya "," ") pada $kata;
    $kata= ganti("ku "," ") pada $kata;
    $kata= ganti("mu "," " ) pada $kata;
End

Sub Modul hapus_prefiks
    $kata= ganti(" ber”," ") pada $kata;
    $kata= ganti(" belajar”," ajar") pada $kata;
    $kata= ganti(" be"," ") pada $kata;
    $kata= ganti(" ter"," ") pada $kata;
    $kata= ganti(" me{l|r|w|y}"," ") pada $kata;
    $kata= ganti(" mem{b|f|v}"," ") pada $kata;
    $kata= ganti(" mempe{r|l}"," ") pada $kata;
    $kata= ganti(" mem{r|V}"," {m|p}") pada $kata;
    $kata= ganti(" men{c|d|j|z}"," " ) pada $kata;
    $kata= ganti(" menV"," {nV|tV}") pada $kata;
```

 ISSN:2302-8149
e-ISSN:2540-9719

```
    $kata= ganti(" meng{g|h|q}"," ") pada $kata;
    $kata= ganti(" mengV"," ") pada $kata;
    $kata= ganti(" menyV"," ") pada $kata;
    $kata= ganti(" mempV"," ") pada $kata;
    $kata= ganti(" pe{w|y}"," " ) pada $kata;
    $kata= ganti(" perV"," { |r}" ) pada $kata;
    $kata= ganti(" perC"," " ) pada $kata;
    $kata= ganti(" pem{b|f|v}"," ") pada $kata;
    $kata= ganti(" pen{c|d|j|z}"," ") pada $kata;
    $kata= ganti(" penV"," ") pada $kata;
    $kata= ganti(" peng"," {g|h|q}") pada $kata;
    $kata= ganti(" pengV"," { |k}") pada $kata;
    $kata= ganti(" penyV"," ") pada $kata;
    $kata= ganti(" pelV"," " ) pada $kata;
    $kata= ganti(" peC"," er") pada $kata;
End

Main Algorithm
    Cek_Kata
    dissallowed_combination(Jika diperlukan)
    hapus_derifation_suffixes1(Jika diperlukan)
    Cek_Kata(Jika diperlukan)
    Hapus_infleksional_sufiks(Jika diperlukan)
    Cek_Kata(Jika diperlukan)
    Hapus_sufiks(Jika diperlukan)
    Cek_Kata(Jika diperlukan)
    hapus_prefiks(Jika diperlukan)
    Cek_Kata(Jika diperlukan)
End
```

D. Testing

The following are the results of testing basic word searches using the Nazief & Adriani Algorithm that have been carried out with web applications.

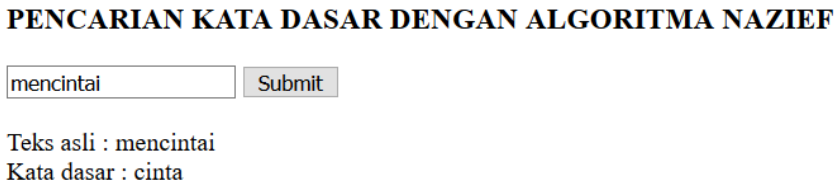

Figure 8. Stemming Result DP+RW+DS

Figure 8 is the result of stemming the word "mencintai", the program executes or removes the Derivation Prefix (men-) and removes the Derivation Sufix (-i) resulting in the Root Word (cinta).

 ISSN:2302-8149
e-ISSN:2540-9719

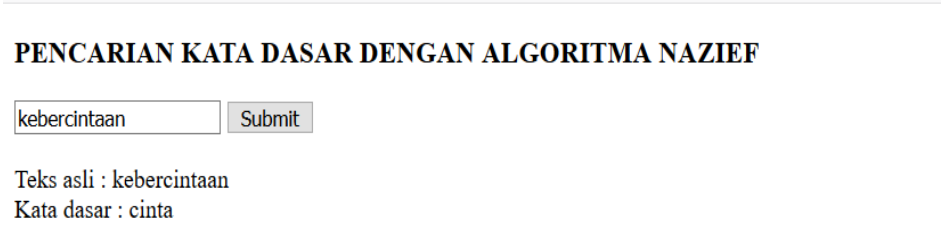

Figure 9. Stemming Result DP+DP+RW+DS

Based on Figure 9, Stemming results in the word 'love' by removing Derivation Prefix (ke - and ber -) removing Derivation Suffix (-an) so that the program is able to produce Root Word (love).

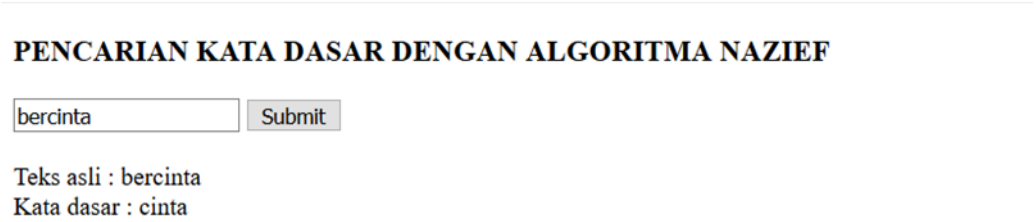

Figure 10. Stemming Result DP+RW

The results of trials conducted on the web application, in accordance with Figure 10, with the word 'love' the program is run by removing the Derivation Prefix (ber -) first so that it can produce the Root Word (love).

Meanwhile, in the stemming result of the word 'love', this program will remove the Derivation Prefix (ber -) and Derivation Sufix (-lah) so that it can produce the base word (love) with the stemming result DP+RW+DS. However, a similar experiment is also applied to the word “berilmu”, and the stemming result produces the base word “beril”, as shown in Figure 11.

Figure 11. Stemming Result “Berilmu”

The stemming result with the word "berilmu" becomes "beril". In the Nazief Adriani algorithm, the process of removing suffixes first can cause understemming (stemming too early) from the proper base word "science".

There is a weakness of the Nazief & Adriani algorithm because it performs stemming from the Particle (P) suffix so that words containing suffixes such as 'mu' with the word 'knowledge' will become 'beril'. The solution to this shortcoming is to do stemming in the Prefix position.

5 Conclusion

Based on the results of research that has been carried out in several stages in the previous discussion, the implementation of a web-based system can produce accurate word solving using 20 affix

 ISSN:2302-8149
e-ISSN:2540-9719

words with the base word "love". As for the concept of the application used in the application of this word stemming, the reading of affixes into basic words begins with the order of removal of affixes from suffixes (suffixes), prefixes (prefixes), then into basic words (root words). So it can be concluded that the application of the nazief & adriani stemming algorithm with the finite state automata model can be implemented with a web-based system that can produce a model for solving affix words into Indonesian basic words according to the concept of the nazief & adriani stemming algorithm appropriately.There are some weaknesses in stemming from suffixes and the solution is to test stemming from prefixes. - This program still needs to be developed by testing more words to determine the accuracy of the stemming process.

Reference

[1] O. Mailani, I. Nuraeni, S. A. Syakila, dan J. Lazuardi, "Bahasa sebagai Alat Komunikasi dalam Kehidupan Manusia," *Kampret J.*, vol. 1, no. 1, hal. 1–10, 2022, doi: 10.35335/v1i1.8.

[2] R. W. Eriyanti, K. T. Syarifuddin, K. Datoh, dan E. Yuliana, *Linguistik Umum*. Uwais Inspirasi Indonesia, 2020.

[3] J. Jumadi, D. S. Maylawati, L. D. Pratiwi, dan M. A. Ramdhani, "Comparison of Nazief-Adriani and Paice-Husk algorithm for Indonesian text stemming process," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1098, no. 3, hal. 032044, 2021, doi: 10.1088/1757-899x/1098/3/032044.

[4] S. Huber, H. Wiemer, D. Schneider, dan S. Ihlenfeldt, "DMME: Data mining methodology for engineering applications - A holistic extension to the CRISP-DM model," *Procedia CIRP*, vol. 79, hal. 403–408, 2019, doi: 10.1016/j.procir.2019.02.106.

[5] J. Asian, H. E. Williams, dan S. M. M. Tahaghoghi, "Stemming Indonesian," *Conf. Res. Pract. Inf. Technol. Ser.*, vol. 38, hal. 307–314, 2019, doi: 10.1145/1316457.1316459.

[6] W. Hidayat, E. Utami, dan A. D. Hartanto, "Effect of Stemming Nazief Adriani on the Ratcliff/Obershelp algorithm in identifying level of similarity between slang and formal words," *2020 3rd Int. Conf. Inf. Commun. Technol. ICOIACT 2020*, hal. 22–27, 2020, doi: 10.1109/ICOIACT50329.2020.9331973.

[7] A. Yudhana, A. Fadlil, dan M. Rosidin, "Indonesian words error detection system using nazief adriani stemmer algorithm," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 12, hal. 219–225, 2019, doi: 10.14569/ijacsa.2019.0101231.

[8] A. Sinaga dan S. P. Nainggolan, "Analisis Perbandingan Akurasi dan Waktu Proses Algoritma Stemming Arifin-Setiono dan Nazief-Adriani pada Dokumen Teks Bahasa Indonesia," *Sebatik*, vol. 27, no. 1, hal. 63–69, 2023, doi: 10.46984/sebatik.v27i1.2072.

[9] T. Hari Wicaksono, F. Dwiki Amrizal, H. Atun Mumtahana, dan J. Setia Budi No, "Pemodelan Vending Machine dengan Metode FSA (Finite State Automata)," *J. Comput. Inf. Technol.*, vol. 2, no. 2, hal. 66–69, 2019, [Daring]. Tersedia pada: http://e-journal.unipma.ac.id/index.php/doubleclick/article/view/3901.

[10] A. T. Ni'mah, D. A. Suryaningrum, dan A. Z. Arifin, "Autonomy Stemmer Algorithm for Legal and Illegal Affix Detection use Finite-State Automata Method," *EPI Int. J. Eng.*, vol. 2, no. 1, hal. 46–55, 2019, doi: 10.25042/epi-ije.022019.09.

[11] I. Mulyana, A. Suhendra, Ernastuti, dan W. Bheta Agus, "Development of indonesian stemming algorithms through modification of grouping, sequencing and removing of affixes based on morphophonemic," *Int. J. Recent Technol. Eng.*, vol. 8, no. 2 Special Issue 7, hal. 179–184, 2019, doi: 10.35940/ijrte.B1044.0782S719.

[12] Rianto, A. B. Mutiara, E. P. Wibowo, dan P. I. Santosa, "Improving the accuracy of text classification using stemming method, a case of non-formal Indonesian conversation," *J. Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-021-00413-1.

[13] D. Mustikasari, I. Widaningrum, R. Arifin, dan W. H. E. Putri, "Comparison of Effectiveness of Stemming Algorithms in Indonesian Documents," *Proc. 2nd Borobudur Int. Symp. Sci. Technol. (BIS-STE 2020)*, vol. 203, hal. 154–158, 2021, doi: 10.2991/aer.k.210810.025.

[14] R. T. Rizki, Afian Syafaadi, Aris Tjahyanto, "Comparison of stemming algorithms on Indonesian text processing," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol.

 ISSN:2302-8149
e-ISSN:2540-9719

17, no. 1, hal. 95–102, 2019.

[15] I. P. M. Wirayasa, I. M. A. Wirawan, dan I. M. A. Pradnyana, "Algoritma Bastal: Adaptasi Algoritma Nazief & Adriani untuk Stemming Teks Bahasa Bali," *J. Nas. Pendidik. Tek. Inform.*, vol. 8, no. 1, hal. 60, 2019, doi: 10.23887/janapati.v8i1.13500.

[16] A. C. Herlingga, I. P. E. Prismana, D. R. Prehanto, dan D. A. Dermawan, "Algoritma Stemming Nazief & Adriani dengan Metode Cosine Similarity untuk Chatbot Telegram Terintegrasi dengan E-layanan," *J. Informatics Comput. Sci.*, vol. 2, no. 01, hal. 19–26, 2020, doi: 10.26740/jinacs.v2n01.p19-26.

[17] L. A. Fitriana, A. Mustopa, M. R. Firdaus, dan R. Dahlia, "Application of the Finite State Automata (FSA) Method in Indonesian Stemming using the Nazief & Adriani Algorithm," 2024.