Penerapan Algoritma *String Matching* dan *Levensthein Distance* Untuk Identifikasi Kemiripan Judul Makalah dengan Data Makalah Yang Telah Lalu

Atabik Muhammad Azfa Shofi / 13520159[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13520159@std.stei.itb.ac.id

***Abstraksi*—Dalam penulisan sebuah karya tulis, penentuan judul merupakan hal yang paling dasar dan penting. Penulisan judul tersebut tidak terlepas dari pentingnya mengidentifikasi kemiripan dengan judul-judul yang telah ada sebelumnya. Algoritma string matching dapat diterapkan pada kasus ini untuk menemukan kecocokan atau menentukan kemiripan antara judul-judul. Lebih jauh lagi, konsep levensthein distance dapat digunakan untuk membantu menentukan kemiripan tersebut.**

***Kata Kunci*—Judul, string matching, levensthein distance**

I. PENDAHULUAN

Dalam dunia akademis, menulis sebuah karya tulis baik berupa makalah, *paper*, ataupun jenis lainnya, adalah hal yang biasa dilakukan. Penulisan karya tulis tentunya berhubungan dengan penentuan topik dan judul karya. Judul yang dianggap baik tentunya yang masih relevan dan baru dalam bidangnya, tidak memiliki struktur kalimat dan kata-kata yang sangat mirip dengan judul-judul karya yang telah dibuat sebelumnya.

Algoritma *string matching* adalah salah satu algoritma untuk melakukan pencocokan atau pencarian sebuah pattern string terhadap suatu teks tertentu. Algoritma ini dapat menjadi salah satu alternatif untuk melakukan perbandingan judul makalah yang baru dengan judul-judul yang telah ada sebelumnya. Sehingga, judul yang baru dapat terlebih dahulu diidentifikasi kemiripannya dengan judul-judul yang sudah ada agar tidak membahas hal yang telah ada maupun teridentifikasi plagiarisme. Ada banyak jenis algoritma *string matching* yang dapat digunakan, beberapa di antaranya yang akan sering dibahas dan dipakai pada makalah ini adalah algoritma *Knuth-Morris Pratt* (KMP) dan algoritma *Boyer-Moore* (BM).

Levensthein distance adalah konsep yang digagas oleh Vladimir Levensthein yang merupakan konsep perbandingan dua sekuens string untuk menentukan nilai dari minimum langkah yang perlu dilakukan untuk merubah string pertama menjadi string kedua. Konsep ini dapat digunakan untuk membantu dalam algoritma *string matching* yang akan diterapkan dan mempermudah perhitungan kemiripan judul makalah yang akan dibandingkan.

II. LANDASAN TEORI

1. String Matching

Misalnya didefinisikan sebuah pattern adalah suatu baris yang berisi karakter sepanjang m dan sebuah teks yang merupukan kumpulan karakter sepanjang n dengan n >> m (Panjang n lebih besar dari m). *String matching* atau pencocokan string, adalah algoritma untuk mencari kecocokan sebuah string pattern di dalam sebuah teks dalam kemunculan pertama atau lebih jauh lagi adalah mecocokan string satu dengan string lainnya dan membandingkan tingkat kemiripannya, seperti yang akan dilakukan pada pembahasan makalah ini.

Algoritma string matching ini dapat digunakan pada berbagai macam bidang, misalnya pencarian kata pada suatu teks editor, penelusuran pencarian pada suatu search engine, pencocokan gambar pada analisis citra khususnya pencocokan sidik jari, pencocokan sekuens DNA pada bidang bioinformatics, dan masih banyak yang lainnya.

Dalam konsep string itu sendiri, dikenal konsep prefix dan suffix yang menggambarkan substring awal dan substring akhir dari sebuah string. Misalnya, sebuah string S panjangnya m, maka prefixnya adalah S[0…k] dengan suffixnya adalah S[k…m-1] dengan k adalah index berapapun antara 0 sampai m. Sebagai contoh, sebuah string “Contoh”, maka kumpulan prefix dan suffixnya adalah.

Prefix = [“C”, “Co”, “Con”, “Cont”, “Conto”, “Contoh”]

Suffix = [“Contoh”, “ontoh”, “ntoh”, “toh”, “oh”, “h”]

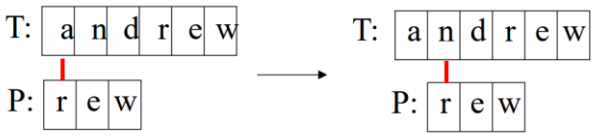
Banyak cara yang dapat dijadikan pendekatan untuk melakukan string matching ini. Di antaranya adalah :

- Algoritma Brute-Force
- Algoritma *Knuth-Morris Pratt* (KMP)
- Algoritma *Boyer-Moore* (BM)
- Algoritma Aho-Corasick
- Algoritma Rabin Karp

Dan beberapa algoritma lainnya yang intinya kurang lebih sama, yaitu mencari kecocokan. Namun, biasanya perbedaannya terdapat pada algoritma pergeseran apabila

tidak ditemukan kecocokan. Pada pembahasan kali ini, yang akan dijelaskan lebih lanjut adalah algoritma Brrute-Force, KMP, dan BM.

- Algoritma Brute-Force
  Algoritma Brute-Force adalah algoritma untuk menyelesaikan persoalan secara *straight-forward* atau menyelesaikan persoalan secara langsung. Pada masalah string matching ini, algoritma brute-force digunakan dengan cara mencocokkan seluruh bagian pattern terhadap teks dari kiri ke kanan dimulai dari indeks pertama hingga terakhir. Apabila ditemukan ketidakcocokan pada indeks tertentu dari pattern terhadap indeks tertentu dari teks, maka pencocokan dilakukan dari ulang dengan bagian teks digeser sebanyak 1 buah. Ilustrasinya dapat dilihat pada gambar berikut :



P moves 1 char at a time through T
Gambar 2.1 string matching menggunakan brute-force
sumber
(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Worst Case (kasus terburuk) :
Pada algoritma ini, kasus terburuknya adalah apabila seluruh karakter pada pattern selalu benar saat dilakukan pengujian namun karakter terakhir pattern hanya ada pada karakter terakhir teks. Sehingga jumlah pencarian yang dilakukan adalah sebanyak n-m+1 kali pada teks, dan perbandingan yang dilakukan tiap pencarian adalah m kali (sebanyak karakter pada pattern).
Jumlah perbandingan : m(n-m+1) : O(mn)
Contoh :
Teks = “hhhhhhhhhhhhhhhe”
Pattern = “hhe”

Best Case (kasus terbaik) :
Kasus terbaiknya adalah apabila karakter pattern pertama tidak pernah ditemukan pada teks. Maka jumlah perbandingan yang dilakukan maksimal hanya n kali.
Jumlah perbandingan : n : O(n)
Contoh :
Teks = “asdskadksaffaksbvjhhh”
Pattern = “hhh”

Average Case (kasus normal) :
Rata-rata kasus normal (kasus selain best dan worst) adalah O(m+n).

Biasanya, algoritma brute-force lebih baik apabila kemungkinan karakter yang dicocokkan lebih besar, misalnya alphabet (26 kemungkinan), dan kurang baik apabila kemungkinan kecil, misalnya biner (2 kemungkinan).

- Algoritma KMP
  Algoritma Knuth-Morris-Pratt (KMP) dikenalkan oleh Donald Ervin Knuth, adalah algoritma string matching yang mencari pola pada text dengan urutan kiri ke kanan seperti algoritma Brute-Force tetapi bergerak sedikit lebih cerdas apabila tidak ditemukan kecocokan pada teks, dibandingkan dengan algoritma Brute-Force. Apabila terdapat sebuah ketidaksamaan pada text dengan pattern pada indeks j, maka pergeseran akan dilakukan berdasarkan prefix (P[0..j-1]) terkecil dari pattern yang juga merupakan suffix (P[1..j]) dari pattern tersebut.

  Untuk melakukan hal tersebut, maka akan dibuat border function dari pattern tersebut. Border function b(k) didefinisikan sebagai ukuran terbesar prefix dari P[0..k-1] yang juga merupakan suffix dari P[1..k]. Secara detailnya, algoritma KMP memodifikasi algoritma brute force dengan algoritma berikut:

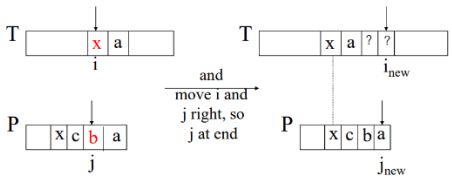
  Apabila terjadi ketidaksamaan elemen pada P[j] (P[j] != T[i]), maka:
  k = j-1;
  j = b(k)

  Kompleksitas waktu dari algoritma KMP adalah O(m+n), yang merupakan dari hasil pertambahan algoritma perhitungan fungsi pinggiran (O(m)) dan algoritma pencarian string (O(n)). Dari kompleksitas waktunya, algoritma KMP merupakan algoritma yang lebih efisien dibandingkan dengan brute force dalam string matching.

- Algoritma BM
  Algoritma Boyer-Moore (BM) merupakan algoritma string matching yang mencari pola pada text berdasarkan dua teknik yaitu looking-glass technique atau mencari dengan urutan mundur (dari kanan ke kiri) dan character-jump technique yaitu melakukan lompatan yang tepat jika pattern ditemukan tidak sesuai dengan bagian text yang sedang dicocokkan.

  Idenya adalah dengan menyimpan data last occurence dari tiap karakter yang terdapat pada pattern. Ada 3 kemungkinan ketidakcocokan yang dapat ditemukan pada algoritma ini.

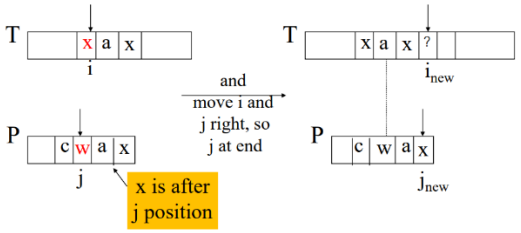
  Jika ketidakcocokan terjadi pada karakter X(misal) pada text, lalu pada pattern terdapat X dan mungkin dilakukan pergeseran, maka pencocokan diulang pada posisi X pattern sejajar dengan X text.



Gambar 2.2 string matching menggunakan BM-1

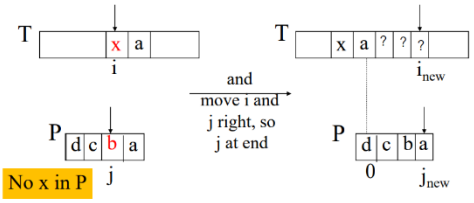
sumber
(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Jika pada pattern terdapat X namun posisi X tidak memungkinkan melakukan pergeseran, maka pencocokan diulang dengan pattern digeser satu karakter ke kanan dari text.



Gambar 2.3 string matching menggunakan BM-2
sumber
(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Jika pada pattern tidak terdapat X, maka pencocokan diulang dengan posisi pattern[0] (karakter pertama pattern) berada pada posisi X + 1.



Gambar 2.4 string matching menggunakan BM-3
sumber
(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Worst case dari algoritma BM ini berjalan pada kompleksitas O(mn + A). Sama seperti algoritma Brute Force, algoritma BM akan baik berjalan apabila kemungkinan karakter banyak seperti alfabet dan kurang baik pada biner.

2. Levensthein Distance
Levensthein distance adalah konsep yang diperkenalkan oleh Vladimir Levensthein, untuk perbandingan dua sekuens dan menentukan banyaknya minimum langkah yang dibutuhkan untuk merubah sekuens sat uke sekuens lainnya.

Secara matematis, levensthein distance dapat dituliskan sebagai berikut :

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

Gambar 2.5 rumus formal levensthein distance
sumber (https://www.cuelogic.com/blog/the-levenshtein-algorithm)

Contohnya, sebuah teks “Wagon” dan “Wenger”, maka levensthein distance antara keduanya adalah 4. Penjelasannya sebagai berikut :
W A G O N → W E G O N (substitusi +1)
W E G O N → W E N G O N (insersi +1)
W E N G O N → W E N G E N (substitusi +1)
W E N G E N → W E N G E R (substitusi +1)
total 4 kali perubahan.

Contoh lainnya untuk lebih jelas, misalnya teks “Lapar” dan “Lalat”. maka levensthein distance antara keduanya adalah 2. Penjelasannya sebagai berikut :
L A P A R → L A L A R (substitusi +1)
L A L A R → L A L A T (substitusi +1)
total ada 2 kali perubahan.

Levensthein distance dapat digunakan untuk membantu dalam menentukan kemiripan atau kecocokan sebuah string pattern terhadap suatu teks tertentu.

III. PEMBAHASAN

Untuk melakukan perbandingan antara judul makalah yang baru dengan judul yang lama, akan dilakukan dua jenis perbandingan. Yaitu penerapan string matching pada pattern berupa masing-masing kata pada judul baru, dan teks berupa judul lama yang utuh, dan penerapan penggunaan levensthein distance untuk masing-masing kata pada judul baru terhadap seluruh kata pada judul lama. Pengujian akan dilakukan secara berulang sehingga seluruh judul pada data judul lama telah selesai dibandingkan dan dapat diambil kesimpulan.

1. String Matching

Akan dilakukan pencocokan string dengan pattern berupa kata-kata yang terdapat pada judul baru dan teks berupa judul lama. Misalnya, judul baru yang digunakan adalah “Cara Efektif Mencari Masalah Pada Penulisan Makalah” dan data judul lama yang akan dibandingkan adalah {“Analisis Keefektifan Penggunaan Lilin Terhadap Fotosintesis Kacang Hijau”, “Cara Menemukan Judul Yang Tepat Pada Penulisan Makalah”, “Perbandingan Nilai Mata Uang Rupiah Dengan Euro Tahun 2022”}.

Pertama, perlu dilakukan dekomposisi pada judul yang baru menjadi sebuah array yang mengandung string, dekomposisi dilakukan dengan memisahkan judul pada tiap spasi (“ “). Hasil array dari judul baru yang telah ditentukan adalah sebagai berikut :
Array = {“Cara”, “Efektif”, “Mencari”, “Masalah”, “Pada”, “Penulisan”, “Makalah”}

Selanjutnya, untuk masing-masing elemen pada array tersebut akan dilakukan string matching terhadap judul lama dengan menggunakan algoritma apapun, baik brute-force, KMP, BM, maupun algoritma string matching yang lainnya.

Dari hasil string matching yang dilakukan terhadap judul pertama “Analisis Keefektifan Penggunaan Lilin Terhadap Fotosintesis Kacang Hijau”, didapatkan hasil seperti pada tabel berikut:

Elemen	Hasil
“Cara”	Tidak ditemukan
“Efektif”	Ditemukan pada kata “keefektifan”

“Mencari”	Tidak ditemukan
“Masalah”	Tidak ditemukan
“Pada”	Tidak ditemukan
“Penulisan”	Tidak ditemukan
“Makalah”	Tidak ditemukan
Total ditemukan	1

Dapat dilihat, dari hasil string matching yang dilakukan, kata yang ditemukan pada judul lama tersebut hanya terdapat total 1 dari 7 kata. Artinya, dapat dinilai tingkat kemiripan judul baru terhadap judul lama adalah 1 / 7 atau sekitar 14.3%.

Sedangkan, untuk judul kedua “Cara Menemukan Judul Yang Tepat Pada Penulisan Makalah” didapatkan tabel sebagai berikut:

Elemen	Hasil
“Cara”	Ditemukan pada kata pertama “Cara”
“Efektif”	Tidak ditemukan
“Mencari”	Tidak ditemukan
“Masalah”	Tidak ditemukan
“Pada”	Ditemukan pada kata keenam “Pada”
“Penulisan”	Ditemukan pada kata ketujuh “Penulisan”
“Makalah”	Ditemukan pada kata kedelapan “Makalah”
Total ditemukan	4

Dapat dilihat, dari hasil string matching yang dilakukan, kata yang ditemukan ada 4 dari 7 buah. Artinya, dapat dinilai tingkat kemiripan judul baru terhadap judul lama adalah 4 / 7 atau sekitar 57.1%.

Sedangkan, untuk judul terakhir “Perbandingan Nilai Mata Uang Rupiah Dengan Euro Tahun 2022” didapatkan tabel sebagai berikut:

Elemen	Hasil
“Cara”	Tidak ditemukan
“Efektif”	Tidak ditemukan
“Mencari”	Tidak ditemukan
“Masalah”	Tidak ditemukan
“Pada”	Tidak ditemukan
“Penulisan”	Tidak ditemukan
“Makalah”	Tidak ditemukan
Total ditemukan	0

Dapat dilihat, dari hasil string matching yang dilakukan, kata yang ditemukan ada 0 dari 7 buah. Artinya, dapat dinilai tingkat kemiripan judul baru terhadap judul lama adalah 0 / 7 atau 0%.

Dari ketiga judul yang terdapat pada data, dapat dilihat bahwa seluruh pengujian menggunakan string matching menunjukkan kemiripan di bawah 80-90%, sehingga judul yang baru masih dapat dianggap aman untuk digunakan. Meskipun demikian, bukan berarti pengujian menggunakan string matching sudah pasti akurat terhadap kemungkinan plagiarisme karena mungkin banyaknya parafrasa yang digunakan dan lain sebagainya. Oleh karena itu, perlu dilakukan pengujian lanjutan menggunakan levensthein distance untuk tiap kata pada judul baru terhadap seluruh kata pada judul lama.

2. Levensthein distance

Akan dilakukan pengujian kemiripan judul baru dengan data judul lama dengan judul-judul yang sama pada bagian sebelumnya, dengan menggunakan levensthein distance.

Pertama, lakukan hal yang sama seperti bagian sebelumnya yaitu dekomposisi judul menjadi array. Hasilnya sebagai berikut:
Array = {“Cara”, “Efektif”, “Mencari”, “Masalah”, “Pada”, “Penulisan”, “Makalah”}

Selanjutnya, dekomposisi juga perlu dilakukan pada masing-masing judul lama, dan dilakukan pengujian kecocokan. Untuk judul pertama, hasil dekomposisinya sebagai berikut:
Array = {“Analisis”, “Keefektifan”, “Penggunaan”, “Lilin”, “Terhadap”, “Fotosintesis”, “Kacang”, “Hijau”}

Pengujian akan dilakukan dengan mencari nilai terendah dari LD/Maksimum_LD dari tiap kata pada judul baru. Lalu, hasil nilai terendah tersebut akan dirata-ratakan. Apabila nilainya di bawah 20-30% mungkin judul dapat dikatakn mirip. Karena, jumah rata-rata yang dibutuhkan tiap kata untuk dirubah menjadi judul yang dibandingkan sangat rendah.

Hasil Levensthein distance dari kedua judul di atas dapat dilihat pada tabel berikut:

Kata-1	Kata-2	LD/Max	Persentase(%)
“Cara”	“Analisis”	8/8	100
	“Keefektifan”	10/11	90.9
	“Penggunaan”	10/10	100
	“Lilin”	5/5	100
	“Terhadap”	6/8	75
	“Fotosintesis”	12/12	100
	“Kacang”	4/6	66.7
	“Hijau”	4/5	80
	Min	66.7%	
“Efektif”	“Analisis”	7/8	87.5
	“Keefektifan”	4/11	36.4
	“Penggunaan”	9/10	90
	“Lilin”	6/7	85.7
	“Terhadap”	7/8	87.5
	“Fotosintesis”	11/12	91.7
	“Kacang”	7/7	100
	“Hijau”	7/7	100
	Min	36.4%	
“Mencari”	“Analisis”	7/8	87.5
	“Keefektifan”	9/11	81.8
	“Penggunaan”	8/10	80
	“Lilin”	6/7	85.7
	“Terhadap”	6/8	75
	“Fotosintesis”	11/12	91.7

	“Kacang”	5/7	71.4
	“Hijau”	6/7	85.7
	Min	71.4%	
“Masalah”	“Analisis”	6/8	87.5
	“Keefektifan”	11/11	100
	“Penggunaan”	10/10	100
	“Lilin”	6/7	85.7
	“Terhadap”	6/8	75
	“Fotosintesis”	11/12	91.7
	“Kacang”	5/7	71.4
	“Hijau”	6/7	85.7
	Min	71.4%	
“Pada”	“Analisis”	7/8	87.5
	“Keefektifan”	10/11	90.9
	“Penggunaan”	8/10	80
	“Lilin”	5/5	100
	“Terhadap”	5/8	62.5
	“Fotosintesis”	12/12	100
	“Kacang”	4/6	66.7
	“Hijau”	4/5	80
	Min	62.5	
“Penulisan”	“Analisis”	5/9	55.5
	“Keefektifan”	7/11	63.6
	“Penggunaan”	5/10	50
	“Lilin”	6/9	66.6
	“Terhadap”	7/9	77.8
	“Fotosintesis”	11/12	91.7
	“Kacang”	8/9	88.9
	“Hijau”	8/9	88.9
	Min	50	
“Makalah”	“Analisis”	6/8	75
	“Keefektifan”	10/11	90.9
	“Penggunaan”	9/10	90
	“Lilin”	6/7	85.7
	“Terhadap”	6/8	75
	“Fotosintesis”	12/12	100
	“Kacang”	5/7	71.4
	“Hijau”	6/7	85.7
	Min	71.4	
Avg	61.4%		

Dari hasil tabel, dapat dilihat bahwa nilai rata-rata perubahan minimal yang dilakukan adalah 61.4% yaitu berada sangat jauh di atas 20-30%, sehingga dapat dikatakan judul baru tidak mirip dengan judul yang lama sehingga judul aman untuk digunakan. Selanjutnya, perlu dilakukan perhitungan yang sama untuk setiap judul yang terdapat pada data daftar judul, sehingga keputusan akhir untuk menggunakan judul dapat diambil.

IV. KESIMPULAN DAN SARAN

Penentuan judul suatu karya tulis adalah bagian yang penting dalam proses penuisan karya tulis, sehingga perlu dilakukan identifikasi kemiripan judul dengan judul-judul yang telah ada. Algoritma string matching dapat diterapkan untuk menyelesaikan masalah ini, dengan melakukan perbandingan jumlah kata yang mirip antara kedua judul, maupun perbandingan levensthein distance tiap kata pada judul awal terhadap seluruh kata pada judul yang akan dibandingkan.

Sebagai saran, mungkin kedepannya perhitungan dapat dibuat secara automatis menggunakan komputer dengan membuat program dan source-code sendiri, sehingga perhitungan untuk seluruh data judul dapat dilakukan dengan mudah dan kesimpulan lebih mudah cepta dan tepat untuk ditarik.

V. UCAPAN TERIMA KASIH

Alhamdulillah, segala puji serta syukur saya panjatkan kepada Allah Swt. yang telah memberi kesempatan bagi saya untuk menyelesaikan makalah ini. Selain itu, ucapan terima kasih juga saya sampaikan kepada dosen pengampu mata kuliah Strategi Algoritma yang telah memberi saya banyak ilmu sehingga saya dapat menyelesaikan makalah ini. Saya juga berterima kasih kepada kedua orangtua dan saudara-saudara saya, yang selalu mendukung dan mendoakan saya.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. Pencocokan String (2021). Strategi Algoritma. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf (diakses pada 19 Mei 2022 pukul 17.30 WIB).

[2] https://www.cuelogic.com/blog/the-levenshtein-algorithm (diakses pada 19 Mei 2022 pukul 20.00 WIB).

[3] https://www.geeksforgeeks.org/applications-of-string-matching-algorithms/ (diakses pada 19 Mei 2022 pukul 20.15 WIB).

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Tangerang Selatan, 23 Mei 2022

Atabik Muhammad Azfa Shofi, 13520159