

**PERBANDINGAN ALGORITMA *WINNOWING* DAN ALGORITMA *RABIN-KARP PADA* APLIKASI PENDETEKSI KESAMAAN DOKUMEN SKRIPSI**

Jumadil Nangia\*, Ida Bagus Gede Pala Asmarab\*, Muh. Ihsan Saritac, Laode Muh. Golok Jayad, Hasmina Tari Mokuie, LM Tajidunf

a Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo

b Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo

c Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo

d Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo

e Teknik Elektro, Fakultas Teknik, Universitas Halu Oleo

f Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo

*Naskah Diterima : 11 Maret 2015; Diterima Publikasi : 13 April 2015*

*DOI :* 10.21456/vol7iss1pp

**Abstract**

*Plagiarism is often found in the academic world and is considered a serious offense because it involves taking other people's ideas, opinions, or writings. Thesis is the final work of students that must meet scientific standards and be tailored to their field of study. It is important to check the similarity of theses using a web-based system to prevent plagiarism and ensure academic integrity. This system will be created using two different methods namely Winnowing algorithm and Rabin-Karp algorithm. In principle, these two methods are the same, namely searching for strings that use a hashing function to compare the searched string (m) with the compared string (n) by comparing the results of the hashing function used. However, in the Winnowing algorithm, the difference is that it does not use all hash values from each set of grams formed. The hash value formed in the previous stage will be divided into windows of size w. In this study, system testing used thesis data of informatics engineering students at Halu Oleo University to facilitate checking the level of thesis plagiarism against the Rabin-Karp algorithm and the Winnowing algorithm. In this study, system testing was carried out with blackbox testing. This test is expected to detect the level of similarity of the thesis of each algorithm.*

***Keywords*** : *Detection System, Thesis, Web, Winnowing, Rabin-Karp*

**Abstrak**

Plagiarisme sering ditemukan didunia akademik dan dianggap sebagai pelanggaran serius karena melibatkan pengambilan ide, pendapat, atau tulisan orang lain. Skripsi adalah karya akhir mahasiswa yang harus memenuhi standar ilmiah dan disesuaikan dengan bidang studi mereka. Penting untuk melakukan pengecekan kemiripan skripsi menggunakan sebuah sistem berbasis *web*site untuk mencegah plagiat dan memastikan integritas akademik. Sistem ini akan dibuat menggunakan dua metode yang berbeda yaitu algoritma *Winnowing* dan algoritma *Rabin-karp*. Pada prinsipnya kedua metode ini sama yaitu melakukan pencarian string yang menggunakan fungsi *hashing* untuk membandingkan string yang dicari (m) dengan string yang dibandingkan (n) dengan membandingkan hasil dari fungsi hashing yang digunakan. Namun pada algoritma *Winnowing* perbedaannya tidak menggunakan semua nilai hash dari setiap rangkaian gram yang dibentuk. Nilai *hash* yang dibentuk pada tahap sebelumnya akan dibagi ke dalam *window* berukuran w. Pada penelitian ini pengujian sistem menggunakan data skripsi mahasiswa Teknik Informatika Universitas Halu Oleo untuk memudahkan pengecekan tingkat plagiarisme skripsi terhadap algoritma *Rabin-Karp* dan algoritma *Winnowing*. Pada penelitian ini algoritma *Rabin-Karp* dan algoritma *Winnowing* telah diimplementasikan dengan baik pada sistem pengecekkan kemiripan skripsi mahasiswa. Hasil pengujian akurasi yang diujikan melalui *confusion* *matrix* menghasilkan 100%.

***Keywords*** : *Sistem Pendeteksi, Skripsi, Web, Winnowing, Rabin-Karp*

**1. Pendahuluan**

Perkembangan teknologi informasi pada saat ini berkembang begitu pesat, dengan adanya teknologi dapat membantu atau meringankan pekerjaan manusia. Selain itu berkembangnya suatu teknologi tentu ada dampak positif dan negatifnya, salah satu contoh dampak negatifnya adalah orang-orang cenderung ingin sesuatu yang instan, bentuknya yaitu seperti plagiat. Plagiat atau plagiarisme adalah pengambilan karangan, pendapat, dan sebagainya dari orang lain dan menjadikannya seakan-akan karangan atau pendapat sendiri, misalnya menerbitkan karya tulis orang lain atas nama dirinya sendiri.

Plagiarisme selalu menjadi sorotan, tak terkecuali pada sektor akademis baik dari tingkat sekolahan hingga jenjang perguruan tinggi. Tindakan ini sangat tidak mencerminkan seorang yang terpelajar, selain mematikan kreativitas juga kebebasan berinovasi menjadi sempit karena kurang menghargai karya orang lain. Plagiarisme atau penjiplakan karya tulis orang lain dalam dunia akademik dipicu oleh banyak faktor. Salah satu faktor pemicunya adalah penulis (mahasiswa) ingin segera menyelesaikan skripsinya agar bisa meraih gelar akademik secepatnya tanpa harus bekerja keras sesuai proses riset dan penulisan ilmiah yang benar. Salah satu plagiarisme yang sering terjadi di kalangan mahasiswa adalah plagiarisme di dalam tugas akhir (skripsi).

Tugas akhir (skripsi) merupakan salah satu bentuk karya tulis ilmiah yang dibuat oleh mahasiswa pada tahap akhir dari masa studinya. Tugas akhir (skripsi) dibuat berdasarkan hasil penelitian, kajian terhadap permasalahan yang diperoleh dari pelaksanaan PKL (Praktik Kerja Lapangan), atau permasalahan rill lainnya. Jadi dengan adanya sistem ini dapat digunakan oleh dosen dan pihak jurusan untuk mendeteksi tingkat plagiarisme tugas akhir (skripsi) mahasiswa.

Tindakan plagiarisme secara perlahan harus dicegah dan dihilangkan dengan melakukan pendeteksian plagiat secara manual maupun dengan memanfaatkan metode pencocokan *string*. Dengan demikian melakukan pendeteksian plagiarisme secara manual sangat tidak efektif sehingga algoritma *Winnowing* atau algoritma *Rabin-Karp* dapat menjadi solusi untuk menyelesaikan hal tersebut.

Algoritma *Winnowing* adalah sebuah metode yang digunakan untuk mendeteksi tindakan plagiarisme dengan menggunakan teknik *hashing*. Fungsinya sebagai sebuah *fingerprint* dokumen, dengan *input* berupa dokumen teks. Algoritma ini akan menghasilkan sekumpulan nilai *hash* yang dihasilkan dari perhitungan ASCII pada setiap karakter dalam dokumen. Nilai-nilai *hash* ini kemudian digunakan sebagai *fingerprint* untuk mendeteksi adanya plagiarisme (Sunardi dkk., 2018).

Algoritma *Rabin-Karp* adalah metode pencarian string yang dikembangkan oleh *Michael Rabin* dan *Richard Karp*. Algoritma ini menggunakan teknik *hashing* untuk membandingkan *string* yang dicari dengan *string* yang dibandingkan. Fungsi *hashing* digunakan untuk mengubah data menjadi bilangan bulat yang relatif kecil yang dapat digunakan sebagai indeks pada *array*. Proses *hashing* merupakan bagian penting dari algoritma *Rabin-Karp*, dimana karakter atau tanda baca ditransformasikan menjadi nilai atau angka dengan menggunakan representasi ASCII (Suryati dkk., 2018).

Penelitian ini menggunakan algoritma *Winnowing* dan algoritma *Rabin-Karp* karena kedua algoritma ini memiliki proses sama dalam pengambilan *fingerprint* yaitu menggunakan nilai-nilai *hash* dari representasi ASCII sebagai data dalam mendeteksi tingkat kemiripan suatu *string*. Algoritma *Winnowing* danalgoritma *Rabin-Karp* juga dianggap cocok untuk diimplementasikan pada aplikasi pendeteksi tingkat plagiarisme tugas akhir (skipsi) mahasiswa. Peneliti akan mengimplementasikan kedua algoritma tersebut serta melakukan perbandingan antar kedua algoritma.

Berdasarkan uraian di atas di harapkan dapat merancang dan mengembangkan sebuah sistem pendeteksi plagiarisme tugas akhir (skripsi) mahasiswa yang berbasis web. Sistem ini bertujuan untuk membantu mengatasi masalah plagiarisme dengan menerapkan algoritma *Winnowing* dan algoritma *Rabin-Karp*. Bagaimana menjelaskan rancang bangun sistem ini dapat mendeteksi kesamaan dokumen tugas akhir (skripsi) secara efisien. Selain itu, akan membahas implementasi kedua algoritma tersebut dalam konteks pengembangan sistem ini. Sistem ini diharapkan dapat memberikan kontribusi dalam mencegah tindakan plagiarisme di dunia akademik dengan memberikan alat yang efektif bagi dosen dan pihak jurusan untuk mendeteksi plagiarisme dalam tugas akhir (skripsi) mahasiswa.

**2. Kerangka Teori**

*2.1.* *Information retrieval*

Information retrieval adalah bidang yang mempelajari tentang sistem pengindeksan, pencarian, dan penyimpanan data yang tidak terstruktur, khususnya teks. Dalam proses pengindeksan, terdapat tahap stemming yang merupakan proses mengubah bentuk kata menjadi kata dasar. Proses ini sangat dipengaruhi oleh bahasa yang digunakan, karena harus mengaplikasikan aturan morfologi dari bahasa tersebut (Zulfikar, 2017).

*2.2. ASCII (American Standard Code for Information Interchange)*

ASCII (American Standard Code for Information Interchange) adalah standar internasional untuk kode huruf dan simbol seperti Hex dan Unicode. Namun, ASCII lebih bersifat universal dan selalu digunakan oleh komputer dan alat komunikasi lain untuk menunjukkan teks. ASCII memiliki komposisi bilangan biner sebanyak 7 bit, namun disimpan sebagai kode 8 bit dengan menambahkan satu bit tambahan yang sering digunakan untuk uji prioritas (Tantoni dan Zaen, 2018).

****

Gambar 1. Tabel ASCII

*2.3. Algoritma Rabin-Karp*

 Algoritma *Rabin-Karp* adalah metode pencarian *string* yang menggunakan fungsi hashing untuk membandingkan *string* yang dicari (m) dengan *string* yang dibandingkan (n) dengan membandingkan hasil dari fungsi *hashing* yang digunakan. Algoritma ini menggunakan fungsi *hashing* untuk menemukan pola dalam *string* teks. Algoritma ini menggunakan *k-gram* dan *hashing* sebagai karakteristiknya dan dilakukan setelah proses *preprocessing* (Purba dan Situmorang, 2017)*.* Fungsi *hashing* adalah proses penting dalam algoritma *Rabin-Karp*. Ini adalah fungsi matematis yang mengubah data menjadi bilangan bulat yang relatif kecil yang dapat digunakan sebagai indeks pada *array* (Suryati dkk., 2018).

*2.4. Rolling Hash*

*Rolling hash* merupakan tahapan melakukan proses *hashing* pada *string* yang telah dibagi menjadi *k-gram*. Fungsi ini akan mengubah atau mentransformasikan data untuk menciptakan *fingerprint* yang disebut *hash value*. Fungsi *hash* yang baik adalah yang menghasilkan sedikit *hash collision*. Solusi yang digunakan untuk mengatasi *hash collision* adalah menggunakan *rolling hash* (Ramadhani, 2015). Dalam prosesnya, digunakan basis yang biasanya adalah bilangan prima yang cukup besar untuk meminimalkan terjadinya tabrakan. Salah satu *hash* yang sering digunakan adalah *rolling hash* dengan persamaan (1) :

 (1)

*2.5 Dice’s Similarity Coficient (Deteksi Kemiripan)*

*Dice coefficient* atau *dice similarity* adalah metode untuk menghitung tingkat kemiripan antara 2 objek dengan cara mengalikan jumlah nilai irisan antara dokumen dan *query*, lalu membagi dengan jumlah nilai dari dokumen dan *query* (Mangundap dkk., 2022). Terdapat pada persamaan (2) :

 (2)

*2.3. Algoritma Winnowing*

Algoritma *Winnowing* adalah metode yang digunakan untuk mengecek kemiripan kata (*fingerprint* dokumen) untuk mendeteksi plagiarisme. Secara teknis, *Winnowing* adalah perluasan dari implementasi algoritma *Rabin-Karp fingerprint* dengan penambahan metode *window* (Apridiansyah dkk., 2022).

Tahapan *text preprocessing* sebagaimana yang telah dilakukan pada algoritma *Rabin-Karp* juga dilakukan pada algoritma *Winnowing*. Lanjutan proses memotong *string* sepanjang k dalam proses perhitungan *k-gram* dan lanjutan proses *hashing* terhadap seluruh pecahan *string* juga sama antara kedua algoritma. Perbedaannya dimana algoritma *Winnowing* tidak menggunakan semua nilai *hash* dari setiap rangkaian *gram* yang dibentuk. Nilai *hash* yang dibentuk pada tahap sebelumnya akan dibagi ke dalam *window* berukuran w. *Window* adalah proses pembentukan *substring* dari nilai *hash* sepanjang *w-gram*. Dari proses *Winnowing* akan menghasilkan *fingerprint* yang nanti akan digunakan untuk pencocokan plagiasi. *Window* pertama berisi nilai *hash* pertama sampai nilai *hash* ke-w. *Window* kedua dibentuk dari nilai *hash* kedua sampai nilai *hash* ke-w+1 dan seterusnya sampai terbentuk *window* dari seluruh nilai *hash*. Dari *window* yang telah dibentuk dilakukan pemilihan nilai *hash* terkecil pada tiap *window* untuk dijadikan *fingerprint* tiap dokumen. Dari rangkaian nilai *hash* inilah akan dipilih nilai *hash* terkecil, jika terdapat 2 atau lebih nilai yang sama, nilai *hash* terkecil yang paling kanan yang akan di pilih (Sugiono dkk., 2018).

*2.4. Uji Validasi Normalized Mean Absolute Error (NMAE)*

*Normalized Mean Absolute Error* (NMAE) adalah metrik yang digunakan untuk mengukur tingkat kesalahan atau error antara prediksi dan nilai sebenarnya dalam konteks yang mempertimbangkan rentang atau skala nilai sebenarnya (William I dkk, 2019). Kesalahan absolut mengacu pada perbedaan besar tanpa memperhatikan arah kesalahan.

**3. Metode**

Pada bagian ini menjelaskan mengenai langkah-langkah yang dilakukan untuk mencapai tujuan penelitian.

*3.1. Metode Pengumpulan Data*

 Dalam penelitian ini metode pengumpulan data yang digunakan dalam perancangan sistem dengan melakukan studi literatur. Studi literatur dilakukan dengan cara mengumpulkan referensi yang berkaitan dengan objek penelitian yang dilakukan dengan mencari dan mempelajari buku-buku, jurnal maupun *e*-*book*, penelusuan *internet* yang berkaitan dengan algoritma *Winnowing* dan algoritma *Rabin-Karp* sebagai penunjang yang dapat dijadikan sebagai landasan penelitian. Sehingga algoritma *Winnowing* dan algoritma *Rabin-Karp* dapat diimplementasikan dalam aplikasi pendeteksi plagiarisme tugas akhir (skripsi) Mahasiswa Teknik Informatika di Universitas Halu Oleo.

*3.2. Metode Pengembangan Sistem*

Metode pengembangan sistem yang digunakan pada penelitian ini adalah metode *Rational Unified Process* (RUP). Dalam metode pengembangan sistem ini, terdiri dari 4 tahapan pengembangan perangkat lunak, yaitu :

1. Permulaan (*Inception)*
2. Perluasan/Perencanaan (*Elaboration)*
3. Konstruksi (*Construction*)
4. Transisi (*Transition*)

*3.3. Preprocessing Data*

*Text* *preprocessing* merupakan langkah awal yang akan dilakukan untuk memproses teks agar dapat digunakan dalam proses *tokenizing* teks. Tujuan dari *text* *preprocessing* adalah untuk mengembalikan teks menjadi bahasa yang alami. Secara umum dalam tahapan *text* *preprocessing*, yaitu :

1. Data *Input*
2. *Case Folding*
3. *Punctuation Removal*
4. *Stopword Removal*
5. *Stemming*
6. Penghapusan Spasi

*3.4. Tokenizing k-gram*

*Tokenizing* proses membagi karakter-karakter teks yang telah melalui proses *text preprocessing* sepanjang nilai k yang dibaca berurutan dari awal hingga akhir. Pada contoh kasus ini peneliti menggunakan nilai *k-gram* yaitu 4-*gram*.

*3.5. Algoritma Rabin-Karp*

Pada proses algoritma *Rabin-Karp* dalam kasus ini akan mengambil nilai *hash* berdasarkan algoritma *Rabin-Karp* lalu melakukan perhitungan persentase kemiripan teks (*similarity*) menggunakan metode *dice’s similarity coeficient*.

*3.6. Algoritma Winnowing*

Pada proses algoritma *Winnowing* dalam kasus ini akan membentuk *window* dari nilai *hash* berdasarkan algoritma *Winnowing*, lalu melakukan perhitungan persentase kemiripan teks (*similarity*) menggunakan metode *dice’s similarity coficient*. Pada kasus ini peneliti menggunakan nila *w-gram* yaitu 4-*gram.*

**4. Hasil dan Pembahasan**

*4.1. Flowchart*

*4.1.1. Flowchart Sistem*

Gambar 2 merupakan flowchart system dimana user akan diminta melakukan login terlebih dahulu untuk mengakses system. Setelah berhasil maka user akan dialihkan ke halaman dashboard. Pada tampilan dashboard, user dapat melihat data. Kemudian user dapat mengupload file skripsi untuk melakukan pengecekan plagiarisme yang berada pada database. Kemudian akan menampilkan persentase kemiripan dari file skripsi tersebut.



Gambar 2 *Flowchart* Algoritma Sistem

*4.1.2. Flowchart Algoritma Rabin-Karp*

Gambar 3 merupakan *flowchart* algoritma *rabin-karp* user mulai mengupload file skripsi dan sistem akan mengambil file skripsi yang terdapat pada database kemudian akan masuk ke proses *text preprocessing,* setelah melakukan proses *text preprocessing,* selanjutnya ke tahap *tokenizing* kemudian dari proses tersebut akan diambil nilai *hash* nya berdasarkan algoritma *rabin-karp*, kemudian dihitung nilai *dice’s similarity* menggunakan *dice’s similarity conficient*.



Gambar 3. *Flowchart* Algoritma *Rabin-Karp*

* + 1. *Flowchart Algoritma Winnowing*

Gambar 4 merupakan *flowchart* Algoritma *winnowing* user mulai mengupload file skripsi dan sistem akan mengambil file skripsi yang terdapat pada database kemudian akan masuk ke proses *text preprocessing,* setelah melakukan proses *text preprocessing,* selanjutnya ke tahap *tokenizing* kemudian dari proses tersebut akan membentuk nilai *hash* menjadi *window* (*W-Gram*) kemudian akandiambil nilai *hash* nya berdasarkan algoritma *winnowing*, kemudian dihitung nilai *dice’s similarity* menggunakan *dice’s similarity conficient*



Gambar 4. *Flowchart* Algoritma *Winowwing*

*4.2. Use Case*



Gambar 5. *Use Case* Diagram

Gambar 4 merupakan use diagram dari sistem SKRIPSEA dimana admin dan mahasiswa dapat melakukan log in terlebih dahulu. Admin dapat melakukan kelola data skripsi, kelola kategori skripsi, kelola data mahasiswa dan *logout* sistem. Sedangkan mahasiswa dapat melakukan upload skripsi, cek kesamaan skripsi, lihat data skripsi dan *logout* pada sistem.

*4.3. Implementasi Interface*

Pada Gambar 6. merupakan tampilan halaman *login user* dimana admin dan mahasiswa memasukkan *username* dan *password* terlebih dahulu untuk masuk ke dalam aplikasi.

****

Gambar 6. Tampilan Halaman *Login User*

Gambar 7 merupakan halaman *Dashboard* merupakan halaman awal ketika *user login* sebagai mahasiswa dan admin.

****

Gambar 7. Tampilan Halaman *Dashboard*

Gambar 8 merupakan halaman data mahasiswa merupakan halaman dimana terdapat data-data mahasiswa. Pada halaman ini terdapat fitur tambah, edit, detail dan hapus data mahasiswa.



Gambar 8. Tampilan Halaman Data Mahasiswa

Gambar 9 halaman ini berisi tentang data skripsi mahasiswa. Pada halaman ini terdapat fitur tambah, detail, edit dan hapus data skripsi dan terdapat fitur pengecekan plagiarisme.

****

Gambar 9. Tampilan Halaman Data Skripsi

Gambar 10 halaman ini terdapat kategori dari skripsi yang terbagi menjadi 3 yaitu Komputasi Berbasis Jaringan, Komputasi Cerdas dan Visualisasi dan Rekayasa Perangkat Lunak. Dimana terdapat fitur tambah data kategori, hapus, edit dan detail kategori.

****

Gamabar 10. Tampilan Halaman Kategori Skripsi

Gambar 11 merupakan halaman mahasiswa dapat mengunggah *File* skripsi yang ingin dicek plagiarismenya yang kemudian sistem akan membandingkan *File* skripsi anda dengan semua *File* skripsi yang sudah ada dalam sistem.



Gambar 11. Tampilan Halaman Plagiarisme

Gambar 12 merupakan halaman saata mahasiswa telah melakukan upload file di halaman plagiarisme kemudian akan lanjut ke halaman pengecekan plagiarisme dimana pada halaman ini akan menampilkan detail dari skripsi seperti judul, jumlah halaman dan *File* skripsi. Kemudian lakukan pengecekan plagiarisme dengan menekan tombol mulai pengecekan.



Gambar 12. Tampilan Halaman Pengecekan Plagiarisme

Gambar 13 merupakan halaman hasil dari proses cek plagiarisme. Kemudian akan tampil tabel kemiripan skripsi yang sudah diunggah dengan skripsi yang lainnya.



Gambar 13. Tampilan Halaman Hasil Pengecekan Plagiarisme

Gambar 14 merupakan halaman apabila mahasiswa hanya ingin membandingkan antara dua dokumen saja untuk dicek plagiarismenya.



Gambar 14. Tampilan Halaman Bandingkan Dua Dokumen

Gambar 15 merupakan halaman setelah *user* memasukan atau mengunggah dua dokumen skripsi yang ingin di bandingkan. Kemudian akan tampil halaman detail dari dua dokumen tersebut. Untuk melakukan pengecekan mahasiswa dapat menekan tombol mulai pengecekan.



Gambar 15. Tampilan Halaman Detail Perbandingan Dua Dokumen

Gambar 16 merupakan halaman yang akan menampilkan tabel hasil dari pengecekan plagiarisme dari perbandingan dua dokumen dengan menggunakan dua algoritma yaitu *Rabin-karp* dan *Winnowing* dengan menampilkan *similarity* dan waktu proses dokumen dari kedua algoritma tesebut.

****

Gambar 16. Tampilan Halaman Hasil Perbandingan Dua Dokumen

*4.4. Pengujian Black Box*

Berdasarkan pengujian menggunakan metode *blackbox testing*, ditemukan bahwa semua fitur yang telah direncanakan berfungsi dengan baik sesuai dengan perancangan yang telah dibuat.

Tabel 1. Pengujian *Black Box*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Nama Form | Detail Uji | *Output* | Hasil Uji |  |
| 1.  | Pengujian *Login* | Menu *Login* | Menampilkan Halaman *login* user | Sesuai |  |
| 2.  | Pengujian Menu Mahasiswa | Menu tambah mahasiswa | Menampilkan halaman tambah data mahasiswa | Sesuai |  |
| Menu edit mahasiswa | Menampilkan halaman edit data mahasiswa | Sesuai |  |
| Menu hapus mahasiswa | Menampilkan halaman hapus data mahasiswa | Sesuai |  |
| 3. | Pengujian Menu Skripsi | Menu tambah skripsi | Menampilkan halaman tambah data skripsi | Sesuai |  |
| Menu edit skripsi | Menampilkan halaman edit data skripsi | Sesuai |  |
| Menu hapus skripsi | Menampilkan halaman hapus data skripsi | Sesuai |  |
| 4. | Pengujian Menu Kategori Skripsi | Menu tambah kategori skripsi | Menampilkan halaman tambah kategori skripsi | Sesuai |  |
| Menu edit kategori skripsi | Menampilkan halaman edit kategori data skripsi | Sesuai |  |
|  | Menu hapus kategori skripsi | Menampilkan halaman hapus kategori data skripsi | Sesuai |  |
| 5. | Pengujian Menu Cek Kesamaan Skripsi | Menu *Upload* dokumen skripsi | Menampilkan halaman dokumen skripsi yang telah diupload  | Sesuai |  |
| 6. | Pengujian Menu Perbandingan Dua Dokumen Skripsi | Menu Perbandingan Dua Dokumen Skripsi | Menampilkan halaman dokumen hasil kemiripan dokumen skripsi  | Sesuai |  |

*4.5.* *Pengujian Validasi NMAE (Normalized Mean Absolute Error)*

Dalam pengujian validasi NMAE ini akan mengukur tingkat kesalahan atau error antara prediksi dalam hal ini algoritma *Rabin-Karp* dan algoritma *Winnowing* terhadap nilai sebenarnya dalam hal *platform* pengecekan kesamaan yaitu *Plagiarism Checker X* sebagai aplikasi atau *platform* yang dipercaya dalam konteks yang mempertimbangkan rentang atau skala nilai sebenarnya.

Tabel 2. Konversi Nilai *Similarity* Ke Bilangan Desimal

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| NO | Dokumen Skripsi 1 | Dokumen Skripsi 2 | *Rabin-Karp* | *Winnowing* |  | *Plagiarism Checker X* |
| 1 | Skripsi\_1 | Skripsi\_2 | 0.1336 | 0.1068 |  | 0.11 |
| 2 | Skripsi\_1 | Skripsi\_3 | 0.2366 | 0.1818 |  | 0.18 |
| 3 | Skripsi\_1 | Skripsi\_4 | 0.0622 | 0.051 |  | 0.06 |
| 4 | Skripsi\_1 | Skripsi\_5 | 0.1172 | 0.0881 |  | 0.1 |
| 5 | Skripsi\_1 | Skripsi\_6 | 0.0682 | 0.0506 |  | 0.07 |
| No | Nama Form | Detail Uji | *Output* | Hasil Uji |  | No |
| 6 | Skripsi\_1 | Skripsi\_7 | 0.1415 | 0.0978 |  | 0.1415 |
| 7 | Skripsi\_1 | Skripsi\_8 | 0.2511 | 0.2223 |  | 0.2 |
| 8 | Skripsi\_1 | Skripsi\_9 | 0.1705 | 0.1267 |  | 0.13 |
| 9 | Skripsi\_1 | Skripsi\_10 | 0.1219 | 0.0769 |  | 0.08 |
| 10 | Skripsi\_1 | Skripsi\_11 | 0.1673 | 0.1333 |  | 0.14 |
| 11 | Skripsi\_1 | Skripsi\_12 | 0.1015 | 0.0647 |  | 0.07 |
| 12 | Skripsi\_1 | Skripsi\_13 | 0.1739 | 0.1026 |  | 0.1 |
| 13 | Skripsi\_1 | Skripsi\_14 | 0.1629 | 0.1278 |  | 0.12 |
| 14 | Skripsi\_1 | Skripsi\_15 | 0.1381 | 0.0833 |  | 0.09 |
| 15 | Skripsi\_1 | Skripsi\_16 | 0.0912 | 0.0686 |  | 0.08 |
| 16 | Skripsi\_1 | Skripsi\_17 | 0.1232 | 0.0837 |  | 0.09 |
| 17 | Skripsi\_1 | Skripsi\_18 | 0.115 | 0.0736 |  | 0.08 |
| 18 | Skripsi\_1 | Skripsi\_19 | 0.1204 | 0.0913 |  | 0.08 |
| 19 | Skripsi\_1 | Skripsi\_20 | 0.1313 | 0.0834 |  | 0.08 |
| 20 | Skripsi\_1 | Skripsi\_21 | 0.1417 | 0.0888 |  | 0.11 |

Berikut perhitungan NMAE *Rabin-Karp* terhadap *Plagiarism Checker X* :

Berikut perhitungan NMAE *Winnowing* terhadap *Plagiarism Checker X* :

Jadi berdasarkan perhitungan NMAE di atas diperoleh nilai NMAE (*Rabin-Karp, Plagiarism Checker X*)yaitu 0.182967033 dan nilai NMAE (*Winnowing, Plagiarism Checker X*) yaitu 0.019481654, maka algoritma *Winnowing* memiliki kinerja yang lebih baik daripada algoritma *Rabin-Karp*.

*4.6. Space Complexity*

*Space Complexity* adalah salah satu aspek penting dalam analisis kompleksitas algoritma yang mengukur berapa banyak memori (ruang) yang dibutuhkan oleh suatu algoritma selama proses. Pemahaman *Space Complexity* penting karena penggunaan memori yang berlebihan dapat mengakibatkan kinerja yang buruk dan bahkan kegagalan eksekusi program pada sistem dengan keterbatasan sumber daya.

Tabel 3. *Space Complexity*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Dokumen Skripsi 1 | Dokumen Skripsi 2 | *Rabin-Karp* | *Winnowing* |
| 1 | Skripsi\_1 | Skripsi\_2 | 26.88 Mb | 94.16 Mb |
| 2 | Skripsi\_1 | Skripsi\_3 | 26.85 Mb | 94.91 Mb |
| 3 | Skripsi\_1 | Skripsi\_4 | 26.81 Mb | 93.46 Mb |
| 4 | Skripsi\_1 | Skripsi\_5 | 33 Mb | 123.26 Mb |
| 5 | Skripsi\_1 | Skripsi\_6 | 28.22 Mb | 106.01 Mb |
| 6 | Skripsi\_1 | Skripsi\_7 | 26.64 Mb | 91.25 Mb |
| 7 | Skripsi\_1 | Skripsi\_8 | 27.12 Mb | 90.28 Mb |
| 8 | Skripsi\_1 | Skripsi\_9 | 27.95 Mb | 102 Mb |
| 9 | Skripsi\_1 | Skripsi\_10 | 27.68 Mb | 98.07 Mb |
| 10 | Skripsi\_1 | Skripsi\_11 | 28 Mb | 102.88 Mb |
| 11 | Skripsi\_1 | Skripsi\_12 | 27.29 Mb | 99.61 Mb |
| 12 | Skripsi\_1 | Skripsi\_13 | 28.36 Mb | 108.61 Mb |
| 13 | Skripsi\_1 | Skripsi\_14 | 26.76 Mb | 93.61 Mb |
| 14 | Skripsi\_1 | Skripsi\_15 | 32.92 Mb | 121.72 Mb |
| 15 | Skripsi\_1 | Skripsi\_16 | 28.4 Mb | 108.51 Mb |
| 16 | Skripsi\_1 | Skripsi\_17 | 28.14 Mb | 103.66 Mb |
| 17 | Skripsi\_1 | Skripsi\_18 | 32.65 Mb | 118.13 Mb |
| 18 | Skripsi\_1 | Skripsi\_19 | 26.61 Mb |  91.08 Mb |
| 19 | Skripsi\_1 | Skripsi\_20 | 27.06 Mb | 95.95 Mb |
| 20 | Skripsi\_1 | Skripsi\_21 | 33.39 Mb | 125.86 Mb |

Berdasarkan analisis tabel *Space Complexity* di atas maka, algoritma *Rabin-Karp* memiliki rata-rata *Space Complexity* sebesar 28.05 Mb dan algoritma *Winnowing* memiliki rata-rata *Space Complexity* sebesar 97.85 Mb. Jika dilihat besar rata-rata nilai *Space Complexity* dari masing-masing algoritma maka, algoritma *Winnowing* memiliki rata-rata *Space Complexity* yang besar dibandingkan algoritma *Rabin-Karp.*

**5. Kesimpulan**

Berdasarkan hasil penelitian yang telah dilakukan mengenai Implementasi Algoritma *Rabin-Karp* dan *Winnowing* dalam Pengecekkan Kemiripan Skripsi Mahasiswa diperoleh beberapa kesimpulan sebagai berikut:

Berdasarkan hasil evaluasi uji coba sistem yang telah dibuat dapat diketahui bahwa performa hasil persentase dari implementasi algoritma *Rabin-Karp* dan *Winnowing* dalam pengecekkan kemiripan skripsi mahasiswa memiliki tingkat akurasi dengan hasil yaitu 100 %.

Algoritma *Rabin-Karp* memiliki tingkat *similarity* yang tinggi daripada algoritma *Winnnowing* dan aplikasi *Plagiarism* *Checker X*. Namun, algoritma *Winnowing* memiliki tingkat konsistensi atau rata-rata *similarity* yang sama dengan aplikasi *Plagiarism Checker X*.

Algoritm *Rabin-Karp* memiliki waktu proses yang cenderung lebih cepat daripada algoritma *Winnowing*.

Berdasarkan nilai rata-rata *Space Complexity*, algoritma *Winnowing* memiliki nilai rata-rata *Space Compelxity* yang lebih besar dibandingkan algoritma *Rabin-Karp*.

Algoritm *Rabin-Karp* memiliki waktu proses yang cenderung lebih cepat daripada algoritma *Winnowing*.

Jumlah skripsi yang terdaftar dalam sistem, besarnya ukuran skripsi, dan banyaknya karakter di dalamnya yang akan diuji untuk kesamaannya akan memiliki dampak pada waktu proses pengecekan kesamaan (*similarity*).

**Daftar Pustaka**

Apridiansyah, Y., A. Wijaya, Dan A. Purjiawan, 2022, Penerapan Fungsi Metode Rolling Hash Pada Algoritma Winnowing Untuk Mendeteksi Kemiripan Teks Abstrak Berbasis Web, *Jurnal Media Infotama*, 18, 1, 128–133.

Purba, A. H. Dan Z. Situmorang, 2017, Analisis Perbandingan Algoritma Rabin-Karp Dan Levenshtein Distance Dalam Menghitung Kemiripan Teks *Jurnal Teknik Informatika Unika St. Thomas (Jtiust)*, 2, 2, 24–32.

Ramadhani, S, 2015, Sistem Pencegahan Plagiarism Tugas Akhir Menggunakan Algoritma Rabin-Karp (Studi Kasus: Sekolah Tinggi Teknik Payakumbuh), *Jurnal Teknologi Informasi & Komunikasi Digital Zone*, 6, 1, 44–52.

Sugiono, Herwin, Hamdani, Dan Erlin, 2018, Aplikasi Pendeteksi Tingkat Kesamaan Dokumen Teks: Algoritma Rabin Karp Vs. Winnowing, *Jurnal Teknologi & Komunikasi Digital Zone*, 9, 1, 82–93.

Sunardi, A. Yudhana, Dan I. A. Mukaromah, 2018, Implementasi Deteksi Plagiarisme Menggunakan Metode N-Gram Dan Jaccard Similarity Terhadap Algoritma Winnowing*, Transmisi,* 20, 3, 105-110.

Sunardi, A. Yudhana, Dan I. A. Mukaromah, 2019, Indonesia Words Detection Using Fingerprint Winnowing Algorithm, *Jurnal Informatika*, 13, 1, 7–15.

Suryati, T., Y. Wibisono, Dan Y. Wihardi, 2018, Aplikasi Deteksi Plagiarisme Dokumen Skripsi Dengan Algoritma Rabin-Karp, *Jurnal Teori Dan Aplikasi Ilmu Komputer*, 1, 2, 91–95.

Tantoni, A. Dan M. T. A. Zaen, 2018, Implementasi Double Caesar Cipher Menggunakan ASCII, *Jurnal Informatika & Rekayasa Elektronika (Jire)*, 1, 2, 24–32.

Zulfikar, A. F, 2017, Pengembangan Algoritma Stemming Bahasa Indonesia Dengan Pendekatan Dictionary Base Stemming Untuk Menentukan Kata Dasar Dari Kata Yang Berimbuhan, *Jurnal Informatika Universitas Pamulang*, 2, 3, 143–146.