RANCANG BANGUN APLIKASI DETEKSI PLAGIARISME MENGGUNAKAN ALGORITMA *MANBER* DENGAN PENDEKATAN *BIWORD*

SKRIPSI

Oleh :
FAUZIYAH AMINI
NIM. 16650102

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2020

RANCANG BANGUN APLIKASI DETEKSI PLAGIARISME MENGGUNAKAN ALGORITMA *MANBER* DENGAN PENDEKATAN *BIWORD*

SKRIPSI

Diajukan kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
FAUZIYAH AMINI
NIM. 16650102

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG

2020
HALAMAN PERSETUJUAN

RANCANG BANGUN APLIKASI DETEKSI PLAGIARISME MENGGUNAKAN ALGORITMA *MANBER* DENGAN PENDEKATAN *BIWORD*

SKRIPSI

Oleh:
FAUZIYAH AMINI
NIM. 16650102

Telah Diperiksa dan Disetujui untuk Diuji

Tanggal : 11 Mei 2020

Dosen Pembimbing I	Dosen Pembimbing II
Fresy Nugroho, M.T	Dr. Cahyo Crysdian
NIP. 19710722 201101 1 001	NIP. 19740424 200901 1 008

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian

NIP. 19740424 200901 1 008

RANCANG BANGUN APLIKASI DETEKSI PLAGIARISME MENGGUNAKAN ALGORITMA *MANBER* DENGAN PENDEKATAN *BIWORD*

SKRIPSI

Oleh:
FAUZIYAH AMINI
NIM. 16650102

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 15 Juni 2020

Susunan Dewan Penguji :			**Tanda Tangan**
Penguji Utama	:	Yunifa Miftachul Arif, M.T NIP. 196830616 2001101 1 004	( )
Ketua Penguji	:	Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )
Sekertaris Penguji	:	Fresy Nugroho, M.T NIP. 19710722 201101 1 001	( )
Anggota Penguji	:	Dr. Cahyo Crysdian NIP. 19740424 200901 1 008	( )

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama : Fauziyah Amini

NIM : 16650102

Fakultas/Jurusan : Sains dan Teknologi/Teknik Infomatika

Judul Skripsi : Rancang Bangun Aplikasi Deteksi Plagiarisme Menggunakan Algoritma *Manber* Dengan Pendekatan *Biword*

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilalihan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan Skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 15 Juni 2020
Yang membuat pernyataan,

Fauziyah Amini
NIM. 16650102

HALAMAN MOTTO

"*Penuhilah hidupmu dengan berbagi manfaat kepada orang lain, dimanapun kamu berada*"

HALAMAN PERSEMBAHAN

الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ

Puji syukur kehadirat Allah, shalawat dan salam bagi Rasul-Nya

Penulis persembahkan sebuah karya ini kepada:

Kedua orang tua tercinta penulis, Bapak Nurcahyo dan Ibu Kismayanti yang selalu memberikan motivasi agar terus semangat dalam rangka menuntut ilmu. Serta berkat doa merekalah penulis bisa berada sampai di titik pencapaian sekarang ini.

Dosen pembimbing penulis Bapak Fresy Nugroho, M.T dan Bapak Dr. Cahyo Crysdian yang selalu memberikan masukan dan saran dalam berjalannya penelitian ini, sehingga penulis dapat menyelesaikan skripsi dengan tepat waktu.

Seluruh dosen Teknik Informatika UIN Maulana Malik Ibrahim Malang, yang telah memberikan ilmu dan pengalaman yang akan berguna pada masa depan penulis kelak.

Orang-orang yang penulis penulis sayangi, yang tak bisa penulis sebutkan satu per satu yang selalu memberikan support dan mendoakan penulis untuk menyelesaikan skripsi ini.

Penulis ucapkan terimakasih banyak. Semoga tali persaudaraan kita tetap terjaga dan selalu diridhoi Allah SWT. Aamiin Allahumma Aamiin.

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Puji dan syukur kehadirat Allah subhanahu wa ta'ala yang telah melimpahkan rahmat dan karuniaNya kepada kita, sehingga penulis bisa menyelesaikan skripsi dengan tepat waktu, yang berjudul "Rancang Bangun Aplikasi Deteksi Plagiarisme Menggunakan Algoritma *Manber* Dengan Pendekatan *Biword*". Tujuan dari penyusunan skripsi ini guna memenuhi salah satu syarat untuk bisa menempuh ujian sarjana komputer pada Fakultas Sains dan Teknologi (FSAINTEK) Program Studi Teknik Informatika di Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang. Dalam pengerjaan skripsi ini telah melibatkan banyak pihak yang sangat membantu dan memotivasi dalam proses pengerjaan skripsi ini. Oleh sebab itu, penulis sampaikan rasa terima kasih kepada:

1. Prof. Dr. Abdul Haris, M.Ag selaku Rektor Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
2. Dr. Sri Harini,M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
3. Dr. Cahyo Crysdian, Selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang sekaligus selaku Dosen Pembimbing II.
4. Fresy Nugroho, M.T, selaku Dosen Pembimbing I sekaligus selaku Dosen wali penulis yang telah memberi arahan mulai dari perencanaan penelitian skripsi hingga tahapan penyusunan skripsi selesai.
5. Para staff laboran dan Admin jurusan Teknik Informatika UIN Malang yang telah membantu dalam urusan administrasi penulis dalam menyelesaikan penyusunan skripsi.

6. Orang tua dan keluarga penulis yang telah banyak memberikan doa dan dukungan baik secara moral ataupun materi.
7. Teman-teman seperjuangan Teknik Informatika UIN Maulana Malik Ibrahim Malang angkatan 2016 yang selalu menemani suka maupun duka dalam perjalanan menuntut ilmu.
8. Semua pihak yang telah membantu dalam penyusunan skripsi ini yang tidak bisa penulis sebutkan semuanya.

Penulis menyadari bahwa dalam skripsi ini masih terdapat kekurangan, namun penulis berharap semoga skripsi ini bisa memberikan manfaat untuk UIN Maulana Malik Ibrahim Malang khususnya bagi penulis secara pribadi.

Malang, 15 Juni 2020

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Amini, Fauziyah. 2020. **Rancang Bangun Aplikasi Deteksi Plagiarisme Menggunakan Algoritma *Manber* Dengan Pendekatan *Biword***. Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
Pembimbing: (I) Fresy Nugroho, M.T. (II) Dr. Cahyo Crysdian.

Kata kunci: *Biword*, *Manber*, Plagiarisme.

Plagiarisme semakin tidak dapat dihindari dalam dunia pendidikan, khususnya pada kalangan mahasiswa. Maka dari itu, pada penelitian ini akan dibahas mengenai rancang bangun aplikasi deteksi plagiarisme, sebagai upaya untuk meminimalisir praktik plagiarisme di kalangan mahasiswa. Pada penelitian ini, penulis menggunakan algoritma *Manber* dengan pendekatan *Biword* untuk menghitung presentase tingkat plagiarisme dalam sebuah dokumen. Algortima *Manber* menggunakan teknik *fingerprinting* untuk mencari kesamaan pada teks. Algoritma *Manber* memiliki dua parameter, yaitu nilai prima dan nilai *chunk*(P), parameter inilah yang mempengaruhi nilai *error* yang dihasilkan dari algoritma *Manber*. Pada penelitian ini dilakukan uji coba terhadap lima konfigurasi nilai parameter yang berbeda, dari hasil pengujian penulis mendapatkan hasil rata-rata nilai *error* konfigurasi I = 4.5%, konfigurasi II = 4.9%, konfigurasi III = 4.7%, konfigurasi IV = 4.8%, konfigurasi V = 5.9%. Berdasarkan pengujian yang telah dilakukan, konfigurasi I menghasilkan nilai *error* terkecil, yaitu 4,5%, dengan nilai P*(chunk)* = 4 dan nilai prima=2. Konfigurasi I juga menghasilkan rata-rata waktu proses tercepat dalam mendeteksi plagiarisme yaitu 2.06 detik. Hal tersebut membuktikan bahwa konfigurasi tersebut merupakan konfigurasi terbaik dibandingkan konfigurasi lainnya.

ABSTRACT

Amini, Fauziyah. 2020. ***Design and Build Plagiarism Detection Application Using Winnowing Algorithm with Triword Approach***. Undergraduate thesis. Department of Informatics Engineering, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University of Malang. Supervisor: Fresy Nugroho, M.T. (II) Dr. Cahyo Crysdian.

Keywords: *Biword, Manber, Plagiarism.*

Plagiarism is increasingly unavoidable in the world of education, especially among students. Thus, this research will discuss the design of plagiarism detection applications, as an effort to minimize the practice of plagiarism among students. In this study, the authors used the Manber algorithm with the Biword approach to calculate the percentage level of plagiarism in a document. Algortima Manber uses fingerprinting techniques to look for similarities in text. Manber's algorithm has two parameters, namely prime value and chunk value (P), this parameter which affects the error value generated from the Manber algorithm. In this study, a trial was conducted on 5 different parameter configuration values, from the test results the writer got the average value of configuration error I = 4.5%, configuration II = 4.9%, configuration III = 4.7%, configuration IV = 4.8%, configuration V = 5.9%. Based on testing that has been done, configuration I produces the smallest error value, which is 4.5%, with a value of P (chunk) = 4 and prime value = 2. Configuration I also produces the fastest average processing time in detecting plagiarism which is 2.06 seconds. This proves that the configuration is the best configuration compared to other configurations.

الملخص

ميني، فوزية .2020. الخطط في بناء آلة بحث الانتحال باستخدام اللوغاريتم النيبري بنظرية بيورد .البحث .كلية العلوم قسم الهندسة المعلوماتية و التكنولوجية جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج
المشرف: (1) فرسي نوغرهو الماجستير (2) الدكتور جهيو كريسديان.

الكلمات الرئيسية : الانتحال ,النيبيري ,بيورد

الانتحال لا يستطيع أن يبتعد في التعاليم .خصوصا بين طلاب وطالبات المرحلة الجامعية .فلذلك في هذا البحث سأفيض الخطط في بناء آلة بحث الانتحال كالمحاولة في امتناعه .المؤلف يستخدم اللوغاريتم النيبري بنظرية بيورد لحساب نسبة الانتحال في ملف .اللوغاريتم النيبري يستخدم تقنية البصمات لبحث المساواة في نص .اللوغاريتم النيبري يملك معيارين و هو القيمة الأولية والقيمة القطعة (P). هذان اللذان يؤثران قيمة الخطأ التي حصلت من اللوغاريتم النيبري. ويقام في هذا البحث على خمس القيم المختلفة .و قد حصل المؤلف من التجارب بمعدل 4,9 %والتشكيل II بمعدل 4,5 %والتشكيل I على قيمة الخطأ في التشكيل بمعدل 5,9 .%استند V بمعدل 4,8 %والتشكيل IV بمعدل 4,7 %والتشكيل III إلى التجارب أن التشكيل الأول هو أقل قيمة الخطأ بنتيجة 4,5 %و القيمة القطعة و القيمة الأولية =2 و التشكيل الأول هو أسرع في بحث الانتحال بمعدل 4 = ثانية .فأستنتج أن التشكيل الأول هو خير من الآخر 2,06

BAB I
PENDAHULUAN

1.1 Latar Belakang

Fenomena plagiarisme kini semakin marak di kalangan masyarakat umum, bahkan plagiarisme juga sudah memasuki ranah dunia pendidikan. Hal ini jelas menurunkan kualitas pendidikan serta bertolak belakang dengan prinsip pendidikan yang ingin mengeluarkan *output* berupa sumber daya manusia yang berilmu dan berakhlak mulia. Plagiarisme termasuk dalam salah satu tindak kejahatan akademik, karena plagiarisme merupakan pencurian ide atau gagasan orang lain tanpa mencantumkan sumber aslinya. Senada dengan hal tersebut, Tyantoro (2014) secara gamblang mengatakan bahwa plagiarisme adalah kejahatan akademik dan hal itu termasuk kejahatan akademik level tertinggi .

Menurut Kamus Besar Bahasa Indonesia plagiarisme merupakan penjiplakan yang melanggar hak cipta. Sedangkan menurut (Ridhatillah, 2003) plagiarisme adalah tindakan penyalahgunaan, pencurian atau perampasan, penerbitan, pernyataan atau menyatakan sebagai milik sendiri sebuah pikiran, ide, tulisan, atau ciptaan yang sebenarnya milik orang lain. Dalam pandangan agama, praktik plagiarisme juga menjadi hal yang harus dihindari oleh setiap orang. Hal ini sesuai dengan yang tercantum dalam firman Allah surat Asy-Syu'ara ayat 183 :

وَلَا تَبۡخَسُواْ ٱلنَّاسَ أَشۡيَآءَهُمۡ وَلَا تَعۡثَوۡاْ فِي ٱلۡأَرۡضِ مُفۡسِدِينَ ١٨٣

"*Dan janganlah kamu merugikan manusia pada hak-haknya dan janganlah kamu merajalela di muka bumi dengan membuat kerusakan"*

Islam merupakan agama yang sempurna dengan segala ketentuan-ketentuan yang telah diatur di dalam Al-Quran dan Sunnah Rasulullah SAW. Islam sudah mengatur sedemikian rupa segala hal dalam kehidupan manusia. Sebagai seorang yang beragama Islam atau muslim yang taat dan beriman kepada Allah SWT hendaknya memiliki akhlak yang sesuai dengan Al-Quran dan Sunnah, karena Al-Quran merupakan petunjuk bagi umat manusia dalam mengarungi hidup di dunia ini agar selamat menuju kehidupan akhir yang sesungguhnya. Berdasarkan firman Allah diatas Allah melarang hamba nya untuk merugikan manusia pada hak-hak nya. Dalam kasus plagiarisme, mengakui atau mengambil hasil karya orang lain baik sebagian atau seluruhnya termasuk merugikan hak orang lain.

Di Indonesia sebenarnya sudah diberlakukan UU tentang plagiarisme yang terdapat pada Peraturan Menteri Pendidikan Nasional No.17/2010 dan pelakunya diancam dengan hukuman yang cukup berat. Sesuai UU No.20/2003, dijelaskan bahwa pelaku tindak plagiat diberikan sanksi bahwa lulusan perguruan tinggi yang karya ilmiahnya digunakan untuk memperoleh gelar akademik, profesi atau vokasi, terbukti merupakan jiplakan, dicabut gelarnya (Pasal 25 Ayat 2). Kemudian lulusan yang tersebut pada Pasal 25 Ayat 2 dipidana dengan pidana penjara paling lama dua tahun, dan atau pidana denda paling banyak Rp. 200.000.000,- (dua ratus juta rupiah).

Namun pemberlakuan peraturan ini masih tidak mampu mengatasi tindak plagiarisme di kalangan mahasiswa. Di kalangan mahasiswa, praktek plagiarisme

sudah tidak dapat dihindari, mengingat mahasiswa sangat akrab dengan dunia internet serta didukung dengan mudahnya akses pencarian informasi, hal ini memudahkan pengguna untuk menyimpan, mencari, dan mengolah data dengan cepat. Sehingga mempermudah untuk melakukan plagiarisme dengan bebas, baik untuk kepentingan tugas harian, maupun tugas akhir. Berdasarkan permasalahan yang telah dipaparkan di atas diperlukan aplikasi pendeteksi plagiarisme dalam rangka meminimalisasi praktik plagiarisme di kalangan mahasiswa.

Ada beberapa metode yang dapat digunakan untuk membangun aplikasi pendeteksi plagiarisme. Salah satunya adalah metode dokumen *fingerprinting*, prinsip kerja dari metode dokumen *fingerprinting* adalah dengan menggunakan teknik *hash*ing. Teknik *hash*ing adalah sebuah fungsi yang mengkonversi setiap *string* menjadi bilangan kemudian menyimpannya dalam sebuah skema atau bagan. Skema digital dokumen *fingerprinting* terdiri dari sejumlah posisi yang diberi tanda di dalam dokumen, algoritma *fingerprinting* yang akan memilih tanda yang akan di tambahkan untuk setiap posisi tergantung pada jumlah salinan. Algoritma yang digunakan dalam metode dokumen *fingerprinting* adalah algoritma *Rabin Karp*, *Winowwing*, dan *Manber*. Dalam penelitian ini, peneliti akan menggunakan algoritma *Manber*. Algoritma *Manber* melakukan pemilihan nilai *fingerprint*s yang telah diperoleh dari proses *hash*ing dengan memilih nilai *fingerprint*s yang memenuhi kriteria 0 *mod p.* Selain menggunakan algoritma *Manber*, dalam penelitian ini juga dilakukan pendekatan menggunakan pendekatan konsep *Biword*. Pendekatan konsep *Biword* ini dilakukan agar menjaga keutuhan frasa dari sebuah kalimat.

Algoritma *Manber* dengan pendekatan *Biword* dapat diimplementasikan untuk mendeteksi kemiripan antar teks. Algoritma ini dalam menjalankan prosesnya lebih sederhana, lebih cepat dari algoritma *Winowwing*, serta mudah untuk diimplementasikan. Tingkat presentase nilai *error* dari Algoritma ini juga dapat ditingkatkan hingga batas maksimal dengan melakukan pengujian terhadap nilai parameter nilai *prima*, dan nilai *P(chunk)*. Dengan beberapa keunggulan tersebut, penulis berasumsi bahwa Algoritma *Manber* dengan pendekatan *Biword* dapat diimplementasikan dalam aplikasi pendeteksi plagiarisme.

Berdasarkan latar belakang yang telah dipaparkan diatas, penulis berencana membangun aplikasi pendeteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*. Penggunaan algoritma *Manber* dengan pendekatan *Biword* ini diharapkan dapat memberikan hasil yang akurat dan efektif dalam mendeteksi kemiripan dokumen antar teks, sehingga dapat meminimalisasi plagiarisme di kalangan mahasiswa.

1.2 Pernyataan Masalah

1. Seberapa tinggi presentase nilai *error* dari algoritma *Manber* dengan pendekatan *Biword* dalam mendeteksi plagiarisme ?
2. Seberapa cepat waktu yang diperlukan oleh algoritma *Manber* dengan pendekatan *Biword* dalam mendeteksi plagiarisme?

1.3 Tujuan Penelitian

1. Mengukur presentase nilai *error* dari algoritma *Manber* dengan pendekatan *Biword* dalam mendeteksi plagiarisme
2. Mengukur kecepatan algoritma *Manber* dengan pendekatan *Biword* dalam mendeteksi plagiarisme.

1.4 Manfaat Penelitian

Adapun manfaat penelitian ini bagi Universitas adalah :

1. Sebagai alat bantu dalam mendeteksi tingkat plagiarisme pada tugas akhir mahasiswa.
2. Mempercepat waktu dalam mendeteksi tingkat plagiarisme pada tugas akhir mahasiswa.
3. Meminimalisir praktek plagiarisme di kalangan mahasiswa.

1.5 Batasan Penelitian

Batasan masalah yang ada pada penelitian ini adalah :

1. Data yang digunakan dalam penelitian ini adalah abstrak tugas akhir mahasiswa jurusan Teknik Informatika UIN Maulana Malik Ibrahim Malang.
2. *Input* aplikasi ini berupa dokumen teks berekstensi *.doc*
3. *Output* yang dihasilkan dari aplikasi ini adalah presentase tingkat plagiarisme dalam persen.
4. Aplikasi ini berbasis *website*
5. Aplikasi ini hanya memproses *input* berupa teks dokumen.

BAB II
TINJAUAN PUSTAKA

2.1 Deteksi Plagiarisme

Terdapat beberapa penelitian yang terkait dengan deteksi plagiarisme, yaitu :

Elbegbayan (2005) meneliti tentang performa algoritma *Winowwing* dengan metode dokumen *fingerprinting* dalam mendeteksi kesamaan antar dua teks. Penelitian ini membandingkan beberapa algoritma dalam metode *fingerprint*, dalam rangka menganalisis kelebihan dan kekurangan masing-masing algoritma. Berdasarkan pengujian pada penelitian tersebut disimpulkan bahwa algoritma *Winowwing* merupakan algoritma yang efisien dan dapat memberikan jaminan hasil yang tepat dalam mendeteksi kesamaan antar teks.

Penelitian selanjutnya dilakukan oleh Ceska (2008). Dalam penelitiannya Ceska mengusulkan pendeteksi plagiarisme berdasarkan *singular value decompotition*. Metode ini disebut *SVDPlag*. Penelitian mengusulkan metode baru yang memecahkan asosiasi frasa yang terkandung dalam dokumen teks. Untuk menguji efisiensi metode *SVGPlag* ini, peneliti menggunakan corpus eksperimental 950 dokumen teks tentang politik, yang dibuat dari corpus CTK standar. Percobaan menunjukkan bahwa metode *SVGPlag* secara signifikan meningkatkan akurasi deteksi plagiarisme dan lebih unggul dari metode lain.

Kemudian disusul oleh penelitian yang dilakukan oleh Ridho (2013). Pada penelitian ini ridho mendeteksi presentasi kesamaan antar teks menggunakan algoritma *Winnowing* dengan pendekatan *Biword.* Pada penelitian ini dilakukan pengujian kombinasi nilai masukan *input* yang paling akurat. Berdasarkan hasil

pengujian yang dilakukan, disimpulkan bahwa jika masukan(*input*) nilai bilangan prima yang terkecil menghasilkan nilai *similarity* tertinggi.

Penelitian tentang deteksi tingkat plagiarisme terus dikembangkan oleh Putri *et.al* (2015). Pada penelitian mereka melakukan pencarian pola yang sama pada DNA kanker hati menggunakan algoritma *Wu-Manber*. Dalam penelitian ini dapat menampilkan tingkat kesamaan dari dua buah sekuen DNA kanker hati. Masukan pada penelitian ini berupa barisan DNA manusia, dan keluaran dari penelitian ini adalah presentasi kesamaan serta eksekusi proses. Berdasarkan pengujian pada penelitian tersebut disimpulkan bahwa semakin besar jumlah pola yang ditemukan dan semakin besar *minimum length* serta semakin pendek karakter sekuen pola, maka persentase kesamaan akan semakin besar.

Pada tahun yang sama, Rafieian (2015) Pada penelitian ini Rafieian melakukan riset terhadap pengecekan plagiarisme dalam teks bahasa Persia. Metode yang digunakan adalah *hash-based tree representative fingerprinting.* Penelitian ini menyajikan solusi untuk mendeteksi plagiarisme dengan mengkombinasikan beberapa metode, selain itu pada penelitian ini dapat mendeteksi *similarity* pada makna, kata, dan sinonim. Metode dalam penelitian ini menerapkan representasi *tree* berdasarkan dokumen *fingerprint* yang disusun ke dalam tiga kata (3-*grams)*. Kemudian *gram* dihilangkan dari dokumen menggunakan fungsi *hash* lalu disimpan sebagai *fingerprint* dokumen dalam repositori dokumen referensi, untuk mendeteksi nilai *similarity* pada dokumen tersebut. Berdasarkan hasil dari pengujian penelitian tersebut, menunjukan bahwa metode ini memiiliki 21,15% rata-rata peningkatan jika dibandingkan dengan

algoritma *Winnowing*. Tingkat akurasi dalam menghitung nilai *similarity* pada metode ini adalah 31,65%.

Penelitian terus berlanjut hingga tahun 2018, Sukmana *et.al* (2018) telah melakukan penelitian untuk membandingkan metode *Rabin Karp* murni dengan metode *Rabin Karp* dengan penambahan *stemming* Nazief Adriani. Penelitian ini melakukan perbandingan dengan menghitung kecepatan dan juga hasil *similarity* dari algoritma *Rabin Karp* tanpa menggunakan *stemming* dan dengan menggunakan *stemming*. Berdasarkan pengujian yang dilakukan pada penelitian tersebut dapat disimpulan bahwa *stemming* Nazief Adriani dapat mempercepat waktu eksekusi *Rabin Karp* lebih cepat dengan hasil *similarity* yang hampir sama.

BAB III
DESAIN PENELITIAN

3.1. Pengumpulan Data

Pengumpulan data merupakan kegiatan mengumpulkan data-data yang akan digunakan oleh peneliti dalam rangka mencapai tujuan penelitian. Pengumpulan data dibagi menjadi dua, yaitu pengumpulan data primer dan pengumpulan data sekunder. Pengumpulan data primer dilakukan langsung oleh peneliti dari subyek/obyek penelitian. Pengumpulan data sekunder yaitu pengumpulan data yang diperoleh dari sumber yang sudah ada.

Pada penelitian ini dilakukan beberapa tahap yang dilakukan untuk mengumpulkan data, tahapan ini meliputi :

1. Tugas Akhir Mahasiswa

Obyek yang peneliti gunakan adalah abstrak tugas akhir mahasiswa UIN Maulana Malik Ibrahim Malang jurusan Teknik Informatika. Peneliti mengambil kumpulan abstrak mahasiswa UIN Maulana Malik Ibrahim Malang jurusan Teknik Informatika yang tersedia di *website* http://etheses.uin-malang.ac.id/

2. Konversi PDF

Doukmen Abstrak yang didapatkan peneliti berbentuk format PDF, maka peneliti melakukan konversi dari format .PDF menjadi format .DOC. Hal ini dilakukan agar peneliti dapat menhitung nilai *similarity* dokumen abstrak tersebut.

3. Abstrak Tugas Akhir

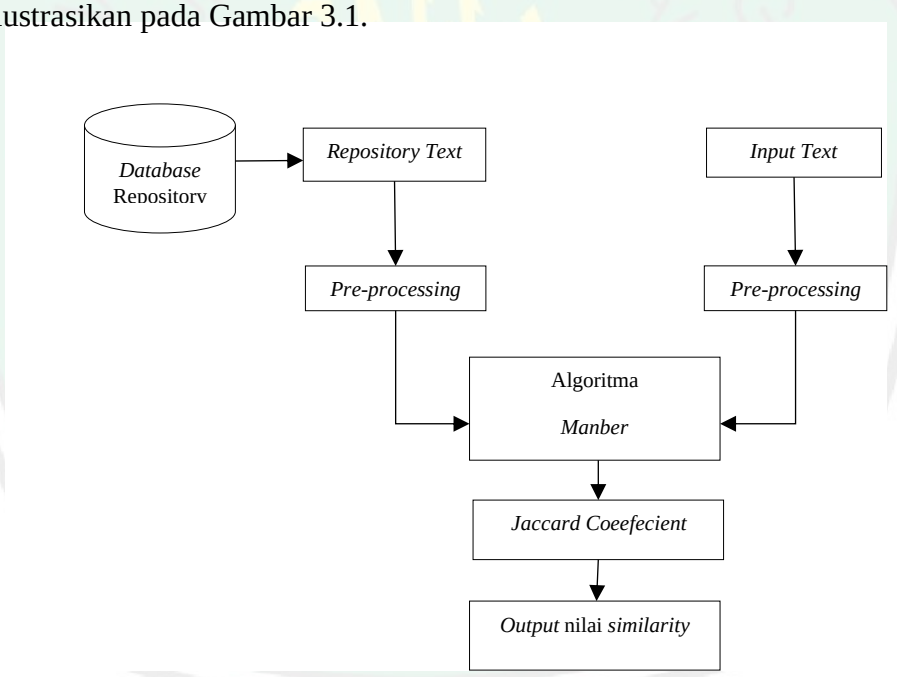
Setelah dokumen ber ekstensi format .DOC, maka peneliti memperoleh abstrak tugas akhir dalam bentuk teks, yang selanjutnya akan dilakukan proses penghitungan nilai *similarity*.

4. *Database*

Pada tahap ini, peneliti memasukan data yang telah didapatkan kedalam *database*. Data yang ada di dalam *database* ini lah yang akan dijadikan acuan perbandingan dalam aplikasi pendeteksi plagiarisme.

3.2 Desain Sistem

Perancangan desain sistem dilakukan agar dapat memenuhi kebutuhan pernggguna sistem dan memberikan gambaran yang rinci kepada pemakai sistem. Sehingga peneliti dapat mengetahui secara jelas langkah apa yang harus dilakukan dalam mencapai tujuan penelitian. Perancangan *design* sistem pada penelitian ini diilustrasikan pada Gambar 3.1.



Gambar 3.1. Desain Sistem

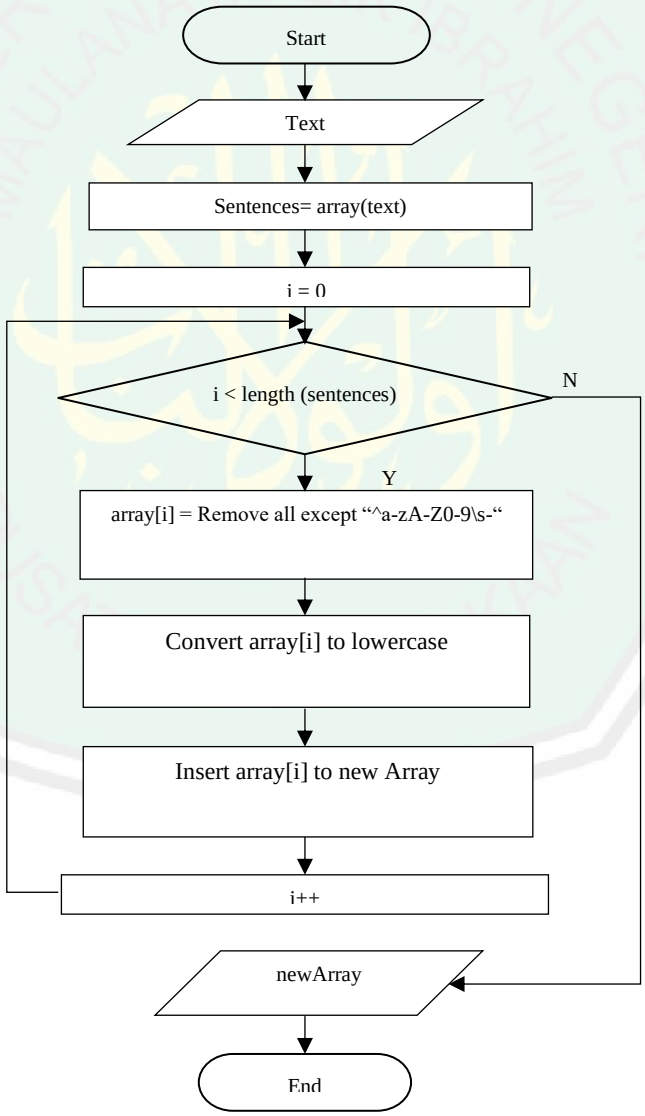
3.2.1 *Database Repository*

Database Repsitory merupakan tempat untuk menyimpan kumpulan abstrak tugas akhir mahasiswa jurusan Teknik Informatika UIN Maulana Malik

Ibrahim Malang. Data yang berada dalam *database repository* ini digunakan sebagai pembanding dalam mendeteksi tingkat plagiarisme.

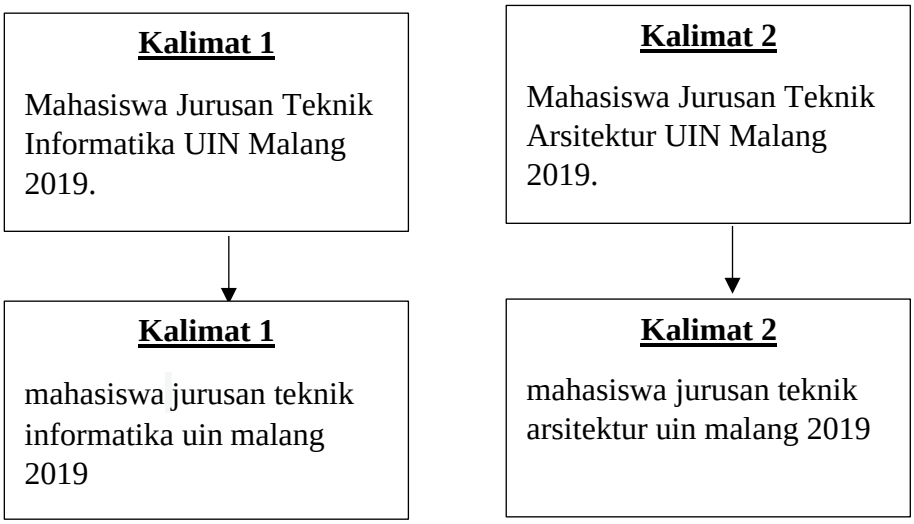
3.2.2 Preprocessing

Tahap *preprocessing* terdiri dari beberapa tahap yaitu, *whitespace insensivity*, *tokenizing*, enkripsi, dan *rolling hash*. *Whitespace insensivity* adalah penghapusan karakter yang tidak relevan seperti tanda baca dan mengubah huruf besar menjadi huruf kecil. Adapun *flowchart* proses *whitespace insensivity* diilustrasikan pada Gambar 3.2.



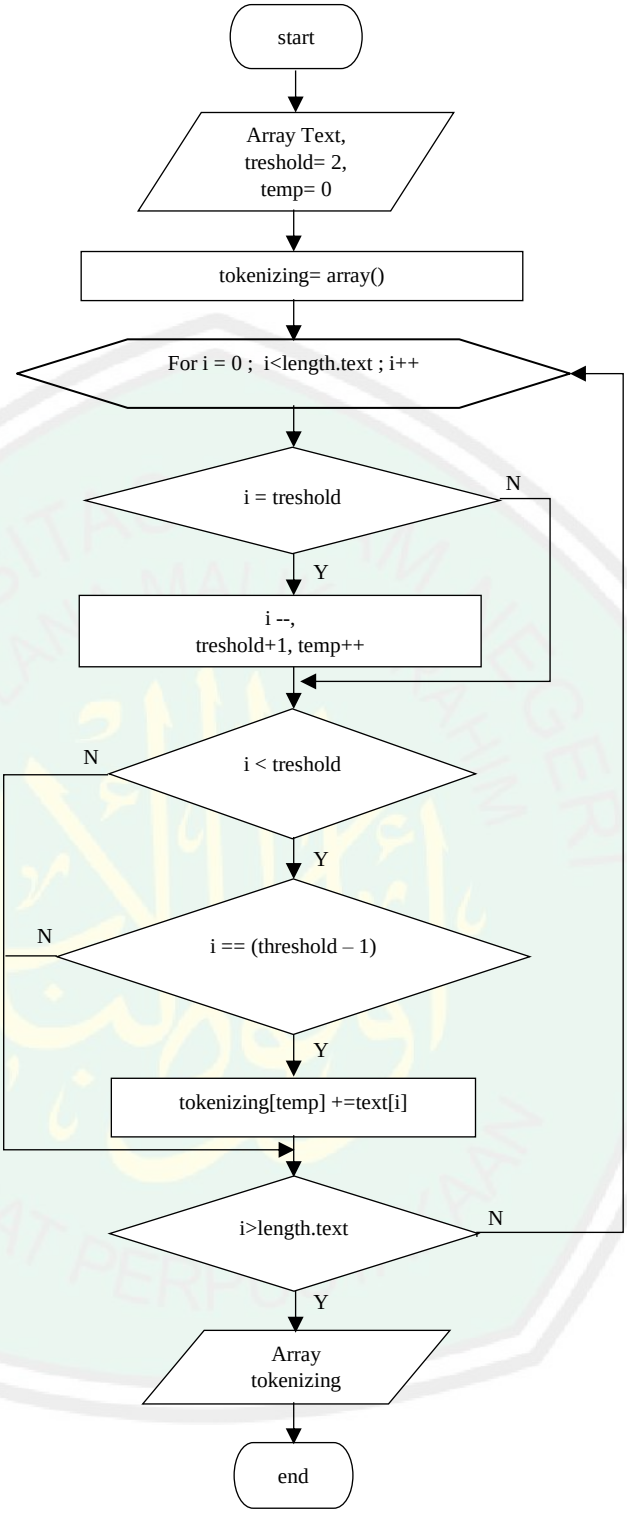
Gambar 3.2. *Flowchart* Proses *Whitespace Insensivity*

Berdasarkan Gambar 3.2 dokumen abstrak tugas akhir mahasiswa yang telah di masukan ke dalam sistem akan dilakukan penghapusan pada semua huruf yang bukan A-Z, a-z, 0-9. Kemudian teks diubah menjadi huruf kecil semua sehingga nantinya hanya karakter-karakter yang berupa huruf atau angka yang akan diproses lebih lanjut. Adapun contoh dari proses *whitespace insensivity* dapat dilihat pada Gambar 3.3.



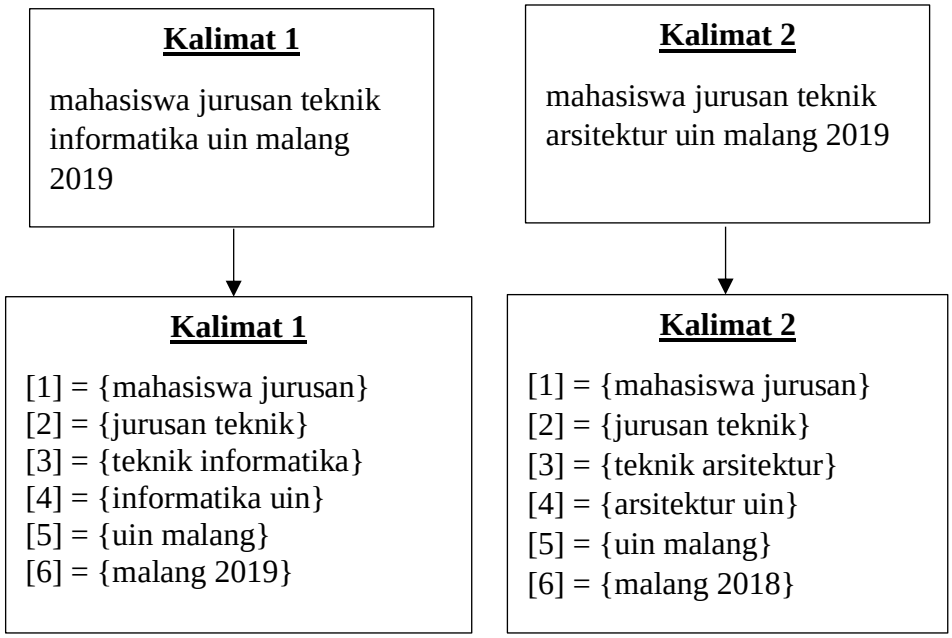
Gambar 3.3. Contoh Proses *Whitespace Insensivity*

Selanjutnya dilakukan proses *tokenizing* atau pemisahan kata, pada tahap ini dokumen teks di bentuk menjadi serangkaian string frasa dalam satu set *Biword* (2 kata). *Biword* adalah salah satu yang termasuk dalam teknik *matching* ketika proses tokenisasi (pemisahan kata) pada teks dokumen. Konsep *Biword* ini mengacu kepada teknik *phrase-based.* Adapun *flowchart* proses *tokenizing* diilustrasikan pada Gambar 3.4.



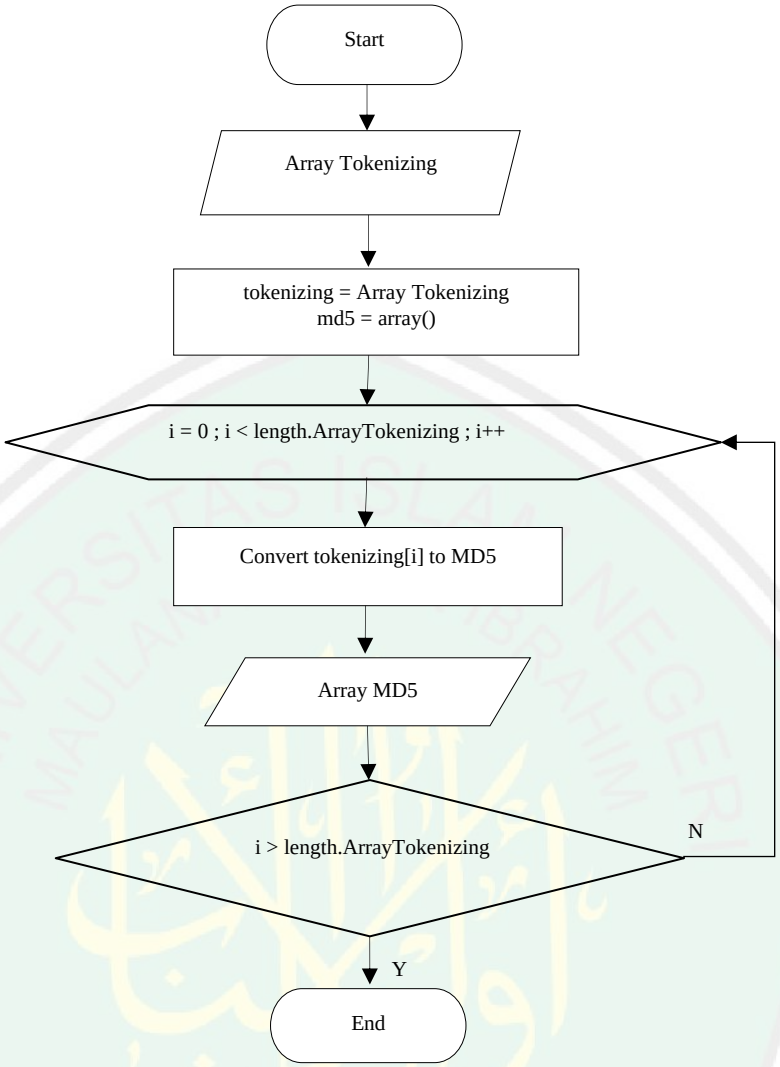
Gambar 3.4. *Flowchart* Proses *Tokenizing*

Berdasarkan Gambar 3.4 teks dokumen yang telah melalui proses *whitespace insensivity*, akan dikelompokan menjadi beberapa *string*. Setiap *string* terdiri dari dua kata (*Biword*). *String* yang sudah terdiri dari dua kata kemudian dimasukan ke dalam *array* untuk melalui tahap selanjutnya. Adapun contoh dari proses *tokenizing* dapat dilihat pada Gambar 3.5.



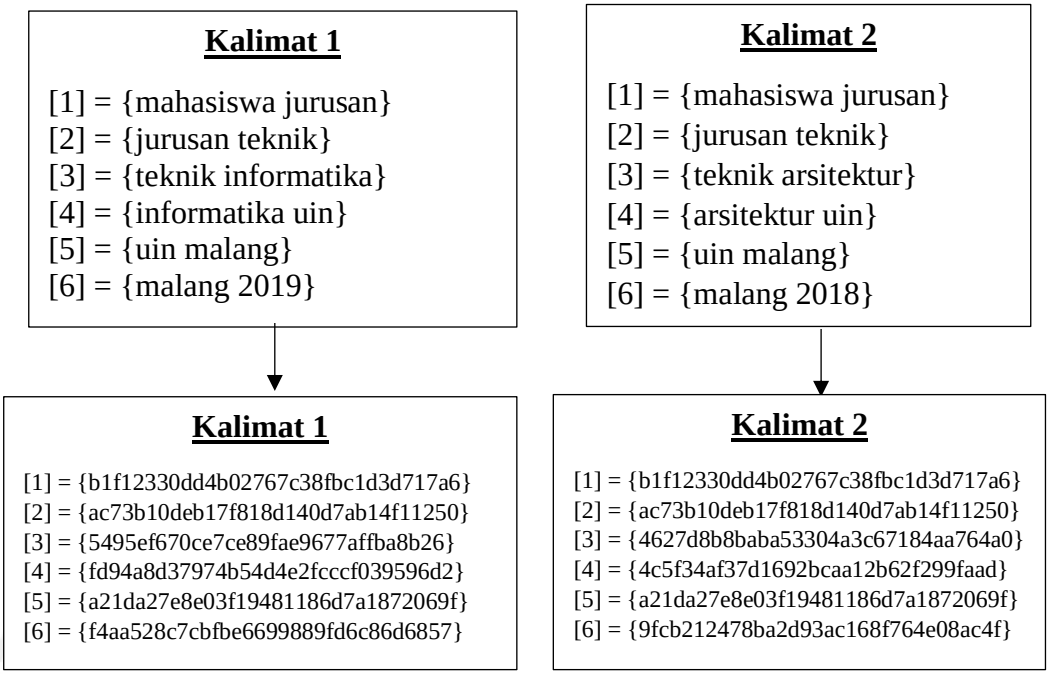
Gambar 3.5. Contoh Proses *Tokenizing*

Setelah teks melalui tahap *tokenizing*, selanjutnya teks akan di enkripsi menggunakan MD5. Pada proses ini setiap *array string* yang sudah terbentuk pada proses sebelumnya di enkripsi menggunakan MD5 agar setiap *array string* mempunyai panjang yang sama. Adapun *flowchart* enkripsi diilustrasikan pada Gambar 3.6.



Gambar 3.6. *Flowchart* Proses Enkripsi

Berdasarkan Gambar 3.6 masing masing *array string* di enkripsi menggunakan algoritma MD5. Hasil enkripsi dimasukan kedalam *array* baru yang akan diproses pada tahap selanjutnya. Adapun contoh dari proses enkripsi MD 5 dapat dilihat pada Gambar 3.7.



Gambar 3.7. Contoh Proses Enkripsi MD5

Langkah selanjutnya yaitu mengkonversi / mengubah nilai-nilai MD5 menjadi nilai *Hash* menggunakan rumus persamaan *Rolling Hash.* Fungsi *hash* merupakan sebuah algoritma yang mengubah teks menjadi sederetan karakter acak yang memiliki jumlah karakter yang sama.

Fungsi yang digunakan untuk menghasilkan nilai *hash* dari rangkaian *string* dalam algoritma *Manber* adalah *rolling hash* . Fungsi *hash* $H_{c1\ldots ck}$ didefinisikan pada persamaan 3.1 :

$$c_1 \times b^{(k-1)} + c_2 \times b^{(k-2)} + \cdots + c(k-1) \times b + c_k \qquad (3.1)$$

Keterangan :

c : nilai ascii karakter

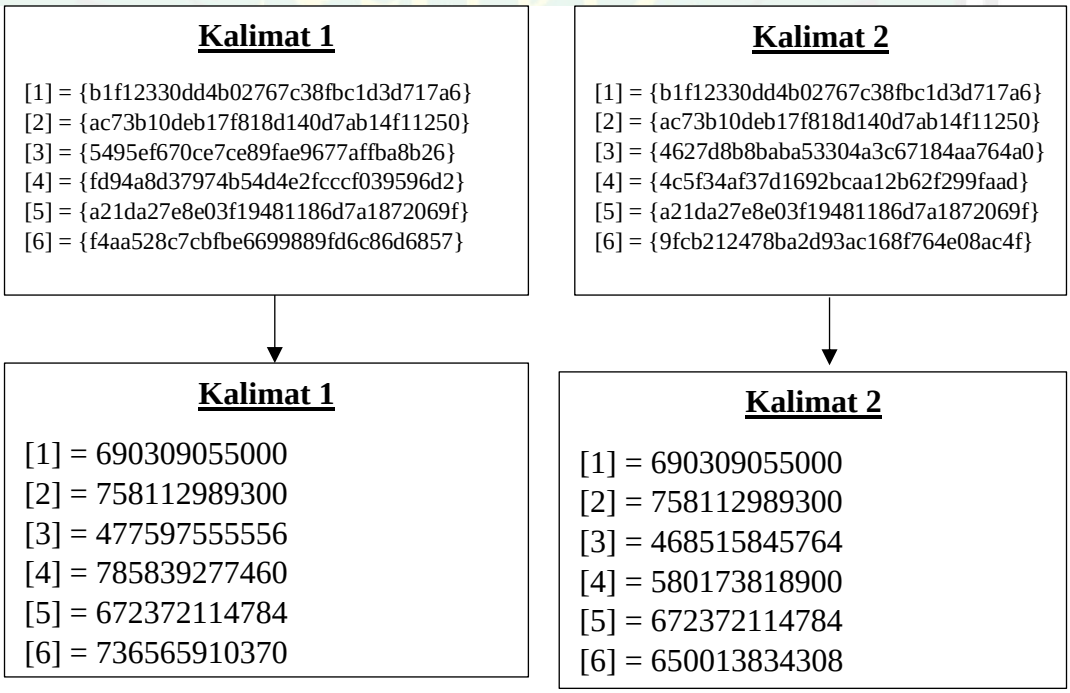
b : basis (bilangan prima)

k : banyak karakter

Berikut adalah contoh perhitungan Fungsi *hash* pada kata “mahasiswa” dengan nilai *prime = 2* :

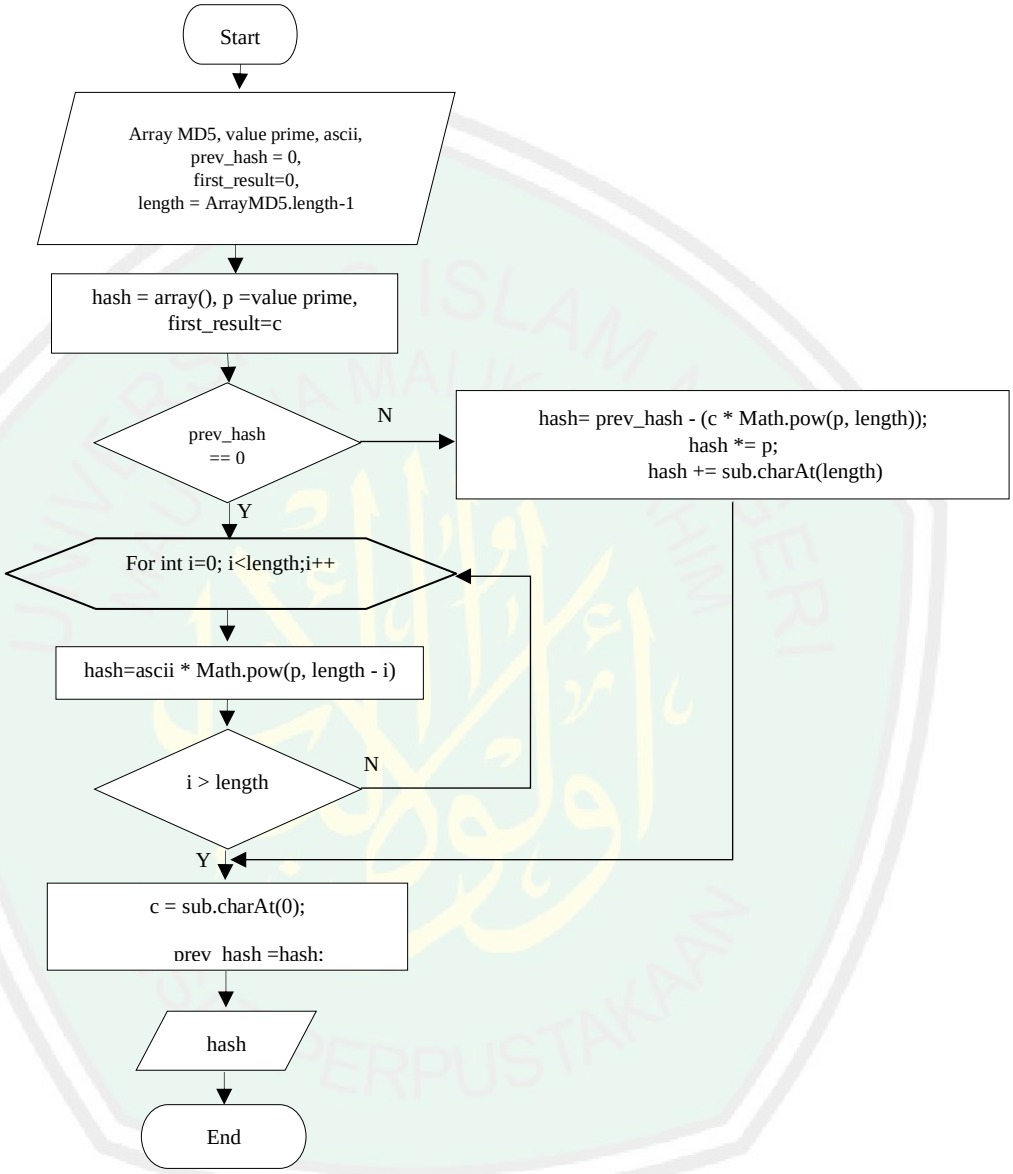
$H_{(mahasiswa)}$ = ascii(m) * $2^{(8)}$ + ascii(a) * $2^{(7)}$ + ascii(h) * $2^{(6)}$ + ascii(a) * $2^{(5)}$ + ascii(s) * $2^{(4)}$ + ascii(i) * $2^{(3)}$ + ascii(s) * $2^{(2)}$ + ascii(w) * $2^{(1)}$ + ascii(a) * $2^{(0)}$
= 109 * 256 + 97 * 128 + 104 * 64 + 97 * 32 + 115 * 16 + 105 * 8 + 115 * 4 +119 * 2 + 97 * 1
= 27904+ 12416 + 6656 + 3104 + 1840+ 840 + 460 + 238 + 97
= 53555

Adapun contoh dari proses *rolling hash* dapat dilihat pada Gambar 3.8.



Gambar 3.8. Proses *Rolling Hash*

Adapun *Flowchart* mengkonversi teks menjadi kode ASCII dengan rumus *rolling hash* dijelaskan pada Gambar 3.9.



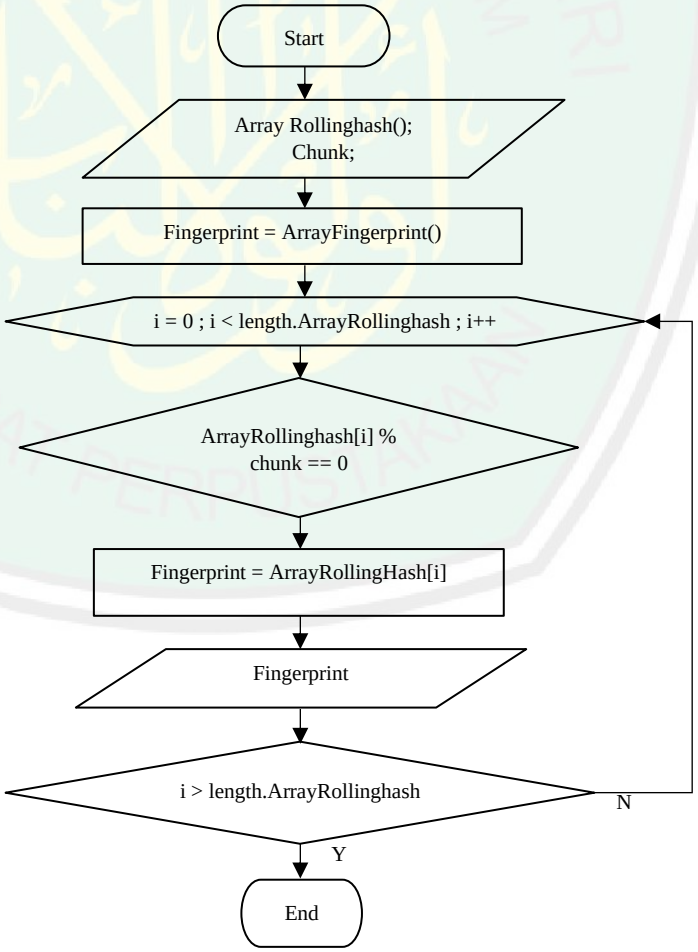
Gambar 3.9. *Flowchart* Proses *Rolling Hash*

Berdasarkan gambar 3.9 *array* hasil enkripsi dari proses sebelumnya akan dikonversi kedalam nilai *hash* menggunakan rumus *rolling hash*. Pertama, nilai basis/prima dan nilai ascii diinisialisasikan kedalam sistem sebagai parameter rumus *rolling hash*. Selanjutnya, *array* hasil enkripsi dimasukan ke dalam rumus

rolling hash sehingga menghasilkan nilai *hash*. Nilai *hash* kemudian dimasukan kedalam array baru untuk diproses pada tahap selanjutnya

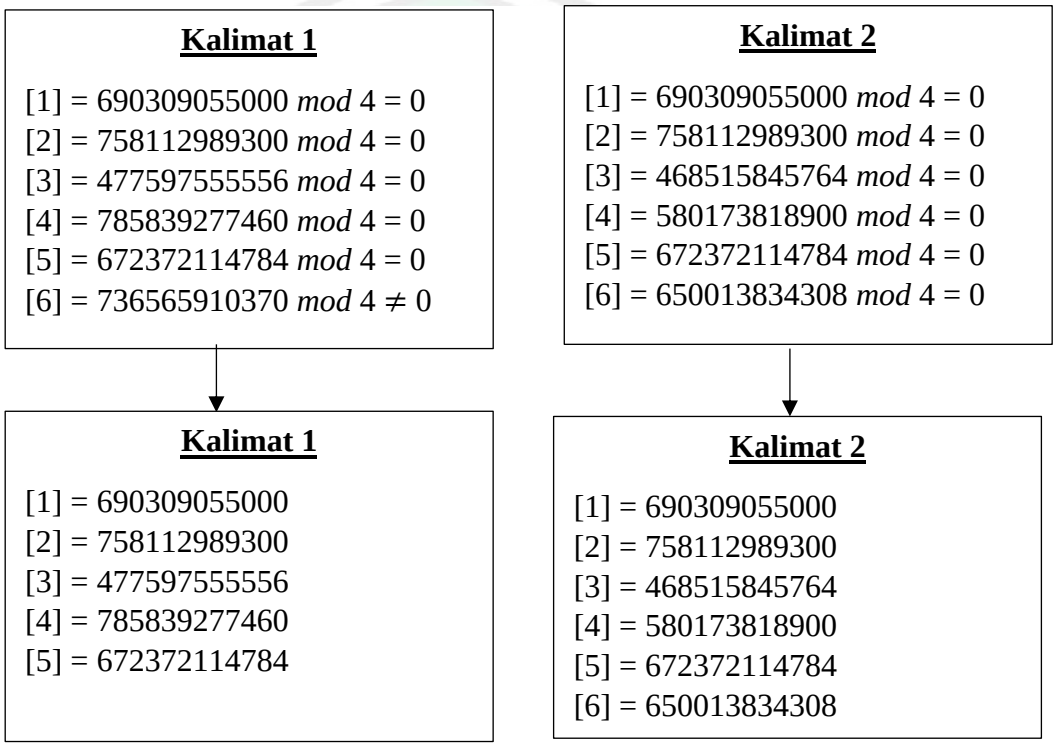
3.2.3 Algoritma *Manber*

Algoritma *Manber* merupakan algoritma yang bisa digunakan untuk mendeteksi kemiripan antar teks. Algoritma *Manber* menggunakan dokumen *fingerprint* yang sama sebagai parameter nilai *similarity* pada teks. Pemililihan nilai *fingerprint* pada penerapan algoritma *Manber* menggunakan nilai *hash* yang habis dibagi *nilai chunk* (p). Atau dengan kata lain 0 mod *p*, dengan *p*=4. Adapun *flowchart* proses pemilihan *fingerprint* dalam algoritma *Manber* diilustrasikan pada Gambar 3.10.



Gambar 3.10. *Flowchart* Proses Pemilihan Nilai *Fingerprint*

Berdasarkan Gambar 3.10 nilai *hash* yang telah didapatkan dari proses sebelumnya akan dibagi dengan nilai p(*chunk*). Nilai hash yang habis dibagi dengan nilai p(*chunk*) dipilih menjadi dokumen *fingerprints* yang akan diproses lebih lanjut pada tahap selanjutnya. Adapun contoh dari proses pemilihan *fingerprint* pada algoritma *Manber* dapat dilihat pada Gambar 3.11.



Gambar 3.11. Proses Pemilihan Nilai *Fingerprint*

3.2.4 *Jaccard Coefficient*

Jaccard Coefficient adalah sebuah persamaan pengukuran dalam menghitung nilai *similarity* antar teks. Dalam melakukan perhitungan presesntase kemiripan pada algoritma *Manber*, menggunakan persamaan *Jaccard Coefficient*. Berdasarkan *fingerprint* yang telah didapatkan pada perhitungan diatas, maka persamaan *Jaccard Coefficient* sebagai berikut :

- *Union* (Gabungan *Fingerprints* 1 dan 2) = *6 + 5 = 11*
- *Intersection* (*fingerprints* yang sama) = *3*
- *Union - Intersection* = 11 – 3 = 8

Jaccard Coefficient $= \frac{|A \cap B|}{|A \cup B|} \times 100\%$

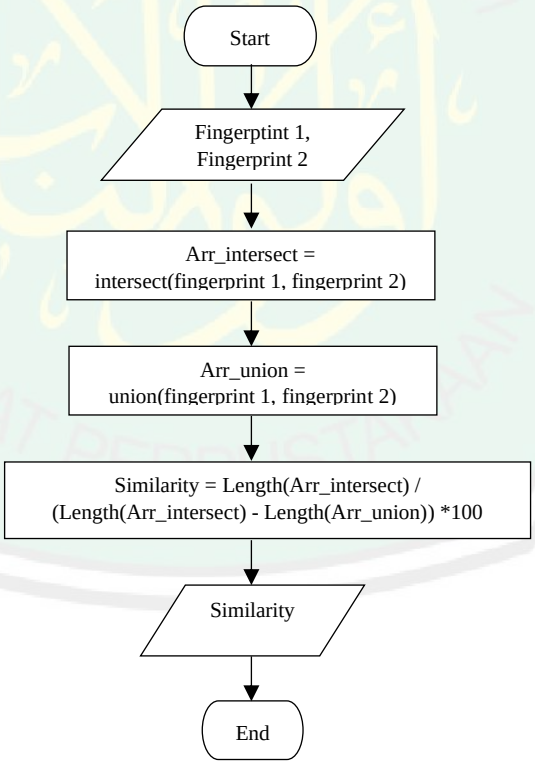
$= \frac{3}{8} \times 100 =$ ***37.5 %***

Keterangan:

A : nilai *fingerprints* dokumen 1

B : nilai *fingerprints* dokumen 2

Dimana D(A,B) adalah nilai *similarity*, $|A \cap B|$ adalah jumlah *fingerprint* yang sama dari dokumen 1 dan 2, $|A \cup B|$ adalah jumlah seluruh *fingerprint* dari dokumen 1 dan 2. Adapun *flowchart* proses penghitungan nilai *similarity* menggunakan *Jaccard Coefficient* diilustrasikan pada Gambar 3.12.



Gambar 3.12. *Flowchart Jaccard Coefficient*

Berdasarkan Gambar 3.12 dokumen *fingerprints* yang telah didapatkan pada proses sebelumnya akan dimasukan kedalam persamaan *Jaccard Coefficient* untuk mendapatkan nilai *similarity* pada sebuah dokumen. Pertama, sistem mencari dokumen *fingerprints* yang sama antar dokumen, kemudian jumlah *fingerprints* yang sama dibandingkan dengan jumlah seluruh dokumen *fingerprints*. Hasil dari persamaan *Jaccard Coefficient* ini adalah presentase plagiarisme sebuah teks dokumen.

3.3 Implementasi Algoritma *Manber* dengan pendekatan *Biword*

Pada tahap ini peneliti melakukan implementasi algoritma *Manber* dengan pendekatan *Biword* ke dalam sistem dengan menerjemahkan rumus dari algoritma *Manber* kedalam bentuk *source code*, sehingga sistem dapat menampilkan hasil nilai *similarity* sebuah dokumen menggunakan algoritma *Manber* menggunakan pendekatan *Biword*. Berikut ini adalah *source code* aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan biword.

1 *Preprocessing*

Preprocessing pada algoritma *Manber* terbagi menjadi 4 tahap, yaitu: *whitespace insensivity*, *tokenizing*, enkripsi, dan *rolling hash*. *Whitespace insensivity* merupakan proses pembuangan karakter yang tidak relevan seperti tanda baca. Adapun *source code* proses *Whitespace insensivity dapat* dilihat pada gambar 3.13.

```
$word = strtolower(str_replace(' ','
',preg_replace("/[^a-zA-Z0-9\s-]/",
"", $word)));

$arr_word = explode(' ' , $word);
```

Gambar 3.13 *Source Code* Proses *Whitespace Insensivity*

Tokenizing merupakan proses pemisahan teks kedalam beberapa kata. Pada penelitian ini teks dipisahkan menjadi 2 kata (*Biword*). Adapun *source code* proses *tokenizing dapat* dilihat pada gambar 3.14.

```
public function GetBiword($word){
$biword = array('');
$batas = 2; $temp = 0;
          for ($i=0; $i < count($arr_word); $i++) {
               if ($i == $batas) {
                    $batas += 1     ;
                    $temp++;
                    $i--;
               }
               if ($i < $batas) {
                    if ($i == ($batas-1)) {
                    $biword[$temp] .= $arr_word[$i].'';
                    } else {
                    $biword[$temp] .= $arr_word[$i].' ';
                    }
               }
          }
          return $biword;
```

Gambar 3.14 *Source Code* Proses *Tokenizing*

Enkripsi pada penelitian ini menggunakan MD5. Proses enkripsi dilakukan agar setiap *array string* mempunyai panjang yang sama. Adapun *source code* proses *tokenizing dapat* dilihat pada gambar 3.15.

```
public function GetMd5($biword){
            for ($i=0; $i < count($biword);
$i++) {
                  $biword[$i] =
md5($biword[$i]);
            }
            return $biword;
      }
```

Gambar 3.15 *Source Code* Proses Enkripsi

Rolling Hash merupakan proses mengkonversi / mengubah nilai-nilai enkripsi MD5 menjadi nilai *Hash* menggunakan rumus persamaan *Rolling Hash.* Adapun *source code* proses *Rolling Hash dapat* dilihat pada gambar 3.16.

```
private function rollinghash($string,$prima)
{
      if (strlen($string) == 1) {
            return ord($string);
      } else {
            $result = 0;
            $k = strlen($string);
            for ($i = 0; $i < $k; $i++) {
                  $result +=
            ord(substr($string, $i, 1)) *
            pow($prima, $k-$i);
                  }
                  return $result;
            }
      }
```

Gambar 3.16 *Source Code* Proses *Rolling Hash*

2 Algoritma *Manber*

Pada tahap ini, nilai hash yang telah didapatkan pada tahap preprocessing akan di proses lebih lanjut untuk menentukan nilai fingerprint. Nilai fingerprint ini digunakan sebagai parameter untuk menghitung presentase plagiarisme sebuah dokumen. Adapun *source code* proses Algoritma *Manber* dapat dilihat pada gambar 3.17.

```
private function fingerprints($rolling_hash, $P){
            $roll_hash = array();
            $length = count($rolling_hash);
            $chunk = array();
            for ($i=0; $i < $length ; $i++) {
                  if ($rolling_hash[$i]%$P == 0) {
                        $chunk[]=$rolling_hash[$i];
                  }
            }
            return $chunk;
      }
```

Gambar 3.17 *Source Code* Proses Algoritma *Manber*

3 *Jaccard Coefficient*

Persamaan *Jaccard Coefficient* digunakan untuk menghitung presentase tingkat plagiarisme sebuah dokumen. Adapun *source code* proses *Jaccard Coefficient* dapat dilihat pada gambar 3.18.

```
private function jaccard_coefficient($fingerprint1,
$fingerprint2){
            $arr_intersect = array_intersect(
$fingerprint1, $fingerprint2 );
            $arr_union = array_merge( $fingerprint1,
$fingerprint2 );

            $count_intersect_fingers =
count($arr_intersect);
            $count_union_fingers = count( $arr_union
);

            $coefficient = $count_intersect_fingers /
                  ($count_union_fingers -
$count_intersect_fingers );

            return $coefficient;
      }
```

Gambar 3.18 *Source Code* Proses *Jaccard Coefficient*

BAB IV
UJI COBA DAN PEMBAHASAN

Pada bab ini peneliti akan memaparkan hasil uji coba dan pembahasan aplikasi pendeteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword* yang telah di bangun oleh peneliti.

4.1 Skenario Uji Coba

Skenario uji coba ini dilakukan agar mengetahui apakah algoritma *Manber* dengan pendekatan *Biword* cocok digunakan untuk mendeteksi plagiarisme dan juga menguji kelayakan aplikasi ini untuk digunakan oleh *user*. Langkah – langkah yang dilakukan dalam skenario uji coba adalah sebagai berikut :

1. Mengunduh data dari etheses.uin-malang.ac.id. Data yang diunduh adalah tugas akhir/ skripsi mahasiswa jurusan Teknik Informatika UIN Maulana Malik Ibrahim Malang. Data yang diunduh berekstensi *pdf* yang kemudian akan dijadikan sebagai data uji tingkat plagiarisme.
2. Mengambil abstrak dari *file* tugas akhir/skripsi mahasiswa, kemudian file disimpan dalam ekstensi *.doc*. *File* Abstrak yang telah berekstensi *.doc* kemudian disimpan dalam *database*.
3. Menguji tingkat plagiarisme terhadap data *input* menggunakan aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*. Dalam pengujian ini dilakukan uji coba terhadap 30 data dengan 5 konfigurasi yang berbeda. Dalam pengujian ini, peneliti menggunakan data yang berbeda untuk setiap konfigurasi. Adapun konfigurasi nilai prima dan nilai *chunk(p)* yang digunakan adalah :

1) Konfigurasi I : Nilai prima 2 dan Nilai *chunk(p)* 4
2) Konfigurasi II : Nilai prima 3 dan Nilai *chunk(p)* 2
3) Konfigurasi III: Nilai prima 5 dan Nilai *chunk(p)* 4
4) Konfigurasi IV : Nilai prima 7 dan Nilai *chunk(p)* 2
5) Konfigurasi V : Nilai prima 11 dan Nilai *chunk(p)* 4

Hasil tingkat plagiarisme dari *output* uji coba ini akan dibandingkan dengan *output* tingkat plagiarisme aplikasi *plagiarism checker x* guna mendapatkan presentase nilai *error* dari aplikasi ini.

4. Menghitung presentase nilai *error* dari aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*. Pengujian ini dilakukan untuk mengetahui seberapa dekat *output* yang dihasilkan dari sistem dengan *output* nilai sebenarnya (*ground truth*). Presentase nilai *error* didapatkan dari hasil *output* data uji setiap konfigurasi dibandingkan dengan nilai *output* sebenarnya dari aplikasi *plagiarism checker x*. Pengujian ini dilakukan guna mengetahui konfigurasi mana yang paling menghasilkan presentase nilai *error* tertinggi. Untuk mendapatkan presentase nilai *error* digunakan rumus dibawah ini :

$$\textbf{\textit{Error}} = \frac{|Y_x - \hat{Y}x|}{Y_x} \times 100\% \qquad (4.1)$$

Keterangan :

Y_x = Nilai *Output* sebenarnya

$\hat{Y}x$ = Nilai *Output* sistem

5. Menghitung kecepatan aplikasi deteksi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*. Pengujian ini dilakukan guna mengetahui waktu yang dibutuhkan algoritma *Manber* dalam mendeteksi tingkat plagiarisme. Dalam melakukan pengujian rata-rata kecepatan algoritma *Manber*, digunakan menggunakan rumus *Mean* seperti dibawah ini :

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} xi \qquad (4.2)$$

Keterangan:

$\bar{X}$ = Rata-rata Kecepatan proses pengujian.

n = Banyaknya pengujian.

Xi = Kecepatan proses pengujian ke-i.

4.2 Hasil Uji Coba

Hasil uji coba yang akan dipaparkan adalah nilai presentase *error* dan nilai rata-rata kecepatan *output* tingkat plagiarisme algoritma *Manber* dengan pendekatan *Biword* dari masing - masing konfigurasi. Setiap konfigurasi akan diuji dengan 30 data *input* yang berbeda, kemudian data *input* akan dibandingkan dengan data yang ada di dalam *database*.

4.2.1 Konfigurasi I

Konfigurasi nilai *chunk(p)* dan nilai prima yang dilakukan pada pengujian ini adalah *chunk(p)* = 4, prima = 2. Pada pengujian ini dilakukan uji coba deteksi plagiarisme terhadap 30 data yang berbeda, kemudian *output* persentase plagiarisme yang dihasilkan oleh sistem akan dibandingkan dengan *output* nilai

sebenarnya (*ground truth*). Nilai *ground truth* merujuk dari *output* yang dihasilkan dari aplikasi *plagiarism checker x*. Hasil pengujian dari 30 data pada konfigurasi I dapat dilihat pada tabel 4.1.

Tabel 4.1 Hasil Uji Coba Konfigurasi I

Konfigurasi I		Data ke-	Hasil *Manber*	Kecepatan (dtk)	*Plagiarsm checker x*	*Error*
Nilai *chunk* (P)	Nilai Prima					
4	2	1	100%	2,27	100%	0%
		2	100%	2,63	100%	0%
		3	100%	2,39	100%	0%
		4	100%	2,31	95%	5%
		5	98%	1,96	76%	29%
		6	100%	2,04	100%	0%
		7	97%	2,09	94%	3%
		8	100%	2,11	100%	0%
		9	100%	2,24	100%	0%
		10	100%	2,2	100%	0%
		11	100%	1,81	100%	0%
		12	100%	1,95	100%	0%
		13	94%	1,83	54%	74%
		14	100%	2,21	100%	0%
		15	100%	2,25	100%	0%
		16	98%	1,89	90%	9%
		17	100%	2,07	100%	0%
		18	100%	1,97	100%	0%
		19	99%	2,17	87%	14%
		20	100%	2	100%	0%
		21	100%	2,07	100%	0%
		22	100%	2,04	100%	0%
		23	100%	2,29	100%	0%
		24	98%	1,97	97%	1%
		25	100%	1,75	100%	0%
		26	100%	2,06	100%	0%
		27	100%	1,75	100%	0%
		28	100%	2,06	100%	0%
		29	100%	2,15	100%	0%
		30	100%	1,49	100%	0%
Rata-rata				2,06		4,5%

Berdasarkan Tabel 4.1 terdapat 23 *output* yang nilai nya sama dengan nilai sebenarnya (*ground truth*). Maka nilai rata-rata presentase *error* pada konfigurasi *chunk* (p) = 4 dan prima = 2 dapat dihitung sebagai berikut :

$$Error = \frac{(\text{jumlah semua presentase nilai } error)}{(\text{banyaknya pengujian})} \; x \; 100$$

$$Error = \frac{135}{30} \; x \; 100 = 4{,}5\%$$

Rata-rata kecepatan pemrosesan algoritma *Manber* dalam mendeteksi tingkat plagiarisme dapat dihitung sebagai berikut :

$$\text{Kecepatan} = \frac{(\text{jumlah semua kecepatan})}{(\text{banyaknya pengujian})}$$

$$\text{Kecepatan} = \frac{57{,}67}{30} = 2{,}06 \text{ dtk}$$

Berdasarkan perhitungan di atas, konfigurasi I dengan nilai *chunk* (p) = 4 dan prima = 2, didapatkan nilai rata-rata presentase *error* sebesar 4,5% dan rata-rata kecepatan 2,06 detik.

Pada aplikasi ini hasil perhitungan dari algoritma yaitu berupa presentase tingkat plagiarisme dan *marking* kata yang terdeteksi plagiarisme. Gambar 4.1 merupakan *output marking* konfigurasi I dari aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*.

Gambar 4.1 *Output Marking* Konfigurasi I

4.2.2 Konfigurasi II

Konfigurasi nilai *chunk(p)* dan nilai prima yang dilakukan pada pengujian ini adalah *chunk(p)* = 2, prima = 3. Pada pengujian ini dilakukan uji coba deteksi plagiarisme terhadap 30 data yang berbeda, kemudian *output* persentase plagiarisme yang dihasilkan oleh sistem akan dibandingkan dengan *output* nilai sebenarnya (*ground truth*). Nilai *ground truth* merujuk dari *output* yang dihasilkan dari aplikasi *plagiarism checker x*. Hasil pengujian dari 30 data pada konfigurasi II dapat dilihat pada tabel 4.2.

Tabel 4.2 Hasil Uji Coba Konfigurasi II

Konfigurasi II		Data ke-	Hasil *Manber*	Kecepatan (dtk)	Plagiarsm checker x	*Error*
Nilai *chunk*(P)	Nilai Prima					
2	3	1	100%	3,04	95%	5%
		2	98%	3,11	100%	2%
		3	99%	3,33	100%	1%
		4	99%	3,28	95%	4%
		5	98%	2,59	76%	29%
		6	100%	2,81	100%	0%
		7	98%	3,03	94%	4%
		8	100%	2,94	100%	0%
		9	100%	3,35	100%	0%
		10	100%	2,75	100%	0%
		11	100%	2,75	100%	0%
		12	100%	3,05	100%	0%
		13	95%	2,6	54%	76%
		14	100%	2,87	100%	0%
		15	100%	3,21	100%	0%
		16	98%	2,5	90%	9%
		17	99%	2,96	100%	1%
		18	100%	2,90	100%	0%
		19	99%	3	87%	14%
		20	100%	2,51	100%	0%
		21	100%	3,3	100%	0%
		22	98%	3,35	100%	2%
		23	100%	2,58	100%	0%
		24	98%	2,81	97%	1%
		25	100%	2,47	100%	0%
		26	100%	2,91	100%	0%
		27	100%	2,66	100%	0%
		28	100%	2,55	100%	0%
		29	100%	3,99	100%	0%
		30	100%	2,04	100%	0%
Rata-rata				2,91		4,9%

Berdasarkan Tabel 4.2 terdapat 19 *output* yang nilai nya sama dengan nilai sebenarnya(*ground truth*). Maka nilai rata-rata presentase *error* pada konfigurasi *chunk* (p) = 2 dan prima = 3 dapat dihitung sebagai berikut :

$$Error = \frac{(\text{jumlah semua presentase nilai } error)}{(\text{banyaknya pengujian})} \; x \; 100$$

$$Error = \frac{148}{30} \; x \; 100 = 4{,}9\%$$

Rata- rata kecepatan pemrosesan algoritma *Manber* dalam mendeteksi tingkat plagiarisme dapat dihitung sebagai berikut :

$$\text{Kecepatan} = \frac{(\text{jumlah semua kecepatan})}{(\text{banyaknya pengujian})}$$

$$\text{Kecepatan} = \frac{75{,}55}{30} = 2{,}91 \text{ dtk}$$

Berdasarkan perhitungan di atas, konfigurasi II dengan nilai *chunk* (p) = 2 dan prima = 3, didapatkan nilai rata-rata presentase *error* sebesar 4,9% dan rata-rata kecepatan 2,91 detik.

Pada aplikasi ini hasil perhitungan dari algoritma yaitu berupa presentase tingkat plagiarisme dan *marking* kata yang terdeteksi plagiarisme. Gambar 4.2 merupakan *output marking* konfigurasi II dari aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*.

Gambar 4.2 *Output Marking* Konfigurasi II

4.2.3 Konfigurasi III

Konfigurasi nilai *chunk(p)* dan nilai prima yang dilakukan pada pengujian ini adalah *chunk(p)* = 4, prima = 5. Pada pengujian ini dilakukan uji coba deteksi plagiarisme terhadap 30 data yang berbeda, kemudian *output* persentase plagiarisme yang dihasilkan oleh sistem akan dibandingkan dengan *output* nilai sebenarnya (*ground truth*). Nilai *ground truth* merujuk dari *output* yang dihasilkan dari aplikasi *plagiarism checker x*. Hasil pengujian dari 30 data pada konfigurasi III dapat dilihat pada tabel 4.3.

Tabel 4.3 Hasil Uji Coba Konfigurasi III

Konfigurasi III		Data ke-	Hasil *Manber*	Kecepatan (dtk)	Plagiarsm check x	*Error*
Nilai *chunk* (P)	Nilai Prima					
4	5	1	100%	2,77	100%	0%
		2	98%	3,02	100%	2%
		3	99%	3,15	100%	1%
		4	99%	3,73	95%	4%
		5	98%	2,48	76%	29%
		6	100%	3,01	100%	0%
		7	98%	2,94	94%	4%
		8	100%	2,87	100%	0%
		9	100%	3,25	100%	0%
		10	100%	2,64	100%	0%
		11	100%	2,72	100%	0%
		12	100%	2,85	100%	0%
		13	95%	2,71	54%	76%
		14	100%	2,77	100%	0%
		15	100%	3,31	100%	0%
		16	98%	2,60	90%	9%
		17	99%	3,31	100%	1%
		18	100%	2,98	100%	0%
		19	99%	3,07	87%	14%
		20	100%	2,49	100%	0%
		21	100%	3,41	100%	0%
		22	98%	2,88	100%	2%
		23	100%	2,56	100%	0%
		24	98%	2,75	97%	1%
		25	100%	2,56	100%	0%
		26	100%	2,94	100%	0%
		27	100%	2,91	100%	0%
		28	100%	2,61	100%	0%
		29	100%	3,66	100%	0%
		30	100%	2,31	100%	0%
Rata-rata				2,96		4,7

Berdasarkan Tabel 4.3 terdapat 19 *output* yang nilai nya sama dengan nilai sebenarnya(*ground truth*). Maka nilai rata-rata presentase *error* pada konfigurasi *chunk* (p) = 4 dan prima = 5 dapat dihitung sebagai berikut :

$$Error = \frac{(\text{jumlah semua presentase nilai } error)}{(\text{banyaknya pengujian})} \; x \; 100$$

$$Error = \frac{143}{30} \; x \; 100 = 4{,}7\%$$

Kecepatan algoritma *Manber* dalam mendeteksi tingkat plagiarisme dapat dihitung sebagai berikut :

$$\text{Kecepatan} = \frac{(\text{jumlah semua kecepatan})}{(\text{banyaknya pengujian})}$$

$$\text{Kecepatan} = \frac{68{,}13}{30} = 2{,}96 \text{ dtk}$$

Berdasarkan perhitungan di atas, konfigurasi III dengan nilai *chunk* (p) = 4 dan prima = 5, didapatkan nilai rata-rata presentase *error* sebesar 4,7% dan rata-rata kecepatan 2,96 detik.

Pada aplikasi ini hasil perhitungan dari algoritma yaitu berupa presentase tingkat plagiarisme dan *marking* kata yang terdeteksi plagiarisme. Gambar 4.3 merupakan *output marking* konfigurasi III dari aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*.

Gambar 4.3 *Output Marking* Konfigurasi III

4.2.4 Konfigurasi IV

Konfigurasi nilai *chunk(p)* dan nilai prima yang dilakukan pada pengujian ini adalah *chunk(p)* = 2, prima = 7. Pada pengujian ini dilakukan uji coba deteksi plagiarisme terhadap 30 data yang berbeda, kemudian *output* persentase plagiarisme yang dihasilkan oleh sistem akan dibandingkan dengan *output* nilai sebenarnya (*ground truth*). Nilai *ground truth* merujuk dari *output* yang dihasilkan dari aplikasi *plagiarism checker x*. Hasil pengujian dari 30 data pada konfigurasi IV dapat dilihat pada tabel 4.4.

Tabel 4.4 Hasil Uji Coba Konfigurasi IV

Konfigurasi IV		Data ke-	Hasil *Manber*	Kecepatan (dtk)	*Plagiarsm checker x*	*Error*
Nilai *chunk* (P)	Nilai Prima					
2	7	1	100%	2,98	98%	2%
		2	98%	3,01	100%	2%
		3	99%	3,43	100%	1%
		4	99%	3,48	95%	4%
		5	98%	2,51	76%	29%
		6	100%	2,8	100%	0%
		7	98%	2,65	94%	4%
		8	100%	3,07	100%	0%
		9	100%	3,49	100%	0%
		10	100%	2,67	100%	0%
		11	100%	2,72	100%	0%
		12	100%	2,96	100%	0%
		13	95%	2,69	54%	76%
		14	100%	2,82	100%	0%
		15	100%	3,27	100%	0%
		16	98%	2,66	90%	9%
		17	99%	3,17	100%	1%
		18	100%	3,07	100%	0%
		19	99%	3,17	87%	14%
		20	100%	2,43	100%	0%
		21	100%	3,47	100%	0%
		22	98%	3,05	100%	2%
		23	100%	2,97	100%	0%
		24	98%	2,86	97%	1%
		25	100%	2,6	100%	0%
		26	100%	3,08	100%	0%
		27	100%	2,73	100%	0%
		28	100%	2,77	100%	0%
		29	100%	3,75	100%	0%
		30	100%	2,18	100%	0%
Rata-rata				2,91		4,8%

Berdasarkan Tabel 4.4 terdapat 19 *output* yang nilai nya sama dengan nilai sebenarnya(*ground truth*). Maka nilai rata-rata presentase *error* pada konfigurasi *chunk* (p) = 2 dan prima = 7 dapat dihitung sebagai berikut :

$$Error = \frac{(\text{jumlah semua presentase nilai } error)}{(\text{banyaknya pengujian})} \; x \; 100$$

$$Error = \frac{145}{30} \; x \; 100 = 4{,}8\%$$

Rata-rata kecepatan pemrosesan algoritma *Manber* dalam mendeteksi tingkat plagiarisme dapat dihitung sebagai berikut :

$$\text{Kecepatan} = \frac{(\text{jumlah semua kecepatan})}{(\text{banyaknya pengujian})}$$

$$\text{Kecepatan} = \frac{61{,}29}{30} = 2{,}91 \text{ dtk}$$

Berdasarkan perhitungan di atas, konfigurasi IV dengan nilai *chunk* (p) = 2 dan prima = 7, didapatkan nilai rata-rata presentase *error* sebesar 4,8% dan rata-rata kecepatan 2,91 detik.

Pada aplikasi ini hasil perhitungan dari algoritma yaitu berupa presentase tingkat plagiarisme dan *marking* kata yang terdeteksi plagiarisme. Gambar 4.4 merupakan *output marking* konfigurasi IV dari aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*.

Gambar 4.4 *Output Marking* Konfigurasi IV

4.3.5 Konfigurasi V

Konfigurasi nilai *chunk(p)* dan nilai prima yang dilakukan pada pengujian ini adalah *chunk(p)* = 4, prima = 11. Pada pengujian ini dilakukan uji coba deteksi plagiarisme terhadap 30 data yang berbeda, kemudian *output* persentase plagiarisme yang dihasilkan oleh sistem akan dibandingkan dengan *output* nilai sebenarnya (*ground truth*). Nilai *ground truth* merujuk dari *output* yang dihasilkan dari aplikasi *plagiarism checker x*. Hasil pengujian dari 30 data pada konfigurasi V dapat dilihat pada tabel 4.5.

Tabel 4.5 Hasil Uji Coba Konfigurasi V

Konfigurasi V		Data ke-	Hasil *Manber*	Kecepatan (dtk)	*Plagiarsm checker x*	*Error*
Nilai *chunk* (P)	Nilai Prima					
4	11	1	100%	3,02	74%	35%
		2	98%	3,10	100%	2%
		3	99%	3,56	100%	1%
		4	99%	3,29	95%	4%
		5	98%	2,51	76%	29%
		6	100%	2,83	100%	0%
		7	98%	2,91	94%	4%
		8	100%	2,94	100%	0%
		9	100%	3,53	100%	0%
		10	100%	2,55	100%	0%
		11	100%	2,87	100%	0%
		12	100%	3,01	100%	0%
		13	95%	2,77	54%	76%
		14	100%	2,87	100%	0%
		15	100%	3,32	100%	0%
		16	98%	2,63	90%	9%
		17	99%	3,03	100%	1%
		18	100%	3,03	100%	0%
		19	99%	3,11	87%	14%
		20	100%	2,56	100%	0%
		21	100%	3,38	100%	0%
		22	98%	3,04	100%	2%
		23	100%	2,92	100%	0%
		24	98%	3,07	97%	1%
		25	100%	2,54	100%	0%
		26	100%	3,18	100%	0%
		27	100%	2,84	100%	0%
		28	100%	2,78	100%	0%
		29	100%	3,71	100%	0%
		30	100%	2,43	100%	0%
Rata-rata				2,94		5,9%

Berdasarkan Tabel 4.5 terdapat 19 *output* yang nilai nya sama dengan nilai sebenarnya(*ground truth*). Maka nilai rata-rata presentase *error* pada konfigurasi *chunk* (p) = 4 dan prima = 11 dapat dihitung sebagai berikut :

$$Error = \frac{(\text{jumlah semua presentase nilai } error)}{(\text{banyaknya pengujian})} \; x \; 100$$

$$Error = \frac{178}{30} \; x \; 100 = 5{,}9\%$$

Rata-rata kecepatan pemrosesan algoritma *Manber* dalam mendeteksi tingkat plagiarisme dapat dihitung sebagai berikut :

$$\text{Kecepatan} = \frac{(\text{jumlah semua kecepatan})}{(\text{banyaknya pengujian})}$$

$$\text{Kecepatan} = \frac{67{,}63}{30} = 2{,}94 \text{ dtk}$$

Berdasarkan perhitungan di atas konfigurasi V dengan nilai *chunk* (p) = 4 dan prima = 11, didapatkan nilai rata-rata presentase *error* sebesar 5,9% dan rata-rata kecepatan 2,94 detik.

Pada aplikasi ini hasil perhitungan dari algoritma yaitu berupa presentase tingkat plagiarisme dan *marking* kata yang terdeteksi plagiarisme. Gambar 4.5 merupakan *output marking* konfigurasi V dari aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*.

Gambar 4.5 *Output Marking* Konfigurasi V

4.3 Pembahasan

Berdasarkan hasil uji coba yang telah dilakukan oleh peneliti terhadap aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*, didapatkan hasil rata-rata presentase *error* dan rata-rata kecepatan pemrosesan seperti pada tabel 4.6.

Tabel 4.6 Hasil Presentase *Error* Dan Kecepatan Proses

No.	Konfigurasi		*Error*	kecepatan proses
	chunk(P)	Prima		
I	4	2	4,5%	2,06 detik
II	2	3	4,9%	2,91 detik
III	4	5	4,7%	2,96 detik
IV	2	7	4,8%	2,91 detik
V	4	11	5,9%	2,94 detik

Setelah melakukan rangkaian uji coba terhadap Aplikasi Deteksi Plagiarisme Menggunakan Algoritma *Manber* Dengan Pendekatan *Biword* penulis akan membahas hasil percobaan pada tiap konfigurasi. Berikut adalah hasil pembahasan dari tiap konfigurasi:

1. **Konfigurasi I: Prima 2 dan *Chunk(P)* 4**

   Konfigurasi I yang memiliki nilai prima dan *Chunk(P)* terendah dibanding konfigurasi lain. Konfigurasi I mampu menghasilkan *output* dengan persentase *error* sebesar 4,5% dengan rata-rata kecepatan 2,06 detik.

2. **Konfigurasi II: Prima 3 dan *Chunk(P)* 2**

   Konfigurasi II dengan nilai prima dan *Chunk(P)* yg lebih besar dari konfigurasi I, mampu menghasilkan *output* dengan persentase *error* sebesar 4,9% dengan rata-rata kecepatan 2,91 detik.

3. **Konfigurasi III: Prima 5 dan *Chunk(P)* 4**

   Konfigurasi III dengan nilai prima lebih besar dari konfigurasi II, mampu menghasilkan *output* dengan persentase *error* sebesar 4,7% dengan rata-rata kecepatan 2,96 detik.

4. **Konfigurasi IV: Prima 7 dan *Chunk(P)* 2**

   Konfigurasi IV dengan nilai prima dan *Chunk(P)* yg lebih besar dari konfigurasi III, mampu menghasilkan *output* dengan persentase *error* sebesar 4,8% dengan rata-rata kecepatan 2,91detik.

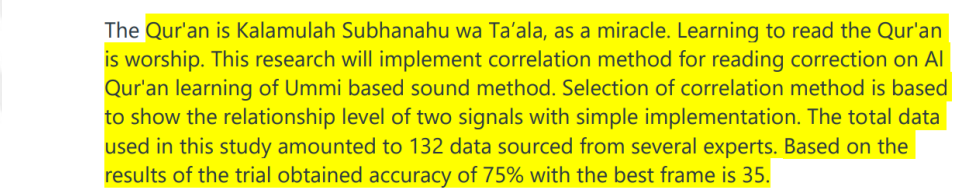
5. **Konfigurasi V: Prima 11 dan *Chunk(P)* 4**

   Konfigurasi V dengan nilai prima dan *Chunk(P)* yg lebih besar dari konfigurasi IV, mampu menghasilkan *output* dengan persentase *error* sebesar 5,9% dengan rata-rata kecepatan 2,94detik.

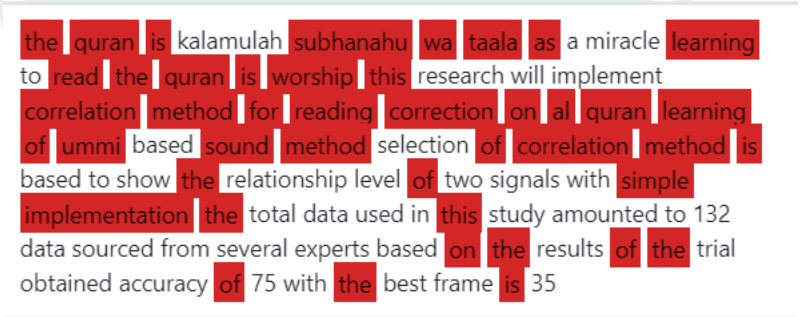
Beradasrkan hasil uji coba yang telah dipaparkan diatas, dapat disimpulkan bahwa kofigurasi I dengan nilai chunk(P) = 4 dan prima = 2 mengasilkan nilai rata-rata presentase *error* terkecil yaitu 4,5%. Konfigurasi I juga memiliki rata-rata kecepatan proses yang paling cepat yaitu 2,06 detik.

Nilai error terjadi karena konfigurasi nilai chunk dan nilai prima yang kurang cocok. Nilai *chunk* dan nilai prima merupakan 2 parameter yang mempengaruhi pemilihan nilai *fingerprint* dalam algoritma *Manber*, sehingga menghasilkan output yang berbeda dari nilai *ground truth* (nilai kebenaran).

Pada sistem ini, proses *marking* masih kurang maksimal dalammencari kata yang sama. Aplikasi ini hanya memotong teks menjadi 2 kata kemudian mencari sumber plagiarisme pada database. Aplikasi plagiarism Checker X mampu memotong teks menjadi beberapa kalimat kemudian mencari sumber plagiarisme dari internet. Gambar 4.6 dan Gambar 4.7 merupakan perbandingan dari *output* kedua aplikasi tersebut.



Gambar 4.6 *Marking Output* Aplikasi *Plagiarism Checker X*



Gambar 4.7 *Output Marking* Aplikasi Deteksi Plagiarisme menggunakan Algoritma *Manber* Dengan Pendekatan *Biword*

Kurang nya database dalam sistem ini juga salah satu faktor yang menyebabkan munculnya nilai error pada sistem ini. Database dalam sistem ini hanya terbatas pada e-theses uin Malang, sehingga dalam sistem ini hanya bisa membandingkan kesamaan teks dengan data pada e-theses uin Malang. Sedangkan aplikasi plagiarism checker dapat membandingkan kesamaan teks dari berbagai sumber di internet.

Dengan nilai error sebesar 4,5% dapat disimpulakan bahwa aplikasi ini belum bisa mencari teks yang mengandung plagiarisme dengan sempurna, namun aplikasi dapat membantu meminimalisir tindak plagiarime, khususnya di kalangan mahasiswa. Tindak plagiarisme merupakan tindakan seseorang yang menjiplak karya tulis, gagasan ide, atau karya orang lain yang bersifat sebagian atau keseluruhan dan mengklaim sebagai karya nya sendiri. Agama islam telah mengatur semua kehidupan manusia, semua tertuang dalam al-quran dan hadist yang menjadi pegangan hidup penganutnya. Islam menyeru kita untuk selalu berbuat baik kepada orang lain dan selalu bersikap jujur. Tindak plagiarisme ini tentu bertentangan dengan ajaran agama islam, plagiarisme merupakan prilaku yang tidak jujur karena mengakui karya orang lain sebagai miliknya sendiri. Sebagaimana firman Allah s.w.t dalam surat al-maidah ayat 119:

قَالَ ٱللَّهُ هَٰذَا يَوۡمُ يَنفَعُ ٱلصَّٰدِقِينَ صِدۡقُهُمۡ لَهُمۡ جَنَّٰتٞ تَجۡرِي مِن تَحۡتِهَا ٱلۡأَنۡهَٰرُ خَٰلِدِينَ فِيهَآ أَبَدٗا رَّضِيَ ٱللَّهُ عَنۡهُمۡ وَرَضُواْ عَنۡهُۚ ذَٰلِكَ ٱلۡفَوۡزُ ٱلۡعَظِيمُ

Artinya : “Inilah saat orang yang jujur memperoleh manfaat dari kejujurannya. Mereka memperoleh surga yang dibawahnya mengalir sungai-sungai, mereka kekal di dalamnya selama-lamanya. Allah ridha kepada mereka dan mereka pun ridha kepada-Nya. Itulah kemenangan yang agung”

Berdasarkan firman Allah diatas, telah jelas bahwa Allah telah memerintahkan hambanya untuk selalu berprilaku jujur dalam hal apapun. Bahkan ayat diatas juga menyebutkan ganjaran yang besar bagi orang yang jujur. Allah pun ridha terhadap orang-orang yang jujur karena prilaku jujurnya merupakan bentuk ketaatan kepada Rabb-Nya. Begitupula sebaliknya, orang yang selama di dunia sering berprilaku dusta atau bohong maka, kebohongan nya itu tidak akan membawa manfaat kepadanya di akhirat kelak dan akan mendapatkan siksa yang pedih.

Maka dari itu, aplikasi pendeteksi plagiarisme ini hadir dengan harapan dapat meminimalisasi tindak plagiarisme di kalangan mahasiswa. Agar mahasiswa selalu bersikap jujur dalam mengerjakan tugas, sehingga ilmu yang didapatkan menjadi barokah dan bermanfaat bagi masa depannya kelak.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah dilakukan uji coba terhadap aplikasi deteksi plagiarisme menggunakan algoritma *Manber* dengan pendekatan *Biword*. Peneliti mendapatkan kesimpulan sebagai berikut :

1. Konfigurasi yang berbeda dapat menghasilkan *output* yang berbeda dalam deteksi tingkat plagiarisme.
2. Rata-rata nilai *error* konfigurasi I = 4.5%, konfigurasi II = 4.9%, konfigurasi III = 4.7%, konfigurasi IV = 4.8%, konfigurasi V = 5.9%. Berdasarkan hasil pengujian tersebut disimpulkan bahwa konfigurasi I menghasilkan nilai error terkecil, yaitu 4,5%, dengan nilai P*(chunk)* = 4 dan nilai prima=2. Hal tersebut membuktikan bahwa konfigurasi tersebut merupakan konfigurasi terbaik dibandingkan konfigurasi lainnya.
3. Rata-rata kecepatan konfigurasi I = 2.06 detik, konfigurasi II = 2.91 detik, konfigurasi III = 2.96 detik, konfigurasi IV = 2.91 detik, konfigurasi V = 2.94 detik. Berdasarkan hasil pengujian tersebut disimpulkan bahwa konfigurasi I menghasilkan rata-rata waktu proses tercepat dalam mendeteksi plagiarisme yaitu 2.06 detik, dengan konfigurasi nilai P*(chunk)* = 4 dan nilai prima = 2. Hal tersebut membuktikan bahwa semakin kecil nilai prima maka akan semakin cepat sistem dalam mendeteksi plagiarisme.

5.2 Saran

Peneliti menyadari bahwa aplikasi ini masih memiliki kekurangan dan masih harus dilakukan penelitian lebih lanjut, guna meningkatkan hasil yang lebih baik. Adapun saran untuk penelitian selanjutnya adalah :

1. Perlunya *database* yang lebih banyak, sehingga dapat membandingkan dengan berbagai sumber yang lebih bervariatif dan hasil yang didapatkan menjadi lebih akurat.
2. Perlunya uji coba dengan beberapa konfigurasi lain, sehingga menghasilkan *output* yang paling akurat.
3. *Input* dalam aplikasi ini hanya dapat mengolah data berekstensi *.doc*. Pada penelitian selanjutnya, *input* aplikasi ini diharapkan dapat mengolah data dari berbagai macam ekstensi file.

DAFTAR PUSTAKA

Alzahrani, S. M., Salim, N., Abraham, A. (2012). "Understanding Plagiarism linguistic patterns, textual features, and detection methods." IEEE Transaction on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 42, no. 2, pp. 133-149.

Ceska, Z. (2008). "Plagiarism Detection Based on Singular Value Decomposition." 5th International Conference on Advances in Natural Language Processing. Berlin, Heidelberg, New York: SpringerVerlag.

Elbegbayan, N. (2005). "Winnowing, a Document Fingerprinting Algorithm." Department of Computer Science, Linkoping University.

Hermawan, B. I. (2015). "Analisis Performansi Algoritma Winnowing Dan algoritma *Manber* Untuk Deteksi Kesamaan Dokumen Teks Berbahasa Indonesia."

Hoad, T., Zobel, J. (2007). "Methods for Identifying Versioned and Plagiarised Documents." In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, pp. 825–826. ISBN 978-1-59593-597-7.

KBBI (Kamus Besar Bahasa Indonesia) Edisi Online. https://kbbi.web.id/. (diunduh pada tanggal 5 November 2019).

Kent, C. K., Salim, N. (2010). "*Features Based Text Similarity Detection*", Malaysia: Faculty of Computer Science and Information System.

Putri, B. B., Ernawati, Puspitaningrum, D. (2015). "Implementasi Metode Wu-*Manber* Berdasarkan *Multi Pattern Matching* Dalam Pencarian Kesamaan DNA (Studi kasus : DNA Kanker Hati)." Jurnal Rekursif, Vol. 3 No.2 November 2015, ISSN 2303-0755.

Rafieian, S., Dastjerdi, A. B. (2015). "Plagiarism checker for Persian (PCP) texts using hash-based tree representative fingerprinting." Journal of AI and Data Mining Vol 4, No 2, 2016, 125-133.

Ridho, M. (2013). "Rancang Bangun Aplikasi Pendeteksi Penjiplakan Dokumen Menggunakan Algoritma Biword Winnowing."

S. Kosinov. 2001. "Evaluation of n-grams conflation approach in text-based information retrieval," in 8th String Processing and Information Retrieval Symposium (SPIRE 2001), Canada, Computing Science Department, 2001, pp. 136--142.

Salmuasih, Sunyoto, A. (2013). "Implementasi Algoritma Rabin Karp untuk Pendeteksian Plagiat Dokumen Teks Menggunakan Konsep *Similarity*." Seminar Nasional Aplikasi Teknologi Informasi (SNATI) 2013. Yogyakarta: STMIK AMIKOM. 15 Juni 2013.

Schleimer, S., Wilkerson, D. S., Aiken, A. (2003). "Winnowing: Local Algorithms for Document Fingerprinting." The 2003 ACM SIGMOD International Conference Management of Data, New York, 2003.

Sukmana, A., Kusrini, Sunyoto, A. (2018). "Perbandingan Pengunaan Stemming Pada Deteksi Kemiripan Dokumen Menggunakan Metode Rabin Karp Dan Jaccard *Similarity*." Seminar Nasional Teknologi Informasi dan Multimedia 2018.

Suarasurabaya, 2014. "Plagiarisme Adalah Kejahatan Akademik Level Tertinggi". Surabaya : Suarasurabaya. https://www.suarasurabaya.net/kelanakota/2014/Plagiarisme-AdalahKejahatan-Akademik-Level-Tertinggi/(diunduh pada tanggal 15 Oktober 2019).