Jurnal Ilmiah Komputer dan Informatika (KOMPUTA)
Edisi. .. Volume. .., Bulan 20.. ISSN : 2089-9033

IMPLEMENTASI ALGORITMA *LESK* UNTUK *SYNONIM RECOGNITION* DAN *RABIN KARP* PADA PENDETEKSIAN *PLAGIARISME*

Moch Nurhalimi ZD[1], Ednawati Rainarli[2]

[1,2] Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-114 Bandung
E-mail : nurhalimm@yahoo.co.id[1], irene_edna@yahoo.com[2]

ABSTRAK

Penjiplakan dokumen digital bukanlah hal yang susah, cukup dengan menggunakan teknik *copy-paste-modify*. Salah satu metoda yang digunakan untuk mendeteksi plagiarisme adalah *string macthing*. Salah satu algoritma string matching yang sering digunakan adalah rabin karp menurut salah satu jurnal algoritma rabin karp menghasilkan efesiensi waktu yang lebih baik dalam mendeteksi string yang memiliki pola banyak. Terkadang plagiarisme juga dilakukan dengan mengganti kata-kata yang mengandung kata - kata sinonim, dengan maksud agar terlihat berbeda dari dokumen aslinya. Sehingga setelah proses *preprocessing* dilakukan proses pengenalan sinonim dan pemilihan kata sinonim (*synonim recognition*). Diperlukan suatu proses untuk mendeteksi kesamaan dokumen yaitu proses *preprocessing*, *synonym recognition*, parsing *k-gram*, konversi *hashing*, pencocokan *string*, hitung *similiarity*.

Berdasarkan pengujian dapat disimpulkan bahwa kombinasi algoritma lesk untuk proses synonym recognition dan rabin rabin karp dalam penerapan pada kasus plagiarism (kesamaan) menghasilkan presentase kesamaan rata – rata yaitu sebesar 85,78%, dibandingkan yang tidak menggunakan proses synonim recognition yaitu menghasilkan rata – rata sebesar 77.45%, meskipun membutuhkan waktu proses yang lebih banyak daripada tidak menggunakan *synonym recognition*

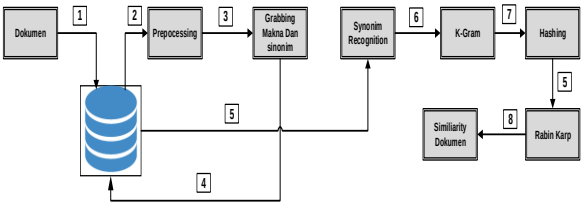
Kata Kunci : *string matching*, Plagirisme, *Synonim recognition* , algoritma *rabin karp*.

1. PENDAHULUAN

Penjiplakan atau plagiarism berarti mencontoh atau meniru tulisan dan karya orang lain yang kemudian diakui sebagai karanganya sendiri dengan ataupun tanpa seizin penulisnya. Penjiplakan dokumen digital bukanlah hal yang susah, cukup dengan menggunakan teknik *copy-paste-modify* pada sebagian isi dokumen dan bahkan keseluruhan isi dokumen sudah bisa dikatakan bahwa dokumen tersebut merupakan hasil duplikasi dari dokumen lain [1]. Terkadang plagiarisme juga dilakukan dengan mengganti kata-kata yang mengandung kata - kata sinonim, dengan maksud agar terlihat berbeda dari dokumen aslinya. Dimana hanya merubah kata – kata yang mengandung sinonim yang terdapat di dokumen asli dengan merubah nya sesuai dengan kata yang ada di Kamus bahasa indonesia , tanpa memperhatikan struktur kalimat dari dokumen aslinya[1].

Pada penelitian ini terdapat beberapan tahapan dalam proses mendeteksi kesamaan dokumen dari 2 (dua) buah dokumen. Diantaranya proses tersebut adalah pertama tahap pertama adalah tahap pemilihan dokumen asli dan dokumen uji yang akan di hitung nilai kesamaanya, setelah itu melalui tahap *prepcoessing* dimana hasil dari tahap ini akan digunakan untuk proses pemilihan dan penentuan kata mana yang akan dipilih buat melengkapi tindak plagiarism pada perubahan kata sinonim menggunakan algoritma *lesk*. Kemudian dilakukan proses parsing *k-gram* dari kedua array kata (dokumen asli dan dokumen uji), dimana buat penentuan nilai K nya bersifat experimental, namun jika semakin kecil nilai K maka nilai *similiairity* akan semakin besar, dan semakin besar nilai K maka nilai *similiairy* akan semakin kecil [2]. Kemudian hasil dari parsing *k-gram* tersebut akan dikonversi kedalam nilai *hashing* dimana nilai tersebut yang nantinya akan di cocokan menggunakan algoritma *rabin karp*. Dan kemudian dilakukan proses perhitungan nilai similiairity dari kesamaan dokumen uji terhadap dokumen asli menggunakan metoda *dice similiairity*.

Adapun tahap pendeteksiap plagiarism dari dokumen asli dan dokumen uji dapat dilihat pada gambar 1 berikut:



Gambar 1. Tahap Pendeteksian Plagiarisme

1.1 Preprocessing

Dokumen yang akan diproses, terlebih dahulu dilakukan tahap pertama yaitu proses *preprocessing*. Tahap *preprocessing* Adapun tahap-tahap dari proses *preprocessing* yang akan dilakukan diantaranya pemisahan kalimat [3], *case folding* [3], *filtering* [3], *tokenizing* [4], *stopword removal* [5] dan *stemming* [6]. Proses *stemming* dilakukan dengan menggunakan algoritma *porter stemmer* bahasa Indonesia [6]. Hasil dari tahapan proses *preprocessing* ini akan digunakan untuk melakuka proses ini akan digunakan untuk melakukann proses *synonym recognition* dengan menggunakan algoritma *lesk.*

1.2 Synonim Recognition

Synonym recognition atau pengenalan sinonim adalah salah satu pendekatan yang digunakan untuk membantu proses pendeteksian plagiarisme. Synonym recognition merupakan pendekatan semantik terhadap dokumen teks. Pendekatan ini memanfaatkan kesamaan makna dalam kata yang kemungkinan banyak terjadi [1]. Algoritma yang digunakan pada proses *synonym recognition* adalah algoritma *lesk.*

Algoritma lesk adalah algoritma yang digunakan untuk menghilangkan ambiguitas makna kata. Algoritma lesk merupakan salah satu algoritma untuk menyelesaikan masalah word sense disambiguation dengan berbasis kamus. Algoritma ini bekerja dengan membandingkan definisi dari kata yang berambigu dengan definisI.definisi dari kata tetangganya berdasarkan definisi kamus. Algoritma lesk menggunakan wordnet sebagai kamus atau acuan

1.3 Parsing *K-Gram*

K-Gram adalah rangkaian terms dengan panjang K. Kebanyakan yang digunakan sebagai terms adalah kata. K-Gram merupakan sebuah metode yang diaplikasikan untuk pembangkitan kata atau karakter. Metode K-Gram ini digunakan untuk mengambil potongan-potongan karakter huruf sejumlah k dari sebuah kata yang secara kontinuitas dibaca dari teks sumber hingga akhir dari dokumen [2].

1.4 *Hashing*

Hashing adalah suatu cara untuk mentransformasi sebuah *string* menjadi suatu nilai yang unik dengan panjang tertentu (*fixed-length*) yang berfungsi sebagai penanda *string* tersebut. *Hash function* atau fungsi *hash* adalah suatu cara menciptakan *fingerprint* dari berbagai data masukan. *Hash function* akan mengganti atau mentranspose-kan data tersebut untuk menciptakan fingerprint, yang biasa disebut hash value [7]. Untuk melakukan pengubahan tersebut menggunakan persamaan (1) berikut :

$$hash = C1 * a^{(k-1)} + C2 * a^{(k-2)} + C3 * a^{(k-3)} + Ck * a0$$

Dan persamaan (2) Berikut

$$rem = hash \,/\, Nilai\ mod.$$

Dimana ci adalah nilai ascii karakter , k adalah nilai paramaterter k-gram yang digunakan dan a adalah nilai basis.

1.5 Pencocokan String Dengan *Rabin Karp*

Pada dasarnya, algoritma Rabin-Karp akan membandingkan nilai *hash* dari string masukan dan substring pada teks. Apabila sama, maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila tidak sama, maka substring akan bergeser ke kanan. Kunci utama performa algoritma ini adala perhitungan yang efisien terhadapa nilai *hash* substring pada saat penggeseran dilakukan.

Dikarenakan algoritma harus membandingkan pola terhadap teks yang hasil modulonya sama tetapi nilai hashnya berbeda. Untuk menghindari pencocokan yang tidak perlu, Singh dan Kochar (2008) memberikan solusi untuk tidak hanya membandingkan sisa hasil bagi, tetapi membandingkan hasil baginya juga [8].

$$\text{REM}\ (n1/q) = \text{REM}\ (n2/q) \qquad (3)$$

$$\text{QUOTIENT}\ (n1/q) = \text{QUOTIENT}\ (n2/q) \qquad (4)$$

Jadi, successful hit harus memenuhi dua syarat, yaitu nilai sisa hasil bagi dan nilai hasil baginya harus sama. Selebihnya adalah unsuccessful hit tanpa perlu melakukan pencocokan lagi. Hal ini berarti tidak ada pemborosan waktu untuk mengecek spurious hit.

1.6 Dice's Similiairity Coeficients

Untuk menghitung nilai similarity dari dokumen fingerprint yang didapat maka digunakan *Dice Similarity Coeficients* dengan cara menghitung nilai dari jumlah K-Gram yang digunakan pada kedua dokumen yang diuji, sedangkan dokumen fingerprint didapat dari jumlah nilai K-Gram yang sama. Nilai Similarity tersebut dapat dihitung dengan menggunakan persamaan berikut :

$$S = \frac{2C}{A+B} \qquad (5)$$

Dimana S merupakan nilai similarity, dan C merupakan jumlah K-Gram yang sama dari dua buah teks yang di bandingkan, sedangkan A, B merupakan jumlah K-Gram dari masing-masing teks yang dibandingkan [5].

2. ISI PENELITIAN

2.1 Data Masukan

Analisis data masukan adalah terdiri dari dua dokumen , yaitu dokumen yang asli dan dokumen uji (dokumen yang telah di manipulasi) dengan format file berektensi .doc dan docx, dokumen di ambil dari latar belakang skripsi jurusan teknik informatika. Pada dokumen uji data hasil manipulasi nya yaitu merubah kata – kata yang merupakan kata sinonim agar dokumen uji tidak sama dengan dokumen asli.

Tabel 1. Data Masukan

Dokumen Asli	Dokumen Uji
Plagiarisme merupakan salah satu ***tindakan*** yang ***dilarang***, karena tindakan tersebut termasuk pelanggaran terhadap hasil ***karya*** seseorang yaitu dengan menjiplak atau mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ***ijin***. Selain itu, plagiat dapat menurunkan kreativitas	Plagiarisme ialah salah satu ***perbuatan*** yang ***diharamkan***, karena ***perbuatan*** merupakan pelanggaran terhadap hasil ***ciptaan*** seseorang yaitu dengan mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ***persetujuan***. Selain itu, plagiat dapat menurunkan
seseorang dalam menciptakan hasil karya. Plagiarisme sering terjadi di berbagai kalangan. Pendeteksian plagiarism dapat dilakukan dengan ***cara*** pencocokan string (*string matching*). Metode ini dapat digunakan untuk menghitung kemiripan teks antara satu dokumen dengan dokumen lainnya.	kreativitas seseorang dalam menciptakan hasil ***ciptaan***. Plagiarisme sering terjadi di berbagai kalangan[1]. Pendeteksian ***penjiplakan*** bisa dilakukan dengan menggunakan ***teknik*** pencocokan string (*string matching*).

2.2 Tahap Preprocessing

Tahap *preprocessing* yang akan dilakukan yaitu pemisahan kalimat, *case folding*, *filtering*, *tokenizing*, *stopword removal* dan *stemming*. Adapun pengertian, contoh serta gambaran sederhana dari tahap *preprocessing* dapat dilihat pada Tabel 2 sampai dengan Tabel 7.

1. Pemisahan Kalimat

Tabel 2 . Pemisahan kalimat

Sebuah Paragraf	Hasil Pemisahan Kalimat Uji		
	No	**Kalimat**	**Ref**
Plagiarisme ialah salah satu ***perbuatan*** yang ***diharamkan***, karena ***perbuatan*** merupakan pelanggaran terhadap hasil ***ciptaan*** seseorang yaitu dengan mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ***persetujuan***. Selain itu, plagiat dapat menurunkan kreativitas seseorang dalam menciptakan hasil ***ciptaan***. Plagiarisme	1	Plagiarisme ialah salah satu ***perbuatan*** yang ***diharamkan***, karena ***perbuatan*** merupakan pelanggaran terhadap hasil ***ciptaan*** seseorang yaitu dengan mengakui hasil ***karya*** orang lain sebagai hasil karya sendiri tanpa adanya ***persetujuan***.	Tidak
Plagiarisme sering terjadi di berbagai kalangan[1]. Pendeteksian ***penjiplakan*** bisa dilakukan dengan menggunakan ***teknik*** pencocokan string (*string matching*).	2	Selain itu, plagiat dapat menurunkan kreativitas seseorang dalam menciptakan hasil ***ciptaan.***	Tidak
	3	Plagiarisme Plagiarisme sering terjadi di berbagai kalangan[1].	Ya
	4	Pendeteksian ***penjiplakan*** bisa dilakukan dengan menggunakan ***teknik*** pencocokan string (*string matching*).	Tidak

Gambar 2. Pemisahan Kalimat

2. Case Folding

Tabel 3 . *Casefolding*

Hasil Pemisahan Kalimat Asli		
No	**Kalimat**	**Ref**
1	Plagiarisme merupakan salah satu ***tindakan*** yang ***dilarang***, karena tindakan tersebut termasuk pelanggaran terhadap hasil ***karya*** seseorang yaitu dengan menjiplak atau mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ***ijin***	Tidak
2	Selain itu, plagiat dapat menurunkan kreativitas seseorang dalam menciptakan hasil ***karya***.	Tidak
3	Pendeteksian plagiarism dapat dilakukan dengan ***cara*** pencocokan string (*string matching*).	Tidak

Hasil Pemisahan Kalimat Asli		
No	**Kalimat**	**Ref**
4	Metode ini dapat digunakan untuk menghitung kemiripan teks antara satu dokumen dengan dokumen lainnya.	Tidak

3. Filtering

Tabel 3 . *Filtering*

No	Kalimat sebelum *filtering*	Kalimat setelah *filtering*
1	plagiarisme merupakan salah satu ***tindakan*** yang ***dilarang***, karena tindakan tersebut termasuk pelanggaran terhadap hasil ***karya*** seseorang yaitu dengan menjiplak atau mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ***ijin***	plagiarisme merupakan salah satu ***tindakan*** yang ***dilarang***, karena tindakan tersebut termasuk pelanggaran terhadap hasil ***karya*** seseorang yaitu dengan menjiplak atau mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ***ijin***
2	selain itu, plagiat dapat menurunkan kreativitas seseorang dalam menciptakan hasil ***karya***.	selain itu, plagiat dapat menurunkan kreativitas seseorang dalam menciptakan hasil ***karya***
3	pendeteksian plagiarism dapat dilakukan dengan ***cara*** pencocokan string (*string matching*).	pendeteksian plagiarism dapat dilakukan dengan ***cara*** pencocokan string string matching
4	metode ini dapat digunakan untuk menghitung kemiripan teks antara satu dokumen dengan dokumen lainnya.	metode ini dapat digunakan untuk menghitung kemiripan teks antara satu dokumen dengan dokumen lainnya

4. Tokenizing

Tabel 5 . *Tokenizing*

No	Kalimat	Proses Tokenizing
1	plagiarisme merupakan salah satu tindakan yang dilarang, karena tindakan tersebut termasuk pelanggaran terhadap hasil karya seseorang yaitu dengan menjiplak atau mengakui hasil karya orang lain sebagai hasil karya sendiri tanpa adanya ijin	1. plagoarism 2. merupakan 3. salah 4. Satu 5. tindakan 6. yang 7. dilarang 8. karena 9. tindakan 10. tersebut 11. termasuk 12. pelanggaran 13. terhadap 14. hasil 15. karya 16. seseorang 17. tindakan 18. tersebut 19. termasuk 20. pelanggaran 21. terhadap 22. hasil 23. karya 24. seseorang

5. Stopword Removal

Menghilangkan kata – kata yang terdapat dalam kamus stopword, seperti kata depan dan kata sambung.

Tabel 6 . *Stopword Removal*

No	Hasil Stopword
1	1. plagiarism 2. salah 3. satu 4. tindakan 5. dilarang 1. tindakan 2. termasuk 3. pelanggaran 4. terhadap 5. hasil 1. karya 2. menjiplak 3. mengakui 4. hasil 5. karya

Gambar 3. Stopword Removal

6. Stemming

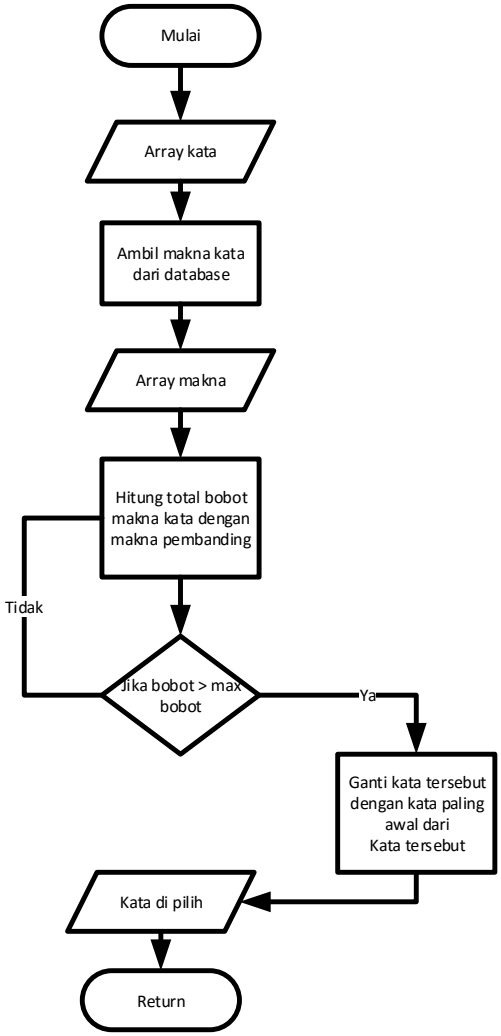
Mengubah kata yang berimbuhan ke dalam kata dasar.

Tabel 7 . *Stemming*

No	Array Kata	Hasil *Stemming*
1	6. plagiarism 7. salah 8. satu 9. tindakan 10. dilarang 6. tindakan 7. termasuk 8. pelanggaran 9. terhadap 10. hasil 6. karya 7. menjiplak 8. mengakui 9. hasil 10. karya	1. plagiarism 2. salah 3. satu 4. tindak 5. larang 11. tindakan 12. termasuk 13. langgar 14. terhadap 15. hasil 11. karya 12. jiplak 13. akui 14. hasil 15. karya

2.3 Tahap Synonim Recognition

Pada tahap ini merupakan tahap pengenalan sinonim dan pemilihan kata sinonim untuk menganti kata pada array kata dokumen asli dan array kata pada dokumen uji agar bisa mendeteksi plagiarisme tingkat subtitusi kata adapaun flowchart tahap *synonym recognition* menggunakan algoritma *lesk* dapat dilihat pada gambar 2 berikut



Gambar 2 *Flowchart Synonim recognition*

Adapun hasil dari tahap proses *synonym Recogniton* dapat dilihat pada Gambar 3 sampai dengan tabel

Kata Sinonim	Makna Kata	Kata Pembanding	Makna Kata Pembanding
Buat	sesuatu yang diperbuat (dilakukan) untuk menghasilkan sesuatu	Sikap	berbuat tidak peduli atau peduli terhadap apa yang akan terjadi menimpa dirinya dan sebagainya
		Larang	memerintahkan supaya tidak melakukan sesuatu; tidak memperbolehkan berbuat sesuatu
		Tindak	sesuatu yang dilakukan perbuatan yang dilaksanakan untuk mengatasi sesuatu
		Termasuk	sudah masuk ke dalam sesuatu
		Terhadap	kata depan untuk menandai arah
		Hasil	sesuatu yang diadakan oleh seseorang
		Karya	hasil perbuatan buatan ciptaan (terutama hasil karangan) yang bukan tiruan
		Izin	pernyataan mengabulkan yang tidak melarang dan sebagainya
Tindak	sesuatu yang dilakukan perbuatan yang dilaksanakan untuk mengatasi sesuatu	Sikap	berbuat tidak peduli atau peduli terhadap apa yang akan terjadi menimpa dirinya dan sebagainya
		Larang	memerintahkan supaya tidak melakukan sesuatu; tidak memperbolehkan berbuat sesuatu
		Tindak	sesuatu yang dilakukan perbuatan yang dilaksanakan untuk mengatasi sesuatu
		Termasuk	sudah masuk ke dalam sesuatu
		Terhadap	kata depan untuk menandai arah
		Hasil	sesuatu yang diadakan oleh seseorang

Gambar 3 Proses Perhitungan Kesamaan Makna kata

Pada tahap ini dilakukan perhitungan bobot dari makna kata sinonim dan kata makna kata pembanding nya yang akan digukana untuk menentukan kata mana yang akan di pilih untuk menggantikan kata sinonim berdasarkan bobot yang tertinggi.

Tabel 8 . Perhitungan Bobot

Kata Sinonim	Kata Pembanding	Nilai Bobot
Buat	Sikap	1
	Larang	2
	Tindak	4
	Termasuk	0
	Terhadap	1

2.4 Tahap Parsing *K-gram*

Setelah melakukan tahap pengenalan sinonim dan penentuan pemilihan kata sinonim untuk menangani masalah subtitusi kata dengan mengunakan kamuss kata sinonim, langkah selanjutnya adalah parsing *kgram*, yaitu memecah kata menjadi potongan-potongan dimana setiap potongan mengandung karakter sebanyak k.

Tabel 9 . Parsing K-Gram

Array Kata	Hasil Parsing K-Gram
Plagiarisme adalah satu perbuatan	1. Plag 2. lagi 3. agia 4. giar 5. iari 6. rism 7. isme 8. smes 9. mesa 10. esal 11. sala 12. alah 13. lahs 14. shsa 15. hsat 16. satu 17. atub 18. tubu 19. ubua 20. buat 21. uatl 22. atla

2.5 Tahap *Hashing*

Setelah kumpulan – kumpulan gram terbentuk maka tahap selanjutnya adalah merubah kumpulan – kumpulan gram tersebut ke bentuk hash, melaluli proses hashing.

Contoh Array Kata “plagi”

$= (112*10^3) + (108*10^2) + (97*10^1) + (103*10^0)$

$= (112{,}000) + (10{,}800) + (970) + (103)$

$= 123{,}873$

Nilai Mod = 123,873 mod 101

= 47

Reminder = 123,873 / 101

= 1,226.47

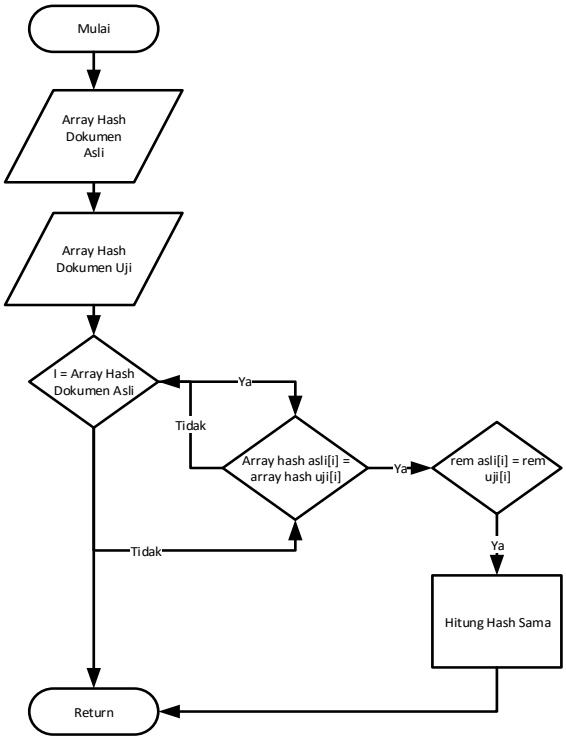
Adapun contoh proses konversi k-gram ke nilai hash dapat dilihat pada tabel 10 berikut :

Tabel 10 . Konversi *hashing*

Konversi *hashing*			
Array K-Gram	Nilai Hashing	Mod	Reminder
1. plag 2. lagi 3. agia 4. giar 5. iari	1. 123873 2. 118835 3. 108447 4. 114584 5. 115945	1. 249 2. 59 3. 276 4. 252 5. 98	1. 1226.46 2. 1176.58 3. 1073.73 4. 1134.49 5. 1147.97

2.6 Pencocokan String Dengan *Rabin Karp*

Setelah melakukan perubahan nilai k-gram ke dalam bentuk *hash* , maka tahap selanjutnya adalah melakukan proses pencocokan nilai hash yang sama dari kumpulan nilai *hash* teks uji dan kumpulan nilai *hash* teks asli dengan menggunakan algoritma *rabin - karp*, Apabila nilai hash dari kedua dokumen yang diuji sama maka,makan akan dilakukan pengecekan terhadap nilai reminder nya,.Yang nanti nya akan di hitung berapa nilai similiaty kedua kumpulan nilai *hash* tersebut. Adapun *flowchart* dari proses pencocokan string dapat dilihat pada Gambar 3.15 berikut



Gambar 4 Proses Pencocokan String

Proses Pencocokan String						
Nilai Hashing	Mod	Reminder	Nilai Hashing	Mod	Reminder	Match
1. 123873 2. 118835 3. 108447 4. 114584 5. 115945	1. 249 2. 59 3. 276 4. 252 5. 98	1. 1226.46 2. 1176.58 3. 1073.73 4. 1134.49 5. 1147.97	1. 123873 2. 118835 3. 108447 4. 114584 5. 108874	1. 249 2. 59 3. 276 4. 252 5. 198	1. 1226.46 2. 1176.58 3. 1073.73 4. 1134.49 5. 1246.30	1. Ya 2. Ya 3. Ya 4. Ya 5. No

Gambar 5 Hasil Proses Pencocokan String

Dimana hasil dari pencocokan string tersebut menghasilkan string yang sama adalah sejumlah 39 buah dari jumlah hash asli 54 buah dan hash uji sejumlah 49 buah.

2.7 Tahap Perhitungan Similiarity (*Dice's Similiarity*)

Setelah melakukan proses pencocokan string, maka dilakukan tahap penghitungan nilai similarity. Untuk menghitung nilai similarity dari dokumen fingerprint yang didapat maka digunakan Dice Similarity Coeficients dengan cara menghitung nilai dari jumlah K-Gram yang digunakan pada kedua dokumen yang diuji, sedangkan dokumen fingerprint didapat dari jumlah nilai K-Gram yang sama.

$$S = \frac{2(39)}{52+47} = \frac{78}{99} = 0.7878$$

Atau 78.78 %

3. PENUTUP

3.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa dari hasil pengujian sejumlah nilai parameter *K-Gram* (K = 2, k = 3, k = 4, k = 5) dapat disimpulkan bahwa kombinasi algoritma *lesk* untuk proses *synonym recognition* dan *rabin rabin karp* dalam penerapan pada kasus plagiarism (kesamaan) menghasilkan presentase rata – rata sebesar 85,78% dibandingkan dengan yang tidak menggunakan kombinasi algoritma *lesk* untun proses *synonim recognition* yang hanya menghasilkan presentase sebesar 77.45%. meskipun membutuhkan waktu proses yang lebih tinggi daripada yang tidak menggunakan *synonym recognition*.

3.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, masih perlu dilakukan beberapa kajian lebih lanjut. Adapun saran untuk penelitian lebih lanjut yaitu :

1. Perlu adanya proses pendeteksian plagirisme untuk mengatasi perubahan kalimat aktif dan pasif.
2. Perlu adanya proses untuk mendeteksi plagiarism pada tingkat perubahan stuktur kalimat (pharafrase).
3. Perlu adanya proses untuk menangani, plagiarisme pada perubahan urutan kalimat.
4. Perlu adanya proses untuk menangani, kata yang mengandung arti serapan dari Bahasa asing.

DAFTAR PUSTAKA

[1] I. a. I. C. S. Dewanto, "DETEKSI PLAGIARISME DOKUMEN TEKS MENGGUNAKAN ALGORITMA RABIN-KARP DENGAN SYNONYM RECOGNITION," *Program Studi Ilmu Komputer, Program Teknologi Informatika dan Ilmu Komputer , Universitas Brawijaya Malang.*.

[2] A. M. Surahman, "PERANCANGAN SISTEM PENENTUAN SIMILARITY," *Program Studi Teknik Informatika.*

[3] W. E. W. d. K. L. Masayu, "Update Summarization Untuk Kumpulan Dokumen Berbahasa Indonesia," *Jurnal Cybermatika*, vol. vol. 1, p. No 2, Desember 2013.

[4] "F. Henry Dan Z. Ery," *Klastering Dokumen Berita dari Web menggunakan Algoritma Single Pass Clustering*, vol. 18, pp. pp. 80-90, 2013.

[5] E. Nugroho, "Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Dengan Menggunakan Algoriam Rabin Karp," *Universitas Brawijaya*, 2011.

[6] A. Ledy, "Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia," *Konferensi Nasional Sistem dan Informatika*, 2009.

[7] H. B. Firdaus, "Deteksi Plagiarisme Dokumen Menggunakan Algoritma Rabin-Karp," *Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.*

[8] R. S. C. A. B. Kochar, "RB-Matcher. String Matching Technique," *Rem (Text)*, vol. 234567, pp. no 11, p.3, 2008.

[9] P. Pitria, Analisis Sentimen Pengguna Twitter pada Akun Resmi Samsung Indonesia dengan menggunakan Naive Bayes, Perpustakaan Unikom, 2014.

[10] R. S. C. a. B. Kochar, "RB-Matcher. String Matching Technique," *Rem (Text)*, Vols. 234567, no 11, p.3, 2008.