

ANALISIS K-GRAM, BASIS DAN MODULO RABIN-KARP SEBAGAI PENENTU AKURASI PERSENTASE KEMIRIPAN DOKUMEN

Andysah Putera Utama Siahaan¹, Sugianto²

¹Fakultas Ilmu Komputer, Universitas Pembangunan Panca Budi, Medan, Indonesia

²Departemen Sistem Informasi, Sekolah Tinggi Teknik Harapan, Medan, Indonesia

E-mail: ¹⁾andiesiahaan@gmail.com, ²⁾soegi1703@gmail.com

Abstrak

Pengujian kemiripan dokumen penting untuk dilaksanakan karena banyak sekali terjadi plagiarisme. Banyak metode yang dapat digunakan untuk menguji dokumen tersebut. Salah satunya adalah metode Rabin-Karp. Ada tiga parameter utama yang berperan menentukan tingkat akurasi kemiripan, yaitu K-Gram, Basis dan Modulo. Pada bagian modulo, metode ini menggunakan bilangan prima yang dapat ditentukan besarnya. Setiap dokumen akan dibentuk menjadi token-token yang kemudian dipecah kembali sesuai dengan panjang K-Gram yang ditentukan. Semakin kecil nilai K-Gram, semakin akurat analisis kemiripan tersebut. Nilai pada K-Gram, Basis dan Modulo dapat diatur sesuai dengan tingkat ketajaman analisa. K-Gram akan memotong kata-kata yang panjang menjadi kata yang mempunyai panjang yang sama. K-Gram kemudian akan membentuk nilai Hash yang berfungsi untuk memberikan ID pada tiap potongan kata. Nilai Hash juga tergantung dari Basis dan Modulo yang diberikan. Kombinasi ketiga nilai tersebut dapat menentukan akurasi persentase dari kemiripan dokumen tersebut. Pengecekan nilai Hash pada dokumen asli dan uji akan menentukan seberapa banyak hash yang memiliki nilai yang sama. Jumlah ini akan menentukan seberapa besar kemiripan dokumen asli dan uji tersebut. Penentuan kombinasi yang tepat akan menghasilkan akurasi yang baik.

Kata kunci: Plagiarisme, Rabin-Karp, Text Mining

1. PENDAHULUAN

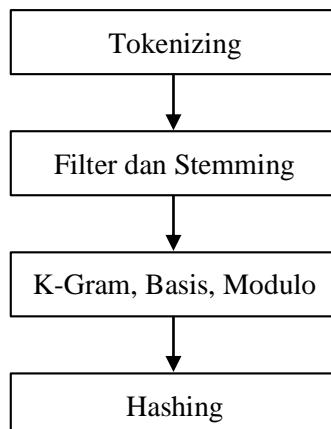
Plagiat adalah kegiatan yang sering terjadi dalam publikasi artikel. Tindakan ini sering dilakukan untuk keuntungan tertentu. Plagiarisme merupakan suatu tindak pencurian ide. Beberapa tidak plagiat diatur dalam undang-undang. Biasanya plagiarisme melakukan pengakuan terhadap suatu karya, ide atau cuplikan pernyataan menjadi milik sendiri tanpa melakukan referensi atau sitasi terhadap sumber aslinya. Plagiarisme berbeda dengan pelanggaran hak cipta. Plagiarisme lebih sering dianggap sebagai pelanggaran moral [2]. Ini dilakukan untuk meningkatkan mutu atau nilai dari seorang plagiator [3]. Kegiatan ini sering terjadi di dunia akademis yang dilakukan untuk meningkatkan angka kredit seorang peneliti.

Plagiat dapat dideteksi dengan bantuan teknik-teknik tertentu untuk mendeteksi kecurangan yang terjadi [4]. Tidak ada satupun kekuatan untuk menghilangkan tindakan plagiat. Tindakan ini berkaitan dengan moral seseorang. Ada banyak metode yang dapat digunakan untuk memeriksa kemiripan dokumen. Salah satu algoritma yang paling sering dipakai adalah Rabin-Karp. Algoritma ini bekerja berdasarkan tiga buah parameter yang dapat ditentukan masing-masing [1]. Teknik yang digunakan untuk mendeteksi tindakan plagiat terkadang mengalami kesalahan. Metode ini mempunyai beberapa kombinasi input yang berfungsi mengatur tingkat akurasi. Permasalahan yang dihadapi adalah bagaimana menentukan nilai-nilai yang digunakan pada parameter tersebut. K-Gram adalah salah satu nilai sebagai penentu keakuratan pada analisis plagiarisme. K-Gram merupakan potongan kata yang memiliki panjang yang sama dalam setiap pengecekan dokumen [5][8]. Nilai K-Gram akan menghasilkan akurasi yang berbeda pada bahasa yang digunakan. Basis dan Modulo akan bekerja menentukan nilai Hash [6][7]. Kombinasi kedua nilai ini juga menentukan seberapa sering nilai Hash tersebut muncul pada kedua dokumen yang dibandingkan. Dengan menentukan kombinasi ketiga nilai parameter ini, diharapkan tingkat akurasi persentase kemiripan dokumen akan lebih

meningkat. Hasil dapat dilihat dengan cara menguji beberapa dokumen dengan menggunakan nilai parameter yang berbeda-beda. Dengan pengujian yang banyak, tingkat akurasi akan diperoleh secara maksimal sehingga nilai parameter yang baik dapat ditentukan.

2. METODE

Tahap ini melakukan analisis semantik dan sintaksis teks. Ini bertujuan untuk menyiapkan teks yang akan di proses untuk menghasilkan nilai Hash. Operasi yang dapat dilakukan pada tahap ini adalah *Tokenizing*, *Filter*, *Stemming* dan *Hashing*. Hal ini dilakukan untuk memilih potongan kata yang telah memenuhi syarat untuk memperoleh nilai Hash. Filter adalah proses klasifikasi untuk menentukan dan menyisihkan kata-kata yang tidak digunakan dalam proses Rabin-Karp. Berikut ini adalah gambaran singkat dari alur proses penentuan nilai Hash.



Metode yang dilakukan pada Rabin-Karp akan dijelaskan pada bagian ini. Rabin-Karp memiliki tiga buah parameter yang befungsi sebagai nilai input.

Parameter tersebut adalah

- K-Gram
- Basis
- Modulo

Untuk menentukan nilai Hash, Rabin-Karp mengolah ketiga buah parameter tersebut dengan rumus berikut ini:

$$H = \left(\sum_{i=1}^n K(i) * b^{n-i} \right) \bmod m$$

Dimana:

- | | | |
|---|---|---------------|
| H | : | nilai Hash |
| K | : | karakter ke n |
| n | : | Jumlah K-Gram |
| b | : | basis |
| m | : | modulo |

Contoh berikut ini adalah penggunaan dari rumus Rabin-Karp. Disini akan dicoba penggunaan K-Gram sebesar 5 dengan potongan kata adalah “**MALANG**”. Untuk mendapatkan nilai Hash dari kata tersebut dapat dilihat dari perhitungan berikut ini:

$$\begin{aligned}
 \text{K-Gram} &: 6 \\
 \text{Basis} &: 7
 \end{aligned}$$

m : 10007

K(1)	:	M	:	77
K(2)	:	A	:	65
K(3)	:	L	:	76
K(4)	:	A	:	65
K(5)	:	N	:	78
K(6)	:	G	:	71

$$\begin{aligned}
 \text{Hash} &: (77 * 7^5) + (65 * 7^4) + (76 * 7^3) + (65 * 7^2) + (78 * 7^1) + (71 * 7^0) \\
 &: 1480074 \bmod 10007 \\
 &: 9045
 \end{aligned}$$

Dari perhitungan sebelumnya diperoleh bahwa potongan K-Gram “MALANG” akan memperoleh nilai Hash sebesar 9045. Perhitungan ini akan dilakukan sebanyak jumlah potongan kata yang ada. Sementara untuk melakukan perhitungan persentase plagiat dapat dilakukan dengan menggunakan rumus berikut ini:

$$P = \frac{2 * IH}{THA + THB} * 100\%$$

Where:

P	:	Percentase Plagiat
IH	:	Hash yang sama
THA	:	Total Hash pada dokumen A
THB	:	Total Hash pada dokumen B

3. HASIL DAN PEMBAHASAN

Bagian ini menyajikan hasil dari analisis Rabin-Karp yang dilakukan terhadap beberapa kombinasi penggunaan K-Gram, Basis dan Modulo secara terpisah. Berikut diberikan dua potongan kalimat sebagai bahan perbandingan. Kalimat pertama adalah “**malang adalah kota yang indah dan lestari**” dan kalimat kedua adalah “**malang merupakan kota yang menawan dan lestari**”.

Tabel 1 Hasil Tokenizing

No.	Token 1	Token 2
1	malang	malang
2	adalah	merupakan
3	kota	kota
4	yang	yang
5	indah	menawan
6	lestari	lestari

Tabel 1 merupakan hasil dari proses Tokenizing yang dilakukan terhadap kalimat pertama dan kalimat kedua.

Tabel 2 Hasil Filtering dan Stemming

No.	Fil./Stem. 1	Fil./Stem. 2
1	malang	malang

2	indah	awan
3	lestari	lestari

Tabel 2 merupakan hasil dari proses Filtering dan Stemming yang dilakukan terhadap kalimat pertama dana kalimat kedua yang telah mengalami proses Tokenizing.

Tabel 3 Nilai Hash kedua kalimat

No.	Hash 1	Hash 2
1	malan	malan
2	alang	alang
3	langi	langa
4	angin	angaw
5	ngind	ngawa
6	ginda	gawan
7	indah	awanl
8	ndahl	wanle
9	dahle	anles
10	ahles	nlest
11	hlest	lesta
12	lesta	estar
13	estar	stari
14	stari	

Tabel 3 adalah nilai Hash dari dua buah dokumen. Perhitungan nilai Hash yang sama akan menentukan tingkat plagiat yang ada pada kedua dokumen tersebut. Berikut perhitungan yang dilakukan untuk mendapatkan nilai Hash.

Perhitungan Hash pada kalimat pertama

```

KGram = [0]
Hash = [109*10^4] + [97*10^3] + [108*10^2] + [97*10^1] + [110*10^0]
Hash = 1198880 Mod 10007
Hash = 8047

KGram = [1]
Hash = [97*10^4] + [108*10^3] + [97*10^2] + [110*10^1] + [103*10^0]
Hash = 1088903 Mod 10007
Hash = 8147

KGram = [2]
Hash = [108*10^4] + [97*10^3] + [110*10^2] + [103*10^1] + [105*10^0]
Hash = 1189135 Mod 10007
Hash = 8309

KGram = [3]
Hash = [97*10^4] + [110*10^3] + [103*10^2] + [105*10^1] + [110*10^0]
Hash = 1091460 Mod 10007
Hash = 697

KGram = [4]
Hash = [110*10^4] + [103*10^3] + [105*10^2] + [110*10^1] + [100*10^0]
Hash = 1214700 Mod 10007
Hash = 3853
  
```

```
KGram = [5]
Hash  = [103*10^4] + [105*10^3] + [110*10^2] + [100*10^1] + [97*10^0]
Hash  = 1147097 Mod 10007
Hash  = 6299

KGram = [6]
Hash  = [105*10^4] + [110*10^3] + [100*10^2] + [97*10^1] + [104*10^0]
Hash  = 1171074 Mod 10007
Hash  = 255

KGram = [7]
Hash  = [110*10^4] + [100*10^3] + [97*10^2] + [104*10^1] + [108*10^0]
Hash  = 1210848 Mod 10007
Hash  = 1

KGram = [8]
Hash  = [100*10^4] + [97*10^3] + [104*10^2] + [108*10^1] + [101*10^0]
Hash  = 1108581 Mod 10007
Hash  = 7811

KGram = [9]
Hash  = [97*10^4] + [104*10^3] + [108*10^2] + [101*10^1] + [115*10^0]
Hash  = 1085925 Mod 10007
Hash  = 5169

KGram = [10]
Hash  = [104*10^4] + [108*10^3] + [101*10^2] + [115*10^1] + [116*10^0]
Hash  = 1159366 Mod 10007
Hash  = 8561

KGram = [11]
Hash  = [108*10^4] + [101*10^3] + [115*10^2] + [116*10^1] + [97*10^0]
Hash  = 1193757 Mod 10007
Hash  = 2924

KGram = [12]
Hash  = [101*10^4] + [115*10^3] + [116*10^2] + [97*10^1] + [114*10^0]
Hash  = 1137684 Mod 10007
Hash  = 6893

KGram = [13]
Hash  = [115*10^4] + [116*10^3] + [97*10^2] + [114*10^1] + [105*10^0]
Hash  = 1276945 Mod 10007
Hash  = 6056
```

Perhitungan Hash pada kalimat kedua

```
KGram = [0]
Hash  = [109*10^4] + [97*10^3] + [108*10^2] + [97*10^1] + [110*10^0]
Hash  = 1198880 Mod 10007
Hash  = 8047

KGram = [1]
Hash  = [97*10^4] + [108*10^3] + [97*10^2] + [110*10^1] + [103*10^0]
Hash  = 1088903 Mod 10007
Hash  = 8147

KGram = [2]
Hash  = [108*10^4] + [97*10^3] + [110*10^2] + [103*10^1] + [97*10^0]
Hash  = 1189127 Mod 10007
```

```

Hash = 8301

KGram = [3]
Hash = [97*10^4] + [110*10^3] + [103*10^2] + [97*10^1] + [119*10^0]
Hash = 1091389 Mod 10007
Hash = 626

KGram = [4]
Hash = [110*10^4] + [103*10^3] + [97*10^2] + [119*10^1] + [97*10^0]
Hash = 1213987 Mod 10007
Hash = 3140

KGram = [5]
Hash = [103*10^4] + [97*10^3] + [119*10^2] + [97*10^1] + [110*10^0]
Hash = 1139980 Mod 10007
Hash = 9189

KGram = [6]
Hash = [97*10^4] + [119*10^3] + [97*10^2] + [110*10^1] + [108*10^0]
Hash = 1099908 Mod 10007
Hash = 9145

KGram = [7]
Hash = [119*10^4] + [97*10^3] + [110*10^2] + [108*10^1] + [101*10^0]
Hash = 1299181 Mod 10007
Hash = 8278

KGram = [8]
Hash = [97*10^4] + [110*10^3] + [108*10^2] + [101*10^1] + [115*10^0]
Hash = 1091925 Mod 10007
Hash = 1162

KGram = [9]
Hash = [110*10^4] + [108*10^3] + [101*10^2] + [115*10^1] + [116*10^0]
Hash = 1219366 Mod 10007
Hash = 8519

KGram = [10]
Hash = [108*10^4] + [101*10^3] + [115*10^2] + [116*10^1] + [97*10^0]
Hash = 1193757 Mod 10007
Hash = 2924

KGram = [11]
Hash = [101*10^4] + [115*10^3] + [116*10^2] + [97*10^1] + [114*10^0]
Hash = 1137684 Mod 10007
Hash = 6893

KGram = [12]
Hash = [115*10^4] + [116*10^3] + [97*10^2] + [114*10^1] + [105*10^0]
Hash = 1276945 Mod 10007
Hash = 6056
  
```

Tabel 4 Hasil perhitungan Hash (K-Gram=5, Basis=10, Modulo=10007)

No.	K-Gram 1	Hash 1	K-Gram 2	Hash 2
1	malan	8047	malan	8047
2	alang	8147	alang	8147
3	langi	8309	langa	8301

4	angin	697	angaw	626
5	ngind	3853	ngawa	3140
6	ginda	6299	gawan	9189
7	indah	255	awanl	9145
8	ndahl	1	wanle	8278
9	dahle	7811	anles	1162
10	ahles	5169	nlest	8519
11	hlest	8561	lesta	2924
12	lesta	2924	estar	6893
13	estar	6893	stari	6056
14	stari	6056		

Pengujian ini menggunakan parameter K-Gram = 5, Basis = 10, dan Modulo = 1007. Total nilai Hash berjumlah 27 sementara nilai Hash yang sama adalah sebanyak 5 buah Hash. Persentase plagiat dapat dilihat pada perhitungan berikut ini.

$$\begin{aligned}
 P &= \frac{2*5}{14+13} * 100\% \\
 &= \frac{10}{27} * 100\% \\
 &= 0.370370 * 100\% \\
 &= 37,0370\%
 \end{aligned}$$

Hasil dari perhitungan menunjukkan tingkat persentase plagiat adalah sebesar 37,037%. Hasil ini akan berbeda dengan kombinasi K-Gram, Basis dan Modulo yang berbeda juga. Tabel 5 dan 6 berikut ini menunjukkan hasil perbandingan dengan menggunakan kombinasi lainnya.

Tabel 5 Hasil perhitungan Hash (K-Gram=3, Basis=7, Modulo=5001)

No.	K-Gram 1	Hash 1	K-Gram 2	Hash 2
1	mal	1127	mal	1127
2	ala	605	ala	605
3	lan	1080	lan	1080
4	ang	625	ang	625
5	ngi	1215	nga	1207
6	gin	891	gaw	844
7	ind	1014	awa	682
8	nda	1186	wan	1619
9	dah	682	anl	630
10	ahl	588	nle	1246
11	hle	952	les	1113
12	les	1113	est	869
13	est	869	sta	1543
14	sta	1543	tar	1476
15	tar	1476	ari	655
16	ari	655		

$$P = \frac{2*10}{16+15} * 100\%$$

$$\begin{aligned}
 &= \frac{20}{31} * 100\% \\
 &= 0.645161 * 100\% \\
 &= 64,5161\%
 \end{aligned}$$

Tabel 6. Hasil perhitungan Hash (K-Gram=10, Basis=20, Modulo=2999)

No.	K-Gram 1	Hash 1	K-Gram 2	Hash 2
1	malanginda	608	malangawan	141
2	alangindah	2803	alangawanl	2464
3	langindahl	1608	langawanle	819
4	angindahle	298	angawanles	2526
5	ngindahles	2498	ngawanlest	2074
6	gindahlest	138	gawanlesta	636
7	indahlest	2336	awanlesttar	317
8	ndahlesttar	1347	wanlestari	2868
9	dahlestari	1099		

$$\begin{aligned}
 P &= \frac{2*0}{9+8} * 100\% \\
 &= \frac{0}{17} * 100\% \\
 &= 0 * 100\% \\
 &= 0\%
 \end{aligned}$$

Tabel 7 Hasil perbandingan percobaan

No.	K-G	B	M	Hash 1	Hash 2	Identik	Plagiat
1	5	10	10007	14	13	5	37,0370
2	3	7	5001	16	15	10	64,5161
3	10	20	2999	9	8	0	0

Tabel 7 menunjukkan hasil perbandingan pada ketiga percobaan. Hasil tingkat plagiat yang dihasilkan bervariasi untuk setiap kombinasi nilai K-Gram, Basis dan Modulo yang digunakan.

4. KESIMPULAN

Dari perhitungan sebelumnya dapat dilihat bahwa tingkat plagiat bervariasi berdasarkan kombinasi penggunaan nilai K-Gram, Basis dan Modulo. Penggunaan ini dapat disesuaikan dengan tingkat plagiat yang ingin dicapai. Menurunkan nilai K-Gram dapat mempertinggi atau menambah akurasi hasil plagiat yang diperoleh. Sementara menurunkan nilai Basis dan Modulo akan meninggikan nilai akurasi tersebut. Akan tetapi, perhitungan ini tidak dapat mengetahui dokumen mana yang sebenarnya asli atau palsu. Ini dapat dilihat dari tanggal penerbitan dokumen tersebut. Metode ini hanya membantu untuk membandingkan artikel sebelum dipublikasikan ke media online.

DAFTAR PUSTAKA

- [1] S. K. Shivaji and P. S., "Plagiarism Detection by using Karp-Rabin and String Matching Algorithm Together," International Journal of Computer Applications, vol. 116, no. 23, pp. 37-41, 2015.
- [2] A. Parker and J. O. Hamblen, "Computer Algorithm for Plagiarism Detection," IEEE Trans. Education, vol. 32, no. 2, pp. 94-99, 1989.

- [3] Sunita, R. Malik and M. Gulia, "Rabin-Karp Algorithm with Hashing a String Matching Tool," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 3, pp. 389-392, 2014.
- [4] A. P. Gope and R. N. Behera, "A Novel Pattern Matching Algorithm in Genome," International Journal of Computer Science and Information Technologies, vol. 5, no. 4, pp. 5450-5457, 2014.
- [5] R. E. Putri and A. P. U. Siahaan, "Examination of Document Similarity Using Rabin-Karp Algorithm," International Journal of Recent Trends in Engineering & Research, vol. 3, no. 8, pp. 196-201, 2017.
- [6] S. Popov, "Algorithm of the Week: Rabin-Karp String Searching," DZone / Java Zone, 3 April 2012. [Online]. Available: <https://dzone.com/articles/algorithm-week-rabin-karp>. [Accessed 20 August 2017].
- [7] J. Sharma and M. Singh, "CUDA based Rabin-Karp Pattern Matching for Deep Packet Inspection on a Multicore GPU," International Journal of Computer Network and Information Security, vol. 10, no. 8, pp. 70-77, 2015.
- [8] A. P. U. Siahaan, "K-Gram as a Determinant of Plagiarism Level in Rabin-Karp Algorithm," International Journal of Scientific & Technology Research, vol. 6, no. 7, pp. 350-353, 2017.