Algoritma Boyer Moore Untuk Penyaringan Pesan Teks Menggunakan Perbandingan Kata Yang Sama

Thesis · April 2017

CITATIONS
0

READS
638

1 author:

Tomy Satria Alasi
STMIK LOGIKA
4 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Seminar Nasional Sains & Teknologi Informasi (SENSASI) ISBN: 978-602-52720-0-4
SENSASI 2018
Juli 2018
Hal: 488 - 495

Algoritma Boyer Moore Untuk Penyaringan Pesan Teks Menggunakan Perbandingan Kata Yang Sama

Tomy Satria Alasi

Prodi Teknik Informatika, STMIK Harapan Ibu, Langsa, Indonesia
Email: tomysatriaalasi@live.com

Abstrak

Berbagi pesan teks yang tersedia saat ini mengirim seluruh kata dan kalimat yang dikirim kepenerima, dari hal tersebut menimbulkan masalah yaitu adanya kata kotor yang ditulis oleh pengirim diteruskan kepenerima maka perlu menyediakan interaksi yang sehat diantara pengguna dalam berbagi pesan teks sebagai kebutuhan sosial dengan penyaringan pesan teks. Penyaringan pesan teks yaitu membandingkan kata-kata yang sama antara kata kotor jika terdapat sesuai dengan pesan teks yang akan dikirim maka solusi ditemukan yaitu diblokir kalimat kemudian dihapus tanpa terkirim, pada ilmu komputer perbandingan kata tersebut diatasi dengan pencarian string. Pencarian string adalah hal yang penting dalam pemrosesan menemukan teks seperti menemukan sebuah teks salah satu algoritma pada pencarian string adalah algoritma boyer-moore. Algoritma Boyer-Moore merupakan menyelesaikan pencarian teks dengan dua proses yaitu teknik kebenaran kaca yaitu menemukan kata yang ingin dicari dengan pencocokan dari kanan dan perpindahan dari satu kondisi ke kondisi selanjutnya dalam menemukan teks.

Kata Kunci: Berbagi pesan teks, Penyaringan pesan teks, Pencarian string, Algoritma Boyer-Moore.

1. PENDAHULUAN

Sistem *chatting* yang tersedia saat ini akan meneruskan seluruh kata dan kalimat yang dikirim kepenerima, masalahnya adalah adanya kata yang kotor yang ditulis oleh pengirim juga akan diteruskan kepenerima. Kebebasan berbagi pesan tersebut menimbulkan masalah yakni adanya penyalahgunaan teknologi yaitu mengirim kata-kata tidak sopan, kata-kata pengancaman atau menteror, kata-kata gangguan (*phising)*, kata-kata jorok dan kata-kata isu sara dikirim kepada penerima sehingga terjadinya pencemaran nama baik, pencemaran suku, pencemaran agama, pencemaran khas, pencemaran institusi dan pencemaran organisasi bahkan pencemaran negara, tentu saja hal ini menyalahkan penggunaan teknologi. Masalah adanya kata-kata yang kotor ditulis oleh pengirim dituruskan kepada penerima dapat diatasi menggunakan penyaringan kata kotor pada pesan teks dengan algoritma Boyer-Moore salah satu dari teknik *string matching*. *String matching* adalah teknim membaca sebuah teks oleh komputer dengan mengenal karakter dan membaca kata tersebut representasi sebagaimana manusia membaca. Algoritma Boyer-Moore pencocokan kata dari kanan ke kekiri menandai ahir dari kata dan menyesuaikan tentu pencarian lebih cepat.

2. METODOLOGI PENELITIAN

String matching merupakan algoritma pencocokan *string*. Sifat algoritma *string* matching adalah mencari sebuah *string* yang terdiri dari beberapa karakter (yang biasa disebut *pattern*). merupakan bagian dari pemrosesan data, pengkompresian data, *lexical analysis*, dan temu balik informasi. Teknik untuk menyelesaikan permasalahan pencocokan string biasanya akan menghasilkan implikasi langsung ke aplikasi *string* lainnya. Pencocokan *string* atau *string matching* adalah proses pencarian semua kemunculan *string* sehingga *string matching* adalah algoritma yang ditujukan untuk melakukan pencocokan *sub string* pada *string besar*. [1][2][3].

Prinsip kerja algoritma *string matching* adalah sebagai berikut : memindai teks dengan bantuan sebuah *window* yang ukurannya sama dengan panjang *pattern*, menempatkan *window* pada awal teks, membandingkan karakter pada *window* dengan karakter dari *pattern*, Setelah pencocokan (baik hasilnya cocok atau tidak cocok), dilakukan *shift* ke kanan pada window. Prosedur ini dilakukan berulang-ulang sampai *window* berada pada akhir teks, mekanisme ini disebut mekanisme *sliding[2]*.

2.1 Algoritma Boyer-Moore

Algoritma Boyer-Moore adalah pencocokan *pattern* dari kanan ke kiri maka, informasi yang didapat akan lebih banyak, seperti *pattern* dengan panjang *pattern* diletakkan pada ujung kiri atas dari

sebuah String, sehingga kedua karakter yang pertama sejajar dan *char* adalah karakter ke-*pattern* dari *string*, maka karakter tersebut akan sejajar dengan karakter terakhir dari *pattern*. Tidak seperti algoritma pencarian string lain, algoritma Boyer-Moore mulai mencocokkan karakter dari sebelah kanan *pattern*, ide dibalik algoritma ini adalah bahwa dengan memulai pencocokkan karakter dari kanan, dan bukan dari kiri, maka akan lebih banyak informasi yang didapat menjelaskan bahwa secara sistematis[4][5].

Langkah-langkah yang dilakukan algoritma Boyer-Moore pada saat mencocokkan *string* adalah :

1. Algoritma Boyer-Moore mulai mencocokkan *pattern* pada awal teks.
2. Dari kanan ke kiri, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
   a. Karakter di pattern dan di teks yang dibandingkan tidak cocok (*mismatch*).
   b. Semua karakter di pattern cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Menggeser *pattern* dengan memaksimalkan nilai penggeseran *good-suffix* dan penggeseran *bad-character*, lalu mengulangi langkah 2 sampai pattern berada di ujung teks. Asumsikan bahwa terjadi ketidakcocokan antara karakter[5]:

   $P\ [i] = b$ dari *pattern* dan karakter $T\ [i + j]$ = sebuah teks dalam suatu upaya pada posisi j..... (1)

   kemudian, $P\ [i + 1 .. m- 1] = T\ [i + j + 1 .. j + m-1] = u$ dan $P\ [i] \neq T\ [i + j]$ pergeseran baik-akhiran terdiri dalam menyelaraskan segmen $T\ [i + j + 1 .. j + m- 1] = P\ [i + 1 .. m-1]$ ..... (2)

   dengan terjadinya paling kanan di P yang diawali dengan karakter yang berbeda dari $P\ [i]$.

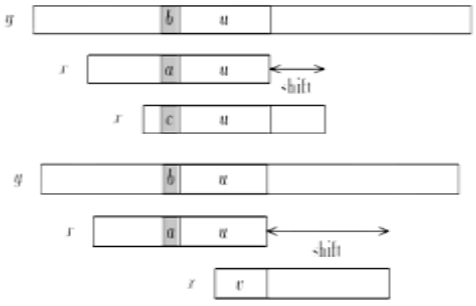
4. Algoritma Boyer-Moore ini juga memiliki beberapa aturan untuk pergeseran *pattern* yaitu *good-suffix rule* dan *bad character rule*.

   a. Good-Suffix Rule

   Good Suffix adalah cara lain mengatakan berapa banyak karakter yang kita dapat melewatkan jika tidak ada suatu kecocokan. Heuristik didasarkan pada urutan mundur pencocokan Boyer-Moore[4][5].

   $T = [i + j + 1 .. j + m- 1]$..... (3)

   dengan awalan pencocokan P



Gambar 1. Good-Suffix Shift, U Terjadi Lagi Didahului Karakter C Berbeda Dari A

Pergeseran dari x[i]=a ke karakter lain yang letaknya lebih kiri dari x[i] dan terletak di sebelah kiri segmen u, jika tidak ada segmen yang sama dengan u, maka dicari u yang merupakan suffiks terpanjang u.

b. Bad-Character Rule

Bad Character menunjukkan seberapa banyak pergeseran karakter dapat dengan aman melompat ke depan dalam teks setelah ketidakcocokan.

Pergeseran *bad-character* terdiri dalam menyelaraskan karakter teks dengan $T\ [i + j]$ dengan terjadinya paling kanan di $P\ [0 ... m-2]$..... (4)

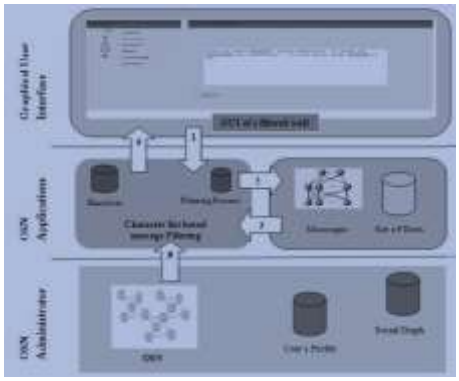
Jika *T [i + j]* tidak terjadi dalam pola *P*, tidak terjadinya *P* di *T* dapat mencakup *T [i j]*, dan ujung kiri *window* sejajar dengan karakter segera setelah *T [i + j]*, yaitu *T [i + j + 1]* ..... (5)

fungsi *bad-character* untuk menghitung posisi membandingkan baru, bergeser ke kanan *P* dengan mengambil maksimum dua nilai tersebut. Jika *bad character* y[i+j] terdapat pada pattern di posisi terkanan k yang lebih kanan dari x[i] maka pattern seharusnya digeser sejauh i-k yang hasilnya negatif (pattern digeser kembali ke kiri). Maka bila kasus ini terjadi akan diabaikan.

Algoritma Boyer-Moore cepat dalam kasus alfabet yang lebih besar. Pada tahap preprocessing, waktu dan kompleksitas ruang O (m + σ), di mana σ adalah ukuran karakter yang terbatas set relevan dengan pola dan teks. Dalam mencari kompleksitas waktu fase di O (mn). Ada perbandingan karakter teks 3n dalam kasus terburuk ketika mencari pola non periodik. Di bawah terbaik kompleksitas waktu kinerja adalah O (n / m). Di bawah kompleksitas waktu terburuk adalah O (mn)[5].

2.2 Penyaringan Pesan Teks

Penyalahgunaan kata-kata pada pengguna komputer saat ini adalah masalah serius. tetapi penyaringan kata-kata tidak sopan yang ada banyak mamakan waktu saat diproses, sehingga perlu mengarah sesederhana mungkin kata yang harus saat di *filter*. Penyaringan kata pulgar terkait dengan penyaringan spam, yang akan kita bahas secara singkat. pemfilteran spam dibagi menjadi dua kategori: menyelidiki isi dari email dengan kata kunci dan menggunakan reputasi pengirim *email*. Sistem informasi penyaringan yang direncanakan untuk mengkategorikan aliran informasi yang dihasilkan secara dinamis sebuah ditransmisikan *asynchronous* oleh *requiments* informasi. Setiap klien penyaringan berpendapat-dasar operasi secara terpisah dan karena ini, harus ada korespondensi antara berpendapat item dan pengguna[6][7][8][9].

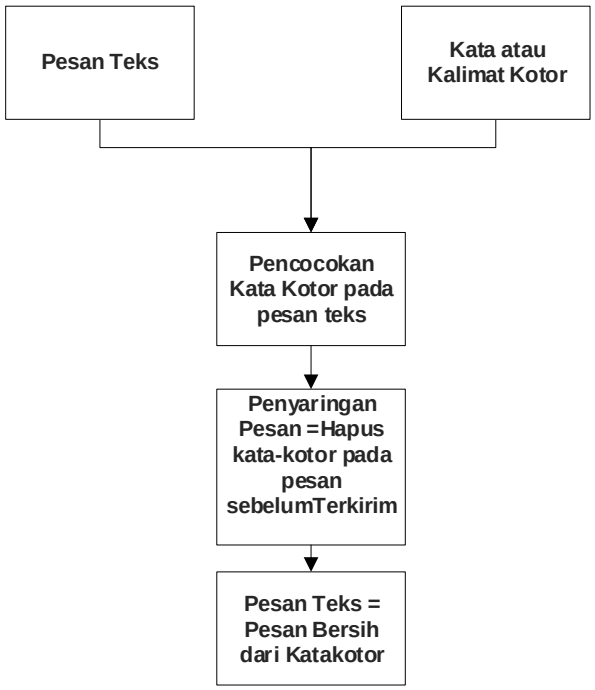


Gambar 2. Sistem Arsitektur Penyaringan Pesan

3. ANALISA DAN PEMBAHASAN

3.1 Pesan Teks dan Kata-kata Disaring

Pesan teks yang diinput merupakan kalimat yang bebas saat diinput untuk berbagi informasi sebagai kebutuhan dikehidupan dikirim kepada penerima, dan kata-kata kotor diinput oleh pengguna dan *administrator* dan kata tersesbut disaring bila telah disetujui oleh *administrator* aplikasi sehingga diperlukan *database* penyenyimpan kata-kata disaring.



Gambar 3. Proses Penyaringan Pesan Teks

Agar lebih memudahkan berikut pesan teks dan kata-kata kotor. Kata kotor tidak dilewatkan atau dikirim kepada penerima sehingga kata kotor tersebut harus dibentuk dan dikumpulkan disebuah tempat penyimpanan data yang besar dimana akan terus bertambah oleh pengguna sendiri. Semakin banyak kata kotor terkumpul semakin banyak proses dilakukan oleh sistem didalam pencarian pesan teks dengan membandingkan antara kalimat pesan dan kata kotor tersebut dengan dilakukan perulangan, kata kotor diambil yaitu kata-kata yang banyak kategori seperti kata kotor kategori isu sara, kata kotor kategori umum, kata kotor kategori cacat, kata kotor kategori kebodohan, kata kotor kategori trans dan psk, kata kotor kategori kelamin, kata kotor kategori binatang, kata kotor.

3.2 Penyaringan Pesan

Menentukan Kata Kotor sebagai pattern

$$oh = p\,[i + 1 \,..\, m - 1]$$

Keterangan:

oh = *integer* : nilai acuransy heuristik
p = *integer* : kata kotor menjadi array
i = *integer* : membuat *array* kata kotor
m = *integer* : array nilai terahir pada kata kotor

Sehingga berikut perhitungan mencari *occurance heuristic.* Mamanfaatkan tabel *pattern* atau kata kotor.

Tabel 1. hasil ahir pencarian occurance heuristik “biadap kau”

	b	i	d	P		k	A	u
0								0
1							1	
2						2		
3					3			

4				4					
5								5	
6			6						
7								7	
8		8							
9	9								

Proses untuk *occurence heuristic* mencari panjang *pattern* Langkah-langkah pemberian nilai *occurance heuristic (oh)* adalah sebagai berikut :

a. Lakukan perhitungan, oh=(length -1 – index) length = 10.
b. Karakter pertama adalah “b” dengan index = 10 oh=(10 - 1 - 0 =9) maka nilai karakter “b” = 9.
c. Karakter kedua adalah “i” dengan index = 9 oh= (9-1-0=8) maka nilai karakter “a” = 8.
d. Karakter ketiga adalah “a” dengan index = 8 oh= (8-1-0=7) maka nilai karakter ‘k” = 7.
e. Karakter keempat adalah "d" dengan index = 7, oh= (7-1-0=6) maka nilai karakter "d" = 6.
f. Karakter kelima adalah "a" dengan index = 6, oh = (6-1-0=5) maka nilai karakter "a"= 5.
g. Karakter keenam adalah "p" dengan index = 5, oh= (5-1-0=4) maka nilai karakter "p"= 4
h. Karakter ketujuh adalah " " dengan index = 4, OH = (4-1-0=3) maka nilai karakter " "= 3
i. Karakter kedelapan adalah "k" dengan index = 3, OH = (3-1-0=2) maka nilai karakter "k" = 2.
j. Karakter kesembilan adalah “a” dengan index=2, oh=(2-1-0=0) maka nilai karakter a=1
k. Karakter kesepuluh adalah "u" dengan index = 1, oh = (1-1-0=0) maka nilai karakter "b" = 0.
l. Jika ada karakter yang berulang ambil nilai oh terkecil, dalam kasus ini ada karakter “a” yang bernilai 1,5 dan 7, maka jadikan karakter “a” bernilai terkecil atau dalam kasus ini bernilai1.

Pencocokan kata-kotor Asumsikan bahwa terjadi ketidakcocokan antara karakter. Cara kerja dari algoritma Boyer-Moore adalah sebagai berikut:

$$P\,[i + 1 \,..\, m\text{-}\,1] = T\,[i + j + 1 \,..\, j + m\text{-}1] = u$$
$$P\,[i] \neq T\,[i + j] \;\;, T\,[i + j + 1 \,..\, j + m\text{-}\,1] = P\,[i + 1 \,..\, m\text{-}1] = P[i]$$

Keterangan :

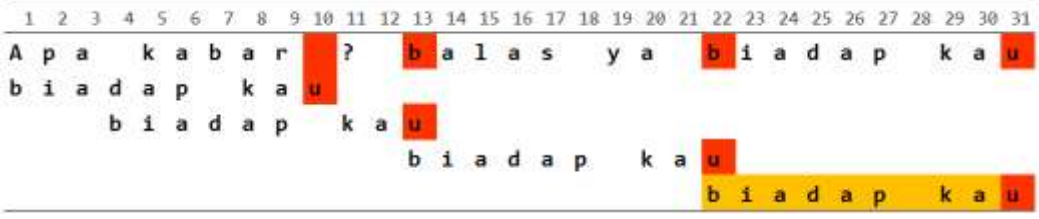
P = *Integer*, patten
T = *Integer, Text*
u = *Integer,*
i,j = *Integer*
m = *integer, array*

Perhitungan dimulai dari angka nol. Berikut adalah tabel pencocokan *string* dengan menggunakan algoritma Boyer-Moore, dijelaskan prinsip rumus diatas yaitu sebagai berikut :

p = “biadap kau” = 10.
i = {1,2,3,4,5,6,7,8,9,10}.
t = “Apa kabar ? balas ya biadap kau” = 31.
m={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31 }.
j = {3}.

maka $P\,[i + 1 \,..\, m\text{-}\,1] = T\,[i + j + 1 \,..\, j + m\text{-}1] = u$ yaitu P10+1 dibandingkan dengan T{31} maka:

Tabel 2. Pencocokan String Pada Kata Kotor



Berikut penjelasan dari proses perhitungan agar lebih mudah dipahami berdasarkan tabel pencarian yang akurat :

a. Ketika pencocokan pertama kali, indeks ke-9 pada teks "u", u bernilai sembilan berdasar tabel acurancy heuristic yaitu tabel hasil ahir pencarian heuristik , tidak cocok dengan pattern pertama paling kanan " " atau spasi. Hal ini membuat pattern digeser sejauh nilai maksimal dari tabel delta atau sejumlah pergeseran teks. Nilai maksimal dari nilai delta huruf " " yakni 3 (lihat tabel delta) dan pergeseran pattern 3 kali. Sehingga pattern digeser sejauh 3 langkah.
b. Pencocokan kedua, indeks ke-9 pada teks "u" tidak cocok dengan pattern pertama paling kanan "b". Hal ini membuat pattern digeser sejauh nilai maksimal dari tabel delta atau sejumlah pergeseran teks. Dicari nilai maksimal dari nilai delta huruf "b" yakni 9 (lihat tabel delta) dan pergeseran pattern masih 9 kali. Sehingga pattern digeser sejauh 9 langkah.
c. Pencocokan ketiga, Indeks ke-9 pada teks "u" tidak cocok dengan pattern pertama paling kanan "b". Hal ini membuat pattern digeser sejauh nilai "b" pada dari tabel delta atau sejumlah pergeseran teks. Dicari nilai maksimal dari nilai delta huruf "b" yakni 9 (lihat tabel delta) dan pergeseran pattern masih 9 kali. Sehingga pattern digeser sejauh 9 langkah.
d. Pencocokan ketiga yakni pada indeks ke-31. Setelah dilakukan pengecekan dari kanan ke kiri teks terhadap pattern, ditemukan kecocokan seluruh pattern pada teks. Ketika Seluruh pattern telah berhasil dicocokan.

3.4 Perancangan Database

Pada sistem ini ada 3 tabel didalam perencanaan database. Berguna untuk menyimpan data-data yang akan diinput oleh program aplikasinya. Berikut tabel yang dibutuhkan terdiri dari:

Tabel 3. Struktur Tabel User

Kolom	Tipe	Ukuran	Keterangan
kode	Varchar	20	Primary Key
password	Varchar	30	
email	Varchar	40	
status	Varchar	10	

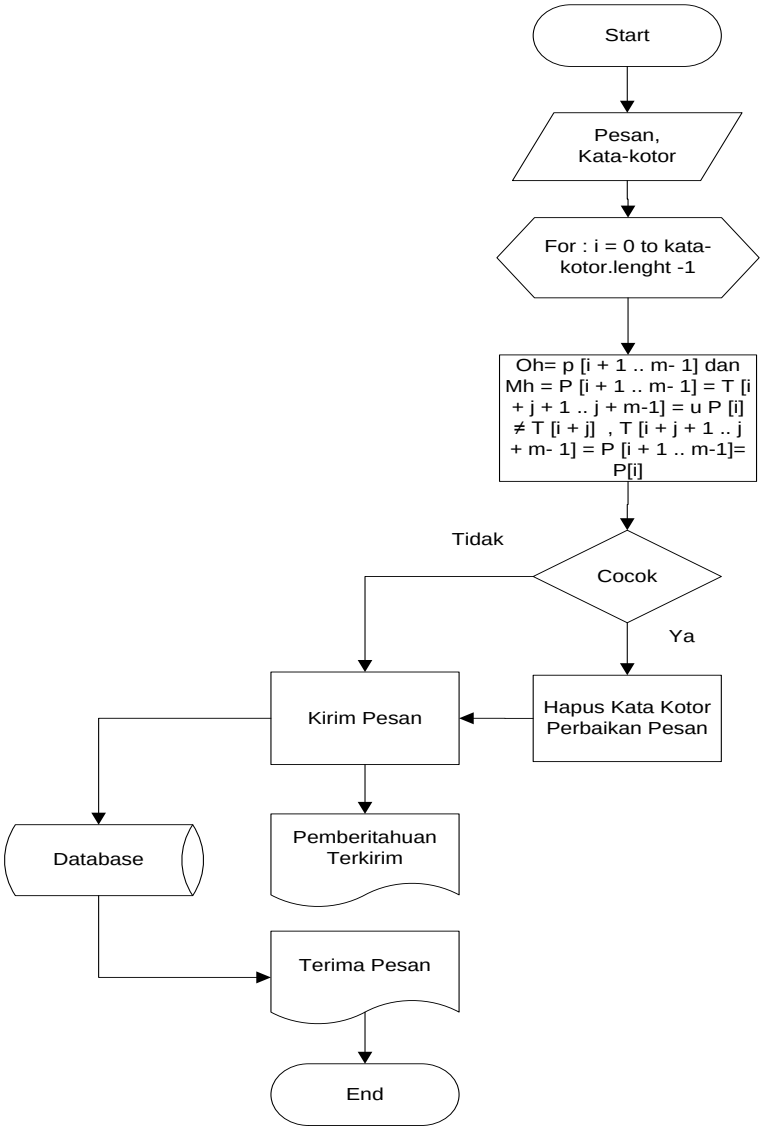
Tabel 4. Struktur Tabel Kata-Kotor

Kolom	Tipe	Ukuran	Keterangan
kode	Int	10	Primary Key
kata	varchar	255	
kategori	varchar	40	
status	varchar	12	
keterangan	varchar	255	

Tabel 5. Struktur Tabel Pesan

Kolom	Tipe	Ukuran	Keterangan
kode	int	10	primary key
pengirim	varchar	20	
penerima	varchar	20	
pesan	varchar	255	
waktu	datetime		

3.5 Flowchart

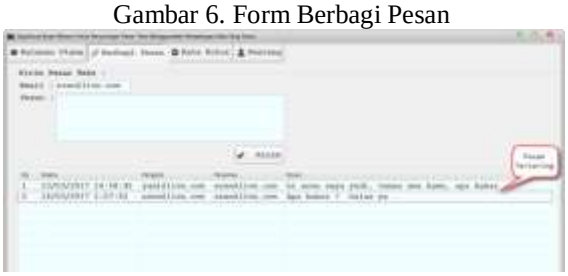


Gambar 4. Flawchart Penyaringan Pesan Teks

Implementasi sistem adalah tahap lanjut setelah perancangan sistem, pada implementasi sistem dijelaskan bagaimana menggunakan dan mengoperasikan sistem atau aplikasi penyaringan pesan teks yang telah dibangun menggunakan algoritma Boyer-Moore untuk penayaringan pesan .

Gambar 5. Form Utama

Gambar 6. Form Berbagi Pesan



Gambar 7. Pesan Tersaring

Gambar 8. Tentang Penulis

4. KESIMPULAN

Perbandingan kata yang sama tersimpan di basis data dan bersifat dinamis dapat bertambah kapan saja pada pesan teks dengan algoritma Boyer-Moore yaitu membandingkan kata dari kanan ke kiri teks dengan menemukan *good suffix* dan *bad character*.

REFERENCES

[1] Kurniawan. D. H, Munir, R, "A New String Matching Algorithm Based on Logical Indexing. IEEE" , 394, 2015
[2] Silalahi. N, "PENERAPAN ALGORITMA STRING MATCHING DALAM PENCARIAN RESEP MASAKAN BERBASIS ANDROID," Sniti., 107-109, 2015.
[3] Danuri , "Pencarian File Teks Berbasis Content dengan Pencocokan String Menggunakan Algoritma Brute" force.sji, 68-75, 2016.
[4] Kristanto. G, S., Rahmat, A., & Gunawan, S, "IMPLEMENTASI ALGORITMA BOYER-MOORE PADA PERMAINAN WORD SEARCH PUZZLE," ReseachGate, 2014.
[5] Ginting. G. L, PENERAPAN ALGORITMA BOYER MOORE PADA PENCARIAN OBJEK WISATA BERBASIS WEBSITE. SNITI, 2015