ISSN 2541-1004

PENCARIAN DATA TIKET *MAINTENANCE* MENGGUNAKAN METODE *BRUTE FORCE*

Anis Mirza
Fakultas Teknik, Universitas Pamulang
e-mail: dosen00289@unpam.ac.id

ABSTRAK

Penelitian ini bertujuan untuk melakukan pencarian data maintenace secara cepat yang terkirim melalui email di storage komputer melalui bantuan metode pencarian query. Pencarian manual yang dilakukan selama ini menjadi issue lamanya penanganan pencarian data maintenance sehingga menghambat flow bussiness berikutnya. Penanganan yang berjalan selama ini adalah dengan menggunakan proses searcing tool konvensional dan memerlukan banyak penggunaan sumber daya yang ada baik aspek sumberdaya manusia dan waktu dalam menagani satu *trouble shooting* yang berdampak pula pada bengkaknya biaya operasional yang ada. Sehingga diperlukan suatu metode pencarian yang lebih layak. Metode pencarian query tersebut adalah metode yang menggunakan string matching yang dalam pengoperasiannya mampu memangkas waktu pencarian data hingga 50% lebih cepat daripada metoda semula sehingga penyelesaian tugas dapat tertangani dengan segera.

Kata Kunci: Pencarian query, flow bussiness, trouble shooting, string matching

1. PENDAHULUAN

Perusahaan yang bergerak dalam bidang sevice sangat dibutuhkan suatu terobosan yang berorientasi customer terutama dalam kaitannya dalam hal ini maintenance ticketing. Berhubungan dengan pencarian data tiket maintenance ini, pencarian data laporan yang dilakuka nmasi hmengandalkan data-data softcopy yang dikirimmelalui email oleh karyawan dan *hardcopy.* Data tersebut berisi informasi detail data *maintenance*. Dengan digunakannya sistem tersebut menyulitkan admin untuk memperoleh informasi laporan maintenance yang diperlukan, karena harus mencari data laporan yang dikirim melalui email oleh karyawan di *storge komputer.* Akibatnya ada beberapa masalah yang timbul, seperti data laporan maintenance tidak terdownload serta kesalahan dalam menyimpan data laporan maintenance sehingga sulit mencari data laporan maintenance saat dibutuhkan.

Berkembangnya teknologi dan kebutuhan akan data, menuntut agar pengguna dapat memperoleh kebutuhan data dengan cepat dan relevan (Pardede, Barmawi, & Pramono, 2013). Berdasarkan hal tersebut dibutuhkan sebuah sistem yang mampu mengakomodasi kebutuhan akan perolehan data tersebut. Sistem yang dapat membantu dalam proses pencarian data dapat dikelompokkan dalam metode pencarian *query*. Penerapan dalam pencarian *query* dapat dilakukan dengan menggunakan *string matching* (B & Hetami, 2015). *String matching* merupakan sebuah metode yang digunakan untuk pencarian kata atau *string* (Sagita & Prasetiyowati, 2013). Pencarian kata atau *string* adalah untuk mencari beberapa karakter dalam sejumlah teks dengan porsi besar pada saat pencarian (Putri, Ernawati, & Puspitaningrum, 2015). Dengan demikian dapat disimpulkan bahwa pencarian *string* adalah pencarian *patter n* dari teks (Sagita & Prasetiyowati, 2013). Beberapa contoh penerapan *string matching* dalam sebuah aplikasi antara lain adalah *text editor*, *Search* dan *Subtitude.* Penerapan *string matching* yang lebih besar lagi adalah pencarian dengan suatu kata kunci pada internet seperti yang dilakukan oleh Google, Yahoo, dan Bing (Ginting, 2014).

Beberapa algoritma yang digunakan oleh peneliti dalam proses *string matching* seperti Knuth Morris Pratt, Boyer Moore dan Brute Force (Utomo, Harjo, & Handoko, 2008). Algoritma tersebut dapat diterapkan dalam sebuah aplikasi pencarian. Algoritma Knuth Morris Pratt dalam pencocokan *string* sama dengan Brute Force, yaitu dari kiri ke kanan. Tetapi perbedaan dari algoritma Brute Force pada pergeserannya. Algoritma Knuth

Morris Pratt melakukan proses awal terhadap *pattern* dengan menghitung fungsi pinggiran, dengan adanya fungsi pinggiran ini maka dapat dicegah pergeseran yang tidak berguna. Selain itu ada juga Algoritma Boyer Moore, yaitu metode yang menggunakan arah dari kanan ke kiri dalam pencocokan *pattern* (Pratiwi, S S, & Wibowo, 2012). Algoritma ini telah banyak digunakan untuk pencocokan *pattern* yang panjang, namun algoritma ini lebih lambat digunakan pada *pattern* yang pendek.

Metode yang mampu menyelesaikan suatu permasalahan dengan cara berpikir yang sederhana dan tidak membutuhkan suatu pemikiran yang lama serta metode yang menggunakan pergerakan dari kiri ke kanan merupakan metode yang paling baik dan praktiknya sehingga pencarian pola akan lebih cepat (Ardiansah, Wibawa, & Widiyaningtyas, 2016). Algoritma Brute Force, adalah algoritma yang menyelesaikan suatu permasalahan dengan cara berpikir yang sederhana dan tidak membutuhkan suatu pemikiran yang lama (Saragih, 2013). Dalam pencocokan *string*, algoritma Brute Force melakukan pencocokan dari kiri ke kanan, merupakan metode yang paling natural karena sesuai dengan arah membaca. Kekurangan algoritma ini adalah lambat dalam melakukan proses pencarian string yang panjang (Marpaung, 2013). Algoritma Brute Force dapat diterapkan dalam pengembangan aplikasi pencarian data laporan maintenance ini, dikarenakan data laporan maintenance memiliki *pattern* yang pendek serta algoritma Brute Force lebih akurat dalam mendapatkan hasil pencarian.

2. METODE PENELITIAN

Algoritma yang dapat digunakan untuk String Matching salah satunya adalah algoritma Brute Force. Algoritma *brute force* memecahkan masalah dengan sangat sederhana, langsung, dan jelas. Algortima *brute-force* merupakan suatu teknik yang biasa digunakan bila si penyusun algoritma lebih mempertimbangkan memperoleh solusi dari problem secara langsung. Cara kerja yang dilakukan algoritma *brute force* adalah membandingkan karakter demi karakter antar kata atau *string* yang dicari dengan *string* sumber. Apabila tidak sesuai maka akan dilakukan penggeseran posisi dari kiri ke kanan. Demikian seterusnya sampai ditemukan *string* yang dicari (Danuri, 2016).

Secara rinci langkah-langkah yang digunakan algoritma Brute Force untuk mencocokkan *string* adalah sebagai berikut:

1) Pencocokan *pattern* Algoritma Brute Force dimulai dari awal teks.
2) Algoritma Brute Force akan mencocokkan karakter per karakter *pattern* dengan karakter pada teks yang bersesuaian dari kiri ke kanan, sampai salah satu kondisi berikut terpenuhi:
   a. Karakter di *pattern* yang dibandingkan cocok maka pencarian selesai.
   b. Apabila dijumpai ketidak-cocokan antara *pattern* dengan teks, maka pencarian tidak cocok dan belum berhasil.
3) Kemudian algoritma Brute Force akan melakukan penggeseran *pattern* sebesar satu ke kanan dan mengulangi langkah ke-2 sampai *pattern* berada di ujung teks.

Contoh kasus:
Teks (X) : HRONIZATION
Pattern (Y) : SYNCHRONIZATION

Langkah ke-1

X	H	R	O	N	I	Z	A	T	I	O	N				
Y	S	Y	N	C	H	R	O	N	I	Z	A	T	I	O	N

Pada langkah pertama karakter pertama berbeda dengan pattern yang sudah ditentukan, selanjutnya digeser ke sebelah kanan sebanyak satu kali.

Langkah ke-2

X		H	R	O	N	I	Z	A	T	I	O	N			
Y	S	Y	N	C	H	R	O	N	I	Z	A	T	I	O	N

Pada langkah kedua masih belum ditemukan kecocokan karakter pertama pada teks dengan karakter kedua dari *pattern* maka digeser satu kali ke sebelah kanan.

Langkah ke-3

X			H	R	O	N	I	Z	A	T	I	O	N		
Y	S	Y	N	C	H	R	O	N	I	Z	A	T	I	O	N

Pada langkah ketiga masih tidak ditemukan kecocokan antara karakter pertama dalam teks dengan karakter ketiga dari *pattern* maka digeser sebanyak satu kali ke sebelah kanan.

Langkah ke-4

X				H	R	O	N	I	Z	A	T	I	O	N	
Y	S	Y	N	C	H	R	O	N	I	Z	A	T	I	O	N

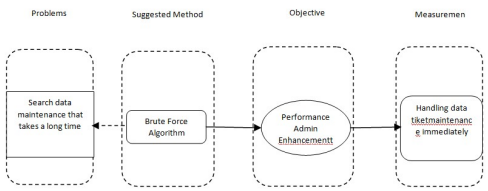
Pada langkah keempat masih belum ditemukan kecocokan antara karakter pertama dalam teks dengan karakter keempat dari *pattern* maka digeser sebanyak satu kali ke sebelah kanan.

Langkah ke-5

X					H	R	O	N	I	Z	A	T	I	O	N
Y	S	Y	N	C	H	R	O	N	I	Z	A	T	I	O	N

Pada langkah terakhir ditemukan kecocokan antara teks dengan *pattern.* Contoh di atas menunjukkan bagaimana algoritma Brute Force berjalan. Algoritma Brute Force akan mencocokkan *string* dari kiri ke kanan apabila tidak cocok maka akan terus digeser ke sebelah kanan sampai *string* yang dicari cocok dengan *pattern.*

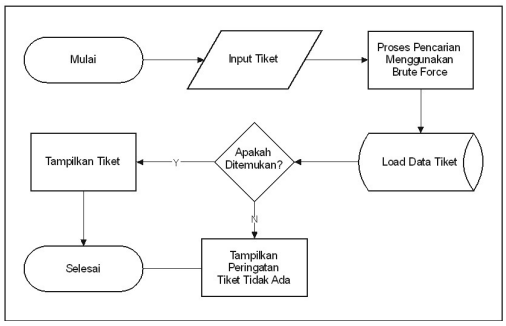
Penelitian ini didasarkan issue penanganan masalah yang lama terutama dalam proses pencarian data maintenance sehingga dibutuhkan suatu terobosan yang dapat memangkas waktu dengan menggunakan Algoritma Brute Force. Penjelasan Kerangka Pemikrannya dituangkan dalam gambar berikut:



Gambar 1. Kerangka Pemikiran

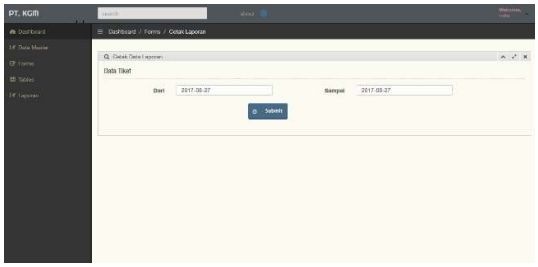
3. PEMBAHASAN

Dalam pembuatan aplikasi pencarian data Maintenance terdapat menu *search.* Untuk itu akan digambarkan alur yang akan terjadi saat pengguna menggunakan menu *search* yang mengimplementasikan algoritma Brute Force.



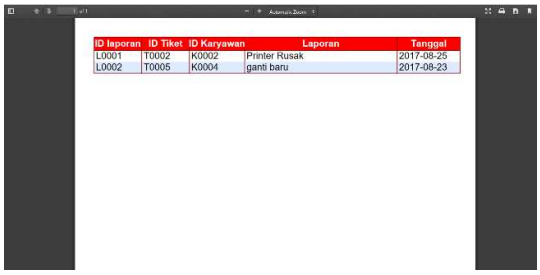
Gambar 2. Alur pencarian tiket maintenance

Pada Gambar 2 dapat ditarik kesimpulan bahwa dalam pencarian tiket maintenance pengguna harus memasukkan id tiket terlebih dahulu. Selanjutnya sistem akan melakukan pencarian dengan bantuan algoritma Brute Force pada *query.* Apabila ditemukan maka sistem akan mengeluarkan tiket maintenance tersebut dan apabila tidak ditemukan maka sistem akan mengeluarkan peringatan bahwa tiket maintenance tidak ditemukan



Gambar 3: Tampilan Form Cetak Laporan

Dalam tampilan form cetak laporan pengguna dapat mencetak laporan berdasarkan periode tanggal.



ID laporan	ID Tiket	ID Karyawan	Laporan	Tanggal
L0001	T0002	K0002	Printer Rusak	2017-08-25
L0002	T0005	K0004	ganti baru	2017-08-23

Gambar 4: Tampilan Hasi lCetak Laporan

Tampilan di atas adalah hasil cetak dari laporan tiket maintenance TIKET, pengguna dapat melakukan cetak laporan berdasarkan periode tanggal yang diinginkan.

4. KESIMPULAN

Algoritma Brute Force yang diterapkan dalam pencarian Data Tiket Maintenance memberikan solusi yang cukup sinifikan dalam pemangkasan waktu penanganan sehingga goal perusahaan untuk meningkatkan performa PIC yang bersangkutan menjadi meningkat dan metode ini dapat menjadi pertimbangan dalam membantu penanganan pencarian data tiket maintenance secara segera.

5. SARAN

Saran – saran yang dapat diberikan dari penelitian ini adalah sebagai berikut:

a. Dapat dibuatkan sistem terpadu antara aplikasi berbasis Algoritma Brute Force dengan database lainnya.
b. Meminimalisisr penggunaakan memery sehingga dapat meningkatkan performa aplikasi

DAFTAR PUSTAKA

[1] Ginting, G. L. (2014). Penerapan Algoritma Boyer-Moore pada Aplikasi Pengajuan Judul Skripsi Berbasi Web. *Jurnal Informasi dan Teknologi Ilmiah (INTI)*, 123-131.

[2] Putri, B. B., Ernawati, & Puspitaningrum, D. (2015). Implementasi Metode WU-MANBER Berdasarkan Multi Pattern Matching dalam Pencarian Kesamaan DNA. *Jurnal rekursif*, 157-170.

[3] Sagita, V., & Prasetiyowati, M. I. (2013). Studi Perbandingan Implementasi Algoritma Boyer-Moore, Turbo Boyer-Moore, dan Tuned Boyer-Moore dalam Pencarian String. *Jurnal ultimatic*, 21-37.

[4] Sugiarti, Y., Nuryasin, & Fitriani, N. (2015). Analisis dan Perancangan Sistem Informasi Rawat Inap. *Jurnal Sistem Informasi*, 1-11.

[5] Anisya. (2013). Aplikasi Sistem Database Rumah Sakit Terpusat pada Rumah Sakit Umum (Rsu) 'Aisyiyah Padang dengan Menerapkan Open Source (PHP – Mysql). *Jurnal Momentum*, 51.

[6] Ardiansah, J. T., Wibawa, A. P., & Widiyaningtyas, T. (2016). Penerapan Algoritma Start End Mid untuk Mendeteksi Kesalahan Logika Structured Query Languange. *Seminar Nasional Teknologi Informasi dan Multimedia, STMIK Amikom, Yogyakarta*, 1-6.

[7] B, D. W., & Hetami, A. (2015). Perancangan Information Retrival untuk Pencarian Ide Pokok Teks Artikel Berbahasa Inggris dengan Pembobotan Vector Space Model. *Jurnal ilmiah teknologi dan informasi ASIA*, 53-59.

[8] Danuri. (2016). Pencarian File Teks Berbasis Content dengan Pencocokan String Menggunakan Algoritma Brute force. *Scientific Journal of Informatics, 3*, 68-75.

[9] I. Y., Aeni , N. H., & Arham, Z. (2011). Rancang Bangun Sistem Informasi Penggajian Karyawan (Studi Kasus: Bank Pembiayaan Rakyat Syariah Harta Insan Karimah). *Studia Informatika*, 3.

[10]Marpaung, I. D. (2013). Implementasi Algoritma String Matching pada Kamus Istilah-istilah Kedokteran Berbasis Android. *Pelita Informatika Budi Darma*, 163-167.

[11] Mujiati, H. (2008). Analisa dan Perancangan Sistem Informasi Stok Obat pada Apotek Arjowinangun. *Indonesian Jurnal on Computer Science*, 1-6.

[12] Palevi, A. R., & Krisnawati. (2013). Analisis dan Perancangan Sistem Informasi Penerimaan Peserta Didik Baru Berbasis Website pada SMP Negeri 2 Mojosongo Boyolali. *Jurnal Ilmiah DASI, 14*, 4.

[13] Pangkerego, A. T., & Pungus, S. R. (2016). Perancangan Aplikasi Laporan Kegiatan Berbasis Web pada BPJN XI Satuan Kerja wilayah II Sulawesi Utara. *STMIK AMIKOM Yogyakarta*, 31.

[14] Pardede, J., Barmawi, M. M., & Pramono, W. D. (2013). Implementasi Metode Generalized Vector Space Model pada Aplikasi Information Retrival. *Jurnal informatika*, 57-67.

[15] Permana, B. D. (2014, 10 16). *Fungsional dari Notepad++*. Dipetik 08 18, 2017, dari bayudwiarta.wordpress.com: https://bayudwiarta.wordpress.com/2014/10/16/fungsional-dari-notepad/

[16] Pratiwi, H. R., S S, D. S., & Wibowo, A. (2012). Prototype Aplikasi SMS Content Filtering Menggunakan Metode String Matching. *Jurnal Teknik Informatika*, 1-8.

[17] Santoso, B. W., Sundawa, F., & Muhammad , A. (2016). Implementasi Algoritma Brute Force Sebagai Mesin Pencari (Search Engine) Berbasis Web Pada Datbase. *JURNAL SISFOTEK GLOBAL, 6*, 1-8.

[18] Saragih, M. A. (2013). Implementasi Algoritma Brute Force Dalam Pencocokan Teks Font Italic untuk Kata Berbahasa Inggris Pada Dokumen Microsoft Office

Word. *Pelita Informatika Budi Darma*, 84-87.

[19] Syahputra, W. (2015). Implementasi Algoritma Brute Force Pada Perancangan Aplikasi Pengecekan Struktur Kata Serapan Bahasa Indonesia. *Pelita Informatika Budi Darma, IX*, 161-165.

[20] Syarif, M. (2017). Implementasi Algoritma String Matching Dalam Pencarian Surah Dan Ayat Dalam Al-Quran Berbasis Web. *Indonesian Journal on Networking and Security*, 71-72.

[21] Utomo, D., Harjo, E. W., & Handoko. (2008). Perbandingan Algoritma String Searching Brute Force, Knuth Morris Pratt, Boyer Moore dan Karp Rabin pada Teks Alkitab Bahasa Indonesia. *Jurnal Techne Jurnal Ilmiah Elektronika SKSW*, 1-13.