JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2262
Hal 455-458

Implementasi Algoritma Raita Pada Aplikasi Hukum Kanonik Berbasis Mobile

Efidoren L Nainggolan, Muhammad Syahrizal, Saidi Ramadan Siregar

Prodi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: [1]epidoren23@gmail.com, [2]syahrizal83.budidarma@gmail.com, [3]saidiramadan@stmik-budidarma.ac.id
Email Penulis Korespondensi: epidoren23@gmail.com
Submitted 05-06-2020; Accepted 13-06-2020; Published 14-06-2020

Abstrak

Hukum kanonik adalah hukum gereja internal yang mengatur gereja katolik, gereja ortodoks timur,gereja ortodoks timur, komuni anglikan. Bagaimana hukum gereja itu diatur, diinterpretasikan dan kadang-kadang ditelaah berbeda secara mendasar di antara ketiga tubuh gereja tersebut. di dalam ketiga tradisi, sebuah kanon mulanya adalah sebuah aturan yang diterima oleh sebuah majelis, kanon-kanon ini membentuk dasar bagi hukum kanonik. Algoritma raita merupakan bagian dari algoritma exact string matching yaitu pencocokan string secara tepat dengan susunan karakter dalam string yang dicocokkan memiliki jumlah maupun urutan karakter dalam string yang sama. Pencocokan string pada algoritma raita yang dilakukan melalui pergeseran dari kanan karakter kemudian ke kiri karakter dan ke tengah karakter. Adapun masalah pada penelitian ini adalah isi hukum kanonik pada umumnya terdiri dari jumlah halaman buku yang sangat banyak, hal ini membuat para pengguna hukum kanonik kesulitan dalam mencari isi yang diperlukan maka dalam pencarian tersebut memerlukan waktu untuk menemukan isi hukum kanonik yang dicari terlalu banyak masalah pencarian yaitu terlalu banyaknya waktu yang harus dibutuhkan untuk menemukan isi hukum kanonik yang cari.

Kata Kunci: Hukum kanonik,Algoritma Raita,Pencarian Kata.

Abstract

Canonical law is an internal church law governing the Catholic Church, Eastern Orthodox Church, Eastern Orthodox Church, Anglican Communion. How the laws of the church are governed, interpreted and sometimes examined differ fundamentally between the three church bodies. in all three traditions, a canon was originally a rule accepted by an assembly, these canons formed the basis for canon law. Raita algorithm is part of the exact string matching algorithm, which is matching the string exactly with the arrangement of characters in the matched string that has the same number or sequence of characters in the string. Matching strings on the raita algorithm is done through a shift from the right of the character then to the left of the character and to the middle of the character. The problem in this research is the content of canon law in general consists of a very large number of pages of books, this makes it difficult for canonical law users to find the contents needed, then in the search it takes time to find the contents of canonical law that are searched for too many search problems. that is, too much time must be needed to find the contents of the canonical law sought.

Keywords: Canonical Law, Raita Algorithm, Word Search.

1. PENDAHULUAN

Pencarian merupakan pencarian informasi dalam suatu aplikasi, dengan suatu kunci *(key)* pencarian diperlukan untuk mencari informasi khusus dari table pada saat lokasi yang pasti dari informasi tersebut sebelumnya tidak diketahui. Pencarian selalu dinyatakan dengan referensi pada adanya sekelompok data yang tersimpan secara terorganisasi. Hukum kanonik adalah hukum gerejawi internal yang mengatur gereja ortodoks timur,gereja ortodoks timur, komuni anglikan. Bagaimana hukum gereja itu diatur, diinterpretasikan dan kadang-kadang ditelaah berbeda secara mendasar di antara ketiga tubuh gereja tersebut. di dalam ketiga tradisi, sebuah kanon mulanya adalah sebuah aturan yang diterima oleh sebuah majelis, kanon-kanon ini membentuk dasar bagi hukum kanonik. Hukum Kanonik sebagian besar terbagi dalam paragraf-paragraf dan nomor. Oleh karenanya sebuah kutipan dari kitab ini biasa ditulis dengan Kan atau Kanon [1].

Adapun masalah pada penelitian ini adalah isi hukum *kanonik* pada umumnya terdiri dari jumlah halaman buku yang sangat banyak, hal ini membuat para pengguna hukum *kanonik* kesulitan dalam mencari isi yang diperlukan maka dalam pencarian tersebut memerlukan waktu untuk menemukan isi hukum kanonik yang dicari terlalu banyak masalah pencarian yaitu terlalu banyaknya waktu yang harus dibutuhkan untuk menemukan isi hukum *kanonik* yang cari.

Algoritma *raita* merupakan bagian dari algoritma *exact string matching* yaitu pencocokan *string* secara tepat dengan susunan karakter dalam *string* yang dicocokkan memiliki jumlah maupun urutan karakter dalam *string* yang sama. Pencocokan *string* pada algoritma raita yang dilakukan melalui pergeseran dari kanan karakter kemudian ke kiri karakter dan ke tengah karakter. Penelitian terdahulu dilakukan oleh Dedi Rudi Bawanto dan Nidia Rosmawati menyimpulkan dari 8 teknik pengujian yaitu match case, part case, 32 record data, 64 record data, 96 record data, tipe data string, tipe data numeric dan tipe data date dengan 5 kali simulasi dari setiap kata kunci pada teknik pengujian, algoritma raita lebih cepat 3% dari algoritma binary search dalam pencarian data dari keseluruhan data sampel yang diuji dengan total waktu 2.5116 ms [2].

2. METODE PENELITIAN

2.1 Hukum Kanonik

Hukum kanonik adalah hukum gereja internal yang mengatur gereja katolik, gereja ortodoks timur,gereja ortodoks timur, komuni anglikan. Bagaimana hukum gereja itu diatur, diinterpretasikan dan kadang-kadang ditelaah berbeda secara mendasar di antara ketiga tubuh gereja

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2262
Hal 455-458

tersebut. di dalam ketiga tradisi, sebuah kanon mulanya adalah sebuah aturan yang diterima oleh sebuah majelis, kanon-kanon ini membentuk dasar bagi hukum kanonik [3].

2.2 *String Matching*

String Matching adalah proses pencarian semua kemunculan *query* yang selanjutnya disebut *pattern* ke dalam *string* yang lebih panjang atau teks. String Matching dirumuskan sebagai berikut :
x = x[0…m-1]
y = y[0…n-1]
Dimana :
x adalah *pattern*
m adalah panjang *pattern*
y adalah teks
n adalah panjang teks

Kedua *string* terdiri dari sekumpulan karakter yang disebut alfabet yang dilambangkan dengan ∑ (sigma) dan mempunyai ukuran σ (tao). *String*
matching dibagi menjadi dua, yakni *exact matching* dan *heuristic* atau *statistical matching*. *Exact Matching* digunakan untuk menemukan pattern yang berasal dari satu teks. Contoh pencarian *exact matching* adalah pencarian kata "pelajar" dalam kalimat "saya seorang pelajar" atau "saya seorang siswa". Sistem akan memberikan hasil bahwa kalimat pertama mengandung kata "pelajar" sedangkan
kalimat kedua tidak, meskipun kenyataannya pelajar dan siswa adalah kata yang bersinonim. Algoritma *exact matching* diklasifikasi menjadi tiga bagian menurut arah pencariannya, yaitu :

1. Arah pembacaan dari kiri ke kanan.
   Algoritma yang termasuk kategori ini adalah *Brute Force*, Morris dan Pratt (yang kemudian dikembangkan oleh Knuth, Morris, dan Pratt).
2. Arah pembacaan dari kanan ke kiri.
   Algoritma yang termasuk kategori ini adalah *Boyer* dan *Moore* yang kemudian dikembangkan menjadi algoritma *turbo Boyer-Moore, tuned Boyer-Moore, dan Zhu-Takaoka*.
3. Arah pencarian yang ditentukan program. Algoritma yang termasuk kategori ini adalah algoritma *Colussi*, *Crochemore-Perrin*.
   Heuristic matching adalah teknik yang digunakan untuk menghubukangkan dua data terpisah ketika *exact matching* tidak mampu mengatasi karena pembatasan pada daat yang tersedia. *Heuristic matching* dapat dilakukan dengan perhitungan *distance* antara *pattern* dengan teks. *Exact* dan *heuristic matching* memiliki kelemahan dalam menemukan kata yang memiliki kemiripan makna tetapi berbeda tulisan [4].

2.3 Algoritma *Raita*

Raita merupakan bagian dari algoritma *exact string matching* yaitu pencocokan *string* secara tepat dengan susunan karakter dalam *string* yang dicocokkan memiliki jumlah maupun urutan karakter dalam *string* yang sama. Raita merancang sebuah algoritma dengan membandingkan karakter yang terakhir dari pola yang diawali dari karakter paling kanan dari "jendela". Jika mereka cocok, kemudian karakter pertama dari pola teks paling kiri dari jendela juga dibandingkan. Jika mereka cocok, maka akan dibandingkan karakter tengah pola dengan karakter teks tengah jendela. Pada akhirnya, jika mereka benar-benar cocok, maka algoritma membandingkan karakter lain mulai dari pola karakter kedua ke karakter kedua terakhir, dan akan membandingkan dengan karakter tengah lagi [5].

Langkah *Raita* dalam melakukan pencocokan *string* terdiri dari 5 (lima) tahapan. Berikut tahapan langkah – langkah *Raita* dalam melakukan pencocokan *string* :
Buat tabel pergeseran pola yang dicari sebagai kata yang akan dicari pada teks.

1 jika dalam proses pembandingan terjadi ketidakcocokan antara pasangan karakter pada akhir pola dengan karakter teks, pergeseran dilakukan sesuai nilai karakter teks pada tabel *BmBc*
2 Jika dalam proses pembandingan akhir pola terjadi ketidakcocokan lagi maka karakter akan digeser lagi sesuai tabel *BmBc*
3 Jika karakter akhir pola dengan karakter pada teks yang sedang dibandingkan cocok, maka posisi karakter pada pola dan teks akan memiliki nilai nol (0), dan dilanjutkan pencocokan pada karakter awal pola. Jika cocok maka dilanjutkan pencocokan dengan karakter tengah pola.
4 Jika akhir, awal dan tengah pola telah cocok. Pencocokan dilanjutkan dengan bagian kanan dari awal karakter pada pola, jika cocok maka dicocokkan pada bagian kanan tengah pola [5].

3. HASIL DAN PEMBAHASAN

Analisa sistem merupakan tahapan yang dilakukan oleh penulis untuk memberikan pemahaman secara detail terhadap kebutuhan-kebutuhan sistem dan menggambarkan proses-proses yang ada di dalam sistem tersebut untuk menghasilkan keluaran atau *output* yang sesuai dengan kebutuhan *user* atau pengguna. Analisa sistem yang akurat merupakan salah satu tahapan yang penting dilakukan karena kesalahan dalam tahap ini akan menyebabkan kesalahan di tahap selanjutnya.

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2262
Hal 455-458

Implementasi algoritma *raita* pada aplikasi hukum *kanonik* berbasis *mobile* yang dibangun menggunakan editor *eclipse juno* dan *database SQLite* pada penelitian ini bertujuan untuk menyelesaikan masalah yang terjadi pada pencarian deskripsi daripada isi hukum *kanonik* agar dapat mempermudah *user* menemukan hasil pencarian dalam waktu yang cepat dan tepat.

Algoritma *raita* merupakan bagian dari algoritma *exact string matching* yaitu pencocokan *string* secara tepat dengan susunan karakter dalam *string* yang dicocokkan memiliki jumlah maupun urutan karakter dalam *string* yang sama. Pencocokan *string* pada algoritma raita yang dilakukan melalui pergeseran dari arah kanan karakter kemudian ke arah kiri karakter dan ke arah tengah karakter dapat menemukan pencocokan karakter dengan efektif dan efesien. Langkah – langkah dalam pencarian algoritma Raita :

1. Buat tabel pergeseran pola yang dicari sebagai kata yang akan dicari pada teks.
2. Jika dalam proses pembandingan terjadi ketidakcocokan antara pasangan karakter pada akhir pola dengan karakter teks, pergeseran dilakukan sesuai nilai karakter pada tabel BmBc.
3. Jika dalam proses pembandingan akhir pola terjadi ketidakcocokan lagi maka karakter akan digeser lagi sesuai tabel BmBc.
4. Jika karakter akhir pola dengan karakter pada teks yang sedang dibandingkan cocok, maka posisi karakter pada pola dan teks akan memiliki nilai (0), dan dilanjutkan pencocokan pada karakter awal pola. Jika cocok maka dilanjutkan pencocokan dengan karakter tengah pola.
5. Jika akhir, awal dan tengah pola telah cocok. Pencocokan dilanjutkan dengan bagian kanan dari awal karakter pada pola, jika cocok maka dicocokkan pada bagian kanan tengah pola. Sebagai contoh perhitungan algoritma Raita akan diberikan teks dan pattern berikut Teks.

Kelebihan daripada aplikasi hukum kanonik berbasis *android* dengan menerapkan algoritma *raita* yang dibangun oleh penulis pada penelitian ini yaitu dapat dioperasikan dengan mudah oleh *user* atau pengguna aplikasi hukum kanonik berbasis *android* pada *smartphone* yang didukung sistem operasi *android* secara *offline* dimana dan kapanpun.

Algoritma *raita* pada penelitian ini penulis melakukan pencarian pada aplikasi hukum *kanonik* berbasis *mobile*, kasus pada penelitian ini penulis mengimplementasikan algoritma *raita* untuk pencocokan *pattern* "PAUS" dengan text "KETAATAN KEPADA PAUS". Maka diketahui pula perhitungannya sebagai berikut :

Text = KETAATAN KEPADA PAUS
Pattern = PAUS
diketahui bahwa :
Text = PAUS (Text yang di cari)
Pattern = 4 (Panjang pola)
Untuk melakukan perhitungan maka dibuat tabel BmBc dengan persamaan sebagai berikut:
m-2
Berfungsi sebagai batas pencarian karakter pada pola
m-i-1
Berfungsi sebagai pencarian nilai karakter pada tabel BmBc
P A U S
0 1 2 3

Tabel 1. Tabel BmBc

×	BC	BM
0	**P**	**3**
1	**A**	**2**
2	**U**	**1**
3	*	**4**

Berdasarkan tabel 1. Dapat diketahui perhitungan tabel BmBc dengan persamaan :
m-2
= 4-2
=2
Maka i = 0-2
Selanjutnya untuk mencari niai geser pada tabel BmBc digunakan rumus m-i-1
BM[BC[P]=4-0-1=3
BM[BC[A]=4-1-1=2
BM[BC[U]=4-2-1=1
(*) karakter yang tidak dikanali.
Panjang pola pada huruf S di atas adalah sebesar 4.Maka untuk karakter yang tidak ada pada tabel diinsealisasikan dengan tanda(*) yang nilainya sesuai dengan panjang pola
Langkah selanjutnya adalah melakukan pencocokan karakter menggunakan algoritma *raita* dengan tahap-tahap berikut ini :

1. Tahap pertama, yaitu mencocokan karakter S dengan A.

Karakter	K	E	T	A	A	T	A	N		K	E	P	A	D	A		P	A	U	S
	P	A	U	S																

Karena karakter S dengan A tidak sama, maka dilakukan persegesaran sebesar nilai *BmBc* A yaitu sebanyak 2 (dua) langkah.

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2262
Hal 455-458

2. Tahap kedua, yaitu mencocokkan karakter S dengan T

Karakter	K	E	T	A	A	T	A	N		K	E	P	A	D	A		P	A	U	S
			P	A	U	S														

Karena karakter S dengan T tidak sama, maka dilakukan persegesaran sesuai jumlah karakter *pattern*, yaitu sebanyak 4 (empat) langkah.

3. Tahap ketiga, yaitu mencocokkan karakter S dengan K

Karakter	K	E	T	A	A	T	A	N		K	E	P	A	D	A		P	A	U	S
							P	A	U	S										

Karena karakter S dengan K tidak sama, maka dilakukan persegesaran sesuai jumlah karakter *pattern*, yaitu sebanyak 4 (empat) langkah.

4. Tahap keempat, yaitu mencocokkan karakter S dengan D

Karakter	K	E	T	A	A	T	A	N		K	E	P	A	D	A		P	A	U	S
											P	A	U	S						

Karena karakter S dengan D tidak sama, maka dilakukan persegesaran sesuai jumlah karakter *pattern*, yaitu sebanyak 4 (empat) langkah.

5. Tahap kelima, yaitu mencocokkan karakter S dengan A

Karakter	K	E	T	A	A	T	A	N		K	E	P	A	D	A		P	A	U	S
															P	A	U	S		

Karena karakter S dengan A tidak sama, maka dilakukan persegesaran sebesar nilai *BmBc* A yaitu sebanyak 2 (dua) langkah.

6. Tahap keenam, yaitu mencocokkan karakter S dengan A

Karakter	K	E	T	A	A	T	A	N		K	E	P	A	D	A		P	A	U	S
															P	A	U	S		

Dapat dilihat pada pencocokan karakter S dengan S, P dengan P, U dengan U, A dengan A adalah sama, Maka pencocokan karakter berhenti. Maka poses pencarian hukum *kanonik* dengan algoritma rita pada kata KETAATAN KEPADA PAUS dengan *pattern* PAUS dapat ditemukan dengan proses 6 pencocokan data.

4. KESIMPULAN

Adapun kesimpulan yang penulis peroleh berdasarkan hasil dari skripsi ini adalah sebagai berikut :

1. Proses pencarian isi hukum kanonik dengan menggunakan algoritma raita, sangat membantu pengguna dalam mencari isi hukum kanonik karena algoritma raita memiliki proses pencarian yang efektif dan efisien. Aplikasi hukum kanonik berbasis mobile dibangun menggunakan editor eclipse juno dan dapat dioperasikan pada smartphone android.
2. Perancangan aplikasi hukum kanonik berbasis mobile dengan menerapkan algoritma raita dibangun pada penelitian dapat menjadi solusi untuk mempelajari isi hukum kanonik.

REFERENCES

[1] Bawanto Rudi Dedi and Rosmawanti Nidia."Perbandingan Algoritma Binary Search Dan Raita Dalam Pecarian Data."pp1311-1448,2017.
[2] Charras C.&Lecroq T,Handbook of Exact String-matching Algorithms.London:King's College Publications,2004
[3] A. Azhar, N. Marbun, S. Aripin dan E. Buulolo, “Implementasi Algoritma Horspool Pada Aplikasi Istilah Fashion,” KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer), vol. 3, no. 1, 2019.
[4] Implementasi Dan Perbandingan Algoritma Berry-Ravindran Dan Algoritma ZhuTakaoka Pada Aplikasi Kamus Bahasa Indonesia-Batak-Toba,:
[5] Algoritma Dan Pemograman,USU press,Medan,2007 Muhammad Zarlis dan Handrizal,.
[6] Implementasi Algoritma Brute Force Dalam Pencarian Data Katalog Buku Perpustakaan Mesran,.
[7] Charras C.&Lecroq T,Handbook of Exact String-matching Algorithms.London:King's College Publications,2004.