Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

Implementasi Algoritma Zhu-Takaoka pada Aplikasi Pencarian Makanan Khas di Dunia Berbasis Android

Ardiansyah Reza Saragih

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia

Abstrak– Salah satu hal yang tidak boleh dilewatkan oleh seorang *traveler* adalah mencicipi makanan khas dari negara yang sedang dikunjungi. Tetapi, mereka masih belum tahu makanan khas apa saja yang ada di negara tersebut. Bisa saja menggunakan layanan *Google Search* untuk mencari informasi dari makanan khas tersebut, akan tetapi membutuhkan koneksi internet. Masalahnya, mungkin saja negara yang sedang dikunjungi sulit untuk mendapatkan koneksi internet, atau traveler tersebut kehabisan dan lupa mengisi kuota internet *smartphone* miliknya, atau mungkin juga operator seluler smartphone miliknya memang belum bisa dipakai di luar negeri. Jika membeli sim card baru yang sesuai dengan negara yang dikungi tersebut, tentu itu merepotkan. Karena setiap berpindah negara harus membeli sim card baru yang sesuai dengan negara yang dikunjungi. Untuk mengatasi masalah tersebut dibutuhkan sebuah aplikasi sederhana yang dioperasikan pada smartphone berbasis Android yang berguna membantu kita mencari makanan khas dari berbagai negara. Aplikasi ini sangat sederhana dan bersifat *offline*, yang artinya tidak memerlukan koneksi intenet. Pengguna hanya tinggal memasukan nama negara kemudian akan muncul makanan-makanan khas apa saja yang ada pada negara tersebut. Aplikasi yang dirancangan menggunakan salah satu algoritma pencarian, yaitu *Zhu-Takaoka*. Algoritma *Zhu-Takaoka* adalah perkembangan dari algoritma *Boyer-Moore*.

Kata Kunci: Android, Makanan Khas, Zhu-Takaoka

Abstract– One thing that should not be missed by a traveler is tasting the typical food of the country that is being visited. However, they still don't know what special foods are in the country. You can use the Google Search service to find information on these special foods, but you need an internet connection. The problem is, maybe the country that is being visited is difficult to get an internet connection, or the traveler runs out and forgets to fill out his smartphone's internet quota, or maybe his smartphone cellular operator can't be used abroad. If you buy a new sim card that is suitable for the country you visit, of course it is troublesome. Because every time you change countries you have to buy a new sim card that suits the country you are visiting. To overcome this problem, a simple application that is operated on an Android-based smartphone that is useful helps us find special foods from various countries. This application is very simple and is offline, which means it does not require an internet connection. Users only need to enter the name of the country and what foods will appear in that country. Application designed using one of the search algorithms, namely Zhu-Takaoka. The Zhu-Takaoka algorithm is a development of the Boyer-Moore algorithm.

Keywords: Android, Typical Food, Zhu-Takaoka

1. PENDAHULUAN

Bagi sebagian orang, saat ini *smartphone* sudah menjadi salah satu kebutuhan pokok. Hampir semua orang sudah pernah menggunakan *smartphone* mulai dari anak kecil hingga orang dewasa. Seiring pesatnya perkembangan *smartphone* saat inilah, banyak aplikasi-aplikasi *smartphone* yang bermunculan untuk melengkapi kecanggihan *smartphone* tersebut. Aplikasi-aplikasi yang ada pada *smartphone* bisa dimanfaatkan untuk membantu dan mempercepat pekerjaan kita. Misalkan saja aplikasi kalkulator untuk membantu melakukan perhitungan, aplikasi kamus untuk menerjemahkan bahasa tertentu. Bahkan saat ini sudah ada aplikasi pemesanan tiket yang dapat kita pesan hanya menggunakan *smartphone* yang kita miliki.

Setiap negara di dunia ini pasti memiliki ciri khas masing-masing yang melambangkan negara tersebut. Sama halnya dengan hal kuliner, setiap negara memiliki makanan. Makanan khas adalah makanan yang biasa dikonsumsi oleh masyarakat yang hanya ada di negara tersebut dan tidak ada di tempat lain. Biasanya makanan khas dibuat dari bahan-bahan yang hanya ada dari negara tersebut serta memiliki cita rasa dan keunikannya masing-masing [1].

Salah satu hal yang tidak boleh dilewatkan oleh seorang *traveler* adalah mencicipi makanan khas dari negara yang sedang dikunjungi. Tetapi, mereka masih belum tahu informasi tentang makanan khas apa saja yang ada di negara tersebut. Untuk mengatasi masalah tersebut dibutuhkan sebuah aplikasi sederhana yang dioperasikan pada *smartphone* berbasis Android yang berguna membantu kita mencari makanan khas dari berbagai negara. Aplikasi ini sangat sederhana dan bersifat *offline*, yang artinya tidak memerlukan koneksi intenet. Pengguna hanya tinggal memasukan nama negara kemudian akan muncul makanan-makanan khas apa saja yang ada pada negara tersebut. Aplikasi ini nantinya akan menggunakan algoritma pencarian.

Algoritma pencarian adalah algoritma untuk melakukan pencarian sebuah *string* yang lebih pendek atau disebut *pattern* dari *string* yang lebih panjang atau disebut *text*. Ada beberapa algoritma yang dapat digunakan untuk melakukan pencarian, salah satunya adalah algoritma *Zhu-Takaoka*. Algoritma *Zhu-Takaoka* ditemukan oleh R.F.Zhu dan T.Takaoka pada tahun 1989. Algoritma ini menggunakan dua *text* karakter untuk menghitung pergeseran karakter berdasarkan *bad-character*. Untuk mencari ketidakcocokan atau mencari seluruh *text* menggunakan *preprocessing hashing*. Hal ini efektif untuk pencarian *string* dua dimensi [2].

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

Berdasarkan penelitian yang berjudul, Implementasi Algoritma *Zhu-Takaoka* Pada Aplikasi Kamus Istilah Musik Berbasis *Android* yang ditulis oleh, Gutman Togatorop, Aan Erlansari, dan Funny Farady Coastera, dalam penelitian tersebut algortima *Zhu-Takaoka* berhasil melakukan proses pencarian pada istilah-istilah musik.

2. METODE PENELITIAN

2.1 Makanan Khas

Makanan khas adalah makanan asli hasil dari olahan dari suatu tempat tertentu yang biasanya dikonsumsi oleh masyarakat di tempat tersebut. Biasanya makanan khas ini dibuat dari bahan-bahan yang ada di tempat tersebut dan memiliki rasa yang unik, melambangkan tempat dimana makanan khas tersebut berasal [1].

2.2 Algoritma

Algoritma adalah suatu prosedur yang jelas untuk menyelesaikan suatu persoalan dengan menggunakan langkah-langkah tertentu dan terbatas jumlahnya. Istilah algoritma pertama kali ditemukan oleh seorang ahli dalam bidang matematika yang berkebangsaan Arab bernama Abu Ja'far Muhammad Ibnu Musa Al-Kwarizmi. Yang dikatakan benar dari sebuah algoritma adalah, jika algoritma tersebut berhasil mengeluarkan *output* yang benar dari semua kemungkinan *input*. Algoritma sering memiliki langkah pengulangan (iterasi) atau memerlukan keputusan (logika *boolean* dan perbandingan) sampai tugasnya selesai. Biasanya juga sebuah algoritma memiliki sifat bisa dihitung (*computable*) atau bisa diukur (*measurable*) [3].

2.3 Algoritma Zhu-Takaoka

Algoritma *Zhu-Takaoka* merupakan algoritma hasil modifikasi dari algoritma pencocokan *string* algoritma *Boyer-Moore* yang diciptakan oleh Boyer R.S dan Moore J.S pada tahun 1977. Algoritma *Zhu-Takaoka* kemudian dikembangkan oleh Zhu Rui Feng dan Tadao Takaoka yang dipublikasikan pada tahun 1986. Algoritma *Zhu-Takaoka* memiliki ciri-ciri yang sama dengan algoritma *Boyer-Moor*e dalam proses pencarian *string*, yaitu adanya *preprocessing*, *right-to-left scan*, *bad character rule*, dan *good suffixes rule*. Walaupun memiliki persamaan dalam proses pencarian *string*, dua algoritma ini tetap memiliki perbedaan, yaitu terletak pada tahap penentuan *bad character rule*. Dimana pada algoritma *Boyer-Moore*, *bad character* hanya terdiri *array* satu dimensi, sedangkan pada algoritma *Zhu-Takaoka* dimodifikasi menjadi *array* dua dimensi.

Pada tahap *preprocessing* algoritma *Zhu-Takaoka* membangun tabel *bad character* 2 dimensi karena algoritma tersebut menghitung untuk pasangan karakter. Kompleksitas waktu dari tahap *preprocessing* adalah O(m+o2) dan kompleksitas waktu fase pencarian adalah O(mn). Proses inti pencarian algoritma *Zhu-Takaoka* yaitu dilakukan dengan teknik *right-to-left scan rule*. Teknik ini membandingkan *pattern* yang dicari dengan sumber text dimulai dari kanan ke kiri [2].

Menjalankan prosedur *preZTBc* (*Preprocessing Zhu-Takaoka Bad Character*) dan *preBmGs* (*Preprocessing Boyer-Moore Good Suffixes*) untuk mendapatkan inisialisasi.

1. Menjalankan prosedur *preZTBc*. Fungsi dari prosedur ini adalah untuk menentukan berapa besar pergeseran yang dibutuhkan untuk mencapai karakter tertentu pada pattern dari dua karakter pattern terakhir/terkanan. Hasil dari prosedur *preZTBc* disimpan pada tabel *ztBc*.
2. Menjalankan prosedur *preBmGs*. Sebelum menjalankan isi prosedur ini, prosedur *suffix* dijalankan terlebih dulu pada *pattern*. Fungsi dari prosedur *suffix* adalah memeriksa kecocokan sejumlah karakter yang dimulai dari karakter terakhir/terkanan dengan sejumlah karakter yang dimulai dari setiap karakter yang lebih kiri dari karakter terkanan tadi. Hasil dari prosedur *suffix* disimpan pada tabel *suff*. Jadi *suff*[i] mencatat panjang dari *suffix* yang cocok dengan segmen dari *pattern* yang diakhiri karakter ke-i.
3. Dengan prosedur *preBmGs*, dapat diketahui berapa banyak langkah pada *pattern* dari sebeuah segmen ke segmen lain yang sama yang letaknya lebih kiri dengan karakter di sebelah kiri segmen yang berbeda. Prosedur *preBmGs* menggunakan tabel *suff* untuk mengetahui semua pasangan segmen yang sama. Hasil dari prosedur *preBmGs* disimpan pada tabel *bmGs*. Kemudian dilakukan proses pencarian string dengan menggunakan hasil dari prosedur *preBmGc* dan *preBmGs*, yaitu tabel *bmBc* dan *BmGs*.

3. ANALISA DAN PEMBAHASAN

Pencarian makanan khas di dunia menggunakan salah satu algoritma pencarian, yaitu *Zhu-Takaoka*. Dimana algoritma akan melakukan proses pencarian semua karakter pada *text* berdasarkan *pattern* yang sudah ditentukan. Pada algoritma *Zhu-Takaoka*, ketika *pattern* yang dicari sudah ditemukan, proses pencarian akan terus dilakukan jika panjang karakter pada *text* masih memungkinan.

Pada proses pencarian makanan khas di dunia ini, menentukan terlebih dahulu *pattern* atau kata kunci yang akan dicari. Kemudian algortima akan melakukan proses pencocokan pada tiap karakter *text* untuk menemukan *pattern* yang dicari. Setelah selesai, akan muncul hasil ditemukan atau tidaknya *pattern* yang dicari. Jika ditemukan, kemungkinan akan ada beberapa pilihan nama negara yang cocok dengan *pattern* atau kata kunci yang

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

dicari. Kemudian memilih salah satu negara, maka akan muncul daftar beberapa makanan khas yang ada di negara tersebut.

Aplikasi pencarian makanan khas dirancang menggunakan algoritma pencarian yaitu algoritma *Zhu-Takaoka* sebagai solusi dalam melakukan pencarian terhadap *pattern* yang dijadikan sebagai acuan untuk melakukan pencarian karakter yang sesuai dengan *pattern* tersebut. Dalam algoritma pencarian secara umum dirumuskan sebagai berikut:

1. Teks (*text*), yaitu sebuah *long string* yang panjang n karakter.
2. *Pattern*, yaitu sebuah *string* dengan panjang m karekater (m<n) yang akan dicari dalam *text*.

Dalam algoritma pencocokan *string*, teks diasumsikan berada dalam memori sehingga bila ingin mencari *string* di dalam sebuah arsip, maka semua isi arsip perlu dibaca terlebih dahulu kemudian disimpan di dalam memori. Jika *pattern* muncul lebih dari sekali di dalam teks, maka pencarian hanya akan memberikan keluaran berupa lokasi *pattern* ditemukan pertama kali.

Berikut ini adalah langkah-langkah atau cara kerja dari algortima *Zhu-Takaoka* dalam contoh kasus melakukan pencarian *pattern* pada *text* dengan contoh kasus seperti berikut.

Tabel 1. Data Sampel Makanan Khas

No.	Nama Negara	Makanan Khas
1.	Indonesia	Karedok, Trancam, Sate Sarepeh
2.	Malaysia	Char Kuey Tiaw, Nasi Lemak, Ikan Asam
3.	Singapura	Ayam Buah Kluwak, Mi Hokkian, Popiah
4.	Vietnam	Pho Xao, Beef Pho, Ca Kho To
5.	Thailand	Tom Kha Gai, Yum Nua, Som Tam
6.	Filipina	Bulalo, Lumpiang Ubod, Kare-Kare
7.	Korea	Bulgogi, Miyeok Guk, Kimbap
8.	Jepang	Takoyaki, Tempura Handmaki, Miso Soup
9.	India	Kedgeree, Egg Paratha Aloo Gobi
10.	Tiongkok	Capcay, Fuyunghai, Pek Cam Kee

Langkah 1: Membuat Tabel *ztBc*

Untuk memudahkan proses mencari tabel *ztBc* langkah awal adalah dengan menemukan tabel *bmBc*, yaitu dengan mengidentifikasi setiap perpindahan.

Tabel 2. *bmBc* Awal

Index	0	1	2
Karakter	S	I	A
Nilai OH	?	?	?

Keterangan:

1. *Index* adalah indeks dari *pattern*.
2. Karakter adalah *pattern* yang akan dicari.
3. Nilai *Occurrence Heuristic* (OH) adalah nilai pergeseran *Bad Character*.

Untuk mengisi nilai OH, melakukan pencacahan di setiap karakter pada *pattern*. Pencacahan dimulai dari karakter ke-2 paling kanan (*Index* = *Length* (*pattern*) - 2) dari *pattern*. Untuk langkah pertama perpindahan diisi dengan angka bernilai 1.

Tabel 3. Mencari Nilai OH (Pergeseran ke-1)

Index = *Length* (*pattern*) - 2 = 3 - 2 = 1			
Index	0	1	2
Karakter	S	I	A
Pindah = 1			

Bandingkan setiap karakter yang dicacah terhadap karakter yang terdapat pada tabel *bmBc* sebelumnya. Jika karakter yang dicacah tidak ditemukan dalam tabel tersebut, maka tambahkan karakter tersebut *bmBc* dan nilai OH diisi dengan jumlah pindah karakter yang sedang dibandingkan. Untuk perbandingan pertama, bandingkan karakter pertama dengan nilai *null*.

Tabel 4. Mencari Nilai OH (Perbandingan ke-1)

Bandingkan I dengan *null*		Hasil	
bmBc (awal)		*bmBc* (akhir)	
Karakter	*Null*	Karakter	I
Nilai OH	*Null*	Nilai OH	1

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

Pada pencacahan karakter selanjutnya, dilakukan pada *index* sebelumnya dikurangi 1, yaitu pada *index* 0. Dan jumlah perpindahan diubah menjadi 6.

Tabel 5. Mencari Nilai OH (Pergeseran ke-2)

Index = 1 - 1 = 0			
Index	0	1	2
Karakter	S	I	A
Pindah = 2			

Pada perbandingan kedua, bandingkan karakter S yang ada pada *index* 0 dengan karakter I yang berada pada *index* sebelumnya. Karena karakter S tidak ditemukan, maka karakter S dimasukan kedalam tabel *bmBc* dengan nilai *OH* sejumlah perpindahan, yaitu 2.

Tabel 6. Mencari Nilai OH (Perbandingan ke-2)

Bandingkan S dengan I		Hasil		
bmBc (awal)		*bmBc* (akhir)		
Karakter	I	Karakter	I	S
Nilai OH	1	Nilai OH	1	2

Pencacahan selanjutnya merupakan proses pencacahan terakhir, karena *index* sebelumnya adalah 0, oleh sebab itu pencacahan dilakukan pada *index* paling kanan dikurang dengan 1, yaitu pada *index* 2. Dan jumlah perpindahan diubah menjadi 3.

Tabel 7. Mencari Nilai OH (Pergeseran ke-3)

Index = 3 - 1 = 3			
Index	0	1	2
Karakter	S	I	A
Pindah = 3			

Pada perbandingan kedua, bandingkan karakter A yang ada pada *index* 2 dengan karakter I dan S yang berada pada *index* sebelumnya. Karena karakter A tidak ditemukan, maka karakter A dimasukan kedalam tabel *bmBc* dengan nilai OH sejumlah perpindahan, yaitu 3.

Tabel 8. Mencari Nilai OH (Perbandingan ke-3)

Bandingkan A dengan I, S			Hasil			
bmBc (Awal)			*bmBc* (Akhir)			
Karakter	I	S	Karakter	I	S	A
Nilai OH	1	2	Nilai OH	1	2	3

Untuk mendapatkan hasil akhir dari proses ini pada bagian *bmBc* (akhir), masukan setiap Nilai OH masing-masing karakter ke dalam karakter *pattern*.

Tabel 9. *bmBc* Akhir

Index	0	1	2
Karakter	S	I	A
Nilai OH	2	1	3

Tabel *bmBc* akhir merupakan acuan untuk menentukan tabel *ztBc*. Langkah selanjutnya adalah masukan semua karakter yang terdapat pada sumber teks dan buatlah menjadi dua dimensi, karakter yang dimasukan diurutkan sesuai abjad tak berulang.

Tabel 10. *ztBc* Awal

	A	D	E	I	N	O	S
A							
D							
E							
I							
N							
O							
S							

Setelah tabel dua dimensi terbentuk, langkah selanjutnya adalah mengisi tabel dengan cara melihat karakter pada baris dan karakter pada kolom. Untuk mengisi nilai pada tersebut, ada beberapa aturan sebagai berikut:

1. Jika pola yang dicari ada, maka masukan nilai OH karakter kolom.
2. Jika pola yang dicari tidak ada, maka masukan panjang *pattern*.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

3. Jika karakter pada suatu kolom yang dituju adalah karakter pertama pada *pattern*, maka masukan nilai OH karakter tersebut, tanpa melihat karakter baris.

Untuk baris 1 kolom 1 didapat karakter A dan A. Lihat baris *pattern* pada tabel 8. Untuk pola AA, pola ini tidak ada pada *pattern*. Maka masukan 3 (aturan 2). Hasilnya dapat dilihat pada tabel berikut

Tabel 11. *ztBc* (Perbandingan ke-1)

	A	D	E	I	N	O	S
A	3						
D							
E							
I							
N							
O							
S							

Selanjutnya mengisi baris 1 kolom 7 didapat karakter A dan S. Lihat pada tabel 8, pola AS tidak terdapat pada *pattern*. Tetapi karakter pada kolom 7 adalah baris 1 adalah karakter pertama pada *pattern*, maka masukan 2 (aturan 3). Hasilnya dapat dilihat pada tabel berikut.

Tabel 12. *ztBc* (Perbandingan ke-2)

	A	D	E	I	N	O	S
A	3						2
D							
E							
I							
N							
O							
S							

Selanjutnya misalkan mengisi baris 7 kolom 4 didapat karakter S dan I. Lihat pada tabel 8, pola SI terdapat pada *pattern*. Masukan nilai 1 (aturan 1). Hasilnya dapat dilihat pada tabel berikut.

Tabel 13. *ztBc* (Perbandingan ke-3)

	A	D	E	I	N	O	S
A	3						2
D							
E							
I							
N							
O							
S				1			

Langkah tersebut dilakukan sampai tabel *ztBc* terisi nilai semua dengan mengikuti langkah-langkah sebelumnya. Maka hasil akhirnya dapat dilihat pada tabel berikut.

Tabel 14. *ztBc* Akhir

	A	D	E	I	N	O	S
A	3	3	3	3	3	3	2
D	3	3	3	3	3	3	2
E	3	3	3	3	3	3	2
I	3	3	3	3	3	3	2
N	3	3	3	3	3	3	2
O	3	3	3	3	3	3	2
S	3	3	3	1	3	3	2

Langkah 2: Membuat Tabel *bmGs*

Setelah mendapatkan hasil akhir pada tabel *ztBc*, maka dilanjutkan dengan membuat tabel akhiran/*suffix* dari *pattern*.

Tabel 15. *bmGs* Awal

Index	0	1	2
Karakter	S	I	A
Nilai MH	?	?	?

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

Keterangan:
1. *Index* adalah indeks dari *pattern*.
2. Karakter adalah *pattern* yang akan dicari.
3. Nilai *Match Heuristic* (MH) adalah nilai pergeseran *good suffix*, nilai MH nantinya akan digunakan ketika ditemukan kecocokan pada saat mencocokan string karakter pertama atau lebih.

Untuk mengisi nilai MH, langkah pertama adalah membuat tabel *suffix* dari kanan ke kiri dan dari kiri ke kanan.
1. Tabel *suffix* dari kanan ke kiri, kolom *suffix* diperoleh dari pencacahan *pattern* yang dimulai dari kanan ke kiri, yaitu dari karakter A sampai karakter S.
2. Tabel *suffix* dari kiri ke kanan, kolom *suffix* diperoleh dari pencacahan *pattern* yang dimulai dari kiri ke kanan, yaitu dari S sampai karakter A.

Tabel 16. *Suffix* (kanan ke kiri dan kiri ke kanan)

Pattern: SIA (kanan ke kiri)					*Pattern*: SIA (kiri ke kanan)			
Index	*Prefix*	*Suffix*	*Len*		*Index*	*Prefix*	*Suffix*	*Len*
2	A	SI	2		0	S	IA	2
1	IA	S	1		1	SI	A	1
0	SIA	*Null*	0		2	SIA	*null*	0

Keterangan:
1. Pembuatan tabel *suffix* dari kanan ke kiri berfungsi untuk dijadikan *suffix comparator* pada *tabel suffix*.
2. Pembuatan *suffix* dari kiri ke kanan berfungsi sebagai *suffix* yang nantinya akan dibandingkan dan digunakan untuk mengisi nilai MH.

Dari tabel 16 akan terbentuk tabel.17, tabel ini berfungsi untuk mencari nilai MH. *Suffix Comparator* yang terdapat pada tabel 17 diambil dari kolom *suffix* yang ada pada tabel 3 *suffix* (kanan ke kiri). Sedangkan baris *suffix* yang terdapat pada tabel 17 diambil dari kolom *suffix* yang ada pada tabel 3 *suffix* (kiri ke kanan). Maka akan terlihat seperti Tabel 17.

Tabel 17. *Suffix* Lengkap Beserta *Suffix Comparator*

Suffix				
Index	0	1	2	*Move*
Prefix	S	I	A	
Suffix	IA	A	*null*	
Suffix Comparator	SI			1
	S			2
	Null			3
Nilai MH	?	?	?	

Langkah selanjutnya, adalah melakukan pemecahan akhiran pada *suffix* terhadap tabel *suffix comparator*. Berikan nilai MH pada tabel *bmGs* dengan *moving long* yang berkesesuaian. Terdapat beberapa ketentuan yang dapat menjadi acuan dalam mencari nilai MH, yaitu:
1. Cacah *suffix* pada masing-masing *index* terhadap seluruh *suffix comparator*.
2. Jika *length* kedua *suffix* tidak sama panjang, maka potong salah satu *suffix* yang memiliki *length* terbesar. Besaran pemotongan *suffix* disesuaikan dengan nilai *length* terkecil kearah paling kanan karakter. Jika ditemukan kecocokan antar *suffix*, bandingkan *prefix* kedua *suffix* (jika masing-masing *suffix* memiliki *prefix*). Jika *prefix* yang diperbandingkan adalah sama, maka kecocokan tidak dapat diterima.
3. Lakukan poin 2 hingga *index suffix* mencapai 0.
4. Untuk *index suffix* terbesar, secara default akan diberikan nilai 1 pada nilai MH, sedangkan yang lainnya diberikan nilai sesuai dengan nilai *move* yang diraih.

Untuk proses pertama diberikan nilai 1 pada nilai MH, karena *index* terbesar (ketentuan 4). Prosesnya dapat dilihat pada tabel 18.

Tabel 18. Proses 1 *preBmGs*

Proses #1-01 Tabel *Suffix*		
Index	2	*Move*
Prefix	A	

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

Suffix	*null*	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	1	

bmGs			
Index	0	1	2
Karakter	S	I	A
Nilai MH	?	?	1

Proses kedua, bandingkan karakter *suffix* dengan *suffix comparator*. Jumlah karakter yang dibandingkan disesuaikan dengan jumlah karakter yang paling sedikit (ketentuan 2). Ternyata karakter *suffix* tidak ada yang sama dengan *suffix comparator* pada setiap perbandingan. Sehingga jumlah perpindahan dimasukan ke dalam nilai MH yaitu 3.

Tabel 19. Proses 2 *preBmGs*

Proses #2-01 Tabel *Suffix*		
Index	1	Move
Prefix	I	
Suffix	A	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	?	

Proses #2-02 Tabel *Suffix*		
Index	1	Move
Prefix	I	
Suffix	A	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	?	

Proses #2-03 Tabel *Suffix*		
Index	1	Move
Prefix	I	
Suffix	A	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	?	

bmGs			
Index	0	1	2
Karakter	S	I	A
Nilai MH	?	3	1

Proses ketiga, bandingkan karakter *suffix* dengan *suffix comparator*. Jumlah karakter yang dibandingkan disesuaikan dengan jumlah karakter yang paling sedikit (ketentuan 2). Ternyata karakter *suffix* tidak ada yang sama dengan *suffix comparator* pada setiap perbandingan. Sehingga jumlah perpindahan dimasukan ke dalam nilai MH yaitu 3.

Tabel 20. Proses 3 *preBmGs*

Proses #3-01 Tabel *Suffix*		
Index	0	Move
Prefix	S	
Suffix	IA	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	?	

Proses #3-02 Tabel *Suffix*		
Index	0	Move
Prefix	S	
Suffix	IA	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	?	

Proses #3-03 Tabel *Suffix*		
Index	0	Move
Prefix	S	
Suffix	IA	
Suffix Comparator	SI	1
	S	2
	null	3
Nilai MH	?	

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

bmGs			
Index	0	1	2
Karakter	S	I	A
Nilai MH	3	3	1

Hasil akhir dari setiap langkah-langkah sebelumnya dapat dilihat pada tabel 21 berikut.

Tabel 21. *bmGs/ztGs*

Index	0	1	2
Karakter	S	I	A
Nilai MH	3	3	1

Karena tabel *bmGs* dan *ztGs* sama, maka untuk proses *preprocessing* sudah selesai dengan hasil yang dapat dilihat pada tabel 14 untuk tabel *ztBc*, dan pada tabel 21 untuk tabel *bmGs* atau *ztGs*. Selanjutnya sudah bisa melakukan proses pencocokan.

Langkah 3: Melakukan Pergeseran

Text : INDONESIA
Pattern: SIA
Percobaan ke-1:

Index	0	1	2	3	4	5	6	7	8
Text	I	N	D	O	N	E	S	I	A
Pattern	S	I	A						

Dari hasil percobaan ke-1, karakter *pattern* A sejajar dengan karakter *text* D. Artinya pada percobaan ini terjadi ketidakcocokan. Maka dilakukan pergeseran sejauh 3 karakter (lihat tabel 14 *ztBc*).
Pergeseran = 3 (ztBc [N][D])

Percobaan ke-2:

Index	0	1	2	3	4	5	6	7	8
Text	I	N	D	O	N	E	S	I	A
Pattern				S	I	A			

Dari hasil percobaan ke-2, karakter *pattern* A sejajar dengan karakter *text* E. Artinya pada percobaan ini terjadi ketidakcocokan. Maka dilakukan pergeseran sejauh 3 karakter (lihat tabel 14 *ztBc*).
Pergeseran = 3 (ztBc [N][E])

Percobaan ke-3:

Index	0	1	2	3	4	5	6	7	8
Text	I	N	D	O	N	E	S	I	A
Pattern							S	I	A

Dari hasil percobaan ke-3, karakter *pattern* A sejajar dengan karakter *text* A. Artinya pada percobaan ini terjadi kecocokan. Maka pencocokan dilanjutkan pada karakter sebelum karakter A, yaitu karakter I dan S pada *pattern*. Terjadi kecocokan pada semua karakter *pattern*, kemudian algoritma akan menandai lokasi penemuan *pattern*. Pencocokan tidak dapat dilanjutkan karena karakter *text* tidak mencukupi. Maka dapat disimpulkan proses percobaan pencocokan selesai.
Kesimpulan:

1. Pada tahap pencocokan dengan algoritma *Zhu-Takaoka* di atas, pencocokan dilakukan sebanyak 3 iterasi. Ditemukan pada iterasi ke-3.
2. Jumlah *index* pada *text* adalah 8, pattern ditemukan pada indeks ke 6-8.
3. Sebagai hasil dari pencarian di atas maka akan muncul nama negara Indonesia dengan beberapa makanan khas yang ada, misalkan Karedok, Trancam, dan Sate Sarepeh.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 1, No. 1, September 2019
ISSN 2685-998X (Media Online)
DOI: 10.30865/json.v1i1.1365

4. KESIMPULAN

Dari hasil pembahasan disimpulkan, yakni:

1. Proses pencarian makanan khas dilakukan dengan melakukan pencocokan *text* dengan *pattern*, dimana *text* berupa nama suatu negara dan *pattern* kata kunci yang dicari menggunakan algoritma *Zhu-Takaoka*.
2. Algoritma *Zhu-Takaoka* dapat diterapkan dalam perancangan aplikasi pencarian makanan khas di dunia, sehingga diharapkan memudahkan pengguna untuk mencari makanan khas dari suatu negara.

REFERENCES

[1] V. Nopitasari, I. Oktaviani, and I. Nofikasari, "Aplikasi Resep Masakan Tradisional Berbasis Mobile," vol. 12, pp. 61–80, 2017.
[2] G. Togarotop, A. Erlansari, and F. F. Coastera, "Implementasi Algoritma Zhu-Tajkaoka Pada Aplikasi Kamus Istilah Musik Berbasis Android," vol. 5, no. 2, pp. 147–153, 2017.
[3] Drs. Suarga. M.Sc. M.Math. Ph.D, *Algoritma dan Pemrograman*. ANDI Yogyakarta, 2012.