semanTIK, Vol.5, No.1, Jan-Jun 2019, pp. 131-142
ISSN: 2502-8928 (Online)
DOI : 10.5281/zenodo.3242287
Terakreditasi "Peringkat 4 (Sinta 4)" oleh Kemenristekdikti
Website : http://ojs.uho.ac.id/index.php/semantik

PERBANDINGAN ALGORITMA *HORSPOOL* DAN ALGORITMA *RAITA* PADA APLIKASI ISTILAH PSIKOLOGI BERBASIS ANDROID

Mamta Culkari Puding*[1], Jumadil Nangi[2], Adha Mashur Sajiah[3]
[*1,2,3]Jurusan Teknik Informatika, Fakultas Teknik Universitas Halu Oleo, Kendari
e-mail : [*1]**mamtaculkari97@gmail.com**, [2]jumadilnangi87@gmail.com, [3]adha.m.sajiah@gmail.com

Abstrak

Minimnya tenaga kesehatan jiwa profesional di Indonesia mengakibatkan banyaknya penderita gangguan jiwa yang belum tertangani secara medis. Untuk memudahkan para tenaga kesehatan jiwa khususnya mahasiswa psikologi untuk mempelajari istilah-istilah dalam ilmu psikologi. maka perlu adanya suatu kamus digital, karena kamus berbentuk buku cetak menyulitkan untuk dibawa kemana-mana dan proses pencarian istilah yang diinginkan memakan waktu yang cukup lama. Berdasarkan hal itu maka dibuatlah aplikasi kamus istilah psikologi berbasis Android.

Untuk membuat aplikasi ini maka diimplementasikan metode Pencocokan *String* (*String Matching*) pada pencarian kata istilah. *String Matching* memiliki beberapa algoritma diantaranya adalah algoritma *Horspool* dan algoritma *Raita*.

Dalam penelitian ini dilakukan analisis perbandingan antara algoritma *Horspool* dan algortima *Raita*, untuk menentukan algoritma yang paling baik digunakan dalam aplikasi Kamus Istilah Psikologi. Parameter yang digunakan untuk membandingkan kedua algoritma tersebut adalah waktu pencarian (ms) dan kompleksitas algoritma (*Big-O*).

Hasil dari penelitian ini menunjukkan bahwa algoritma *Raita* memiliki kecepatan waktu pencarian yang lebih cepat dibandingkan algoritma *Horspool* serta kompleksitas algoritma *Raita* lebih cepat dibandingkan algoritma *Horspool*. Rata-rata total kecepatan waktu untuk algoritma *Raita* adalah 13,60 ms dan algortima *Horspool* adalah 14,79 ms. Kompleksitas algoritma *Raita* adalah T(n) = Θ(MN) dan kompleksitas algoritma *Horspool* adalah T(n) = Θ(M(N-2)).

Kata Kunci—Kamus, Psikologi, Pencocokan *String*, Algoritma *Horspool*, Algoritma *Raita*

Abstract

The lack of professional health personnel in Indonesia has resulted in many people with mental disorders who have not been treated medically. To make it easier for the mental health workers, especially psychology students to study the term in psychology. It is necessary to have a digital dictionary because a tick printed book dictionary makes it difficult to carry around and the search term desired requires a long time. Based on this, a dictionary application for the psychology term based on Android was made.

To make this application implemented the string matching method on the codeword search. String matching has several algorithms, some of which are Horspool algorithm and Raita algorithm.

In this research study an analysis of the comparison between the Horspool algorithm and Raita algorithm, to determine the most useful algorithms in the dictionary of psychological terms. The parameter used to compare the two algorithms search time (ms) and complexity algorithms (Big-O).

The results of this study show that Raita Algorithm has a faster search than Horspool Algorithm and the complexity of Raita algorithm has faster than Horspool algorithm. The average total velocity

of time for Raita algorithm is 13,60 ms and Horspool algorithm are 14,79 ms. The complexity of Raita algorithm is T(n) = Θ(MN) and the complexity of Horspool algorithm is T(n) = Θ(M(N-2)).

Keywords*—Dictionary, Psychology, String Matching, Horspool Algorithm, Raita Algorithm*

1. PENDAHULUAN

Kamus merupakan sumber rujukan yang membuat kosakata dan penjelasan maknanya, kamus bertujuan untuk menyediakan makna yang tepat bagi kata yang dicari oleh pengguna [1]. Pada umumnya kamus berbentuk buku cetak yang penggunaannya menyulitkan, karena pengguna harus mencari arti dan istilah secara manual. Di sisi lain kamus yang tebal dan berat menyulitkan dibawa kemana-mana dan proses pencarian istilah yang diinginkan memakan waktu yang cukup lama.

Untuk mengatasi masalah tersebut dibutuhkan suatu teknologi yang dapat digunakan untuk menggantikan kamus berbentuk buku yang penggunaannya masih secara manual. Salah satu cara yang dapat digunakan untuk mengatasi hal tersebut adalah dibutuhkan sebuah aplikasi *smartphone* yang dapat membantu dalam mencari kata atau istilah, yaitu kamus digital berbasis Android.

Banyaknya penderita gangguan jiwa yang belum tertangani secara medis, dikarenakan masih minimnya tenaga kesehatan jiwa profesional di Indonesia. Hal tersebut tentunya semakin menghambat upaya pencegahan dan penanganan persoalan kesehatan jiwa masyarakat. Jumlah tenaga kesehatan jiwa profesional di Indonesia masih belum mampu memenuhi kuota minimal yang telah ditetapkan oleh Organisasi Kesehatan Dunia atau WHO. Saat ini Indonesia dengan penduduk sekitar 250 juta jiwa baru memiliki sekitar 451 psikolog klinis (0,15 per 100.000 penduduk), 773 psikiater (0,32 per 100.000 penduduk), dan perawat jiwa 6.500 orang (2 per 100.000 penduduk) [2]. WHO telahmenetapkan standar jumlah tenaga psikolog dan psikiater dengan jumlah penduduk adalah 1:30 ribu orang, atau 0,03 per 100.000 penduduk. Dari data tersebut maka perlu adanya suatu kamus digital untuk memudahkan para tenaga kesehatan jiwa khususnya mahasiswa psikologi untuk mempelajari istilah-istilah dalam ilmu psikologi.

Pembuatan kamus istilah dalam bentuk teknologi digital dapat diimplementasikan dengan menggunakan metode Pencocokan *String*. Pencocokan *String* banyak digunakan dalam aplikasi pengolahan teks untuk pencarian kata dalam berkas teks. Pencarian *string* di dalam teks disebut juga *String Matching* [3]. Adapun algoritma yang digunakan yaitu algoritma *Horspool* dan algoritma *Raita.* Dalam penelitian ini Penulis membandingkan kedua metode tersebut.

2. METODE PENELITIAN

2.1 Kamus Istilah

Menurut Kamus Besar Bahasa Indonesia, Kamus Istilah (*Glosarium*) adalah kamus dalam bentuk yang ringkas dengan daftar kata beserta penjelasannya dalam bidang tertentu. *Glosarium* adalah suatu daftar alfabetis istilah dalam suatu ranah pengetahuan tertentu yang dilengkapi dengan definisi untuk istilah-istilah tersebut. Biasanya *glosarium* ada di bagian akhir suatu buku dan menyertakan istilah-istilah dalam buku tersebut yang baru diperkenalkan atau tidak umum ditemukan. *Glosarium* dwibahasa adalah daftar istilah dalam satu bahasa yang didefinisikan dalam bahasa lain atau diberi sinonim (atau paling tidak sinonim terdekat) dalam bahasa lain.

2.2 Psikologi

Psikologi berasal dari bahasa Yunani yaitu *psyche* yang artinya jiwa dan *logos* artinya ilmu. Jadi secara etimologi psikologi artinya ilmu yang mempelajari jiwa, baik mengenai macam-macam gejalanya, proses maupun latar belakangnya. Jiwa secara harfiah berasal dari perkataan sansekerta JIV, yang berarti lembaga hidup (*levenbeginsel*), atau daya hidup (*levenscracht)*. Oleh karena itu jiwa merupakan pengertian yang abstrak, tidak bisa dilihat dan belum bisa diungkapkan secara lengkap dan jelas, maka orang lebih cenderung mempelajari "jiwa yang memateri" atau gejala "jiwa yang meraga/menjasmani", yaitu bentuk tingkah laku manusia (segala aktivitas, perbuatan, penampilan diri) sepanjang hidupnya [4].

2.3 Pencarian (*Searching*)

Algoritma pencarian atau *searching algorithm* adalah algoritma yang menerima sebuah argumen kunci dengan langkah-langkah tertentu akan mencari rekaman dengan kunci tersebut. Setelah proses pencarian dilaksanakan, akan diperoleh salah satu dari dua kemungkinan, yaitu data yang dicari ditemukan (*successful*) atau tidak ditemukan (*unsuccessful*) [5].

2.4 Algoritma *Horspool*

Algoritma *Horspool* adalah penyederhanaan dari algoritma *Boyer-Moore* yang dibuat oleh R. Nigel Horspool. Algoritma *Horspool* bekerja dengan metode yang hampir sama dengan algoritma *Boyer-Moore* namun tidak melakukan lompatan berdasarkan karakter pada *pattern* yang ditemukan tidak cocok pada teks [6].

Terdapat dua tahap pada pencocokan *string* menggunakan algoritma *Horspool* [7], yaitu:

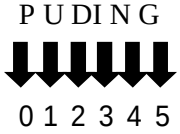
1. Tahap Praproses

Pada tahap ini, dilakukan *observasi pattern* terhadap teks untuk membangun sebuah tabel *bad-match* yang berisi nilai *shift* ketika ketidakcocokan antara *pattern* dan teks terjadi. Adapun, langkah-langkah yang dilakukan algoritma *Horspool* pada tahap praproses adalah :

a. Algoritma *Horspool* melakukan pencocokan karakter paling kanan *pattern*.
b. Setiap karakter pada *pattern* ditambah ke dalam Tabel BmBc dan dihitung nilai *shiftnya*.
c. Karakter yang berada pada ujung *pattern* tidak dihitung dan tidak dijadikan karakter ter-kanan dari karakter yang sama dengannya.
d. Apabila terdapat dua karakter yang sama dan salah satunya bukan karakter terkanan, maka karakter dengan indeks terbesar yang dihitung nilai *shift*-nya.
e. Algoritma *Horspool* menyimpan panjang dari *pattern* sebagai panjang nilai *shift* secara*default* apabila karakter pada teks tidak ditemukan dalam *pattern*.
f. Nilai (BM) *shift* yang akan digunakan dapat dicari dengan perhitungan panjang, dari *pattern* dikurang indeks terakhir karakter dikurang 1, untuk masing-masing karakter (BC).

Sebagai contoh, dapat dilihat pada Tabel 1.

Pattern : PUDING



Tabel 1 Tabel BmBc pada Praproses Kata PUDING

BC	BM
P	5
U	4
D	3
I	2
N	1
*	6

Untuk mencari nilai, geser pada Tabel BmBc, digunakan Persamaan (1).

$$BM[i] = m - i - 1 \qquad (1)$$

BM [0] = 6 – 0 – 1 = 5
BM [1] = 6 – 1 – 1 = 4
BM [2] = 6 – 2 – 1 = 3
BM [3] = 6 – 3 – 1 = 2
BM [4] = 6 – 4 – 1 = 1
(*) : karakter yang tidak dikenali

2. Tahap Pencarian

Secara sistematis, langkah-langkah yang dilakukan algoritma *Horspool* pada tahap pencarian adalah:

a. Dilakukan perbandingan karakter paling kanan *pattern* terhadap *window*.
b. Tabel BmBc digunakan untuk melewati karakter ketika ketidakcocokan terjadi.
c. Ketika ada ketidakcocokan, maka karakter paling kanan pada *window* berfungsi sebagai landasan untuk menentukan jarak *shift* yang akan dilakukan.
d. Setelah melakukan pencocokan (baik hasilnya cocok atau tidak cocok) dilakukan pergeseran ke kanan pada *window*.
e. Prosedur ini dilakukan berulang-ulang sampai *window* berada pada akhir teks atau ketika *pattern* cocok dengan teks.

Sebagai contoh dalam mendeskripsikan algoritma *Hoorspol* diberikan teks dan *pattern* sebagai berikut :

Teks = **MAMTA CULKARI PUDING**
Pattern = **PUDING**

Inisialisasi awal dan pembuatan BmBc terlihat pada Tabel 2 dan Tabel 3.

Tabel 2 Inisialisasi Awal *BmBc*

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P	P	U	D	I	N	G														
I	0	1	2	3	4	5														

Tabel 3 Pembuatan BmBc

P	P	U	D	I	N	*
I	0	1	2	3	4	-
S	5	4	3	2	1	6

Dilihat dari Tabel 2, inisialisasi awal BmBc dilakukan. Setiap teks dan *pattern* masing masing diberi nilai *m* dan *i*, dimana *m* adalah panjang *pattern* dan *i* adalah indeks. Tabel 3 menunjukkan nilai pergeseran BmBc dengan menghitung nilai BM seperti yang telah dilakukan pada Tabel 1. Pada tahap awal pencarian, dilakukan perbandingan karakter paling kanan *pattern* terhadap *window*. Apabila terjadi ketidakcocokan maka akan dilakukan pergeseran ke kanan untuk melawati karakter yang tidak cocok di mana nilai pergeserannya terdapat pada Tabel BmBc. Karakter paling kanan teks pada *window* berfungsi sebagai landasan untuk menentukan jarak geser yang akan dilakukan. Hal ini terlihat pada Tabel 4.

Tabel 4 Iterasi Algoritma *Horspool* Pertama

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P	P	U	D	I	N	G														
I	0	1	2	3	4	5														

Pada Tabel 4 terdapat ketidakcocokan antara karakter “Spasi” dan “G”. Karakter “Spasi” tidak terdapat pada Tabel BmBc sehingga digantikan oleh tanda (*). Tanda (*) bernilai sebesar 6 sehingga dilakukan pergeseran sebanyak 6 kali. Hal ini terlihat pada Tabel 5.

Tabel 5 Iterasi Algoritma *Horspool* Kedua

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P							P	U	D	I	N	G								
I							0	1	2	3	4	5								

Pada Tabel 5 terdapat ketidakcocokan kembali antara karakter “R” dan “G”. Karakter “R” tidak terdapat pada Tabel BmBc sehingga digantikan oleh tanda (*). Tanda (*) bernilai sebesar 6 sehingga dilakukan pergeseran sebanyak 6 kali. Hal ini terlihat pada Tabel 6.

Tabel 6 Iterasi Algoritma *Horspool* Ketiga

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P													P	U	D	I	N	G		
I													0	1	2	3	4	5		

Pada Tabel 6 terdapat ketidakcocokan kembali antara karakter “I” dan “G”. Karakter “I” terdapat pada Tabel BmBc yang bernilai sebesar 2 sehingga dilakukan pergeseran sebanyak 2 kali. Hal ini terlihat pada Tabel 7.

Tabel 7 Iterasi Algoritma *Horspool* Keempat

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P															P	U	D	I	N	G
I															0	1	2	3	4	5

Pada Tabel 7 terdapat kecocokan pada karakter “G” dan “G”, karena karakter cocok, maka pencocokan dilanjutkan pada karakter sebelum karakter “G” yaitu karakter “N”. Hal ini terlihat pada Tabel 8.

Tabel 8 Iterasi Algoritma *Horspool* Kelima

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P															P	U	D	I	N	G
I															0	1	2	3	4	5

Pada Tabel 8 terdapat kecocokan lagi pada karakter “N” dan “N”, karena karakter cocok, maka pencocokan dilanjutkan pada karakter sebelum karakter “N” yaitu karakter “I”. Hal ini terlihat pada Tabel 9.

Tabel 9 Iterasi Algoritma *Horspool* Keenam

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P															P	U	D	I	N	G
I															0	1	2	3	4	5

Pada Tabel 9 terdapat kecocokan lagi pada karakter “I” dan “I”, karena karakter cocok, maka pencocokan dilanjutkan pada karakter sebelum karakter “I” yaitu karakter “D”. Hal ini terlihat pada Tabel 10.

Pada Tabel 10 terdapat kecocokan lagi pada karakter “D” dan “D”, karena karakter cocok, maka pencocokan dilanjutkan pada karakter sebelum karakter “D” yaitu karakter “U”. Hal ini terlihat pada Tabel 11.

Tabel 10 Iterasi Algoritma *Horspool* Ketujuh

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P															P	U	D	I	N	G
I															0	1	2	3	4	5

Tabel 11 Iterasi Algoritma *Horspool* Kedelapan

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P															P	U	D	I	N	G
I															0	1	2	3	4	5

Pada Tabel 11 terdapat kecocokan lagi pada karakter "U" dan "U", karena karakter cocok, maka pencocokan dilanjutkan pada karakter sebelum karakter "U" yaitu karakter "P". Hal ini terlihat pada Tabel 12.

Tabel 12 Iterasi Algoritma *Horspool* Kesembilan

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P															P	U	D	I	N	G
I															0	1	2	3	4	5

Pada Tabel 12 *window* telah berada pada akhir teks dan semua *pattern* cocok dengan teks. Seluruh pencocokan karakter menggunakan algoritma *Horspool* telah selesai dan berhenti pada iterasi kesembilan.

2.5 Algoritma *Raita*

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. *Raita* merancang sebuah algoritma dengan membandingkan karakter yang terakhir dari pola dari karakter paling kanan dari *window*. Jika mereka cocok, kemudian karakter pertama dari pola teks paling kiri dari *window* juga dibandingkan. Jika mereka cocok, maka akan dibandingkan karakter tengah pola dengan karakter teks tengah *window*. Pada akhirnya, jika mereka benar-benar cocok, maka algoritma membandingkan karakter lain mulai dari karakter kedua ke karakter kedua terakhir, dan mungkin membandingkan dengan karakter tengah lagi [8].

Tahap pencarian algoritma *Raita* :

a. Membuat tabel pergeseran pola yang dicari sebagai kata yang akan dicari pada teks.
b. Jika dalam proses pembandingan terjadi ketidakcocokan antara pasangan karakter pada akhir pola dengan karakter teks, pergeseran dilakukan sesuai nilai karakter pada Tabel BmBc.
c. Jika dalam proses pembandingan akhir pola terjadi ketidakcocokan lagi maka karakter akan digeser lagi sesuai Tabel BmBc.
d. 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.
e. 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 = **MAMTA CULKARI PUDING**
Pattern = **PUDING**

Untuk melakukan perhitungan maka dibuat Tabel BmBc dengan Persamaan (2).

$$k = m - 2 \quad (2)$$

Persamaan (3) berfungsi sebagai batas pencarian karakter pada pola.

$$BM[i] = m - i - 1 \quad (3)$$

Berfungsi sebagai pencari nilai karakter pada Tabel BmBc. Tabel 13 menunjukkan Tabel BmBc.

P U D I N G

0 1 2 3 4 5

Tabel 13 Tabel BmBc

BC	BM
P	5
U	4
D	3
I	2
N	1
*	6

Berdasarkan Tabel 13, dapat diketahui perhitungan Tabel BmBC dengan Persamaan (2) yaitu :

$$k = m - 2 = 6 - 2 = 4$$

Maka $i = 0 - 4$

Untuk mencari nilai geser pada Tabel BmBc, digunakan Persamaan (3) yaitu :

$$BM\ [BC[P]] = 6 - 0 - 1 = 5$$
$$BM\ [BC[U]] = 6 - 1 - 1 = 4$$
$$BM\ [BC[D]] = 6 - 2 - 1 = 3$$
$$BM\ [BC[I]] = 6 - 3 - 1 = 2$$
$$BM\ [BC[N]] = 6 - 4 - 1 = 1$$

(*) : karakter yang tidak dikenali

Panjang pola pada contoh adalah sebesar 6. Maka untuk karakter yang tidak ada pada tabel diinisialisasikan dengan tanda (*) yang nilainya sesuai dengan panjang pola (m).

Pencarian algoritma *Raita* tahap pertama yaitu mencocokkan akhir pola dengan teks, jika terjadi ketidakcocokan maka pola akan bergeser ke kanan sebanyak nilai teks yang ada di Tabel 14.

Tabel 14 Pencarian pada Teks Proses Pertama

T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P	P	U	D	I	N	G														

Pada Tabel 14 terdapat ketidakcocokan antara karakter “Spasi” dan “G”. Karakter “Spasi” tidak terdapat pada Tabel BmBc sehingga digantikan oleh tanda (*). Tanda (*) bernilai sebesar 6 sehingga dilakukan pergeseran sebanyak 6 kali. Pencarian algoritma *Raita* tahap kedua yaitu melakukan pergeseran sebanyak 6 kali (sesuai dengan karakter “Spasi”).

Tabel 15 Pencarian pada Teks Proses Kedua

T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P							P	U	D	I	N	G								

Dilihat pada Tabel 15 terjadi ketidakcocokan antara karakter “R” dan “G”. Karakter “R” tidak terdapat dalam Tabel BmBc sehingga digantikan dengan tanda (*). Tanda (*) memiliki nilai sebesar 6 sehingga dilakukan pergeseran sebanyak 6 kali.

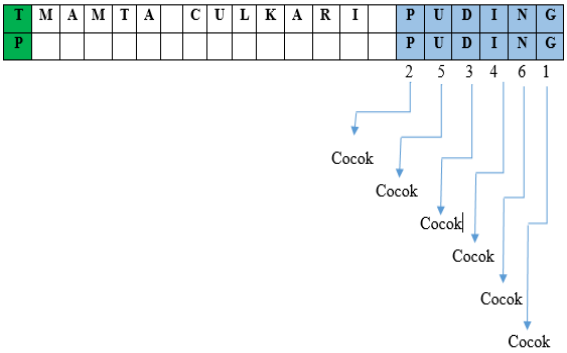
Pencarian algoritma *Raita* tahap ketiga yaitu melakukan pergeseran sebanyak 6 kali (sesuai dengan karakter “R”).

Tabel 16 Pencarian pada Teks Proses Ketiga

T	M	A	M	T	A		C	U	L	K	A	R	I		P	U	D	I	N	G
P													P	U	D	I	N	G		

Pada Tabel 16 terdapat ketidakcocokan kembali antara karakter “I” dan “G”. Karakter “I” terdapat pada Tabel BmBc yang bernilai sebesar 2 sehingga dilakukan pergeseran sebanyak 2 kali. Hal ini terlihat pada Tabel 17.

Tabel 17 Pencarian pada Teks Proses Keempat



Dilihat pada Tabel 17 terjadi kecocokan antara teks dan pola. Seluruh pencocokan karakter menggunakan algoritma *Raita* telah selesai dan berhenti pada pencarian teks proses keempat.

2.6 Kompleksitas Algoritma

Suatu masalah dapat mempunyai banyak algoritma penyelesaian. Algoritma yang digunakan tidak saja harus benar, namun juga harus efisien. Efisien suatu algoritma dapat diukur dari waktu eksekusi algoritma dan kebutuhan ruang memori. Algoritma yang eifisien adalah algoritma yang meminimumkan kebutuhan waktu dan ruang. Dengan menganalisis beberapa algoritma untuk suatu masalah, dapat diidentifikasi suatu algoritma yang paling efisien. Besaran yang digunakan untuk menjelaskan model pengukuran waktu dan ruang ini adalah kompleksitas algoritma [9].

Kompleksitas waktu, dinyatakan oleh T(*n*), diukur dari jumlah tahapan komputasi yang dibutuhkan untuk menjalankan algoritma sebagai fungsi dari ukuran masukan *n*, dimana ukuran masukkan (*n*) merupakan jumlah data yang diproses oleh sebuah algoritma. Sedangkan kompleksitas ruang, S(*n*), diukur dari memori yang digunakan oleh struktur data yang terdapat di dalam algoritma sebagai fungsi dari masukan *n*. Dengan menggunakan kompleksitas waktu dan kompleksitas ruang, dapat ditentukan laju peningkatan waktu dan ruang yang diperlukan algoritma, seiring dengan ukuran masukan (*n*) [9].

2.7 Notasi *Big-O*

Notasi *O* menyatakan *running time* dari suatu algoritma untuk kemungkinan kasus

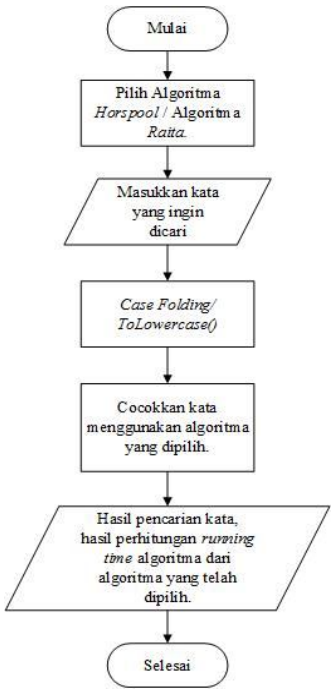
terburuk. Notasi *O* memiliki dari beberapa bentuk. Notasi *O* dapat berupa salah satu bentuk maupun kombinasi dari bentuk-bentuk tersebut.

3. HASIL DAN PEMBAHASAN

3.1 *Flowchart* Sistem

Flowchart sistem untuk Perbandingan Algoritma *Horspool* dan Algoritma *Raita* Pada Aplikasi Kamus Istilah Psikologi Berbasis Android ditunjukkan oleh Gambar 1. Adapun alur kerja *flowchart diagram* sistem adalah sebagai berikut:

a. *User* memilih algoritma pencarian yang ingin dipakai.
b. *User* memasukkan kata yang ingin dicari.
c. Kata yang dimasukkan akan melalui proses *case folding/toLowerCase* untuk mengubah kata menjadi huruf kecil.
d. Kata yang dicari akan dicocokkan menggunakan algoritma yang dipilih.
e. Kemudian sistem akan menampilkan kata yang dicari beserta waktu pencarian dari metode tersebut.



Gambar 1 *Flowchart* Sistem

3.2 Pengujian Sistem

1) Pengujian Pencarian dengan Menggunakan 1 Kata

Pengujian ini dilakukan untuk memeriksa ketepatan dan kecepatan pencarian kata istilah psikologi yang akan dicari oleh *user*. Pengujian ini dilakukan dengan memasukkan satu kata yang akan dicari ke dalam *form* aplikasi untuk menguji performa dari aplikasi. Tabel 18 menunjukkan hasil pengujian menggunakan 1 kata (Algoritma *Horspool*) dan Tabel 19 menunjukkan hasil pengujian menggunakan 1 kata (Algoritma *Raita*).

Tabel 18 Pengujian Menggunakan 1 Kata (Algoritma *Horspool*)

No	Kata yang dicari	Percobaan Algoritma *Horspool* ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Normal	16.7	14.5	14.4	16.4	14.7	15.5	15.9	15.1	14.9	14.0	15.21
2	Phobia	15.2	16.6	15.3	15.9	15.2	17.8	15.9	14.9	19.5	18.2	16.45
3	Hyper	15.6	16.5	17.8	16.7	15.5	14.9	17.7	15.0	17.9	17.0	16.46
4	Impulsif	15.6	15.6	16.8	15.6	15.3	14.9	13.1	14.8	14.4	15.6	15.17
5	Liberosis	18.9	18.5	17.9	18.1	17.1	15.5	16.9	18.4	18.8	18.2	17.83
6	Somnabulisme	12.1	12.0	14.4	14.2	12.0	12.1	13.3	13.2	14.5	14.5	13.23
7	Narkolepsi	16.7	13.0	14.6	13.2	14.4	13.6	13.1	13.3	17.8	16.6	14.63
8	Hipnotis	16.7	16.7	16.9	18.4	16.3	16.9	15.4	16.2	15.1	16.4	16.50
9	Vellichor	18.1	17.5	19.0	18.4	19.0	19.3	17.0	19.0	19.4	18.2	18.49
10	Zydis	16.4	17.7	15.9	16.6	15.4	14.6	15.6	16.4	17.5	15.2	16.13
Rata-rata :												**16.01**

Tabel 19 Pengujian Menggunakan 1 Kata (Algoritma *Raita*)

No.	Kata yang dicari	Percobaan Algoritma *Raita* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Normal	15.2	15.3	15.6	14.6	15.2	16.6	15.3	17.2	16.4	16.6	15.80
2	Phobia	14.2	14.1	13.6	14.6	14.2	15.5	15.8	14.7	14.5	15.7	14.69
3	Hyper	14.7	15.6	15.2	15.9	16.2	17.4	16.8	16.9	16.5	15.8	16.10
4	Impulsip	12.6	13	11.8	12.4	12.4	13.3	11.3	11.8	11.1	10.7	12.04
5	Liberosis	13.4	13.6	13.8	14.7	14.3	15.4	16.4	16.8	17.2	16.6	15.22
6	Somnabulisme	10.8	10.5	10.9	11	10.7	10.9	10.3	11.1	10	11.6	10.78
7	Narkolepsi	12.3	13.7	12.8	12.6	14.3	14.8	12.7	14.5	15.8	14.1	13.76
8	Hipnotis	17.5	18.8	17.7	17.1	18.5	18.7	18.9	16.4	18.4	17.1	17.91
9	Vellichor	18.4	17.8	18.3	18.8	19.2	17.7	17.2	17.3	17.4	18.2	18.03
10	Zydis	14.4	13.6	15.8	14	14.5	13.1	14.7	13.8	15.1	15.6	14.46
Rata-rata :												**14.88**

Berdasarkan pengujian pencarian dengan menggunakan 1 kata pada masing-masing istilah diperoleh hasil rata-rata waktu untuk algoritma *Horspool* adalah 16.01 ms, sedangkan untuk algoritma *Raita* adalah 14.88 ms. Berdasarkan hasil tersebut pada pengujian pencarian dengan meggunakan 1 kata, algoritma *Raita* lebih cepat dibandingkan algoritma *Horspool* dengan selisih waktu kecepatan 1.13 ms.

2) Pengujian Pencarian dengan Menggunakan 2 Kata

Pengujian ini dilakukan untuk memeriksa ketepatan dan kecepatan pencarian kata istilah psikologi yang akan dicari oleh *user*. Pengujian ini dilakukan dengan memasukkan sepuluh kata yang akan dicari ke dalam *form* aplikasi untuk menguji performa dari aplikasi. Tabel 20 menunjukkan hasil pengujian menggunakan 2 kata (Algoritma *Horspool*) dan Tabel 21 menunjukkan hasil pengujian menggunakan 2 kata (Algoritma *Raita*).

Tabel 20 Pengujian Menggunakan 2 Kata (Algoritma *Horspool*)

No.	Kata yang dicari	Percobaan Algoritma *Horspool* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Art Therapy	17.6	16.2	17.3	18.0	16.4	16.2	15.0	16.5	17.8	16.2	16.72
2	Bettelheim Bruno	12.1	12.7	12.7	13.5	12.8	12.6	12.0	12.6	12.4	13.3	12.67
3	Down Syndrome	14.8	13.4	14.8	13.8	14.8	15.3	15.1	15.3	16.0	14.4	14.77
4	Episodic Memory	15.6	13.3	13.0	15.1	12.6	13.8	13.2	13.6	13.7	12.1	13.60
5	Identitas Gender	15.5	11.4	15.7	11.9	12.5	13.6	13.2	11.1	12.7	13.9	13.15
6	Insanity Defense	13.4	14.0	12.9	12.8	11.5	13.2	12.7	11.9	12.4	12.3	12.71
7	Jet Lag	21.0	21.4	20.1	23.0	21.3	21.6	20.6	20.3	20.4	21.4	21.11
8	Usher Syndrome	15.78	14.2	14.2	14.6	17.0	14.6	14.5	14.6	14.1	15.0	14.86
9	Waking Hypnosis	14.9	13.8	13.9	13.4	13.5	16.9	13.3	13.1	12.9	17.8	14.35
10	Zeigarnik Effect	14.2	14.4	13.2	14.4	13.5	13.5	12.7	12.2	13.4	12.9	13.44
Rata-rata :												**14.74**

Tabel 21 Pengujian Menggunakan 2 Kata (Algoritma *Raita*)

No.	Kata yang dicari	Percobaan Algoritma *Raita* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Art Therapy	12.1	13.2	13.7	15.6	14.2	15.2	13.4	14.2	12.3	12.1	13.60
2	Bettelheim Bruno	13.4	14.7	13.5	14.9	15.2	15.6	14.3	14.1	14.6	15.3	14.56
3	Down Syndrome	14.3	14.9	14.8	13.7	15.1	14.2	14.1	15.3	15.4	15.2	14.70
4	Episodic Memory	12.9	13.7	13.8	10.5	12.3	12.5	13.7	13.2	13.5	13.6	12.50
5	Identitas Gender	11.6	11.4	11.7	11.8	12.1	12	13.3	12.4	12	11.9	12.02
6	Insanity Defense	12.5	12.2	12.3	12.6	12.1	11.2	10.2	12.7	11.5	11.8	11.91
7	Jet Lag	18.7	18.6	17.2	16.6	17.3	15.1	15.8	15.78	15.8	15.8	16.67
8	Usher Syndrome	15.2	14.4	15.9	16.1	14.6	16.3	15.7	14.8	14	14.2	15.12
9	Waking Hypnosis	13.7	14.9	15.3	14.8	14.4	15.2	13.1	13.5	14.3	13.3	14.25
10	Zeigarnik Effect	13.7	12.5	11.9	13	12.9	12.5	13.3	13	13.2	12.6	12.86
Rata-rata :												**13.82**

Berdasarkan pengujian pencarian dengan menggunakan 2 kata pada masing-masing istilah diperoleh hasil rata-rata waktu untuk algoritma *Horspool* adalah 14.74 ms, sedangkan untuk algoritma *Raita* adalah 13.83 ms. Berdasarkan hasil tersebut pada pengujian pencarian dengan menggunakan 2 kata, algoritma *Raita* lebih cepat dibandingkan algoritma *Horspool* dengan selisih waktu kecepatan yang sangat kecil yaitu 0.91 ms.

3) Pengujian Pencarian dengan Menggunakan 3 Kata

Pengujian ini dilakukan untuk memeriksa ketepatan dan kecepatan pencarian kata istilah psikologi yang akan dicari oleh *user*. Pengujian ini dilakukan dengan memasukkan tiga kata yang akan dicari ke dalam *form* aplikasi untuk menguji performa dari aplikasi. Tabel 22 menunjukkan hasil pengujian menggunakan 3 kata (Algoritma *Horspool*) dan Tabel 23 menunjukkan hasil pengujian menggunakan 3 kata (Algoritma *Raita*).

Tabel 22 Pengujian Menggunakan 3 Kata (Algoritma *Horspool*)

No.	Kata yang dicari	Percobaan Algoritma *Horspool* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Battered Child Syndrome	11.9	11.7	11.3	11.1	13.1	12.4	12.2	12.7	12.7	11.4	12.05
2	Client Centered Therapy	11.6	11.7	12.8	11.4	13.0	11.6	11.8	11.3	12.2	12.7	12.01
3	Free Radical Theory	13.4	13.8	13.6	12.5	12.1	12.5	13.0	13.2	13.5	13.1	13.07
4	Maternal Blood Screening	11.7	12.0	11.4	10.3	12.1	11.3	10.9	11.6	10.4	11.7	11.34
5	Normative Life Event	10.8	13.6	13.5	12.0	12.7	11.1	10.5	10.0	11.5	10.2	11.59
6	Post Partum Depression	10.6	10.4	11.2	11.3	11.9	11.5	10.6	11.8	11.2	10.7	11.12
7	Short Term Memory	10.7	12.7	11.1	11.8	11.7	11.6	13.9	11.8	11.2	11.4	11.79
8	Stability Change Issue	11.8	11.7	11.6	11.7	10.7	11.5	11.8	12.3	12.2	12.7	11.80
9	Uncued Panic Attack	11.6	12.5	11.9	12.4	12.5	13.5	12.5	12.3	12.3	12.8	12.43
10	Weapon Fokus Effect	12.8	11.5	12.5	11.6	12.8	12.5	12.7	12.4	12.5	12.0	12.33
Rata-rata :												**11.95**

Berdasarkan pengujian pencarian dengan menggunakan 3 kata pada masing-masing istilah diperoleh hasil rata-rata waktu untuk algoritma *Horspool* adalah 11.95 ms, sedangkan untuk algoritma *Raita* adalah 11.46 ms. Berdasarkan hasil tersebut pada pengujian pencarian dengan meggunakan 3 kata, algoritma *Raita* lebih cepat dibandingkan algoritma *Horspool* dengan selisih waktu kecepatan 0.56 ms.

4) Pengujian Pencarian Kata Tidak Terdapat di Aplikasi

Pengujian ini dilakukan untuk memeriksa ketepatan dan kecepatan pencarian kata istilah psikologi yang akan dicari oleh *user*. Pengujian ini dilakukan dengan memasukkan kata yang tidak terdapat di aplikasi kamus

bahasa istilah untuk menguji performa dari aplikasi. Tabel 24 menunjukkan hasil pengujian pencarian kata yang tidak terdapat pada aplikasi (Algoritma *Horspool*) dan Tabel 25 menunjukkan hasil pengujian pencarian kata yang tidak terdapat pada aplikasi (Algoritma *Raita*).

Tabel 23 Pengujian Menggunakan 3 Kata (Algoritma *Raita*)

No.	Kata yang dicari	Percobaan Algoritma *Raita* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Battered Child Syndrome	10.0	11.8	12.0	10.2	11.7	12.3	12.7	12.7	11.5	11.2	11.61
2	Client Centered Therapy	10.4	10.9	11.9	11.2	12.7	10.5	12.0	10.3	11.4	11.1	11.24
3	Free Radical Theory	10.1	10.7	11.3	12.7	12.3	11.7	11.2	11.8	12.2	11.1	11.51
4	Maternal Blood Screening	11.7	11.5	10.8	10.8	11.3	11.1	11.5	11.1	11.5	10.2	11.15
5	Normative Life Event	11.5	12.2	12.5	13.0	12.2	11.5	10.2	11.3	10.3	11.5	11.62
6	Post Partum Depression	11.8	10.8	11.5	11.3	11.0	11.8	11.5	12.2	11.5	11.9	11.53
7	Short Term Memory	11.9	12.5	12.6	12.2	12.2	12.4	11.0	11.3	12.9	12.1	12.11
8	Stability Change Issue	11.5	11.4	12.0	11.1	11.2	11.4	10.5	11.6	10.6	11.3	11.26
9	Uncued Panic Attack	9.7	12.1	12.2	10.1	12.8	12.2	11.1	12.2	11.1	11.3	11.48
10	Weapon Fokus Effect	10.4	11.5	11.6	10.4	11.2	12.1	11.8	10.1	11.2	10.4	11.07
Rata-rata :												**11.46**

Tabel 24 Pengujian Pencarian Kata yang Tidak Terdapat pada Aplikasi (Algoritma *Horspool*)

No.	Kata yang dicari	Percobaan Algoritma *Horzpool* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Depreszions	14.0	15.2	15.8	14.6	15.5	16.3	15.5	16.3	16.3	16.5	15.60
2	Pengobatan	16.0	16.5	16.8	15.8	15.2	17.6	16.9	16.8	16.8	15.3	16.37
3	Coding	18.2	20.1	20.2	18.4	17.2	18.5	18.4	18.3	20.6	18.3	18.82
4	Gangguan saraf	13.1	13.9	14.5	13.5	14.6	14.7	13.8	13.5	13.3	13.8	13.87
5	Saraf Terjepit	14.6	15.0	13.9	13.8	13.6	14.1	13.8	13.8	14.4	14.9	14.19
6	Gila	25.2	26.7	25.3	26.0	23.2	26.8	25.3	25.1	25.0	25.4	25.40
7	Terapi Keterbelakangan Mental	13.3	11.9	10.7	12.7	15.7	11.5	11.4	11.5	12.0	11.5	12.22
8	Obat Saraf	'15.79	15.2	15.6	17.7	16.3	16.6	17.1	17.9	15.0	16.9	16.48
9	Pendeteksi Kejiwaan	12.7	12.1	12.0	11.8	12.2	12.7	12.6	12.2	13.0	12.7	12.40
10	Medical	20.3	19.3	19.4	20.7	18.0	21.0	15.9	17.1	20.6	19.6	19.19
Rata-rata:												**16.45**

Berdasarkan pengujian pencarian kata yang tidak terdapat pada aplikasi. Masing-masing istilah diperoleh hasil rata-rata waktu untuk algoritma *Horspool* adalah 16.45 ms, sedangkan untuk algoritma *Raita* adalah 14.22 ms. Berdasarkan hasil tersebut pada pengujian pencarian pada kata yang tidak terdapat pada aplikasi, algoritma *Raita* lebih cepat dibandingkan algoritma *Horspool* dengan selisih waktu kecepatan 1.85 ms. Tabel 26 menunjukkan rata- rata dari semua hasil pengujian pencarian kata (Algoritma *Horspool* dan Algoritma *Raita*).

Tabel 25 Pengujian Pencarian Kata yang Tidak Terdapat pada Aplikasi (Algoritma *Raita*)

No.	Kata yang dicari	Percobaan Algoritma *Raita* Ke-										Rata-rata (ms)
		1	2	3	4	5	6	7	8	9	10	
1	Depreszions	12.3	13.8	12.7	13.3	13.7	12.7	12.5	14.4	15.8	15.6	13.68
2	Pengobatan	18.1	18.5	18.0	16.7	17.2	17.4	17.7	16.5	17.7	17.3	17.51
3	Coding	12.5	12.4	12.7	13.9	13.6	14.7	14.8	13.2	13.6	14.5	13.59
4	Gangguan saraf	12.7	13.7	13.4	13.7	14.1	13.0	12.7	11.2	12.7	12.9	13.01
5	Saraf Terjepit	13.2	13.2	14.1	14.9	14.9	14.1	14.9	12.9	13.1	13.5	13.88
6	Gila	15.4	16.4	17.3	17.1	14.2	14.9	16.7	17.6	17.2	15.9	16.27
7	Terapi Keterbelakangan Mental	11.3	11.5	10.9	12.1	13.5	15.0	11.5	12.5	12.3	12.1	12.27
8	Obat Saraf	13.3	14.3	15.3	12.5	13.3	14.0	13.4	15.4	15.3	14.7	14.15
9	Pendeteksi Kejiwaan	14.9	14.3	13.7	12.2	11.2	12.9	12.2	12.7	13.2	12.1	12.94
10	Medical	15.5	14.6	15.3	13.4	15.7	14.8	14.0	15.6	13.7	15.9	14.85
Rata-rata:												**14.22**

Tabel 26 Rata- rata dari Semua Hasil Pengujian Pencarian Kata (Algoritma *Horspool* dan Algoritma *Raita*)

Algoritma	Pengujian Pencarian Kata			
	1 Kata	2 Kata	3 Kata	Tidak ada kata
Horspool	16.01 *ms*	14.74 *ms*	11.95 *ms*	16.45 *ms*
Raita	14.88 *ms*	13.82 *ms*	11.46 *ms*	14.22 *ms*

Keterangan :
Kuning = Lebih tinggi
Hijau = Lebih rendah

Berdasarkan pengujian yang telah dilakukan pada pencarian 1 kata, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 16.01 ms dan algoritma *Raita* selama 14.88 ms. Pada pencarian 2 kata, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 14.74 ms dan algoritma *Raita* selama 13.82 ms. Pada pencarian 3 kata, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 11.95 ms dan algoritma *Raita* selama 11.46 ms, dan untuk pencarian kata yang tidak terdapat dalam aplikasi, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 16.45 ms dan algoritma *Raita* selama 14.22 ms. Hal ini membuktikan bahwa algoritma *Raita* melakukan pencarian *string* lebih cepat dibandingkan dengan algoritma *Horspool.*

4. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan mengenai Perbandingan Algoritma *Horspool* dan Algoritma *Raita* pada Aplikasi Kamus Psikologi berbasis Android, maka diperoleh kesimpulan yaitu:

1. Algoritma *Horspool* melakukan pergeseran karakter sebanyak satu karakter paling kanan apabila cocok melakukan pergeseran karakter berikutnya ke kiri, sedangkan algoritma *Raita* melakukan pergeseran karakter sebanyak satu karakter paling kanan apabila cocok melakukan pergeseran ke awal karakter dan tengah karakter. Hal ini menunjukkan bahwa kedua algoritma tersebut memiliki persamaan dalam pencarian karakter pada pencocokan karakter paling kanan dari teks.
2. Total rata-rata kecepatan waktu untuk algoritma *Horspool* adalah 14.79 ms dan algortima *Raita* adalah 13.60 ms. Pada skenario pencarian 1 kata, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 16.01 ms dan algoritma *Raita* selama 14.88 ms, pada pencarian 2 kata, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 14.74 ms dan algoritma *Raita* selama 13.82 ms, pada pencarian 3 kata, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 11.95 ms dan algoritma *Raita* selama 11.46 ms, dan untuk pencarian kata yang tidak terdapat dalam aplikasi, algoritma *Horspool* memiliki rata-rata kecepatan waktu selama 16.45 ms dan algoritma *Raita* selama 14.72 ms. Hal ini membuktikan bahwa algoritma *Raita* melakukan pencarian *string* lebih cepat dibandingkan algoritma *Horspool*.
3. Hasil kompleksitas algoritma *Horspool* adalah $T(n) = \Theta(MN)$, sedangkan hasil kompleksitas algoritma *Raita* adalah $T(n) = \Theta(M(N-2))$. Maka pencarian dengan menggunakan algoritma *Horspool* dan algoritma *Raita* pada kondisi terburuk, algoritma *Raita* lebih baik daripada algoritma *Horspool*.

5. SARAN

Adapun beberapa saran untuk pengembangan lebih lanjut dari aplikasi Perbandingan Algoritma *Horspool* dan Algoritma *Raita* pada Aplikasi Kamus Istilah Psikologi berbasis *Android*, yaitu:

1. Algoritma yang digunakan pada aplikasi kamus ini dapat dikembangkan dengan algoritma atau metode yang lain.
2. Penerapan algoritma *Horspool* dan algoritma *Raita* ini dapat dikembangkan bukan hanya pada aplikasi kamus saja, tetapi bisa juga diterapkan pada aplikasi pencarian kata lainnya

DAFTAR PUSTAKA

[1] V. Mutiawani, Juwita, and Irvanizam, "Aplikasi Kamus Dwibahasa Aceh-Indonesia Berbasis Java untuk Telepon Genggam," 2011. [Online]. Available: http://www.informatika.unsyiah.ac.id/irvanizam/publications/kamusaceh-indonesia.pdf. [Accessed: 10-Nov-2018].

[2] K. Kesehatan, "Riset Kesehatan Dasar," 2013. [Online]. Available: http://www.depkes.go.id/resources/download/*general*/Hasil Riskesdas 2013.pdf. [Accessed: 15-Nov-2018].

[3] R. R. Valba, "Perbandingan Algoritma Horspool dan Algoritma Raita pada Aplikasi Kamus Bahasa Indonesia - Jerman Berbasis Web," Universitas Sumatera Utara, 2017.

[4] A. R. Ekawati, D. P. Danarjati, and A. R. Ekawati, *Pengantar Psikologi Umum*, 1st ed. Yogyakarta: Graha Ilmu, 2013.

[5] J. P. Sembiring, "Perancangan Aplikasi Kamus Bahasa Indonesia - KARO Online Berbasis Web dengan Metode Sequential Search," *Pelita Inform. Budi Darma*, Vol. 4, No. 2, pp. 28–33, 2013.

[6] R. N. Horspool, "Practical Fast Searching in Strings," *Softw. Pract. Exp.*, Vol. 10, pp. 501–506, 1980.

[7] H. N. Verma and R. Singh, "A Fast String Matching *Algorithm*," *Technol. Appl.*, Vol. 2, No. 6, pp. 1877–1883, 2011.

[8] C. Charras and T. Lecroc, *Handbook of Exact String-Matching Algorithms*.

United Kingdom: Oxford University Press.

[9] U. N. Azizah, “Perbandingan Detektor Tepi Prewit dan Detektor Tepi Laplacian Berdasarkan Kompleksitas Waktu dan Citra Hasil,” Universitas Pendidikan Indonesia, 2013.