Penerapan Algoritma Pencocokan String dan Regex untuk Memberikan Rekomendasi Berita Berdasarkan Riwayat

Rozan Fadhil Al Hafidz – 13520039
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520039@std.stei.itb.ac.id

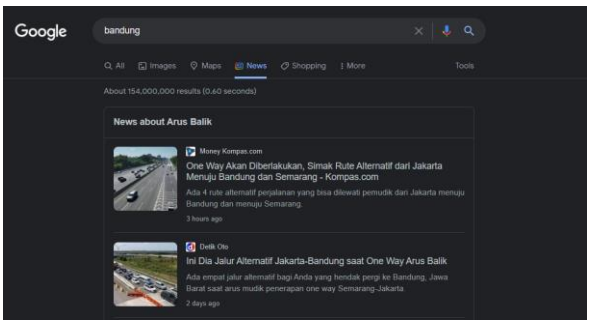
***Abstrak*—Mengonsumsi berita sudah menjadi salah satu rutinitas bagi sebagian orang. Saat ini, berita bisa didapatkan dari banyak sumber, seperti koran, televisi, bahkan internet. Berita juga memiliki banyak kategori, seperti berita olahraga, berita politik, bahkan berita di dunia hiburan. Selain kategori umum, berita juga bisa dibagi menjadi kategori khusus, misalnya berita olahraga sepakbola di Indonesia. Oleh karena itu, penulis ingin mencoba mempraktikkan algoritma pencocokan string dan regex untuk membuat program yang dapat memberikan rekomendasi berita berdasarkan riwayat berita yang pernah dibaca.**

Kata kunci—pencocokan string; regex; rekomendasi.

I. PENDAHULUAN

Menurut KBBI, berita adalah cerita atau keterangan mengenai kejadian atau peristiwa yang hangat. Berita merupakan sumber informasi untuk mengetahui kejadian yang terjadi di bagian dunia lain. Pada zaman dahulu, berita yang dapat dikonsumsi manusia dengan mudah hanya sebatas satu daerah saja. Namun, seiring perkembangan zaman, berita dari luar daerah dapat dikonsumsi melalui koran dan televisi. Pada saat ini, berita dari seluruh dunia dapat dikonsumsi dengan mudah kapan pun dan di mana pun dengan menggunakan internet.

Di dalam internet, ada sangat banyak pilihan berita yang bisa dikonsumsi manusia. Beberapa situs *online* yang menyediakan berita di Indonesia yaitu detikNews, CNN Indonesia, Tekno Tempo, dan Kompas. Semua situs tersebut dapat dibuka secara gratis dengan cara langsung ke situs web masing-masing ataupun melalui Google.



Gambar. 1. Contoh hasil pencarian berita di Google.
(Sumber : Penulis)

Berita juga dapat dikategorikan berdasarkan isinya, misalnya berita olahraga, berita politik, dan bahkan berita di dunia hiburan. Kategori tersebut bisa dibagi lagi menjadi kategori yang lebih spesifik, misalnya berita olahraga bisa dikategorikan lagi menjadi berita sepakbola, berita bulu tangkis, berita bola basket, dan sebagainya.

Dengan banyaknya kategori tersebut, setiap orang pasti mempunyai preferensi dalam memilih berita yang akan dikonsumsi. Google menggunakan *artificial intelligence* yang kompleks untuk memberikan rekomendasi berita kepada penggunanya berdasarkan riwayat berita yang pernah dibaca. Oleh karena itu, pengguna Google bisa mendapatkan berita sesuai dengan keinginannya dengan mudah.

Pada makalah ini, penulis mencoba membuat program yang memberikan rekomendasi berita sederhana berdasarkan riwayat berita yang pernah dibaca dengan menggunakan algoritma pencocokan string Boyer Moore dengan bantuan *regular expression*. *Regular expression* digunakan untuk memfilter kata partikel pada judul berita, kemudian algoritma Boyer-Moore digunakan untuk mencocokan kata kunci yang diambil dari riwayat dengan kata kunci berita yang tersedia. Algoritma Boyer-Moore dipilih karena isi judul berita merupakan bahasa sehari-hari yang memiliki banyak alfabet, sehingga algoritma Boyer-Moore lebih efektif.

II. LANDASAN TEORI

A. *Pencocokan String (String Matching)*

String merupakan salah satu struktur data di dalam pemrograman yang berisi deretan karakter (angka, huruf, karakter special) [1]. Pencocokan string merupakan algoritma untuk mengecek kemunculan sebuah string pattern dengan panjang n pada sebuah string text dengan panjang m, dengan n bernilai lebih kecil atau sama dengan m. Ada banyak algoritma untuk melakukan pencocokan string. Algoritma tersebut memiliki keunggulan dan kelemahan masing-masing. Berikut ini adalah beberapa algoritma pencocokan string [2].

- Algoritma Brute force
- Algoritma Boyer-Moore (BM)
- Algoritma Knuth-Morris-Pratt (KMP)
- Algoritma Automation Matcher
- Algoritma Aho-Corasick

B. *Algoritma Boyer-Moore*

Algoritma Boyer-Moore merupakan salah satu algoritma yang populer untuk melakukan pencocokan string (*string matching*). Misalkan *pattern* P akan dicocokkan dengan teks T. Algoritma Boyer-Moore melakukan pencocokan string menggunakan dua teknik [3], yaitu :

1. Teknik *looking-glass*
   Teknik *looking-glass* merupakan metode untuk melakukan proses pencocokan string dimulai dari akhir dan bergerak ke depan. Jika karakter T[i] tidak sama dengan karakter P[j], maka *pattern* digeser dengan menggunakan teknik *character-jump*.

2. Teknik *character-jump*
   Dilakukan ketika karakter yang dicocokkan tidak sama. Misalkan huruf yang tidak sama pada T adalah x. Terdapat 3 kasus
   2.1. Kasus 1 :
   Jika P mengandung x, cobalah geser P ke kanan untuk menyesuaikan posisi x di P dengan T[i]
   2.2. Kasus 2 :
   Jika P mengandung x, namun penggeseran ke kemunculan terakhir tidak mungkin, maka geser P sebesar 1 karakter ke T[i+1]
   2.3. Kasus 3 :
   Jika kasus 1 dan kasus 2 tidak memenuhi, geser P sehingga menyesuaikan P[0] dengan T[i+1]

   Pada kasus 1 dan 2, algoritma BM membutuhkan informasi mengenai kemunculan terakhir sebuah karakter pada T dalam pola P. Oleh karena itu, sebelum menjalankan pengecekan string, mula-mula algoritma ini melakukan pembuatan larik yang menyimpan kemunculan terakhir elemen pada P dengan menggunakan fungsi *Last Occurrence* L(x).

   Misalkan P sebagai berikut

a	b	a	c	a	b
0	1	2	3	4	5

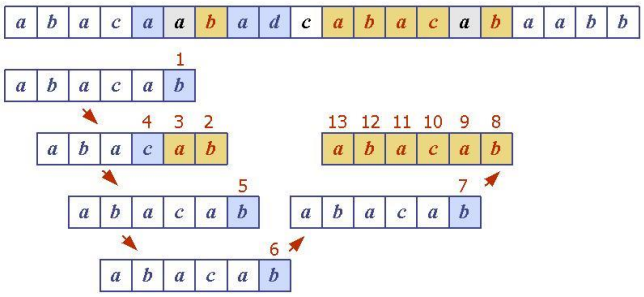
   Dengan A = {a, b, c, d}

   Maka larik yang berisi hasil dari fungsi *Last Occurrence* sebagai berikut

x	a	b	c	d
L(x)	4	5	3	-1

   Larik tersebut digunakan untuk menentukan lokasi pergeseran.

Berikut ini adalah contoh pencocokan string dengan menggunakan Boyer-Moore dalam melakukan pencarian *pattern* “abacab” pada teks “abacaabadcabacabaabb"



Gambar. 2. Contoh Algoritma Boyer-Moore
Sumber :
https://koding4fun.files.wordpress.com/2010/05/complete_example.jpg

Penjelasan :

1. Ketidakcocokan terjadi pada saat teks menunjuk ke karakter `a` dan *pattern* menunjuk ke karakter `b`. *Pattern* mengandung huruf `a` (kasus 1). Geser *pattern* sehingga menyesuaikan karakter `a` pada teks dengan karakter `a` pada *pattern*.
4. Ketidakcocokan terjadi pada saat teks menunjuk ke karakter `a` dan *pattern* menunjuk ke karakter `c`. *Pattern* mengandung huruf `a` tetapi *pattern* tidak boleh bergerak mundur (kasus 2). Geser *pattern* sejauh satu langkah.
5. Ketidakcocokan terjadi pada saat teks menunjuk ke karakter `a` dan *pattern* menunjuk ke karakter `b`. *Pattern* mengandung huruf `a` (kasus 1). Geser *pattern* sehingga menyesuaikan karakter `a` pada teks dengan karakter `a` pada *pattern*.
6. Ketidakcocokan terjadi pada saat teks menunjuk ke karakter `d` dan *pattern* menunjuk ke karakter `b`. *Pattern* tidak mengandung huruf `d` (kasus 3). Geser *pattern* ke kanan sejauh panjang *pattern*.
7. Ketidakcocokan terjadi pada saat teks menunjuk ke karakter `a` dan *pattern* menunjuk ke karakter `b`.

Pattern mengandung huruf `a` (kasus 1). Geser *pattern* sehingga menyesuaikan karakter `a` pada teks dengan karakter `a` pada *pattern*.

8. Teks cocok dengan *pattern*.

C. *Regular Expression (Regex)*

Regular Expression atau yang biasa disingkat sebagai regex adalah pola yang mendeskripsikan sejumlah teks tertentu [4]. Regex digunakan untuk mencari, mengganti, memanipulasi, atau memproses suatu string. Selain itu, regex juga dapat digunakan untuk memvalidasi string. Berikut ini adalah beberapa aturan dalam menulis regex.

1. *Groups* dan *Ranges*

Karakter	Makna
x|y	Cocok dengan "x" atau "y"
[xyz]	Cocok dengan "x", "y", atau "z"
[x-z]	Cocok dengan "x" sampai "z" (sama dengan [xyz])
[^xyz]	Cocok dengan selain "x", "y", atau "z"
[^x-z]	Cocok dengan selain "x" sampai "z" (sama dengan [^xyz])
[a-zA-Z]	Cocok dengan "a" sampai "z" dan "A" sampai "Z"

2. *Character Classes*

Karakter	Makna
.	Cocok dengan sebuah karakter apapun selain garis baru
\d	Cocok dengan bilangan
\D	Cocok dengan non-bilangan
\w	Cocok dengan karakter alfanumerik
\W	Cocok dengan karakter non-alfanumerik
\s	Cocok dengan spasi tunggal
\S	Cocok dengan non-spasi tunggal

3. *Quantifiers*

Karakter	Makna
x?	Cocok jika x muncul sekali atau tidak sama sekali
x*	Cocok jika x muncul 0 kali atau lebih
x+	Cocok jika x muncul 1 kali atau lebih
x{n}	Cocok jika x muncul tepat n kali
x{n,}	Cocok jika x muncul setidaknya n kali
x{n,m}	Cocok jika x muncul antara n sampai m kali

4. *Boundary Matchers*

Karakter	Makna
^	Awal baris
$	Akhir baris
\b	Batas kata
\B	Batas bukan kata
\G	Akhir kecocokan sebelumnya
\Z	Akhir dari input tapi bukan final terminator jika ada
\z	Akhir dari input

Berikut ini adalah contoh ekspresi regex untuk melakukan validasi email

```
/^([a-zA-Z0-9_\-\.]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})$/
```

Penjelasan dari ekspresi regex tersebut adalah sebagai berikut:

- ^([a-zA-Z0-9_\-\.]+)
  Ekspresi tersebut cocok dengan karakter di awal string termasuk salah satu dari :
  - Huruf kecil (a sampai z)
  - Huruf kapital (A sampai Z)
  - Angka (0 sampai 9)
  - Garis bawah (_)
  - Strip (-)
  - Titik (.)

  Dan berjumlah satu atau lebih

- @
  Ekspresi tersebut cocok dengan karakter at (@)

- \.
  Ekspresi tersebut cocok dengan karakter titik (.)

- ([a-zA-Z]{2,5})$
  Ekspresi tersebut cocok dengan karakter di akhir string termasuk salah satu dari :
  - Huruf kecil (a sampai z)
  - Huruf kapital (A sampai Z)

  Dan berjumlah antara 2 sampai 5

Contoh string yang diterima ekspresi regex tersebut adalah "abc@yyy.zzz" dan contoh string yang akan ditolak ekspresi tersebut adalah "www.com".

III. Implementasi dan Pengujian

Pada makalah ini, saya mengimplementasikan program dalam bahasa Python. Saya menggunakan data sampel judul berita untuk pengujian dari https://data.jakarta.go.id/.

A. Menerapkan Algoritma Boyer-Moore

Pertama-tama, saya menerapkan algoritma Boyer-Moore dalam bahasa Python. Terdapat dua fungsi untuk menerapkan algoritma Boyer-Moore, yaitu:

1. buildLast(pattern)

   Fungsi buildLast adalah fungsi untuk mencari kemunculan terakhir tiap huruf pada *pattern* yang akan dicek. Berikut ini adalah implementasi fungsi buildLast pada bahasa Python

```
def buildLast(pattern : str):
    '''
    Fungsi untuk membuat array yang mengandung kemunculan terakhir tiap huruf dalam pattern

    Parameters:
        pattern (str): pattern yang akan dicek
    Return:
        (int[]) array yang mengandung kemunculan terakhir tiap huruf dalam pattern
    '''
    last = [-1 for _ in range(128)] # 128 adalah jumlah karakter ASCII yang digunakan
    for i in range(len(pattern)):
        last[ord(pattern[i])] = i   # ord() untuk mengubah karakter menjadi kode ASCII

    return last
```

2. bmMatch(pattern, text)

   Fungsi bmMatch adalah fungsi utama pada algoritma Boyer-Moore. Fungsi ini digunakan untuk mecari indeks kemunculan pertama pattern pada teks. Jika pattern tidak muncul sama sekali, fungsi ini akan mengembalikan nilai -1. Berikut ini adalah fungsi bmMatch dalam bahasa Python.

```
def bmMatch(text : str, pattern : str):
    '''
    Fungsi untuk mencari pattern pada text dengan algoritma Boyer-Moore
    Parameters:
        teks (str): teks yang akan dicek
        pattern (str): pattern yang akan dicek
    Return:
        (int) index teks yang sesuai dengan pattern, -1 jika tidak ada
    '''

    # buat array yang mengandung kemunculan terakhir tiap huruf dalam pattern
    last = buildLast(pattern)

    # buat variabel yang mengandung panjang pattern
    n = len(text)
    m = len(pattern)
    i = m - 1

    # Cek apakah pattern lebih panjang dari text
    if i > n - 1:
        return -1

    # Lakukan pencarian pattern pada teks
    j = m - 1
    while i <= n - 1:
        # cek apakah huruf pada text sama dengan huruf pada pattern
        if text[i] == pattern[j]:
            # jika j == 0, artinya pattern sudah sesuai dengan teks
            if j == 0:
                return i
            # jika j != 0, lakukan pengecekan huruf selanjutnya
            else:
                i -= 1
                j -= 1
        # jika huruf pada text tidak sama dengan huruf pada pattern
        else:
            # lompat ke huruf selanjutnya sesuai dengan last ocurrence
            lo = last[ord(text[i])]
            i = i + m - min(j, 1 + lo)
            j = m - 1
    # jika tidak ada pattern yang sesuai, return -1
    return -1
```

B. Pembuatan Kamus Pengguna

Langkah selanjutnya adalah membuat kamus pengguna yang berisi daftar kata yang berasal dari berita yang pernah dibaca pengguna.

```
daftarJudulSaya = []
daftarKataJudulSaya = {}
```

Sebelum membuat fungsi untuk menambahkan daftar kata, buat ekspresi regular expression untuk memfilter judul berita agar kata pada judul berita yang berawalan dengan huruf kecil tidak dimasukkan ke dalam kamus. Berikut ini adalah ekspresi regular ekspression yang saya buat.

```
/^[A-Z]/
```

Regular expression tersebut cocok dengan kata yang berawalan dengan huruf kapital (antara A sampai Z). Setelah itu, buat fungsi untuk menambahkan daftar kata ke dalam kamus sebagai berikut.

```
def tambahDaftarKataSaya(judul, daftarKataSaya):
    '''
    Fungsi untuk menambahkan kata-kata dari judul ke daftar kata-kata yang ada
```

```
    Parameters:
        judul (str): judul berita
        daftarKataSaya (dict): daftar kata-kata yang
ada
    '''
    for kata in judul.split():
        # Tambahkan kata jika kata tersebut tidak
diawali dengan huruf kecil
        if re.match("^[A-Z]", kata):
            if kata in daftarKataSaya:
                daftarKataSaya[kata] += 1
            else:
                # Buat kata baru jika kata tersebut
belum ada
                daftarKataSaya[kata] = 1
```

Langkah selanjutnya adalah menyiapkan data uji. Saya menggunakan data judul berita daerah DKI Jakarta tahun 2021 yang didapatkan dari tautan berikut

https://data.jakarta.go.id/read-resource/json/data-judul-berita-yang-di-publish-pada-website-berita-resmi-pemprov-dki-jakarta-tahun-2021/b1e3ea7bb5d907d1b9a2870079ebb92e

Data tersebut berisi 581 judul berita. Berikut ini adalah contoh judul berita yang didapat dari tautan tersebut

```
0 UPPL DKI Gelar Sosialisasi Jakarta Kota Bebas Pungli di Kantor DPMPTSP
1 23 HPR di Karang Anyar Divaksin Rabies
2 Perkembangan Data Kasus dan Vaksinasi COVID-19 di Jakarta per 30 September 2021
3 Perolehan ZIS di Jaktim Mencapai 75 Persen
4 Bangkitkan Pariwisata di Masa Pandemi, Gubernur Anies Resmikan Toraja and Beyond
Tourism Week 2021
5 PD Dharma Jaya Sepakati Kerja Sama Produk Olahan Daging Siap Saji
6 Minggu, Ada Rekayasa Lalu Lintas di Jl I Gusti Ngurah Rai
7 174 HPR di Tebet Barat Divaksin Rabies
8 RSUD Cilincing Adakan Layanan Jemput Bola Pemeriksaan Mata Gratis
9 288 Lansia di Kelurahan Kwitang Terima Bansos KLJ
```

Gambar. 3. Contoh Judul Berita yang Digunakan
Sumber Gambar : Penulis

Buat fungsi untuk memisahkan kata-kata yang terdapat pada judul dengan menggunakan regex.

```
def pisahJudul(judul):
    '''
    Fungsi untuk memisahkan judul berita menjadi kata-
kata
    Parameters:
        judul (str): judul berita
    Return:
        (list) daftar kata-kata
    '''
    kataJudul = judul.split()
    kataJudulFilter = []
    for kata in kataJudul:
        if re.match("^[A-Z]", kata):
            kataJudulFilter.append(kata)
    return kataJudulFilter
```

Kemudian, buat fungsi untuk mencari rekomendasi berita berdasarkan kamus pengguna sebagai berikut.

```
def cari5RekomendasiBerita(daftarKataJudulSaya,
daftarJudul):
    '''
    Fungsi untuk mencari 5 rekomendasi berita
berdasarkan riwayat berita yang sudah dibaca
sebelumnya
    Parameters:
        daftarKataJudulSaya (dict): daftar kata-kata
yang ada
        daftarJudul (list): daftar judul berita
    Return:
        (list) daftar judul berita yang sesuai
    '''
    rekomendasi = []
    count = 0

    kata_terbanyak = max(daftarKataJudulSaya.keys(),
key=(lambda k: daftarKataJudulSaya[k]))

    for i in range(len(daftarJudul)):
        kataJudul = pisahJudul(daftarJudul[i])
        for kata in kataJudul:
            if bmMatch(kata, kata_terbanyak) != -1:
                rekomendasi.append(daftarJudul[i])
                count += 1
                if count == 5:
                    return rekomendasi
                else :
                    continue

    return rekomendasi
```

Fungsi tersebut menggunakan algorimta Boyer-Moore untuk melakukan pencocokan string. Selain karena lebih cepat, algoritma Boyer-Moore juga bisa mengecek apakah teks (kata pada judul berita) mengandung substring *pattern* (kata terbanyak). Fungsi tersebut mencari 5 berita teratas yang mengandung kata yang paling sering dijumpai pada berita yang pernah dibaca.

C. Pengujian Kamus Pengguna

Langkah selanjutnya adalah menguji kamus yang sudah dibuat. Prosedur pengujiannya adalah saya memasukkan judul-judul berita yang akan diproses oleh fungsi tambahDaftarKataSaya. Berikut ini adalah judul-judul berita yang saya masukkan untuk pengujian

```
Perkembangan Data Kasus dan Vaksinasi COVID-19 di Jakarta
per 8 September 2021

Waspada Potensi Hujan Disertai Angin Kencang di Jakpus,
Jakbar, Jaksel dan Jaktim

Wagub Ariza Apresiasi Kolaborasi NU Gelar Kegiatan 500
Vaksinasi COVID-19 Dosis Kedua

160 Relawan Nakes COVID-19 City Tour ke Sejumlah
Destinasi Wisata di Jakarta
```

```
Disdik DKI Tidak Temukan Kasus Positif pada 25 Sekolah Klaster COVID-19
```

Berikut ini adalah hasil yang didapatkan

```
Masukkan judul berita: Perkembangan Data Kasus dan Vaksinasi COVID-19 di Jakarta per 8 September 2021
---------------------------------------------------------------
{'Perkembangan': 1, 'Data': 1, 'Kasus': 1, 'Vaksinasi': 1, 'COVID-19': 1, 'Jakarta': 1, 'September': 1}
Masukkan judul berita: Waspada Potensi Hujan Disertai Angin Kencang di Jakpus, Jakbar, Jaksel dan Jaktim
---------------------------------------------------------------
{'Perkembangan': 1, 'Data': 1, 'Kasus': 1, 'Vaksinasi': 1, 'COVID-19': 1, 'Jakarta': 1, 'September': 1, 'Waspada': 1, 'Potensi': 1, 'Hujan': 1, 'Disertai': 1, 'Angin': 1, 'Kencang': 1, 'Jakpus,': 1, 'Jakbar,': 1, 'Jaksel': 1, 'Jaktim': 1}
Masukkan judul berita: Wagub Ariza Apresiasi Kolaborasi NU Gelar Kegiatan 500 Vaksinasi COVID-19 Dosis Kedua
---------------------------------------------------------------
{'Perkembangan': 1, 'Data': 1, 'Kasus': 1, 'Vaksinasi': 2, 'COVID-19': 2, 'Jakarta': 1, 'September': 1, 'Waspada': 1, 'Potensi': 1, 'Hujan': 1, 'Disertai': 1, 'Angin': 1, 'Kencang': 1, 'Jakpus,': 1, 'Jakbar,': 1, 'Jaksel': 1, 'Jaktim': 1, 'Wagub': 1, 'Ariza': 1, 'Apresiasi': 1, 'Kolaborasi': 1, 'NU': 1, 'Gelar': 1, 'Kegiatan': 1, 'Dosis': 1, 'Kedua': 1}
Masukkan judul berita: 160 Relawan Nakes COVID-19 City Tour ke Sejumlah Destinasi Wisata di Jakarta
---------------------------------------------------------------
{'Perkembangan': 1, 'Data': 1, 'Kasus': 1, 'Vaksinasi': 2, 'COVID-19': 3, 'Jakarta': 2, 'September': 1, 'Waspada': 1, 'Potensi': 1, 'Hujan': 1, 'Disertai': 1, 'Angin': 1, 'Kencang': 1, 'Jakpus,': 1, 'Jakbar,': 1, 'Jaksel': 1, 'Jaktim': 1, 'Wagub': 1, 'Ariza': 1, 'Apresiasi': 1, 'Kolaborasi': 1, 'NU': 1, 'Gelar': 1, 'Kegiatan': 1, 'Dosis': 1, 'Kedua': 1, 'Relawan': 1, 'Nakes': 1, 'City': 1, 'Tour': 1, 'Sejumlah': 1, 'Destinasi': 1, 'Wisata': 1}
Masukkan judul berita: Disdik DKI Tidak Temukan Kasus Positif pada 25 Sekolah Klaster COVID-19
---------------------------------------------------------------
{'Perkembangan': 1, 'Data': 1, 'Kasus': 2, 'Vaksinasi': 2, 'COVID-19': 4, 'Jakarta': 2, 'September': 1, 'Waspada': 1, 'Potensi': 1, 'Hujan': 1, 'Disertai': 1, 'Angin': 1, 'Kencang': 1, 'Jakpus,': 1, 'Jakbar,': 1, 'Jaksel': 1, 'Jaktim': 1, 'Wagub': 1, 'Ariza': 1, 'Apresiasi': 1, 'Kolaborasi': 1, 'NU': 1, 'Gelar': 1, 'Kegiatan': 1, 'Dosis': 1, 'Kedua': 1, 'Relawan': 1, 'Nakes': 1, 'City': 1, 'Tour': 1, 'Sejumlah': 1, 'Destinasi': 1, 'Wisata': 1, 'Disdik': 1, 'DKI': 1, 'Tidak': 1, 'Temukan': 1, 'Positif': 1, 'Sekolah': 1, 'Klaster': 1}
Masukkan judul berita:
```

Gambar. 4. Pengujian Kamus Pengguna
Sumber Gambar : Penulis

Dari pengujian tersebut, terlihat kata yang paling sering muncul pada judul berita yang dimasukkan adalah “COVID-19”

D. Pengujian Rekomendasi Berita

Setelah selesai menguji kamus pengguna, selanjutnya adalah menguji pencarian rekomendasi berita menggunakan fungsi yang telah dibuat sebelumnya. Berikut ini adalah hasilnya

```
Rekomendasi berita:
Perkembangan Data Kasus dan Vaksinasi COVID-19 di Jakarta per 30 September 2021
Perkembangan Data Kasus dan Vaksinasi COVID-19 di Jakarta per 29 September 2021
Vaksinasi COVID-19 di Kecamatan Kemayoran Capai 74 Persen
160 Relawan Nakes COVID-19 City Tour ke Sejumlah Destinasi Wisata di Jakarta
Perkembangan Data Kasus dan Vaksinasi COVID-19 di Jakarta per 28 September 2021
```

Gambar. 5. Hasil Pengujian Rekomendasi Berita
Sumber Gambar : Penulis

Berdasarkan hasil tersebut, terlihat bahwa semua berita yang direkomendasikan mengandung kata “COVID-19”. Oleh karena itu, pengujian yang dilakukan kali ini berhasil.

E. Pengujian Kasus Lain

Berikut ini adalah judul-judul berita yang saya masukkan untuk pengujian kasus lain.

```
Normalisasi Saluran Air di Gang Udang Cempaka Baru Capai 73,91 Persen

Disparekraf DKI Sosialisasikan Penggunaan Aplikasi PeduliLindungi kepada Pelaku Usaha Restoran, Rumah Makan dan Kafe

Personel Pasukan Oranye Bersihkan Tali Air di Kemang Selatan

Pemprov DKI Terus Buat Terobosan untuk Capaian Target Layanan Air Bersih

168 Peserta Ikuti Kelas Berkebun Sesi Tiga di Jakpus
```

Berikut ini adalah kamus yang dihasilkan dari kasus pengujian kedua.

```
{'Normalisasi': 1, 'Saluran': 1, 'Air': 3, 'Gang': 1, 'Udang': 1, 'Cempaka': 1, 'Baru': 1, 'Capai': 1, 'Persen': 1, 'Disparekraf': 1, 'DKI': 2, 'Sosialisasikan': 1, 'Penggunaan': 1, 'Aplikasi': 1, 'PeduliLindungi': 1, 'Pelaku': 1, 'Usaha': 1, 'Restoran,': 1, 'Rumah': 1, 'Makan': 1, 'Kafe': 1, 'Personel': 1, 'Pasukan': 1, 'Oranye': 1, 'Bersihkan': 1, 'Tali': 1, 'Kemang': 1, 'Selatan': 1, 'Pemprov': 1, 'Terus': 1, 'Buat': 1, 'Terobosan': 1, 'Capaian': 1, 'Target': 1, 'Layanan': 1, 'Bersih': 1, 'Peserta': 1, 'Ikuti': 1, 'Kelas': 1, 'Berkebun': 1, 'Sesi': 1, 'Tiga': 1, 'Jakpus': 1}
```

Gambar. 6. Pengujian Kamus Pengguna Kasus Kedua
Sumber Gambar : Penulis

Dari pengujian tersebut, terlihat kata yang paling sering muncul pada judul berita yang dimasukkan adalah “Air”. Berikut ini adalah hasil dari pencarian rekomendasi berdasarkan kamus tadi.

```
Rekomendasi berita:
Pengerjaan Saluran Air di Kompleks Angkasa Pura Rampung
Dinas SDA Ajukan Anggaran untuk Subsidi Air Bersih
DKI Raih Emas Pertama PON XX Papua dari Cabor Polo Air Putri
Tiga Pintu Air di Jaktim Dipastikan Berfungsi Normal
Pembangunan Saluran Air di Jl Kikir Kayu Putih Ditarget Rampung Oktober
```

Gambar. 7. Hasil Pengujian Rekomendasi Berita
Sumber Gambar : Penulis

Berdasarkan hasil tersebut, terlihat bahwa semua berita yang direkomendasikan mengandung kata “Air”. Oleh karena itu, pengujian yang dilakukan kali ini juga berhasil.

IV. KESIMPULAN

Melalui makalah ini, saya menyadari algoritma pencocokan string bisa digunakan untuk berbagai macam kegunaan, salah satunya adalah untuk mencari rekomendasi berita. Menemukan rekomendasi berita harus melalui banyak proses

pengecekan string. Oleh karena itu, algoritma *brute force* saja tidak cukup cepat untuk digunakan. Oleh karena itu, pengembangan algoritma pencocokan string merupakan hal yang penting. Jenis pencocokan string yang digunakan adalah algoritma Boyer-Moore karena string yang terdapat pada judul berita merupakan kalimat berbahasa Inggris. Dengan mencatat kata-kata pada judul berita yang pernah dibaca, kita bisa memberikan rekomendasi berita kepada pembaca dengan cara mengecek seluruh judul berita yang ada dan memilih berita dengan judul yang mengandung kata yang paling sering terdapat pada riwayat berita yang pernah dibaca.

V. Ucapan Terima Kasih

Penulis mengucapkan puji syukur kepada Tuhan YME karena atas rahmat dan karunia-Nya, makalah strategi algoritma yang berjudul "Penerapan Algoritma Pencocokan String dan Regex untuk Memberikan Rekomendasi Berita Berdasarkan Riwayat" dapat diselesaikan tepat pada waktunya. Penulis juga mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi, M.T. selaku dosen mata kuliah IF2211 Strategi Algoritma yang telah membimbing penulis dalam memahami materi yang digunakan dalam pembuatan makalah ini. Terakhir, penulis juga mengucapkan terima kasih kepada kedua orang tua dan keluarga yang mendukung penulis dalam mengerjakan makalah ini.

Video Link di Youtube

https://youtu.be/Y7SpbZAXjQc

Referensi

[1] *Introduction to java*. Introduction to java - MFC 158 G. (n.d.). Diakses 7 Mei 2022 dari http://www.acsu.buffalo.edu/~fineberg/mfc158/week10lecture.htm.

[2] *Applications of String Matching Algorithms - GeeksforGeeks*. GeeksforGeeks. (2022). Diakses 7 Mei 2022 dari https://www.geeksforgeeks.org/applications-of-string-matching-algorithms/.

[3] *Pencocokan String.* Rinaldi Munir. 2022. Diakses 7 Mei 2021 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

[4] Goyvaerts, J. (2022). *Regular Expression Tutorial - Learn How to Use Regular Expressions*. Regular-expressions.info. Diakses 7 Mei 2022 dari https://www.regular-expressions.info/tutorial.html.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Mei 2022

Rozan Fadhil Al Hafidz
13520039