

Tugas Kecil 4
IF2211 Strategi Algoritma

**Ekstraksi Informasi dari Artikel Berita dengan Algoritma
Pencocokan String**



Disusun oleh
Indra Febrio Nugroho 13518016

Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2020

BAB I

TEORI SINGKAT

1.1 Algoritma Knuth-Morris-Pratt

Algoritma Knuth-Morris-Pratt merupakan salah satu algoritma pencarian string. Algoritma ini juga dikenal sebagai algoritma KMP (singkatannya), dikembangkan secara terpisah oleh Donald Knuth dan Vaughan Pratt pada tahun 1970, serta secara independen oleh James H. Morris. Ketiga penemu tersebut mempublikasikan algoritma ini pada tahun 1977. Dari nama algoritma ini kita dapat melihat dibuat berdasarkan tiga nama penemunya.

Algoritma Knuth-Morris-Pratt adalah algoritma yang mencari pattern dalam sebuah text dari kiri ke kanan seperti algoritma Brute Force. Algoritma ini melakukan pergeseran pattern lebih “cerdas” dibandingkan dengan brute force.

Secara sistematis, langkah-langkah yang dilakukan algoritma Knuth-Morris-Pratt pada saat mencocokkan string:

1. String pattern (kata yang dicari) akan dipecah menjadi array karakter. String text (teks, artikel) akan dipecah menjadi array karakter
2. Algoritma Knuth-Morris-Pratt mulai mencocokkan pattern pada awal teks.
3. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter pattern dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
 - A. Karakter di pattern dan di teks yang dibandingkan tidak cocok (mismatch).
 - B. Semua karakter di pattern cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini.
4. Algoritma kemudian menggeser pattern berdasarkan tabel next, lalu mengulangi langkah 2 sampai pattern berada di ujung teks.

Kelebihan dari algoritma Knuth-Morris-Pratt selain cepat juga sangat baik digunakan pada file berukuran besar karena pencarian kecocokan tidak perlu kembali ke belakang pada input teks.

Namun algoritma ini memiliki kekurangan yakni efektifitas dari algoritma ini akan berkurang seiring dengan bertambahnya jumlah jenis karakter dari teks.

1.2 Algoritma Boyer-Moore

Algoritme Boyer-Moore adalah salah satu algoritme pencarian string, dipublikasikan oleh Robert S. Boyer, dan J. Strother Moore pada tahun 1977.

Algoritme ini dianggap sebagai algoritme yang paling efisien pada aplikasi umum. Tidak seperti algoritme pencarian string yang ditemukan sebelumnya, algoritme Boyer-Moore mulai mencocokkan karakter dari sebelah kanan pattern. Ide di balik algoritme ini adalah bahwa dengan memulai pencocokan karakter dari kanan, dan bukan dari kiri, maka akan lebih banyak informasi yang didapat

Secara sistematis, langkah-langkah yang dilakukan algoritme Boyer-Moore pada saat mencocokkan string adalah:

1. Algoritme Boyer-Moore mulai mencocokkan pattern pada awal teks.
2. Dari kanan ke kiri, algoritme ini akan mencocokkan karakter per karakter pattern dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
 1. Karakter di pattern dan di teks yang dibandingkan tidak cocok (mismatch).
 2. Semua karakter di pattern cocok. Kemudian algoritme akan memberitahukan penemuan di posisi ini.
3. Algoritme kemudian menggeser pattern dengan memaksimalkan nilai penggeseran good-suffix dan penggeseran bad-character, lalu mengulangi langkah 2 sampai pattern berada di ujung teks.

1.3 Regex

Regular Expression adalah Sebuah cara mendefinisikan language yang lebih tepat dibandingkan dengan menggunakan cara ellipsis (diakhiri dengan ...)

Dengan memanfaatkan closure, bila $S = \{ x \}$ maka $L4 = S^*$ Dapat juga ditulis sebagai $L4 = \{x\}^*$

Kleene Star tidak hanya dapat diaplikasikan untuk set namun juga langsung ke alphabet. Contoh: x^* Ekpresi sederhana x^* akan dipakai untuk mengekspresikan pengulangan dari x (bisa juga tidak sama sekali)

BAB II

KODE PROGRAM

1. KMP.py

```
# 13518016
# Indra Febrio Nugroho
# Knuth-Morris-Pratt Algorithm

def KMPMatch(text, pattern):
    m = len(pattern)
    n = len(text)

    # create container that will hold the longest prefix suffix values for pattern
    lps = [0 for i in range(m)]
    # calculate the lps
    calculateLPS(pattern, lps)

    total = 0 # total matched pattern in the text
    matchedIdx = [] # container for index of matched pattern in the text
    i = 0 # index for text
    j = 0 # index for pattern
    while i < n:
        if (pattern[j] == text[i]):
            i += 1
            j += 1
            if (j == m):
                return (i-j)
                matchedIdx.append(i-j)
                total += 1
                j = lps[j-1]
            elif (j > 0):
                j = lps[j-1]
            else:
                i += 1

    # if (total > 0) :
    #     print("Total matched pattern in the text: ", str(total))
    #     print("Matched at index: ", end="")
    #     print(matchedIdx)
    if (total == 0):
        # print("Pattern not found")
        return (-1)
```

```
def calculateLPS(pattern, lps):
    m = len(pattern)

    i = 1
    j = 0 # length of the previous longest prefix suffix

    while (i < m):
        if (pattern[i] == pattern[j]):
            lps[i] = j + 1
            i += 1
            j += 1
        elif (j > 0):
            j = lps[j-1]
        else:
            lps[i] = 0
            i += 1
```

2. BM.py

```
def BMMatch(text, pattern):
    m = len(pattern)
    n = len(text)

    i = m - 1
    total = 0 # total matched pattern in the text
    matchedIdx = []

    lastOcc = buildLastOccurrence(pattern)

    if (i > n - 1) :
        return (-1)
    else :
        j = m - 1
        while (i <= n - 1):
            if (pattern[j] == text[i]):
                i -= 1
                j -= 1
                if (j == -1) :
                    return (i+1)
                total += 1
                matchedIdx.append(i+1)
                j = m - 1
                i += 2*m
            else :
                lo = lastOcc[ord(text[i])]
                i += m - min(j, 1+lo)
                j = m - 1

    if (total == 0):
        return (-1)
```

```
def buildLastOccurrence(pattern):
    # length of pattern
    m = len(pattern)
    # Init all occurrence as -1
    lastOcc = [-1 for i in range(128)]

    # Fill the value of last occurrence to the array
    for i in range(m):
        lastOcc[ord(pattern[i])] = i;

    return lastOcc
```

3. Extractor.py

```

# 13518016
# Indra Febrio Nugroho
# Regex and Extractor

import re
import copy
from nltk import sent_tokenize
from BM import BMMatch
from KMP import KMPMatch

def RegexMatch(text, pattern):
    if (re.search(pattern, text)):
        return 1
    else:
        return -1

def algChooser(alg, text, keyword):
    idx = -1
    if (alg == "kmp"):
        idx = KMPMatch(text.lower(), keyword.lower())
    elif (alg == 'bm'):
        idx = BMMatch(text.lower(), keyword.lower())
    else:
        idx = RegexMatch(text.lower(), keyword.lower())
    return idx

def processFile(path, files, keyword, alg):
    listofTotal = []
    listofTime = []
    listofWords = []
    listofFiles = []
    for f in files:
        if (f.endswith(".txt")):
            file = open(path + f, 'r')
            text = file.read()
            reTime = re.compile(r'([A-Za-z]{4,6})\,? \)\(\?d{1,2}[\\\/- ](\d{1,2})[a-zA-Z]{3,})[\\\/- ]?d{2,4})\)? (pukul )?d{1,2}[\\\/- ]d{1,2} ((WITA)|(WI
            timeG = reTime.search(text)
            globalTime = timeG.group(0)
            arrOfWords = sent_tokenize(text)

            for words in arrOfWords:
                idx = algChooser(alg, words, keyword)
                if (idx > 0):
                    reTotal = re.compile(r'(\d{1,3}(\.|\,))*d{1,3} ((([Oo]rang)|([Pp]asien)|([OoPp]dp)|([Kk]jasus))')
                    totalRes = reTotal.search(words)
                    reTime = re.compile(r'([A-Za-z]{4,6})\,? \)\(\?d{1,2}[\\\/- ](\d{1,2})[a-zA-Z]{3,})[\\\/- ]?d{2,4})\)? (pukul )?d{1,2}[\\\/- ]d{1,2} ((W
                    timeRes = reTime.search(words)

                    if (totalRes and timeRes):
                        w = totalRes.group().split(' ')
                        listofTotal.append(copy.deepcopy(w[0]))
                        listofTime.append(copy.deepcopy(timeRes.group()))
                        listofWords.append(copy.deepcopy(words))
                        listofFiles.append(copy.deepcopy(f))

                    elif (totalRes):
                        w = totalRes.group().split(' ')
                        listofTotal.append(copy.deepcopy(w[0]))
                        listofTime.append(copy.deepcopy(globalTime))
                        listofWords.append(copy.deepcopy(words))
                        listofFiles.append(copy.deepcopy(f))

    return listofTotal, listofTime, listofWords, listofFiles

```

4. App.py

```
from flask import Flask, redirect, url_for, render_template, request
from werkzeug.utils import secure_filename
from Extractor import processFile

app = Flask(__name__)

@app.route("/", methods=['GET', 'POST'])
def homePage():
    if (request.method == 'POST'):
        path = "../test/"
        fileFromWeb = request.files.getlist('file[]')
        files = []
        for f in fileFromWeb:
            files.append(f.filename)
        keyword = request.form['kw']
        algorithm = request.form['algo']
        listofTotal, listofTime, listofWords = processFile(path, files, keyword, algorithm)
        length = len(listofTotal)
        return render_template('app.html', Length=length, keyword=keyword, total=listofTotal, time=listofTime, words=listofWords, files=listofFiles)
    else:
        keyword = ""
        listofTotal = []
        listofTime = []
        listofWords = []
        listofFiles = []
        length = len(listofTotal)
        return render_template('app.html', Length=length, keyword=keyword, total=listofTotal, time=listofTime, words=listofWords, files=listofFiles)

if __name__ == "__main__":
    app.run(debug=True)
```

5. app.html

```
{% extends "layout.html" %}
{% block body %}
<!DOCTYPE html>
<h1>My InfoExtraction App</h1>

<form name="allForm" class="form-inline" method="POST" action="/" onsubmit="/" enctype="multipart/form-data">
  <div class="form-group">
    <div class="input-group">
      <p>Files:</p>
      <p><input type="file" name="file[]" multiple=""></p>
      <p>Keyword:</p>
      <p><input type="text" name="kw" /></p>
      <span class="input-group-addon">Algorithm:</span>
      <select name="algo" class="selectpicker form-control">
        <option value="kmp">Knuth-Morris-Pratt</option>
        <option value="bm">Boyer Moore</option>
        <option value="regex">Regex</option>
      </select>
    </div>
    <button type="submit" class="btn btn-default">Show</button>
  </div>
</form>

<p>Search Result</p>

<p>Keyword: {{keyword}}</p>
<p>Result of Extracted Information:</p>

<ol>
{%for i in range(0, length)%}
  <p>Total: {{total[i]}}; Time: {{time[i]}}</p>
  <p>{{words[i]}} ({{files[i]}})</p>
  <br>
{%endfor%}

</ol>

{% endblock %}
```

BAB III

SCREENSHOT I/O PROGRAM

3.1 Input Output Program Case 1 (“positif”, KMP)

My InfoExtraction App

Files:

No file selected

Keyword:

Algorithm:

Search Result

Keyword: positif

Result of Extracted Information:

Total: 421; Time: Sabtu, 11 Apr 2020 20:07 WIB

421 Orang di Jabar Terkonfirmasi Positif COVID-19. (1.txt)

Total: 400; Time: Sabtu, 11 Apr 2020 20:07 WIB

Bandung - Angka positif virus Corona atau COVID-19 di Jawa Barat menembus angka 400 kasus. (1.txt)

Total: 421; Time: Sabtu (11/4/2020) pukul 18.43 WIB

Laman Pusat Informasi dan Koordinasi COVID-19 Jabar (Pikobar) pada Sabtu (11/4/2020) pukul 18.43 WIB, mencatat terdapat 421 orang yang terkonfirmasi positif COVID-19. (1.txt)

Total: 3.842; Time: Sabtu, 11 Apr 2020 20:07 WIB

Sementara itu, secara nasional terdapat 3.842 kasus positif COVID-19. (1.txt)

Total: 375; Time: Selasa (21/4/2020) pukul 12.00 WIB

"Pada hari ini kami dapatkan 375 kasus konfirmasi (positif) yang baru, sehingga total ada 7.135 orang," ujar Achmad Yurianto. (2.txt)

Total: 6.760; Time: Senin, 20 April 2020

Informasi yang diumumkan Gugus Tugas Percepatan Penanganan Covid-19 pada Senin, 20 April 2020, menunjukkan total kasus positif corona di Indonesia telah mencapai 6.760 pasien. (3.txt)

Total: 185; Time: Senin, 20 April 2020

Angka tersebut dihitung setelah ada tambahan kasus positif baru yang terkonfirmasi dalam 24 jam terakhir sebanyak 185 orang. (3.txt)

Total: 5.423; Time: Senin, 20 April 2020

Hingga hari ini, sebanyak 5.423 pasien positif corona masih menjalani perawatan. (3.txt)

Total: 5.923; Time: Kamis (16/4/2020) pukul 12.00 WIB

"Kasus terkonfirmasi positif pada hari ini sebanyak 407, sehingga jumlahnya menjadi 5.923 orang," ujar Yurianto. (6.txt)

Total: 5.516; Time: pada 2 Maret 2020

Kasus baru itu menyebabkan kasus pasien positif Covid-19 mencapai 5.516 kasus, sejak kasus ini muncul pada 2 Maret 2020. (7.txt)

Total: 5.516; Time: Kamis (16/4/2020) pukul 12.00 WIB

"Konfirmasi positif coronavirus atau Covid-19 dari pemeriksaan PCR menjadi 5.516 orang," ujar Achmad Yurianto. (7.txt)

Total: 297; Time: Selasa (14/4/2020) pukul 12.00 WIB

"Secara keseluruhan kasus positif yang kita dapat hari ini adalah 297 pasien sehingga total terkonfirmasi positif akumulatif menjadi 5.136 orang," ujar Achmad Yurianto. (8.txt)

Total: 469; Time: Selasa (14/4/2020) pukul 12.00 WIB

Ada 469 pasien yang tutup usia setelah dinyatakan positif virus corona hingga kemarin siang. (8.txt)

Total: 4.557; Time: Senin (13/4/2020)

Berdasarkan data kasus yang diumumkan pemerintah hingga Senin (13/4/2020) sore, pemerintah menyatakan, ada 4.557 kasus positif Covid-19 di Tanah Air, dengan penambahan 316 pasien dalam 24 jam terakhir. (10.txt)

Total: 316; Time: Senin (13/4/2020)

"Kasus konfirmasi positif sebanyak 316 orang, sehingga total 4.557 orang," ujar Juru Bicara Pemerintah untuk Penanganan Virus Corona Achmad Yurianto, Senin (13/4/2020). (10.txt)

Gambar 1 i/o case 1

3.2 Input Output Program Case 2 (“meninggal”, BM)

My InfoExtraction App

Files:

No file selected

Keyword:

Algorithm:

Search Result

Keyword: meninggal

Result of Extracted Information:

Total: 421; Time: Sabtu, 11 Apr 2020 20:07 WIB

Dari 421 kasus tersebut, 40 orang meninggal dunia dengan keterangan terpapar COVID-19. (1.txt)

Total: 26; Time: Selasa (21/4/2020) pukul 12.00 WIB

Data yang sama juga menyebutkan bahwa tercatat ada penambahan 26 pasien Covid-19 yang meninggal dalam 24 jam terakhir. (2.txt)

Total: 200; Time: Minggu (19/4/2020) pukul 09.35 WIB

Lebih dari 200 orang kini telah meninggal karena COVID-19 dan ibu kota Tokyo tetap menjadi daerah yang paling parah terkena dampaknya. (4.txt)

Total: 15; Time: Sabtu (18/4/2020) pukul 12.00 WIB

Ada 15 pasien Covid-19 yang meninggal dunia dalam kurun waktu Jumat hingga Sabtu siang. (5.txt)

Total: 15; Time: Sabtu (18/4/2020) pukul 12.00 WIB

"Ada (penambahan) 15 pasien meninggal sehingga totalnya menjadi 535," kata Yuri. (5.txt)

Total: 15; Time: Sabtu (18/4/2020) pukul 12.00 WIB

Adapun, penambahan 15 pasien meninggal akibat Covid-19 berasal dari enam provinsi. (5.txt)

Total: 1; Time: Sabtu (18/4/2020) pukul 12.00 WIB

Adapun, Bali, Sumatera Barat, dan Sulawesi Tenggara masing-masing mencatat 1 pasien meninggal. (5.txt)

Total: 520; Time: Kamis (16/4/2020) pukul 12.00 WIB

Total pasien Covid-19 yang meninggal dunia saat ini ada 520 kasus. (6.txt)

Total: 10; Time: Selasa (14/4/2020) pukul 12.00 WIB

Namun, Yuri mengungkapkan kabar duka dengan adanya 10 pasien Covid-19 yang meninggal dunia dalam 24 jam terakhir. (8.txt)

Total: 26; Time: pada 2 Maret 2020

Sementara itu, ada penambahan 26 pasien yang meninggal dunia setelah terinfeksi Covid-19, sehingga total pasien meninggal menjadi 399 orang. (10.txt)

Gambar 2 i/o case 2

3.3 Input Output Program Case 3 (“pdp”, Regex)

My InfoExtraction App

Files:

No file selected

Keyword:

Algorithm:

Search Result

Keyword: pdp

Result of Extracted Information:

Total: 2.278; Time: Sabtu, 11 Apr 2020 20:07 WIB

Sementara itu jumlah Pasien Dalam Pengawasan (PDP) mencapai 2.278 orang. (1.txt)

Total: 16.343; Time: Senin, 20 April 2020

Menurut juru bicara pemerintah untuk penanganan COVID-19, Achmad Yurianto, sampai hari ini, sudah ada 16.343 orang yang berstatus sebagai Pasien Dalam Pengawasan (PDP). (3.txt)

Gambar 3 i/o case 3

3.4 Kesimpulan

Luaran dari tugas ini dapat dimuat secara ringkas dalam tabel berikut

Tabel 1 Output Persoalan

Poin	Ya	Tidak
1. Program berhasil dikompilasi	v	
2. Program berhasil <i>running</i>	v	
3. Program dapat menerima input dan menuliskan output	v	
4. Luaran sudah benar untuk semua data uji	v	

REFERENSI

- Munir, Rinaldi. 2004. *Diktat Bahan Kuliah Strategi Algoritmik*. Bandung: Departemen Teknik Informatika, Institut Teknologi Bandung.
<https://unfinishedcreativework.wordpress.com/2017/06/10/algoritma-knuth-morris-pratt/> diakses pada 22 April 2020, 02:33 WIB
- Lecroq, Thierry Charras, Christian. 2001. Handbook of Exact String Matching Algorithm.
- Boyer, Robert Moore, J. 1977. A Fast String Searching Algorithm. Comm. ACM 20: 762–772