Penerapan Algoritma Pencocokan *String* untuk Validasi Topik Makalah Strategi Algoritma yang Unik

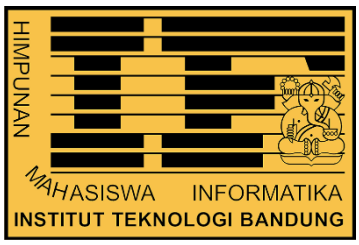
Aldo Azali (13516125)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13516125@std.stei.itb.ac.id

***Abstrak*— Pencocokan *string* (*String Matching*) merupakan teknik pencarian suatu teks atau kalimat di dalam teks atau kalimat lainnya. Algoritma pencocokan *string* ini telah banyak diterapkan pada berbagai macam aplikasi. Penggunaan algoritma *string matching* pada aplikasi - aplikasi telah banyak mempengaruhi kehidupan manusia, baik dari segi akademik, pekerjaan, kesehatan, kuliner, cinta, dan lain-lain. Salah satu contoh penerapan algoritma ini yaitu pada Mesin Pencarian *Web* (*Web Search Engine*) yaitu Google. Selain Google, Algoritma ini juga diterapkan pada aplikasi sehari-hari yang sering kita gunakan seperti media sosial, telepon, SMS, teks editor, *game*.**

Pada beberapa aplikasi yang membutuhkan registrasi dengan keamanan yang baik, validasi data isian identitas diri dan nama akun sangat diperlukan. Validasi data adalah proses untuk memeriksa dan memastikan kebenaran data yang dimasukkan ke dalam sistem. Pada mata kuliah IF2211, Strategi Algoritma, tugas makalah diberikan setiap tahun untuk mahasiswa teknik informatika ITB yang jumlahnya cukup banyak. Agar dapat membuat suatu topik makalah yang unik, maka diperlukan adanya validasi. Salah satu teknik validasi dapat dilakukan dengan algoritma pencocokan *string*.

***Keywords* — Aplikasi, Strategi Algoritma, Pattern Matching, Validasi.**

I. PENDAHULUAN



Mata Kuliah Strategi Algoritma merupakan salah satu mata kuliah wajib Teknik Informatika ITB di Semester 4 dengan kode IF2211. Mata Kuliah ini mempelajari teknik-teknik pendekatan umum untuk memecahkan persoalan secara algoritmis yang dapat diterapkan pada bermacam-macam persoalan dari berbagai bidang komputasi. Selain itu, Strategi Algoritma dapat memberikan panduan untuk merancang algoritma persoalan yang baru dan dapat mengklarifikasi algoritma berdasarkan perancangan yang mendasarinya. Persoalan – persoalan yang diberikan ini dapat dipecahkan dan disolusikan dalam bentuk matematis dan dikomputasikan. Pemecahan masalah ini dapat menggunakan algoritma-algoritma yang dipelajari di mata kuliah IF2211 ini, yaitu Algoritma *Brute-Force*, Algoritma *Greedy*, Algoritma *Divide and Conquer*, Algoritma *Decrease and Conquer*, Algoritma *Backtracking*, Algoritma *Branch and Bound*, Algoritma Pencocokan *String* dan *Dynamic Programming*.

Pada akhir semester 4 setiap tahunnya, Dosen - dosen mata kuliah Strategi Algoritma selalu memberikan tugas pembuatan makalah kepada mahasiswa – mahasiswa mereka. Makalah yang dibuat haruslah makalah ilmiah yang berupa *technical report* yang berkaitan dengan strategi algoritmik yang telah dipelajari selama satu semester. Adapun tujuan penulisan makalah yaitu :

1. Memotivasi mahasiswa agar memiliki kemampuan menulis untuk menuangkan ide-ide atau hasil risetnya.
2. Melakukan eksplorasi terhadap isu, metode, dan masalah yang dipelajari dalam pengembangan serta menyebarkan aplikasi yang mendukung teknologi informasi.
3. Sebagai media untuk berbagi informasi hasil - hasil pemikiran dan penelitian. Semua makalah mahasiswa akan dimuat di dalam website dosen, sehingga siapa pun dapat melihat dan membaca karya ilmiah mahasiswa Teknik Informatika Tahun kedua.

Makalah Mahasiswa Tahun 2011

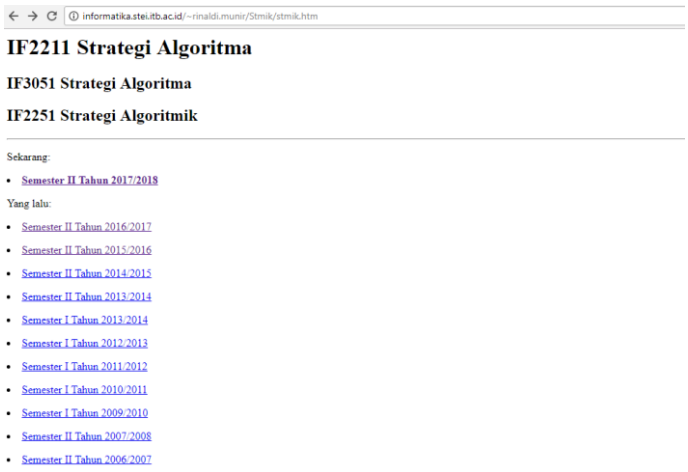
1. Penerapan Algoritma *Greedy* untuk Reservasi Tiket Konser
   Oleh: Aminah Nuraini (13509055)
2. Perbandingan Penggunaan Algoritma *Greedy* dan Modifikasi Algoritma *Brute Force* pada Permainan *Collapse XXL*
   Oleh: Rahadian Dimas Prayudha (13509009)
3. *English Word Segmentation Problem Comparison of Algorithms*
   Oleh: Irvan Jahj(13509099)
4. Pengaplikasian Algoritma *Knuth-Morris-Pratt* dalam Teknik Kompresi Data
   Oleh: I Nyoman Prama Pradnyana (13509032)
5. Penerapan Algoritma *Greedy* dan Algoritma BFS untuk AI pada Permainan *Greedy Spiders*
   Oleh: Rachmawaty (13509071)
6. Pengaturan Sistem Lampu Lalu Lintas dengan Algoritma *Branch and Bound* dengan Waktu Tunggu Menggunakan Algoritma *Greedy*
   Oleh: Aldo Suwandi (13509025)
7. Penerapan Algoritma *Divide and Conquer* untuk Meningkatkan Kecepatan Deteksi dari Mesin *Client Honeypots*
   Oleh: Nugraha (13509096)
8. *Multithreading* untuk Algoritma *Divide and Conquer*
   Oleh: Novan Pamonangan Simanjuntak (13509034)
9. Penerapan Algoritma *A-Star (A*)* untuk Menyelesaikan Masalah *Maze*
   Oleh: Hapsari Tilawah (13509027)
10. *Greedy vs Greedy in Dragon Age : Origins*
    Oleh: Rido Ramadan (13509049)
11. Penerapan Algoritma *Depth-First Search* dan *Backtracking* dalam Program Pembentuk *Maze*
    Oleh: Prisyafandiafif Charifa (13509081)
12. *Dynamic Programming* dalam *Board Game Power Grid*

Gambar 0 : List dari Makalah mahasiswa Teknik Informatika ITB 2011

Selain itu, pembuatan makalah ini memiliki berbagai macam syarat, salah satunya yaitu Makalah haruslah unik, tidak boleh sama dengan makalah yang telah dibuat pada tahun – tahun sebelumnya dan makalah tidak boleh hasil plagiasi dari makalah orang lain. Ketika melihat bahwa pemberian tugas makalah pada mata kuliah ini telah berlangsung lama, maka untuk memenuhi syarat tersebut menjadi sulit dan bermasalah. Ditambah dengan jumlah mahasiswa Informatika ITB yang kian bertambah tiap

tahunnya, maka membuat suatu topik makalah yang unik menjadi suatu kesulitan dan masalah tersendiri bagi mahasiswa Informatika ITB. Untuk memecahkan masalah ini, dibutuhkannya adanya aplikasi yang dapat memvalidasi penentuan topik makalah yang akan dibuat agar memenuhi syarat keunikan tersebut. Selain itu, aplikasi ini dapat mempermudah dosen dalam melakukan pengecekan dan memberi penilaian makalah mahasiswanya.

Aplikasi validasi ini dapat dibuat dengan menggunakan algoritma pattern matching. Cara kerja aplikasi ini dapat mengikuti kemiripan algoritma pencarian pada *Web Search Engine* seperti Google. Akan tetapi aplikasi ini dapat mencari dan membedakan topik algoritma yang mirip tetapi berbeda seperti *Topik Penerapan Greedy pada AI Catur* dan *Topik Penerapan Greedy pada AI Catur Kuantum*. Selain itu, data – data sejumlah topik makalah strategi algoritma yang diperlukan aplikasi dapat diambil dengan melakukan *data scraping* terhadap website strategi algoritma yang telah dibuat untuk mendukung proses pembelajaran mata kuliah oleh dosen Informatika ITB, Bapak Rinaldi Munir.



Gambar 1 : Website IF2211 Strategi Algoritma.

II. DASAR TEORI

Pencocokan *String* atau *String Matching* adalah proses pencarian semua kemunculan *query* yang selanjutnya disebut pattern ke dalam *string* yang lebih panjang atau teks. Pencocokan *String* dapat dirumuskan sebagai berikut :

P = P[0…m-1]
T = T[0…n-1]

Dimana :

P adalah *pattern*
m adalah panjang *pattern*
T adalah teks
n adalah panjang teks

Pencocokan *String* atau *String Matching* dapat dibagi menjadi 2 jenis, yaitu Pencocokan Eksak (*Exact Matching*) dan Pencocokan Regex (*Regex Matching*).
Pencocokaan Eksak memiliki beberapa jenis teknik pencarian, yaitu :

A. Algoritma Brute Force

Algoritma Brute Force adalah pendekatan yang lempang (*straightforward*) untuk memecahkan suatu masalah. Pada kasus Pencocokan *String*, Brute Force melakukan pencarian string dengan melakukan perbandingan pada setiap karakter yang dijumpai hingga menemukan bagian dari teks yang sesuai dengan *pattern*. Teknik pencarian algoritma *Brute Force* dilakukan dengan keterunutan pencarian *pattern* pada teks dari kiri ke kanan. Pada algoritma *Brute Force*, kompleksitas kasus terbaik adalah O(n). Kasus terbaik terjadi ketika karakter pertama *pattern* P tidak pernah sama dengan karakter teks T yang dicocokkan. Jumlah perbandingan yang dilakukan saat kasus terbaik yaitu sebanyak n kali. Sedangkan kasus terburuk terjadi bila semua *prefix* karakter teks T selalu sama dengan prefix karakter *pattern* P. Sehingga jumlah perbandingan yang dilakukan sebanyak m*(n-m+1) dengan kompleksitas O(mn).

B. Algoritma KMP (Knuth-Morris-Pratt)

Algoritma KMP merupakan hasil analisis algoritma yang dikemukakan oleh Donald Knuth, Vaughan Pratt dan James H. Mooris untuk mensolusikan masalah pencocokan string. Algoritma ini mencari pattern di dalam teks secara sekuensial dari kiri ke kanan karakter teks seperti algoritma Brute Force. Tetapi algoritma ini dapat berjalan lebih cepat dan lebih baik daripada algoritma Brute Force. Pada dasarnya, algoritma KMP mirip dengan algoritma Brute Force, tetapi algoritma ini memiliki fungsi tambahan yaitu Fungsi Pinggiran KMP (KMP Border Function) yang berguna untuk mengoptimalkan pencarian. Nilai fungsi pinggiran b(m) KMP didefinisikan sebagai ukuran awalan(prefix) terbesar P[0..m-2] yang juga merupakan suffix dari P[1..m-1]. Dalam koding, b() dapat direpresentasikan sebagai array, seperti tabel data. Contohnya sebagai berikut :

j	0	1	2	3	4	5
P[j]	a	b	a	a	b	a
K	-	0	1	2	3	4
b(k)	-	0	0	1	1	2

b(k) adalah ukuran dengan fungsi pinggiran terbesar.

Kompleksitas waktu algoritma KMP sangat cepat dibandingkan dengan KMP, yaitu O(m+n). Kompleksitas ini didapat dengan menghitung kompleksitas waktu fungsi pinggiran : O(m) ditambah dengan kompleksitas waktu pencarian string : O(n) sehingga O(m) + O(n) = O(m+n). Keuntungan algoritma ini sangat baik untuk memproses file berukuran besar yang dibaca dari perangkat eksternal maupun melalui aliran jaringan (*Network Stream*) karena algoritma ini tidak perlu kembali bergerak mundur dalam masukan teks. Tetapi KMP semakin tidak optimal jika jumlah variasi alfabet semakin banyak. Hal ini dapat meningkatkan jumlah ketidakcocokan apalagi jika ketidakcocokan cenderung terjadi di awal *pattern*.

C. Algoritma Boyer-Moore

Algoritma boyer moore adalah algoritma pencocokan *string* yang dikemukakan oleh Robert S.Boyer dan J. Strother Moore dan menjadi algoritma yang paling efisien dan sudah menjadi standar dari sistem pencarian string. Algoritma ini didasarkan pada 2 teknik pencarian, yaitu teknik looking-glass dan teknik character-jump. Teknik looking-glass melakukan perbandingan karakter pattern dengan teks dimulai dari karakter teks yang paling kanan. Teknik character-jump melakukan lompatan perbandingan karakter ketika sewaktu menemukan ketidakcocokan berdasarkan fungsi *Last Occurrence*. Fungsi Last Occurrence ini mengambil karakter c dari alfabet dan menentukan seberapa jauh pergeseran pola P jika karakter yang sama dengan c ditemukan dalam teks yang tidak sesuai dengan *pattern*. Jika tidak ditemukan maka diberikan nilai -1.

Contohnya sebagai berikut :

A : {a,b,c,d} P : “abacab”

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

Sama seperti halnya pada fungsi pinggiran KMP, fungsi Last Occurrence, L() juga direpresentasikan seperti array dan tabel. Kompleksitas waktu algoritma ini dengan kasus terburuk adalah O(nm+A). Kasus terburuk terjadi ketika teks yang ingin dicari memiliki jumlah variasi alfabet yang sedikit dan pattern memiliki pola alfabet yang menyerupai teks pada akhiran(suffix) pattern. Contohnya :

T : aaaaaaaaa

P : baaaaa

Algoritma Boyer-Moore sangat baik digunakan ketika jumlah variasi alfabet besar, khususnya pada teks berbahasa inggris dan indonesia. Namun algoritma ini tidak cocok untuk pencarian teks binary.

Selain Pencocokan eksak, terdapat juga pencocokan regex. Pencocokan regex adalah pencarian pattern di dalam teks yang bersesuaian dengan pattern dengan bantuan notasi regex. Penggunaan notasi regex bertujuan mencari suatu pattern yang memiliki kemiripan dengan teks dengan tidak eksak atau memiliki perbedaan yang minor seperti alfabet besar dan alfabet kecil.

Contoh yang dapat dicari oleh pencocokan regex yaitu mencari pattern “Lulusan Informatika ITB” di dalam teks “lulusan Teknik Informatika ITB”.

Notasi regex pattern dapat dituliskan sebagai berikut : [Ll]ulusan .*[Ii]nformatika [Ii][Tt][Bb]

Basic Regular Expression Patterns

- Uses of the caret ^ for negation or just to mean ^

RE	Match (single characters)	Example Patterns Matched
[^A-Z]	not an uppercase letter	“Oyfn pripetchik”
[^Ss]	neither ‘S’ nor ‘s’	“I have no exquisite reason for’t”
[^\.]	not a period	“our resident Djinn”
[e^]	either ‘e’ or ‘^’	“look up ^ now”
a^b	the pattern ‘a^b’	“look up a^b now”

- The question-mark ? marks optionality of the previous expression.

RE	Match	Example Patterns Matched
woodchucks?	woodchuck or woodchucks	“woodchuck”
colou?r	color or colour	“colour”

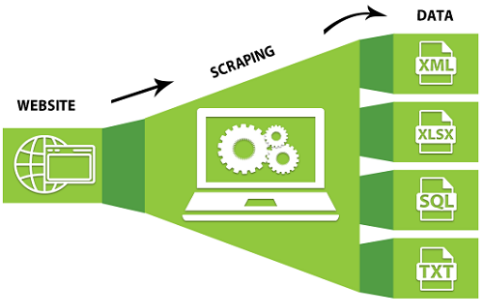
- The use of period . to specify any character

RE	Match	Example Patterns
/beg.n/	any character between *beg* and *n*	begin, beg’n, begun

Gambar 2 : Bentuk-bentuk Notasi Regex.

III. *DATA SCRAPING*

A. Definisi

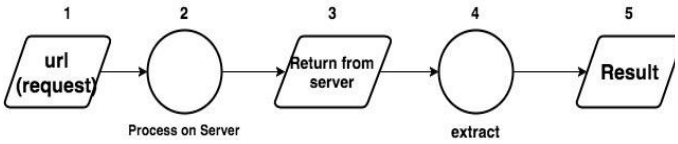


Data scraping adalah kegiatan untuk mendapatkan informasi yang diinginkan dari internet. Tujuan dari *data scraping* adalah mencari jenis informasi tertentu, mengekstrak, dan menggabungkan ke penyimpanan data untuk digunakan. *Data scraping* sendiri terdiri dari beberapa varian yang salah satunya adalah *web scraping*, dimana kegiatan data scraping dilakukan di laman web. Web Scraping adalah proses pengumpulan data atau informasi dari suatu laman web secara otomatis, bukan secara manual menyalinnya. Pada umumnya *Web Scraping* dilakukan dengan mengambil data berbentuk teks yang bertipe HTML atau XHTML.

Pada aplikasi ini, tipe *data scraping* yang akan dilakukan yaitu melakukan *Web Scraping* terhadap laman web Strategi Algoritma yang telah dibuat dosen untuk mendapatkan semua data judul topik makalah. Pada *Web Scraping*, ada 2 teknik cara untuk melakukan *scrapping*, yaitu teknik *search string between two strings dan search json on render page*.

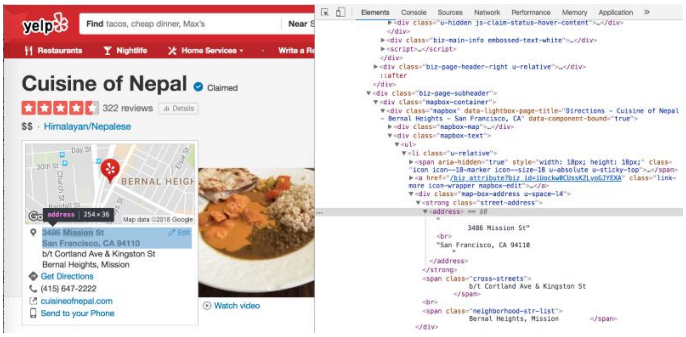
Umumnya kedua teknik diimplementasikan dengan 5 tahap, yaitu :

1. Melakukan *request* url laman yang dijadikan target.
2. *Request* diproses oleh server target.
3. Hasil dari *request url* diterima (biasanya dalam bentuk teks dengan format HTML).
4. Mengekstrak data (Mengambil teks yang diperlukan dari tahap ke -3).
5. Hasil ekstrak (menentukan *output* yang diinginkan).



Gambar 3 : Tahap web Scraping
Sumber : https://cdn-images-1.medium.com/max/1600/0*Ggfgwu2I-3ffG_-t.

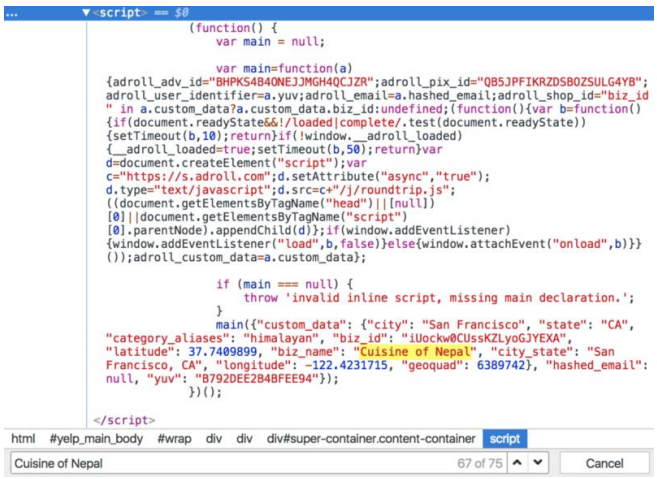
Web Scraping sendiri telah disediakan tools(alat bantu) oleh beberapa website di internet. Contoh – contoh tool yang disediakan internet, yaitu Curl, Data Toolbar, Diffbot, Heritrix, HtmlUnit, HTTrack, iMacros, Selenium, Node.js, jQuery, PhantomJS, dll. Tools ini sangat berguna untuk membantu dalam kodingan untuk data scraping.



Gambar 4 : Contoh Web Scraping.
Sumber : https://cdn-images-1.medium.com/max/800/1*LszoJ8sZqNqtrRPjwaCfNw.png

B. Penerapan pada aplikasi

Pada aplikasi untuk memvalidasi topik makalah Strategi Algoritma yang unik, Web Scraping ditargetkan ke website resmi mata kuliah Strategi Algoritma yang telah dibuat dosen Informatika ITB. Setelah melakukan beberapa kodingan dan mengatur format, maka bisa didapatkan sejumlah topik makalah dari tahun ke tahun dan dapat disimpan ke dalam database aplikasi atau file eksternal untuk dapat diproses aplikasi.



Gambar 5 : Contoh kodingan untuk mendapatkan data dari Web Scraping.

Sumber : https://cdn-images-1.medium.com/max/800/1*9MAtfZAsI2Rko-_5IavpmA.png

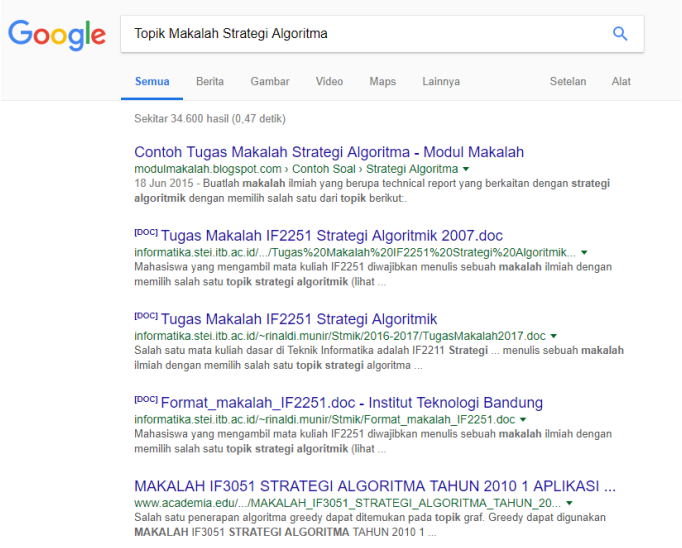

```
{
  "city": "San Francisco",
  "state": "CA",
  "category_aliases": "himalayan",
  "biz_id": "iUockw0CUssKZLyoGJYEXA",
  "latitude": 37.7409899,
  "biz_name": "Cuisine of Nepal",
  "city_state": "San Francisco, CA",
  "longitude": -122.4231715,
  "geoquad": 6389742
}
```


Gambar 6 : Contoh hasil data yang telah berhasil diekstrak dari kodingan gambar 5.

IV. PENERAPAN ALGORITMA PADA APLIKASI

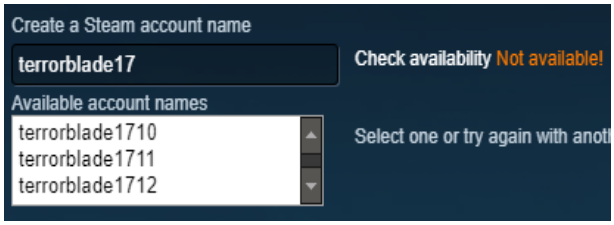
A. Gambaran Umum

Aplikasi yang diperlukan untuk dibuat harus berbasis internet, dimana membutuhkan jaringan internet. Internet ini bertujuan agar dapat melakukan data scraping dari laman website yang akan dituju. URL websitenya yaitu informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/stmik.htm. Website tersebut merupakan werbsite perkuliahan Strategi Algoritma Teknik Informatika ITB. Mahasiswa dapat tinggal membuka aplikasi dan memberi input masukan "pattern" mereka sesuai topik makalah yang ingin mereka buat. Seperti pada Web Search Engine, Google, aplikasi dapat menampilkan hasil pencarian ketemu bila input masukan terdapat pada list dan dapat memberikan keterangan "unik" seperti pada verifikasi pembuatan akun ketika input masukan topik makalah tidak terdapat pada list topik makalah.

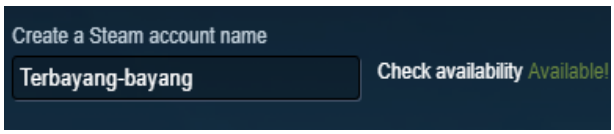


Gambar 7: Contoh Web Search Engine, Google.

Selain itu, Aplikasi dapat dikembangkan untuk mendeteksi pola masukan dari input apakah sesuai dengan topik makalah atau tidak. Sehingga hal ini dapat mengurangi kesalahan penamaan topik pada makalah mahasiswa misalnya salahnya topik makalah yang mereka inginkan dengan syarat topik yang diberikan oleh dosen.
Aplikasi ini dapat dibuat dengan menggunakan bahasa Python, C#, maupun Java dan dapat memanfaatkan tools JSON untuk menggunakan data hasil dari Web Scrapping.



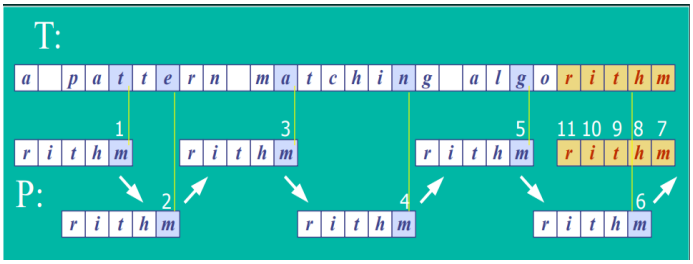
Gambar 8: Contoh pembuatan nama akun tidak unik



Gambar 9: Contoh pembuatan nama akun yang unik.

B. Penggunaan Algoritma

Untuk mendapatkan hasil pencarian yang optimal dan cepat, maka algoritma yang sangat sesuai dipakai adalah algoritma Boyer-Moore, dimana algoritma ini semakin baik digunakan pada jumlah variasi alfabet semakin banyak sehingga cocok digunakan untuk menemukan pattern pada bahasa manusia seperti, bahasa inggris ataupun bahasa indonesia. Agar pencocokan tidak exact matching, maka algoritma Boyer-Moore dapat ditambahkan dengan Regex agar dapat melakukan pencarian dengan hasil yang lebih baik dan mirip dengan input masukan (pattern). Algoritma Brute Force tidak akan digunakan karena algoritma ini tidak menghasilkan kecepatan pencarian yang optimal sedangkan hasil pencariannya sama dengan hasil pencarian dengan algoritma Boyer-Moore. Demikian juga dengan algoritma KMP tidak cocok digunakan pada aplikasi ini karena algoritma KMP lebih cocok dan baik bila jumlah variasi alfabet pada teks dan pattern sedikit sedangkan input masukan berupa bahasa manusia yang memiliki jumlah variasi alfabet yang cukup banyak.



Gambar 10: Pencocokan String secara Boyer-Moore

C. Pseudocode Algoritma Boyer-Moore

Algoritma BuildLast:

```
function buildLast (pattern : array[1..m] of char) → array of integer
{ mengembalikan array sorting index dari last occurrence dari setiap nilai ASCII dari karakter pada pattern}

Kamus Lokal
    i, j : integer
    last : array[0..127] of integer {himpunan karakter ASCII}

Algoritma Lokal
    for i ← 0 to 127 do
            last[i] ← -1
    endfor
    for j ← 0 to m-1 do
            last[pattern[j]] ← j
    endfor
    → last
```

Algoritma Boyer-Moore :

```
function bmMatch (text: array[1..a] of char, pattern: array [1..b]of char ) → integer
{mengembaikan indeks pertama ditemukannya pattern pada teks, mengembalikan -1 jika tidak ketemu}

Kamus Lokal
    i, j, n, m, lo, idx : integer
    last : array of integer

Algoritma Lokal
    last ← buildLast(pattern)
    n ← a
    m ← b
    if  i > n-1 then {tidak cocok jika pattern lebih panjang dari teks}
        idx ← -1
     else
        j ← m-1
        while i ≤ n-1
            if pattern[j] = text[i] then
                if j=0 then
                    idx ← i {cocok}
                else {teknik looking glass}
                    i ← i-1
                    j ← j-1
                endif
            else {teknik character jump}
                lo ← last[text[i]] {last occurence}
                i ← i +m-min(j, 1+lo)
                j ← m-1
                idx ← -1
            endif
        endwhile
    endif
    → idx {mengembalikan indeks pertama ditemukannya pattern pada teks, -1 jika tidak ketemu}
```

D. Proses Pencocokan String

Pada proses pencocokan string, pertama kali input masukan yang menjadi pola(pattern) akan diidentifikasi dan dicocokkan dengan format nama topik makalah yang dibuat. Adapun pencocokan masukan dapat dicocokkan dengan kategori-kategori makalah yang diberikan, yaitu :

1. Brute Force
2. Greedy
3. Divide and Conquer
4. DFS atau BFS
5. Runut Balik atau Backtracking
6. Branch and Bound
7. Program Dinamis atau Dynamic Programming
8. Pencocokan String atau Pattern Matching
9. P, NP, atau NP Complete

Selain input masukan, data yang dimiliki juga telah di cari dan dikategorikan berdasarkan kategori-kategori di atas. Pencocokan pola dan data topik makalah dengan kategori di atas menggunakan algoritma regex agar dapat mengenali pola yang mirip tetapi tidak eksak.

Jika input masukan tidak sesuai kategori di atas, maka aplikasi dapat memberitahu bahwa topik makalah yang diajukan tidak terdapat pada kategori di atas.Setelah mendapatkan jenis kategori masukan, maka input masukan akan dicocokkan dengan data topik makalah sesuai kategorinya. Dengan demikian, hasil pencarian pun semakin cepat didapatkan karena mengurangi sebagian besar topik yang tidak mirip. Jadi misalkan pada data topik makalah terdapat judul makalah X yaitu "Penerapan algoritma greedy pada Permainan Catur" sedangkan input masukan topik makalah yaitu "Aplikasi algoritma DFS pada Permainan Catur". Dengan mengkategorikan jenis algoritma, maka kita akan mendapatkan bahwa topik makalah X tidak akan dicocokkan lagi dengan input masukan sehingga pencarian dapat dilakukan menjadi lebih cepat.

Setelah melakukan pengkategorian, maka input masukan akan dicocokkan dengan semua judul makalah yang berada di kategori tersebut. Dengan bantuan algoritma regex, pemisahan kata-kata tambahan, gabungan dan imbuhan, seperti "di", "dalam", "dan", "atau", dan lain lain, dapat dipisahkan untuk mendapatkan inti dari topik tersebut. Dengan begitu, hasil pencarian dan pencocokan dapat menjadi lebih optimal lagi.

V. KESIMPULAN

Penggunaan algoritma pencocokan string banyak ditemukan pada berbagai jenis aplikasi, baik dari web search engine, media sosial, game, galeri, hingga *Operating System* . Oleh karena itu, penggunaan algoritma ini sangat luas dan dapat dimanfaatkan dalam mendukung kegiatan akademik. Salah satunya yaitu seperti pembuatan aplikasi untuk validasi topik makalah Strategi Algoritma yang unik. Kolaborasi dengan penggunaan data scraping dapat membuat suatu aplikasi yang bertujuan baik untuk membantu mahasiswa dan dosen dalam pembuatan dan penilaian makalah. Selain itu, aplikasi ini dapat dikembangkan dan selalu *up-to-date* karena saat setiap aplikasi dijalankan, maka aplikasi akan melakukan data scraping ke website strategi algoritma yang memuat data terkini.

VI. UCAPAN TERIMA KASIH

Puji Syukur kepada Tuhan Yang Maha Esa sehingga makalah ini dapat diselesaikan dengan sebaik mungkin. Terima kasih sebesar-besarnya kepada semua pihak yang telah membantu dan mendukung pembuatan makalah ini. Selain itu, ucapan terima kasih juga diberikan kepada Ibu Nur Ulfa Maulidevi, Bapak Rinaldi Munir dan Ibu Masayu Leylia Khodra selaku dosen mata kuliah Strategi Algoritma (IF2211) Tahun 2017/2018 dan telah memberikan tugas dan materi-materi Srategi Algoritma serta penerapannya dalam aplikasi. Akhir kata, semoga makalah ini dapat berguna dan memberikan ilmu dan wawasan baru bagi semua pembaca.

VII. REFERENSI

[1] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Pengantar%20Strategi%20Algoritma%20(2015).pdf
[2] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/TugasMakalah2018.pdf
[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Pencocokan-String-(2018).pdf
[4] http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/StringMatch/boyerMoore.htm
[5] http://www.sciedu.ca/journal/index.php/air/article/view/1390
[6] https://blog.javan.co.id/teknik-dasar-web-scraping-aa7d7e223093
[7] https://blog.apify.com/web-scraping-in-2018-forget-html-use-xhrs-metadata-or-javascript-variables-8167f252439c
[8]

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, 14 Mei 2018

Ttd (scan atau foto ttd)

Aldo Azali (13516125)