Aplikasi String Matching dan Regex untuk Memvalidasi suatu Akun pada saat Registrasi

Dwianditya Hanif Raharjanto - 13519046
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519046@std.stei.itb.ac.id / dwihanif.r@gmail.com

***Abstract*—Pada era pandemic covid-19 ini pastilah kita sebagai umat manusia merasakan dampak dengan melakukan banyak aktivitas secara daring seperti sekolah daring, nongkrong daring, dan bekerja secara daring. Tentu dalam melakukan kegiatan daring kita pasti menggunakan suatu aplikasi baik web based maupun handphone based. Disini saya akan membahas mengenai bagaimana suatu sistem dapat membedakan antara satu orang dengan orang yang lain dalam menggunakan sistem tersebut, dimulai dari saat mereka registrasi akun mereka untuk menggunakan sistem. Dengan bantuan algoritma pencocokan string dan regular expression ternyata langkah awal dalam membedakan satu akun dengan akun yang lain dapat terlaksana. Pencocokan string digunakan untuk mengecek apakah email atau username atau elemen yang diharuskan unik sudah terdapat pada suatu database sistem atau belum. Dan regular expression berfungsi untuk memvalidasi data yang diisikan oleh pengguna ketika registrasi untuk menjaga integrias dari database sistem itu sendiri.**

Keywords—pandemi, covid-19, daring, pencocokan string, regular expression, akun, database, registrasi, algoritma.

I. PENDAHULUAN

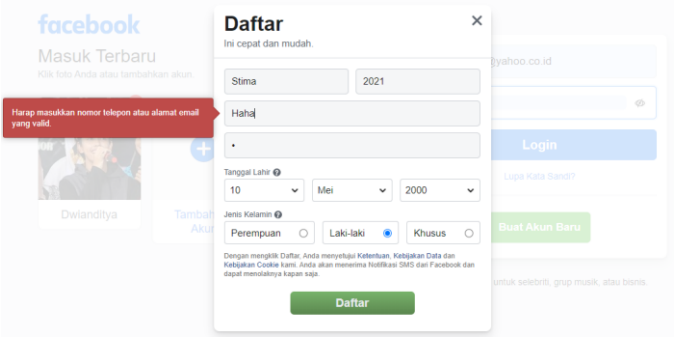
Di era modern seperti sekarang ini layanan-layanan yang dahulunya tersedia secara luring sudah banyak disediakan dalam bentuk daringnya. Hal ini tidak lepas karena musibah Pandemi Covid-19 yang membuat seluruh umat manusia untuk menjalani suatu lifestyle baru yang biasa disebut dengan new normal.

Pandemi Covid-19 ini memaksa kita untuk meminimalkan bertemu secara langsung dengan orang-orang lain dan menganjurkan kita untuk selalu di rumah saja yang bertujuan untuk memutus perkembangan virus Covid-19. Karena di rumah saja kita jadi melakukan banyak hal secara daring contohnya sekolah, bekerja, rapat, nongkrong, dan bahkan lebaran secara daring dan masih banyak kegiatan lain lagi hampir seluruh kegiatan dilakukan secara daring. Bahkan baru-baru ini pemerintah negara Indonesia merilis larangan untuk mudik pada lebaran tahun ini.

Karena mendapatkan dampak Covid-19 yang mengharuskan kita untuk beraktivitas secara daring tentu saja intesintas penggunaan internet meningkat pesat dan banyak orang yang menggunakan aplikasi-aplikasi tertentu untuk menunjang aktivitas daring tersebut. Contoh umumnya adalah sekolah mayoritas menggunakan aplikasi Microsoft Teams atau Google Meet, rapat atau nongkrong juga bisa menggunakan Zoom Meeting atau Google Meet, dan masih banyak aplikasi lain yang digunakan di masa pandemic ini. Selain itu menurut orang-orang yang tidak ingin terhambat keproduktifannya karena pandemic ini mereka mengambil inisiatif untuk mengikuti suatu course online, webinar, workshop dan lain sebagainya untuk menambahkan ilmu pengetahuan, soft skill, hard skill, dan pengalaman mereka.

Namun, pernahkan terlintas dibenak anda bahwa jumlah pengguna untuk aplikasi-aplikasi tersebut akan terus meningkat dari hari ke hari secara eksponensial. Ketika sudah banyak pengguna yang menggunakan bagaimana cara sistem tersebut membedakan pengguna satu dengan yang lainnya. Tentu saja dengan meminta informasi pengguna pada awal sebelum menggunakan aplikasi adalah jawabannya atau yang sering kita kenal dengan registrasi akun pengguna. Tujuan dari registrasi akun sudah jelas untuk mengidentifikasi pengguna yang sudah menggunakan suatu aplikasi tertentu.

Pada saat anda mendaftakan akun pada sebuah aplikasi seperti contoh Facebook, Anda akan diminta untuk mengisi informasi seperti email, nomor telfon, tempat tanggal lahir, password, dan username yang anda inginkan. Sistem dapat membedakan informasi satu pengguna dengan pengguna yang lain ketika informasi tersebut memiliki suatu elemen yang nilainya unik. Biasanya yang diharuskan unik adalah email dan username pengguna. Ketika selesai mengisi informasi yang diminta oleh sistem, sistem akan mengecek secara otomatis dengan data yang ada pada database sitem apakah email dan username yang dimasukkan oleh pengguna sudah digunakan atau belum. Ketika belum digunakan maka registrasi akun pengguna akan berhasil dan informasi yang sudah dituliskan oleh pengguna akan disimpan oleh sistem pada database sistem. Namun, sistem juga melakukan pengecekan untuk memvalidasi alamat email dan password yang diisikan oleh pengguna karena terkadang ada pengguna yang iseng mendaftar dengan memasukkan alamat email secara sembarangan yang tidak sesuai dengan format email pada umumnya dan adanya format password yang harus dipenuhi oleh pengguna seperti minimal delapan karakter dan harus terdapat satu huruf capital atau satu symbol dan lain sebagainya.



Gambar 1.1 Register Page dari Facebook.com

Pengecekan dan validasi saat registrasi ini sangatlah berguna untuk pengguna maupun sistem. Jika dilihat dari sudut pandang pengguna manfaatnya adalah akan meminimalisir kesalahan penginputan data yang typo oleh pengguna karena akan diberikan peringatan ketika terjadi ketidak sesuaian format. Jika dari sudut pandang sistem validasi ini bertujuan untuk meminimalkan data yang tidak valid sehingga integritas suatu database akan lebih terjaga.

II. DASAR TEORI

A. *Pattern Matching*

Pattern Matching adalah suatu proses pencocokan string. Biasanya akan diberikan suatu Teks (T) string yang memiliki panjang N karakter dan suatu Pattern (P) string yang memiliki panjang M karakter dengan asumsi M tidak lebih panjang dari N. Disini Pattern Matching akan mencari lokasi pertama dalam Teks (T) yang bersesuaian denga nisi dari Pattern (P).

CONTOH :

Diberikan suatu Teks T dan Pattern P.

T = Tugas Makalah Strategi Algoritma 2021

P = lah

Pattern Matching ini sudah banyak dimanfaatkan pada zaman modern seperti saat ini, Berikut merupakan contoh pemanfaatan Pattern Matching adalah saat melakukan pencarian pada Editor Text seperti Microsoft Word, pada Web Search Engine seperti Google, pada Analisis Citra, pencocokan rantai asam amino pada rantai DNA di bidang Bionformatics dan masih banyak lagi pemanfaatan lainnya.

Algoritma untuk mencari Pattern Matching sendiri sudah banyak ditemukan. Namun, disini kita akan membahas tiga algoritma pencarian Pattern Matching itu sendiri yaitu algoritma Brute Force, Knuth Morris-Pratt, dan Booyer-Moore. Alasan dari pembahasan ketiga algoritma tersebut karena ketiga algoritma tersebut yang menjadi bahan ajar pada mata kuliah IF2211 Strategi Algoritma 2021.

B. *Brute Force Algorithm*

Seperti yang kita ketahui, algoritma Brute Force ini mutlak akan menemukan suatu solusi untuk suatu masalah yang memiliki solusi tetapi, seperti yang kita ketahui juga algoritma Brute Force ini mutlak menemukan solousi karaena mengunjungi seluruh kemungkinan untuk solusi yang ada pada suatu masalah. Jadi solusi yang didapatkan belum tentu solusi yang paling optimal. Cara kerja dari algoritma Brute Force dalam melakukan pencocokan string adalah sebagai berikut :

1. Pencocokan dimulai dari mencocokan pattern pada awal teks dari kiri ke kanan.
2. Pencocokan dari kiri ke kanan karakter pattern kepada karakter teks satu per satu.
3. Ketika semua karakter pada pattern mendapatkan kesamaan karakter pada teks, maka pattern dinyatakan cocok dengan teks dan pencarian berhenti.
4. Jika karakter pada pattern mengelam ketidaksesuaian dengan karakter yang ada pada teks, maka pattern dan teks akan digeser satu ke kanan dan mengulangi step nomor dua sampai berada pada ujung kanan teks. Perlu ditekankan untuk penggeseran ke kanan teks ini berdasarkan indeks awal pattern yang bersesuaian dengan teks.
5. Ketika sampai ujung kanan teks tidak menemukan kesesuaian, maka dapat disimpulkan bahwa pattern tidak cocok dengan teks yang disediakan.

```
  NOBODY NOTICED HIM
1 NOT
2  NOT
3   NOT
4    NOT
5     NOT
6      NOT
7       NOT
8        NOT
```

Gambar 2.1 Mekanisme Pencocokan String Bruteforce

Berdasarkan cara kerja algoritma Brute Force dapat kita analisa untuk Worst Case nya memalukan jumlah perbandingan sebanyak M(N-M+1) dengan M adalah panjang suatu Pattern dan N adalah panjang suatu Teks. Jika dikonversi dalam notasi big O nya adalah O(MN). Contoh kasus terburuk untuk algoritma brute force dalam pencocokan string adalah ketika Pattern yang cocok berada pada bagian akhir Teks.

CONTOH :

T = aaaaaaaaaaaaaaaaaaaaah

P = aaah

Kemudian untuk kasus terbaiknya memiliki kompleksitas O(N). Hal ini terjadi ketika karakter pertama Pattern tidak pernah sama dengan karakter Teks yang dicocokan. Jadi akan terjadi jumlah perbandingan sebanyak maksimal N kali.

CONTOH :

T = String ini zzz

P = zzz

Terakhir untuk kasus yang sering terjadi atau Average Case memiliki kompleksitas O(M+N) yang bisa terbilang cukup cepat untuk prosesnya.

CONTOH :

T : a string searching example is standard

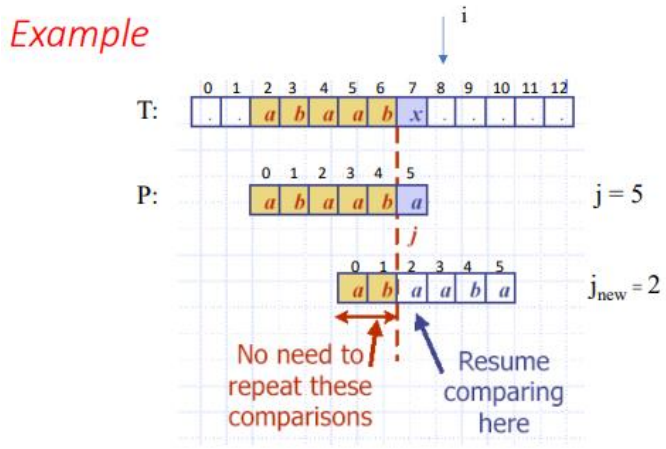
P : store

Algoritma brute force ini menjadi proses yang cepat ketika alphabet dari suatu teks itu besar seperti A..Z, a..z, 1..9, dll. Namun, menjadi semakin lambat ketika alphabet nya kecil seperti 0,1 (pencocokan string pada file binary dll).

C. Knuth Morris Pratt Algorithm

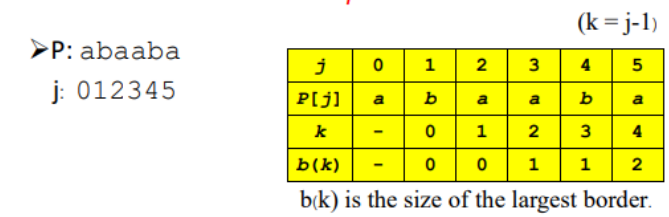
Knuth Morris Pratt algorithm ini mencocokan suatu Pattern pada Teks dengan urutan dari kiri ke kanan (sama sperti algoritma brute force). Namun, pergeseran yang dilakukan lebih cerdas dariapda brute force algorithm sehingga Knuth Morris Pratt ini lebih mangkus dari brute force. Ketika terjadi ketidaksesuaian antara Teks dan Pattern pada P[j] seperti T[i] ≠ P[j] hal apa yang bisa kita lakukan untuk menggeser pattern secara efektif ? Jawabannya adalah menggeser prefix terbesar dari P[0..j-1] yang merupakan suffix dari P[1..j-1]. Pada algoritma KMP ini kita perlu mencari fungsi pembatas atau yang biasa disebut dengan LPS (Largest Prefix Sufix). Cara kerja dari algoritma KMP ini sendiri adalah sebagai berikut :

1. Pencocokan pattern pada awal teks dimulai dari kiri menuju ke kanan (sama seperti bruteforce).
2. Pencocokan karakter pattern pada teks satu per satu dari kiri menuju ke kanan.
3. Ketika semua karakter pada pattern mengalami kesamaan dengan teks, maka dinyatakan pattern tersebut cocok dengan teks dan pencarian diberhentikan.
4. Namun, ketika mendapatkan ketidaksesuaian pattern dengan teks, pattern akan digeser berdasarkan dengan fungsi pembatasnya, lalu mengulangi step nomor dua sampai pattern berada pada ujung kanan teks.
5. Ketika pattern sudah mencapai ujung kanan teks dan belum menemukan kecocokan string, maka dapat disimpulkan bahwa pattern tidak cocok dengan teks dan pencarian diberhentikan.



Gambar 2.2 Mekanisme pencocokan string KMP Algorithm

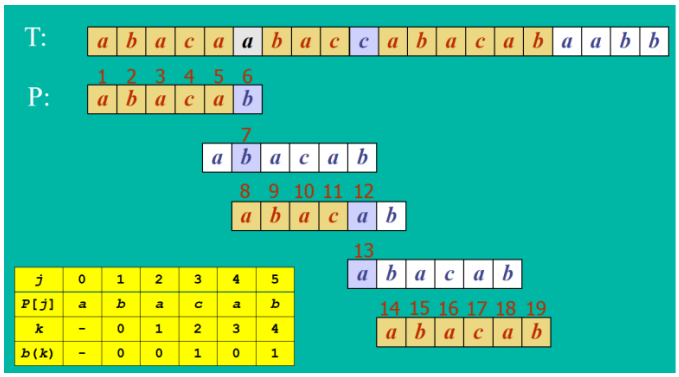
Proses pencocokan string KMP ini memerlukan fungsi pembatas terlebih dahulu yang bertujuan untuk preproses terhadap Pattern untuk mencari suatu pattern dalam pattern itu sendiri. Hal ini biasa disebut dengan prefix. Fungsi pembatas ini merupakan ukuran dari prefix terpanjang dari P[1..l] yang juga merupakan suffix dari P[1..k] seperti contoh :



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

Gambar 2.3 Contoh Pengaplikasian Fungsi Pembatas

Hal inilah yang membedakan antara brute force dengan KMP. Ketika terjadi ketidaksesuaian KMP akan melihat dari fungsi pembatasnya untuk menentukan pergeseran string yang dibandingkan. Nilai dari fungsi pinggiran akan menandakan posisi mulai pencocokan baru pada Pattern untuk Teks. Ketika posisi pada pattern sudah kembali pada posisi yang awal, Teks akan mulai maju satu indeks dan seterusnya.



Gambar 2.4 Contoh Alur Algoritma KMP

Semisal berdasarkan gambar di atas. Kesesuaian string terjadi dari indeks pertama hingga kelima pada karakter pattern dan teks. Indeks keenam mengalami ketidaksesuaian. Yang dilakukan oleh KMP adalah menggeser pattern ke indeks yang nilainya sama dengan prefix terpanjang yang ada, dalam gambar nilai prefix terpanjang adalah satu, jadi menggeser ke indeks satu pada pattern yaitu karakter b. Kita lakukan pencocokan string lagi antara indeks keenam teks dengan indeks satu pattern. Ketika mengalami ketidaksesuaian indeks pattern akan menggeser lagi ke indeks yang nilaianya sama dengan prefix terpanjang yang ada saat itu. Karena prefix terpanjangnya nol maka kembali pada indeks awal pattern dan baru dilakukan penggeseran satu indeks ke kanan pada teks (dari indeks ke enam menuju indeks ke tujuh).

Kompleksitas waktu untuk KMP sendiri adalah ketika menghitung fungsi pembatasnya adalah O(M) dan ketika pencaria string nya adalah O(N) jadi kompleksitas waktu algoritma KMP adalah O(M+N). Sangat lebih cepat dibandingkan dengan Brute Force. Keuntungan dari KMP adalah tidak perlu bergerak mundur ketika membandingkan Teks. Kekurangan dari KMP adalah tidak bekerja dengan baik ketika ukuran dari alphabet ditambah.

D. Boyer-Moore Algorithm

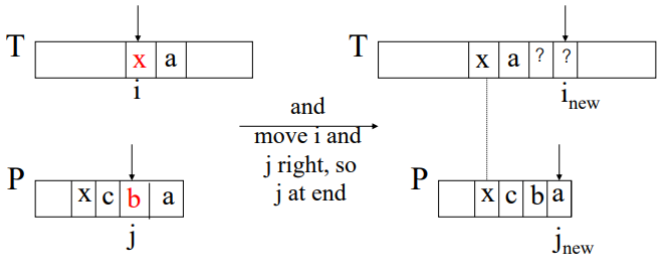
Algoritma pencocokan string yang satu ini berbeda dengan dua yang pertama karena pencocokan string untuk algoritma Boyer Moore ini dimulai dari kanan. Cara kerja algoritma ini adalah sebagai berikut :

1. Pencocokan pattern pada awal teks dimulai dari kanan.
2. Pencocokan karakter pattern pada teks satu per satu dari kanan menuju ke kiri.
3. Ketika semua karakter pada pattern mengalami kesamaan dengan teks, maka dinyatakan pattern tersebut cocok dengan teks dan pencarian diberhentikan.
4. Namun, ketika mendapatkan ketidaksesuaian pattern dengan teks, pattern akan digeser berdasarkan dengan nilai last occurrence-nya, lalu mengulangi step nomor dua sampai pattern berada pada ujung kanan teks.
5. Ketika pattern sudah mencapai ujung kanan teks dan belum menemukan kecocokan string, maka dapat disimpulkan bahwa pattern tidak cocok dengan teks dan pencarian diberhentikan.

Sebenarnya inti dari Boyer Moore ini ada dua teknik yaitu teknik looking-glass dan teknik character-jump. Teknik looking-glass ini bermaksut untuk mencari suatu Pattern P dalam T dengan bergerak secara mundur dari P, berawalan pada akhir P. Teknik character-jump ini bermaksut ketika ketidaksesuaian ditemukan saat T[i] == x maka karakter pada pattern P[j] tidak sama dengan karakter T[i]. Terdapat tiga kasus yang mungkin terjadi pada hal ini.

1. Kasus 1

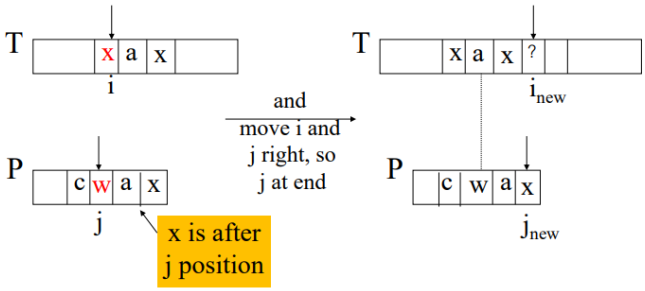
Ketika P memiliki x di suatu tempat, kemudian mencoba menggeser P ke kanan sesuai dengan kemunculan terakhir x di P dengan T[i].



Gambar 2.5 Visualisasi kasus 1 BM

2. Kasus 2

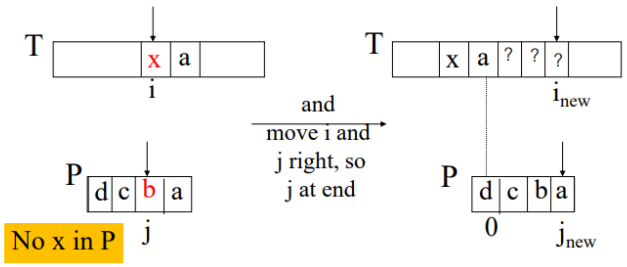
Ketika P memiliki x di suatu tempat, kemudian kemungkinan untuk menggeser ke kanan ke kemunculan terakhir x itu tidak mungkin, maka geser P ke kanan dengan satu karakter ke T[i+1].



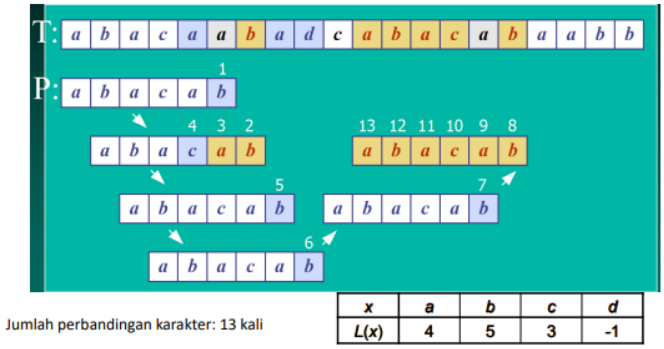
Gambar 2.6 visualisasi kasus 2 algoritma BM

3. Kasus 3

Ketika kasus satu dan dua tidak sesuai, maka geser P ke P[0] dengan T[i+1]



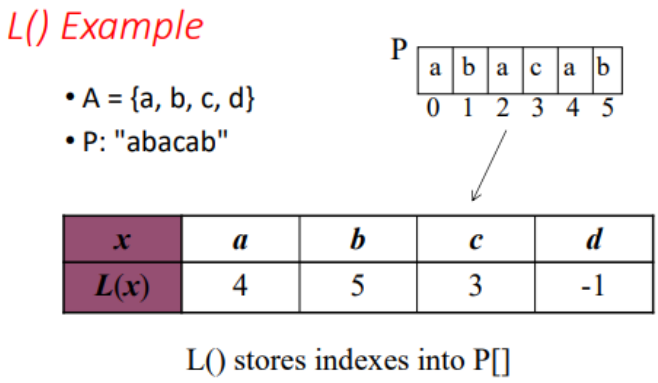
Gambar 2.7 visualisasi kasus 3 algoritma BM



Gambar 2.8 Alur pencocokan string algoritma BM

Karena Boyer-Moore butuh fungsi untuk menghitung occurrence. Berikut adalah cara untuk menghitung occurrence tersebut.

1. Mempreproses pattern P dan alphabet A untuk menemukan fungsi L() (last occurrence).
2. L(x) terdefinisi sebagai indeks terbesar seperti P[i] == x atau -1 ketika indeks tidak ada.



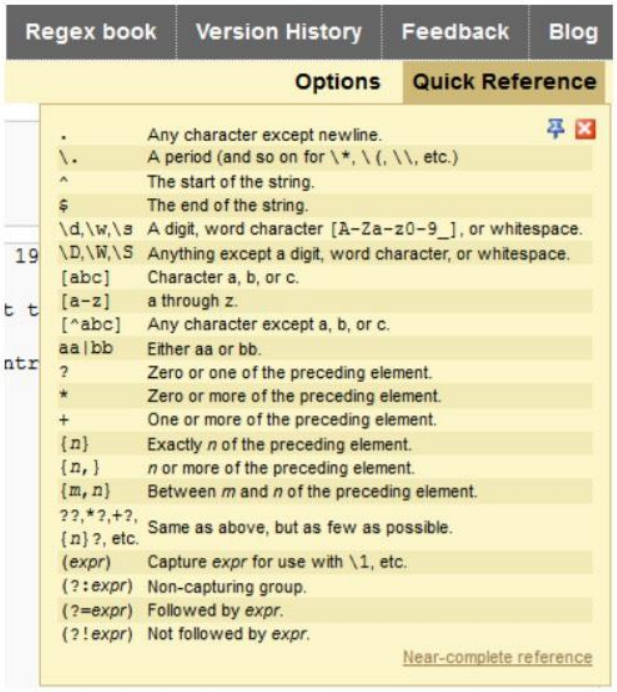
Gambar 2.9 Preproses dalam menentukan last occurrence

Kompleksitas waktu untuk algoritma Boyer-Moore ini jika mendapatkan kasus terburuk adalah O(NM+A). Namun, Boyer-Moore ini cocok ketika mendapatkan kasus alphabet (A) itu beragam, tidak cocok ketika kasus alphabet (A) tidak beragam. Boyer-Moore ini juga perbedaannya signifikan lebih cepat daripada brute force untuk pencarian teks berbahasa Inggris.

E. Regular Expression

Regular Expression atau yang lebih sering dikenal Regex ini merupakan suatu notasi dan karakter yang digunakan untuk suatu "keyword" dalam pendeskripsian suatu pola pada pencarian berbasis huruf. Dengan regex ini kita dapat mengenali pola string yang mempunyai karakteristik tertentu seperti contohnya adalah email, password, nomor hp, kartu kredit, kode mata kuliah dan lain sebagainya.

Contoh pengaplikasian regex adalah "/.udi/ yang akan menerima seluruh string yang mengandung udi. Karakter titik pada regex memiliki arti karakter apapun pada sebelah kiri. Kemudian backslash digunakan ketika ingin menggunakan metacharacter seperti titik. Berikut ada lampiran notasi regex yang ada.



Gambar 2.10 Notasi Regular Expression

Dari contoh yang simpel di atas, notasi regex tentu dapat dikombinasikan menjadi suatu notasi yang rumit yang dapat mendefinisikan sesuatu pola. Seperti contoh regex yang mendefinisikan pola nomor hp di luar negeri "(\(?\d{3}\)?[-])+\d{4}". Ketika pengisian string tidak sesuai dengan aturan tersebut akan menampilkan pesan eror.

III. ANALISIS DAN PEMBAHASAN

Pada validasi yang saya inginkan sebenarnya memiliki dua hal utama yang akan divalidasi. Pertama adalah memvalidasi apakah masukan dari pengguna sudah sesuai dengan format atau pola yang diinginkan oleh sistem atau belum. Hal ini untuk menjaga agar isi data dari database sistem tetap terjaga integritasnya.

Kedua adalah mengecek apakah data yang diisikan oleh pengguna sudah ada pada database atau belum. Istilah mudahnya mungkin mengecek apakah data yang dimasukkan masih tersedia atau tidak. Hal ini bertujuan agar sistem bisa membedakan antara satu akun dengan akun lainnya.

Untuk kasus pertama ini kita menggunakan regular expression untuk mengecek kevalidan masukan dari pengguna. Kasus yang akan kita validasi disini adalah format email dan password dari pengguna. Seperti yang kita ketahui bahwa format email pada umunya adalah karakter bebas terlebih dahulu kemudian ada symbol @ setelah karakter bebas dan

yang terakhir adalah karakter huruf kecil dan titik. Kita ambil contoh ketika kita menginput hanif@yahoo.com maka sistem akan membenarkan pola tersebut. Namun jika kita menginput hanifyahoo.com maka sistem akan memberikan pesan eror karena tidak sesuai dengan format yang diinginkan sistem. Regex yang digunakan untuk mengecek format email valid atau tidak adalah sebagai berikut

$email = "/[\w]+@[a-z.]+/"

Penjelasan dari regex di atas sudah sesuai dengan ketentuan yang penulis jelaskan di atas. Pola pertama adalah username email (sebelum tanda @). Pola ini sebenarnya bisa kita tuliskan sebagai berikut [a-zA-z0-9_] tetapi secara singkat juga bisa diartikan menggunakan metacharacter \w. Kemudian pola yang kedua adanya symbol @. Pola yang ketiga adalah domain yang biasanya terdiri dari karakter huruf kecil dan titik.

Kemudian untuk mengecek apakah format password yang diinput oleh pengguna sudah benar atau belum bisa yang pertama kita tentukan dulu pola yang kita inginkan untuk password pengguna. Misalkan password pengguna harus mengandung huruf capital, huruf non capital, angka, dan minimal delapan karakter. Maka regex yang akan kita buat adalah sebagai berikut.

$password =

"^\S*(?=\S{8,})(?=\S*[a-z])(?=\S*[A-Z])(?=\S*[\d])\S*$"

Penjelasan dari regex di atas adalah sebagai berikut, ^ adalah penanda mulai dari suatu string, \S* ini menandakan seluruh bentuk karakter boleh digunakan, (?=\S{8,}) ini menandakan bahwa panjang karakter minimal delapan, (?=\S*[a-z]) ini menandakan bahwa harus berisi setidaknya satu karakter non capital, (?=\S*[A-Z]) menandakan bahwa harus berisi setidaknya satu karakter capital, (?=\S*[\d]) menandakan bahwa harus setidaknya berisi satu angka, $ menandakan bahwa itu akhir dari sebuah string.

Kemudian untuk meninjau kasus yang kedua kita bisa menggunakan string matching untuk mencocokan apakah sudah pernah ada sebuah data dalam database sistem. Seperti yang kita ketahui untuk membedakan antara satu akun dengan akun yang lain sistem meminta sebuah informasi yang diharuskan unik seperti contohnya email dari pengguna harus unik satu dengan yang lain, username pengguna terkadang juga harus unik. Hal ini dapat kita cek ketersediaan email dan username dengan cara melakukan string matching antara masukan pengguna dengan data yang ada dalam database sistem. Cukup simpel saja pada program nanti akan kita iterasi untuk setiap data yang berada di database kita panggil fungsi string matching yang memiliki parameter untuk patternya adalah masukan dari pengguna dan teksnya adalah data dari database, kita cek apakah fungsi string matching menemukan keluaran yang menemukan kecocokan string pada masukan pengguna dengan data yang ada pada database. Jika misal menemukan kecocokan maka akan muncul pesan eror pada laman registrasi yang menandakan bahwa email atau username tersebut tidak tersedia untuk digunakan karena sudah ada yang menggunakan. Jika

KESIMPULAN

Algoritma pencocokan string dan regular expression memang dapat dimanfaatkan dalam bidang memvalidasi suatu akun saat registrasi baik untuk mendeteksi suatu kesalahan pengisian maupun mendeteksi sudah sudah pernah dibuatnya akun tersebut pada suatu sistem.

Dengan pemanfaatan kedua algoritma di atas, secara tidak langsung pengguna dan sistem merasakan manfaatnya. Dari kacamata pengguna meminimalkan kesalahan data yang diisikan karena ketika mengalami kesalahan akan muncul pesan eror dan untuk sistem menjaga keintegritasan isi data dari database karena data yang diisikan sudah digiring untuk memenuhi format yang ada.

Namun, regex disini hanya dapat membantu sistem untuk memvalidasi suatu elemen berdasarkan formatnya saja, belum bisa memastikan betul apakah suatu elemen yang diisikan (contoh email) itu sudah 100% benar adanya. Bisa saja pengguna mengisikan alamat email yang sesuai format tetapi aslinya belum terdaftarkan pada platform email manapun.

Jadi untuk penutupan, algoritma string matching dan regex ini hanyalah sebagai validasi sederhana untuk sebuah program validasi akun.

VIDEO LINK AT YOUTUBE

https://youtu.be/zxXifYx1580

UCAPAN TERIMAKASIH

Penulis tidak lupa mengucapkan terima kasih serta puja dan puji syukur kehadirat Tuhan Yang Maha Esa karena karunia dan berkat rahmatnya penulis dapat menyelesaikan makalah ini dengan tepat waktu dan tanpa kurang apapun.

Tak lupa penulis juga mengucapkan terima kasih kepada seluruh tim pengajar mata kuliah IF2211 Strategi Algoritma 2021 yang telah membimbing penulis dengan sabar dan penuh kasih sayang selama satu semester ini. Penulis tidak tahu apa jadinya jika tidak mendapat bimbingan dari tim pengajar IF2211 Strategi Algoritma 2021. Kemudian penulis juga mengucapkan terima kasih kepada tim asisten mata kuliah IF2211 yang telah mengawal penulis dalam memberikan ilmu melalui tugas besar dan tugas kecil yang ada.

Ucapan terima kasih selanjutnya penulis ucapkan untuk teman-teman penulis yang memberikan dukungan penulis baik langsung maupun tidak langsung dalam segala bentuk dukungan.

Terakhir, penulis mengucapkan terima kasih kepada keluarga penulis khususnya orang tua penulis yang sudah memberika support dari awal hingga saat ini yang memberikan dampak kepada proses keberjalanan dalam membuat makalah IF2211 Strategi Al

REFERENCES

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

[3] Lecroq, Thierry Charras, Christian. 2001. Handbook of Exact String Matching Algorithm. ISBN 0-9543006-4-5

[4] https://jagongoding.com/web/php/menengah/regular-expression/

[5] https://stackoverflow.com/questions/8141125/regex-for-password-php

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.

Semarang, 11 Mei 2021

Dwianditya Hanif - 13519046