Perbadingan Waktu Eksekusi Algoritma Boyer Moore dengan Algoritma Knuth Morris Pratt

Muhammad Akmal Arifin - 13520037[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13520037@std.itb.ac.id

Abstract*—String matching merupakan salah satu permaslahan yang ada di dunia ilmu informatika. Permaslahan ini mampu menggambarkan berbagai permasalahan yang ada di dunia. Dengan ditemukannya solusi dari permaslahan ini, ditemukan pula manfaat-manfaat lainnya untuk kehidupan manusia. Salah satu solusi dari permaslahan string matching adalah dengan algoritma boyer moore dan algoritma knuth morris pratt. Kedua algoritma ini berhasil memecahkan permaslahan ini dengan time complexity linear yaitu O(m+n). Meskipun keduanya memiliki time complexity yang sama, masing-masing dari solusi ini memiliki kondisi dimana dia dapat bekerja lebih efisien dari yang lain.

Keywords*— Algoritma Boyer Moore, Algoritma Knuth Morris Pratt, String Matching.

I. PENDAHULUAN

Mulai dari abad ke-21 ini, dunia terus berkembang semakin cepat. Ditemukannya teknologi-teknologi baru yang terus mempermudah manusia dalam menjalankan kehidupannya di dunia sehari-hari. Hampir seluruh aspek kehidupan manusia kini telah terpengaruhi oleh teknologi. Mulai dari yang mengubah bagaimana mausia berinteraksi, seperti gawai, laptop, dan internet. Sampai hingga yang hanya menambah alternatif solusi permasalahan, seperti alat masak dan lain-lain. Sampai saat ini, manusia terus menerus mencoba untuk mengembangkan dunia, salah satunya dengan cara menyelesaikan permaslahaan-permaslahan yang pernah ada. Dari hasil solusi permasalahan tersebut, dunia dapat berkembang menjadi lebih modern.

Salah satu permasalahan yang ada yang kini telah ditemukan solusinya ada permasalahan pencarian kata dalam kalimat atau yang biasa disebut *string matching*. *String matching* merupakan permaslahan untuk menentukan dimana letak suatu kata di dalam kalimat. Beberapa permasalahan di dunia yang ada yang merupakan permasalahan string matching antara lain ialah *spell checkers, spam detection system / spam filters, intrusion detection system, search engines / content searching in large databases, plagiarism detection digital forensics,* dan *information retrieval*. Tidak hanya untuk keperluan tekknologi saja, tetapi juga ada manfaatnya untuk bidang lain, yaitu seperti bidang biologi yaitu *bioinformatics* / *DNA sequencing*.

String matching kemudian menjadi salah satu permasalahan konvensional yang ada di bidang ilmu komputer. Di sini kita mencari string di dalam string yang lebih besar. String yang lebih kecil biasa disebut dengan pattern, sedangkan string yang lebih besar biasa disebut dengan text. Masalah ini berkaitan dengan menemukan apakah string pattern terdapat pada string text. Apabila pattern terdapat pada text, maka program harus dapat mengembalikan lokasi dari string pattern pada text.

Kini telah ditemukan berbagai macam algoritma untuk menyelesaikan permaslahan ini. Mulai dari yang paling sederhana yaitu brute force, hingga yang rumit seperti Algoritma Boyer Moore Horspool Sunday. Salah satu algoritma yang cukup terkenal dan sederhana untuk diaplikasikan adalah Algoritma Booyer Moore (BM) dan Algoritma Knuth Morris Pratt (KMP).

Di dalam makalah ini akan dijelaskan bagaimana perbandingan efektif dan efisien dari algoritma Booyer Moore (BM) dan algoritma Knuth Morris Pratt (KMP). Dan menjelaskan secara singkat kelebihan dan kekurangan dari masing-masing algoritma.

II. LANDASAN TEORI

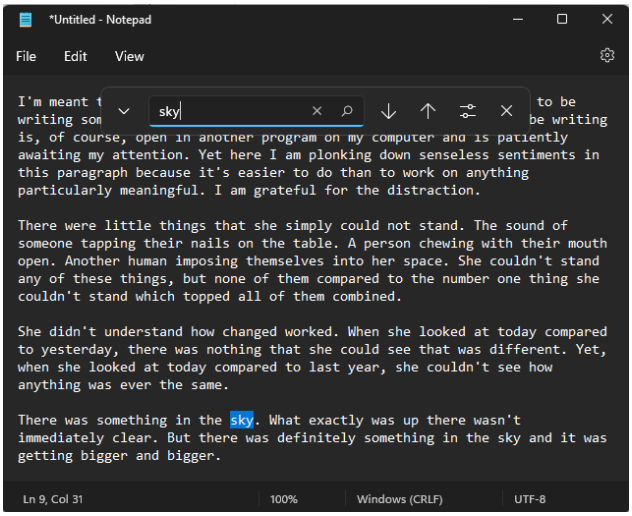
A. String Matching

String matching atau pencocokan string merupakan salah satu permaslaahan yang ada di dunia ilmu komputer. Permasalahan ini berkaitan dengan pencarian suatu huruf, kata, atau kalimat di sebuah teks. Istilah teks (T) digunakan untuk string panjang. Istilah *pattern* (P) digunakan untuk string yang akan dicari di dalam teks. Permasalahan ini akan mengeluarkan index atau lokasi dari string pattern yang terdapat di dalam teks.

Terdapat dua konsep pada string, yaitu prefiks dan sufiks. Prefix dari S berarti substring dari S yang mencakup nilai S dari indeks 0 hingga indeks ke-*k*. Sedangkan sufiks dari S berarti substring dari S yang mencakup nilai S dari indeks k hingga indeks terakhir dari S. Sebagai contoh, misalkan S adalah “Andrew”. “And” merupakan prefix dari S, sedangkan “ndr” bukan merupakan prefix dari S. “rew” merupakan sufiks dari S, sedangkan “dre” bukan merupakan sufiks dari S.

Saat ini penyelsaian algoritma *string matching* memiliki banyak sekali manfaat, baik untuk dunia komkputer itu sendiri, begitu juga dengan bidang lain. Seperti permasalahannya, algoritma string matching juga bermanfaat untuk mencari sebuah kata dalam teks editor, contohnya yaitu notepad atau

word. Mesin pencarian kata di laman web juga menggunakan string matching untuk melakukan *query data* berdasarkan masukkan yang kita beri. Kita juga dapat menggunakan string matching untuk melakukan analisis terhadap gambar. Dan yang terakhir, dalam ilmu Biologi kita mengenal yang Namanya DNA. DNA ini merupakan singkatan dari *dexyribonucleic*. DNA menyimpan informasi dari suatu makhluk hidup. Dengan mengetahui kemirian antar DNA satu sama lian, kita dapat mengetahui banyak hal, seperti keturunan dan penyakit.



Gambar 1. Pencarian kata pada text editor Notepad

Sumber: Dokumentasi Pribadi

B. Algoritma Brute Force

Definisi algoritma brute force merupakan salah satu pola penyelesaian suatu permasalahan dengan pendekatan *straight forward*. Algoritma brute force sering kali dibilang algoritma naif dikarenakan pendekatan solusi yang digunakan tidak memerlukan berpikir dengan dalam terlebih dahulu. Pada umumnya algoritma brute force didasarkan pada pernyataan persoalan atau konsep yang dilibatkan.

Kelebihan dari algoritma brute force ialah algoritma ini hampir dapat diterapkan untuk memecahkan sebagian besar permasalahan yang ada. Selain itu juga, algoritma yang dihasilkan sangat sederhana dan mudah sekali untuk dimengerti. Brute force juga sering kali digunakan untuk membandingkan peforma suatu algoritma, apakah algoritma tersebut lebih efisien dibandingkan dengan algoritma brute force.

Akan tetapi, algoritma ini juga memiliki banyak sekali kekurangan. Hampir seluruh algoritma yang dihasilkan menggunakan brute force, bukanlah algoritma yang efisien. Peformanya cenderung lambat dan membutuhkan memori yang sangat besar. Solusi permasalahan yang terlalu *straight forward* dan kurang kreatif.

Cara kerja algoritma brute force untuk menyelesaikan permasalahan string matching adalah dengan cara melakukan iterasi membandingkan karakter yang ada pada pattern dengan text ke setiap karakter yang ada di di text. Apabila karakter yang dibandingkan antara *text* dan *pattern* sama, maka akan dilanjutkan dengan indeks selanjutnya untuk text dan pattern. Apabila karakter yang dibandingkan tidak sama, maka akan dilakukan *back tracking*, Kembali ke awal indeks *pattern*, kemudian melanjutkan ke indeks selanjutnya pada *text*.

T	A	K	U		I	K	A	N	
P	I	K	A						
		I	K	A					
			I	K	A				
				I	K	A			
					I	K	A		

Tabel 1. Contoh string matching menggunakan algoritma brute force

Jika kita asumsikan panjang karakter *text* adalah sebesar n, dan panjang karakter *pattern* adalah sebesar m. Maka analisis kompleksitas waktu pada solusi algoritma *brute force* untuk permasalahan *string matching* ini adalah;

1. Worst case atau kemungkinan terburuk dari algoritma ini adalah ketika *pattern* ditemukan berada pada akhir dari *text*, atau *pattern* tidak ditemukan sama sekali pada *text*. Contoh:

   Text : aaaaaab

   Pattern : aaab

   Total perbandingan yang dilakukan adalah sebanyak 16 kali. Dengan time complexitynya yaitu O(m*(n-m+1)) = O(mn)

2. Best case atau kemungkinan terbaik dari algoritma ini adalah ketika pattern yang ditemukan berada pada awal dari text. Contoh

   Text : abababab

   Pattern : aba

   Total perbandingan yang dilakukan adalah sebanyak 3 kali. Dengan time complexitynya yaitu O(m).

3. Average case atau rata-rata kemungkinan dari algoritma ini adalah ketika pattern yang ditemukan tidak berada di awal ataupun di akhir dari text. Contoh

   Text : aaaabbaaaa

   Pattern : bb

   Total perbandingan yang dilakukan adalah sebanyak 7 kali. Dengan time complexitynya yaitu O(m+n).

C. Algoritma Boyer Moore (BM)

Algoritma Boyer Moore merupakan algoritma yang ditemukan oleh Robert S. Boyer dan J. Strother Moore di tahun 1977. Algoritma ini digunakan utnuk menyelesaikan permasalahan string matching. Algoritma ini merupakan salah

satu algoritma *string matching* paling efisien jika dibandingkan dengan algoritma *string matching* lainnya. Dikarenakan hal tersebut, banyak aplikasi yang menggunakan algoritma string matching lebih memilih menggunakan algoritma ini dibandingkan dengan algoritma lainnya.

Algoritma ini bekerja dengan berdasarkan dua teknik, yaitu

1. *Looking-glass technique.* Maksud dari Teknik ini ialah ketika melakukan perbandingan antara text dan pattern. Iterasi pada pattern dilakukan secara mundur, yaitu dari indeks paling belakang ke depan.
2. *Character*-jump technique. Maksud dari Teknik ini ialah ketika tejadi ketidakcocokan saat melakukan perbandingan antara text dan pattern, pattern tidak hanya langsung digeser, akan tetapi terdapat 3 kemungkinan yang dapat terjadi. Kita asumsikan S merupakan variable berisi string untuk *text* dan P merupakan variable berisi string untuk *pattern*. Terjadi mismatch pada S[i] dengan P[j]
    a. *Case 1*, apabila string P memiliki karakter S[i] yang berada di sisi kiri atau sebelum P[j], maka P akan bergser ke kanan hingga S[i] sejajar dengan karakter yang sama yang dimiliki oleh string P.
    b. *Case 2*, apabila string P memiliki karakter S[i] namun karakter tersebut berada di sisi kanan atau setelah P[j], maka P akan bergeser ke kanan hingga P[0] sejajar dengan S[i+1]/
    c. *Case 3*, apabila string P tidak memiliki karakter S[i], maka P akan bergeser ke kanan hingga P[0] sejajar dengan S[i+1].

Dalam penerapan algoritma boyer moore terdapat fungsi yang dinamakan *Last Occurrence*. Fungsi ini digunakan untuk menyimpan index dari huruf terakhir pada pattern yang terlibat dalam string matching. Fungsi ini nantinya akan digunakan untuk melakukan pergeseran ketika terjadi mismatch. Contoh fungsi *last occurrence*:

P	a	b	a	c	a	b

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

Tabel 2. Contoh Last Occurrence

X merupakan karakter-karakter yang muncul pada text beserta pattern. L(x) merupakan nilai dari *last occurrence function*. Sedangkan P merupakan string pattern.

Contoh penyelesaian permasalahan menggunakan algoritma boyer moore:

T	**a**	**b**	**a**	**c**	**a**	**a**	**b**	**a**	**d**	**c**	**a**	**b**	**a**	**c**	**a**	**b**
P	a	b	a	c	a	b										
		a	b	a	c	a	b				a	b	a	c	a	b
			a	b	a	c	a	b		a	b	a	c	a	b	
				a	b	a	c	a	b							

Tabel 3. Contoh string matching menggunakan algoritma boyer moore

Jika kita asumsikan panjang karakter *text* adalah sebesar n, dan panjang karakter *pattern* adalah sebesar m. Maka analisis kompleksitas waktu pada solusi algoritma boyer moore untuk permasalahan *string matching* ini adalah O(nm + A), dengan A merupakan ukuran banyaknya karakter yang terlibat.

D. Algoritma Knuth Morris Pratt (KMP)

Algoritma Knuth Morris Pratt merupakan algoritma yang ditemukan oleh Donald Knuth dan Vaughan Pratt, serta James H Morris pada tahun 1970. Algoritma ini digunakan untuk menyelesaikan permasalahan string matching. Algoritma ini merupakan algoritma pertama yang berhasil menyelesaikan permasalahan *string matching* dengan kompleksitas waktu linear O(N).

Berlawanan dengan algoritma boyer moore, algoritma ini melakukan iterasi dari kiri ke kanan. Akan tetapi pada proses pergeseran untuk melakukan string matching lebih pintar dibandingkan dengan cara *brute force*. Cara kerja dari algoritma ini sama seperti dengan algoritma *brute force*. Akan tetapi, ketika terjadi mismatch, kita misalkan S merupakan string text dan P merupakan string pattern, terjadi mismatch pada S[i] dan P[j]. Ketika terjadi mismatch, program akan mencari suffiks dari S[k..i-1] terbesar yang sama dengan prefix dari P[0..k].

Dalam penerapan algoritma knuth morris pratt terdapat fungsi yang dinamakan dengan Border Function. Fungsi ini digunakan untuk menyimpan ukuran terbesar dari prefix P[0..k] yang juga merupakan P[1...k]. Fungsi ini nantinya akan digunakan untuk melakukan pergeseran ketika terjadi mismatch. Contoh fungsi Border adalah:

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

Tabel 4. Contoh Border Function

j merupakan posisi dimana mismatch terjadi pada P[]. K merupakan posisi sebelum mismatch (k = j-1). b(k) merupakan nilai dari *border function*.

Contoh penyelesaian permaslahan menggunakan algoritma knuth morris pratt:

T	**a**	**b**	**a**	**c**	**a**	**a**	**b**	**a**	**c**	**c**	**a**	**b**	**a**	**c**	**a**	**b**
P	a	b	a	c	a	b										
					a	b	a	c	a	b						
						a	b	a	c	a	b					
										a	b	a	c	a	b	
											a	b	a	c	a	b

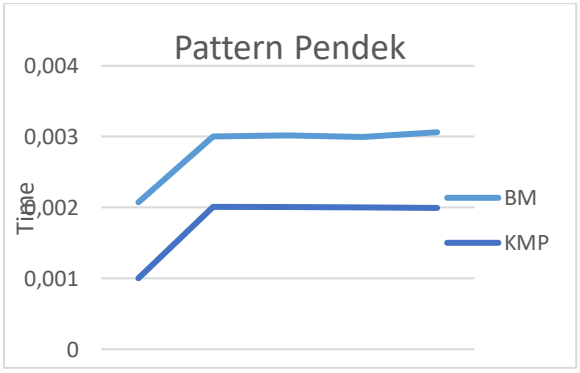
Tabel 5. Contoh string matching menggunakan algoritma knuth morris pratt

Jika kita asumsikan panjang karakter *text* adalah sebesar n, dan panjang karakter *pattern* adalah sebesar m. Maka analisis kompleksitas waktu pada solusi algoritma boyer moore untuk permasalahan *string matching* ini adalah O(m+n).

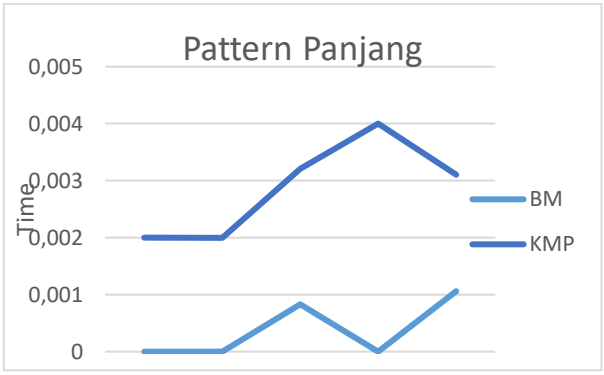
III. Analisis Persoalan

Berdasarkan landasan teori diatas, kita dapat mengetahui bahwa time complexity dari algoritma boyer moore dan algoritma knuth morris pratt adalah sama yaitu O(m+n), namun manakah diantara kedua algoritma tersebut yang memiliki efisiensi program yang terbaik. Seperti yang kita ketahui bahwa semakin berkembangnya zaman, kita akan menhadapi semakin besar data, oleh karena itu perbedaan waktu yang sedikit akan memberikan dampak yang besar ketika kita menerapkannya pada data yang sangat besar.

Dalam percobaan ini, penerapan algoritma knuth morris pratt dan boyer moore menggunakan bahasa Python dan dijalankan melalui *command prompt*. Program akan dijalankan pada komputer yang sama. Nantinya akan dilakukan beberapa test untuk mencari tau algoritma mana yang lebih cepat dalam persoalan tertentu. Akan diambil data dengan dua jenis, yaitu dengan panjang *pattern* pendek dan panjang *pattern* cukup panjang.



Grafik 1.. Hasil percobaan dengan pattern pendek terhadap waktu



Grafik 2.. Hasil percobaan dengan pattern panjang terhadap waktu

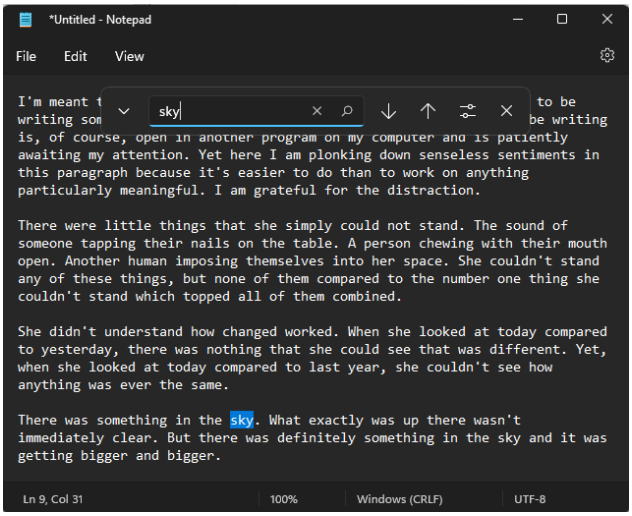
Berdasarkan kedua grafik yang ada diatas terlihat perbedaan yang sangat signifikan. Pada grafik pattern pendek, terlihat bahwa algoritma knuth morris pratt pada umumnya jauh lebih cepat dibandingakn dengan algoritma boyer moore. Sedangkan pada grafik pattern panjang, terlihat bahwa algoritma boyer moore lah yang lebih cepat. Disini terlihat hasil yang terbalik.

Hal ini bisa terjadi dikarenakan pola yang digunakan kedua algoritma ketika melakukan pergeseran berbeda konsep. Algoritma knuth morris pratt menggunakan table border function, table tersebut akan terbentuk sepanjang pattern yang dimiliki, maka semakin Panjang pattern, juga akan semakin Panjang pula table. Sedangkan pada algoritma boyer moore menggunakan last occurrence yang dimana menggunakan *set of alphabet* yang digunakan pada text maupun pattern. Panjang pattern tidak akan terlalu berpengaruh, selama *set of alphabet*-nya sama

IV. Aplikasi

A. Pencarian Pada Text Editor

Text editor merupakan suatu jenis perangkat lunak komputer yang digunakan untuk melakukan edit pada teks. Text editor dapat digunakan untuk membuat dokumen-dokumen yang diperlukan oleh manusia. Berkat text editor ini kita menjadi dipermudah dalam mengurus sebuah dokumen. Pada aplikasi text editor, terdapat penerapan algoritma string matching pada pencariannya. Hal ini akan sangat berguna apabila kita ingin mencari sebuah kata di dalam sebuah teks.



Gambar 2. Pencarian kata pada text editor Notepad

Sumber: Dokumentasi Pribadi

B. Web Search Engine

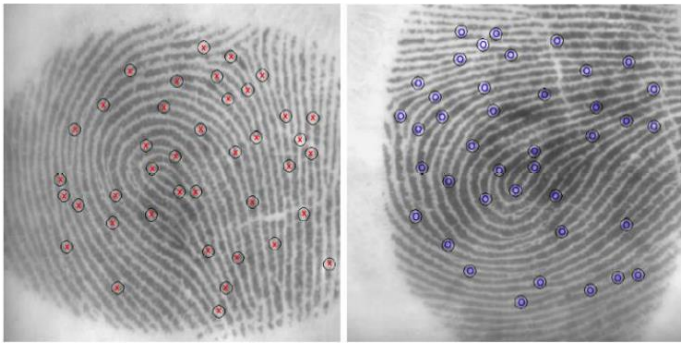
Saat ini internet telah menjadi bagian dari kehidupan. Internet memiliki banyak sekali kegunaan, baik untuk ilmu Pendidikan, penerapan, maupaun yang lainnya. Begitu pula dengan web, situs web merupakan sekumpulan halaman web yang saling berhubungan yang umumnya berisikan mengenai informasi terkait suatu hal. Di dalam web terdapat situs yang digunakan untuk mencari suatu laman web yang fungsinya untuk mencari laman web lainnya, sehingga kita dapat mengaksesnya. Situs ini dinamakan web search engine. Di dalam situs tersebut, terdapat aplikasi dari algoritma string matching pada pencariannya

Gambar 3. Ilustrasi web search engine

Sumber: www.searchenginejournal.com

C. Analisis Citra

Ternyata tidak hanya digunakan untuk melakukan pencarian pada text, algoritma string matching juga dapat melakukan pencarian dalam gambar. Karena gambar sejatinya terdiri atas data-data terkait warna yang digunaknnya pada setiap pixel. Oleh karena itu, algoritma string matching dapat digunakan untuk melakukan analis. Salah satu analisis citra yang bisa dilakukan dengan menggunakan algoritma string matching ialah mendeteksi pola pada sidik jari.

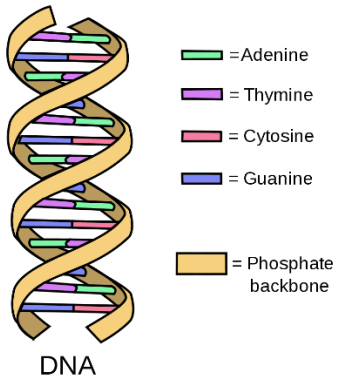


Gambar 4. Ilustrasi analisis pola pada sidik jari menggunakan algoritma string matching

Sumber: PPT Materi Kuliah String Matching

D. Bioinformatics

Bioinformatics merupakan ilmu yang mempelajari penerapan Teknik komputasi untuk mengelola dan menganalisis informasi hayati. Pada umumnya, bioinformatika didefinisikan sebagai aplikasi dari alat komputasi dan Analisa untuk menangkap dan menginterpretasikan data-data biologi. Ilmu ini merupakan ilmu baru yang merangkup berbagai disipilin ilmu mulai dari ilmu komputer, matematika dan fisika. Salah satu manfaat dari algoritma string matching untuk menyelesaikan permasalahan yang ada adalah pada permasalahan pencocokan rantai asam amino pada rantai DNA.



Gambar 5. Ilustrasi DNA

Sumber: www.ashhg.org

V. KESIMPULAN

String matching merupakan salah satu permaslahan yang ada di dunia ilmu komputer. Masalah ini menggambarkan berbagai macam masalah yang ada di dunia. Dengan ditemukannya solusi atas permaslahaan ini, ditemukan pula manfaat-manfaat terbaru terkait hal tersebut. Tidak hanya dibidang ilmu komputer saja, namun juga menuju ranah bidang lain seperti biologi dan lain-lain.

Seiring berjalannya waktu, bermuncullah berbagai macam algoritma untuk menyelesaikan permaslahan ini. Salah satunya adalah algoritma boyer moore dan knuth morris pratt. Kedua algoritma ini telah memiliki efisiensi yang tinggi, yaitu dengan kompleksitas waktu sebesar O(n+m). Namun kedua algoritma

ini memiliki kelebihan pada kondisi masing-masing. Algoritma boyer moore lebih efisien digunakan pada string matching dengan pattern yang panjang. Sedangkan algoritma knuth morris pratt lebih efisien digunakan pad astring matching dengan pattern yang pendek.

VI. Ucapan Terima Kasih

Puji syukur penulis panjatkan kepada kehadirat Allah SWT, Tuhan yang Maha Esa, karena berkat rahmat, hidayah serta karunia-Nya lah pernulis dapat menyelesaikan makalah ini tepat pada waktunya

Tak lupa juga, penulis mengucapkan terima kasih sebesar-besarnya kepada seluruh pihak yang telah turut membantu dalam penulisan dan penyelesaian makalah ini. Terima kasih banyak kepada Bapak Dr. Masayu Leylia Khodra, S.T., M. T. dan dosen pengajar mata kuliah IF2211 Strategi Algoritma, atas bimbingan yang telah beliau berikan kepada penulis. Terima kasih juga kepada seluruh tim pengajar mata kuliah IF2211 Strategi Algoritma, dan yang terakhir terima kasih juga kepada rekan-rekan yang selalu memberi semangat dan membantu penulis dalam penyelesaian makalah ini.

References

[1] Importance of String Matching in Real World Problems, Dr Amit Sinhal. diakses pada 23 Mei 2022

[2] Visual Approach of Searching Process using Boyer-Moore Algorithm, Robbi Rahim, dkk. Diakses pada 23 Mei 2022.

[3] Knuth-Morris-Prat Algorithm: An Analysis. Mireille Reigner. Diaskes pada 23 Mei 2022

[4] Data Structures Tutorials - Knuth-Morris-Pratt Algorithm (btechsmartclass.com), diakses pada tanggal 23 Mei 2022.

[5] Algoritma Brute Force (Bagian 1) – Rinaldi Munir. Diakses pada tanggal 23 Mei 2022

[6] Pencocokan string (String matching/pattern matching). Diakses pada tanggal 23 Mei 2022.

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, 23 Mei 2022

Muhammad Akmal Arifin - 13520037