Tugas Besar III IF2211 Strategi Algoritma

Pencarian Berita pada *News Aggregator* dengan Algoritma Pencocokan *String*

Disusun oleh:

Rizky Faramita 13515055

Dery Rahman Ahaddienata 13515097

Turfa Auliarachman 13515133

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

April 2017

BAB I
DESKRIPSI PERSOALAN

Sistem agregasi berita (*news aggregator*) dikembangkan untuk membantu pembaca berita dalam mengumpulkan informasi berita dari berbagai sumber dan menyajikannya pada satu portal. Dengan sistem ini, pembaca tidak perlu mencari berita sendiri karena aplikasi dapat mengambil berita sesuai kebutuhan dari pembaca (Lasica, 2003). Pencarian berita merupakan salah satu fitur yang terdapat pada sistem ini.

Pada Tugas Besar III ini, dibutuhkan sebuah aplikasi sederhana berbasis *web* yang dikembangkan dengan kakas PHP, JSP, atau ASP. Aplikasi dapat melakukan pencarian berita pada *news aggregator* dengan memanfaatkan algoritma KMP, Boyer-Moore, serta *regular expression* (*regex*) dengan menggunakan bahasa C#. Teks yang akan diproses adalah kumpulan berita berbahasa Indonesia dan Inggris yang setidak-setidaknya berjumlah 20 butir. Pengguna aplikasi akan memberikan masukan berupa kata kunci pencarian. Kemudian, aplikasi akan menampilkan daftar berita yang diurutkan berdasarkan tanggal berita.

Pada implementasinya, pencocokan *string* dilakukan tidak hanya pada judul berita, tetapi juga isi di dalam berita tersebut. Pencocokan *string* juga merupakan *exact matching* untuk algoritma KMP dan Boyer-Moore, sedangkan tidak selalu *exact matching* untuk *regex*. Pencarian juga tidak bersifat *case sensitive* sehingga huruf besar dan huruf kecil dianggap sama. Hal tersebut dapat dilakukan dengan mengganggap seluruh karakter di dalam *pattern* dan teks sebagai huruf kecil semua atau huruf kapital semua).

Pengambilan kumpulan berita dilakukan secara otomatis menggunakan *crawler* berbasis *RSS* (*rich site summary* atau *really simple syndication*) dari situs berita daring berbahasa Indonesia. Saat membaca *RSS* dengan *XML parser*, informasi yang dibutuhkan berupa judul, tanggal berita, dan *URL* berita. Berikut daftar *RSS* yang dapat digunakan http://rss.detik.com/index.php/detikcom, http://tempo.co/rss/terkini, http://rss.vivanews.com/get/all, dan http://www.antaranews.com/rss/terkini.

Untuk setiap *URL*, aplikasi akan mengunduh artikel kemudian melakukan *parsing*. Artikel berupa file HTML dan tidak hanya mengandung konten berita, tetapi juga masih mengandung *header*, *footer*, iklan, dan tambahan informasi pada situs berita tersebut. Untuk itu, dilakukan *parsing* HTML untuk mendapatkan hanya teks konten berita dan foto yang terkait berita tersebut. Salah satu contoh library *html parser* yang dapat digunakan adalah https://jsoup.org/ untuk Java, *RSS parser* untuk .NET dan *XML parser* dalam Bahasa C#.

BAB II
DASAR TEORI

Pencarian *String* adalah proses pencarian untuk menemukan lokasi pada teks di mana *string* pertama kali ditemukan. *String* itu sendiri merupakan sebuah pola bertipe *string* dengan panjang m karakter yang hendak dicari di dalam teks, sedangkan teks adalah kumpulan kalimat yang dalam tugas ini adalah artikel dengan panjang n karakter di mana n selalu lebih besar dibandingkan m. Apabila tidak memenuhi asumsi m < n, maka secara otomatis dapat ditarik kesimpulan bahwa tidak terdapat *string* pada teks. Aplikasi pencarian *string* ini sangat bervariatif, mulai dari pencarian di dalam teks editor, *web search engine,* analisis citra, hingga *bioinformatics.*

Pada tugas besar tiga ini, digunakan algoritma Knuth-Morris-Pratt(KMP), Boyer Moore, dan regex untuk menyelesaikan masalah berdasarkan deskripsi persoalan. Algoritma KMP ditemukan oleh Donald Ervin Knuth, seorang professor di Universitas Stanford. Ide dari algoritma ini adalah mencari pola di dalam teks mulai dari indeks paling kiri hingga paling kanan dengan *shifting* yang lebih efektif daripada algoritma *brute force* (shifting selalu satu indeks ke kanan) dengan memanfaatkan adanya fungsi pinggiran. Fungsi pinggiran pada algoritma KMP berguna untuk menentukan seberapa jauh *shifting* perlu dilakukan ketika *mismatch* antara teks dengan *string* terjadi. Fungsi pinggiran b(k) diartikan sebagai ukuran terbesar dari *prefix* P[1..k] yang juga merupakan *suffix* dari P[1..k] dengan k adalah 1..(string.length()).

Kompleksitas waktu untuk algoritma KMP adalah O(m+n) dengan O(m) untuk menghitung fungsi pinggiran dan O(n) untuk melakukan pencarian *string.* Algoritma ini baik digunakan pada pencarian *string* pada teks dengan ukuran teks yang sangat besar. Tetapi, sayangnya algoritma ini menjadi kurang baik penggunaannya apabila variasi dari alfabet pada teks bertambah sehingga lebih sering terjadi proses pengecekan *mismatch*. Algoritma ini tidak dapat melakukan *trackback* atau pun pemrosesan terhadap alfabet yang menyebabkan *mismacth* pada saat proses pencarian berlangsung. Oleh karena itu, hadirlah algoritma Boyer Moore untuk mengatasi pencarian dengan tingkat variasi *string* dan teks yang tinggi.

Pada algoritma Boyer Moore, digunakan teknik *looking-glass* yaitu proses pencarian *string* secara *backward* sehingga pointer pada *string* akan selalu menunjuk indeks paling kanan lalu bergerak ke kiri hingga ditemukan *dismatch* atau tidak ditemukan *dismatch* yang berarti solusi telah ditemukan. Selain teknik *looking-glass,* terdapat pula teknik *character-jump* yang digunakan untuk menentukan *shifting* pada teks ketika ditemukan *dismatch.* Sebelum proses

pencarian dimulai, akan disimpan kemunculan setiap karakter di indeks terbesarnya pada *string* pada sebuah fungsi L(). Kemudian, proses pencarian dimulai dan setiap kali *dismatch* muncul maka akan dicek karakter apa yang terdapat pada teks, lalu mengkategorikan letak karakter dengan memanfaatkan informasi *current pointer* pada *string* dan fungsi L().

Apabila kemunculan karakter yang *dismatch* ada di sebelah kiri *current pointer,* maka teks di geser untuk disesuaikan dengan karakter yang sama. Apabila kemunculan ada di sebalah kanan *current pointer,* maka teks hanya akan digeser satu ke kanan. Apabila karakter pada teks tidak terdapat pada *string* maka indeks pertama pada *string* akan digeser ke *current pointer* pada teks ditambah satu. Kompleksitas algoritma Boyer Moore adalah O(nm + A) . Algoritma ini bekerja dengan baik contohnya pada teks bahasa Indonesia tetapi buruk untuk pembacaan binari atau berkas lainnya dengan variasi alfabet (A) yang kecil. Selain algoritma KMP dan Boyer Moore terdapat pula Regex yang merupakan salah satu alternatif teknik lain untuk mencari *string* pada suatu teks.

BAB III
ANALISIS PEMECAHAN MASALAH

Berikut ini merupakan langkah-langkah yang dilakukan oleh program untuk dapat menyajikan berita sesuai dengan kata kunci yang dimasukkan, serta metode pencarian *string* dan platform berita yang dipilih:

3.1 Membaca file eksternal yang berisi kumpulan berita yang sudah didapatkan sebelumnya dan mengkonversinya menjadi berbentuk *List of News*

3..2 Membuat *List of News* yang *Content*-nya masih kosong dengan input URL sebuah RSS.

3.3 Pengguna dapat memilih satu dari ketiga buah metode yang disediakan oleh aplikasi yaitu algoritma Boyer Moore, KMP, atau Regex;

3.4 Pengguna dapat memilih salah satu dari beberapa platform yang disediakan oleh aplikasi (Viva, Kompas, atau Detik) atau memilih untuk mencari di semua platform berita (All)

3.5 Kemudian pengguna dapat memasukkan kata kunci yang relevan untuk mendapatkan artikel yang hendak dicari

3.6 Program akan memproses ketiga buah masukan pengguna tersebut

3.7 Program akan menampilkan berapa banyak artikel yang sesuai dengan ketiga masukan tersebut

3.8 Dengan memanfaatkan fungsi Match, StringToMatch akan didapatkan berita yang sesuai

3.9 Untuk mendapatkan artikel, program akan mengambil konten dan melakukan *scraping*

3.10 Kemudian program akan mengambil TitleAndUrl dan ImageUrl sekaligus memastikan bahwa tidak ada duplikasi data sehingga artikel yang sama hanya akan ditampilkan sekali

3.11 Jika pencarian menghasilkan artikel tertentu, maka hasil pencariannya disajikan dalam bentuk judul yang merupakan *link* bagi pengguna untuk dapat mengakses artikel, waktu penulisan artikel, kalimat yang relevan dengan kata kunci, serta gambar yang terletak di sebelah kiri informasi berita.

BAB IV
IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

4.1.1 Struktur Data

Untuk menyelesaikan permasalahan pada tugas ini, kami merancang empat struktur data, yaitu *News*, *ReturnObject, SearchQuery,* dan *SearchResult*. Di bawah ini akan dijelaskan secara detail keempat struktur data tersebut.

Struktur data News digunakan untuk menyimpan informasi tentang setiap berita yang ada. Struktur data ini memiliki properti Url, Title, Content, ImageUrl, dan PubDate.

Struktur data ReturnObject digunakan sebagai format output dari setiap request yang masuk ke API program. Struktur data ini memiliki dua properti, yaitu status yang bertipe boolean dan data yang bertipe Object.

Struktur data SearchQuery digunakan untuk menangkap query pencarian dari user. Struktur data ini memiliki properti Id yang melambangkan algoritma pencarian, Source yang melambangkan sumber berita yang ingin dicari, dan Pattern yang berisi pattern yang ingin dicari dalam berita.

Struktur data yang terakhir adalah SearchResult. Struktur data ini digunakan untuk menyimpan hasil pencarian. Pada dasarnya, struktur data ini mirip dengan struktur data *News*. Pada *SearchResult, Content* diubah menjadi *Match*, karena tidak semua konten berita akan ditampilkan. Hanya potongan berita yang memuat *pattern* saja yang disimpan pada struktur data *SearchResult*.

4.1.2 Fungsi dan Prosedur

Berikut beberapa fungsi dan prosedur yang kami buat untuk mengerjakan tugas ini.

4.1.2.1 *PrepareNewsList* dan *News.GetNewsList*

Fungsi *News.GetNewsList* membaca file eksternal yang berisi kumpulan berita yang sudah didapatkan sebelumnya dan mengkonversinya menjadi berbentuk *List of News*. Prosedur *PrepareNewsList* mengatur file mana saja yang akan dibaca dan menggabungkan *List of News* menjadi satu ketika *user* melakukan pencarian dari seluruh sumber berita.

4.1.2.2 *KmpSearcher.SetPattern, BoyerMooreSearcher.SetPattern, RegexSearcher.SetPattern*

Ketiga prosedur ini berfungsi untuk menyimpan *pattern* pencarian yang akan digunakan. Dalam *KmpSearcher.SetPattern* dan *BoyerMooreSearcher.SetPattern* dilakukan *preprocessing* sesuai algoritmanya.

4.1.2.3 *KmpSearcher.Match, BoyerMooreSearcher.Match,* dan *RegexSearcher.Match*

Ketiga prosedur ini berfungsi untuk mengecek apakah suatu teks *match* dengan *pattern* yang sudah diberikan atau tidak. Jika ya, akan diberitahu pada indeks ke-berapa *pattern* muncul pertama kali dalam teks.

4.1.2.4 *SearchResult.StringToMatch*

Fungsi *SearchResult.StringToMatch* akan memotong *string* dengan panjang yang sudah ditentukan. Jika di awal atau di akhir *string* ada yang terpotong, akan ditambah "..." di awal atau di akhir *string*. Fungsi ini digunakan untuk menjadikan sebuah *News.Content* menjadi *SearchResult.Match*.

4.1.2.5 *RSSScraper.GetTitleAndUrl* dan *RSSScraper.GetImageUrl*

Kedua fungsi ini akan membuat *List of News* yang *Content*-nya masih kosong dengan input URL sebuah RSS.

4.1.2.6 *DetikScraper.GetContent, TempoScraper.GetContent,* dan *VivaScraper.GetContent*

Fungsi-fungsi ini akan mengembalikan konten berita dengan input URL berita. Konten berita ini akan melengkapi fungsi 4.1.2.5.

4.1.2.7 *DetikScraper.Scrape, TempoScraper.Scrape,* dan *VivaScraper.Scrape*

Fungsi ini akan mengembalikan daftar berita dari masing-masing situs dalam bentuk *List of News* dengan memanfaatkan fungsi 4.1.2.5 dan 4.1.2.6.

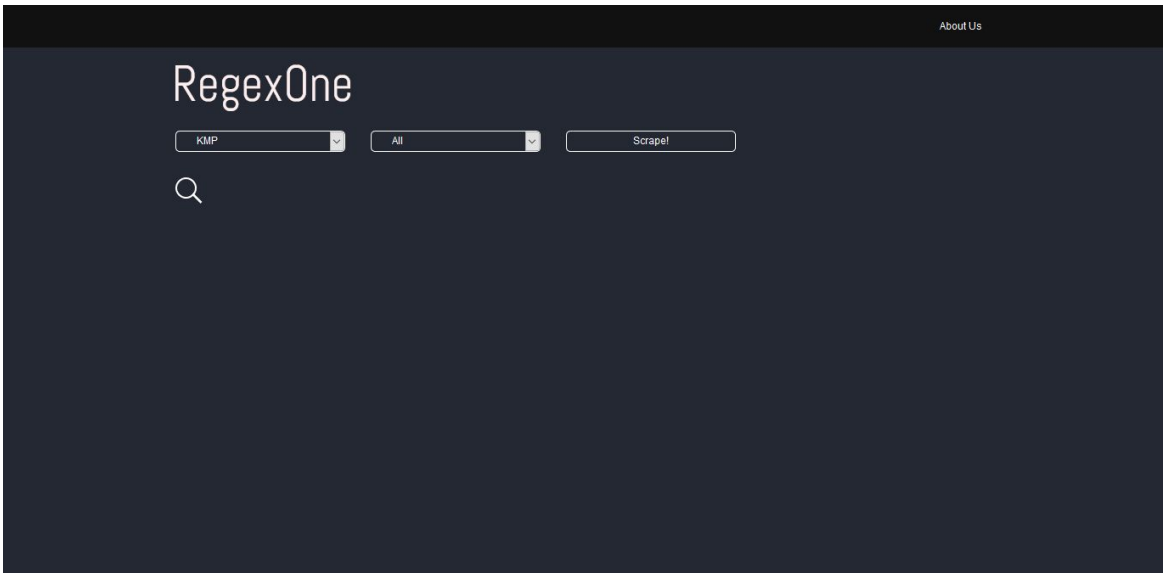
4.1.2.8 *CleanContent*

Prosedur ini menghapus spasi di awal dan akhir *News.Content*. Fungsi ini juga mengubah spasi berlebih menjadi hanya satu spasi saja.

4.1.2.9 *Combine*

Prosedur ini menggabungkan dua *List of News* dan memastikan bahwa tidak ada duplikasi data.

4.1.3 Antarmuka

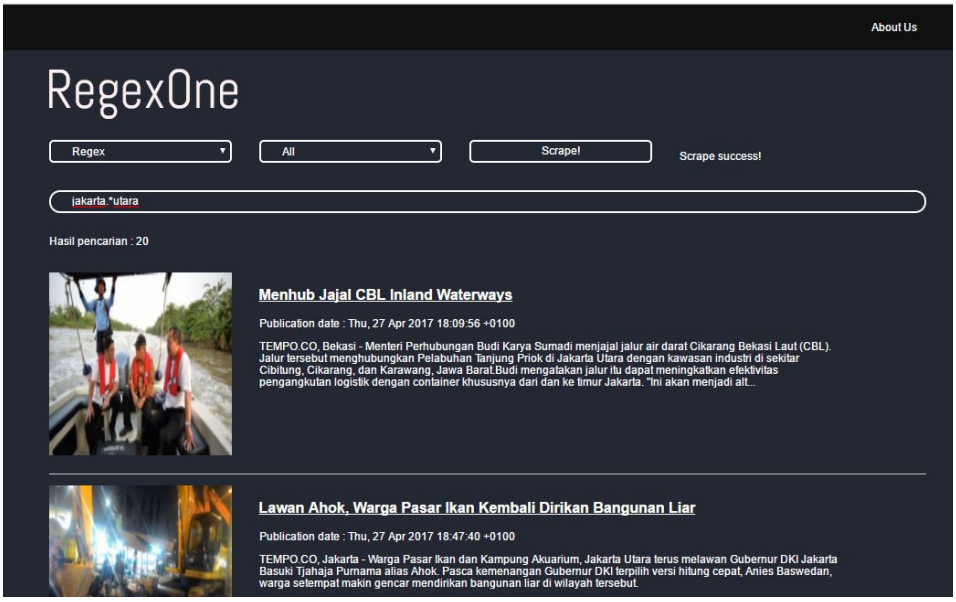


Gambar 1. *Tampilan Awal Aplikasi Pencarian Berita untuk Mode Web*

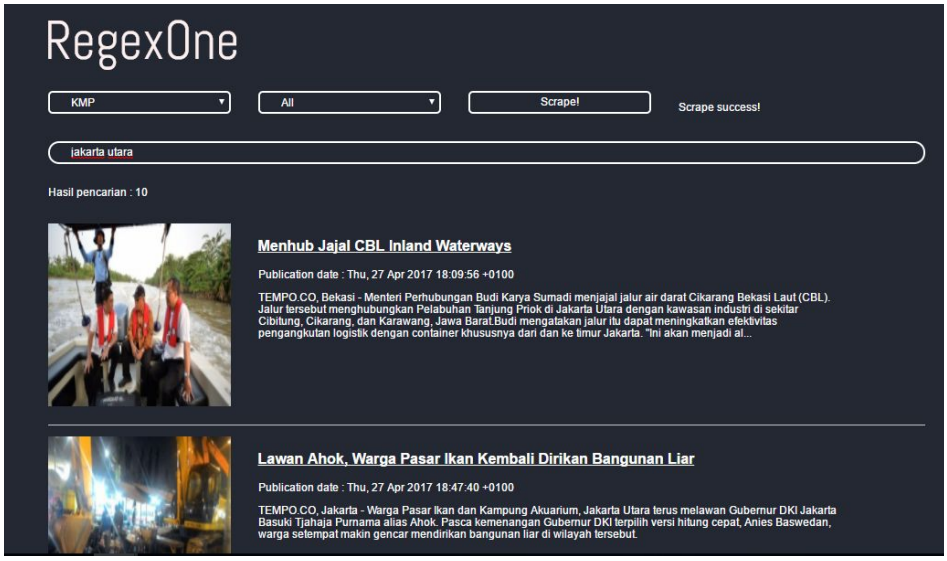
Gambar 2. *Tampilan Awal Aplikasi Pencarian Berita untuk Mode Mobile*

4.2 Pengujian

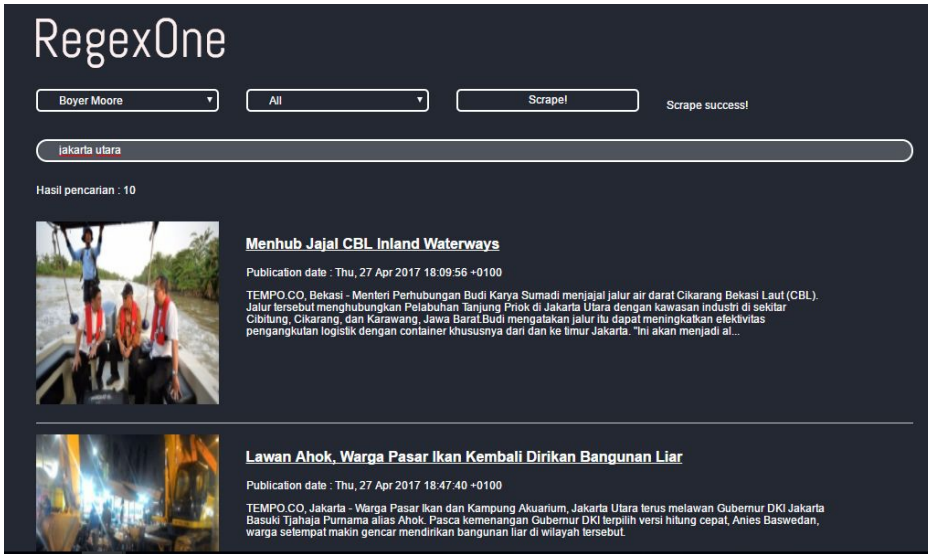
4.2.1 Pengujian Berdasarkan Metode Pencarian *String*



Gambar 3. *Hasil Pencarian Artikel dengan Regex*

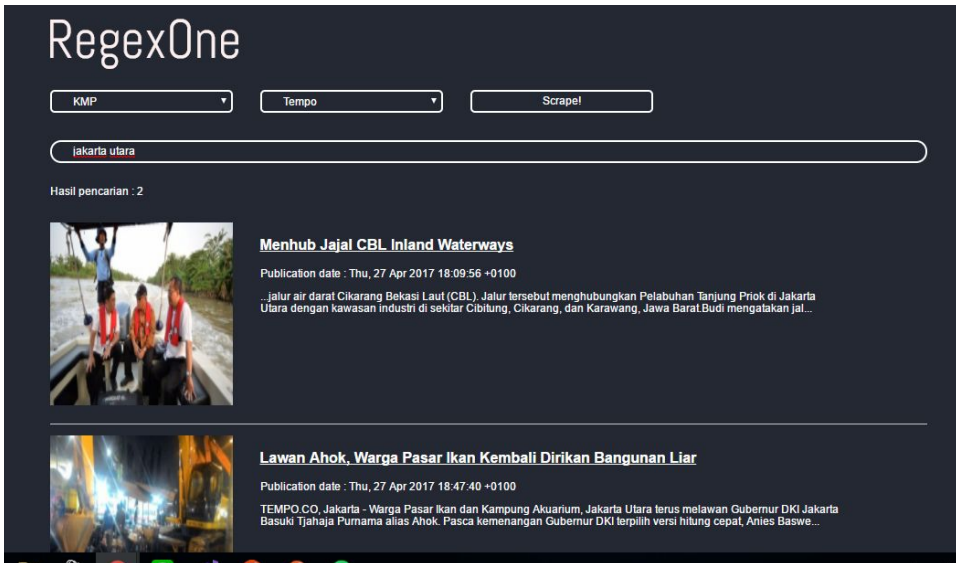


Gambar 4. *Hasil Pencarian Artikel dengan Algoritma KMP*

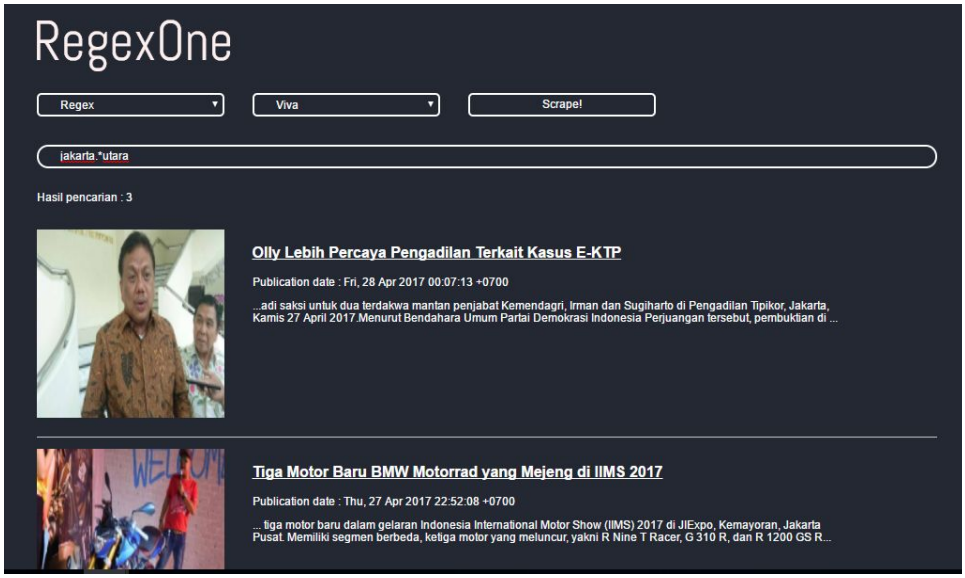


Gambar 5. *Hasil Pencarian Artikel dengan Algoritma Boyer Moore*

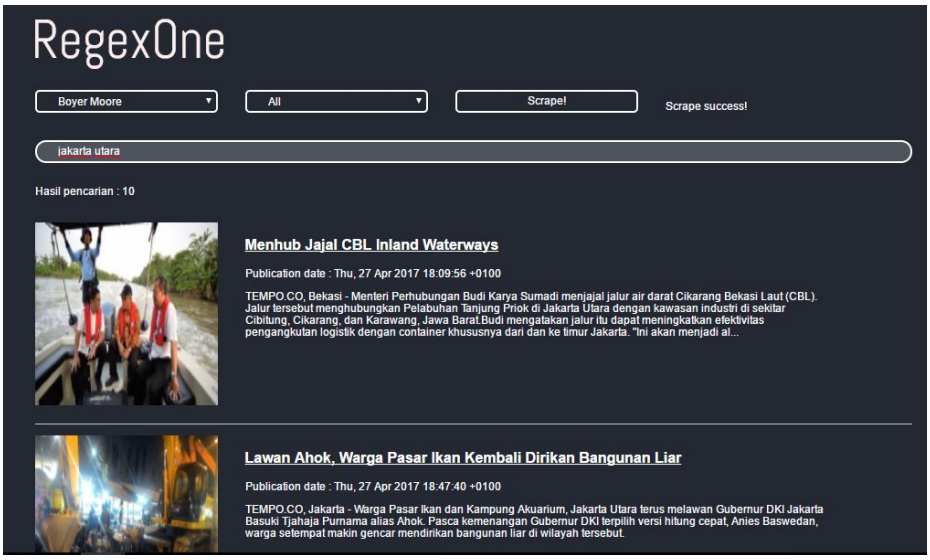
4.2.2 Pengujian Berdasarkan Masukan Berita Pilihan



Gambar 6. *Hasil Pencarian Artikel di Tempo*



Gambar 7. *Hasil Pencarian Artikel di Viva*



Gambar 8. *Hasil Pencarian Artikel di Semua Platform Berita*

4.3 Analisis Hasil Pengujian

4.3.1 Analisis Hasil Pengujian Berdasarkan Metode Pencarian *String*

Dipilih sebuah kata kunci jakarta utara yang dicari dengan menggunakan tiga metode pencarian *string* yang berbeda, yaitu Regex, KMP, dam Booyer Moore. Hasil pencarian dengan menggunakan kata kunci yang sama untuk ketiga metode yang berbeda dapat menampilkan kumpulan artikel yang sama yaitu artikel mengenai Menhub, Ahok, dan masih terdapat delapan buah artikel lainnya. Hal tersebut membuktikan bahwa meskipun metode yang digunakan berbeda, namun apabila implementasi algoritma telah dilakukan dengan benar maka akan memberikan hasil yang sama.

4.3.2 Analisis Pengujian Berdasarkan Masukan Berita Pilihan

Hasil pencarian dengan menggunakan berbagai media berita (Viva, Tempo, Detik, atau All) dapat diimplementasikan dan menghasilkan artikel sesuai dengan kata kunci yang dimasukkan. Pada gambar terlihat adanya perbedaan apabila artikel dicari pada platform berita yang berbeda, yaitu mencari artikel dengan kata kunci jakarta utara di Viva akan menghasilkan artikel yang berbeda dengan artikel di Tempo.

Bab V
Kesimpulan dan Saran

News Aggregator merupakan aplikasi web sederhana yang bertujuan untuk memudahkan pengguna membaca berbagai konten berita dalam satu tempat. Umumnya, *news aggregator* ini mempunyai fitur *search* di dalamnya. Fitur *search* yang kami kembangkan menggunakan bahasa C# dengan menerapkan berbagai algoritma *string matching.* Diantaranya adalah Knuth-Morris-Pratt (KMP) dan Boyer Moore. Ditambah pula regex sebagai teknik tambahan yang dapat diaplikasikan untuk melakukan pencarian suatu *string* di dalam kumpulan berita. Berita-berita tersebut dicari dengan metode pencarian *string* yang sesuai dengan masukan pengguna pada platform pilihan pengguna pula.

Sayangnya, tidak semua platform berita dapat dijadikan acuan untuk mencari artikel. Harus dipastikan terlebih dahulu bahwa platform berita yang hendak digunakan dapat diakses secara normal. Contohnya, pencarian berita di platform berita Detik akan menemukan kegagalan karena terjadi kesalahan teknis pada website di mana Detik meletakkan kumpulan artikel. Selain itu, terdapat pula beberapa artikel dengan gambar yang tidak dapat ditampilkan. Hal tersebut dikarenakan *url* menuju gambar untuk artikel tersebut *broken*. Lebih baik, ketika didapatkan sebuah *broken link,* fungsi GetImageUrl dapat mengidentifikasi tersebut sehingga aplikasi hanya akan menghasilkan artikel yang memiliki gambar untuk ditampilkan.

DAFTAR PUSTAKA

[1] Davison, Dr. Andrew. 240-301, Computer Engineering Lab III (Software), 2006-2007.