Konversi Alfabet Latin ke Aksara Sunda Kaganga

Hendro Triokta Brianto / 13512081
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13512081@std.stei.itb.ac.id

***Abstrak*—Aksara Sunda adalah salah satu dari berbagai macam aksara yang ada di Indonesia. Aksara yang cukup sering dipakai di Jawa Barat yaitu aksara Sunda Kaganga Ngalagena. Dalam kenyataannya, aksara Sunda sudah sangat jarang digunakan banyak orang. Banyak sekolahan umum yang tidak mengajarkan bahasa tersebut ke dalam kurikulumnya. Untuk mengubah alfabet latin ke dalam aksara Sunda dapat digunakan metode pencocokan *string*. Mengubah alfabet ke dalam aksara cukup sulit karena harus mengetahui beberapa aturan. Namun, dengan bantuan algoritma *brute force* hal itu cukup teratasi. Karena dibaca satu per satu hingga membentuk satu aksara.**

***Kata Kunci*—Aksara, Alfabet, Kaganga, pencocokan *string*.**

I. PENDAHULUAN

Di zaman yang modern ini, masyarakat lebih sering memakai alfabet latin sebagai cara mereka berkomunikasi. Mulai dari pesan sampai tulisan berupa buku. Hal ini dikarenakan pemakaian alfabet latin mudah dipahami dan mudah dibaca. Selain itu, setiap bahasa di dunia mudah diterjemahkan dengan mudah ke dalam alfabet latin.

Di Indonesia sendiri memiliki banyak daerah atau bahasa yang memiliki aksara. Sallah satunya di Bumi Pasundan atau Jawa Barat ini. Aksara Sunda saat ini, tidak terlalu dilirik oleh orang kebanyakan. Bahkan untuk masyarakat Jawa Barat sendiri sudah jarang yang menggunakannya. Hanya beberapa daerah yang masih memakainnya. Sekolah – sekolah umum pun banyak yang tidak mengajarkan aksara Sunda ini ke siswanya. Sehingga banyak pelajar yang tidak mengetahui bagaimana aksara Sunda tersebut.

Beruntung, penulis masuk ke sekolah yang tepat. Karena pada saat itu, penulis dikenalkan dengan aksara Sunda. Sebelumnya, penulis tidak mengetahui sama sekali mengenai aksara Sunda tersebut. Aksara Sunda yang penulis pelajari ialah Aksara Sunda Kaganga Ngalagena. Hal pertama yang terlintas saat mempelajari ini ialah seperti membaca tulisan Kanji dari Jepang atau tulisan *Chinese*. Namun, tidak terlalu sulit untuk memahaminya saja. Butuh latihan yang cukup agar dapat hapal seluruh aksaranya.

Oleh karena itu, penulis mengambil tema Konversi Alfabet Latin ke Aksara Sunda Kaganga sebagai cara untuk memperkenalkan aksara Sunda ke masyarakat luas dan berharap agar masyarakat Sunda khususnya dapat mengerti dan berusaha melestarikannya.

II. TEORI DASAR

A. Pencocokan String

Pencocokan *String* adalah suatu algortima untuk mencocokan dua buah *string* dimana *string* kecil sebagai *pattern* dan *string* yang lebih panjang sebagai teks. Algoritma ini digunakan untuk pencarian kata pada sebuah teks kecil atau teks panjang.

Banyak algoritma yang digunakan dalam pencocokan *string*. Di antaranya, *Brute Force,* Knuth-Morris-Pratt (KMP), Boyer-Moore, dan lainnya.

Algoritma *Brute Force*

Algoritma *Brute Force* adalah salah satu algoritma yang bisa dibilang pasti ketemu hasilnya. Dalam dunia komputer, algoritma ini sering disebut algoritma yang tidak cerdas. Karena tidak memerlukan atau tidak mementingkan tingkat kompleksitas.

Algoritma *brute force* memecahkan masalah dengan sederhana, langsung, dan dengan cara yang jelas. Algoritma ini lebih sering digunakan untuk contoh – contoh yang terbilang kecil dan simpel. Algoritma ini juga lebih mudah diimplementasikan daripada algoritma yang lebih canggih. Algoritma ini cocok dan cepat untuk permasalahan yang kecil. Namun, dapat berakibat sangat lama pada persoalan yang cukup besar.

Dalam pencocokan *string*, algoritma ini sangat jarang dipakai. Terkadang digunakan untuk pengujian.

Langkah – langkah yang dilakukan *brute force* pada

pencocokan *string*.[4]

1. Awal *pattern* dicocokan dengan awal teks.
2. Pencocokan dilakukan dengan cara bergerak dari kiri ke kanan. Bandingkan setiap karakter yang ada di *pattern* pada teks sampai memenuhi kondisi : (a) *Pattern* yang dicari ketemu (b) Dijumpai karakter yang tidak cocok sehingga pencarian berhenti.
3. Jika ketemu karakter yang tidak cocok dan teks belum habis *pattern* bergeser satu karakter.

contoh :

```
Teks:    nobody noticed him
Pattern: not
   nobody noticed him
s=0 not
s=1  not
s=2   not
s=3    not
s=4     not
s=5      not
s=6       not
s=7        not
```

Pada algoritma *brute force* terdapat dua kasus. Kasus terbaik dan kasus terburuk. Kasus terbaik pada algoritma ini ialah ketika karakter pertama *pattern* tidak pernah sama dengan karakter teks yang dicocokkan. Jumlah perbandingan yang dilakukan paling banyak n kali, misalnya :

```
Teks: Ini adalah string panjang berakhir dengan zz
Pattern: zz
```

Kasus terburuk membutuhkan *m(n-m+1)* perbandingan dengan kompleksitasnya adalah *O(nm)*, misalnya :

```
Teks:    aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaab
Pattern: aaab
```

Algoritma *brute force* juga memiliki kelebihan dan kekurangan.[2]

Kelebihan:

a. Dapat digunakan untuk memecahkan hampir sebagian besar masahal (*wide applicability*);
b. Mudah dimengerti;
c. Mengahasilkan algoritma yang layak untuk beberapa masalah penting seperti pencarian, pengurutan, pencocokan *string*, perkalian matriks;
d. Menghasilkan algoritma baku untuk tugas – tugas komputasi, seperti penjumlahan/perkalian *n* buah bilangan, menentukan elemen minimum atau maksimum di dalam tabel.

Kekurangan:

a. Jarang menghasilkan algoritma yang efisien;
b. Beberapa algoritma lambat sehingga tidak dapat diterima;
c. Tidak sekontruktif/sekreatif teknik pemecahan masalah lainnya.

Algoritma Knuth-Morris-Pratt (KMP)

Algoritma KMP mirip dengan algoritma *brute force*. Hanya saja pada KMP disimpan informasi mengenai pergeseran sebelumnya. Dengan algoritma ini, waktu pencarian dapat dikurangi secara signifikan. Algoritma ini dikembangkan oleh Donald E. Knuth, bersama dengan James H. Morris dan Vaughan R. Pratt.

Berikut contoh dari algoritma KMP:

```
          1 2 3 4 5 6 7 8 9...
Teks:     bimbingan belajar atau bimbel
Pattern: bimbel
            ↑
            j = 5
```

```
          1 2 3 4 5 6 7 8 9...
Teks:     bimbingan belajar atau bimbel
Pattern:      bimbel
              ↑
              j = 2
```

Dari contoh di atas terlihat ketika *j=5* tidak cocok, pencocokan dilanjutkan tidak bergeser satu karakter, tetapi melanjutkan ke karakter ke-*x*.

Pada algortima KMP ada yang namanya fungsi pinggiran. Fungsi pinggiran didefinisikan sebagai ukuran awalan terpanjang dari *pattern* yang merupakan akhiran *pattern*.

Algoritma Boyer-Moore

Algoritma Boyer-Moore dianggap sebagai algoritma yang efisien pada aplikasi umum. Algoritma yang dipublikasikan oleh Robert S. Boyer dan J. Strother Moore ini mirip dengan algoritma sebelumnya, hanya saja mulai pencocokan dari kanan maka akan lebih banyak informasi yang didapat.[3]

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

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

B. Aksara Latin/Alfabet

Huruf alfabet adalah huruf yang digunakan oleh masyarakat dunia saat ini untuk menuliskan bahasa. Huruf – huruf dari alfabet sebenernya adalah tanda – tanda yang bersuara.[1] Pada sejarahnya manusia memakai gambar atau simbol untuk menuliskan suatu pesan atau juga untuk menggambarkan kejadian yang terjadi pada saat itu. Pada saat itu gambar atau simbol menjadi salah satu alat komunikasi antar manusia.

Istilah *alphabet* sebetulnya berasal dari bahasa Semit. Istilah ini terdiri dari dua kata, yaitu *aleph* yang berarti 'lembu jantan' dan kata *beth* yang berarti 'rumah'. Konotasi pictografis dari pengertian kedua kata ini menjadi sebutan untuk menunjukkan huruf pertama a (*aleph*) dan b (*beth*) dalam urutan huruf-huruf semit (Mario Pei,1971:176).

Gambar 1 Alfabet latin yang kita kenal[2]

Beberapa negara mengadopsi dan memodifikasi alfabet Latin sesuai dengan tata bahasa mereka, karena tidak semuanya dapat dilambangkan dengan huruf latin. Salah satu caranya ialah dengan menambahkan huruf baru (contoh : J, W).

Beberapa negara mengatur penggunaan dwihuruf dalam bahasa resmi mereka, yang melambangkan suatu fonem yang tidak dapat dilambangkan oleh alfabet Latin, misalnya "Th" (untuk bunyi /θ/dan /ð/), "Ng" atau "Nk" (untuk bunyi /ŋ/), "Sch" atau "Sh" (untuk bunyi /ʃ/), "Ph" (untuk bunyi /ɸ/ dan /f/).

Dalam Alfabet Indonesia, dikenal 26 huruf alfabet. Bentuk tulisan dapat berbeda satu sama lain khususnya jika ditulis dalam huruf sambung. Bahasa Indonesia menggunakan beberapa dwihuruf (ng, ny, kh, dan sy) untuk menulisakn lambang bunyi yang tidak tersedia dalam alfabet Latin dasar.

Tabel 1.Nama Huruf

Huruf	Sebutan	IPA
Aa	*A*	/ɑː/
Bb	*Be*	/b/
Cc	*Ce*	/c/, /tʃ/ atau /tɕ/
Dd	*De*	/d/
Ee	*E*	/e, ɛ/ atau /ə/
Ff	*ef*	/f/
Gg	*ge*	/ɡ/
Hh	*ha*	/h/
Ii	*i*	/i/
Jj	*je*	/ʤ/ atau /ʣ/
Kk	*ka*	/k/
Ll	*el*	/l/
Mm	*em*	/m/
Nn	*en*	/n/
Oo	*o*	/ɔ, o/
Pp	*pe*	/p/
Qq	*ki*	/q/
Rr	*er*	/r/
Ss	*es*	/s/
Tt	*te*	/t/
Uu	*u*	/u/
Vv	*ve*	/v, ʋ/ atau /f/
Ww	*we*	/w/
Xx	*eks*	/ks/
Yy	*ye*	/j/
Zz	*zet*	/z/

C. Aksara Sunda Kaganga

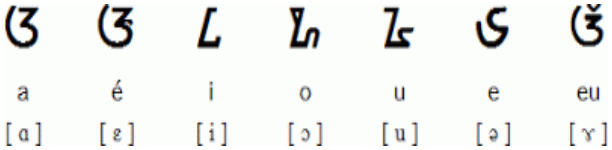
Aksara Sunda Kaganga Atau huruf Sunda Buhun adalah huruf masyarakat Sunda yang bersejarah. Penggunaan Aksara Sunda Kaganga telah ada dari abad ke-8 masehi. Huruf sunda kaganga ini tertulis dalam berbagai bukti-bukti sejarah rakyat sunda seperti peninggalan sejarah berupa piagam dan prasasti-prasasti seperti Batutulis, Sanghayang Tapak, Kawali, Kebantenan, dan Geger Hanjuang. Dan ditemukan juga di berbagai naskah-naskah kuno di sejumlah wilayah tataran sunda.

Huruf Sunda Kaganga ini sekarang telah distandarisasikan atas prakarsa Disdik Jawa Barat bekerja sama dengan UPI dan Unpad, pada tanggal 20 Oktober 2008 di Aula Unpad. Sehingga muncullah unicode aksara Sunda Kaganga.

Namun karena mungkin sosialisasi yang kurang gereget dari pemerintah (seperti yang Kompas katakan "Kurang Gencar, Sosialisasi Aksara Kaganga"), huruf Sunda Kaganga masih belum diketahui mayoritas urang Sunda.

Huruf Sunda Kaganga berjumlah 32 huruf, terdiri dari, 7 aksara swara 'vokal mandiri' dan 25 aksara ngalagena (konsonan).[1]

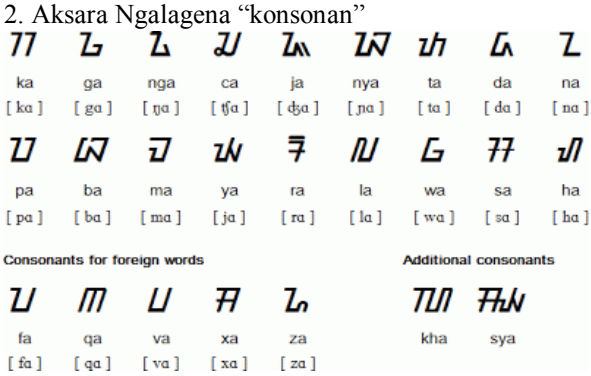
1. Aksara Swara "vokal mandiri"



Gambar 2 aksara swara

[1] Andre Djamil, 2009, "Penemu Huruf Alfabet".
[2] http://www.anneahira.com/images/sejarah-tulisan.jpg

2. Aksara Ngalagena "konsonan"



Gambar 3 Aksara Ngalagena

Di dalam aksara sunda, ada beberapa aturan lagi yang digunakan dalam penulisan. Aturan tersebut dinamakan "Rarangken" yang terdiri dari tiga kategori, atas, bawah, dan sebaris dengan huruf.

1. Untuk yang berada di atas huruf
   - **panghulu,** digunakan untuk mengubah pengucapan /a/ jadi /i/;
   - **pamepet,** digunakan untuk mengubah pengucapan /a/ jadi e;
   - **paneuleung,** digunakan untuk mengubah pengucapan /a/ jadi /eu/;
   - **panglayar,** digunakan untuk menambah huruf /+r/ di akhir kata;
   - **panyecek,** digunakan untuk menambah suaru /+ng/ di akhir kata.
2. Untuk yang berada di bawah huruf
   - **panyuku,** digunakan untuk mengubah pengucapan /a/ jadi /u/;
   - **panyakra,** digunakan untuk menyisipkan huruf /+r/ di antara konsonan dan vokal;
   - **panyiku,** digunakan untuk menyisipkan hurul /+l/ di antara konsonan dan vokal.
3. Untuk yang sebaris dengan huruf
   - **panéléng,** digunakan untuk mengubah pengucapan /a/ jadi /é/;
   - **panolong,** digunakan untuk mengubah pengucapan /a/ jadi /o/;
   - **pamingkal,** digunakan untuk menyisipkan huruf /+y/ di antara konsonan dan vokal;
   - **pangwisad,** digunakan untuk menambah huruf /h+/ di akhir kata;
   - **paten atau pamaeh,** digunakan untuk mematikan huruf konsonan (tanpa vokal).

III. IMPLEMENTASI

Berdasarkan penjelasan algoritma mengenai pencocokan *string* di atas, algoritma yang cocok dalam proses ini ialah algoritma *brute force*. Algoritma tersebut sangatlah sederhana, namun membutuhkan kejelian dalam menggunakannya. Karena algoritma *brute force* dapat berakibat sangat lambat jika salah mengaplikasikannya. Mengingat pada tulisan ini hanya mengonversikan alfabet latin ke aksara Sunda Kaganga, penulis hanya menggunakan algoritma *brute force* dalam melakukan prosesnya.

Proses pencarian dari karakter per karakter sangatlah membantu. Aksara Sunda ataupun aksara lainnya memiliki keunikan tersendiri. Untuk aksara Sunda hanya terdapat 32 huruf, 7 untuk vokal dan 25 untuk aksara konsonan.

Berikut adalah beberapa tahapan dalam proses pengonversian alfabet ke aksara:

1. Huruf pertama pada suatu teks akan diperiksa. Ada beberapa kondisi:
   i. Huruf pertama adalah huruf vokal;
   ii. Huruf kedua adalah konsonan;

   Jika kondisi pertama (i) yang terpenuhi, pemeriksaan akan dilanjutkan dengan karakter selanjutnya tanpa melihat karakter pertama. Jika kondisi kedua (ii) yang terpenuhi, pemeriksaan dilanjutkan dengan melihat karakter pertama.

   Contoh (i):

   Teks : aku
   proses : a ku
   ↑
   ᮃ

   Penjelasan : karena ketika huruf awal adalah huruf vokal, sudah dipastikan tidak perlu melihat karakter selanjutnya.

   Contoh (ii):

   Teks : kau
   proses : k a u
   ↑
   ᮏᮢ

   kemungkinan: ka, ku, ke, ki, ko, keu, ké, kar, kang, kra, kla, kya, kah, dan k

   k a u
   ↑
   ᮏ

2. Huruf pertama konsonan dan huruf kedua konsonan.

   Contoh :

   Teks : khitan (kita lihat tiga huruf pertama yaitu

"khi")

Jika kita tidak menyimpan huruf sebelumnya, maka:

k	h	i
ᮊ᮪	ᮠ᮪	ᮄ

Namun, berbeda jika kita menyimpan huruf sebelumnya.

proses 1 : k	ᮊ᮪
proses 2 : kh	ᮊᮠ᮪
proses 3 : khi	ᮊᮢ᮪

Terlihat berbeda jika kita tidak menyimpan huruf sebelumnya.

3. Jumlah huruf maksimal yang akan diperiksa sejumlah tiga sampai empat karakter untuk setiap aksaranya.
   Contoh string 3 sampai empat karakter:
   keu, kar, kang, kra, kla, kya, kah

4. Ketika huruf pertama dan kedua konsonan periksa karakter ke tiga. Hal ini dikarenakan karakter ketiga bisa masuk ke aksara pertama atau aksara yang selanjutnya.
   Contoh:
   Teks: kang

proses 1: k	ᮊ᮪
proses 2 : ka	ᮊ
proses 3 : kan	ᮊᮀ
proses 4 : kang	ᮊᮂ

Penjelasan : Terlihat pada proses tiga terdapat dua aksara yang berbeda, yaitu aksara "ka = ᮊ" dan aksara "n = ᮀ". Namun, pada karakter ke empat yaitu 'g', aksara kembali menjadi satu aksara "ᮊᮂ".

Contoh lain:

Teks: asdfgh	
proses 1 : a	ᮃ
proses 2 : as	ᮃᮊ᮪
proses 3 : asd	ᮃᮊ᮪ᮓ᮪
proses 4 : asdf	ᮃᮊ᮪ᮓ᮪ᮕ᮪
proses 5 : asdfg	ᮃᮊ᮪ᮓ᮪ᮕ᮪ᮌ᮪
proses 6 : asdfgh	ᮃᮊ᮪ᮓ᮪ᮕ᮪ᮌ᮪ᮠ᮪

Penjelasan : hal ini membuktikan hanya beberapa gabungan konsonan tertentu saja yang dapat membentuk satu aksara.

Untuk konsonan awal atau karakter awal pada satu aksara yaitu, k, g, n, c, j, t, d, p, b, m, y, r, l, w, s, h. Untuk karakter kedua yaitu, g, y, h, r, l. Ada beberapa konsonan tambahan untuk kata asing, yaitu fa, qa, va, xa, za.

Berikut beberapa contoh kalimat dari alfabet latin ke dalam aksara Sunda:

Contoh 1:

teks : nama saya hendro	
proses 1 : n	ᮇ᮪
proses 2 : na	ᮇ
proses 3 : m	ᮙ᮪
proses 4 : ma	ᮙ
proses 5 : s	ᮞ᮪
proses 6 : sa	ᮞ
proses 7 : y	ᮜ᮪
proses 8 : ya	ᮜ
proses 9: h	ᮠ᮪
proses 10: he	ᮦ
proses 11 : n	ᮇ᮪
proses 12 : d	ᮓ᮪
proses 13 : dr	ᮓ᮪ᮕ᮪
proses 14 : dro	ᮓ᮪ᮢ

Penjelasan : pada proses di atas mengabaikan spasi, dan juga pada proses di atas hanya meninjau per aksara saja
Hasil dari proses pengonversian *string* di atas ialah :

ᮇᮙ ᮞᮞᮜ ᮦᮠᮇ᮪ᮓ᮪ᮢ

Pada bahasa Sunda, ada beberapa aturan pengucapan yang cukup berbeda. Yaitu /é/ atau biasa disebut /e/ dengan curek (tanda ' kecil di atas huruf e). Jika tidak memakai insert→simbol akan sulit membedakan antara /e/ biasa dan /é/ dengan curek di atas

IV. KESIMPULAN

Algoritma pencocokan *string* dengan *brute force* cukup efektif untuk contoh kasus yang kecil. Algoritma ini menelusuri teks dari kiri ke kanan sampai teks habis. Pada kasus konveri alfabet latin ke aksara Sunda, *pattern* menelusuri dari satu sampai (maks) empat karakter. Karena hanya empat karakter kombinasi maksimal untuk satu aksara. Menurut penulis, algoritma *brute force*

sangat cocok karena memang tidak membutuhkan algoritma seperti KMP yang menyimpan jumlah pergeseran ataupun Boyer-Moore yang melihat dari belakang tulisan.

V. UCAPAN TERIMA KASIH

Terima kasih saya ucapkan kepada Allah SWT. karena dengan rahmatnya penulis bisa menyelesaikan tulisan ini. Terima kasih juga kepada kedua orang tua penulis yang selalu berdoa dan bersabar. Terima kasih kepada Dr. Ir. Rinaldi Munir dan Ibu Masayu Leyla Khodra yang telah membimbing dan mendukung penulis. Terima kasih juga kepada teman-teman yang senantiasa membantu, dan kepada semua orang yang telah membantu dalam menyelesaikan tulisan ini yang tidak dapat disebutkan satu per satu.

DAFTAR PUSTAKA

[1] _____, 2009, "Apa itu Sunda Kaganga", http://sunda-kaganga.blogspot.com/2009/11/apa-itu-sunda-kaganga.html (diakses 17 Mei 2014).

[2] Alfatih, M. Faris, 2013, "Karakteristik, Kelebihan, dan Kelemahan Algoritma *Brute Force*. http://faris6593.blogspot.com/2013/10/karakteristik-kelebihan-kelemahan-algoritma-brute-force.html (diakses tanggal 17 Mei 2014).

[3] Boyer, Robert Moore, J. 1977. A Fast String Searching Algorithm. Comm. ACM 20: 762–772

[4] Munis, Rinaldi. Diktat Kuliah IF2211 Strategi Algoritma. Program Studi Teknik Informatika STEI ITB.

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, 19 Mei 2014

Hendro Triokta Brianto – 13512081