Membangun Aplikasi Pencarian Rumus Fisika dengan Pengimplementasian Booyer-Moore

Kevin Muharyman Alhaafizh 13516124
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13516124@std.stei.itb.ac.id

***Abstrak*— Bagi sebagian orang menghapalkan rumus – rumus matapelajaran merupakan hal yang sangat sulit, dibutuhkan ketelitian dan pemahaman yang baik agar dapat menjawab soal dengan benar. Penelitian ini dilakukan untuk meneliti cara melakukan pendektesian kata kunci yang terdapat di dalam soal . Algoritma yang dipakai adalah Booyer-Moore , setiap kata kunci disimpan di basis data akan di cocokkan dengan soal yang sudah di pisah perkata dan akan mengeluarkan rumus yang disarankan.**

***Kata Kunvi* — Kata Kunci , Booyer-Moore , Rumus**

I. PENDAHULUAN

Di Jaman Era Globalisasi ini kemajuan teknologi sangat berkembang pesat begitupun dengan perkembangan di bidang pendidikan . Tidak bisa kita pungkiri bahwa semua teknologi yang kita rasakan adalah hasil dari semua kerja keras para peniliti orang terdahulu hingga tercipta teknologi seperti sekarang ini. Teknologi yang tercipta ini meskipun kelihatan memiliki bidangnya masing – masing sebenarnya semua teknologi ini menggunakan permasalahan , pemecahan masalah dan rumus yang mendasar.Hal – hal yang mendasar ini akhirnya dipecahkan oleh ilmuan – ilmuan pada masanya dan menciptakan teori bahkan rumusan untuk menyelesaikan masalah.Seiring berjalannya waktu banyak sekali para peniliti yang menciptakan rumusan rumusan terbaru . Rumus – rumus yang ada di jaman sekarang ini cukup banyak bahkan jika sangat sulit bagi kita untuk menghapalkannya terutama ketika sedang mengerjakan soal ,tidak jarang dari kita harus melihat kembali rumus – rumus yang ada di buku dan ini menjadi permasalahan bagi sebagian orang yang sulit untuk menghapalkan rumus .

Namun jika dilihat dari siswanya sendiri sebenarnya mereka hanya tidak mau melakukan perjuangan lebih . Di jaman sekarang ini banyak sekali kita lihat orang – orang yang terlena akibat adanya kemajuan teknologi sehingga tidak banyak dari kita yang lebih melilih bermalas – malasan . Al hasil mereka tidak akan mendapatkan apa yang mereka inginkan dan tidak berjalan secara optimal.

Penelitian ini bertujuan untuk membantu sebagian orang agar dapat menghapal rumus – rumus yang sudah ada dengan mengacu pada soal yang sudah diberikan sehingga orang – orang sudah terbiasa menjawab soal menggunakan rumus – rumus tersebut jika diberikan persoalan yang sejenis, hal ini diharapkan sangat memudahkan para siswa.

Dalam pengerjaan penelitian ini algoritmayang digunakan adalah algoritma Booyer-Moore ,karena peneliti melihat bahwa algoritma ini sangat cocok untuk pencarian kata kunci yang ada di soal , peneliti berencana untuk membangun sebuah aplikasi berbasis android untuk memunculkan rumus apa yang harus digunakan dalam penyelesaian soal tersebut.

II. DASAR - DASAR TEORI

II.1 Algoritma Booyer - Moore

Algoritma pencarian string diantaranya ada Knuth-Morris-Pratt , Regex , BruteForce dan ada juga Booyer-Moore. Algoritma Booyer-Moore adalah salah satu algoritma pencarian string yang diperkenalkan oleh Robert S.Booyer dan J. Strother Moore pada tahun 1977 . Algoritma Booyer-Moore ini dianggap sebagi algoritma yang paling efisien pada aplikasi yang umum . Algoritma Booyer- Moore bisa dikatakan Algoritma yang unik karena algoritma ini memiliki karakteristik yang unik dan berbeda dengan algoritma lainnya, pada algoritma lain biasanya pencarian string akan dimulai dari kiri ke kanan ,sedangkan pada algoritma Booyer-Moore pencarian string akan dimulai dari kanan ke sebelah kiri. Ide pencarian string terbalik ini bertujuan bahwa jika kita melakukan pencarian dari kanan ke kiri ,bukan dari kiri ke kanan , maka akan semakin banyak informasi yang didapat dalam waktu singkat.

II.2 Langkah – Langkah Algoritma Booyer-Moore

Algoritma Boyer-Moore mempunyai empat konsep dasar di dalam proses pencarian string, yaitu :

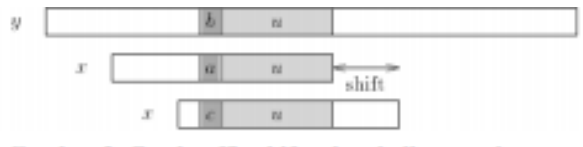
1. Preprocessing
2. Right-to-left scan
3. Bad-character Rule
4. Good-suffix Rule

Precomputation dari algoritma Boyer-Moore terdiri dari bad-character preprocessing dan goodsuffix preprocessing. Prinsip dasar yang pertama dari algoritma Boyer-Moore adalah melakukan perbandingan antara pattern yang dicari dengan teks. Perbandingan pattern dengan teks dilakukan dari arah kanan ke kiri. Perbandingan dimulai dengan membandingkan

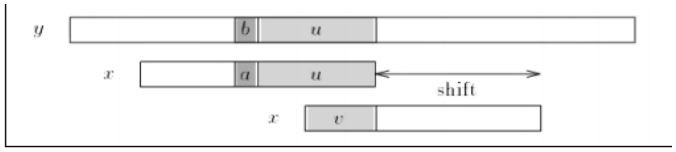
antara karakter paling kanan dari pattern dengan teks. Jika terjadi kecocokkan, maka perbandingan akan dilanjutkan dengan karakter yang disebelah kiri dari yang dibandingkan sampai ke karakter pertama dari pattern. Jika terjadi ketidakcocokkan maka akan dilakukan pergeseran yang ditentukan oleh 2 fungsi pergeseran yaitu bad-character shift dan good-suffix shift. Aturan dari bad-character shift dibutuhkan untuk menghindari pengulangan perbandingan yang gagal dari suatu karakter dalam teks dengan pattern. Aturan dari good-suffix shift dibutuhkan untuk menangani kasus yang di dalamnya terdapat pengulangan karakter pada pattern.

Secara sistematis, langkah-langkah yang dilakukan algoritma Boyer-Moore pada mencocokkan string adalah (Chiquita, Christabella, 2012) :

1. Buat tabel pergeseran string yang dicari (S) dengan pendekatan Match Heuristic (MH) dan Occurence Heuristic (OH), untuk menentukan jumlah pergeseran yang akan dilakukan jika mendapat karakter tidak cocok pada proses pencocokkan dengan string (T).
2. Jika dalam proses perbandingan terjadi ketidakcocokkan antara pasangan karakter pada S dan karakter pada T, pergeseran dilakukan dengan memilih salah satu nilai pergeseran dari dua tabel analisa string yang memiliki nilai pergeseran paling besar.
3. Dua kemungkinan penyelesaian dalam melakukan pergeseran S, jika sebelumnya belum ada karakter yang cocok adalah dengan melihat nilai pergeseran hanya pada tabel Occurence Heuristic, jika karakter yang tidak cocok tiak ada pada S, maka pergeseran adalah sebanyak jumlah karakter pada S, dan jika karakter yang tidak cocok ada pada S, maka banyaknya pergeseran bergantung pada nilai tabel.
4. Jika karakter pada teks yang sedang dibandingkan cocok dengan karakterpada S, maka posisi karakter pada S dan T diturunkan sebanyak 1 posisi, kemudian dilanjutkan dengan pencocokkan pada posisi tersebut dan seterusnya. Jika kemudian terjadi ketidakcocokkan karakter S dan T, maka dipilih nilai pergeseran terbesar dari dua tabel analis pattern, yaitu nilai dari tabel Match Heuristic dan tabel Occurence Heuristic dikurangi dengan jumlah karakter yang telah cocok.
5. Jika semua karakter telah cocok, artinya S telah ditemukan di dalam T, selanjutnya geser pattern sebanyak 1 karakter.
6. Lanjutkan sampai akhir string T.



Gambar 1 . Ilustrasi pencarian string menggunakan algoritma Booyer-Moore 1



Gambar 2 . Ilustrasi pencarian string menggunakan algoritma Booyer-Moore 2

II.3 *Pseudocode* Algoritma Booyer-Moore

Secara garis besar *pseudocode* algoritma Booyer-Moore dibagi menjadi tiga bagian diantaranya adalah *pseudocode* untuk preBmBc yang berfungsi untuk menentukan seberapa besar pergeserakan yang akan dilakukan untuk mencapai karakter tertentu paa *pattern* , *pseudocode* untuk preSuffix untuk mencatat atau memeriksa kecocokan sejumlah karakter yang dimulai dari setiap karakter yang lebih kiri dari karakter terkanan tadi dan *psuedocode* untuk PreBmGs yang memiliki fungsi untuk mengetahui berapa banyak langkah pada *pattern* dari sebuah segmen ke segmen lain.Hasil dari preBmGs dan preBmBc akan dibuat kedalam tabel.Berikut ini adalah *pseudocode* dari algoritma Booyer-Moore.

```
//Algoritma Booyer-Moore fase sebelum pencocokan
//(preBmBc)
procedure preBmBc(input P :array[0..n-1] of char,input n
:integer ,input/outpout bmBc :array[0..n-1] of integer )
  //Kamus
  i : integer ;
  //Algoritma
  for (i = 0  to ASIZE)
        bmBc[i] <- m ;
  endfor
  for (i = 0 to m - 2)
        bmBc[P[i]]=m-i-1 ;
endfor
```

Gambar 3 . *pseudocode* algoritma Booyer-Moore sebelum pencocokan (preBmBc)

```
  //Algoritma Booyer-Moore fase pra-pencocokan ,
preSuffixes.
  procedure preSuffixes(input P : array[0..n-1] of char ,
input n : integer ,input/output suff : array[0..n-1] of
integer)
  //Kamus
        f,g,i : integer
  //Algoritma
        suff[n-1]<-n-1
        g <- n-1
        for (i=n 2 downto 0)
              if(i>g and (suff[i+n-1-f]<i-g))
                    suff[i] <- suff[i+n-1-f]
              else
                    if (i<g)
```

```
                        g<- i
                    endif
                    f <- i-1
                while (g>= 0 and P[g]=P[g+n-1-f])
                        g--
                    endwhile
                    suff[i]=f-g
            endif
    endfor
```

Gambar 4 *.pseudocode* algoritma Booyer-Moore fase setelah pencocokan preSuffix

```
//Algoritma Booyer-Moore fase pra-pencocokkan , preBmGs
  procedure preBmGs (
        input P : array[0..n-1] of char,
        input n : integer ,
        input/output bmGs : array[0..n-1] of integer)
  //Kamus
        i,j : integer ;
        suff : array[0..RuangAlpabet] of integer
  //Algoritma
        preSuffixes(x,n,suff)
        for(i<-1 to m-1)
            bmGs[i]<-n
        endfor
        j<-0
        for (i<n-1 downto 0)
            if (suff[i]=1+1)
                for (j<-j to n-2 - i)
                    if (bmGs[j]=n)
                        bmGs[j]=n-1-i
                    endif
                endfor
            endif
        endfor
        for(i<- 0 to n-2)
            bmGs[n-1-suff[i]]<-n-1-i
        endfor
```

Gambar 5. *pseudocode* Booyer-Moore preBmGs

```
public void kksep()
{   for(int x=0; x<dataList.size(); x++) {
      String kk2 = kk1[x].toString().replace(",", "");
      b = 1;
      int flg=0;
      huruf = "";
      String allkk = kk2 + " ";
      //Log.e("allkk", String.valueOf(allkk.length()));
      for (i = 0; i < allkk.length(); i++) {
        String cek = allkk.substring(b - 1, b);
        //bukan spasi
        if (!cek.equals(" ")) {
          huruf += cek;
          //Log.e("bc", huruf);  }
        else
        {  flg++;
          //Log.e("All", String.valueOf(flag));
          kata1[index] = huruf;
          huruf = "";
          index++; }
        b++; }
flag[x] = flg; }}
```

Gambar 6. Kode program pemisah kata

```
ublic class BM {
  private int R;     // the radix
  private int[] right;     // the bad-character skip array
  private char[] pattern;  // store the pattern as a
haracter array
  String pat;      // or as a string
  public BM(String pat) {
    this.R = 256; this.pat = pat;
    Log.e("pat",pat);
    // position of rightmost occurrence of c in the pattern
    right = new int[R];
    for (int c = 0; c < R; c++)
      right[c] = -1;
    for (int j = 0; j < pat.length(); j++)
      right[pat.charAt(j)] = j;  }
  public BM(char[] pattern, int R) {
    this.R = R;
    this.pattern = new char[pattern.length];
    for (int j = 0; j < pattern.length; j++)
      this.pattern[j] = pattern[j];
    // position of rightmost occurrence of c in the pattern
    right = new int[R];
    for (int c = 0; c < R; c++)
      right[c] = -1;
    for (int j = 0; j < pattern.length; j++)
      right[pattern[j]] = j; }
  public int search(String txt) {
    int M = pat.length();
    int N = txt.length();
    int skip;
    for (int i = 0; i <= N - M; i += skip) {
      skip = 0;
      for (int j = M-1; j >= 0; j--) {
        if (pat.charAt(j) != txt.charAt(i+j)) {
          skip = Math.max(1, j - right[txt.charAt(i+j)]);
          break; }  }
      if (skip == 0) return i;    // found  }
    return N;                     // not found  }
  public int search(char[] text) {
    int M = pattern.length;
    int N = text.length;
    int skip;
    for (int i = 0; i <= N - M; i += skip) {
      skip = 0;
      for (int j = M-1; j >= 0; j--) {
        if (pattern[j] != text[i+j]) {
          skip = Math.max(1, j - right[text[i+j]]);
          break; }  }
      if (skip == 0) return i;    // found  }
    return N;          // not found  } }
```

Gambar 7. Kode Program Algoritma Booyer-Moore

II.4 Kelebihan dan Kekurangan Algoritma Booyer-Moore

Kelebihan dari algoritma Boyer-Moore (Utomo, 2008) ini semakin panjang pola yang dicari maka waktu pencarian semakin singkat.

Sedangkan kekurangan algoritma Boyer-Moore adalah lebih lambat untuk pattern yang pendek dan tidak bagus untuk pencarian binary string. (Aulia, 2008).

III. METODE PENELITIAN

Dalam pembuatan aplikasi berbasis android ini ada beberapa gambaran yang akan ditunjukkan agar memudahkan user dalam meenggunakan aplikasi tersebut salah satu caranya adalah dengan membuat flowchart dari aplikasi yang dibuat yaitu sebuah aplikasi yang dapat melakukan pencarian dan menampilkan rumus yang sesuai dengan kebutuhan dari soal.

III.1 Flowchart

Gambar 7. Flowchart Pencarian Rumus

Flowchart yang berada pada gambar 7 merupakan flowchart menu tampilan atau tahapan – tahapan yang akan dilakukan user saat menggunakan aplikasi untuk pencarian rumus dalam penyelesaian soal – soal fisika dengan menerapkan algoritma Booyer-Moore.Berikut penjelasan dari flowchart di atas :

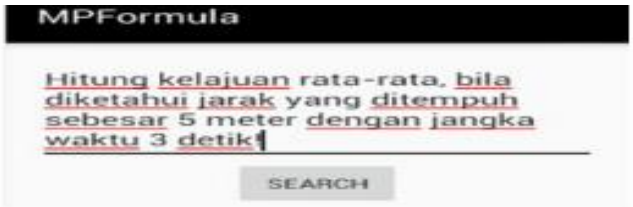
a. User menjalanka aplikasi . Kegiatan ini diwakili oleh simbol terminator yang bertuliskan “START” yang berarti user mengakses menu search.
b. Saat user mengakses menu search , sistem akan mengambil data dari tabel detil berupa id rumus , nama rumus , kata kunci , dll.
c. Saat proses mengambil data sudah selesai kemudian user kan memasukkan soal. Kegiatan tersebut bertuliskan “Masukkn soal “
d. Setelah user memasukkan soal maka sistem akan mengubah semua huruf menjadi huruf kecil
e. Setelah dirubah menjadi huruf kecil maka sistem akan mengubah kata kunci menjadi per kata
f. Setelah huruf diubah menjadi kecil dan kata kunci dirubah menjadi per kata maka akan dilakukan pemilihan yaitu pada kegiatan Pencocokan , jika cocok maka akan muncul list rumus , jika tidak ada kata yang cocok maka akan muncul pesan “tidak ada rumus yang tepat” dan langsung menuju end.
g. Jika cocok maka akan lanjut ke kegiatan selanjutnya yaitu muncul list rumus. Sistem akan memunculkan list rumus yang sesuai dengan kata kunci
h. Kemudian user dapat memilih rumus yang dia rasa mana yang paling sesuai dengan kebutuhan soal
i. Lalu di kegiatan ambil data detail user dapat melihat secara mendalam terhadap rumus yang sudah dipilih
j. Terakhir ialah tampilan halam detil beserta rumus hasil dari pilihan user .
k. Setelah sudah di halaman detil dan tampilan detil rumus ,maka jalan sistem sudah selesai.jika ingin melakukan pencarian lagi maka harus ulang dari poin pertama lagi.

III.2 Arsitektur Sistem

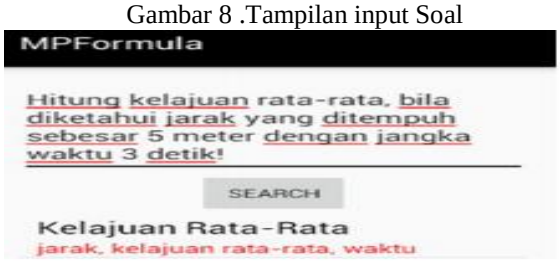
Arsitektur sistem mennjelaskan interaksi sistem dengan pengguna . Dalam ini pengguna berinteraksi dengan Android. Android terhubung ke internet , internet digunakan untuk mengakses server dimana server memiliki database dari sistem.

IV. HASIL PENELITIAN DAN PEMBAHASAN

Pada Aplikasi yang sudah mengimplementasikan algoritma Booyer-Moore , Pengguna dapat mennjalankan aplikasi dengan memilih menu yang sesuai dengan keinginan pengguna atau user.



Gambar 8 .Tampilan input Soal



Gambar 9. Hasil Pencocokan Booyer-Moore

Dari beberapa kali percobaan yang dilakukan didapatkan hasil bahwa dengan mengimplementasikan algoritma Booyer-Moore dalam pencarian rumus fisika yang sesuai dengan kebutuhan soal fisika didapat data bahwa semua soal dapat mengeluarkan rumus yang tepat dan mengeluarkan hasil yang pasti dan tingkat keberhasilan dari 30 percobaan adalah 100 %. Ini menunjukkan bahwa algoritma ini sangat baik dalam melakukan pencarian string yang pada kasus ini adalah pencarian rumus yang tepat sesuai kata kunci yang sudah diberikan.

V. KENDALA UMUM

Kendala umum yang kerap kali dihadapi dalam Membangun sebuah aplikasi menyelesaikan persoalan fisika dengan pengimplementasian algoritma Booyer-Moore untuk pencarian Rumus adalah :

1. Masih banyak para siswa yang tidak mengetahui bagaimana cara untuk mendapatkan rumus.
2. Para siswa meskipun sudah diberikan fasilitas namun jadi semakin malas.
3. Dibutuhkan kerja sama dengan para ahli agar dapat menjamin rumus yang diberikan sudah benar.
4.

VI. KESIMPULAN

Algoritma Booyer-Moore adalah sebuah algoritma yang digunakan untuk pencarian string yang pada kasus ini dilakukan untuk mencari kata kunci dan menampilkan rumus – rumus fisika yang mungkin dari soal yang diajukan.Algoritma Booyer ini merupakan algoritma yang sangat unik karena melakukan pencarian dari kanan ke kiri dengan alasan bahwa jika melakukan pencaria dari kanan maka semakin banyak informasi yang disimpan.Dengan Algoritma ini diharapkan lebih cepat untuk melakukan pencarian kata. Dari 30 percobaan yang dilakukan didapat bahwa tingkat keberhasilan menampilkan rumus yang sesuai dengan yang diinginkan adalah sebesar 100% hasil ini tentunya cukup baik, dan ini bisa menjadi tolak ukur dan menjadi motivasi agar aplikasi ini benar – benar berjalan sehingga dapat memudahkan banyak orang.

VII. UCAPAN TERIMA KASIH

Saya mengucapkan terima kasih kepada Allah SWT atas berkat rahmat-Nya lah saya dapat menyelesaikan makalah ini dengan sebaik mungkin. Kemudian terima kasih kepada Dr. Masayu Leylia Khodra ST.MT ,Dr. Ir. Rinaldi Munir, M.T, Dr. Nur Ulfa Maulidevi, S.T., M.Sc . dan. selaku dosen mata kuliah IF 2211 Strategi Algoritma yang telah membimbing dan memberi materi kepada penulis selama proses pengajaran mata perkuliahan Strategi Algoritma serta telah memberikan tugas ini sebagai pemacu saya untuk dapat memberikan sesuatu yang bermanfaat bagi sesama umat manusia mengenai keilmuan informatika.

Terakhir, saya ucapkan terima kasih kepada orang tua saya, dan teman-teman saya yang senantiasa menyemangati saya dalam menyelesaikan makalah ini serta telah memberikan masukkan kepada saya seputar makalah ini. Saya harap makalah ini dapat bermanfaat bagi pembaca maupun penulis . Tak lupa penulis memohon maaf apabila ada kesalahan dalam pengambilan kata – kata yang kurang berkenaan di hati pembaca.

REFERENCES

[1] https://www.researchgate.net/profile/Antonius_Rachmat/publication/263810958_IMPLEMENTASI_ALGORITMA_BOYER-MOORE_PADA_PERMAINAN_WORD_SEARCH_PUZZLE/links/5428220c0cf238c6ea7cd1d0/IMPLEMENTASI-ALGORITMA-BOYER-MOORE-PADA-PERMAINAN-WORD-SEARCH-PUZZLE.pdf.

[2] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Pencocokan-String-(2018).pdf

[3] http://repository.uksw.edu/bitstream/123456789/3041/2/ART_Darmawan%20Utomo%2C%20Eric%20Wijaya%20Harjo%2C%20Handoko_Perbandingan%20Algoritma%20String_Full%20text.pdf

[4] https://yodymada.blogspot.co.id/2013/09/algoritma-turbo-boyer-moore_7.html

[5] Argakusumah, K.W., & Hansun, S. (2014). Implementasi Algoritma Boyer-Moore pada Aplikasi Kamus Kedokteran Berbasis Android. Jurnal ULTIMATICS Vol.6. No.2.

[6] Ramadhansyah. (2013). Perancangan Aplikasi Kamus Bahasa Gayo Dengan Menggunakan Metode Boyer Moore. Jurnal Pelita Informatika Budi Darma Vol.4. No.3. hal 118-122.

[7] Zulen, A. A. (2009). Penerapa Algoritma Backtracking Pada Permainan Word Search Puzzle. Makalah IF3051 Strategi Algoritma , 1-2.

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

Kevin Muharyman Alhaafizh
13516124