String Matching(Pencocokan String)
Dengan Metode Brute Force dan KMP(Knut Morris Pratt)

Oleh :

Mochamad Irfan Mardani(10112240)

Fahrul Rizaldi (10112272)

Teddy Khairul Jamil (10112282)

Abstrak

Kini pencocokan string atau string matching sudah menjadi masalah yang sangat kompleks. Kita menemukan banyak sekali aplikasi dari pencocokan string terutama dalam bidang komputer, seperti kita akan mencari sebuah artikel di internet kita cukup mengetikkan kata kunci pada search engine dan seach engine akan memberikan beberapa rekomendasi dari kata kunci tersebut. Tidak hanya pada search engine pencocokan string atau string matching pun banyak ditemukan pada aplikasi-aplikasi yang terinstal di komputer kita. Untuk membuat sebuah aplikasi pencarian string tentunya kita membutuhkan suatu algoritma untuk melakukan pencocokan string, adapun algoritma yang akan kita bahas dalam makalah ini yaitu Algoritma Brute force dan Almgoritma KMP(Knuth Morris Pratt). Setiap algoritma yang digunakan pasti memiliki kelebihan dan kekurangan masing-masing, elehkarena itu kami akan membandingkan kedua algoritma tersebut dalam makalah ini.

1. Pendahuluan

Pencocokan String atau string matching memang sangat diperlukan dalam kehidupan kita sehari-hari ada banyak algoritma untuk melakukan pencocokan string, setiap algoritma memiliki kekurangan dan kelebihan masing-masing. Oleh karena itu dari sekian banyak algoritma yang dapat digunakan dalam melakukan string matching kami hanya akan membandingkan dua buah algoritma sebagai penelitian kami, yaitu Algoritma Brute Force dan Algoritma KMP(Knut Morris Pratt).

2. Algoritma Brute Force

Brute force merupakan sebuah pendekatan yang digunakan untuk memecahkan suatu masalah khususnya di bidang komputer. Algoritma Brute Force memecahkan masalah dengan cara memproses seluruh data, algoritma pastinya sangat eksak karena algoritma brute foce meng eksekusi semua kemungkinan yang ada dalam pemecahan masalahnya. Untuk memecahkan masalah dengan data yang kecil algoritma ini masih dapat digunakan. Namun, bila kita berbicara data yang besar algoritma ini sangatlah tidak efisien karena algoritma ini mencoba semua kemungkinan untuk memecahkan masalah. Sehingga kompleksitas waktunya pun semakin tinggi.

2.1 Algoritma Brute Force Dalam Pencocokan String

Dengan asumsi bahwa teks berada di dalam array T[1..n] dan pattern berada di dalam array P[1..m], maka algoritma brute force pencocokan string adalah sebagai berikut :

1. Mula-mula pattern P dicocokan pada awal teks T.
2. Dengan bergerak dari kiri ke kanan, bandingkan setiap karakter di dalam pattern P dengan karakter yang bersesuaian di dalam teks T sampai :
   a. Semua karakter yang dibandingkan cocok atau sama (pencarian berhasil), atau
   b. Dijumpai sebuah ketidak cocokan karakter (pencarian belum berhasil)

3. Bila pattern P belum ditemukan kecocokan dan teks T belum habis, geser pattern P satu karakter ke kanan dan ulangi langkah 2.

Contoh :

Teks : Masa depan di tangan anda
Pattern : pan

Masa de**pan** ada di tangan anda
S=0 pan
S=1 pan
S=2 pan
S=3 pan
S=4 pan
S=5 pan
S=6 pan
S=7 **pan**

Program Brute Force String matching menggunakan bahasa C

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

int main()
{
  bool ketemu=false;
  int cariLength;
  int kataLength;
  int i,a;
  int indeks;
  char cari[20];

  //String
  char kata[100];
  printf("Masukkan sebuah teks (max 100 digit) = ");
  fflush(stdin);
  gets(kata);

  //Pola yang dicari
  printf("Masukkan pattern = ");
  fflush(stdin);
  gets(cari);

  //strlen = fungsi untuk mendapatkan panjang string
  cariLength = strlen(cari);
  kataLength = strlen(kata);

  //Mulai Brute Force dalam Pencarian String

  for(i=0;i<=kataLength-cariLength;i++)
  {
   int j=0;
   while(j<cariLength && kata[i+j] == cari[j])
   {
    j++;
    if(j >= cariLength)
    {
     for(a=0; a<cariLength ; a++)
     {
     indeks = i+a;
     printf(" %c DItemukan Pada Indeks = ke-%d \n",cari[a],indeks);
     ketemu=true;
     }
    }
   }
  }

  if(ketemu==false)
    {
      printf("Pola tidak ditemukan");
    }
  //Akhir Brute Force Pattern Matching

  getch();
  return 0;
}
```

2.2 Kompleksitas Waktu Algoritma Brute Force

Kompleksitas algoritma pencocokan string dihitung dari jumlah operasi perbandingan yang dilakukan.

Kompleksitas kasus terbaik adalah O(n). Kasus terbaik terjadi karakter pertama pattern P tidak pernah sama dengan karakter teks T yang dicocokan (kecuali pada pencocokan yang terakhir). Pada kasus ini, jumlah perbandingan yang dilakukan paling banyak n kali.

Sedangkan kasus terburuk membutuhkan m(n-m+1) perbandingan, yang mana kompleksitasnya adalah O(mn)

3. Algoritma Knuth Morris Pratt (KMP)

Algoritma Knuth-Morris-Pratt dikembang kan secara terpisah oleh Donald E. Knuth pada tahun 1967 dan James H. Morris bersama Vaughan R. Pratt pada tahun 1966, namun keduanya mempublikasikan nya secara bersamaan pada tahun 1977.

Pada algoritma ini juga membandingkan karakter pada teks dari kiri ke kanan, tetapi berbeda pada cara pergeserannya. Ide dari algoritma ini adalah dengan memanfaatkan karakter-karakter pola yang sudah diketahui yang ada di dalam teks sampai terjadinya ketidakcocokan untuk melakukan setiap pergeserannya.

Yang paling utama pada algoritma ini adalah dibutuhkannya sebuah tabel yang berisi nilai pinggiran dari masing-masing karakter untuk pengetesan string. Tabel tersebut dibuat sebelum proses pencarian string dilakukan yang dibentuk dengan prosedur terpisah.

Contoh :

Teks : aabtg caarf aabcd
Pattern : aabc
aabtg caarf **aabc**d
S=0 aabc
S=1 aabc
S=2 **aabc**

Program Knuth Morris Pratt dalam bahasa C

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>



int carikmp(char *string, char *kata, int *ptr,bool *ketemu) {
    int i = 0, j = 0;
    while ((i + j) < strlen(string)) {
        if (kata[j] == string[i + j]) {
            if (j == (strlen(kata) - 1)) {
                printf("%s Berada pada indeks ke : %d\n",kata, i);
                *ketemu=true;
                return;
            }
            j = j + 1;
        } else {
            i = i + j - ptr[j];
            if (ptr[j] > -1) {
                j = ptr[j];
            } else {
                j = 0;
            }
        }
    }
}
/* Mencari kecocokan pada kata dengan pattern */
void carikesamaan(char *kata, int *ptr) {
    int i = 2, j = 0, len = strlen(kata);
    ptr[0] = -1;
    ptr[1] = 0;
    while (i < len) {
        if (kata[i - 1] == kata[j]) {
            j = j + 1;
            ptr[i] = j;
            i = i + 1;
        } else if (j > 0) {
```

```
                j = ptr[j];
            } else {
                ptr[i] = 0;
                i = i + 1;
            }
    }
    return;
}

int main() {
    bool ketemu=false;
    char kata[256], string[1024];;
    int *ptr, i;

    printf("String Matching Dengan Algoritma
KMP\n");
    printf("-----------------------------------------
\n");
    /* Inputan teks */
    printf("Masukan Teks:");
    fgets(string, 1024, stdin);
    string[strlen(string) - 1] = '\0';

    /* Inputan Pattern*/
    printf("Masukan Pattern:");
    fgets(kata, 256, stdin);
    kata[strlen(kata) - 1] = '\0';

    /* Array pinggiran */
    ptr = (int *)calloc(1, sizeof(int) *
(strlen(kata)));

    /* Mencari kesamaan kata dengan pattern
*/
    carikesamaan(kata, ptr);
    /* mencari indeks pattern pada teks */
    carikmp(string, kata, ptr,&ketemu);
    if(ketemu==false){
      printf("\nTidak ditemukan");
    }
    getch();
    return 0;
}
```

2.3 **Kompleksitas waktu**

Kompleksitas waktu pada algoritma Knuth Morris Pratt (KMP) memiliki kompleksitas waktu yang lebih sedikit dibandingkan dengan brute force. Untuk prosedur pengisian nilai pinggiran dibutuhkan waktu O(m),sedangkan pencarian string membutuhkan waktu O(n), sehingga kompleksitas waktu algoritma KMP adalah O(m+n).

3. Kesimpulan

Penggunaan algortma Brute force untuk string matching(pencocokan string) kurang efektif jika di bandingkan dengan algoritma Knuth Morris Pratt (KMP), Apalagi jika Kata yang di cari sudah mencapai ribuan.

REFERENSI

1. Rinaldi Munir. Diktat Kuliah IF3051 Strategi Algoritma.
2. Dwinanto Cahyo. ALGORITMA PENCARIAN STRING DENGAN ALGORITMABRUTE FORCE, KNUTH-MORRIS-PRATT DAN ALGORITMA DUA ARAH