**LAPORAN TUGAS BESAR**

**STRUKTUR DATA DAN ALGORITMA**



Disusun oleh :

Afif Rana Muhammad (171524001)

Fajar Restu Ilahi (171524009)

Zahran Haykal Mubarok (171524031)

PROGRAM STUDI D4 -TEKNIK INFORMATIKA

JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA

POLITEKNIK NEGERI BANDUNG

KAB. BANDUNG BARAT

2018

1. **DESKRIPSI**

Program text similarity merupakan program yang melakukan pengecekan kemiripan atau kesamaan antara isi satu file dengan file lainnya (berekstensi .txt). Pengecekan tersebut dilakukan dengan mengolah statistik kata dari kedua file. Kemudian, dilakukan pembandingan statistik di antara keduanya. Sehingga, hasil akhir yang didapatkan berupa persentase kemiripan kedua text tersebut.

Program dijalankan melalui console dengan pembuatannya melalui Dev C++ IDE.

1. **CARA KERJA PROGRAM**

Program text similarity memiliki cara kerja yang terbagi menjadi beberapa alur yaitu sebagai berikut :

1. Pembuatan AVL Tree Stopword

* Membuka file kamus Stopword (berisi karakter yang bersifat *lower*

*case* dan tidak ada tanda baca).

* Membaca isi file tersebut per karakter, lalu ditampung di array dinamis (String).
* Bila karakter yang ditemukan berupa spasi, maka String yang sudah didapat akan diputus.
* Lalu String tersebut akan dimasukkan ke dalam AVL Tree Stopword.
* Kemudian, pembacaan isi file per karakter akan terus diulang hingga mencapai EOF.
* Hasil akhirnya berupa AVL Tree Stopword.

2. Pembuatan AVL Tree Statistik File Pertama

* Membuka file pertama yang ditentukan (berisi karakter bebas).
* Membaca isi file tersebut per karakter sambil dilakukan proses *case folding*, lalu ditampung di array dinamis (String).
* Bila karakter yang ditemukan berupa spasi, maka String yang sudah didapat akan diputus.
* Kemudian, String tersebut dibandingkan dengan daftar Stopword yang sudah dibuatkan Tree sebelumnya.
* Jika, String tersebut merupakan Stopword, maka akan dilakukan pembacaan karakter berikutnya, dan String tersebut tidak akan disimpan.
* Tetapi, jika String tersebut bukan merupakan Stopword, proses akan dilanjutkan.
* Setelah itu, akan dilakukan proses Stemming terhadap String.
* Setelah didapatkan kata dasarnya, String dibandingkan dengan daftar kata yang ada di Tree Statistik.
* Jika ada, count1 pada kata tersebut akan ditambah satu.
* Jika tidak ada, maka string tersebut akan dimasukkan ke dalam AVL Tree Statistik.
* Pada pembentukkan node baru tersebut, akan dilakukan juga proses penghitungan jumlah node dan jumlah count secara keseluruhan(atau bisa dibilang jumlah kata).
* Proses akan diulang hingga ditemukan EOF pada file.

3. Pembuatan AVL Tree Statistik File Kedua

* Membuka file kedua yang ditentukan (berisi karakter bebas).

Membaca isi file tersebut per karakter sambil dilakukan proses *case folding*, lalu ditampung di array dinamis (String).

* Bila karakter yang ditemukan berupa spasi, maka String yang sudah didapat akan diputus.
* Kemudian, String tersebut dibandingkan dengan daftar Stopword yang sudah dibuatkan Tree sebelumnya.
* Jika, String tersebut merupakan Stopword, maka akan dilakukan pembacaan karakter berikutnya. Dan String tersebut tidak akan disimpan.
* Tetapi, jika String tersebut bukan merupakan Stopword, proses akan dilanjutkan.
* Setelah itu, akan dilakukan proses Stemming terhadap String.
* Setelah didapatkan kata dasarnya, String dibandingkan dengan daftar kata yang ada di Tree Statistik, hasil olahan file pertama yang telah dibuat sebelumnya.
* Jika kata tersebut sudah ada, count2 pada kata tersebut akan ditambah satu.
* Jika tidak ada, maka String tersebut akan dimasukkan ke dalam Tree tatistik tersebut.
* Pada pembentukkan node baru tersebut, akan dilakukan juga proses penghitungan jumlah node dan jumlah count secara keseluruhan(atau bisa dibilang jumlah kata).
* Proses akan diulang hingga ditemukan EOF pada file.

4. Pembandingan Kedua File

* Program akan melakukan traversal pada Tree Statistik untuk mencari node yang memiliki nilai count1 dan count2 yang lebih besar dari nol.
* Kemudian, nilai kedua count, baik count1 maupun count2, akan

dibandingkan.

* Nilai count yang lebih besar akan disimpan di field max, dan yang lebih kecil disimpan di field min.
* Langkah tersebut akan diulang hingga tidak ada lagi node yang

memiliki nilai count1 dan count2 yang lebih besar dari nol.

* Field min dan max akan dimasukkan ke dalam rumus :

n% = (min / max) x 100, dengan n sebagai hasil persentase kemiripan kedua text tersebut.

1. **SPESIFIKASI**

Rancangan Struktur Data

    a. Node :

kata : variabel bertipe String

count1 : integer

count2 : integer

left : pointer of Node

right : pointer of Node

parent : pointer of Node

b. Tree Stopword :

root : pointer of Node

current : pointer of Node (membantu traversal)

   c. Tree Statistik

root : pointer of Node

current : pointer of Node (membantu traversal)

   d.  String :

char \*x : pointer of Char

size : integer

used : integer

1. **PEMBAGIAN TUGAS**
2. Tahap Persiapan

* Mencari kamus Stopword (bahasa Indonesia lebih didahulukan) :
* Mencari algoritma/source code Stemming :
* Menyusun format laporan :

2. Tahap Proses Pembuatan Program

* Pembuatan kode pembangunan AVL Tree Stopword :
* Pembuatan kode pada proses pembacaan per karakter yang dilakukan bersamaan dengan case folding :
* Pembuatan kode pada proses penyeleksian Stopword :
* Pembuatan kode pembangunan AVL Tree Statistik :
* Pembuatan kode pembandingan kedua statistik :
* Pembuatan Interface :
* Tester Program :

1. **TIMELINE**

|  |  |
| --- | --- |
| Minggu ke-1 | Proses pemahaman tugas yang diberikan dan eksplorasi algoritma/*source code* (AVL dan *Stemming*). |
| Pembuatan Laporan Tugas besar yang berisikan Deskripsi/Spek/Lingkup tugas dan pembagian yang akan dikerjakan. |
| Minggu ke-2 | Proses eksplorasi pembacaan file (dengan skema seperti pada alur program) serta eksplorasi pembuatan AVL Tree. |
| Minggu ke-3 | Pembuatan kode case folding dan kode penyeleksian stopword serta kode pembandingan statistik kata dari kedua file. |
| Minggu ke-4 | Pembuatan Interface program dan testing program. Apabila masih terdapat *bug* maka akan dilakukan proses *debugging*. |