UNIVERSITAS INDONESIA

PENGEMBANGAN SISTEM PENCARIAN KATA PADA SIMPLE-O

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

MEIRISAL DWI WALDI
0606078405

FAKULTAS TEKNIK UNIVERSITAS INDONESIA
DEPARTEMEN TEKNIK ELEKTRO
DEPOK
JULI 2010

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama	**: Meirisal Dwi Waldi**
NPM	**: 0606078405**
Tanda Tangan	**:**
Tanggal	**: 9 Juli 2010**

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh	:
Nama	: Meirisal Dwi Waldi
NPM	: 0606078405
Program Studi	: Teknik Komputer
Judul Skripsi	: Pengembangan Sistem Pencarian Kata Pada Simple-O

Telah berhasil dipertahankan di hadapan Dewan Penguji dan dterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Komputer, Fakultas Teknik, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing	: Dr. Ir. Anak Agung Putri Ratna M.Eng	( )
Penguji	: Muhammad Salman ST., MIT	( )
Penguji	: Ir. Endang Sriningsih MT, Si	( )

Ditetapkan di : Depok

Tanggal : 9 Juli 2010

KATA PENGANTAR

Puji syukur saya panjatkan kehadirat Allah SWT, karena atas segala rahmat dan hidayat-Nya saya dapat menyelesaikan skripsi ini. Saya menyadari bahwa terselesainya skripsi ini adalah berkat bantuan dari beberapa pihak. Oleh karena itu, saya mengucapkan terima kasih kepada :

1. Tuhan Yang Maha Kuasa, karena dengan rahmat dan hidayat-Nya saya dapat menyelesaikan skripsi ini tepat waktu;
2. Ibu Dr. Ir. Anak Agung Putri Ratna M.Eng selaku pembimbing skripsi ini, terima kasih atas dukungan dan sarannya selama skripsi berlangsung;
3. Orang tua, kakak, adik serta seluruh keluarga yang selalu memberikan dukungannya serta perhatian yang penuh hingga terselesaikannya skripsi ini;
4. Teman, sahabat serta orang-orang terdekat yang telah menjadi tempat diskusi serta tempat saya mengeluarkan segala uneg-uneg;
5. Rekan-rekan satu bimbingan : Boma A. Adhi, Nara Brahma “Bram”, Gregorius Handoyo “Iyus” dan Vivi Zulvi F, terima kasih atas kerja samanya dalam skripsi ini;
6. Teman-teman Elektro Komputer UI 2006;
7. Seluruh sivitas akademik Departemen Teknik Elektro yang tidak dapat saya sebutkan satu persatu.

Depok, Juli 2010

Meirisal Dwi Waldi

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	: Meirisal Dwi Waldi
NPM	: 0606078405
Program Studi	: Teknik Komputer
Departemen	: Teknik Elektro
Fakultas	: Teknik
Jenis Karya	: Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada **Universitas Indonesia Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*)** atas karya ilmiah saya yang berjudul :

PENGEMBANGAN SISTEM PENCARIAN KATA PADA SIMPLE-O

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada Tanggal : 9 Juli 2010

Yang Menyatakan

(Meirisal Dwi Waldi)

ABSTRAK

Nama : Meirisal Dwi Waldi

Program Studi : Teknik Komputer

Judul : Pengembangan Sistem Pencarian Kata pada Simple-O

Teknologi yang sedang dikembangkan saat ini adalah penggunaan sebuah program yang dirancang untuk mengakomodasi ujian essay melalui komputer dimana penilaiannya dilakukan oleh komputer melalui proses perhitungan dengan metode *Latent Semantic Analysis* (LSA). Sistem yang dikenal dengan nama *essay grading* ini memiliki beberapa program pendukung lainnya, salah satunya adalah penggunaan database sebagai tempat penyimpanan informasi dalam sistem ini. Database berperan penting dalam sistem ini, sehingga untuk mendapatkan hasil yang maksimum maka diperlukan sebuah database yang handal. Salah satu proses yang dilakukan dalam sistem ini adalah proses pencarian kata dalam tabel persamaan kata yang berjumlah ribuan kata. Untuk mempercepat proses pencarian ini maka dibutuhkan sebuah metode pencarian kata yang lebih efektif. Metode pencarian kata yang diterapkan untuk memfokuskan proses pencarian kata dengan membagi-bagi tabel persamaan kata telah berhasil diterapkan dan dapat meningkatkan kecepatan proses aplikasi Simple-O dengan peningkatan waktu mencapai 4,185549 detik yang berarti meningkat hingga 7,21 kali lebih cepat.

Kata kunci :

Latent Semantic Analysis, essay grading, database, metode pencarian kata

ABSTRACT

Name : Meirisal Dwi Waldi

Study Program: Computer Engineering

Title : Keyword Search Systems Development at Simple-O

The technology currently being developed is the use of a program designed to accommodate an essay exam on a computer where the assessment done by a computer through a process of calculation by the method of Latent Semantic Analysis (LSA). The system known as grading essays has several other support programs, one of them is the use of a database as a storage of information in this system. Database plays an important role in this system, so to get maximum results will require a reliable database. One of the processes undertaken in this system is the process of finding words in the synonym table numbering a thousand words. To accelerate the search process so needed a word search methods more effective. Word search methods were applied to focus the search process said to divide the tables in common word has been successfully implemented and can improve speed-o simple application process with increased time to reach 4.185549 seconds, which means increased to 7.21 times faster.

Key Words :

Latent Semantic Analysis, essay grading, database, keyword search method

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi komputer khususnya dalam bidang tekonologi informasi dan telekomunikasi yang begitu pesat saat ini telah banyak dapat dimanfaatkan dalam berbagai bidang. Salah satunya adalah dalam bidang pendidikan. Saat ini telah banyak program-program yang dibuat guna membantu dalam proses pembelajaran dalam bidang pendidikan, dimana saat ini yang tengah berkembang adalah penggunaan sebuah program yang dirancang untuk mengakomodasi ujian *essay* melalui komputer baik *online* maupun *offline*.

Saat ini telah berkembang sistem ujian yang bersifat *online* dimana mahasiswa dapat mengerjakan soal ujian melalui secara online, namun masih terbatas pada ujian yang bersifat multiple choice atau pilihan ganda dengan alasan kemudahan dalam mengoreksi jawaban dari ujian tersebut, karena keterbatasan itu lah saat ini tengah dikembangkan sebuah program yang dapat memungkinkan ujian online tersebut dalam bentuk essay yaitu *Essay Grading*.

Pada sistem ujian online ini diperlukan penggunaan database untuk menyimpan berbagai macam data yang diperlukan, seperti data *user*, data soal serta jawaban dan juga data nilai. Selain itu, database juga digunakan untuk menyimpan kata-kata yang akan digunakan untuk membentuk matriks persamaan antara jawaban siswa atau mahasiswa dengan kunci jawaban sebenarnya.

Kecepatan waktu proses pengeksekusian sistem ini juga dipengaruhi oleh tingkat keefektifan perancangan database, dalam hal ini dilihat dari kebutuhan sebuah sistem untuk mencari data dalam database dengan jumlah data yang besar. Kebutuhan akan peningkatan kinerja ini akan sangat terasa ketika sistem ini menjalankan sebuah perintah pencarian data dalam suatu database dalam jumlah yang sangat besar.

Dalam skripsi ini akan mencoba untuk meningkatkan kinerja proses pencarian kata dalam database dimana proses pencarian kata ini berlangsung sebelum proses perhitungan nilai. Peningkatan kinerja pencarian kata ini

diharapkan juga dapat meningkatkan kecepatan sistem dalam mengeksekusi sitem ujian *essay online* ini secara keseluruhan.

Dengan adanya program ini diharapkan dapat membantu proses pembelajaran yang telah ada sekarang menjadi lebih baik.

1.2 Tujuan Penulisan

Tujuan dari penulisan ini adalah untuk mengimplementasikan rancangan peningkatan kinerja database d alam *Essay Grading* dengan keefektifan penggunaan tabel.

1.3 Batasan Masalah

Permasalahan dibatasi pada proses pencarian persamaan kata dalam database kata-kata guna meningkatkan kecepatan pencarian kata dalam database.

1.4 Sistematika Penulisan

Skripsi ini terdiri dari lima bab, dimana masing-masing bab akan berisikan sebagai berikut:

a) Bab 1: Pendahuluan

   Pada bab ini, akan membahas mengenai latar belakang, tujuan penulisan, batasan masalah serta sistematika penulisan untuk menggambarkan isi dari penulisan skripsi ini.

b) Bab 2: *Essay Grading* dan Program Pendukungnya

   Pada bab ini, akan membahas mengenai *essay grading*, LSA, SVD dan algoritma program serta program-program pendukung lain dari *Essay Grading*.

c) Bab 3: Perancangan Metode *Searching* Pada Simple-O

   Pada bab ini, akan membahas mengenai rancangan sistem yang akan digunakan dalam database.

d) Bab 4: Analisa Metode *Seraching* Persamaan Kata Pada Simple-O

   Pada bab ini, akan dijelaskan mengenai penerapan serta hasil uji coba dari sistem *Essay Grading* dengan sistem database yang telah diperbaiki.

e) Bab 5: Kesimpulan

Pada bab ini, akan dijelaskan mengenai kesimpulan yang diperoleh dari buku skripsi ini.

BAB 2
ESSAY GRADING DAN PROGRAM PENDUKUNGNYA

2.1 Sekilas mengenai *Essay Grading*

Di beberapa negara khususnya Indonesia, umumnya terdapat dua jenis ujian yang biasa dilakukan oleh instansi-instansi pendidikan, yaitu ujian berbentuk objektif, salah satunya adalah pilihan ganda. Bentuk soal pilihan ganda selama ini telah banyak digunakan, namun bentuk soal ini dirasa kurang mampu menilai kemampuan dari pelajar yang sebenarnya. Bentuk soal pilihan ganda yang umumnya digunakan, dirasa kurang dapat memberikan penilaian secara tepat mengenai kemampuan dari pelajar itu sendiri. Bentuk soal berupa pilihan ganda ini menyebabkan pelajar tidak kreatif dalam menjawab pertanyaan karena banyak diantaranya yang hanya menghafal tanpa mengerti akan maksud dari materi yang diujikan tersebut [1]. Selain itu, dengan bentuk soal pilihan ganda ini, memungkinkan para pelajar menjawab soal ujian dengan hanya menerka tanpa mencoba berpikir dahulu.

Sejak tahun 1990-an hingga saat ini Indonesia terlibat dalam tes internasional yang diikuti siswa dari negara-negara maju dan negara berkembang, yakni *Programme for International Student Assesment (PISA)* di bidang membaca, Matematika, dan Sains untuk siswa SMP. Indonesia juga mengikuti *Progress in International Reading Literacy Study (PIRLS)* bidang membaca untuk siswa SD serta *Trends in International Mathematics and Science Study (TIMSS)* bidang Matematika dan Sains untuk siswa SMP. Hasil tes menunjukkan, kemampuan siswa Indonesia di bawah standar internasional. Kemampuan rata-rata siswa Indonesia dalam merespons item format uraian lebih rendah dibandingkan pilihan ganda. Kondisi itu secara umum menunjukkan siswa Indonesia lemah untuk melakukan analisis, prediksi, dan membuat kesimpulan [2].

Dari data tersebut, para peneliti juga mengatakan bahwa sistem ujian yang telah ada perlu diperbaiki untuk mendorong daya kreatifitas dan kemampuan berpikir para pelajar Indonesia. Oleh karena itu, bentuk ujian esai merupakan alternatif yang dapat digunakan untuk mendorong daya kreatifitas dan

kemampuan berpikir serta menalar suatu masalah. Esai disadari oleh para peneliti sebagai cara yang paling berguna untuk menilai hasil dari pembelajaran dimana, menyiratkan kemampuan untuk mengingat, mengatur dan mengintegrasikan ide, serta kemampuan untuk mengekspresikan diri sendiri dalam menulis.

Namun demikian, penggunaan esai tidak lah terlepas dari masalah yang akan ditimbulkan selanjutnya. Para peneliti berpendapat bahwa salah satu kesulitan terbesar yang terdapat pada sistem ujian esai adalah dalam penilaian esai tersebut yaitu tingkat subjektivitas yang dirasakan dari pihak penilai. Para peneliti menyatakan bahwa sifat subjektivitas penilaian esai mengarah pada variasi dari penilaian yang diberikan oleh penilai yang merupakan orang yang berbeda.

Seiring perkembangan teknologi yang begitu pesat akhir-akhir ini, membawa dampak juga terhadap bidang pendidikan. *Essay grading* merupakan salah satu contoh perkembangan teknologi yang terjadi dibidang pendidikan saat ini. *Essay grading* merupakan suatu sistem yang dibangun dengan tujuan untuk memungkinkan melakukan sebuah ujian berbentuk esai dimana penilaian jawaban terhadap esai tersebut dilakukan secara otomatis oleh komputer. Sebelumnya sistem seperti ini telah ada untuk memungkinkan ujian yang berbentuk pilihan ganda dengan penilaian secara otomatis, oleh karena itu dikembangkan sebuah sistem yang memungkinkan sistem penilaian otomatis ini dapat dilakukan dalam bentuk ujian esai.

2.2 Simple-O

Sistem *Essay Grading* dengan metode LSA (*Latent Semantic Analysis)* ini terdiri dari 3 (tiga) modul, yaitu :

1. Modul Login,
2. Modul Dosen,
3. Modul Mahasiswa

2.2.1 Modul Login

Modul login digunakan untuk membedakan siapa yang masuk kedalam sistem, dimana terdapat login sebagai mahasiswa, login sebagai dosen dan login sebagai admin. Masing-masing jenis login ini akan mendapatkan fasilitas yang

berbeda, apabila login sebagai ID dosen maka akan mendapat fasilitas sebagai dosen begitu pula untuk login sebagai mahasiswa akan mendapat fasilitas sebagai mahasiswa. Selain itu, terdapat juga ID root yang berfungsi sebagai super *user* dari sistem ini. Root memiliki kemampuan untuk mendaftarkan *user* baru baik sebagai mahasiswa ataupun dosen sekaligus dapat menentukan mata kuliahnya. Algortiman tampilan untuk modul login ditunjukkan pada Gambar 2.1.

```
prog();
    [Pengecekan idetifikasi user]

if idlogin = 1 then                    {login dosen}
    (
       [tampilan utama untuk dosen]
       if userName = root then
    (
                [tampilkan menu tambahan untuk root]
       }
       read (pil);

    [ke sub modul sesuai pilihan : listing nilai, tampilkan      dan mengisi soal,
    registrasi atau logout]
     )
    if idlogin = 2 then          {login mahasiswa}
    (
       [tampilan utama untuk mahasiswa]
       read (pil);

       [ke sub modul sesuai pilihan : listing nilai, tampilkan dan
        menjawab soal, registrasi atau logout]
    )
 Eprog
```

Gambar 2.1. Algoritma Tampilan Utama dan Pilihan pada Menu Utama

(Sumber: Anak Agung Putri Ratna, 2007)

2.2.2 Modul Dosen

Untuk modul dosen terdiri dari 4 (empat) modul, yaitu :

1. Modul List Nilai

   Merupakan modul untuk melihat nilai dari mahasiswa yang mengambil ujian untuk mata kuliah tersebut

2. Modul Soal

Merupakan modul yang digunakan untuk menambahkan soal yang baru, mengedit soal dan menghapus soal

3. Modul Mata Kuliah

   Merupakan modul untuk melihat mata kuliah yang tersedia dalam sistem

4. Modul Registrasi

   Modul untuk admin untuk dapat melakukan registrasi pada sistem

2.2.3 Modul List Nilai

Dosen dapat melihat nilai dari mahasiswa yang mengikuti ujian pada mata kuliah yang dikelola oleh dosen tersebut dengan modul list nilai ini. Algoritma untuk modul list nilai terdapat pada Gambar 2.2.

```
Proc listnilai ()
        [load nilai_mhs untuk matkul dari database]
        i=0;
        while(score =! EOF)
                i++;
                write ("idmk");
                write ("nama_mhs");
                write ("score");
        ewhile
Eproc
```

Gambar 2.2. Algoritma melihat listing nilai

(Sumber: Anak Agung Putri Ratna, 2007)

2.2.4 Modul Soal

Adapun hal-hal yang dapat dilakukan dalam modul soal ini antara lain :

1. Mengedit soal
2. Menghapus soal
3. Memasukkan atau menambahkan soal
4. Memilih kata bobot

Pada modul soal, dosen lah yang dapat melakukan beberapa hal seperti diatas. Algoritma Global untuk modul soal ditunjukkan oleh Gambar 2.3.

```
Proc soal ()

    [load mata kuliah untuk user dari database, dapat di delete, edit dan
input soal mata kuliah tersebut]
    read (pil);

    if pil = Delete then {bagian untuk menghapus soal}
        [hapus record dari database]
    else

    if pil = Edit then {bagian untuk mengedit soal}

        [mengedit soal]

    if pil = Input Soal then {bagian untuk menambah soal}

        [mengisi soal]
Eproc
```

Gambar 2.3 Algoritma Global untuk modul soal

(Sumber: Anak Agung Putri Ratna, 2007)

Pada modul ini dapat dibuat beberapa fitur guna meningkatkan kinerja dari metode LSA ini. Adapun fitur-fiturnya sebagai berikut.

a. Fitur penambahan bobot dari *keyword.*

   Pada fitur ini, *keyword* akan dipilih sebanyak 2 kali oleh dosen yang bersangkutan. Untuk ujicoba tahapan pertama adalah *keyword* yang biasa, dimana untuk satu jawaban, *keyword* biasa dapat terdiri sari 10 - 20 *keyword*. Untuk percobaan kedua, *keyword* biasa terdiri dari ditentukan oleh minimal 3 dosen yang kompeten. *Keyword* yang kedua adalah *keyword bobot*, yang mencakup hal yang dianggap penting sekali. Pada riset ini *keyword* bobot terdiri dari 5 - 8 kata *keyword* perjawaban untuk percobaan pertama dan untuk percobaan kedua juga ditentukan oleh minimal 3 dosen yang kompeten. *Keyword* biasa memiliki pembobotan matriks 1 sedangkan untuk *keyword* bobot pembobotannya adalah dikali dengan 2. Cuplikan algoritma untuk fitur penambahan nilai dan bobot dari *keyword* ditunjukkan pada Gambar 2.4 dan 2.5.

```
if pil = Input Soal then
                {bagian untuk menambah soal}
    [input soal]
    [input jawaban]
    [input kata kunci]
    [input kata kunci bobot]
    [simpan soal dan kata kunci ke database]
```

Gambar 2.4. Cuplikan Algoritma tambah soal

(Sumber: Anak Agung Putri Ratna, 2007)

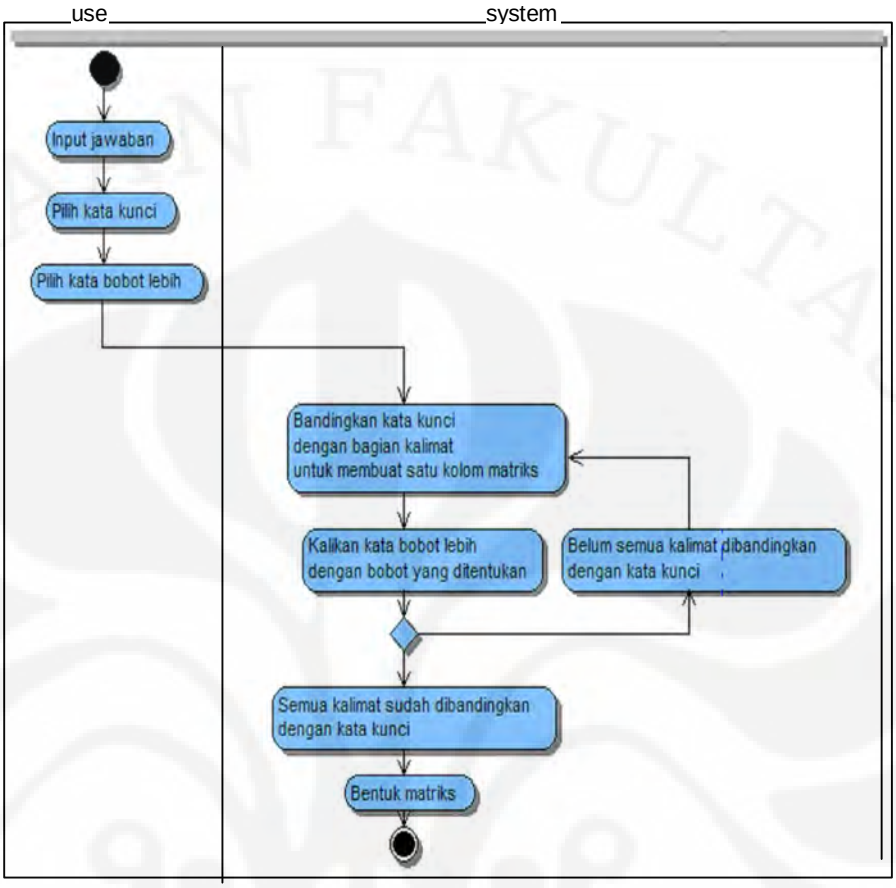
```
if pil = Pilih Kata Bobot then
                {bagian untuk menambah soal}
    [pilih matkul]
    [pilih soal]
    [input kata kunci bobot]
    [bentuk matriks]
    [Proses SVD]
    [Simpan nilai frobenius/cos Alfa yang sesuai]
```

Gambar 2.5. Cuplikan Algoritma pembobotan

(Sumber: Anak Agung Putri Ratna, 2007)

b. Fitur persamaan kata

Pada fitur persamaan kata, kata yang sama dan memiliki arti yang sama akan dianggap sama. Persamaan ini berdasarkan tabel yang dibentuk seperti pada contoh Tabel 2.1. *Activity diagram* konversi matriks dan pembobotan ditunjukkan pada Gambar 2.6



Gambar 2.6. *Activity diagram* konversi matriks dan pembobotan

(Sumber: Anak Agung Putri Ratna, 2007)

Tabel 2.1. Persamaan Kata

no	kata	kata dasar	kode kata	kode persamaan
1	memiliki	milik	1	1
2	mempunyai	punya	2	1
3	berkesempatan	sempat	0	2
4	bisa	bisa	9	2
5	dapat	dapat	9	2
6	kemungkinan	mungkin	0	2
7	mampu	mampu	0	2
8	mungkin	mungkin	0	2
9	butuhkan	butuh	3	3
10	dibutuhkan	butuh	3	3
11	diperlukan	perlu	3	3

Tabel 2.1. Persamaan Kata (lanjutan)

no	kata	kata dasar	kode kata	kode persamaan
12	Perlukan	perlu	3	3
13	Butuh	butuh	2	4
14	membutuhkan	butuh	2	4
15	memerlukan	perlu	2	4
16	Perlu	perlu	2	4
17	kebutuhan	butuh	1	5
18	keperluan	perlu	1	5
19	Dimiliki	milik	2	6
20	Dipunyai	punya	3	6
21	menambah	tambah	2	7
22	menambahkan	tambah	2	7
23	menjumlah	jumlah	2	7
24	menjumlahkan	jumlah	2	7
25	Modem	modem	0	8
26	connector	connect	0	9
27	Konektor	konektor	0	9
28	penyambung	sambung	0	9
29	Dijumlah	jumlah	3	10
30	dijumlahkan	jumlah	3	10
31	ditambah	tambah	3	10
32	ditambahkan	tambah	3	10
33	ditingkatkan	tingkat	3	10
34	penambahan	tambah	1	11
35	penjumlahan	jumlah	1	11
36	penambah	tambah	1	12
37	penjumlah	jumlah	0	12
38	Standar	standar	1	13
39	Standard	standard	0	13
40	Standart	standar	1	13
41	Kirim	kirim	0	14
42	penghantaran	hantar	0	14
43	Pengirim	kirim	0	14
44	pengiriman	kirim	1	14
45	pengirimannya	kirim	0	14
46	pentransferan	transfer	1	14

(Sumber: Anak Agung Putri Ratna, 2007)

Tabel 2.1. Persamaan Kata (lanjutan)

no	kata	kata dasar	kode kata	kode persamaan
47	Sender	send	0	14
48	Sending	send	0	14
49	Transfer	transfer	0	14

Kolom pertama pada Tabel 2.1, menunjukkan kolom no yang merupakan urutan kata. Kolom kedua adalah kolom kata yang mungkin digunakan oleh mahasiswa untuk menjawab ujian, dimana semua kata ini dapat berbentuk kata kerja transitif, intransitif dan sebagainya. Kolom ketiga merupakan kolom kata dasar, dimana semua kata yang berada didalam kolom kedua akan dikembalikan kedalam bentuk kata dasar. Kolom keempat adalah kolom kode kata, dimana semua kata dikategorikan pada daftar dibawah ini.

a. Kata benda : 1.
b. Kata kerja aktif : 2.
c. Kata kerja pasif : 3.
d. Kata sifat : 4.
e. Kata keterangan : 5.
f. Benda satuan : 6.
g. Kata majemuk : 7.
h. Ajektif (kata keterangan) : 8.
i. Adverb (kata sifat) : 9.
j. Kata sambung : 10.

Untuk sistem yang saat ini dikembangkan, kolom ke 4 tidak dipergunakan. Kolom kelima adalah kolom untuk menyatakan kode persamaan kata.

Sistem kerja pada fitur persamaan kata ini adalah sebagai berikut. Pertama-tama, kalimat yang dimasukkan oleh mahasiswa sebagai jawaban dalam ujian akan diuraikan menjadi kata-kata tersendiri. Kata-kata tersebut kemudian disesuaikan dengan kata kunci jawaban referensi dosen yang telah dimasukkan dalam sistem sebelumnya. Kemudian, kata-kata tersebut akan dicek kedalam database persamaan kata. Bila terdapat kata yang kode persamaannya sama, maka

kata tersebut akan diproses sama dengan kata kunci yang ada pada jawaban referensi dosen. Algoritma untuk modul ini ditunjukkan pada Gambar 2.7.

```
Proc ujian ()
        [load matkul untuk user dari database]
        while matkul =! EOF
                [tampilkan mata kuliah yang dipilih]
        ewhile
        read (pil);
        if pil = Back then
                goto Halaman_Muka;
        else

        if pil = Lihat Soal then
                write ("Soal", soal[ ]);
                write ("Jawaban");
                read jawab_mhs[ ];
                if submit = true then
                        [cek persamaan kata-kata jawaban
         mahasiswa]
                        [bentuk matriks dari jawaban mahasiswa];
                        [bandingkan matriks jawaban dengan
        referensi];
                        [kirim nilai ke tabel database];
                else
                ( );
        else
        ( );
Eproc
```

Gambar 2.7. Algoritma pengecekan persamaan kata

(Sumber: Anak Agung Putri Ratna, 2007)

2.2.5 Modul Mahasiswa

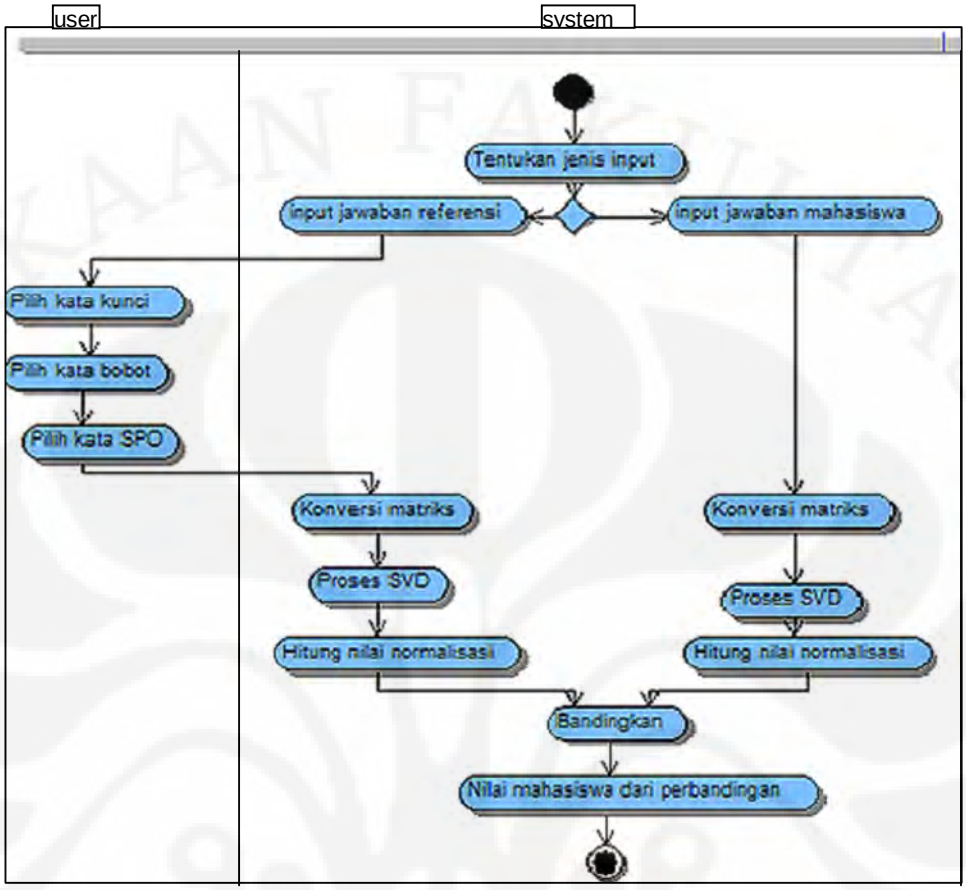
Terdapat dua modul pada modul mahasiswa ini, yaitu modul lihat nilai dan modul ujian. Modul lihat nilai digunakan oleh mahasiswa untuk melihat nilai ujiannya, dimana mahasiswa hanya dapat mengikuti ujian satu kali. Algoritma untuk mengikuti ujian yang dikembangkan dengan metode LSA ditunjukkan pada Gambar 2.8.

```
Proc ujian ()
        [load matkul untuk user dari database]
        i = 0;
        write (“Mata Kuliah”);
        while matkul =! EOF
                i++;
                write ["matkul"];
                write (“Lihat Soal”);
                write (“Back”);
        ewhile
        read (pil);
        if pil = Back then
                goto Halaman_Muka;
        else

        if pil = Lihat Soal then
                write (“Soal”, soal[ ]);
                write (“Jawaban”);
                read jawab_mhs[ ];
                if submit = true then
                        [cek persamaan kata-kata jawaban mahasiswa]
                        [bentuk matriks dari jawaban mahasiswa dengan pembobotan
                        pada kata kunci bobot];
                        [bandingkan matriks jawaban dengan referensi,dengan
                        membandingkan nilai normalisasi frobenius dan nilai kosinus
                        alfa];
                        [kirim nilai ke tabel database];
                else
                ( );
        else
        ( );
Eproc
```

Gambar 2.8. Algoritma mengikuti ujian dengan metoda LSA yang dikembangkan

(Sumber: Anak Agung Putri Ratna, 2007)



Gambar 2.9. *Activity diagram* mengikuti ujian dengan metoda LSA yang dikembangkan

(Sumber: Anak Agung Putri Ratna, 2007)

2.3 *Essay Grading* dengan *Latent Semantic Analysis (LSA)*

Perkembangan dibidang *essay grading* masih terus berlanjut, banyak metode yang digunakan dalam *essay grading* ini. Setiap metode yang digunakan pada *essay grading* memiliki teknik penilaian yang berbeda. Meskipun teknik penilaian yang digunakan berbeda-beda, namun sistem ini memiliki tujuan yang sama yaitu membangun sebuah sistem yang dapat memberikan penilaian terhadap jawaban esai seobjektif mungkin. Salah satu metode yang dipakai pada *Essay Grading* adalah *Latent Semantic Analysis*. *Latent Semantic Analysis (LSA)* merupakan representasi terhadap jumlah dan kebolehjadian kata untuk dibandingkan secara geometris (matrik).

SVD (*Singular Value Decomposition)* merupakan bagian penting dalam pemrosesan LSA dimana SVD ini mengkompresi informasi berkaitan dalam

jumlah besar ke dalam ruang yang lebih kecil. LSA akan merepresentasikan isi kata dalam matriks dua dimensi yang besar. Dengan menggunakan SVD yang merupakan teknik aljabar matriks, hubungan baru antara kata dan dokumen ditentukan dan dimodifikasi untuk mewakili arti sebenarnya. Tiap kolom merepresentasikan tiap analasis kata sedangkan tiap baris mewakili kalimat, paragraf, dan sub divisi lainnya yang berkaitan.

Menurut Landauer, LSA telah diujikan dengan lima skema penilaian dengan perlakuan yang berbeda masing-masing, terutama berkaitan dengan vektor yang dikomputasi, dimana esai mahasiswa dibandingkan dengan esai referensi. Hasil penilaian yang didapat dari LSA hampir sama handalnya dengan hasil penilaian manusia. Dari tes esai pada GMAT, kesepakatan antara penilaian manusia dan sistem LSA berkisar antara 85% sampai 91%.

2.4 *Singular Value Decomposition* (SVD)

Teknik SVD digunakan untuk melakukan estimasi atau perkiraan struktur dalam penggunaan kata dalam dokumen-dokumen. Pada dasarnya SVD merupakan teknik untuk melakukan estimasi *rank* dari matriks. Jika diketahui matriks A dengan dimensi m x n, dimana nilai $m \geq n$ dan rank (A) = r maka *singular value decomposition* dari A, dinotasikan sebagai SVD (A), didefinisikan melalui persamaan

$$A = U \sum V^T$$

dimana

$$U^T U = V^T V = I_n$$

dan memenuhi kondisi

$$\sum = \text{diag}\,(\sigma_1,...,\sigma_n)$$

dimana

$\sigma_i > 0$ untuk $1 \leq i \leq r$

$\sigma_j > 0$ untuk $j \leq r + 1$

Kolom r pertama dari matriks U dan V mendefinisikan vektor eigen orthonormal yang bersesuaian dengan r nilai vektor eigen tidak-nol dari matriks

AA^T dan A^TA berturut-turut. Kolom dari matriks *U* dan *V* berisi vektor, masing-masing disebut vektor singular kiri dan kanan.

Nilai singular dari *A* merupakan elemen diagonal dari matriks $\sum$, dimana nilai singular didapat dari akar pangkat dua dari nilai absolur dari sejumlah *n* nilai eigen dari AA^T [3].

2.5 Program yang Mendukung LSA

Metode LSA yang kini telah dikembangkan pada otomasi *essay grading* merupakan suatu sistem terpadu yang dibangun dengan bahasa *scripting* PHP dan HTML serta dengan menggunakan database MySQL. Selain dukungan dari tiga program tersebut, metode LSA ini juga menggunakan program pendukung lainnya yaitu modul JAMA. *Web server* yang digunakan dalam pembuatan sistem ini adalah Apache. Selanjutnya, untuk dapat mengakses sistem yang dibuat ini dapat dilakukan dengan menggunakan berbagai web browser seperti, Internet Explorer, Mozzila Firefox, Opera maupun web browser lainnya.

2.5.1 PHP

PHP atau *Hypertext Preprocessor* merupakan bahasa *scripting* yang diletakkan dalam server yang umumnya digunakan untuk membuat aplikasi berbasis web yang dinamis. Sebagian besar perintah yang digunakan dalam PHP berasal dari C, Java dan Pearl. Penggunaan PHP memungkinkan para pembuat aplikasi web menyajikan halaman HTML yang dinamis, dimana halaman web nya menjadi lebih interaktif.

PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995 dimana saat itu bentuknya merupakan sekumpulan *script* yang digunakan untuk mengolah data form dari web. Perkembangan PHP terus berlanjut dan dilakukan oleh programmer lain karena sifatnya yang merupakan *open source software*.

Adapun kelebihan PHP antara lain :

1. PHP didukung oleh banyak web server seperti apache, IIS, Lightpd dan lainnya.
2. PHP dapat digunakan dalam berbagai macam operating system seperti Linux, Unix, Windows dan lainnya.

2.5.1.1 Tipe Data String

Tipe data string dinyatakan dengan menggunakan tanda petik tunggal (‘ ‘) ataupun tanda petik ganda (“ “). Perbedaannya terletak dimana apabila menggunakan tanda petik tunggal maka apabila dalam sebuah *variable* memiliki sebuah data string, maka yang akan dicetak adalah nama variable itu sendiri. Adapun beberapa string yang digunakan pada program untuk optimasi kinerja pencarian kata ini adalah sebagai berikut :

1. Explode ()

   Explode memiliki fungsi untuk memecah suatu string menjadi beberapa bagian dengan pola tertentu.

2. Substring ()

   Substring memiliki fungsi untuk memotong suatu string dengan menentukan posisi awal pemotongan dan jumlah karakter yang akan dicetak.

   Pengaturan string ini ditentukan berdasarkan index, karakter pertama dari suatu string ber-index 0, karakter kedua ber-index 1 dan begitu seterusnya. Dengan penggunaan substring ini, kita dapat memanipulasi data string yang kita gunakan, dimana dalam sebuah kata kita dapat mendeteksi awalan atau beberapa potong karakter dalam kata tersebut dengan tujuan yang kita kehendaki.

   Bentuk umum penulisan :

   Substr (variable, posisi awal pemotongan, jumlah karakter yang dicetak);

   Contoh :

   ```
   <?
   $kata=”belajar”;
   echo substr($kata,0,2);
   ?>
   ```

   Kata yang akan dicetak apabila menggunakan substring seperti diatas adalah “be”.

2.5.2 MySQL

MySQL merupakan software yang tergolong sebagai DBMS *(Database Management System)* yang bersifat *Open Source*. MySQL pada awalnya dibuat di Swedia oleh perusahaan konsultan bernama TcX. Selanjutnya pengembangan MySQL berada dibawah naungan perusahaan MySQL AB. Sun Microsystems, Inc kemudian mengumumkan aksi korporasi – akusisi terhadap MySQL AB pada tanggal 16 Januari 2008 sehingga menjadikan Sun sebagai salah satu perusahaan dengan produk platform open source terbesar seperti Java, OpenSolaris dan akhirnya MySQL. MySQL sendiri merupakan sistem manajemen database relasional, dimana database ini menyimpan data dalam tabel terpisah.

Berselang setahun kemudian, tepatnya pada tanggal 20 April 2009 giliran Oracle melakukan akusisi terhadap Sun Microsystems.

Sebagai software DBMS, MySQL memiliki beberapa fitur, diantaranya :

1. Multiplatform

   MySQL dapat digunakan dalam banyak paltform, seperti Windows, Linux, Unix dan platform lainnya.
2. Multiuser

   MySQL dapat digunakan oleh beberapa user dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
3. Perintah dan Fungsi

   MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (*query*).
4. Antar Muka

   MySQL memiliki interface (antar muka) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (Application Programming Interface).
5. Andal, cepat dan mudah digunakan

   MySQL tergolong sebagai *database server* yang andal, dapat menangani database yang besar dengan kecepatan tinggi, mendukung banyak fungsi untuk mengakses database, dan sekaligus mudah untuk digunakan.
6. Jaminan keamaanan akses

MySQL mendukung pengamanan database dengan berbagai kriteria pengaksesan. Contohnya, pada MySQL dapat dimungkinkan untuk mengatur user tertentu agar bisa mengakses data yang bersifat pribadi dimana user lainnya tidak dapat mengakses data tersebut.

7. Dukungan SQL

   MySQL mendukung perintah SQL *(Structured Query Language)*.

SQL *(Structured Query Language)* merupakan bahasa yang yang digunakan untuk mengakses database relasional. Secara umum, database memiliki arti kumpulan data yang saling terkait, dimana penyusunan data ini dibuat terstruktur dan disimpan dalam sebuah mesin penyimpan seperti *hard disk* dengan tujuan agar data tersebut dapat diakses dengan cepat dan mudah. Terdapat berbagai macam database, antara lain, database hierarkis, database jaringan dan database relasional. Sebuah database relasional tersusun atas sejumlah tabel, contohnya yaitu database akademis mencakup tabel-tabel seperti dosen, mahasiswa, nilai-nilai dan lain-lain.

Tidak semua fitur SQL didukung oleh vendor perangkat lunak, namun MySQL sebagai database server mendukung perintah SQL. MySQL juga menambahkan beberapa fungsi tambahan yang membuat perintah SQL pada MySQL sangat variatif. Contoh penambahan fungsi tersebut yaitu pengambilan jam yang berlaku pada saat sekarang pada server.

Perintah yang dapat dipahami oleh database server MySQL disebut dengan istilah pernyataan. Pernyataan merupakan sebuah perintah yang dapat dikerjakan oleh MySQL dengan ciri-ciri diakhiri dengan tanda titik koma (;).

2.5.3 Apache

HTTP Apache server merupakan web server multiplatform dimana dapat dijalankan diberbagai *operating system* seperti Unix, Linux, Windows maupun *operating system* lainnya. Apache server ini berguna untuk melayani dan mengfungsikan web dimana protokol yang digunakan yaitu HTTP.

Berbagai macam fitur yang disediakan oleh *apache server* diantaranya pesan kesalahan yang dapat dikonfigurasi serta autentikasi berbasis database dan

lain-lain. Interface yang digunakan yang berupa GUI memungkinkan penanganan server menjadi lebih mudah di apache. Apache bebas diretribusikan kepada siapa saja karena bersifat *open source* dimana mulai populer pada tahun 1996. Agar dokumen-dokumen web baik yang berupa HTML maupun PHP dapat diakses oleh browser maka dokumen-dokumen tersebut diletakkan dalam direktori khusus yang diatur oleh Apache.

2.5.4 Java Matrix (JAMA)

Pada awalnya program matlab digunakan untuk dapat melakukan perhitungan SVD. Namun, pada saat sistem dijalankan, waktu yang dibutuhkan untuk melakukan pehitungan SVD untuk satu orang pada matlab sangatlah besar. Hal ini tentu saja menjadi kendala bagis sistem otomasi *essay grading* yang akan digunakan untuk banyak user. Oleh karena itu, dibutuhkan sebuah program perhitungan yang lebih cepat dan lebih baik pada sistem.

Sebagai pengganti Matlab, digunakan program JAMA (Java Matrix) yang mampu melakukan perhitungan lebih cepat dibandingkan matlab. Java Matrix yang digunakan merupakan skrip PHP untuk perhitungan matriks kompleks.

BAB 3
PERANCANGAN METODE SEARCHING PADA SIMPLE-O

3.1 Database

Penggunaan database sangatlah luas dalam berbagai bidang. Beberapa contoh penggunaan database dalam berbagai bidang tersebut antara lain :

1. Perbankan : untuk informasi nasabah, rekening, pinjaman serta transaksi bank lainnya.
2. *Airlines* : untuk informasi pemesanan dan jadwal penerbangan.
3. Pendidikan : untuk informasi pelajar, registrasi pendidikan serta informasi tingkatan.
4. Telekomunikasi : untuk menyimpan data panggilan telekomunikasi, menghitung biaya bulanan dan menyimpan segala informasi mengenai jaringan komunikasi.
5. *Manufacturing* : untuk manajemen persediaan barang, pelacakan produksi barang dalam pabrik, persediaan barang di gudang atau toko, dan pemesanan barang.
6. Sumber daya manusia : untuk informasi mengenai pegawai, gaji, pajak gaji dan lainnya.

Berdasarkan ilusrtasi diatas, dapat diketahui bahwa database sudah banyak digunakan dalam berbagai bidang saat ini. Adapun pengertian database dari berbagai peneliti, antara lain :

1. Menurut Gordon C. Everest (1974), database adalah koleksi atau kumpulan data yang mekanis, terbagi, terdefinisi secara formal dan dikontrol terpusat pada organisasi.
2. Menurut C.J Date (1981), database adalah koleksi “data operasional” yang tersimpan dan dipakai oleh sistem aplikasi dari suatu organisasi. Data operasional sendiri merupakan data yang tersimpan pada sistem.
3. Menurut Toni Fabbri (1992), database adalah sebuah sistem file-file yang terintegrasi yang mempunyai niminal primary key untuk pengulangan data.

Jadi, database atau basis data adalah kumpulan data atau informasi yang tersimpan dalam suatu sistem. Pengertian database dalam bidang ilmu komputer adalah kumpulan informasi atau data yang disimpan dalam sebuah komputer secara sistematis dimana data atau informasi tersebut dapat dikelola dan dipanggil kembali dengan menggunakan sebuah program komputer yang dikenal dengan sistem manajemen basis data (*database management system*, DBMS).

3.1.1 Database Sistem *versus* File Sistem

Sebelum adanya sistem database, penggunaan *file-processing system* menjadi cara yang paling banyak dihandalkan dalam mengolah dan menyimpan data. *File-processing system* ini sendiri umumnya merupakan sebuah program atau aplikasi yang dibuat untuk membantu mengolah dan menyimpan data untuk sebuah kepentingan dan biasanya untuk kepentingan yang berbeda maka akan membutuhkan program yang berbeda pula. Contohnya dalam *essay grading ini*, apabila ditambahkan sebuah mata kuliah baru dalam sistem ini, maka akan dibuat sebuah program baru yang berisikan mengenai mata kuliah baru ini. Hal ini mungkin terjadi karena setiap mata kuliah memiliki atribut yang berbeda, seperti dosen pengajar dan isi dari ujian tersebut.

Penggunaan *file-processing system* memiliki beberapa kekurangan, antara lain :

1. Pengulangan dan Inkonsistensi Data.

   Pengulangan dan inkonsistensi data sangat mungkin terjadi dalam *file-processing system* ini. Pengulangan data dapat mengarah kepada penggunaan sistem penyimpanan yang lebih besar serta peningkatan biaya akses. Hal ini dapat terjadi dikarenakan beberapa hal, seperti informasi alamat dan nomor telepon yang akan muncul dalam dua aplikasi yang berbeda, contohnya adalah munculnya dua informasi tersebut dalam setiap aplikasi mata kuliah yang berbeda. Inkonsistensi data dapat terjadi dikarenakan perubahan suatu informasi pada sebuah aplikasi dimana pada aplikasi lainnya yang mengandung informasi yang sama tidak dilakukan perubahan data.

2. Akses Data Sulit

   Anggap bahwa seorang petugas universitas menginginkan informasi mengenai nama-nama mahasiswa atau dosen yang tinggal dalam sebuah daerah tertentu. Untuk dapat menemukan informasi yang diperlukan, diperlukan sebuah aplikasi yang dapat mencari informasi tersebut dengan cepat, namun perancangan aplikasi tidak mengantisipasi hal ini dikarenakan jenis permintaan yang bersifat spesifik. Oleh karena itu, untuk dapat menghasilkan informasi yang dibutuhkan diperlukan sebuah aplikasi tambahan untuk memenuhi kebutuhan tersebut. Hal ini tentu saja akan memakan waktu dan biaya lebih.

3. Masalah Keamanan

   Tidak setiap pemakai sistem database harus dapat mengkases semua data. Sebagai contoh, bagian mahasiswa hanya perlu mengakses bagian database yang berhubungan dengan mahasiswa tanpa dapat melihat bagian database dosen.

3.1.2 Abstraksi Data

Para pengembang dibidang sistem database menggunakan beberapa tingkatan abstraksi guna memudahkan para pengguna sistem database untuk dapat berinteraksi dengan database [4].

1. *Physical Level*

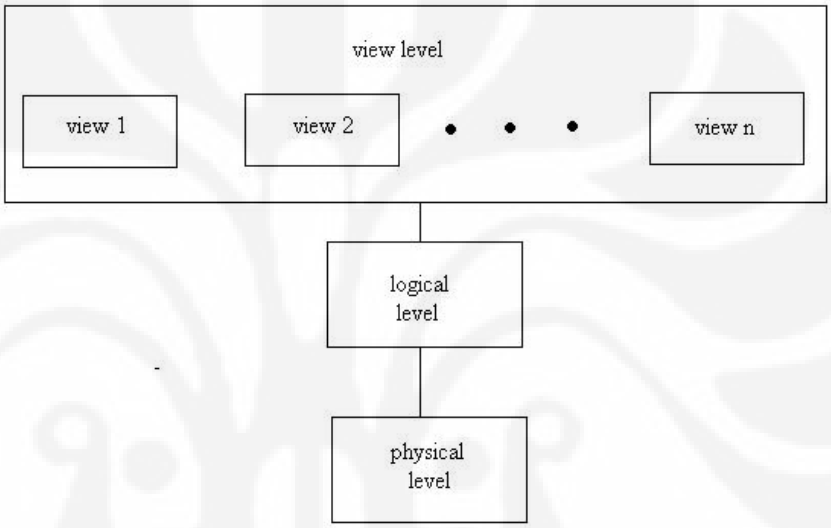
   Tingkat terendah abstraksi data dimana menggambarkan tentang bagaimana data tersebut sebenarnya disimpan.

2. *Logical Level*

   Tingkatan selanjutnya dari abstraksi data dimana menggambarkan data apa saja yang disimpan dalam database, dan hubungan antara data-data tersebut. *Logical level* juga digunakan oleh administrator database untuk menentukan informasi apa saja yang perlu dijaga dalam database.

3. *View Level*

   Tingkatan tertingi abstraksi data dimana hanya menggambarkan sebagian dari keseluruhan database. Para pengguna database tidak memerlukan informasi mengenai bagaiamana data disimpan serta apa saja data yang disimpan, oleh karena itu, pada tingkatan ini para pengguna hanya perlu mengakses sebagian dari database guna menyederhanakan interaksi mereka dengan sistem.

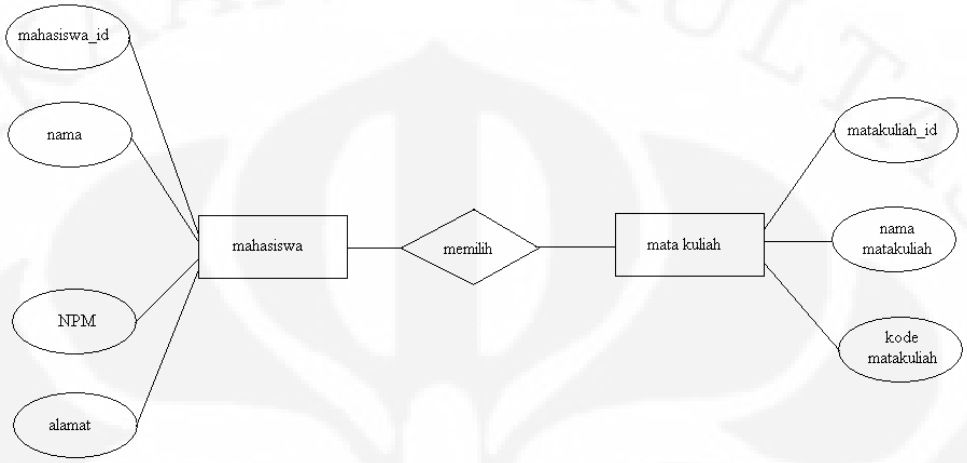


Gambar 3.1 Tiga tingkatan abstraksi data

3.1.3 *Entity-Relationship Model (E-R Model)*

Entity-relationship data model representasi logika dari data yang mengandung kumpulan objek dasar, yang disebut sebagai entitas (*entity)* dan hubungan (*relationships*) diantara objek tersebut. Entitas adalah suatu "benda" atau "objek" yang dapat dibedakan dari objek lainnya. Seperti dalam *essay grading* ini, mahasiswa adalah suatu entitas. Entitas dapat dideskripsikan dalam dalam suatu database dari sekumpulan atribut. Sebagai contoh, alamat, NPM (Nomor Pokok Mahasiswa), nomor telepon adalah atribut dari suatu entitas mahasiswa. Selain itu, umumnya ditambahkan extra atribut seperti id mahasiswa untuk menghindari kesamaan nama dari mahasiswa dalam database.

Relationship (hubungan) adalah asosiasi diantara beberapa entitas. Contoh, entitas mahasiswa memiliki *relationship* dengan entitas mata kuliah dapat terlihat pada gambar 3.2.



Gambar 3.2 *Entity-Relationship Model*

3.2 Database dalam *Essay Grading*

Dalam *essay grading*, banyak informasi yang perlu disimpan dalam suatu komputer secara sistematis. Oleh karena itu, digunakanlah sistem database untuk menyimpan informasi-informasi tersebut. Informasi-informasi yang disimpan dalam database ini akan disimpan dalam bentuk tabel-tabel dimana antara satu tabel dengan tabel lainnya memiliki hubungan *(relationships)*. Adapun informasi-informasi yang tersimpan dalam database pada *essay grading* ini antara lain :

1. Tabel User

   Berisikan semua data tentang user yang dapat mengakses database tersebut, seperti id *user*, nama *user*, *e-mail user*, mata kuliah yang diajar atau diambil.

2. Tabel Grup

   Berisikan pembagian grup untuk mahasiswa dan dosen.

3. Tabel Mata Kuliah

   Berisikan nama-nama mata kuliah yang diujikan.

4. Tabel Persamaan Kata

   Berisikan kata-kata yang mungkin digunakan mahasiswa maupun dosen dalam *essay grading* dan padanan katanya.

5. Tabel Jawaban Mahasiswa

   Berisikan jawaban mahasiswa setelah melalui serangkaian proses pengambilan kata kunci yang akan digunakan untuk melakukan perhitungan nilai.

6. Tabel Soal dan Jawaban

   Berisikan soal-soal ujian yang dibuat oleh dosen serta jawaban-jawaban yang telah melalui proses pembobotan nilai.

7. Tabel Nilai

   Berisikan informasi mengenai hasil atau nilai dari ujian yang diperoleh mahasiswa setelah melalui proses perhitungan dengan menggunakan *Java Matrix*.

3.3 Permasalahan dalam Database

Hal yang masih menjadi kendala dalam proses sistem *essay grading* ini yang kaitannya dengan database, yaitu banyaknya kemungkinan kata yang memiliki persamaan arti yang digunakan dalam menjawab pertanyaan berbentuk essai.

3.3.1 Persamaan Kata

Salah satu kesulitan dalam proses penilaian otomasi dalam ujian esai adalah banyaknya kemungkinan kata yang memiliki persamaan arti yang mungkin digunakan oleh mahasiswa dalam menjawab pertanyaan berbentuk esai. Contohnya, dalam penggunaan kata “esai”, terdapat banyak kemungkinan penulisan kata “esai” tersebut, seperti “*essay”*, “esay”, “essai” tergantung dari pemikiran mahasiswa tersebut dalam penulisan kata “esai”. Selain itu, hal ini juga dapat terjadi dalam kasus lain, seperti penggunaan bahasa Inggris dalam menjawab pertanyaan.

Penggunaan kalimat dengan bahasa Inggris bukan hal yang jarang terjadi lagi, seperti penggunaan kata “*networking”* untuk menggantikan kata jaringan,

maupun "*database*" untuk menggantikan basis data. Banyaknya variasi kata-kata yang dimiliki dan sering digunakan oleh mahasiswa dalam menjawab pertanyaan dapat diatasi dengan penggunaan tabel persamaan kata dimana setiap kata akan diberi sebuah nilai untuk mengidentifikasi kata-kata lainnya yang memiliki arti kata yang sama. Contohnya adalah, pemberian nilai kode persamaan 1 terhadap kata "memiliki" dan padanan katanya "menyerupai".

Banyaknya kata-kata yang tersimpan dalam satu tabel persamaan kata yang mencapai ribuan record akan memperlambat proses sistem *Essay Grading*, dimana proses pencarian kata pada tabel persamaan kata akan terasa lebih lambat karena banyaknya kata yang harus diperiksa.

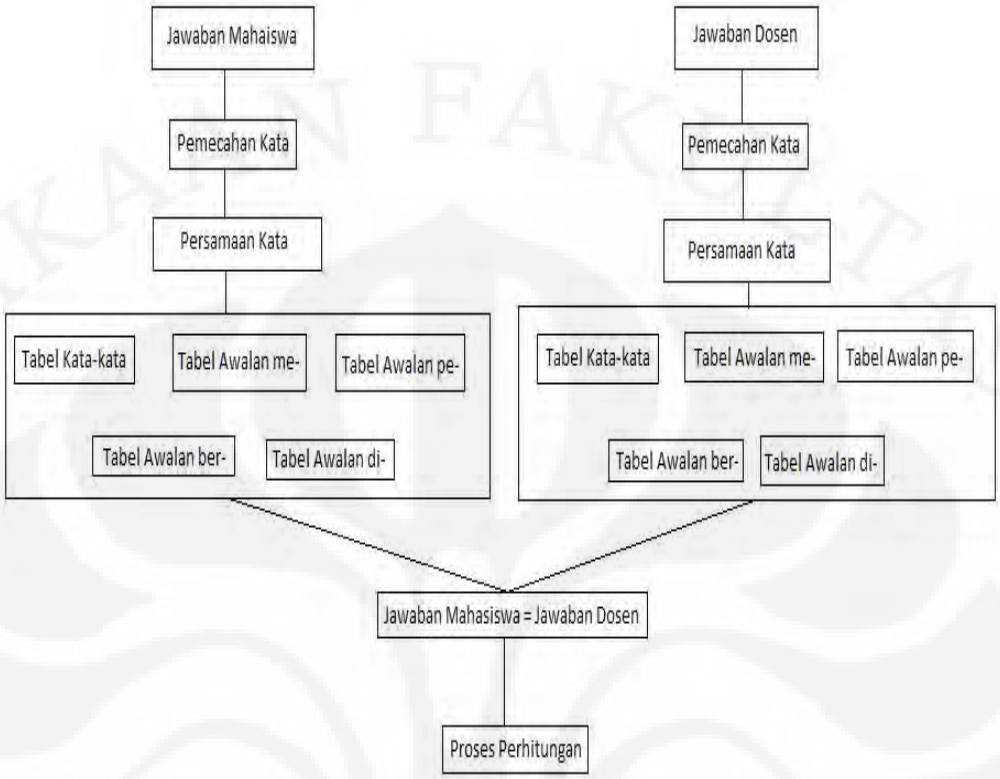
3.4 Rancangan Optimasi Metode Searching Persamaan Kata

Pewaktuan proses penilaian sistem *essay grading* ini salah satunya merupakan pengaruh dari proses pencarian persamaan kata dalam database. Pada sistem *essay grading* ini, hanya digunakan satu tabel persamaan kata untuk menampung jumlah data yang mencapai ribuan kata. Penggunaan satu tabel akan terasa cukup efektif apabila jumlah kata yang terdapat dalam tabel tersebut tidak lah terlampau banyak. Oleh karena itu, penulis mencoba melakukan cara agar mempercepat proses pencarian kata dalam tabel persamaan kata dengan memecah tabel persamaan kata menjadi 5 buah tabel tergantung dari banyaknya kemungkinan kata yang memiliki persamaan karakter/huruf sehingga dapat disatukan kedalam sebuah tabel.

Proses pemecahan tabel ini dirasa merupakan cara yang cukup efektif untuk meningkatkan performansi pencarian kata dalam database. Berikut merupakan gambaran perbandingan dari penggunaan satu tabel dibandingkan dengan penggunaan beberapa tabel.

Gambar 3.3. Proses Pencarian Kata Satu Tabel

Gambar 3.3. merupakan gambaran sistem awal yang dikembangkan dimana proses pencarian kata untuk mencari padanan atau persamaan kata dilakukan pada satu tabel, dimana seluruh jawaban mahasiswa maupun jawaban dosen yang telah dipecah menjadi kata-kata akan dicari padanannya pada tabel persamaan kata. Dengan jumlah data yang banyak, proses pencarian kata ini akan menghabiskan waktu yang cukup lama, sehingga keseluruhan sistem *essay grading* pun akan memakan waktu yang cukup lama. Setelah proses pencarian kata dilakukan, kata-kata tersebut akan dirubah bentuknya menjadi angka yang telah ditetapkan dalam tabel guna proses perhitungan nilai dalam bentuk matriks. Angka-angka ini lah yang mengidentifikasikan kata-kata yang memiliki persamaan arti, dimana kata yang memiliki persamaan arti akan memiliki nilai yang sama.



Gambar 3.4. Proses Pencarian Kata dengan beberapa tabel

Untuk meningkatkan proses pencarian persamaan kata, penulis mencari cara yang dirasa efektif guna meningkatkan proses kerja sistem *essay grading*. Cara yang dirasa efektif tersebut adalah dengan memetakan kata-kata berdasarkan penggunaan huruf awal pada sebuah kata yang banyak digunakan. Dalam hal ini, penulis membagi tabel persamaan kata dengan jumlah data yang mencapai ribuan menjadi 5 buah tabel utama. Pembagian ini didasarkan pada banyaknya huruf yang sama yang terdapat pada sebuah kata, dimana jumlah kata yang memiliki huruf tersebut cukup banyak dan sering digunakan. Pembagian kata ini tidak mengacu pada imbuhan menurut bahasa Indonesia yang sebenarnya.

Adapun tabel yang digunakan adalah tabel me-, pe-, di-, ber- da tabel kata-kata. Tabel me- merupakan tabel yang berisi kata-kata dengan dua huruf awal mengandung “me”, tabel pe- beiriskan kata-kata dengan dua huruf awal “pe”, tabel di berisikan kata-kata dengan dua huruf awal “di”, tabel ber berisikan kata-kata dengan tiga huruf awal “di” dan tabel kata-kata merupakan tabel yang berisi

kata-kata diluar kata-kata yang mengandung huruf “me”, “pe”, “di” serta “ber”. Gambaran sistem ini dapat terlihat pada gambar 3.4.

Cara kerja proses pencarian kata ini adalah, sistem akan mendeteksi semua kata-kata yang masuk baik dari jawaban mahasiswa maupun jawaban dosen dimana pendeteksian ini menggunakan fungsi PHP yaitu fungsi substring. Fungsi ini akan mengenali huruf atau karakter dari sebuah kata yang kita kehendaki. Pada sistem ini, dibuat sebuah fungsi dimana kata-kata yang masuk akan langsung diperiksa apakah mengandung huruf-huruf dengan awalan tersebut atau tidak. Ketika fungsi ini tidak menemukan huruf-huruf tersebut, maka secara otomatis sistem akan mencari kata yang dimaksud dalam tabel kata-kata. Namun apabila sistem menemukan kata yang memiliki karakter yang telah dikhususkan tadi, maka fungsi ini akan mengirimkan kata tersebut untuk dicari padanannya pada tabel yang sesuai dengan awalan huruf kata tersebut. Sebagai contoh, apabila kata yang dideteksi adalah kata “memiliki”, maka sistem akan mendeteksi penggunaan huruf me-, sehingga sistem akan mencari padanan kata tersebut pada tabel me-, dan begitu seterusnya.

Secara umum, proses ini seharusnya dapat menghasilkan kinerja yang lebih cepat dibandingkan dengan proses yang lama, karena kata-kata yang akan dicari sudah dipetakan menurut penggunaannya sehingga sistem dapat langsung mengetahui kemana harus mencari padanan kata tersebut tanpa harus menelusuri sebuah tabel dengan jumlah data ribuan.

Dengan cara ini, diharapkan proses pencarian persamaan kata akan mengalami peningkatan sehingga akan berpengaruh pada proses kinerja database secara menyeluruh pada sistem *essay grading*.

BAB 4
ANALISA METODE SEARCHING PERSAMAAN KATA PADA SIMPLE-O

4.1. Uji Coba Aplikasi

Aplikasi ini merupakan aplikasi berbasis web, sehingga dalam pelaksanaannya diperlukan *web server*, dimana tentu saja dalam *web server* tersebut dibutuhkan PHP *engine* dan sistem database. Pengambilan data ini akan dilakukan pada sebuah komputer dengan spesifikasi sebagai berikut :

Motherboard : ECS 945GZT-M v: 1.0
Prosesor : Pentium(D) 2.80 GHz
Memory (RAM) : 2 GB
Sistem Operasi : Ubuntu 10.4
Web Server : Apache 2.2.14
Sistem Basis Data : MySQL 5.1.41
PHP Engine : PHP 5.3.2

Proses pengujian dilakukan dengan tiga tahap berbeda, yaitu :

1. Mengambil sample dari pencarian satu kata
2. Mengambil sample dari soal ujian, dimana akan diambil data pertahap dengan melihat banyaknya jumlah kata yang akan dicari. Dalam hal ini, akan diambil sample dari penggunaan satu buah soal hingga delapan buah soal.
3. Mengambil sample dari banyaknya user yang mengakses simple-o secara bersamaan, terdiri dari 10, 20 dan 30 user.

Proses pengujian dilakukan dengan sistem *sever-client*, dimana sebuah komputer *server* akan terhubung dengan beberapa komputer *client* dengan menggunakan *internet*. Pengguna dapat menggunakan aplikasi Simple-O dengan alamat http://www.ee.ui.ac.id/logic/simple-o untuk mengakses Simple-O yang sudah dimodifikasi proses pencarian persamaan kata dan alamat

http://www.ee.ui.ac.id/logic/bobotsama untuk mengakses aplikasi Simple-O sebelum dimodifikasi pencarian persamaan katanya.

4.1.1 Analisa Penerapan Metode *Searching*

Penerapan *metode searching* pada proses pencarian persamaan kata bekerja berdasarkan karakter awal yang sudah ditentukan. Proses pencarian ini mempercepat pencarian kata dan berpengaruh terhadap keseluruhan proses dari aplikasi Simple-O ini. Metode yang dipergunakan adalah dengan membagi tabel persamaan kata menjadi 5 buah tabel guna mengurangi proses pencarian dan berdampak juga pada waktu pencarian menjadi lebih singkat.

Proses pencarian kata ini dapat dianalogikan sebagai berikut:

1. Terdapat dua buah sistem aplikasi Simple-O, sistem A dan sistem B, dimana A merupakan aplikasi Simple-O lama dengan hanya satu tabel dan B adalah aplikasi Simple-O baru dengan modifikasi penggunaan 5 buah tabel.
2. Pada sistem B, pembagian tabel dilakukan dengan melihat banyaknya penggunaan kata yang tercatat, dalam hal ini, penggunaan kata dengan awalan ber-, di-, me-, dan pe- dipisah masing-masing menjadi satu tabel dan kata-kata yang tidak mengandung awalan tersebut diletakkan pada satu tabel, sehingga terdapat lima tabel berisi kata-kata.
3. Dikondisikan, akan dilakukan proses pencarian kata sebanyak 10 (sepuluh) kata pada kedua sistem, dimana kata yang akan dicari adalah sebagai berikut, bertambah, broadcast, diimplementasikan, domain, jaringan, mekanisme, multimedia, pemrograman, protocol dan wireless. Pengambilan sample kata-kata ini didasarkan pada pembagian tabel untuk melihat unjuk kerja dari proses pencarian kata tersebut.
4. Pengambilan sample akan dilakukan sebanyak 10 (lima) kali untuk setiap kata lalu akan diambil rata-rata waktu pencarian kata tersebut.
5. Pada sistem A, dengan mengacu pada urutan kata dari tabel kata-kata pada aplikasi Simple-o, maka untuk mencari kata-kata yang akan dicari akan melalui proses pencarian yang ditunjukan pada tabel 4.1 berikut:

Tabel 4.1 Tabel Urutan Kata Sistem A

Kata	Banyak kata yang harus dilalui
bertambah	123
broadcast	143
diimplementasikan	247
domain	301
jaringan	424
mekanisme	590
multimedia	735
pemrograman	795
protocol	885
wireless	1139

6. Sedangkan pada sistem B, akan diperoleh hasil yang dapat dilihat pada tabel 4.2.

Tabel 4.2 Tabel Urutan Kata Sistem B

Kata	Banyak kata yang harus dilalui
bertambah	32
broadcast	108
diimplementasikan	26
domain	188

Tabel 4.2 Tabel Urutan Kata Sistem B (lanjutan)

Kata	Banyak kata yang harus dilalui
jaringan	311
mekanisme	3
multimedia	487
pemrograman	17
protocol	556
wireless	811

Berdasarkan data tersebut, dapat diketahui, bahwa pencarian kata pada sistem B memiliki proses yang lebih singkat bila dilihat dari urutan proses pencarian kata. Hal ini dikarenakan, dengan sistem B, proses pencarian kata tidak akan melalui seluruh kata yang terkandung dalam tabel kata-kata seperti yang terdapat pada sistem A, dimana terdapat jumlah data hingga 1500 kata. Sebagai contoh, untuk mencari kata pemrograman, database akan mengarahkan pencarian kata langsung ketabel pe, karena kata pemrograman mengandung kata “pe” didalamnya. Apabila dibandingkan dengan sistem A, kata pemrograman terletak pada urutan 795 pada tabel kata-kata, sehingga database akan melakukan proses sebanyak 794 kali sebelum mendapatkan kata pemrograman. Sedangkan pada sistem B, database akan melakukan proses inisialisasi terlebih dahulu pada kata yang ingin dicari dengan melihat kandungan awal kata yang ingin dicari tersebut. Selanjutnya, tabel akan melakukan pencarian pada tabel yang telah diketahui melalui proses inisialisai tersebut, yakni tabel pe-. Pada tabel pe-, urutan kata pemrogaman terletak pada urutan 17. Berdasarkan urutan proses tersebut, dapat diperoleh bahwa hasil pencarian akan lebih cepat pada sistem B, dan hasil perhitungan waktu kecepatan proses akan dijelaskan selanjutnya. Proses inisialisasi ini dilakukan dengan memanfaatkan fungsi substring pada PHP. Adapun algoritma proses inisialisasi tersebut adalah sebagai berikut:

```
Proc cari kata()
$ber = substr($kata,0,3); //mengambil huruf awal ber
$di = substr($kata,0,2); //mengambil huruf awal di
$me = substr($kata,0,2); //mengambil huruf awal me
$pe = substr($kata,0,2); //mengambil huruf awal pe

if ($string2=="ber") cari kata dalam tabel ber;
elseif ($string1=="di") cari kata dalam tabel di;
elseif ($string1=="me") cari kata dalam tabel me;
elseif ($string1=="pe") cari kata dalam tabel pe;
else $query=$db-> cari kata dalam tabel kata-kata;

if (!$query->EOF) kembalikan kata ke nilai persamaan;
else return 0;
Eproc
```

Gambar 4.1 Algoritma pencarian kata

4.2 Uji Coba dan Analisa Pencarian Kata

Pengujian metode yang digunakan pada sistem B tersebut dilakukan dengan beberapa macam perhitungan guna mendapatkan persentasi peningkatan kinerja dari metode tersebut.

4.2.1 Pencarian Satu Kata

Uji coba yang dilakukan pada tahap ini merupakan uji coba pada pencarian sebuah kata guna membandingkan kecepatan proses pencarian kata pada sistem A dengan sistem B. Adapun hasil rata-rata pengukuran pewaktuan proses pencarian kata tersebut adalah sebagai berikut:

Tabel 4.3 Sample Pewaktuan Pencarian Satu Kata

No	Sistem A	Column1	Sistem B	Column2
	Kata	Waktu Rata-rata	Kata	Waktu Rata-rata
1	bertambah	0.000737929	bertambah	0.000375319
2	broadcast	0.000690961	broadcast	0.000615549

Tabel 4.3 Sample Pewaktuan Pencarian Satu Kata (lanjutan)

No	Sistem A	Column1	Sistem B	Column2
3	diimplementasikan	0.000688005	diimplementasikan	0.000372005
4	domain	0.00071311	domain	0.000638986
5	jaringan	0.00075655	jaringan	0.000606275
6	mekanisme	0.000711894	mekanisme	0.000427318
7	multimedia	0.001368856	multimedia	0.000594306
8	pemrograman	0.001413775	pemrograman	0.000395584
9	protocol	0.000776315	protocol	0.000662732
10	wireless	0.000937604	wireless	0.000678206

Berdasarkan tabel 4.3 tersebut, dapat diperoleh hasil bahwa proses pencarian kata pada sistem B yang menggunakan *metode searching* yang telah dimodifikasi lebih cepat dibandingkan dengan sistem A. Untuk pencarian kata pada sistem B dengan pencarian kata yang sudah diinisialisasikan dan memiliki tabel tersendiri, proses pencariannya memiliki presentase hingga dua (2) kali lebih cepat dibandingkan pada sistem A. Terlihat pada salah satu sample pencarian kata "diimplementasikan", pada sistem A membutuhkan waktu mencapai 0,000688005 detik, sedangkan pada sistem B hanya dibutuhkan waktu sebesar 0,000372005 detik, dimana terlihat jelas bahwa proses pencarian sistem B lebih cepat hampir dua (kali) daripada sistem B. Sedangkan untuk kata-kata yang tidak diinisialisasikan, juga terlihat peningkatan waktu proses pencarian dengan tingkat peningkatan beragam. Keragaman nilai tingkat pencarian ini dipengaruhi oleh letak urut kata-kata yang terdapat pada tabel kata-kata, semakin jauh letak urutan kata tersebut, maka akan semakin lama proses pencarian kata tersebut. Contohnya, kata yang terdapat pada awal tabel seperti pada sample yaitu kata "*broadcast*" membutuhkan waktu pencarian sebesar 0.000690961 detik untuk sistem A dan sebesar 0.000615549 detik untuk sistem B. Sistem B memiliki tingkat kecepatan sedikit lebih cepat dibandingkan dengan sistem A, karena pada sistem B tidak lagi ada kata dengan berawalan huruf ber- karena telah dipetakan menjadi sebuah tabel sehingga proses pencarian menjadi lebih cepat. Selanjutnya, pada contoh kata "wireless", terjadi peningkatan kecepatan yang cukup tinggi yaitu dari 0.000937604 detik pada sistem A dan 0.000678206 detik

pada sistem B. Hal ini disebabkan karena proses pencarian kata pada sistem B tidak lagi sebanyak pada sistem A.

Dari data-data tersebut, dapat terlihat, manfaat penggunaan fungsi substring untuk menginisialisasikan proses pencarian pada tabel-tabel tertentu.

4.2.2 Uji Coba Pada Aplikasi Simple-O Berdasarkan Banyak Data

Uji coba kedua ini dilakukan pada proses pencarian kata dengan data yang dicari lebih banyak. Pada proses ini akan dibandingkan kecepatan proses keseluruhan dari aplikasi Simple-O yaitu antara aplikasi Simple-O yang lama (belum ada modifikasi terhadap pencarian kata) dan aplikasi Simple-O yang baru (telah ada modifikasi pencarian kata). Dalam aplikasi Simple-O, proses pencarian kata ini terdapat sebelum proses perhitungan nilai. Adapun proses pencarian kata ini dipergunakan untuk mencari nilai persamaan kata sebelum dibentuk kedalam matriks untuk dilakukan perhitungan. Pada aplikasi Simple-O ini, kemungkinan penggunaan kata yang berbeda namun memiliki makna sama dari masing-masing individu sangat sering terjadi, untuk mengurangi kesalahan penafsiran oleh sistem, maka diperlukan sebuah tabel berisi kata-kata yang mungkin dipergunakan oleh mahasiswa untuk menjawab pertanyaan atau pun dosen ketika memasukkan sebuah jawaban referensi. Kata-kata yang memiliki makna sama tersebut dihubungkan dengan nilai persamaan, dimana setiap kata yang sama akan diberikan nilai yang sama untuk kemudian dibentuk matriksnya. Proses pencarian kata ini dianggap menjadi salah satu kendala dalam proses keseluruhan dari aplikasi Simpel-O, oleh karena itu diterapkan sistem pencarian kata seperti yang dijelaskan pada contoh sistem B diatas pada aplikasi ini.

Pada uji coba ini akan dilakukan secara bertahap dengan melihat banyak kata yang dicari, dimana akan dilakukan uji coba dengan penggunaan satu soal, dua soal, hingga seterusnya delapan soal. Selanjutnya akan diperoleh hasil kecepatan dari proses keseluruhan aplikasi Simple-O ini.

	idtime	userid	idmk	time
	30	user20	1	2.75498
	31	user1	1	0.744472
	32	user1	21	8.3125
	33	user1	21	9.23494
	34	user1	21	8.1345
	35	user2	21	0.813588
	36	user2	21	0.683471
	37	user2	21	0.67818
	38	user2	21	0.680334
	39	user2	21	0.682103
	40	user2	21	0.670764
	41	user2	21	0.679567
☑	45	mei	21	0.145327
☑	44	mei	21	0.06951
☑	46	mei	21	0.526194
☑	47	mei	21	1.19804
☑	48	mei	21	1.76513
☑	49	mei	21	1.80809
☑	50	mei	21	3.3285
☑	51	mei	21	4.86002

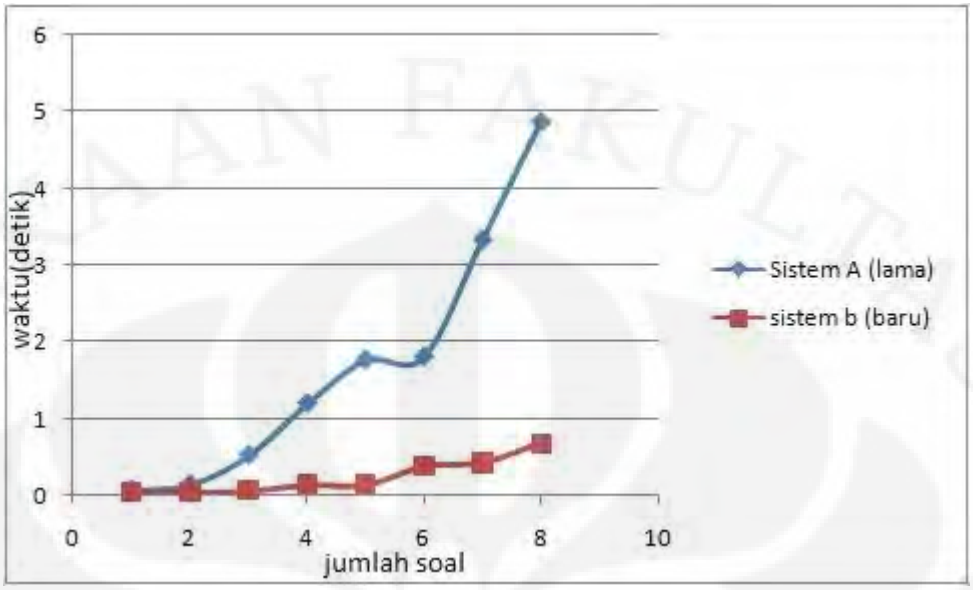
Gambar 4.2 Hasil Uji Coba pada Simple-O lama

	idtime	userid	idmk	time
	80	0906488924	22	0.040056
	81	0906638212	22	0.0404232
	82	0906632410	22	0.0414479
	83	0906632556	22	0.0400441
	84	0906488930	22	0.041096
	85	0906557631	22	0.0433991
	86	0906638490	22	0.039851
	87		22	0.040205
	88	0906632650	21	0.036422
	89	iyuzh	21	0.117217
	90	iyuzh	21	1.42086
	91	iyuzh	21	1.79777
	107	mei	21	0.0452809
	108	mei	21	0.0452602
	109	mei	21	0.064019
	110	mei	21	0.140123
	111	mei	21	0.140713
	112	mei	21	0.379604
	113	mei	21	0.42609
	114	mei	21	0.674471

Gambar 4.3 Hasil Uji Coba pada Simple-O baru

Tabel 4.4. Tabel hasil pengujian berdasarkan banyak data

soal	waktu A (detik)	soal	waktu B (detik)
1 soal	0.06951	1 soal	0.0452809
2 soal	0.145327	2 soal	0.0452602
3 soal	0.526194	3 soal	0.064019
4 soal	1.19804	4 soal	0.140123
5 soal	1.76513	5 soal	0.140713
6 soal	1.80809	6 soal	0.379604
7 soal	3.3285	7 soal	0.42609
8 soal	4.86002	8 soal	0.674471



Gambar 4.4. Grafik Perbandingan Pengujian berdasarkan banyak data

Gambar 4.2 dan Gambar 4.3 merupakan hasil dari uji coba dalam aplikasi Simple-O dimana uji coba ini merupakan simulasi dari ujian yang dilakukan oleh mahasiswa. Kedua gambar tersebut menunjukkan proses keseluruhan aplikasi Simple-O dalam menjalankan program penilaian. Data tersebut merupakan data dari aplikasi Simple-O dalam menjalankan proses penilaian, dimana proses yang dijalankan berurut dari pengerjaan satu buah soal ujian, hingga delapan soal ujian sebenarnya.

Dari data tersebut dapat dilihat, bahwa untuk proses pengerjaan sebuah kumpulan data yang beragam, dibutuhkan waktu yang beragam pula, untuk proses dengan data yang semakin besar, maka waktu yang dibutuhkan juga semakin besar. Apabila diambil satu sample, yaitu pada jumlah data yang paling banyak, maka dapat diketahui bahwa waktu yang dibutuhkan untuk menjalankan sekali aplikasi Simple-O pada sistem baru jauh lebih cepat dibandingkan dengan sistem yang lama, hal ini salah satunya adalah pengaruh dari proses pencarian kata yang sudah diinisialisasikan. Berdasarkan uji coba ini didapatkan peningkatan hingga 7,21 kali lebih cepat.

4.2.3 Uji Coba Pada Aplikasi Simple-O

Untuk menguji tingkat performansi dari program ini, diperlukan skenario penggunaan aplikasi Simple-O ini. Tujuan dari pengujian ini adalah untuk menganalisa perbandingan tingkat kecepatan proses aplikasi Simple-O antara sistem yang telah diterapkan metode pencarian kata lebih spesifik dengan aplikasi yang belum dimodifikasi sistem pencarian katanya. Proses pengujian dilakukan dengan sistem *server-client*, dimana terdapat satu komputer *server* dan beberapa komputer *client* yang terhubung dengan *internet.* Skenario yang digunakan adalah:

1) Pengujian dilakukan dengan melakukan proses pengerjaan ujian pada aplikasi Simple-O yang belum diterapkan sistem pencarian kata yang baru dan aplikasi Simple-O yang telah diterapkan sistem pencarian kata yang baru dengan jumlah pengguna sebanyak 10 orang.
2) Pengujian dilakukan dengan melakukan proses pengerjaan ujian pada aplikasi Simple-O yang belum diterapkan sistem pencarian kata yang baru dan aplikasi Simple-O yang telah diterapkan sistem pencarian kata yang baru dengan jumlah pengguna sebanyak 20 orang.
3) Pengujian dilakukan dengan melakukan proses pengerjaan ujian pada aplikasi Simple-O yang belum diterapkan sistem pencarian kata yang baru dan aplikasi Simple-O yang telah diterapkan sistem pencarian kata yang baru dengan jumlah pengguna sebanyak 30 orang.

4.2.3.1 Skenario Pengujian 1

Pada skenario pengujian 1 ini, dilakukan dengan menggunakan 10 komputer *client* yang menggunakan dua buah aplikasi, yaitu aplikasi Simple-O yang belum diterapkan proses pencarian kata yang baru dan aplikasi Simple-O yang telah diterapkan proses pencarian kata yang baru. Aplikasi lama yang merupakan aplikasi yang belum diterapkan sistem pencarian kata yang baru menjadi parameter dalam proses pengujian ini. Skenario ini dimulai ketika pengguna melakukan proses *submit* secara bersamaan setelah mengerjakan ujian, dimana waktu terhitung semenjak

pengguna menekan tombol *submit* hingga muncul tampilan berupa ucapan terima kasih yang menandakan proses telah selesai dilakukan.

	idtime	userid	idmk	time	submit_time
	35	user5	21	12.6522	2010-06-04 14:43:33
	36	user10	21	12.8699	2010-06-04 14:43:33
	37	user6	21	12.112	2010-06-04 14:43:34
	38	user7	21	14.1976	2010-06-04 14:43:35
	39	user8	21	15.0347	2010-06-04 14:43:35
	40	user4	21	15.2084	2010-06-04 14:43:35
	41	user3	21	18.4057	2010-06-04 14:43:38
	42	user2	21	18.9512	2010-06-04 14:43:39
	43	user9	21	14.5857	2010-06-04 14:43:39
	44	user1	21	20.1664	2010-06-04 14:43:40

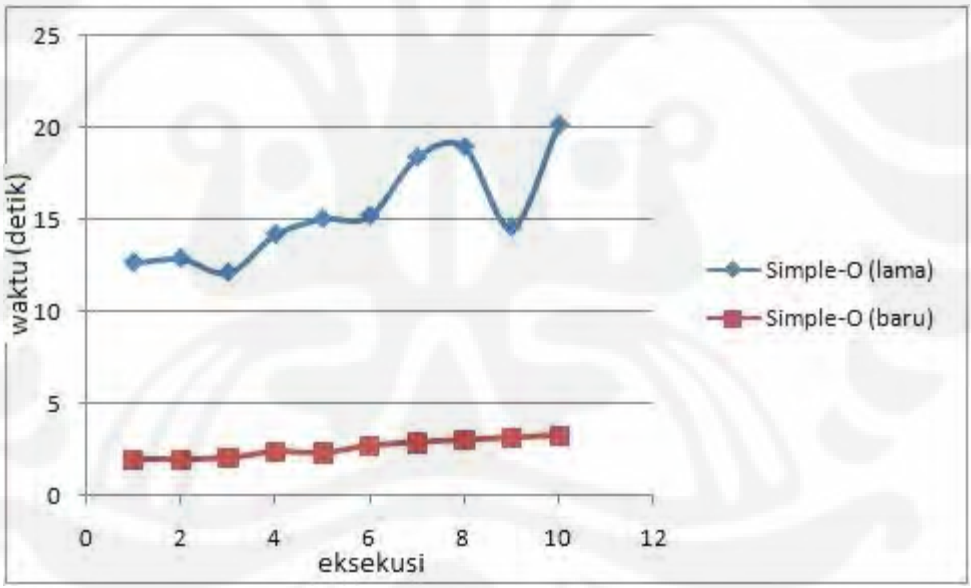
Gambar 4.5 Hasil Pengujian 1 pada Simple-O lama

	idtime	userid	idmk	time	submit_time
	92	user10	21	2.00018	2010-06-04 14:48:50
	93	user6	21	1.98772	2010-06-04 14:48:50
	94	user7	21	2.06722	2010-06-04 14:48:50
	95	user4	21	2.3856	2010-06-04 14:48:50
	96	user5	21	2.33582	2010-06-04 14:48:50
	97	user8	21	2.72093	2010-06-04 14:48:50
	98	user9	21	2.89789	2010-06-04 14:48:51
	99	user3	21	3.02707	2010-06-04 14:48:51
	100	user2	21	3.14183	2010-06-04 14:48:51
	101	user1	21	3.25766	2010-06-04 14:48:51

Gambar 4.6 Hasil Pengujian 1 pada Simple-O baru

Tabel 4.5. Tabel hasil pengujian 1

10 Peserta			
Eksekusi	Waktu Simple-O lama (detik)	Eksekusi	Waktu Simple-O baru (detik)
eksekusi1	12.6522	eksekusi1	2.00018
eksekusi2	12.8699	eksekusi2	1.98772
eksekusi3	12.112	eksekusi3	2.06722
eksekusi4	14.1976	eksekusi4	2.3856
eksekusi5	15.0347	eksekusi5	2.33582
eksekusi6	15.2084	eksekusi6	2.72093
eksekusi7	18.4057	eksekusi7	2.89789
eksekusi8	18.9512	eksekusi8	3.02707
eksekusi9	14.5857	eksekusi9	3.14183
eksekusi10	20.1664	eksekusi10	3.25766
rata-rata	15.41838	rata-rata	2.582192



Gambar 4.7. Grafik Perbandingan Simple-O lama dan Simple-O baru Uji Coba 1

Dengan menggunakan algoritma perhitungan waktu pada Gambar 4.6 dibawah, maka akan diperoleh hasil perhitungan waktu yang dibutuhkan untuk melakukan proses kinerja dari aplikasi Simple-O.

```
Proc delay_login()
$starttime = microtime(true);

        [Proses Perhitungan Aplikasi Simple-O]

$endtime = microtime(true);
$delay = $starttime - $endtime;
Eproc
```

Gambar 4.8. Algoritma perhitungan waktu (PHP manual)

Berdasarkan data diatas, maka rata-rata kecepatan proses aplikasi Simple-O lama adalah 15.41838 detik, sedangkan untuk Simple-o baru adalah 2.582192 detik. Selisih rata-rata yang diperoleh adalah sebesar 12.83619 detik.

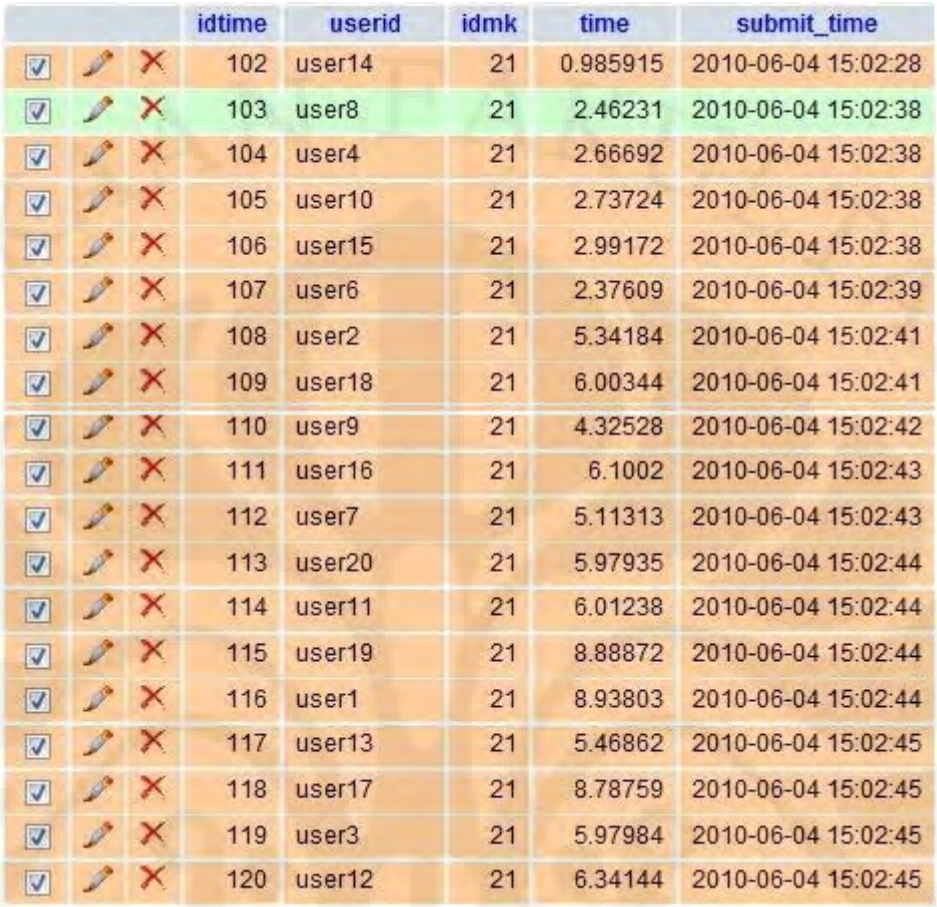
Pada data diatas juga dapat diperoleh data bahwa kecepatan proses aplikasi yang dijalankan secara bersamaan ini bervariasi dan terlihat lebih lambat dibandingkan dengan percobaan sebelumnya. Pada Simple-O lama diperoleh waktu maksimum yang diperlukan adalah sebesar 18.9512 detik dan waktu minimumnya adalah sebesar 12.112 detik. Sedangkan pada Simple-O baru waktu maksimumnya adalah sebesar 3.25766 detik dan minimumnya adalah 1.98772 detik.

4.2.3.2 Skenario Pengujian 2

Pada skenario pengujian 2 ini, dilakukan dengan menggunakan 20 komputer *client* yang menggunakan dua buah aplikasi, yaitu aplikasi Simple-O yang belum diterapkan proses pencarian kata yang baru dan aplikasi Simple-O yang telah diterapkan proses pencarian kata yang baru. Aplikasi lama yang merupakan aplikasi yang belum diterapkan sistem pencarian kata yang baru menjadi parameter dalam proses pengujian ini. Skenario ini dimulai ketika pengguna melakukan proses *submit* secara bersamaan setelah mengerjakan ujian, dimana waktu terhitung semenjak pengguna menekan tombol *submit* hingga muncul tampilan berupa ucapan terima kasih yang menandakan proses telah selesai dilakukan.

			idtime	userid	idmk	time	submit_time
☑	✎	✕	45	user15	21	24.7258	2010-06-04 14:56:18
☑	✎	✕	46	user6	21	30.6231	2010-06-04 14:56:28
☑	✎	✕	47	user10	21	35.0135	2010-06-04 14:56:29
☑	✎	✕	48	user7	21	32.1108	2010-06-04 14:56:29
☑	✎	✕	49	user8	21	36.0409	2010-06-04 14:56:31
☑	✎	✕	50	user5	21	39.4318	2010-06-04 14:56:36
☑	✎	✕	51	user4	21	44.1996	2010-06-04 14:56:38
☑	✎	✕	52	user9	21	53.0358	2010-06-04 14:56:47
☑	✎	✕	53	user3	21	62.3341	2010-06-04 14:57:00
☑	✎	✕	54	user2	21	63.3435	2010-06-04 14:57:00
☑	✎	✕	55	user14	21	67.5007	2010-06-04 14:57:01
☑	✎	✕	56	user13	21	67.9182	2010-06-04 14:57:04
☑	✎	✕	57	user11	21	67.3156	2010-06-04 14:57:05
☑	✎	✕	58	user16	21	72.3443	2010-06-04 14:57:06
☑	✎	✕	59	user1	21	73.9241	2010-06-04 14:57:08
☑	✎	✕	60	user19	21	73.2022	2010-06-04 14:57:11
☑	✎	✕	61	user12	21	74.7851	2010-06-04 14:57:11
☑	✎	✕	62	user17	21	75.7643	2010-06-04 14:57:11
☑	✎	✕	63	user20	21	78.7215	2010-06-04 14:57:12
☑	✎	✕	64	user18	21	79.4247	2010-06-04 14:57:13

Gambar 4.9. Hasil Pengujian 2 pada Simple-O lama



	idtime	userid	idmk	time	submit_time
	102	user14	21	0.985915	2010-06-04 15:02:28
	103	user8	21	2.46231	2010-06-04 15:02:38
	104	user4	21	2.66692	2010-06-04 15:02:38
	105	user10	21	2.73724	2010-06-04 15:02:38
	106	user15	21	2.99172	2010-06-04 15:02:38
	107	user6	21	2.37609	2010-06-04 15:02:39
	108	user2	21	5.34184	2010-06-04 15:02:41
	109	user18	21	6.00344	2010-06-04 15:02:41
	110	user9	21	4.32528	2010-06-04 15:02:42
	111	user16	21	6.1002	2010-06-04 15:02:43
	112	user7	21	5.11313	2010-06-04 15:02:43
	113	user20	21	5.97935	2010-06-04 15:02:44
	114	user11	21	6.01238	2010-06-04 15:02:44
	115	user19	21	8.88872	2010-06-04 15:02:44
	116	user1	21	8.93803	2010-06-04 15:02:44
	117	user13	21	5.46862	2010-06-04 15:02:45
	118	user17	21	8.78759	2010-06-04 15:02:45
	119	user3	21	5.97984	2010-06-04 15:02:45
	120	user12	21	6.34144	2010-06-04 15:02:45

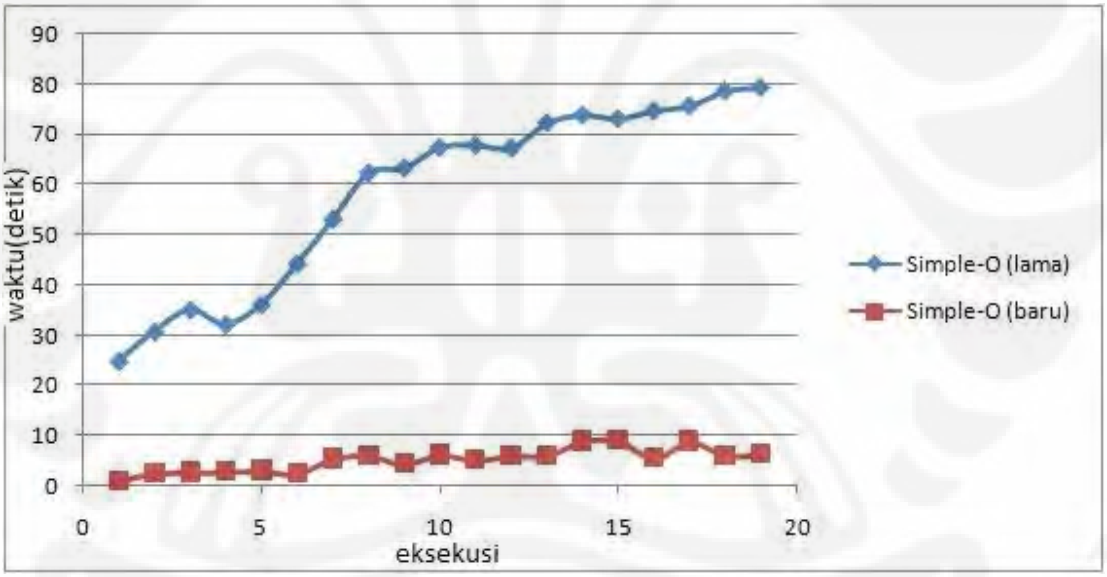
Gambar 4.10. Hasil Pengujian 2 pada Simple-O baru

Tabel 4.6. Tabel hasil pengujian 2

20 Peserta			
Eksekusi	Waktu Simple-O lama (detik)	Eksekusi	Waktu Simple-O baru (detik)
eksekusi1	24.7258	eksekusi1	0.985915
eksekusi2	30.6231	eksekusi2	2.46231
eksekusi3	35.0135	eksekusi3	2.66692
eksekusi4	32.1108	eksekusi4	2.73724
eksekusi5	36.0409	eksekusi5	2.99172
eksekusi6	error	eksekusi6	2.37609
eksekusi7	44.1996	eksekusi7	5.34184
eksekusi8	53.0358	eksekusi8	6.00344
eksekusi9	62.3341	eksekusi9	4.32528
eksekusi10	63.3435	eksekusi10	6.1002
eksekusi11	67.5007	eksekusi11	5.11313

Tabel 4.6. Tabel hasil pengujian 2 (lanjutan)

20 Peserta			
Eksekusi	Waktu Simple-O lama (detik)	Eksekusi	Waktu Simple-O baru (detik)
eksekusi12	67.9182	eksekusi12	5.97935
eksekusi13	67.3156	eksekusi13	6.01238
eksekusi14	72.3443	eksekusi14	8.88872
eksekusi15	73.9241	eksekusi15	8.93803
eksekusi16	73.2022	eksekusi16	5.46862
eksekusi17	74.7851	eksekusi17	8.78759
eksekusi18	75.7643	eksekusi18	5.97984
eksekusi19	78.7215	eksekusi19	6.34144
eksekusi20	79.4247	eksekusi20	error
rata-rata	58.54356842	rata-rata	5.131581842



Gambar 4.11. Grafik Perbandingan Simple-O lama dan Simple-O baru Uji Coba 2

Pada proses perhitungan rata-rata pengujian 2 ini hanya akan diambil 19 pengguna, karena terdapat kesalahan pada proses alamat Simple-O dimana pewaktuannya tidak tercatat kedalam database. Rata-rata dari aplikasi Simple-O lama adalah sebesar 58.54357 detik, sedangkan untuk Simple-O baru adalah sebesar 5.131582 detik, sehingga selisih diantara keduanya adalah 53.41199 detik.

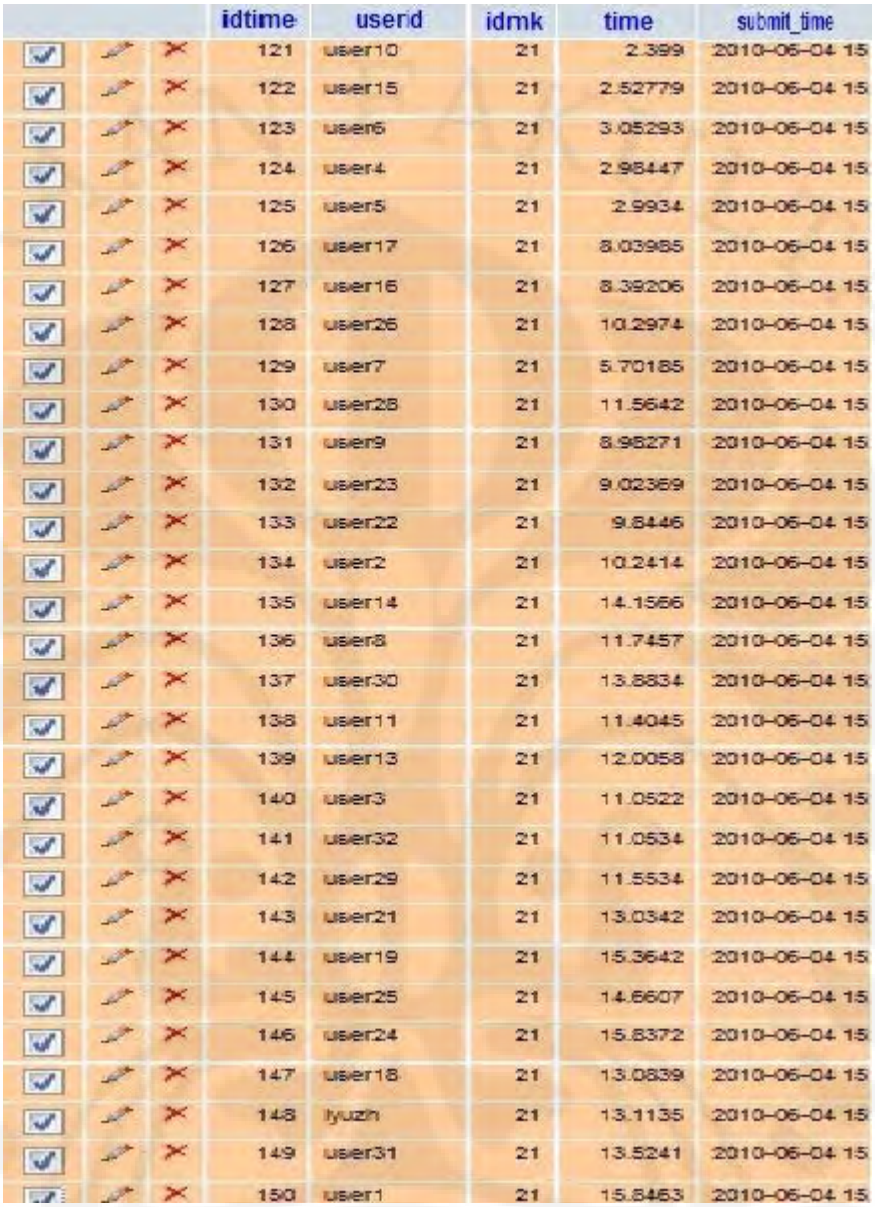
Pada Simple-O lama diperoleh waktu maksimum yang diperlukan adalah sebesar 79.4247 detik dan waktu minimumnya adalah sebesar 24.7258 detik. Sedangkan pada Simple-O baru waktu maksimumnya adalah sebesar 8.93803 detik dan minimumnya adalah 0.985915 detik.

4.2.3.3 Skenario Pengujian 3

Pada skenario pengujian 3 ini, dilakukan dengan menggunakan 30 komputer *client* yang menggunakan dua buah aplikasi, yaitu aplikasi Simple-O yang belum diterapkan proses pencarian kata yang baru dan aplikasi Simple-O yang telah diterapkan proses pencarian kata yang baru. Aplikasi lama yang merupakan aplikasi yang belum diterapkan sistem pencarian kata yang baru menjadi parameter dalam proses pengujian ini. Skenario ini dimulai ketika pengguna melakukan proses *submit* secara bersamaan setelah mengerjakan ujian, dimana waktu terhitung semenjak pengguna menekan tombol *submit* hingga muncul tampilan berupa ucapan terima kasih yang menandakan proses telah selesai dilakukan.

			idtime	user	idmk	time	submit_time
☑	✎	✕	65	user10	21	47.9598	2010-06-04 15:11:32
☑	✎	✕	66	user15	21	50.9479	2010-06-04 15:11:35
☑	✎	✕	67	user7	21	56.9902	2010-06-04 15:11:46
☑	✎	✕	68	user6	21	62.9532	2010-06-04 15:11:51
☑	✎	✕	69	user4	21	72.0167	2010-06-04 15:11:56
☑	✎	✕	70	user22	21	100.462	2010-06-04 15:12:26
☑	✎	✕	71	user5	21	98.4678	2010-06-04 15:12:26
☑	✎	✕	72	user8	21	112.666	2010-06-04 15:12:39
☑	✎	✕	73	user29	21	113.854	2010-06-04 15:12:43
☑	✎	✕	74	user14	21	131.38	2010-06-04 15:12:55
☑	✎	✕	75	user16	21	133.344	2010-06-04 15:12:57
☑	✎	✕	76	user23	21	130.766	2010-06-04 15:12:58
☑	✎	✕	77	user28	21	129.966	2010-06-04 15:12:59
☑	✎	✕	78	user9	21	133.368	2010-06-04 15:13:01
☑	✎	✕	79	user3	21	143.397	2010-06-04 15:13:12
☑	✎	✕	80	user2	21	146.028	2010-06-04 15:13:14
☑	✎	✕	81	user32	21	148.513	2010-06-04 15:13:17
☑	✎	✕	82	user33	21	151.784	2010-06-04 15:13:19
☑	✎	✕	83	user1	21	154.611	2010-06-04 15:13:21
☑	✎	✕	84	user19	21	158.497	2010-06-04 15:13:22
☑	✎	✕	85	user13	21	158.078	2010-06-04 15:13:22
☑	✎	✕	86	user11	21	154.095	2010-06-04 15:13:22
☑	✎	✕	87	user31	21	154.57	2010-06-04 15:13:23
☑	✎	✕	88	user20	21	161.445	2010-06-04 15:13:25
☑	✎	✕	89	user17	21	158.764	2010-06-04 15:13:25
☑	✎	✕	90	user24	21	164.154	2010-06-04 15:13:30
☑	✎	✕	91	user12	21	163.066	2010-06-04 15:13:31
☑	✎	✕	92	user26	21	164.601	2010-06-04 15:13:33
☑	✎	✕	93	user21	21	166.816	2010-06-04 15:13:34
☑	✎	✕	94	user18	21	165.502	2010-06-04 15:13:34

Gambar 4.12. Hasil Pengujian 3 pada Simple-O lama



	idtime	userid	idmk	time	submit_time
	121	user10	21	2.399	2010-06-04 15
	122	user15	21	2.52779	2010-06-04 15
	123	user6	21	3.05293	2010-06-04 15
	124	user4	21	2.98447	2010-06-04 15
	125	user5	21	2.9934	2010-06-04 15
	126	user17	21	8.03985	2010-06-04 15
	127	user16	21	8.39206	2010-06-04 15
	128	user26	21	10.2974	2010-06-04 15
	129	user7	21	5.70185	2010-06-04 15
	130	user28	21	11.5642	2010-06-04 15
	131	user9	21	8.96271	2010-06-04 15
	132	user23	21	9.02369	2010-06-04 15
	133	user22	21	9.6446	2010-06-04 15
	134	user2	21	10.2414	2010-06-04 15
	135	user14	21	14.1566	2010-06-04 15
	136	user8	21	11.7457	2010-06-04 15
	137	user30	21	13.8834	2010-06-04 15
	138	user11	21	11.4045	2010-06-04 15
	139	user13	21	12.0058	2010-06-04 15
	140	user3	21	11.0522	2010-06-04 15
	141	user32	21	11.0534	2010-06-04 15
	142	user29	21	11.5534	2010-06-04 15
	143	user21	21	13.0342	2010-06-04 15
	144	user19	21	15.3642	2010-06-04 15
	145	user25	21	14.6607	2010-06-04 15
	146	user24	21	15.8372	2010-06-04 15
	147	user18	21	13.0839	2010-06-04 15
	148	iyuzh	21	13.1135	2010-06-04 15
	149	user31	21	13.5241	2010-06-04 15
	150	user1	21	15.8463	2010-06-04 15

Gambar 4.13. Hasil Pengujian 3 pada Simple-o baru

Tabel 4.7. Tabel hasil pengujian 3

30 Peserta			
Eksekusi	Waktu Simple-O lama (detik)	Eksekusi	Waktu Simple-O baru (detik)
eksekusi1	47.9598	eksekusi1	2.399
eksekusi2	50.9479	eksekusi2	2.52779
eksekusi3	56.9902	eksekusi3	3.05293
eksekusi4	62.9532	eksekusi4	2.98447

Tabel 4.7. Tabel hasil pengujian 3 (lanjutan)

30 Peserta			
Id Pengguna	Waktu Simple-O lama (detik)	Id Pengguna	Waktu Simple-O baru (detik)
eksekusi5	72.0167	eksekusi5	2.9934
eksekusi6	100.462	eksekusi6	8.03985
eksekusi7	98.4678	eksekusi7	8.39206
eksekusi8	112.666	eksekusi8	10.2974
eksekusi9	113.854	eksekusi9	5.70185
eksekusi10	131.38	eksekusi10	11.5642
eksekusi11	133.344	eksekusi11	8.98271
eksekusi12	130.766	eksekusi12	9.02369
eksekusi13	129.966	eksekusi13	9.8446
eksekusi14	133.368	eksekusi14	10.2414
eksekusi15	143.397	eksekusi15	14.1566
eksekusi16	146.028	eksekusi16	11.7457
eksekusi17	148.513	eksekusi17	13.8834
eksekusi18	151.784	eksekusi18	11.4045
eksekusi19	154.611	eksekusi19	12.0058
eksekusi20	158.497	eksekusi20	11.0522
eksekusi21	158.078	eksekusi21	11.0534
eksekusi22	154.095	eksekusi22	11.5534
eksekusi23	154.57	eksekusi23	13.0342
eksekusi24	161.445	eksekusi24	15.3642
eksekusi25	158.764	eksekusi25	14.6607
eksekusi26	164.154	eksekusi26	15.8372
eksekusi27	163.066	eksekusi27	13.0839
eksekusi28	164.601	eksekusi28	13.1135
eksekusi29	166.816	eksekusi29	13.5241
eksekusi30	165.502	eksekusi30	15.8463
rata-rata	129.63542	rata-rata	10.24548167

Gambar 4.14. Grafik Perbandingan Simple-O lama dengan Simple-O baru Uji Coba 3

Berdasarkan data diatas, maka rata-rata dari aplikasi Simple-O lama adalah sebesar 129.6354 detik, sedangkan untuk Simple-O baru adalah sebesar 10.24548 detik, sehingga selisih diantara keduanya adalah 119.3899 detik.

Pada Simple-O lama diperoleh waktu maksimum yang diperlukan adalah sebesar 166.816 detik dan waktu minimumnya adalah sebesar 47.9598 detik. Sedangkan pada Simple-O baru waktu maksimumnya adalah sebesar 15.8463 detik dan minimumnya adalah 2.399 detik.

4.3 Analisis Kecepatan Proses

Berdasarkan hasil pengujian diatas, diperoleh hasil bahwa proses *submit* jawaban bersamaan akan memakan waktu proses yang berbeda-beda tergantung pada jumlah penggunanya. Semakin banyak pengguna yang mengakses ujian tersebut, maka akan semakin besar waktu yang diperlukan untuk menjalankan proses tersebut.

Berdasarkan data diatas, dapat diperoleh data perbandingan antara sistem Simple-O lama dengan sistem Simple-O baru yaitu:

1. Uji coba 1, memiliki selisih rata-rata hingga 12.83619 detik dan meningkatkan proses dan meningkatkan proses Simple-o baru mencapai 5,97 kali lebih cepat dibanding Simple-O lama.

2. Uji coba 2, memiliki selisih rata-rata hingga 53.41199 detik dan meningkatkan proses Simple-O baru mencapai 11,41 kali lebih cepat dibanding Simple-O lama.
3. Uji coba 3, memiliki selisih rata-rata hingga 119.3899 detik dan meningkatkan proses Simple-O baru mencapai 12,65 kali lebih cepat dibanding Simple-O lama.

Perbedaan peningkatan waktu yang cukup besar ini terjadi salah satunya karena pada sistem Simple-o telah menggunakan proses pencarian kata yang lebih baik sehingga kinerja database akan menjadi optimal dan tidak terlalu terbebani ketika digunakan oleh banyak pengguna.

BAB 5
KESIMPULAN

Dari hasil ujicoba dan analisis yang telah dilakukan dapat diambil beberapa kesimpulan:

1. Modifikasi Simple-O berhasil diterapkan guna meningkatkan proses kinerja database dalam pencarian persamaan kata.

2. Penggunaan kalimat dengan kata-kata yang mengandung huruf ber-, di-, me- dan pe- banyak digunakan dalam pengunaan bahasa dalam ujian pada Simple-O.

3. Penerapan metode *searching* pada aplikasi Simple-O dengan inisialisasi penggunaan tabel meningkatkan proses pencarian persamaan kata dan proses kinerja aplikasi Simple-O secara keseluruhan. Dari uji coba diperoleh hasil sebagai berikut:

   - Uji satu user, memiliki selisih 4.185549 detik dan meningkatkan proses mencapai 7,21 kali lebih cepat.
   - Uji coba 1 dengan 10 user secara bersamaan, memiliki selisih rata-rata hingga 12.83619 detik dan meningkatkan proses Simple-O baru mencapai hampir 5,97 kali lebih cepat dibanding Simple-O lama.
   - Uji coba 2 dengan 20 user, memiliki selisih rata-rata hingga 53.41199 detik dan meningkatkan proses Simple-O baru mencapai 11,41 kali lebih cepat dibanding Simple-O lama.
   - Uji coba 3 dengan 30 user, memiliki selisih rata-rata hingga 119.3899 detik dan meningkatkan proses Simple-O baru mencapai 12,65 kali lebih cepat dibanding Simple-O lama.

DAFTAR PUSTAKA

1. Anak Agung Putri Ratna, Bagio Budiardjo dan Djoko Hartanto, "SIMPLE: Sistem Penilaian Esei Otomatis untuk Menilai Ujian dalam Bahasa Insonesia", Jurnal Makara Seri Teknologi, volume 11, April 2007, ISSN : 1693-6698

2. Anak Agung Putri Ratna, Adhe W. Astato, Bagio Budiardjo, Djoko Hartanto, "Simple-O: Web Based Automated Essay Grading System Using Latent Semantic Analysis method for Indonesian Language Considering Weight Word and Word Synonym", The 10th International Conference on Quality in Research, Faculty of Engineering, University of Indonesia, 4 – 6 December 2007, Depok, Indonesia.

3. C.J Date, An Introduction to Database System, Vol I Reading, Mass.: Addison-Wesley, 1981.

4. Everest, Gordon D., "The Future of Database Management", ACM, 1974.

5. Hermawandi, Dudi, "Implementasi Skema Pembobotan SICBI pada Aplikasi Essay Grading Metode Latent Semantic Analysis", Laporan Akhir Skripsi, 2008.

6. Kadir, Abdul, "Belajar Database menggunakan MySQL", Penerbit ANDI Yogyakarta, 2008.

7. Lahdenmaki, Tapio., Leach Michael, "Relational Database Index Design and The Optimizers, John Wiley & Sons, Inc., 2005.

8. Palmer, J., Williams, R., Dreher, H., "Automated Essay Grading System Applied to a First Year University Subject-How Can We do it Better?", InSITE-"Where Parallels Intersect". Pp 1221-1229, Informing Science, Perth, Australia, June 2002.Schwab, Robert A., Fabbri, Tony, “Practical Database Management, 1992.

9. Sukarno, Mohamad, “Membangun Website Dinamis Interaktif dengan PHP-MySQL (Windows dan Linux), Penerbit Eska Media Press Jakarta, 2006.

10. Basis data, 3 Desember 2009. http://id.wikipedia.org/wiki/Basis_data/

11. Database, 3 Desember, 2009. http://en.wikipedia.org/wiki/Database/

12. MySQL, 3 Desember, 2009. http://id.wikipedia.org/wiki/MySQL/

Daftar Acuan

[1] Ngongo, Adrianus, “Win Win Solution Ujian Nasioanl”, http://www.timorexpress.com/index.php?act=news&nid=38398, Desember 2009.

[2] Napitupulu, Ester Lince, “Soal Pilihan Ganda Menjerumuskan”, http://nasional.kompas.com/read/xml/2009/11/01/19445564/soal.pilihan.ganda.menjerumuskan/, Laporan Wartawan KOMPAS, 2009.

[3] Haley, D. T., et al, "The Learning Grid and E-Assessment using Latent Semantic Analysis", Computing Research Centre, UK, 2004.

[4] Silberschatz−Korth−Sudarshan, “Database System Concepts, Fourth Edition”, The McGraw-Hill Company, 2001.