UJM 11 (2). 112-119
UNNES Journal of Mathematics
http://journal.unnes.ac.id/sju/index.php/ujm

ANALISIS KOMPRESI FILE TEKS MENGGUNAKAN ALGORITMA LEMPEL ZIV WELCH (LZW)

Muhammad Iqbal ✉, Rusbi Anggi Prayogi, Dwi Yunitasari

Jurusan Matematika, Universitas Negeri Semarang, Indonesia
Kampus Sekaran Gunungpati, Semarang, 50229

Info Artikel

Sejarah Artikel:
Diterima Oktober 2022
Disetujui November 2022
Dipublikasikan November 2022

Keywords:
Kompresi, file teks, Algoritma Lempel-Ziv-Welch

Abstrak

Meningkatnya pengguna komputer saat ini, membuat semakin banyak kebutuhan akan penyimpanan data. Salah satu cara untuk mengatasinya, dengan mengembangkan berbagai algoritma kompresi yang digunakan untuk memampatkan data, salah satunya adalah Algoritma LZW. Algoritma LZW ini memampatkan data dengan memanfaatkan dictionary, di mana indeks yang diperoleh dari sebuah "kamus" menggantikan fragmen-fragmen teks. Aplikasi kompresi yang telah dibuat dengan menerapkan Algoritma LZW, didapatkan hasil pengujian terhadap 40 file berjenis *.doc, *.txt, *.rtf, *.html yaitu file jenis *.doc memiliki rasio kompresi paling rendah.

Abstract

*The increase in computer users today, makes more and more needs for data storage. One way to overcome this is by developing various compression algorithms that are used to compress data, one of which is the LZW Algorithm. This LZW algorithm compresses data by utilizing a dictionary, where the index obtained from a "dictionary" replaces text fragments. The compression application that has been made by applying the LZW Algorithm, obtained test results on 40 files of type *.doc, *.txt, *.rtf, *.html, namely *.doc type files have the lowest compression ratio*

How to cite:

Iqbal, M., Prayogi, R. A. & Yunitasari, D. 2022. Analisis Kompresi File Teks Menggunakan Algoritma Lempel Ziv Welch (LZW). *UNNES Journal of Mathematics*. 11(2):112-119.

✉Alamat korespondensi:
E-mail: muhammadiqbal@mail.unnes.ac.id

p-ISSN 2252-6943
e-ISSN 2460-5859

PENDAHULUAN

Semakin meningkatnya penggunaan komputer pada kehidupan sehari-hari, membuat semakin banyak pula kebutuhan penyimpanan data. Data-data seperti file text, gambar, audio, dan video merupakan data yang sering diproses menggunakan komputer. Ukuran suatu data semakin besar, semakin besar tempat penyimpanan yang dibutuhkan. Ukuran data yang besar juga berpengaruh pada proses pengiriman data melalui media transmisi, semakin besar ukuran data, maka waktu yang dibutuhkan untuk pengiriman data tersebut semakin besar. Oleh karena itu, pengembangan algoritma-algoritma kompresi muncul yang bertujuan untuk memampatkan ukuran data. Menggunakan algoritma Lempel Ziv Welch (LZW) dalam melakukan pemampatan data pada beberapa penelitian sebelumnya tentang teknologi kompresi data berbasis dictionary LZW (Darwis et al., 2018) Dengan dimampatkannya data – data tersebut, dapat mempercepat proses pengiriman data (Solomon, 2007).

Kompresi data adalah teknik untuk memampatkan data agar ukuran data yang lebih kecil daripada ukuran aslinya, sehingga dapat mempersingkat waktu pertukaran data serta lebih efisien dalam penyimpanannya (Pu, 2006). Kompresi juga dapat menghemat biaya dikeluarkan guna menambah fasilitas media penyimpanan data pada komputer serta mempercepat waktu proses transmisi data (Anton, 2005). Algoritma kompresi data telah banyak dikembangkan untuk keperluan kompresi data dengan tipe-tipe data tertentu (Laia, Y., 2016). Misalkan untuk kompresi file teks, terdapat beberapa algoritma seperti algoritma Huffman, Ziv and Lempel 77 (LZ77), Ziv and Lempel 78 (LZ78), Lempel Ziv Welch (LZW), Dynamic Markov Compression (DMC), dan Run Length Encoding (RLE) (Linawati et al., 2004; Mahesa, K., 2017; Satyapratama, A., et al., 2015).

Pada penelitian ini akan digunakan algoritma LZW untuk kompresi file text. Algoritma LZW dikembangkan dari metode kompresi yang dibuat oleh Ziv dan Lempel pada tahun 1977. Algoritma ini melakukan kompresi dengan menggunakan dictionary, dimana fragmen-fragmen teks digantikan dengan indeks yang diperoleh dari sebuah "kamus". Teknik ini bersifat adaptif dan efektif karena karakter dikodekan dengan mengacu pada string yang muncul dalam teks. Teknik kompresi tercapai jika referensi dalam bentuk pointer dapat disimpan dalam jumlah bit yang lebih sedikit dibandingkan string aslinya (Munir, 2005)

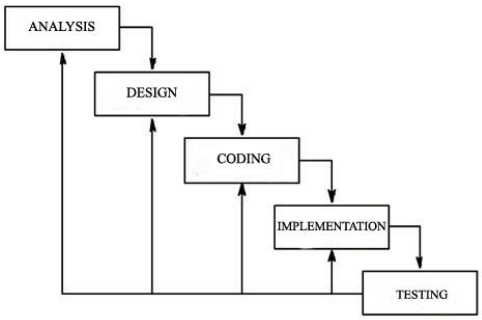
Penelitian tentang kompresi teks telah cukup banyak dilakukan oleh para peneliti seperti analisis perbandingan kinerja Algoritma Kompresi Lempel Ziv 77, Lempel Ziv 78 dan Lempel Ziv Welch (Pratama, 2010). Dari hasil penelitiannya tersebut, di antara ketiga algoritma LZ 77, LZ 78, dan LZW, algoritma yang memiliki performa terbaik untuk file teks adalah algoritma LZW, diikuti oleh algoritma LZ 77 dan terakhir algoritma LZ 78 (Hidayat, H. et al., 2013; Putra, M. A. et al., 2018; Hutapea, D. (2021).

METODE PENELITIAN

Pada penelitian ini, secara umum terdiri tiga tahapan yaitu perumusan masalah dan batasan masalah, studi literatur, dan pengembangan perangkat lunak dengan menggunakan metode Waterfall. Pada tahap perumusan masalah dan pembatasan masalah, dilakukan perumusan masalah apa saja yang akan dibahas dan batasan masalah tentang implementasi algoritma LZW pada kompresi teks yang akan dibahas. Tahapan ini dilakukan agar perancangan dan pengembangan aplikasi tidak melebar dan terfokus pada masalah tertentu.

Tahap kedua adalah melakukan studi literatur, yaitu dengan mencari sumber-sumber baik berupa buku, jurnal, makalah, ataupun situs-situs yang berkaitan dengan teknik implementasi algoritma LZW pada kompresi teks. Sumber-sumber tersebut tentunya akan dijadikan referensi pada perancangan dan pengembangan perangkat lunak yang dibangun.

Tahap ketiga adalah pengembangan perangkat lunak menggunakan metode Waterfall (Roger, S. P. et al., 2015). Metode ini dalam pengembangan perangkat lunak menggunakan pendekatan secara berurutan. Dalam pengembangan sistem, metode ini menganalisis kebutuhan spesifikasi sistem di awal sehingga tidak banyak perubahan (Arisantoso, A. et al., 2022). Diagram metode Waterfall dapat dilihat pada Gambar 1



Gambar 1. Metode Waterfall

Analisis

Pada tahapan ini data dan informasi yang telah didapat melalui studi literatur dan analisis data, kemudian semua data atau informasi yang berkaitan dengan algoritma LZW, serta bahasa pemrograman Java dianalisis dan didefinisikan kembali untuk memenuhi data-data dari sistem yang dibangun.

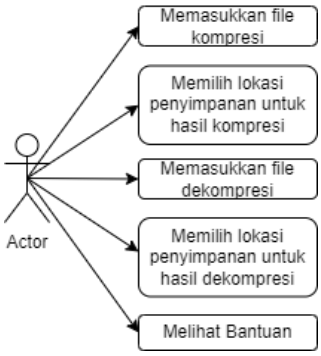
Desain

Pada tahap design ini, akan dirancang perangkat lunak kompresi teks yang akan memiliki kemampuan untuk:

Melakukan kompresi file dengan ekstensi dan kapasitas file yang sudah diketahui. Pengkompresian file dengan memampatkan ukuran aslinya menjadi lebih kecil. Sedangkan dekompresi file merupakan kebalikannya, agar kembali ke ukuran dan bentuk file aslinya dengan tetap file tersebut dapat digunakan sebagaimana mestinya.

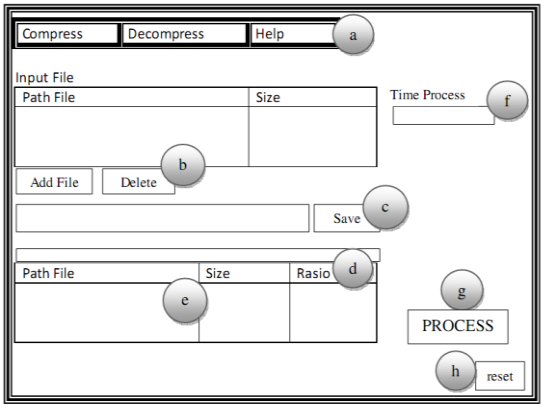
Perangkat lunak yang akan dibangun memiliki dua tahapan pengkompresian, yaitu tahapan kompresi dan tahapan dekompresi. Tahapan kompresi adalah untuk proses pemampatan file menjadi lebih kecil ukurannya, sedangkan tahapan dekompresi adalah untuk mengembalikan file yang telah dikompresi menjadi file semula.

Pada Gambar 2 Diagram use case terdapat satu aktor yaitu user. User melakukan kompresi file dengan memasukkan file yang belum terkompresi berupa file teks yang memiliki ekstensi *.txt, *.html, *.rtf, atau *.doc. Kemudian, user dapat memilih lokasi penyimpanan file hasil kompresi. Selanjutnya, user dapat melakukan dekompresi file dengan memasukkan file yang telah terkompresi, yaitu file yang mempunyai ekstensi *.ubi, agar kembali ke jenis file aslinya. Kemudian user juga dapat melihat menu bantuan untuk mengetahui lebih jelas bagaimana perangkat lunak ini dijalankan.



Gambar 2. Diagram Use Case

Desain user interface dari aplikasi yang dibuat ditampilkan pada Gambar 3 di bawah ini.



Gambar 3. Desain interface aplikasi kompresi

Adapun keterangan bagian-bagian yang ada dalam Gambar 3 adalah:

- Main menu terdapat sub menu kompresi, dekompresi, dan bantuan.
- Tombol Add File untuk memasukkan file input yang akan diproses baik kompresi maupun dekompresi, yaitu berupa nama file yang disertai direktori penyimpanannya, dan ukuran file tersebut.
- Komponen TextField tempat menentukan direktori penyimpanan dari hasil proses kompresi atau proses dekompresi, dan tombol Save adalah tombol pencarian direktori untuk menyimpan.

- ProgressBar berfungsi untuk mengetahui berjalannya program saat pemrosesan kompresi atau dekompresi.
- Tabel berisi keterangan file output hasil dari proses kompresi maupun dekompresi, yaitu berupa nama file, direktori penyimpanan, ukuran file dan rasio kompresi.
- TextField yang menampilkan waktu yang dibutuhkan selama proses berlangsung baik proses kompresi maupun dekompresi.
- Tombol Process berfungsi menjalankan proses kompresi atau dekompresi.
- Tombol Reset berfungsi menghapus semua hasil proses kompresi atau dekompresi.

Pengkodean dan Implementasi

Pada tahap desain yang telah dibuat kemudian diterjemahkan dalam kode-kode program menggunakan bahasa pemrograman Java yang diimplementasikan ke dalam aplikasi.

Pengujian

Tahap pengujian adalah melakukan pengujian terhadap perangkat lunak yang telah dikembangkan untuk melihat apakah perangkat lunak tersebut berjalan fungsi. Pengujian dilakukan dengan melakukan evaluasi keluaran yang dihasilkan dari proses kompresi dengan melihat ukuran file hasil kompresi, waktu yang dibutuhkan untuk proses yang dijalankan serta rasio kompresi yang dicapai. Sedangkan keluaran yang dihasilkan dari proses dekompresi akan dievaluasi sama dengan yang dilakukan pada proses dekompresi.

Rasio kompresi diukur dengan membandingkan antara ukuran file sebelum dikompresi dengan ukuran file setelah dikompresi.

$$\text{Rasio} = \frac{Ukuran\ File\ Asli}{Ukuran\ File\ Kompresi}$$

Dan jika dinyatakan dalam persentase, maka dapat dituliskan dalam persamaan berikut :

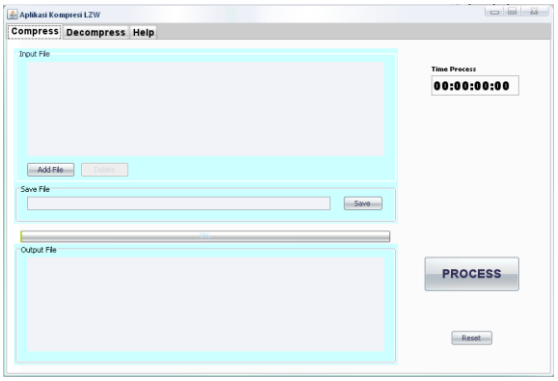
$$\text{P} = \left(\frac{Ukuran\ File\ Kompresi}{Ukuran\ File\ Asli}\right) * 100\%$$

Semakin kecil persentase rasio suatu kompresi data maka semakin efektif metode kompresi tersebut. Pada saat proses dekompresi, rasio kompresi akan berselang-seling berdasarkan pengaruh data terhadap algoritma yang digunakan. Dengan demikian maka perlu diperhatikan adalah rasio kompresi rata-rata. Bukan pada rasio yang dicapai pada suatu waktu tertentu. Dalam pengujian akan digunakan masing- masing berupa 10 file *.txt, *.doc, *.rtf, dan *.html.

HASIL DAN PEMBAHASAN

Implementasi Desain Sistem

User interface aplikasi kompresi file terdiri dari 3 menu inti yaitu menu kompresi, menu dekompresi, menu bantuan. *User interface* halaman aplikasi dapat dilihat pada Gambar 4.



Gambar 4. Halaman Kompresi, Dekompresi dan Informasi.

Implementasi Aplikasi Kompresi

Pada implementasi aplikasi kompresi terdapat 3 tahapan, yaitu desimalisasi, binerisasi, dan kompresi algoritma LZW, sedangkan proses dekompresi memiliki 2 tahapan, yaitu desimalisasi, dan dekompresi algoritma LZW. Pada tahapan selanjutnya membahas implementasi dari tahapan kompresi dan dekompresi tersebut.

1. Desimilasasi dan Binerisasi Proses Kompresi

Pada desimalisasi, langkah-langkah yang akan dilakukan adalah mengubah semua karakter ASCII yang berbentuk huruf, angka dan simbol menjadi desimal dari kode ASCII, kemudian dilakukan proses binerisasi. Proses binerisasi dilakukan dengan mengubah karakter

yang sudah didesimalkan berdasarkan kode ASCII menjadi binary.

2. Kompresi Algoritma LZW

Tahap kompresi menerapkan Algoritma LZW ini dimulai dengan input Byte pertama dari file, disimpan ke dalam variabel newstring. Kemudian masukan Byte berikutnya kemudian disimpan ke dalam variabel newchar. Newstring dan newchar dimasukkan ke dalam sebuah variabel dengan nama UCode. UCode kemudian dicocokkan di tabel dictionary yang telah disiapkan, dengan indeks awal 256 dan indeks akhir 4095. Apabila UCode berada dalam tabel, maka newstring baru diisi dengan newstring lama ditambah newchar. Apabila UCode tidak berada dalam tabel, maka output variable newstring. Kemudian UCode ditambahkan ke dalam tabel dan newstring diisi dengan newchar. Hal tersebut dilakukan sampai dengan akhir file.

3. Dekompresi Algoritma LZW

Tahap dekompresi mengimplementasikan Algoritma LZW dengan menyimpan kode awal hasil kompresi kemudian disimpan ke dalam variabel Ocode dan keluarkan variabel Ocode. Kemudian masukkan kode berikutnya dan disimpan ke dalam variabel Ncode. Variabel Ncode dicocokkan dalam dictionary. Jika ada dalam dictionary, berarti newstring diisi dengan Ncode. Jika tidak ada dalam dictionary, maka newstring diisi dengan Ocode dan newstring baru diisi dengan newstring lama ditambah dengan newchar. Kemudian keluarkan newstring. Dilakukan juga proses newchar diisi dengan karakter pertama dari newstring. Variabel Ocode ditambah dengan variabel newchar dimasukkan dalam dictionary dan variabel Ocode diisi dengan variabel Ncode. Hal tersebut dilakukan seterusnya sampai akhir file.

Pengujian Sistem

Pengujian aplikasi dilakukan dengan melakukan testing terhadap data 40 file teks yang masing-masing terdiri dari 10 file *.doc, *.txt, *.rtf, dan *.html, dengan berbagai macam ukuran file yang berbeda. Hasil pengujian kemudian akan disajikan dalam bentuk tabel yang memperlihatkan hasil uji coba yang kemudian akan dilakukan analisa terhadap data-data tersebut. Adapun file – file yang akan digunakan dalam pengujian ini disajikan pada Tabel 1.

Tabel 1 File – File Pengujian

No	Nama File	Tipe File	Ukuran File (bytes)
1	Abstrak Iqbal	*.doc	22528
2	BAB I	*.doc	108032
3	BAB I,II,III,IV,V,VI	*.doc	2833920
4	DAFTAR ISI	*.doc	67584
5	Laporan KP	*.doc	1164288
6	Kompilasi	*.doc	75264
7	Laporan	*.doc	207360
8	Riwayat Hidup	*.doc	34304
9	Secondhand Serenade	*.doc	23552
10	Strukdat	*.doc	90624
11	Abstract	*.txt	5186
12	Coding	*.txt	63178
13	Kalkulator	*.txt	10528
14	Laporan KP	*.txt	836263
15	LAPORAN	*.txt	294495
16	Copian	*.txt	141132
17	Friend	*.txt	4296
18	Instal Linux	*.txt	12758
19	Pengertian Class	*.txt	2486
20	PERL	*.txt	383062
21	BAB IV	*.rtf	56412
22	Coding	*.rtf	41694
23	Daftar Pustaka	*.rtf	6270
24	Laporan ubie	*.rtf	3598528
25	Laporan sinul	*.rtf	1076770
26	Cara instal suse	*.rtf	18882
27	Coding LZW	*.rtf	41359
28	Perl coding	*.rtf	45088
29	Program contoh	*.rtf	41302
30	Code java	*.rtf	1879
31	ayo-kita-menganalisa-jaringan	*.html	68880
32	gawk	*.html	1934181
33	ns-man	*.html	86901
34	ping-command-di-linux	*.html	134639
35	ucb-tutorial	*.html	4828
36	2007_04_01_archive	*.html	109075
37	Boot Loader Windows NT	*.html	10568
38	Instalasi RedHat	*.html	14964
39	Instalasi Slackware	*.html	11308
40	Instalasi SuSE	*.html	20717

Analisa Hasil Pengujian Kompresi

Hasil pengujian ditampilkan dalam tabel 2 berikut ini :

Tabel 2 Pengujian File *.Doc

No	Nama File	Ukuran File Asli (bytes)	Ukuran File Terkompresi (bytes)	Rasio Kompresi
1	Abstrak iqbal	22528	8289	36,79 %
2	BAB I	108032	14984	13,87 %
3	BAB I, II, III, IV, V, VI	2833920	2336328	82,44 %

No	Nama File	Ukuran File Asli (bytes)	Ukuran File Terkompresi (bytes)	Rasio Kompresi
4	DAFTAR ISI	67584	39219	58,03 %
5	Laporan KP	1164288	942297	80,93 %
6	Kompilasi2	75264	56795	75,46 %
7	Laporan	207360	162527	78,38 %
8	Riwayat Hidup	34304	17226	50,22 %
9	Secondhand Serenade	23552	8138	34,55 %
10	Strukdat Fix	90624	64406	71,07 %

Hasil pengujian sistem berupa file *.doc yang disajikan pada Tabel 4.2 memiliki rasio rata-rata yang dicapai oleh sistem untuk file bertipe *.doc adalah sebesar 58.17%.

Tabel 3 Pengujian File *.Txt

No	Nama File	Ukuran File Asli (bytes)	Ukuran File Terkompresi (bytes)	Rasio Kompresi
1	Abstract	5186	4133	79,70 %
2	Coding	63178	22305	35,31 %
3	Kalkulator	10528	5219	49,57 %
4	Laporan KP	836263	563772	67,42 %
5	LAPORAN	294495	210923	71,62 %
6	Copian	141132	104390	73,97 %
7	Friend	4296	3456	80,45 %
8	Instal Linux	12758	9591	75,18 %
9	Pengertian Class	2486	1937	77,92 %
10	PERL	383062	28004	73,57 %

Hasil pengujian sistem berupa file *.txt yang ditunjukkan pada Tabel 4.3 memiliki rasio rata-rata yang dicapai oleh sistem adalah sebesar 68,46%.

Tabel 4 Pengujian File *.Rtf

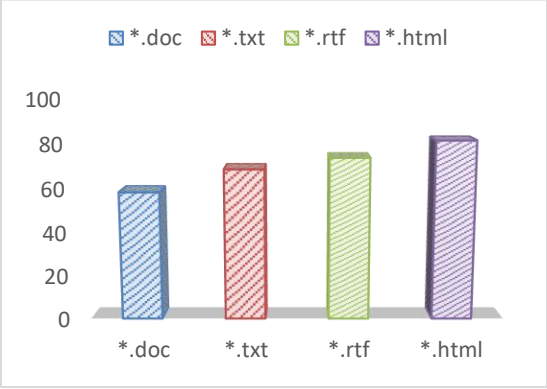
No	Nama File	Ukuran File Asli (bytes)	Ukuran File Terkompresi (bytes)	Rasio Kompresi
1	BAB IV	56412	42888	76,03 %
2	Coding	41694	24963	59,87 %
3	Daftar Pustaka	6270	5397	86,08 %
4	Laporan ubie	3598528	2887800	80,25 %
5	Laporan sinul	1076770	978270	90,85 %
6	Cara instal suse	18882	13490	71,44 %
7	Coding LZW	41359	24609	59,50 %
8	Perl coding	45088	30047	66,64 %
9	Program contoh	41302	27681	67,02 %
10	Code java	1879	1651	78,21 %

Hasil pengujian sistem berupa file *.rtf disajikan pada Tabel 4, rasio rata-rata yang dicapai oleh sistem untuk file bertipe *.rtf adalah sebesar 73,58%. Tingkat rasio kompresi untuk sebuah file sangat dipengaruhi oleh komposisi data yang bersangkutan.

Tabel 5 Pengujian File *.Html

No	Nama File	Ukuran File Asli (bytes)	Ukuran File Terkompresi (bytes)	Rasio Kompresi
1	ayo-kita-menganalisa-jaringan	68880	58454	84,86 %
2	gawk	1934181	1456887	75,32 %
3	ns-man	86901	63549	73,13 %
4	ping-command-di-linux	134639	108722	80,75 %
5	ucb-tutorial	4828	4488	92,96 %
6	2007_04_01_archive	109075	94887	86,99 %
7	Boot Loader Windows NT	10568	8657	81,92 %
8	Instalasi RedHat	14964	11688	78,11 %
9	Instalasi Slackware	11308	9429	83,38 %
10	Instalasi SuSE	20717	15543	75,03 %

Hasil pengujian sistem berupa file *.html disajikan pada Tabel 5, rasio rata-rata yang dicapai oleh sistem untuk file bertipe *.html adalah sebesar 81,24%.



Gambar 5. Diagram rasio rata-rata kompresi

Dari Gambar 5, keempat jenis file teks yang telah diujikan, file teks bertipe *.doc memiki rasio kompresi rata-rata terkecil dengan 58,17%, diikuti dengan file jenis *.txt dengan 68,46%, file jenis *.rtf dengan 73,58%, dan terakhir file jenis *.html dengan 81,24%.

Analisis Hasil Pengujian Dekompresi

Hasil uji coba untuk proses dekompresi ditunjukkan pada Tabel 6, Tabel 7, Tabel 8, dan Tabel 9. Proses dekompresi dari sistem terbukti dapat mengembalikan sebuah file *.ubi menjadi seperti file aslinya. Waktu proses yang dibutuhkan oleh sistem sebenarnya sangatlah dipengaruhi oleh komposisi data yang bersangkutan serta perangkat keras yang digunakan oleh user.

Tabel 6 Tabel Hasil Dekompresi File *.Doc

No	Nama File	Ukuran File Terkompresi (bytes)	Ukuran File Dekompresi (bytes)
1	Abstrak iqbal	8289	22528
2	BAB I	14984	108032
3	BAB I,II,III,IV,V,VI	2336328	2833920
4	DAFTAR ISI	39219	67584
5	Laporan KP	942297	1164288
6	Kompilasi2	56795	75264
7	Laporan	162527	207360
8	Riwayat Hidup	17226	34304
9	Secondhand Serenade	8138	23552
10	Strukdat Fix	64406	90624

Tabel 7 Tabel Hasil Dekompresi File *.Txt

No	Nama File	Ukuran File Terkompresi (bytes)	Ukuran File Dekompresi (bytes)
1	Abstract	4133	5186
2	Coding	22305	63178
3	Kalkulator	5219	10528
4	Laporan KP	563772	836263
5	LAPORAN	210923	294495
6	Copian	104390	141132
7	Friend	3456	4296
8	Instal Linux	9591	12758
9	Pengertian Class	1937	2486
10	PERL	28004	383062

Tabel 8 Tabel Hasil Dekompresi File *.Rtf

No	Nama File	Ukuran File Terkompresi (bytes)	Ukuran File Dekompresi (bytes)
1	BAB IV	42888	56412
2	Coding	24963	41694
3	Daftar Pustaka	5397	6270
4	Laporan ubie	2887800	3598528
5	Laporan sinul	978270	1076770
6	Cara instal suse	13490	18882
7	Coding LZW	24609	41359
8	Perl coding	30047	45088
9	Program contoh	27681	41302
10	Code java	1651	1879

Tabel 9 Tabel Hasil Dekompresi File *.Html

No	Nama File	Ukuran File Terkompresi (bytes)	Ukuran File Dekompresi (bytes)
1	ayo-kita-menganalisa-jaringan	58454	68880
2	gawk	1456887	1934181
3	ns-man	63549	86901
4	ping-command-di-linux	108722	134639
5	ucb-tutorial	4488	4828
6	2007_04_01_archive	94887	109075
7	Boot Loader Windows NT	8657	10568
8	Instalasi RedHat	11688	14964
9	Instalasi Slackware	9429	11308
10	Instalasi SuSE	15543	20717

PENUTUP

Kesimpulan yang didapatkan antara lain bahwa aplikasi kompresi file teks menggunakan algoritma LZW yang dibangun telah dapat mengkompres file menjadi ukuran lebih kecil dan juga mendekompresikannya kembali. Dari pengujian yang telah dilakukan, dengan melakukan pengkompresian terhadap beberapa jenis file teks, algoritma LZW memiliki kemampuan yang lebih baik pada kompresi file jenis *.doc. Kemudian dari hasil analisis perbandingan yang dilakukan dengan aplikasi WinRar, aplikasi kompresi algoritma LZW yang dibangun memiliki keunggulan dalam hal kecepatan, sedangkan dalam pemampatan data aplikasi WinRar memampatkan data dengan baik.

Saran untuk pengembangan lebih lanjut antara lain memperbaiki proses pembacaan karakter yang dilakukan algoritma LZW, sehingga proses kompresi mampu memampatkan semua data file dengan lebih baik. Serta menggabungkan beberapa algoritma kompresi sejenis untuk mendapatkan hasil kompresi yang lebih baik.

DAFTAR PUSTAKA

Anton. (2005). Kompresi dan Teks. Fakultas Teknik Informatika. Univesitas Kristen Duta Wacana. Diunduh di http://lecturer.ukdw.ac.id/anton/download/multime dia6.pdf.

Arisantoso, A., Harjanti, T. W., & Yulianti, S. D. (2022). Modul Pembelajaran Rekayasa Perangkat Lunak.

Darwis, D., Prabowo, R., & Hotimah, N. (2018). Kombinasi Gifshuffle, Enkripsi AES dan Kompresi Data Huffman Untuk Meningkatkan Keamanan Data. Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), 5(4), 389-394.

Hidayat, H., Pamungkas, T., & Zarman, W. (2013). Implementasi Algoritma Kompresi Lzw Pada Database Server. Komputa: Jurnal Ilmiah Komputer dan Informatika, 2(1).

Hutapea, D. (2021). Implementasi Algoritma Kriptografi Rabin Dan Lempel-Ziv-Welch (Lzw) Dalam Pengamanan Dan Kompresi File Citra. KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer), 5(1).

Laia, Y. (2016). Optimasi Rasio Kompresi Dan Kompleksitas Waktu Kompresi File Teks Menggunakan Algoritma Lempel-ZIV-Welch Dengan Fibonacci Search. Sinkron: jurnal dan penelitian teknik informatika, 1(1), 6-6.

Linawati, Henry P. Panggabean (2004). Perbandingan Kinerja Algoritma Kompresi Huffman, LZW, Dan DMC Pada Berbagai Tipe File. Universitas Katholik Parahyangan Bandung.

Mahesa, K. (2017). Rancang Bangun Aplikasi Kompresi Dan Dekompresi Pada Citra Digital Menggunakan Metode Huffman. Jurnal Processor, 12(1), 948-963.

Pratama, A. (2010). Studi Perbandingan Kinerja Algoritma Kompresi Lempel Ziv 77, Lempel Ziv 78 Dan Lempel Ziv Welch Pada File Text.

Pu, I. M. (2005). Fundamental data compression. Butterworth-Heinemann.

Putra, M. A., & Solichin, A. (2018). Optimasi Web Service dengan Penerapan Algoritma Kompresi LZW: Studi Kasus Aplikasi BluCampus Universitas Budi Luhur. SKANIKA, 1(2), 455-462.

Rinaldi, M. (2005). Diktat Kuliah IF2251: Strategi Algoritmik, Penerbit ITB. Bandung.

Roger, S. P., & Bruce, R. M. (2015). Software engineering: a practitioner's approach. McGraw-Hill Education.

Satyapratama, A., Widjianto, W., & Yunus, M. (2015). Analisis Perbandingan Algoritma Lzw Dan Huffman Pada Kompresi File Gambar Bmp Dan Png. Jurnal Teknologi Informasi: Teori, Konsep, dan Implementasi, 6(2), 69-81.

Solomon, D. (2007). Data Compression The Complete Refrerence. 4th Edition. London : Springer-Verlag.

Suharso, A., Zaelani, J., & Juardi, D. (2020). KOMPRESI FILE MENGGUNAKAN ALGORITMA LEMPEL ZIV WELCH (LZW). KOMPUTASI, 17(2), 372-380.