Implementasi Metode Cepat Kompresi File Document Otomatis Pada Aplikasi Berbasis Web Menggunakan Algoritma Additive Code

Yuza Reswan[1], Leo Tri Anggoro[2].
[1]yuzareswan@umb.ac.id, [2]leotri20177@gmail.com
[1,2, 3,4] Program Studi Teknik Informatika Fakultas Teknik Universitas Muhammadiyah Bengkulu
Jl. Bali, Po Box 118 Telp. (0736) 22756 Fax. (0736) 26161 Fakultas Teknik Universitas Muhammadiyah Bengkulu

(Received: Nopember 2024, Revised : Februari 2024, Accepied : April 2024)

Abstract- *Online data entry systems have largely replaced manual file submission in modern data transmission. Data is often very large. Therefore, a method is needed to reduce the size of the data. File compression, or compression in general, is the name of this method. Compressing files involves transforming multiple document files into a standard encoding format to reduce storage space requirements or increase upload speed. The results of this research inform the use of the Additive Code technique application for file compression, which makes it easier to upload document files automatically and is useful for users. Therefore, the author conducted research with the aim of developing a web-based application. This system can make it easier to upload document files so that it can reduce processing time. Using this method in online applications is useful for creating large document files that can be compressed into smaller ones by following the compression procedure. Data compression test results were obtained by running 10 samples of each document and PDF data file at a speed of less than 3 seconds, with accuracy ranging from 27% to 67% for each sample. We successfully tested ten Word documents and ten PDFs that were uploaded and compressed.*

Keywords: *Compression, Document, Application, Additive Code, Testing.*

Intisari- Sistem pengisian data online sebagian besar telah menggantikan pengiriman file manual dalam transmisi data modern. Data seringkali berukuran sangat besar. Oleh karena itu diperlukan suatu metode untuk memperkecil ukuran data. Kompresi file, atau kompresi secara umum, adalah nama metode ini. Mengompresi file melibatkan transformasi beberapa file dokumen ke dalam format pengkodean standar untuk mengurangi kebutuhan ruang penyimpanan atau meningkatkan kecepatan unggah. Hasil penelitian ini menginformasikan penggunaan aplikasi teknik Additive Code untuk kompresi file, yang memudahkan pengunggahan file dokumen secara otomatis dan bermanfaat bagi pengguna. Oleh karena itu, penulis melakukan penelitian dengan tujuan untuk mengembangkan aplikasi berbasis web. Sistem ini dapat memudahkan pengunggahan file dokumen sehingga dapat mengurangi waktu pemrosesan. Pemanfaatan cara ini pada aplikasi online bermanfaat untuk membuat file dokumen berukuran besar yang dapat dikompres menjadi lebih kecil dengan mengikuti prosedur kompresi. Hasil pengujian kompresi data diperoleh dengan menjalankan 10 sampel setiap dokumen dan file data PDF dengan kecepatan kurang dari 3 detik, dengan akurasi berkisar antara 27% hingga 67% untuk setiap sampel. Kami berhasil menguji sepuluh dokumen Word dan sepuluh PDF yang telah diunggah dan dikompresi.

I. PENDAHULUAN

Pada beberapa website dalam proses pengiriman file dokumen pada form aplikasi seperti website pendaftaran penerimaan CPNS dan web penerimaan mahasiswa baru pada website universitas yang menyertakan pengiriman dokumen, pengguna seringkali terkendala dengan ukuran file yang tidak sesuai dengan ukuran/kapasitas yang diperbolehkan oleh penyedia form aplikasi, serta seringkali juga website tidak menyertakan keterangan tenggang ukuran/kapasitas file yang sesuai. Oleh karena itu, pengguna harus mengubah ukuran file tersebut pada aplikasi lain. Pengguna harus membuka aplikasi yang dapat mengubah atau memadatkan ukuran/kapasitas file dokumen diluar website yang sedang diakses. Hal ini membuat pengguna membutuhkan waktu dan biaya tambahan untuk pekerjaan tersebut. Sebagai solusi dari permasalahan tersebut, semestinya form aplikasi berbasis web yang menyediakan pengunggahan file dokumen harus menyediakan fasilitas (tools) yang mampu secara otomatis dan cepat dapat mengubah ukuran file dokumen ke ukuran yg sudah ditentukan. Sehingga para pengguna website dapat bekerja lebih cepat dan efesien, dapat menghemat waktu dan biaya. Pengguna tidak perlu lagi harus membuka aplikasi lain untuk melakukan kompresi file dokumen. Sebelumnya telah dieksplorasi bagaimana menerapkan campuran algoritma Huffman dan Run Long Encoding. Selain itu, peneliti tertarik untuk menentukan rasio kompresi file yang berbeda [1]. Penggunaan teknik LZSS untuk mengompres file teks dengan mempertimbangkan faktor kecepatan dan rasio kompresi dibahas dalam penelitian lain. Teknik Kode Aditif terbukti paling efektif dalam beberapa pengujian; Oleh karena itu, saya menyarankan untuk menggunakannya dalam penelitian ini sebagai aplikasi berbasis web dalam bentuk file dokumen terkompresi [2]. Almurtada dan Syahrizal (2018) menyatakan bahwa penelitian mereka menunjukkan bahwa mengenkripsi file teks dapat mengompresinya dan mencegah orang yang tidak berwenang membukanya. Teknik ini dapat meningkatkan keamanan sekaligus mengurangi ukuran file. Perbandingan dilakukan antara keadaan sebelum dan sesudah kompresi hingga 50% dari temuan file teks [3]. Penelitian ini bertujuan untuk membuat aplikasi berbasis

web yang memudahkan pengunggahan file dokumen, menghemat waktu, sesuai dengan uraian masalah di atas.

II. TINJAUAN PUSTAKA

Konsep Algoritma Additive Code

Algoritma kode Golbach dan algoritma kode gamma Elias adalah dua metode kompresi yang digabungkan dalam pendekatan kode aditif untuk mengompres file dengan lebih efisien. Meskipun ada kemungkinan bahwa algoritma pengkodean aditif dapat mengompresi data lebih efektif dibandingkan teknik lainnya, topik ini masih banyak yang belum dieksplorasi [4].

Teori Kompresi

Kompresi data adalah proses mengubah aliran data masukan menjadi aliran data baru yang lebih kecil. Buffer dalam file atau memori adalah aliran yang bermasalah. Ada banyak metode untuk mengompresi data. Ada dua jenis metode kompresi berbasis integritas data: lossy dan lossless. Mengompresi konten (konten tinggi memperkecil ukuran tanpa mengurangi keabsahan materi) adalah salah satu metode kompresi teks [5].

Kompresi adalah proses mengubah aliran data masukan, terkadang disebut sebagai aliran sumber atau data mentah asli, menjadi aliran bit keluaran yang lebih kecil atau aliran terkompresi. Tujuan kompresi dalam ilmu komputer adalah untuk meminimalkan jumlah ruang penyimpanan yang dibutuhkan untuk data.

1. Kompresi lossy adalah jenis kompresi data yang menyisakan file terkompresi yang tidak dapat dibongkar hingga terkompresi sempurna. Ketika data yang dikompresi didekodekan lagi, beberapa data asli akan hilang karena data yang didekodekan tidak dapat dimodifikasi agar sesuai dengan data asli.
2. Kompresi lossless adalah proses kompresi data sehingga meskipun semua modifikasi telah dilakukan, file data yang dikompresi dapat dikembalikan ke keadaan semula. Jenis kompresi ini ideal untuk kompresi teks. Teknik kompresi lossless menggunakan algoritma seperti Huffman dan pengkodean kamus [6].

Penggunaan Algoritma Additive Code

File terkompresi dapat kembali ke ukuran aslinya karena skema pengkodean aditif dan lossless. Algoritma Elias Gamma Code untuk penentuan codeword dan algoritma Goldbach untuk penentuan nilai indeks merupakan dua teknik kompresi yang digabungkan dalam pendekatan kode aditif untuk memaksimalkan kompresi data [7]. Biner file dokumen harus dibaca terlebih dahulu untuk mengekstrak data heksadesimal sebelum file dapat dikompresi. Gunakan alat Binery Viewer untuk membaca data heksadesimal dari file dokumen dan mencari nilai biner di dalamnya [6].

DOCX

The first papers to use the DOC extension were those created with the word processor WordPerfect in 1980. Documents created using Microsoft Word usually have the file extension DOC, which is an abbreviation for "document". In the 1990s, Microsoft made the decision to use the *.DOC extension for its Microsoft Word word processor. Different formats are available for files ending in "DOC". Word 2007 and 2010 replaced the *.doc format with the docx extension, but Word 97 and 2003 still use files with the *.DOC extension. Therefore, it can be said that the document file storage format used by Microsoft Office is a DOCX file.

Penelitian Relevan

Pada penelitian sebelumnya yang telah dilakukan oleh Nidia Enjelita Saragih dan Fitriana Harahap dalam sebuah jurnal, telah disimpulkan bahwa Kompresi merupakan proses pengubahan sekumpulan data menjadi suatu bentuk kode untuk menghemat kebutuhan tempat penyimpanan dan waktu untuk transmisi data[1]. Sedangkan menurut Sukiman dan Chandra kompresi data atau juga dikenal sebagai pemadatan data adalah teknik yang dipakai untuk mengurangi data atau memperkecil data menjadi bentuk data lain dimana data tersebut diubah menjadi simbol yang lebih sederhana [4].[1]

Berdasarkan penelitian yang dilakukan oleh Rizki Yannur Tanjung pada tahun 2020 tentang “Perancangan Aplikasi Kompresi File Dokumen Menggunakan Algoritma Additive Code” disimpulkan bahwa hasil dari kompresi file dokumen dengan menggunakan algoritma Additive Code memiliki rasio sampai 64% [4].

Algoritma additive code merupakan algoritma yang mencakup dua algoritma kompresi yaitu algoritma golbach code dan algoritma elias gamma code yang dimana memungkinkan lebih efisien dalam mengkompres sebuah file.

III. METODOLOGI PENELITIAN

Pada saat ini, penelitian metode menggunakan algoritma additive code masih sangat sedikit padahal algoritma additive code dimungkinkan dapat lebih efisien dalam mengkompres sebuah file.

IV. HASIL DAN PEMBAHASAN

A. Hasil

Untuk membantu mempercepat si pengguna dalam mengkompresi file dokumen tanpa harus keluar dari website tersebut. Oleh karena itu, peneliti mengembangkan aplikasi berbasis web dengan

menggunakan algoritma *Addictive Code* sebagai bahasa pemrograman. Algoritma Addictive Code merupakan suatu fasilitas (tools) yang digunakan untuk mengubah atau mengkompresi ukuran suatu file berbentuk teks. Adapun tampilan dari algoritma Addictive Code adalah sebagai berikut :

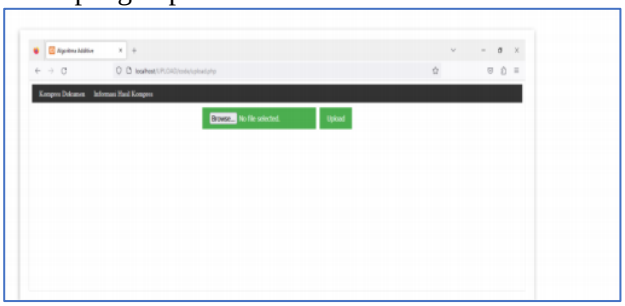
Halaman Utama

Pengguna dapat melihat atau mempelajari lebih lanjut cara kompres file dokumen menggunakan teknik Additive Code di halaman utama.

Gambar 1 Halaman Utama

Implementasi Halaman Upload

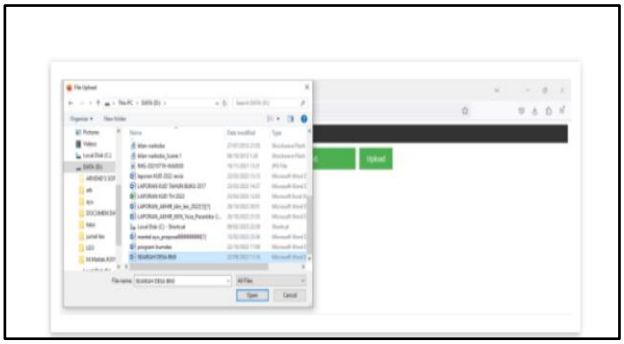
Halaman *upload* merupakan halaman dimana pengguna dapat melakukan *upload file* dokumen untuk pengompres suatu file dokumen.



Gambar 2 Halaman Upload

Implementasi Input File

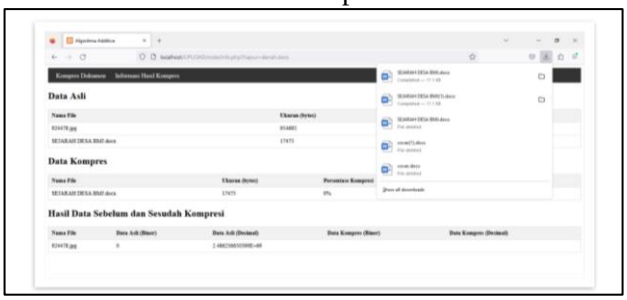
Biner file dokumen harus dibaca terlebih dahulu untuk mengekstrak data heksadesimal sebelum file dapat dikompresi. Contoh file dokumen terkompresi dan terdekompresi disediakan di bawah ini.



Gambar 3 Sampel Input File

Implementasi Informasi Hasil Kompresi

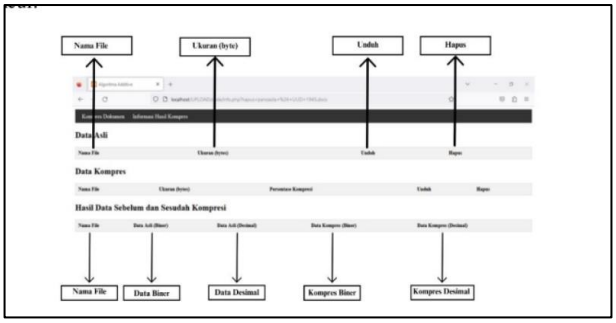
Halaman informasi hasil informasi merupakan dimana dapat melihat dan membaca kompresi tertera, dokumen yang terupload lebih awal juga dapat dihapus oleh sipengguna. Implementasi halaman informasi hasil kompresi file dokumen



Gambar 4 Halaman Informasi Hasil

Pembahasan

Temuan dan analisis pengujian harus didiskusikan agar pembaca dan pengembang dapat memahami hasilnya. Berikut pembahasan evaluasi sistem kompresi file dengan metode Additive Code. Ada banyak program kompresi yang tersedia, masing-masing dengan hasil kinerja berbeda dan pengurangan kapasitas data asli sebesar 50–75%. Kapasitas algoritma kompresi untuk mengekstraksi jumlah bit yang sama dari bahan sumber adalah penyebabnya. Dalam situasi ketika metode tersebut menggunakan algoritma Additive Code [8] Algoritma pengkodean aditif adalah salah satu teknik kompresi yang, tergantung pada jenis objek yang dikompresi, dapat meminimalkan data. Pada kode aditif, karakter yang paling sering muncul memiliki bit paling sedikit, sedangkan karakter yang paling sedikit muncul akan memiliki bit paling banyak. Algoritme kode aditif akan diterapkan dengan cara ini [9]. Setiap file yang digunakan sebagai kumpulan data untuk kompresi file telah dikompresi secara efektif dan berhasil. Hasil pengujian teknik Additive Code pada sistem kompresi file adalah sebagai berikut. Metode kompresi lossless yang disebut algoritma Additive Code dapat mengecilkan suatu file sesuai dengan karakter teks pada item yang akan dikompresi. Algoritma Kode Aditif akan digunakan berdasarkan karakter dengan bit paling sedikit yang sering muncul.



Gambar 5 Implementasi informasi hasil kompersi

 P-ISSN 1858-2680
e-ISSN 2723-4673

1. Nama file
2. Ukuran (byte)

Blok yang terdiri atas 8 bit. Ukuran byte digunakan untuk mewakili karakter seperti symbol, huruf, dan angka dalam bentuk ASCII (American Standart Code for Information). Sebagai contoh, Ketika anda mengektik angka 1, maka komputer akan menghitung sebagai 1 byte.

3. Unduh
4. Hapus
5. Data biner
6. Data yang unitnya hanya dapat mnegambil dua kemungkinan keadaan. Ini sering diberi label sebagai 0 dan 1 sesuai dengan sistem angka biner dan aljabar Boolean.
7. Data desimal
8. Beberapa Bahasa pemrograman atau kompilernya menyediakan tipe data decimal bawaan (primitife) atau Pustaka untuk mewakili pecahan decimal yang tidak beulang seperti 0,2 dan -1,17 tanpa pembulataan, dan untuk melakukan aritmatika pada bagian tersubut.
9. Kompres biner
10. Kompers biner merujuk pada proses mengurangi ukuran data dengan merepresentasukam informasi menggunakan lebih sedikit bit. Ini membantu menghemat ruang penyimpanan dan mempercepat transfer file.
11. Kompersi desimal

Pengujian Sistem

Metode pengujian perangkat lunak yang disebut pengujian "kotak hitam" berkonsentrasi pada spesifikasi program [10]. Pengembang perangkat lunak dapat merancang serangkaian kondisi masukan yang akan menjalankan setiap persyaratan fungsional suatu program dengan menggunakan pengujian black-box. Aplikasi kompresi file dokumen yang diuji disini ditunjukkan pada tabel 1.

Tabel 1. Pengujian

No	Nama Dokumen	Ukuran Sebelum Dikompres	Ukuran sesudah Dikonfresi	Presentase Hasil Kompresi
1	Dokumen 1	1300	850	65%
2	Dokumen 2	536	266	50%
3	Dokumen 3	3.781	1899	50%
4	Dokumen 4	1.176	678	58%
5	Dokumen 5	262	98	37%
6	Dokumen 6	467	115	25%
7	Dokumen 7	768	278	36%
8	Dokumen 8	983	389	40%
9	Dokumen 9	117	56	48%
10	Dokumen 10	356	102	29%
11	PDF 1	714	225	32%
12	PDF 2	958	357	37%
13	PDF 3	793	245	31%
14	PDF 4	397	110	28%
15	PDF 5	1.227	780	64%
16	PDF 6	930	338	36%
17	PDF 7	2.349	1576	67%
18	PDF 8	533	292	55%
19	PDF 9	299	82	27%
20	PDF 10	322	98	30%

Pada pengujian ini di lakukan masing-masing dengan 10 sampel setiap file data dokumen dan PDF mendapat akurasi 0% dari setipa sampel yang di ujikan. Pada saat proses kompres file dokumen kecepatan dalam pengompresan *file* yang di dapat pada pengujian ini sama karena pada saat file di upload *file* akan otomatis terkompres dengan 1 (satu) *file* kecepatan kurang dari 3 detik.

Dari pengujian 10 sampel dokumen berbetuk *word* dan 10 PDF yang berhasil teruplod dan terkompres keseluruhan berhasil teruji.

III. PENUTUP

A. Kesimpulan

Berdasarkan dari pembahasan penelitian yang telah dilakukan, maka penulis dapat menyimpulkan bahwa :

a. Setelah mengikuti prosedur kompresi dengan menggunakan algoritma *adeditive code* menghasilakan bahwa suatu *file* dokumen memiliki ukuran yang cukup besar dapat dikompres menjadi *file* dokumen yang ukuran yang lebih kecil.
b. Aplikasi kompresi *file* dokumen dengan menerapkan algoritma *additive code* menggunakan aplikasi berbasis web.

B. Saran

Pada penelitian ini, terdapat beberapa kekurangan sebagai berikut:

a. Implementasi dalam aplikasi masih dapat dikembangkan lagi untuk proses kompresi *file* dokumen .
b. Besaran *file* yang terkompres dapat diperkecilkan lebih rendah lagi.

DAFTAR PUSTAKA

[1] E. Prayoga and K. M. Suryaningrum, “Implementasi Algoritma Huffman dan Run

Length Encoding pada Aplikasi Kompresi Berbasis Web," vol. IV, no. 2, pp. 92–101, 2018.

[2] J. Pardede, M. M. B, and L. Yudhianto, "Implementasi Algoritma Lzss pada Aplikasi Kompresi dan Dekompresi File Dokumen," vol. 2, no. 1, pp. 68–81, 2017.

[3] I. Almurtada, M. Syahrizal, "Penerapan Algoritma Goldbach Codes Pada Kompresi File Teks Terenkripsi Hill Cihper," Vol. 6, No. April, Pp. 473–478, 2018.

[4] E. Hariska et al., "Perancangan Aplikasi Kompresi File Gambar Menggunakan Algoritma Additive Code," vol. 5, pp. 193–202, 2021, doi: 10.30865/komik.v5i1.3671.

[5] R. D. Pratiwi, S. D. Nasution, and F. Fadlina, "Perancangan Aplikasi Kompresi File Teks Dengan Menerapkan Algortima Fixed Length Binary Encoding (Flbe)," *J. Media Inform. Budidarma*, vol. 2, no. 1, pp. 10–14, 2018, doi: 10.30865/mib.v2i1.813

[6] R. Y. Tanjung, "Perancangan Aplikasi Kompresi File Dokumen Menggunakan Algoritma Adiitive Code," vol. 8, no. 4, pp. 108–113, 2021, doi: 10.30865/jurikom.v8i4.3593.

[7] W. D. Apriliani, " Perancangan Aplikasi Kompresi File Video Dengan Menggunakan Algoritma Additive Code," Vol.5, no. 1, pp.203-212, 2021, DOI: 10.30865/komik.v5i1.3673

[8] G. A. Nabila, "Analisis Perbandingan Algoritma Boldi Vigna Z1 Code Dan Algoritma Even Rodeh Code Pada Kompresi File Teks," Universitas Sumatera Utara Medan, 2019.

[9] R. Krasmala, A. B. Purba, U. T. Lenggana, and T. Informatika, "Kompresi Citra Dengan Menggabungkan Metode Discrete Cosine Transform (DCT) dan Algoritma Huffman," ISSN 2527-9165, vol. 2, no. 1, pp. 1–9, 2017.

[10] F. C. Ningrum, D. Suherman, S. Aryanti, H. A. Prasetya, dan A. Saifudin, "Pengujian Black Box pada Aplikasi Sistem Seleksi Sales Terbaik Menggunakan Teknik Equivalence Partitions," *Jurnal Informasi Univiversitas Pamulang*, vol. 4, no. 4, hlm. 125–130, 2019