Penerapan Metode Huffman dalam Kompresi Data

Moch. Lazuardi Imani[1], Rani Rotul Muhima[2], dan Siti Agustini[3]
Teknik Informatika Institut Teknologi Adhi Tama Surabaya[1,2,3]
e-mail: lazuardiimani988@gmail.com

ABSTRACT

Data compression refers to a science presenting more concise information to reduce total bits spent for saving and sending data. This research employed Huffman algorithm for data compression to produce more spaces for storage. Huffman method has advantages in compressing text data. This research applied text data having the sizes of 1MB to 10 MB, 11MB to 20 MB, and 21MB to 30MB. The results of compression through Huffman method demonstrated that the file was successfully compressed more than 45%. In addition, the ratio of test obtained the ranges of 46.53% as the lowest value and 47.08% as highest one.

Keywords: *Data teks, Kompresi, Metode Huffman.*

ABSTRAK

Kompresi data adalah ilmu yang menyajikan informasi dalam bentuk yang lebih ringkas. Tujuan dari kompresi adalah untuk mengurangi jumlah bit yang digunakan untuk menyimpan dan mengirim data. Pada penelitian ini Algoritma Huffman diterapkan dalam kompresi dengan tujuan akan menghasilkan lebih banyak ruang dalam penyimpanan dan metode Huffman sangat unggul dalam pengompresian data teks. Data yang digunakan adalah data teks yang berukuran kurang atau sama dengan 1MB sampai 10 MB, 11MB sampai 20 MB, dan 21MB sampai 30MB. Hasil pengujian kompresi dengan menggunakan metode Huffman menunjukkan file yang berhasil dimampatkan lebih dari 45% ukuran sebelum dikompresi. Ratio pengujian memiliki range antara 46,53% untuk nilai terendah dan 47,08% untuk nilai tertinggi.

Kata kunci: *Data teks, Kompresi, Metode Huffman*.

PENDAHULUAN

Dalam proses transmisi data terdapat beberapa kendala yang umumnya dihadapi, salah satu diantaranya adalah durasi pengiriman yang memakan waktu karena besarnya ukuran dari data yang akan ditransmisi. Dalam sebuah file teks terdapat kumpulan strings yang berisi banyak karakter di dalamnya yang selalu menimbulkan masalah di perangkat penyimpanan yang terbatas dan durasi dalam proses transmisi data. Hal ini dapat diatasi dengan kompresi data sehingga ukuran file menjadi lebih kecil dan dapat mempercepat proses pengiriman data [1].

Kompresi data dapat menghasilkan ukuran file yang lebih kecil dari ukuran asli file dan tidak ada informasi data yang hilang selama proses kompresi, sehingga algoritma kompresi data yang dibutuhkan adalah metode yang bersifat *lossless* yang mana setelah proses dekompresi, jumlah bit (byte) data atau informasi dalam keseluruhan file akan sama seperti file semula. Kompresi dengan sifat *lossless* salah satunya adalah menggunakan Algoritma Huffan [5].

Algoritma Huffman menggunakan pengkodean yang serupa dengan sandi morse. Algoritma ini termasuk klasik dan sering digunakan dalam pengompresian file berbentuk teks [7]. Prinsip kode Huffman didasarkan pada karakter yang paling sering muncul dan dikodekan dengan kode yang jumlah bitnya lebih sedikit, sedangkan karakter yang jarang muncul dikodekan dengan kode yang jumlah bitnya lebih panjang[8]. Kode Huffman di kenal sebagai kode yang digunakan untuk melakukan proses kompresi data. Kode ini dibentuk dengan memanfaatkan frekuensi kemunculan string karakter ASCII pada data yang akan dikompresi. Kode yang diperoleh ialah berupa bilangan bit biner yang kemudian diubah ke dalam karakter yang sesuai dengan bilangan biner tersebut [9].

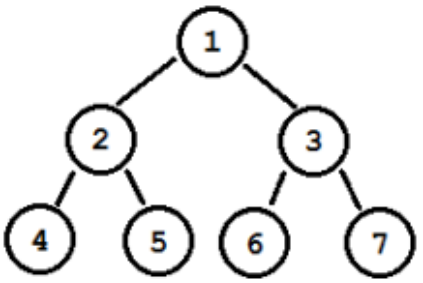
Kelebihan dari penggunaan algoritma ini adalah file yang dikompresi tidak mengalami perubahan ketika dilakukan proses dekompresi sehingga cocok digunakan untuk kompresi data teks [2]. Penelitian terkait kompresi data teks pernah dilakukan oleh Pujianto, *et al* (2020) yang membandingkan Metode Huffman dan Run Length Encoding pada kompresi document mendapatkan hasil bahwa kompresi data menggunakan Metode Huffman lebih baik karena terdapat penurunan ukuran data yang signifikan. Hasil serupa juga diuji oleh Pattiasina (2012) yang menyimpulkan bahwa Algoritma Huffman dapat melakukan kompresi teks sekitar 70% jika dibandingkan dengan menggunakan kode ASCII dan sekitar 25% jika dibandingkan dengan menggunakan 3 bit kode [4]. Berdasarkan pengujian dan analisa dari penelitian Sunardi, *et al* (2019) menunjukkan bahwa dalam kompresi menggunakan Metode Huffman semakin besar variasi data maka kapasitas hasil kompresi juga semakin besar, besarnya prosentase rasio hasil kompresi berbanding lurus dengan kapasitas file [6].

Berdasarkan latar belakang diatas penelitian ini bertujuan untuk mengaplikasikan Metode Huffman dalam pengompresian data teks.

TINJAUAN PUSTAKA

Pohon Biner

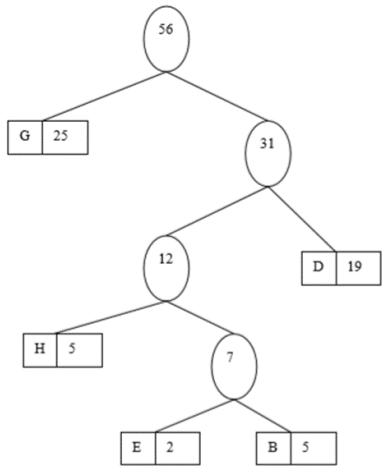
Pohon biner merupakan pohon berakar dimana setiap simpul cabangnya mempunyai paling banyak dua buah anak. Pohon biner mengandung suatu struktur data yang terdiri dari simpul-simpul atau nodes yang terhubung satu sama lain. Masing-masing simpul di dalam suatu pohon dapat memiliki atau tidak memiliki anak atau orangtua. Tiap simpul dapat memiliki maksimal satu orangtua, sedangkan jumlah anak yang dapat dimiliki suatu simpul adalah tidak terbatas [3]. Contoh dari pohon biner disajikan pada Gambar 1.



Gambar 1. Pohon Biner

Algoritma Huffman

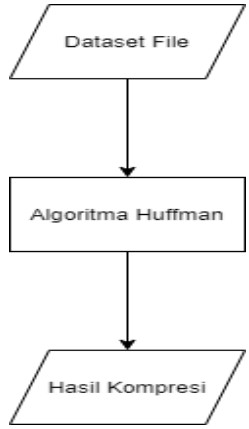
Algorithma merupakan prosedur dan langkah untuk melakukan perhitungan, algoritma juga digunakan untuk melakukan perhitungan, perosesan data dan penalaran secara otomatis [10]. Algoritma Huffman termasuk dalam metode *Lossless Compression* yang memanfaatkan kenyataan bahwa tidak semua byte muncul dengan frekuensi yang sama dalam suatu data. Dengan begitu, penghematan ukuran data dapat dicapai memvariasikan panjang kode yang mewakili simbol-simbol di dalam data tersebut. Simbol yang lebih sering ditemukan dikodekan dengan jumlah bit yang lebih sedikit. Hal ini dapat dicapai dengan beberapa langkah diantaranya yang pertama adalah membuat *list* frekuensi kemunculan setiap symbol, kemudian dari daftar frekuensi tersebut dapat dibentuk Pohon Biner Huffman yang dilampirkan pada Gambar 2 [3].



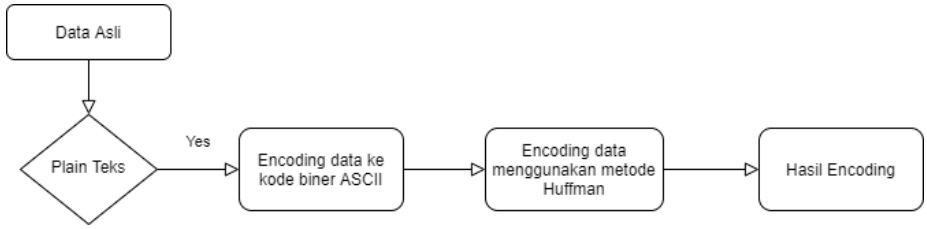
Gambar 2. Contoh Pohon Biner Huffman

METODE

Penerapan Metode Huffman untuk kompresi data teks dilakukan pengujian dengan besar data awal 1MB sampai 10 MB, 11MB sampai 20 MB, 21MB sampai 30MB. Jumlah file adalah 3 file dengan format MS.Word untuk setiap ukuran data file. Bahasa pemrograman yang digunakan adalah java NetBeans. *Flowchart* dan alur pengujian disajikan pada Gambar 3 dan 4.



Gambar 3. *Flowchart* Gambaran Umum Sistem.



Gambar 4. Proses *Encoding*

Proses encoding dan decoding dimulai dari menginputkan data yang akan di kompresi, memulai proses *encoding* data teks ke kode biner ASCII, *encoding* data teks ke Metode Huffman, membandingkan hasil pengompresian dengan data asli, kemudian rasio hasil kompresi.

HASIL DAN PEMBAHASAN

Data Proses Kompresi dan Dekompresi

Dalam tahap awal *kompresi* diperlukan sebuah data untuk melakukan proses pemampatan data. Dalam penelitian ini digunakan 3 varian data berbentuk "*.Txt*". Variasi ukuran data file (1MB sampai 10 MB, 11MB sampai 20 MB, 21MB sampai 30 MB). Jumlah file adalah 3 file untuk masing masing variasi. Berikut adalah ukuran data pada masing-masing variasi yang disajikan dalam Tabel 1, 2, dan 3.

Pada Tabel 1, 2, dan 3 menyajikan data yang digunakan dalam proses kompresi dengan metode Huffman. Pada tabel 4.1 menunjukkan varian pertama dengan ukuran file 4,36 MB, 8 MB, dan 10,1 MB. Tabel 4.2 menampilkan ukuran data pada varian kedua yaitu file dengan ukuran 14,5 MB, 18,1 MB, dan 20,3 MB. Pada tabel 4.3 berisi ukuran data pada varian ketiga dengan ukuran file 21 – 30 MB, diantaranya 24 MB, 28,3 MB, dan 30,5 MB.

Tabel 1. Varian 1 (1MB sampai 10 MB)

Nama File	Ukuran File
V1D1	4,36
V1D2	8
V1D3	10,1

Tabel 2. Varian 2 (11MB sampai 20 MB)

Nama File	Ukuran File
V2D1	14,5
V2D2	18,1
V2D3	20,3

Tabel 3. Varian 3 (21MB sampai 30 MB)

Nama File	Ukuran File
V3D1	24
V3D2	28,3
V3D3	30,5

Hasil Uji Kompresi

Pengujian pertama dilakukan untuk mendapatkan rasio kompresi. Hasil kompresi data ditunjukkan pada Tabel 4. Berdasarkan Tabel 4 dengan 9 ukuran data didapatkan hasil kompresi file dengan ratio tertinggi pada V3D1 sebesar 47,08% dan ratio terendah pada V1D3 sebesar 46,53%. V1D1 dengan file asli 4,36 MB menjadi 2,31 MB dengan ratio kompresi sebesar 47,02%. V1D2 dengan file asli 8 MB menjadi 4,24 MB dengan ratio kompresi sebesar 47%. V2D1 dengan file asli 14,5 MB menjadi 7,71 MB dengan ratio kompresi sebesar 46,83%. V2D2 dengan file asli 18,1 MB menjadi 9,64 MB dengan ratio kompresi sebesar 46,74%. V2D3 dengan file asli 20,3 MB menjadi 10,8 MB dengan ratio kompresi sebesar 46,8%. V3D2 dengan file asli 28,3 MB menjadi 15 MB dengan ratio kompresi sebesar 47%. V3D3 dengan file asli 30,5 MB menjadi 16,2 MB dengan ratio kompresi sebesar 46,89%.

Tabel 4. Hasil Kompresi Data

Nama File	Ukuran File Asli (MB)	Ukuran File Kompres (MB)	Ratio (%)
V1D1	4,36	2,31	47,02
V1D2	8	4,24	47
V1D3	10,1	5,40	46,53
V2D1	14,5	7,71	46,83
V2D2	18,1	9,64	46,74
V2D3	20,3	10,8	46,8
V3D1	24	12,7	47,08
V3D2	28,3	15	47
V3D3	30,5	16,2	46,89

Hasil Uji Dekompresi

Setelah didapat hasil kompresi, selanjutnya dilakukan proses dekompresi. Dekompresi adalah proses pengembalian data ke bentuk original. Dalam pengujian ini, hasil dekompresi data dengan file yang berjenis teks ditunjukan pada Tabel 5.

Tabel 5. Hasil Dekompresi Data

Nama File	Ukuran File Kompres (MB)	Ukuran File Dekompres (MB)
V1D1	2,31	4,36
V1D2	4,24	8
V1D3	5,40	10,1
V2D1	7,71	14,5
V2D2	9,64	18,1
V2D3	10,8	20,3
V3D1	12,7	24
V3D2	15	28,3
V3D3	16,2	30,5

Berdasarkan Tabel 5 dengan 9 ukuran data didapatkan hasil dekompresi file yang sesuai dengan ukuran file original sebelum dikompresi.

Analisa Sistem

Hasil pengujian kompresi dengan menggunakan metode Huffman menunjukkan file yang berhasil dimampatkan lebih dari 45% ukuran sebelum dikompresi. Ratio pengujian memiliki range antara 46,53% untuk nilai terendah dan 47,08% untuk nilai tertinggi. Nilai ini cukup terutama dalam mengkompresi file berukuran besar misalnya berukuran di atas 10 MB. Dari hasil tersebut juga menunjukkan bahwa persentase kompresi atau dekompresi file tidak bergantung pada ukuran file melainkan bergantung pada isi data pada file tersebut. Semakin banyak perulangan data yang terdapat pada bagian chunk data file maka rasio kompresi akan semakin rendah. File yang telah dikompresi bila dilakukan proses kompresi sekali lagi maka ukuran file tidak akan tereduksi banyak bahkan hampir tidak ada perbedaan, karena Huffman merupakan *optimal compression* jadi file yang dilakukan kompresi sebanyak dua kali, maka proses terakhir tidak akan mereduksi ukuran file lagi.

KESIMPULAN

Berdasarkan pengujian dan evaluasi Implementasi Algoritma Huffman untuk Kompresi Data Teks, maka dapat ditarik kesimpulan bahwa hasil pengujian kompresi dengan menggunakan

metode Huffman menunjukkan file yang berhasil dimampatkan lebih dari 45% ukuran sebelum dikompresi. Ratio pengujian memiliki range antara 46,53% untuk nilai terendah dan 47,08% untuk nilai tertinggi.

DAFTAR PUSTAKA

[1] Siahaan, Adyansyah Putera Utama, "Implementasi Teknik Kompresi Teks Huffman," Jurnal Informatika., vol. 10, no. 2, Juli 2016.

[2] Pujianto *et al*., "Perbandingan Metode Huffman dan Run Length Encoding Pada Kompresi Document, " *INFOTEKJAR: Jurnal Nasional Informatika Dan Teknologi Jaringan.*, vol. 5 no.1, September 2020.

[3] Zuhriyah, Sitti, "Penggunaan Pohon Biner Huffman untuk Kompresi Data Teks," *Jurnal IT STMIK Handayani.*, vol. 15, Desember 2014.

[4] Pattiasina, Timothy John, "Analisa Kode Huffman untuk Kompresi Data Teks," *TEKNIKA.*, vol. 1, no. 1, Juli 2012.

[5] Suchendra, Devie R. dan Sandra Wulandari, "Implementasi Kompresi Data Text Menggunakan Huffman Coding," *Jurnal LPKIA.*, vol. 1, no. 1, Desember 2012.

[6] Sunardi, Samsu Alam, dan Suci Rahma Dani, "Implementasi Aplikasi Kompresi Data dengan Metode Huffman Code," *PROSIDING SEMINAR ILMIAH SISTEM INFORMASI DAN TEKNOLOGI INFORMASI.*, vol. 8, no. 2, Agustus 2019.

[7] Komala, Margaret Lydia, dan Maria A. K., "Analisis Kinerja Kompresi untuk Data Bertipe Teks dengan Menggunakan Algoritma Lossless Compression," *Seminar Nasional Riset dan Inovasi Teknologi (SEMNAS RISTEK)*, Januari 2017.

[8] Yansyah, Dedek Andri., "Perbandingan Metode Punctured Elias Code dan Huffman pada Kompresi File Text," *Jurnal Riset Komputer (JURIKOM)*, Desember 2015.

[9] Pahdi, Akhmad., "Algoritma Huffman dalam Pemampatan dan Enksripsi Data," *Indonesian Journal on Networking and Security*, vol. 6, no. 3, 2017.

[10] Latifah, Fitri., "Penerapan Algorithma Pohon untuk Operasi Pengolahan dan Penyimpanan Data Dalam Teknik Pemrograman (Kajian Algorithma Pohon pada Teknik Pemrograman)," *Jurnal Techno Nusa Mandiri*, vol. 13, no. 2, September 2016.