Jurnal Riset Komputer (JURIKOM), Vol. 3 No. 6, Desember 2016
ISSN 2407-389X (Media Cetak)
Hal : 42-44

IMPLEMENTASI ALGORITMA HUFFMAN DAN LZ78 UNTUK KOMPRESI DATA

Gusri Indah Yana[1], Rivalri Kristianto Hondro[2]

Mahasiswa Teknik Informatika STMIK Budi Darma Medan[1]
Dosen Tetap STMIK Budi Darma Medan[2]
Jl. Sisingamangaraja No. 338 Simpang Limun Medan[12]

ABSTRAK

Data adalah sesuatu yang belum mempunyai arti bagi penerimanya dan masih memerlukan adanya suatu pengolahan. Data bisa berujut suatu keadaan, gambar, suara, huruf, angka, matematika, bahasa ataupun simbol-simbol lainnya yang bisa kita gunakan sebagai bahan untuk melihat lingkungan, obyek, kejadian ataupunsuatu konsep. Besarnya ukuran data yang harus dikirim melampaui kecepatan transmisi yang dimiliki oleh perangkat keras yang ada, sehingga masih terdapat delay time yang relatife besar. Selain media penyimpanan seperti floppy disk, hard disk, dan compact disk mempunyai kapasitas yang terbatas. Ada banyak teori dan algoritma untuk kompresi data, diantaranya metode Huffman, Run- Legth Encoding (RLE), Lempel-Zip-Weleh (LZW), Shanon – Fano, dan beberapa algoritma lainnya.

Kata Kunci : Kriptografi, Algoritma Huffman, Algoritma LZ78, Kompresi, Dekompresi, Data.

I. PENDAHULUAN

Kompresi data dalam konteks ilmu komputer merupakan ilmu yang menampilkan informasi dalam bentuk yang pendek. Kompresi data bertujuan untuk mengurangi jumlah *bit* yang digunakan untuk menyimpan atau mengirim informasi. Berbagai jenis kompresi data secara umum digunakan pada komputer pribadi, termasuk kompresi pada program biner,data, suara, dan gambar. (Nelson,1996).

Ada dua teknik yang dapat dilakukan dalam melakukan kompresi data yaitu *Lossless Compression* dan *Lossy Compression. Lossless Compression* merupakan kompresi data dimana hasil dekompresi dari data yang terkompresi sama dengan data aslinya dan tidak ada informasi yang hilang. Sedangkan *Lossy Compression* adalah kompresi data dimana hasil dekompresi dari data yang terkompresi tidak sama dengan data aslinya karena ada informasi yang hilang, tetapi masih dapat ditolerir oleh persepsi mata. Jenis kompresi ini digunakan untuk menyimpan *database, spreadsheet,* atau untuk memproses *file* teks. Ada banyak sekali teori dan algoritma kompresi data, diantaranya algoritma *Huffman, Run-Legth Encoding (RLE), Lempel-Zip-Welch (LZW), Lempel-Zip-78 (LZ78), Shanon-Fano*, dan beberapa algoritma lainnya.

Lempel-Ziv-78 adalah algoritma kompresi *loseless* universal yang diciptakan Abraham Lempel, Jacob Ziv, 1978. Algoritma ini dirancang untuk cepat dalam implementasi tetapi biasanya tidak optimal karena hanya melakukan kompresi dengan hanya menggunakan *dictionary,* dimana *fragmen – fragmen* teks digantikan dengan indeks yang diperoleh dari suatu kamus.

Data adalah sesuatu yang belum mempunyai arti bagi penerimanya dan masih memerlukan adanya suatu pengolahan. Data bisa berujut suatu keadaan, gambar, suara, huruf, angka, matematika, bahasa ataupun simbol-simbol lainnya yang bisa kita gunakan sebagai bahan untuk melihat lingkungan, obyek, kejadian ataupunsuatu konsep.

Teks adalah bahasa yang berfungsi, maksudnya adalah bahasa yang sedang melaksanakan tugas tertentu (menyampaikan pesan atau informasi) dalam konteks situasi, berlainan dengan kata-kata atau kalimat-kalimat lepas yang mungkin dituliskan di papan tulis. Hal penting mengenai sifat teks ialah bahwa meskipun teks itu bila kita tuliskan tampak seakan-akan terdiri dari kata-kata dan kalimat, namun sesungguhnya terdiri dari makna-makna. Memang makna-makna atau maksud yang ingin kita sampaikan kepada orang lain haruslah dikodekan dalm tuturan lisan atau kalimat-kalimat supaya dapat dikomunikasikan.

II. TEORITIS

A. Algoritma Huffman

Algoritma Huffman adalah suatu algoritma kompresi tertua yang disusun oleh David Huffman pad tahun 1952. Algoritma tersebut digunakan untuk membuat kompresijenis Lossy compression, yaitu pemanpaatan data dimana tidak ada satu byte data yang hilang sehingga data tersebut utuh dan disimpan sesuai dengan aslinya.

Tahapan proses kompresi algoritma Huffman antara lain :

1. Hitung banyaknya jenis karakter dan jumlah dari masing masing karakter yang terdapat dalam sebuah *file*.
2. Susun setiap jenis karakter dengan urutan jenis karakter yang jumlahnya paling sedikit ke yang jumlahnya paling banyak.
3. Buat pohon biner berdasarkan berdasarkan urutan karakter dari yang jumlahnya terkecil ke yang terbesar dan member kode untuk tiap karakter.
4. Ganti data yang ada dengan kode bit berdasarkan pohon biner.
5. Simpan jumlah bit untuk kode bit yang terbesar, jenis karakter yang diurutkan dari frekuensi keluarnya terbesar ke terkecil beserta data yang sudah berubah menjadi kode bit sebagai data hasil kompresi.

B. Algoritma LZ78

Algoritma Lempel Ziv 78 (LZ78), dikembangkan 1 tahun setelah LZ77, yaitu pada tahun 1978. Oleh Abraham Lempel dan Jacob Ziv. Karena dipublikasikan oleh pihak yang sama. Algoritma LZ78 dikembangkan dengan cara membuat suatu *dictionary* yang dapat menyimpan pola secara permanen selama proses pengkodean berlangsung. *Token* yang digunakan sebagai *index* untuk menemukan pola tersebut, berkurang dari 3 menjadi 2 buah. Sebuah *token* dalam LZ78 berbentuk (f,c). Dimana f adalah *pointer* yang merujuk pada *dictionary* dimana pola yang ditemukan dan c adalah karakter pertama sesudah pola.

III. ANALISA DAN PEMBAHASAN

Algoritma Huffman dikembangkan oleh David Huffman, seorang mahasiswa MIT. Jika ditinjau dari segi teknik pengkodean karakter yang digunakan, algoritma Huffman ini termasuk dalam algoritma yang menggunakan metode *symbolwise*. Yang dimaksud dengan metode *symbolwise* adalah suatu metode yang menghitung probabilitas kemunculan suatu karakter dalam suatu waktu. Kode Huffman merupakan *prefix code* yang berisi himpunan *bit* biner dari suatu karakter yang tidak mungkin menjadi awalan (*prefix*) dari himpunan *bit* biner karakter lainnya.

Langkah-langkah pembentukan pohon Huffman adalah sebagai berikut :

1. Baca semua karakter di dalam teks untuk menghitung frekuensi kemunculan setiap karakter. Setiap karakter penyusun teks dinyatakan sebagai pohon bersimpul tunggal. Setiap simpul di-*assign* dengan frekuensi kemunculan karakter tersebut.
2. Terapkan strategi algoritma *greedy* sebagai berikut : gabungkan dua buah pohon yang mempunyai frekuensi terkecil pada sebuah akar. Setelah digabungkan akar tersebut akan mempunyai frekuensi yang merupakan jumlah dari frekuensi dua buah pohon-pohon penyusunnya.
3. Ulangi langkah 2 sampai hanya tersisa satu buah pohon Huffman. Agar pemilihan dua pohon yang akan digabungkan berlangsung cepat, maka semua yang ada selalu terurut menaik berdasarkan frekuensi.

Sebuah file yang akan dimampatkan berisi karakter-karakter “MUSLIM”, huruf yang sama hanya di tuliskan 1 kali, sehingga menjadi

M = 4DH = 01001101B
U = 55H = 01010101B
S = 53H = 01010011B
L = 4CH = 01001100B
I = 49H = 01001001B

Maka jika diubah dalam rangkaian bit, “MUSLIM” menjadi berukuran 48 bit :

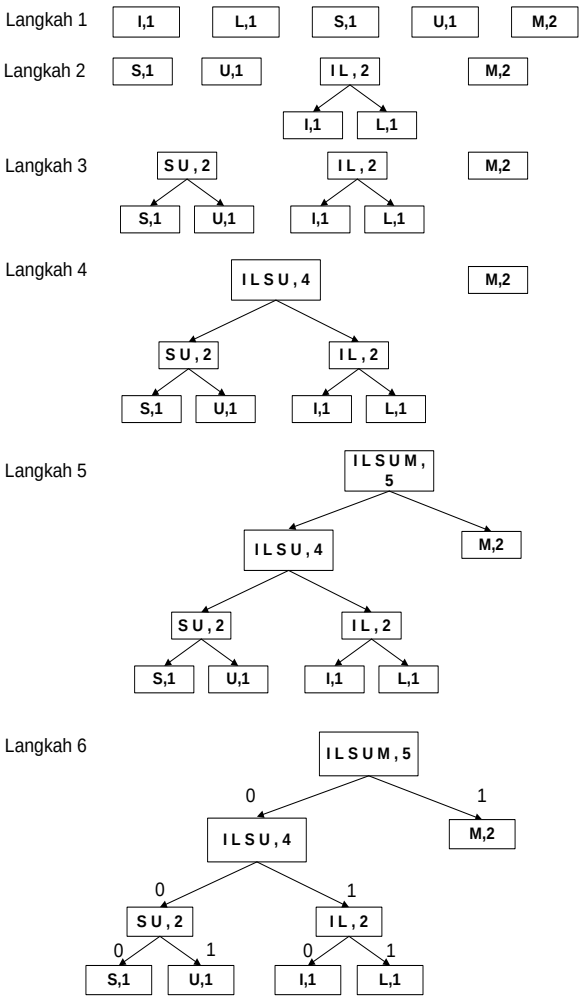
01001101 01010101 01010011
01001100 01001001
01001101

M U S L
I M

Pada *string* di atas, frekuensi kemunculan M = 2, U = 1, S = 1, L = 1 dan I = 1,

Langkah – langkah pembentukan *encoding* pohon Huffman

Gambar 1: Langkah Pembentukan Encoding

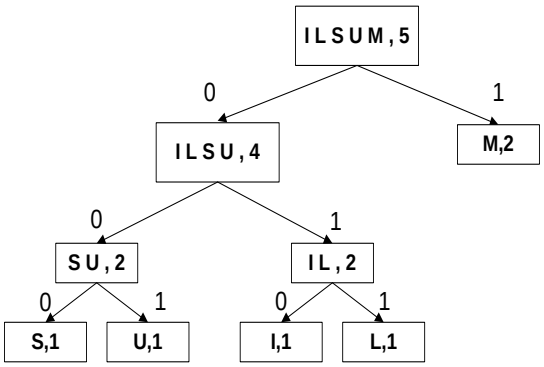


Sehingga hasil dari *encoding* untuk Huffman adalah sebagai berikut :

I = 010
L = 011
S = 000
U = 001
M = 1

Langkah – langkah pembentukan *decoding* pohon Huffman

Gambar 2: Langkah Pembentukan Decoding



IV. IMPLEMENTASI

A. Tampilan Menu Utama

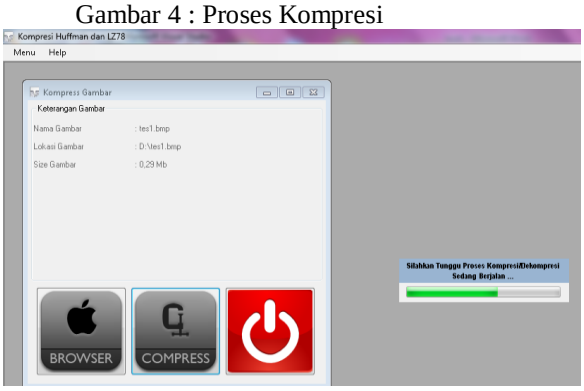
From ini digunakan untuk menampilkan menu utama Kompresi Huffman dan LZ78, Adapun gambarnya dapat dilihat dibawah ini :

Gambar 3: MenuUtama

B. Proses Kompresi

Form ini berisikan proses kompresi file gambar yang sedang berjalan, adapun gambarnya dapat dilihat dibawah ini :

Gambar 4 : Proses Kompresi



C. Proses Dekompresi

Form ini berisikan proses Encoding untuk Kompresi gambar, adapun gambarnya dapat dilihat dibawah ini :

Gambar 5 : Proses Dekompresi

Setelah proses input data dilakukan maka teks akan keluar dan kapasitas dari teks tersebut juga keluar. Langkah selanjutnya yang dilakukan adalah menekan tombol kompresi dan dekompresi untuk mengkompresi dan mendekompresi**.**

V. KESIMPULAN

Berdasarkan analisa dan pembahasan yang dilakukan, maka dapat disimpulkan sebagai berikut :

1. Algoritma Kompresi data dapat digunakan untuk mengurangi penggunaan memory yang terlalu besar pada saat menyimpan data, juga untuk mempercepat proses pengiriman data.
2. Untuk Kompresi dalam bentuk Teks, Algoritma huffman lebih tepat digunakan daripada algoritma LZ78.
3. Algoritma huffman dan LZ78 dapat digunakan dengan menyesuaikan type data yang akan dikompresi.
4. Algoritma Huffman tidak tepat digunakan untuk citra dengan type GIF.

VI. DAFTAR PUSTAKA

1. Hari Antoni Musril (2012), ISSN : 2086-4981, Volume : V, Studi Komparasi Metode *Aritmhatic Coding* & *Coding* dalam Algoritma *Entropy* untuk Kompresi Citra Digital.
2. Rinaldi Munir, "**Kriptografi**" Penerbit Informatika, Bandung, 2006.
3. Fuja Suhastra, " Implementasi Algoritma Kompresi Lempel Ziv ", 2010.
4. Sugiarti Yuni, " Analisis & Perancangan *UML* ", 2013.
5. Josua Marinus, " Studi Perbandingan Algoritma Huffman & Shanon –fano, 2014.
6. Riyanto, M.Z., 2007. *Pengamanan Pesan Rahasia Menggunakan Algoritma Kriptografi Elgamal Atas Grup Pergandaan Zp**. Skripsi. Yogyakarta : Universitas Gajah Mada.