ANALISIS PERBANDINGAN KINERJA ALGORITMA KOMPRESI LZW MENGGUNAKAN BINARY SEARCH TREE BASED DICTIONARY DAN ALGORITMA KOMPRESI HUFFMAN PADA CIPHERTEXT ENKRIPSI RIJNDAEL

Harold Setiawan[1], Maman Abdurohman[2], Ema Rachmawati[3]

[1]Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

---

Abstrak

Algoritma Lempel Ziv Welch (LZW) menggunakan teknik dictionary dalam melakukan kompresi, dictionary merupakan sebuah kamus referensi untuk menyimpan berbagai macam kombinasi karakter.namun dictionary yang bersifat array based memiliki kelemahan dalam hal memasukkan string dan pencarian string secara brute force untuk itu maka diterapkan Binary Search Tree pada dictionary LZW. Kedua metode memiliki persamaan yaitu sama-sama melakukan proses inisialisasi 256 karakter ASCII. Sedangkan Algoritma Huffman menggunakan teknik statistik dalam melakukan kompresi. Fase pertama yang dilakukan adalah menghitung frekuensi kemunculan simbol kemudian membuat sebuah pohon biner yang berisikan urutan kemunculan simbol dari yang terbanyak sampai yang paling sedikit. Kode Huffman dari setiap simbol adalah output kompresi berupa rangkaian biner yang dibaca dari daun sampai akar yang bersangkutan.

Pada tugas akhir ini akan menganalisis kinerja dua algoritma kompresi yaitu Huffman dan LZW Binary Search Tree dalam proses kompresi maupun dekompresi sebuah file plaintext dan ciphertext Rijndael, dengan mengamati parameter rasio kompresi dan waktu kompresi dan dekompresi yang dibutuhkan.

Berdasarkan hasil penelitian yang dilakukan maka dapat disimpulkan bahwa penerapan struktur dictionary berupa binary search tree pada algoritma LZW masih belum optimal jika dibandingkan terhadap algoritma Huffman.

Kata Kunci : Kompresi, LZW, Binary Search Tree, Dictionary, Huffman, rasio, waktu

---

Abstract

The Lempel Ziv Welch (LZW) Algorithm used dictionary technique to compress data. Dictionary is the source to keep many kind of symbols combination, but the array based dictionary has weakness in searching and inserting string by brute force way so that Binary Search Tree is implemented to the LZW's dictionary. Both of that two methods have similarity which implement initialization process to 256 ASCII's characters. Whereas The Huffman Algorithm use statistic technique to compress data. The first phase of Huffman is counting the frequency of each simbol appearance then making binary tree which contains sequence of each simbol from the most till the least.The Huffman code for each simbol is compression output that kind of code chain read from leaf till its root.

This final assignment will analyze the performance of two compression algorithm which are Huffman Algorithm and LZW Binary Search Tree Algorithm in compressing and decompressing process of plaintext and Rijndael ciphertext file by observing parameters, ratio, compression time and decompression time.

According to the result of this experiment, it can be concluded that implementing binary search tree structure on LZW algorithm's dictionary has not yet optimal if compared with Huffman algorithm.

Keywords : compression, LZW, Binary Search Tree, Dictionary, Huffman, ratio, time

---

1. Pendahuluan

1.1 Latar belakang

Kompresi merupakan proses untuk memampatkan atau mengecilkan ukuran data. Dengan data yang telah dikompresi akan memudahkan proses penyimpanannya dalam media dengan *space* yang sangat terbatas dan mempersingkat waktu pengiriman data. Berdasarkan *output* nya kompresi dapat dibedakan menjadi dua, yaitu kompresi *lossless* dan *lossy*. Perbedaan antara keduanya adalah, kompresi *lossless* memungkinkan untuk mengembalikan data hasil kompresi tepat sama dengan data sebelum kompresi, sebaliknya kompresi *lossy* tidak dapat mengembalikan data hasil kompresi tepat sama dengan data sebelum kompresi.

Untuk mengamankan data yang sangat penting dan harus dirahasiakan maka dilakukan proses enkripsi atau pengacakan data. Data multimedia adalah data *hypermedia* yang dapat diamankan lewat proses enkripsi. *Ciphertext* enkripsi multimedia yang berukuran besar memerlukan *space* penyimpanan yang besar dan waktu pengiriman akan lebih lama, sehingga diperlukan kompresi untuk mengatasi masalah tersebut. Mengingat data yang akan dikompresi bernilai sangat penting maka algoritma kompresi yang diterapkan harus menjamin keakuratan data hasil dekompresi tepat sama dengan data sebelum kompresi, untuk itu sangat tepat bila digunakan metode kompresi *lossless*.

Algoritma kompresi Lempel Ziv Welch (LZW) adalah salah satu contoh algoritma kompresi *lossless* yang menggunakan teknik *dictionary* dalam proses kompresinya. Algoritma LZW bersifat *array based dictionary* yang mana setiap karakter *string* digantikan oleh kode tabel yang dibuat setiap ada *string* yang masuk. Ukuran tabel *dictionary* pada algoritma LZW asli adalah 4096 sampel (2^{12}) dimana 256 sampel pertama digunakan untuk tabel karakter tunggal (*extended ASCII*) dan sisanya digunakan untuk pasangan karakter atau *string* dalam data *input*.

Algoritma kompresi Huffman adalah contoh algoritma kompresi *lossless* lainnya yang menggunakan teknik statistik dalam proses kompresinya. Algoritma Huffman melakukan dua kali pembacaan pada file yang akan dikompresi yaitu, pertama untuk menghitung frekuensi kemunculan karakter dalam pembentukkan Pohon Huffman dan kedua, untuk mengkodekan simbol dalam Kode Huffman.

Algoritma LZW yang bersifat *array based dictionary* memiliki kelemahan saat memasukkan *string* dan pencarian *string* dalam *dictionary* yang bersifat *brute force* untuk mengatasi masalah tersebut kita dapat menggunakan *binary search tree based dictionary*.

Dalam pembuatan tugas akhir, penulis akan membandingkan kinerja Algoritma LZW setelah diterapkannya *binary search tree based dictionary* terhadap kinerja Algoritma Huffman dalam mengkompresi file *ciphertext* enkripsi Rijndael. Hipotesis awal dari tugas akhir ini adalah dengan menerapkan *binary search tree based dictionary* pada Algoritma LZW akan meningkatkan kemampuan rasio kompresi, waktu kompresi dan waktu dekompresinya.

1.2 Perumusan masalah

Berdasarkan latar belakang permasalahan diatas, maka dapat dirumuskan masalah pokok sebagai berikut :

1. Bagaimana menerapkan *binary search tree based dictionary* pada Algoritma LZW.
2. Bagaimana analisis perbandingan algoritma LZW *binary search tree based dictionary* terhadap algoritma Huffman untuk menentukan mana yang paling baik kinerjanya dengan meneliti rasio kompresi, waktu kompresi dan waktu dekompresi.

1.3 Tujuan

Tujuan yang ingin dicapai adalah :

1. Menerapkan *binary search tree based dictionary* pada Algoritma LZW.
2. Menganalisis perbandingan rasio kompresi, waktu kompresi dan waktu dekompresi Algoritma LZW *binary search tree based dictionary* terhadap Algoritma Huffman.

1.4 Batasan masalah

Dalam implementasi tugas akhir ini, akan dilakukan pembatasan pada beberapa hal sebagai berikut :

1. Tugas akhir ini tidak membahas analisis kriptografi.
2. Algoritma kriptografi yang digunakan adalah Rijndael AES-128.
3. File uji yang digunakan berupa file gambar dalam format raw.

1.5 Metodologi Penyelesaian Masalah

Pengerjaan tugas akhir ini menggunakan metodologi :

1. **Studi Literatur**
   Studi literatur akan dilakukan pada seluruh proses pengerjaan tugas akhir.

2. **Pengumpulan Data**
   Pada tahap ini, dilakukan pengumpulan file uji.

3. **Analisis dan Perancangan Sistem**
   Analisis terhadap keseluruhan sistem meliputi perancangan sistem, kebutuhan perangkat keras dan lunak untuk pembangunan sistem.

4. **Implementasi Sistem**
   Implementasi fungsionalitas enkripsi-dekripsi Rijndael, kompresi-dekompresi dengan algoritma LZW binary search tree based dictionary dan algoritma Huffman.

5. **Pengujian Sistem**
   Sistem akan diuji coba untuk mengenkripsi file uji kemudian mengkompresi *ciphertext*, mendekompresinya lalu mendekripsi file tersebut dan terakhir melakukan analisis terhadap hasil pengujian.

6. **Penyusunan Laporan**
   Pada tahap ini, bertujuan untuk mendokumentasikan setiap tahap pembuatan tugas akhir mulai dari landasan teori sampai dengan kesimpulan dari penelitian yang dilakukan.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Dari hasil penelitian tugas akhir ini maka dapat ditarik beberapa kesimpulan sebagai berikut :

1. Tidak terjadi peningkatan rasio kompresi setelah *dictionary* LZW diubah menjadi *binary search tree*. Berdasarkan pengukuran rasio kompresi diperoleh nilai rasio yang sama antara LZW *array* dan LZW *binary search tree* .

2. Waktu rata-rata kompresi algoritma LZW dengan *binary search tree based dictionary* menjadi lebih cepat daripada LZW *array* berdasarkan hasil pengukuran dengan *input* berupa *ciphertext* Rijndael namun waktu kompresi rata-rata algoritma LZW *binary search tree* menjadi lebih lama daripada LZW *array* saat pengukuran dilakukan dengan input berupa *plaintext*.

3. Waktu rata-rata dekompresi algoritma LZW dengan *binary search tree based dictionary* menjadi lebih cepat daripada LZW *array* berdasarkan hasil pengukuran dengan input berupa *plaintext* namun waktu rata-rata dekompresi LZW *binary search tree* pada pengujian *ciphertext* memberikan hasil yang berbeda. Dimana waktu rata-rata dekompresi menjadi lebih cepat berdasarkan pengukuran waktu rata-rata pada file uji ke-17 sd 30 dan lebih lambat berdasarkan pengukuran waktu rata-rata pada file uji ke-1 sd 16.

5.2 Saran

Dari hasil penelitian tugas akhir ini, penulis menyarankan beberapa hal untuk penelitian lebih lanjut, sebagai berikut :

1. Karena ukuran output kompresi algoritma LZW ditentukan oleh pemilihan bit dictionary untuk merepresentasikan output maka perlu diteliti bagaimana cara penanganan bit dictionary agar diperoleh output dengan rasio kompresi yang lebih tinggi.
2. Melakukan pengujian dengan jenis file yang lebih beragam seperti file dokumen, file suara, file program dan lain-lain dengan jumlah sampel yang lebih banyak agar hasil pengukuran lebih akurat.

Daftar Pustaka

[1] Daemen John, Rijmen Vincent, 2000, "The Design of Rijndael. AES – The Advanced Encryption Standard". Springer.

[2] Delfs Hans, Knebl Helmut, 2007, "Intorduction to Cryptography – Principles and Applications 2nd edition". Springer.

[3] Hariyanto Bambang, 2008. "Rekayasa Sistem Berorientasi Objek". Informatika.

[4] Haryono Arif, Heryanto Imam, Raharjo Budi, 2009. "Mudah belajar JAVA". Informatika.

[5] Munir Rinaldi, 2006. "Kriptografi". Informatika.

[6] Ngoen Thompson Susabda, 2003, "AVL Tree". Universitas Bina Nusantara Fakultas Ilmu Komputer. Diunduh pada tanggal 20 Mei 2011.

[7] Sanjaya Dwi, 2001. "Berpetualang dengan Struktur Data di Planet Pascal". J & J Learning.

[8] Sayoud Khalid, 2000. "An Introduction to Data Compression 2nd Edition". Academic Press.

[9] Sayoud Khalid, 2003. "Lossless Data Compression Handbook". Academic Press.

[10] Setiawati Yoshitho Andrian, 2008. "Implementasi Algoritma Kriptografi Anubis dan Rijndael untuk Penyandian Data". IT Telkom.

[11] Solomon David, 2004. "Data Compression". Academic Press.

[12] Yuniati Voni, Indriyanta Gani, C. Antonius Rachmat, 2009. "Enkripsi dan Dekripsi Dengan Algoritma AES 256 Untuk Semua Jenis File". Universitas Kristen Duta Wacana. Diunduh pada tanggal 20 Mei 2011.