BAB II

TINJAUAN PUSTAKA

Terdapat beberapa literatur yang mengulas kembali algoritma JBE (Sadiq, et al., 2013; Sharma, et al., 2014; Singla & Kumar, 2014; Adewumi, 2015), namun belum pernah terdapat literatur yang memuat pengembangan atau modifikasi algoritma JBE. Bagian ini akan meninjau kembali beberapa penelitian terdahulu yang berhubungan dengan tema penelitian ini untuk mendapatkan gambaran perspektif mengenai kompresi data, BWCA dan JBE.

A. Sejarah Kompresi Data

Model awal dari kompresi data dikembangkan pada pertengahan abad ke-18. Pada tahun 1820-an Louis Braille mengembangkan sistem penulisan yang menggunakan pola titik-titik timbul untuk menuliskan karakter di atas kertas. Tertarik dengan *Barbiers Night Writing* yang diperkenalkan oleh Kapten Charles Barbier de la Serre disekolahnya, Braille mengoptimalkan sistem Barbier yang terdiri dari 6x2 titik menjadi 3x2 titik untuk mempermudah pembacaan kode dengan menggunakan jari (Abualkishik & Omar, 2008 ; Priyanto & Nur, 2014). Masing-masing dari enam titik dalam grup (mungkin datar atau timbul), menyiratkan bahwa kandungan informasi dari suatu grup adalah setara dengan enam bit atau dimungkinkan untuk membuat 64 kombinasi yang berbeda. Oleh karena total jumlah huruf, angka, dan tanda baca yang umum tidak mencapai 64, sehingga bagian yang tersisa dapat digunakan untuk mengkodekan kata-kata yang

sering digunakan seperti *and*, *for*, *from* dan string umum seperti *ound*, *asi*, dan *th* (Sayood, 2006; David, 2007)

Model kompresi lainnya hadir di abad ke-19 yaitu kode Morse yang dirancang oleh Samuel Morse pada tahun 1932-1943 untuk pengiriman pesan melalui telegraf. Menurut David (2006), Samuel Morse menggunakan pendekatan *variable-size codes* dalam perancangannya (David, 2006). Secara lebih jelas, Sayood (2006) dan David (2007), menjelaskan bahwa huruf-huruf yang lebih sering digunakan dalam pesan berbahasa Inggris, dikodekan oleh Morse dengan urutan yang lebih pendek seperti e (·) dan a (· -), dan huruf-huruf yang jarang digunakan, dikodekan dengan urutan yang lebih panjang seperti q (- - · -) dan j (· - - -). Pendekatan tersebut bertujuan untuk mengurangi rata-rata waktu yang diperlukan dalam mengirim pesan (Sayood, 2006; David, 2007).

Lebih dari 100 tahun setelah kode Braile dan kode Morse diperkenalkan, tepatnya pada tahun 1948, metode kompresi data mengalami peningkatan yang signifikan dengan hadirnya Teori Informasi yang dirumuskan oleh Claude E. Shannon. Karya Claude Shannon ini memberikan cara formal untuk mengukur informasi dan ketidakpastian yang merupakan dasar untuk kompresi data (Steinruecken, 2014). David (2007) berpendapat bahwa dua konsep dari Teori Informasi yaitu entropi dan redundansi, merupakan dasar penting untuk penerapan *variable-lenght codes* untuk kompresi data (David, 2007).

Pengkodean data berdasarkan probabilitas pertama kali dirancang pada tahun 1949 oleh Claude E. Shannon dan Robert Fano yang dikenal sebagai algoritma Shannon-Fano, namun metode yang optimal untuk melakukan hal ini ditemukan

kemudian oleh David A. Huffman pada tahun 1951 (Rathore, et al., 2013; Ambadekar, et al., 2015). Menurut Jain, et al., (2013), ide Huffman adalah menggantikan *fixed-length codes* seperti *American Standard Code for Information Interchange* (ASCII) dengan *variable-length codes* (Jain, et al., 2013). Suchendra & Wulandari (2012) menyatakan bahwa prinsip pengkodean Huffman mirip dengan kode Morse, yaitu tiap karakter (simbol) dikodekan hanya dengan rangkaian beberapa bit, dimana karakter yang sering muncul dikodekan dengan rangkaian bit yang pendek dan karakter yang jarang muncul dikodekan dengan rangkaian bit yang lebih panjang (Suchendra & Wulandari, 2012).

Pada akhir 1970-an, Abraham Lempel dan Jacob Ziv menerbitkan algoritma inovatif mereka untuk kompresi data yaitu LZ77 dan LZ78 yang menggunakan kamus. Lebih khususnya, LZ77 menggunakan kamus dinamis yang disebut *sliding window* (R, et al., 2014).

B. Penelitian Tentang BWCA dan JBE

Apabila data yang akan dikompresi diatur terlebih dahulu, maka data hasil pengaturan tersebut akan dapat meningkatkan rasio kompresi. Sebagai contoh, misalkan data teks berisikan string "AABCCBBBAAAA" dengan kapasitas 12 *byte*, jika tanpa diatur dan dikompresi dengan rle, maka akan menghasilkan string "2A1B2C3B4A" dengan kapasitas 10 *byte*. Tapi jika string tersebut diatur dengan mengurutkannya berdasarkan simbol sehingga menghasilkan string "AAAAAABBBBCC" dan dikompresi dengan rle, maka akan menghasilkan string "6A4B2C" dengan kapasitas 6 *byte*.

Cara pengurutan seperti contoh di atas tentu saja tidak reversibel. Pada tahun 1994, Burrows dan Wheeler (Burrows & Wheeler, 1994), mempublikasikan makalah mereka yang menunjukkan cara pengurutan reversibel yang mirip seperti contoh di atas. Cara yang diusulkan ini dikenal sebagai transformasi Burrows-Wheeler atau disingkat BWT (Chatterjee, 2013).

Dengan mengutip pernyataan Seward (Seward, 2007), Karkkainen (2007) menyatakan bahwa dalam melakukan komputasi BWT biasanya membutuhkan ruang dan waktu yang lebih banyak dibandingkan dengan langkah selanjutnya pada BWCA (Karkkainen, 2007). Dalam penelitiannya, Chatterjee (2013) membandingkan kinerja BWT dengan beberapa algoritma lainnya yaitu pengkodean Lempel dan Ziv (LZ), *Prediction by Partial Matching* (PPM) dan *Context Mixing* (CM), sebagaimana dapat dilihat dalam tabel 2.1 berikut (Chatterjee, 2013):

Tabel 2.1 Kinerja BWT Dibandingkan Algoritma Lainnya

Class	*Introduced In*	*Compr. Effectivity*	*Compr. Speed*	*Decompr. Speed*	*Memory Usage*
LZ	1977	*Average*	*High*	*Very high*	*Low*
BWT	1994	*Good*	*Average*	*Average*	*Average*
PPM	1984	*Very good*	*Low*	*Low*	*High*
CM	2002	*Best*	*Very low*	*Very low*	*Very high*

Berdasarkan tabel 2.1 dapat dikatakan bahwa efisiensi dari BWT tergolong dalam kategori rata-rata dan efektifitas dari BWT tergolong baik.

Wiseman (2007) menyatakan bahwa untuk mengurangi waktu eksekusi agar efisiensi dari BWT dapat ditingkatkan, biasanya data yang akan dikompresi dengan BWT dibagi terlebih dahulu menjadi beberapa blok. Namun dengan mengurangi

ukuran blok dapat menyebabkan kinerja kompresi menjadi buruk karena pengulangan string akan menjadi lebih berkurang sehingga efektifitas dari BWT menjadi menurun (Wiseman, 2007).

Burrows dan Wheeler juga mengevaluasi pengaruh ukuran blok terhadap kinerja kompresi dalam paper mereka, dan menyatakan bahwa keuntungan masih dapat diperoleh apabila ukuran blok diperbesar. Dan tentu saja, keuntungan masih akan diperoleh apabila file yang dikompresi memiliki ukuran yang lebih besar dari ukuran blok (Adjeroh, et al., 2008).

Agar dapat berjalan pada semua jenis komputer yang beredar dengan berbagai ragam spesifikasinya, aplikasi kompresi data bzip2 (Seward, 2007) mengimplementasikan BWCA dengan menggunakan ukuran blok antara 100 KB – 900 KB. Namun dalam papernya, Abel (2005) menyatakan bahwa umumnya ukuran blok BWCA berada dalam kisaran 1 MB – 10 MB (Abel, 2005).

Untuk meningkatkan rasio kompresi dari BWCA banyak penelitian yang mengembangkan algoritma untuk memproses output BWT dengan memodifikasi atau menggantikan MTF seperti *Move One From Front* (M1FF), *Move One From Front Two* (M1FF2), *Incremental Frequency Count* (IFC), *Distance Coding* (DC) dan *Inversion Frequencies* (IF). M1FF merupakan modifikasi dari MTF yang diusulkan oleh Balkenhol, et al (Balkenhol , et al., 1999). Berbeda dengan MTF yang mengubah simbol masukan dari posisi yang tinggi ke posisi awal, M1FF bekerja dengan mengubah simbol masukan dari posisi pertama atau kedua dalam daftar dipindahkan ke posisi pertama dan masukan dari posisi yang lebih tinggi dipindahkan ke posisi kedua (Syahrul, et al., 2011).

Menurut Fenwick (2007), data output hasil BWT dan MTF sering didominasi oleh simbol nol (umumnya sekitar 60%) dan umumnya simbol tersebut terletak secara berurutan (Fenwick, 2007). Melihat kesesuaian antara algoritma yang diusulkannya dengan output dari kombinasi BWT dan MTF, dalam publikasinya Suarjaya (2012) mengusulkan penggunaan algoritmanya dengan menggunakan skema RLE + BWT + MTF + JBE + AC (Suarjaya, 2012).

Dengan hanya menggunakan satu file percobaan, Ambadekar et al. (2015) menguji kinerja dari tiga algoritma kompresi data yaitu RLE, Lempel Ziv Welch (LZW) dan AC, tanpa dikombinasikan dengan JBE dan dikombinasikan dengan JBE. Berdasarkan hasil yang diperoleh, diyatakan bahwa kombinasi algoritma LZW dan JBE menghasilkan rasio kompresi yang lebih baik (Ambadekar, et al., 2015).

Grewal dan Kaur (2014), dalam penelitiannya mengusulkan kombinasi kriptografi dan kompresi data untuk pengamanan pesan pada file video atau audio dengan menggabungkan teknik kriptografi dan kompresi data yaitu *Advanced Encryption Standard* (AES) dan JBE (Grewal & Kaur, 2014).

Oleh karena penelitian tentang pengembangan atau modifikasi algoritma JBE hingga saat ini belum pernah diterbitkan maka dalam penelitian ini diusulkan modifikasi algoritma JBE.