Aplikasi Graf dan Algoritma Viterbi pada Proses *Decoding* Sinyal yang di-*Encode* Menggunakan Kode Konvolusi dalam Komunikasi Digital

Jonet Wira Murti 13518083
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13518083@std.stei.itb.ac.id

***Abstraksi*—Perkembangan teknologi telekomunikasi semakin pesat, semakin banyak pula data yang ditransmisi dari suatu tempat ke tempat lain. Untuk itu, perlu sebuah metode untuk mendeteksi atau mengoreksi kerusakan yang terjadi pada data yang dikirim. Salah satu caranya adalah menggunakan metode pengkodean, yaitu metode kode konvolusi. Data yang telah di-*encode* perlu di-*decode* sehingga kembali ke bentuk semula. Beberapa cara *decoding* adalah dengan menerapkan algoritma terhadap data yang di-*encode,* salah satunya adalah algoritma Viterbi yang mengimplementasikan konsep graf dalam proses *decoding* sebuah data atau sinyal.**

***Kata Kunci*—kode konvolusional, algoritma viterbi, *parity bit*, *state machine*.**

I. Pendahuluan

Perkembangan teknologi telekomunikasi yang semakin pesat berbanding lurus dengan semakin banyaknya jumlah data yang ditransmisikan dari suatu tempat ke tempat lainnya. Tidak jarang dari banyaknya data yang dikirim, sebagian informasinya rusak, sehingga data yang kemudian diterima pada tempat tujuan tidak sesuai dengan yang telah dikirim.

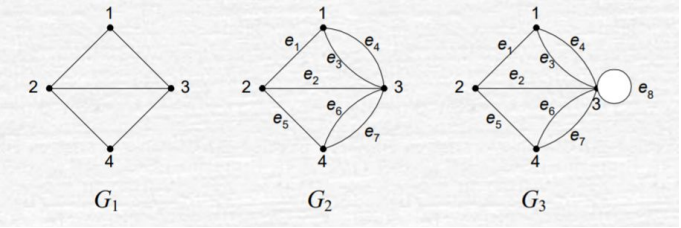
Dalam mengatasi hal tersebut dibutuhkan suatu metode untuk meminimalisasi kerusakan yang terdapat pada data yang ditransmisikan. Salah satu caranya adalah dengan menerapkan sistem pengkodean. Dengan sistem pengkodean, data dikirim dengan menambahkan sejumlah bit redundan atau *parity bit* pada bagian akhir data, sehingga ketika data telah sampai pada tempat tujuan, kerusakan data dapat dianalisis berdasarkan *parity bit* yang telah ditambahkan pada data. Salah satu teknik pengkodean yang cukup terkenal dalam bidang telekomunikasi adalah metode pengkodean konvolusional.

Ketika data yang dikirim menggunakan metode pengkodean tersebut telah sampai pada titik tujuan, maka diperlukan suatu proses pemecahan kode atau *decoding* terhadap data. Terdapat beberapa algoritma yang digunakan untuk melakukan proses *decoding* data yang di-*encode* menggunakan metode kode konvolusi, salah satunya adalah menggunakan algoritma Viterbi. Algoritma ini merupakan sebuah algoritma yang mengaplikasikan graf dalam proses pemecahan kode yang masuk ke dalam *decoder*, sehingga data hasil *decoding* tadi merupakan representasi data yang paling mendekati dengan data asli yang dikirim.

II. Landasan Teori

A. Graf

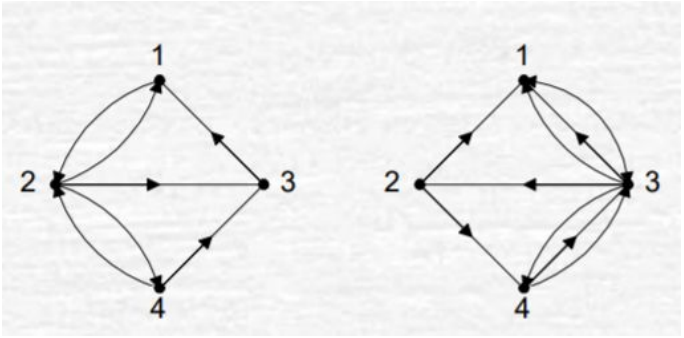
Graf merupakan sebuah struktur atau model (G = (V, E)), yang terdiri dari V yang merupakan himpunan tidak kosong dari simpul (*vertex*) dan E yang merupakan himpunan sisi (*edge*). graf biasa digunakan dalam merepresentasikan objek diskrit dan keterhubungan dari tiap-tiap objek tersebut.



Gambar 1. Jenis-jenis graf tak-berarah
sumber :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf

Graf diklasifikasikan berdasarkan ada atau tidaknya gelang dan sisi ganda serta berdasarkan orientasi arah pada sisi-sisi graf. Berdasarkan ada atau tidaknya gelang dan sisi ganda, graf dibagi menjadi graf sederhana, graf ganda, dan graf semu. Graf sederhana merupakan jenis graf yang tidak mengandung sisi ganda dan gelang. Graf ganda merupakan jenis graf yang mengandung sisi ganda, yaitu dua atau lebih sisi yang sama-sama menghubungkan dua simpul A dan B, namun tidak mengandung gelang. Graf semu atau *pseudograph* merupakan jenis graf yang mengandung gelang, yaitu sisi yang menghubungkan simpul yang sama. Berdasarkan orientasi arah

pada sisi, graf dibagi menjadi graf tak-berarah dan graf berarah. Graf tak-berarah merupakan jenis graf yang sisi-sisinya tidak memiliki orientasi arah. Graf berarah merupakan jenis graf yang setiap sisinya memiliki orientasi arah.



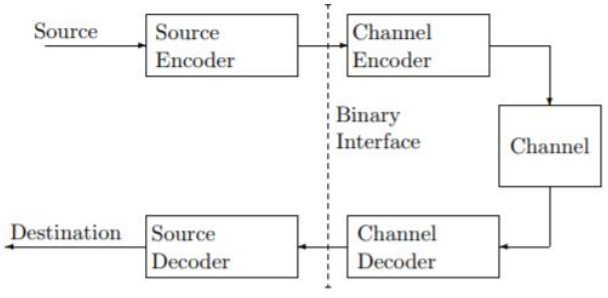
Gambar 2. Jenis-jenis graf berarah
sumber :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf

Dalam pembahasan mengenai graf, terdapat beberapa terminologi terkait dengan graf itu sendiri. Terminologi-terminologi tersebut adalah sebagai berikut.

1. Ketetanggaan (*Adjacent*)
   Dua buah simpul *u* dan *v* dalam sebuah graf G dikatakan bertetangga jika *u* dan *v* merupakan titik-titik ujung dari sebuah sisi *e* pada G.
2. Bersisian (*Incidency*)
   Sebuah sisi *e* pada graf G dikatakan bersisian dengan simpul *u* jika *u* merupakan salah satu ujung dari *e*.
3. Simpul Terpencil (*Isolated Vertex*)
   Simpul terpencil merupakan simpul yang tidak memiliki sisi yang bersisian dengannya.
4. Graf Kosong (*Null Graph* atau *Empty Graph*)
   Graf kosong merupakan graf G = (V, E), dengan E merupakan himpunan kosong, yaitu himpunan sisinya merupakan himpunan kosong.
5. Derajat (*Degree*)
   Derajat dari sebuah simpul pada graf tak-berarah merupakan jumlah sisi yang bersisian dengan graf tersebut. Untuk sisi yang membentuk gelang pada suatu simpul, maka sisi tersebut bernilai dua derajat bagi simpul tersebut. Derajat suatu simpul dituliskan sebagai $d(v)$. Pada graf berarah, $d_{in}(v)$ merupakan jumlah sisi yang memiliki titik ujung (*terminal*) *v*, sedangkan $d_{out}(v)$ merupakan jumlah sisi dengan *v* sebagai titik pangkal (*initial*).
6. Lintasan (*Path*)
   Lintasan merupakan barisan berselang antara simpul dengan sisi dengan panjang n, simpul awal v_0, dan simpul akhir v_n, dituliskan seperti $v_0, e_1, v_1, e_2, v_2, e_3, \ldots, e_n, v_n$, sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, ……, $e_n = (v_{n-1}, v_n)$ adalah sisi-sisi pada graf. Panjang lintasan merupakan jumlah sisi dalam lintasan tersebut.
7. Siklus (*Cycle*) atau Sirkuit (*Circuit*)
   Siklus atau sirkuit merupakan lintasan yang berawal dan berakhir pada simpul yang sama.
8. Terhubung (*Connected*)
   Dua simpul v_i dan v_j dikatakan terhubung jika terdapat lintasan antara dua simpul tersebut. Pada graf tak-berarah, graf disebut graf terhubung jika untuk setiap pasang simpul v_i dan v_j terdapat lintasan dari v_i ke v_j . Jika terdapat pasangan simpul yang tidak terhubung pada graf tersebut, maka disebut graf tak-terhubung. Pada graf berarah, graf disebut terhubung jika graf tak-berarahnya terhubung. Graf berarah disebut terhubung kuat jika terdapat lintasan berarah dari *u* ke *v* dan lintasan berarah dari *v* ke *u*. Jika tidak ada tetapi graf tak-berarahnya terhubung, maka disebut graf terhubung lemah.
9. Upagraf (*Subgraph*) dan Komplemen Upagraf
   Sebuah graf G1 = (V1, E1) merupakan upagraf dari graf G = (V, E) jika V1 adalah himpunan bagian dari V dan E1 adalah himpunan bagian dari E. Komplemen dari upagraf G1 terhadap G adalah graf G2 = (V2, E2) sedemikian sehingga E2 = E - E1 dan V2 merupakan himpunan simpul yang anggota-anggota E2 bersisian dengannya.
10. Upagraf Rentang (*Spanning Subgraph*)
    Graf G1 = (V1, E1) dikatakan sebagai upagraf rentang dari G jika V1 = V, yaitu G1 mengandung semua simpul dalam G.
11. Graf Berbobot (*weighted Graph*)
    Graf berbobot merupakan graf yang setiap sisinya diberi nilai atau bobot.

B. Komunikasi Digital

Sistem komunikasi digital merupakan sebuah sistem komunikasi yang memanfaatkan data atau sinyal dalam bentuk biner sebagai sumber atau informasi dari sistem komunikasi tersebut. Sinyal biner atau digital ini didapat dari proses konversi sinyal analog maupun diskrit menjadi sinyal dalam bentuk biner dengan menggunakan *source encoder*, dan akan kembali diubah seperti semula ketika telah mencapai tujuan menggunakan *source decoder*.



Gambar 3. Skema Komunikasi Digital
sumber : http://www.mit.edu/~6.450/handouts/6.450book.pdf

Dalam proses transmisi sinyal digital, data yang dikirim dapat rusak karena adanya gangguan (*noise*). Kerusakan ini bisa terjadi hanya pada satu bit dari data biner tersebut atau pada beberapa bit yang terdapat dalam data. Untuk memperkecil kemungkinan tersebut, dilakukan berbagai cara, salah satunya dengan metode pengkodean sinyal. Dalam metode pengkodean sinyal, sinyal digital akan melalui proses *encoding*, salah prosesnya yaitu dengan menambahkan beberapa bit yang redundan ke dalam sinyal yang akan ditransmisikan, kemudian di-*decode* pada bagian penerima sinyal. Beberapa metode pengkodean sinyal yang digunakan adalah metode kode blok dan kode konvolusi.

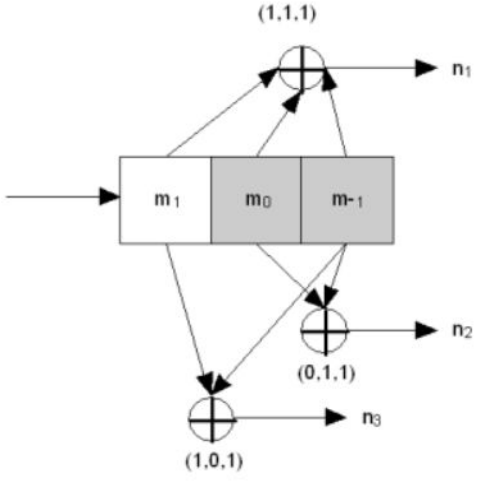
C. Kode Konvolusi

C.1 Definisi Kode Konvolusi

Kode konvolusi (*convolutional code*) merupakan salah satu kode yang digunakan dalam proses pengkodean sinyal digital. Kode konvolusi mengubah data dalam bentuk biner menjadi sebuah data baru dalam bentuk biner juga yang disebut sebagai *codeword*. Metode kode konvolusi digunakan untuk mengatasi atau meminimalisasi kerusakan terhadap data yang ditransmisi. Ketika telah sampai di tempat tujuan data yang dikirim akan di-*decode*. Salah satu cara men-*decode* data yang di-*encode* menggunakan kode konvolusi adalah dengan menerapkan algoritma Viterbi.

Layaknya proses pengkodean sinyal secara umum, proses *encoding* dari kode konvolusi juga menerapkan transmisi bit redundan atau *parity bit*. Namun, perbedaan mendasarnya dengan teknik pengkodean lain adalah pada kode konvolusi, data yang ditransmisikan bukan merupakan data awal dengan penyisipan *parity bit*, namun yang ditransmisikan adalah *parity bit* itu sendiri. Pembentukan *parity bit* dipengaruhi oleh data sumber yang akan ditransmisi.

Dalam merancang sebuah kode konvolusi untuk meng-*encode* sinyal digital, kita perlu menentukan *constraint length* (dinotasikan dengan K) dan jumlah *parity bit* (r) yang akan ditransmisikan ke sistem penerima.



Gambar 4. Skema Kode Konvolusi
sumber : wikipedia.org

Gambar 4 merupakan contoh skema kode konvolusi dengan jumlah *parity bit* 3 dan *constraint length* 3.

C.2 Cara Kerja Sederhana Kode Konvolusi

Secara sederhana, kode konvolusi bekerja dengan cara memproses satu-persatu bit dari data yang masuk ke *encoder*, kemudian mengeluarkan sebanyak r *parity bit* untuk ditransmisikan ke penerima, begitu seterusnya hingga bit pada data sumber habis. Dalam membentuk r bit data untuk ditransmisikan, digunakan r buah fungsi *parity* yang masing-masing bergantung pada nilai bit dari data sumber sebanyak maksimal K (*constraint length*) bit dimulai dari bit ke-n (bit posisi bit yang sedang diproses) sampai bit ke n-K+1 (bit sebelumnya yang telah diproses). Maka, secara umum kita dapat menuliskan fungsi *parity* untuk setiap bit yang ditransmisikan sebagai berikut.

$$p_i[n] = (\sum_{j=0}^{k-1} g_i[j]x[n-j]) \bmod 2.$$

Pada fungsi yang diambil dari http://web.mit.edu/6.02/www/f2010/handouts/lectures/L8.pdf tersebut, p merupakan *parity bit* ke i, g merupakan sebuah bit konvolusi yang menyatakan pengaruh bit ke n-j terhadap *parity bit,* 0 menyatakan bahwa bit tersebut tidak berpengaruh, 1 menyatakan sebaliknya. g dikalikan dengan nilai bit ke n-j, kemudian dijumlahkan dari j = 0 hingga j = k-1. Hasil dari penjumlahan tersebut di modulo 2 sehingga *parity bit* yang mungkin hanya 0 atau 1. Cara lain dalam menghitung *parity bit* adalah dengan melakukan operasi *xor* untuk setiap bit x[n - j] yang nilai g-nya 1. Penentuan fungsi *parity* mempengaruhi struktur rangkaian yang *encoder* sinyal tersebut.

Cara lain memandang cara kerja kode konvolusi adalah dengan memandang proses pengkodean sebagai sebuah model *state machine*. Maka, untuk kode konvolusi dengan *constraint length* sepanjang K, terdapat 2^{k-1} kemungkinan *state* yang ditandai dengan K-1 bit dari bit ke n-1 hingga bit ke n-K+1 yang diproses sebelum bit ke n.

D. Jarak Hamming

Misal *a* dan *b* adalah dua *bit strings* dengan panjang sama, yaitu jumlah bit di *a* sama dengan jumlah bit di *b*. Misalkan juga a_i dan b_i merupakan elemen *string a* dan *b* pada posisi *i*. Maka, jarak Hamming antara dua data biner ini didefinisikan sebagai jumlah bit yang berbeda antara *a* dan *b* untuk setiap elemen a_i dan b_i yang berkorespondensi.

Sebagai contoh, misal sebuah deret bit *a* = 10 dan sebuah deret bit *b* = 01. Untuk setiap posisi *a* dan *b* yang berkorespondensi, terdapat sebanyak dua bit elemen yang berbeda antara *a* dan *b*, sehingga jarak Hamming antara *a* dan *b* adalah 2. Cara lain dalam menentukan jarak Hamming antara dua *string* biner adalah dengan melakukan operasi *xor* antaranya keduanya sehingga menghasilkan *string* baru, kemudian menghitung jumlah angka satu yang muncul pada *string* yang baru.

E. Algoritma Viterbi

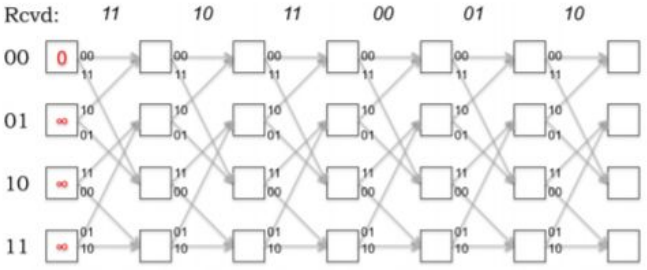
Gambar 5. Andrew Viterbi
sumber : wikipedia.org

Algoritma Viterbi merupakan algoritma yang ditemukan oleh Andrew Viterbi pada tahun 1967. Algoritma ini merupakan salah satu jenis algoritma yang bisa digunakan untuk men-*decode* kode konvolusi. Idenya adalah untuk mencari kombinasi bit paling memungkinkan ketika diberikan sekumpulan *parity bit* dari hasil kode konvolusi, atau dengan kata lain, mencari kemungkinan terbaik *state-state* yang pernah dilalui pada proses *encoding* dengan kode konvolusi.

III. Pembahasan

A. Overview

Algoritma viterbi bekerja untuk mencari kemungkinan terbaik dari data yang dikirim dengan cara membandingkan setiap r bit yang sampai pada penerima dengan r bit yang mungkin dikirim dari sejumlah *state* yang mungkin ada untuk suatu kode konvolusi. Dalam menggunakan algoritma Viterbi untuk men-*decode* kode konvolusi, kita perlu membuat graf dengan jumlah simpul sebanyak 2^{k-1} x ((Panjang data/r) + 1). 2^{k-1} merupakan jumlah *state* yang mungkin untuk sebuah kode konvolusi, ((Panjang data/r) + 1) merupakan banyaknya pergantian *state*, karena untuk setiap *parity bit* yang dikirim, maka kode konvolusi akan berganti *state*. Graf dapat digambarkan berbentuk persegi panjang dengan lebar sebanyak 2^{k-1} simpul dan panjang sebanyak ((Panjang data/r) + 1) simpul, kemudian untuk setiap baris kita berikan nama *state* berupa bit ke n-1 hingga n-K+1 yang sedang diproses pada kode konvolusi. Kemudian, untuk tiap kolom (terdapat 2^{k-1} *state*) kita tarik garis dari setiap *state* dalam satu kolom ke *state* pada kolom selanjutnya yang mungkin ketika mendapat *input* 1 atau 0. Agar mempermudah juga, kita dapat menuliskan tiap r *parity bit* yang diterima di antara 2 kolom yang berurutan. Sebagai contoh, graf yang dibuat untuk men-*decode* sebuah kode konvolusi dengan r = 2, K = 3, dan panjang data yang diterima 12 adalah sebagai berikut.

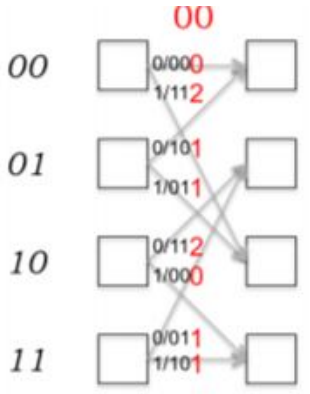


Gambar 6. Graf Algoritma Viterbi
sumber :
http://web.mit.edu/6.02/www/f2010/handouts/lectures/L9.pdf

Pada graf tersebut, 4 baris di kiri menandakan 4 *state* yang mungkin dari kode konvolusinya, karena $2^{3-1} = 2^2 = 4$. Kemudian, jumlah kolom dari gambar graf tersebut berjumlah ((12/2) + 1) = 7, sehingga jumlah total simpul pada graf tersebut adalah 28. Kemudian, setiap sisi yang menghubungkan dua buah simpul merepresentasikan *parity bit* yang mungkin dikeluarkan oleh kode konvolusi jika pada suatu waktu kode tersebut sedang berada pada suatu *state* S dan mendapat bit masukan 1 atau 0. Hal ini yang menyebabkan setiap *state* pada graf memiliki $d_{out} = 2$, yaitu karena terdapat dua kemungkinan masukan (0 atau 1), kecuali untuk *state* akhir dari kode konvolusi. Kemudian, setiap r bit (dalam hal ini r=2) yang berada di antara dua kolom yang berurutan merupakan *parity bit* yang diterima oleh penerima sinyal, yaitu sebanyak r setiap waktu.

B. Cara Kerja Algoritma Viterbi

Dalam menerapkan algoritma Viterbi, hal pertama yang perlu dilakukan adalah menentukan nilai atau bobot untuk setiap sisi yang ada pada graf dengan cara mencari jarak Hamming antara *parity bit* yang diterima (lebih tepatnya yang berada di atas kolom) dengan *parity bit* yang mungkin dikeluarkan di setiap sisi. Misalnya sebuah *state* 00 ketika menerima masukan 1 akan mengeluarkan *parity bit* 11 dan masuk ke *state* 10. Misalnya untuk saat itu *parity bit* sebenarnya yang diterima adalah 11. Maka, nilai yang dituliskan pada sisi tersebut adalah jarak Hamming antara *parity bit* sebenarnya (11) dengan *output* untuk *state* 00 ke *state* 10 dengan masukan 1 (11), yaitu 0. jarak Hamming yang dituliskan pada tiap sisi merepresentasikan banyaknya kerusakan bit pada sisi tersebut. Langkah ini dilakukan untuk setiap sisi yang ada pada graf.



Gambar 7. Penentuan Jarak Hamming
sumber :
http://web.mit.edu/6.02/www/f2010/handouts/lectures/L9.pdf

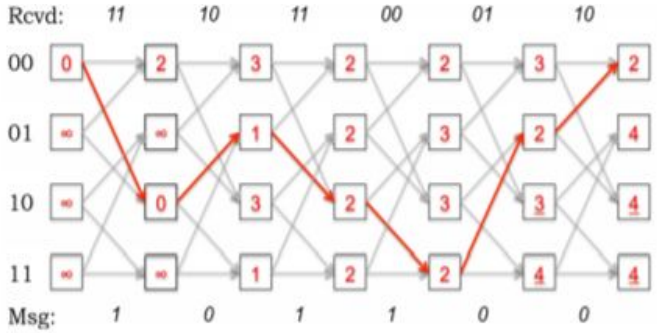
Selanjutnya, hal yang harus dilakukan adalah memberi nilai pada tiap simpul. tiap simpul bukan pada kolom pertama dapat dicapai dari dua simpul lain sebelumnya. misal *a* adalah jarak Hamming yang ada pada sisi graf ditambah dengan nilai pada simpul di pangkal sisi tersebut. Maka untuk tiap simpul bukan pada kolom pertama terdapat dua nilai *a* karena simpul tersebut dapat dicapai melalui dua sisi. Maka untuk menentukan nilai dari simpul tersebut, kita ambil nilai minimum antara dua nilai *a* yang ada pada simpul tersebut. Hal ini dilakukan karena *a* merupakan nilai akumulasi kerusakan pada bit yang diterima, sehingga semakin kecil nilainya, semakin kecil kerusakan yang dialami *parity bit* tersebut. Khusus untuk simpul-simpul atau *state-state* pada kolom pertama, karena kode konvolusi bernilai 0 (sebanyak K) sebelum ada masukan apapun, maka kode tersebut selalu berada pada *state* 0 (sebanyak K-1) pada kondisi awal, sehingga nilai simpul 0 (sebanyak K-1) adalah 0 dan nilai simpul lainnya pada kolom pertama adalah ∞.

Jika kita telah memberi nilai pada seluruh simpul dalam graf, maka kita pilih simpul pada kolom terakhir dengan nilai simpul yang paling minimum. Jika terdapat lebih dari satu nilai yang sama, kita dapat memilih yang manapun. Setelah memilih sebuah nilai, maka kita akan mengikuti lintasan pada graf dari simpul akhir tersebut hingga ke simpul awal, yaitu simpul 0 (sebanyak K-1). Lintasan yang dimaksud adalah sisi-sisi yang sebelumnya dipilih karena memiliki nilai *a* yang lebih minimum dari pada sisi lainnya. Dengan memilih lintasan ini, kita memilih bit *input* dengan probabilitas kerusakan paling kecil.

Secara singkat, langkah kerja algoritma Viterbi dapat disimpulkan sebagai berikut.

1. Menentukan nilai sisi untuk setiap sisi pada graf menggunakan jarak Hamming *parity bit* sesungguhnya dengan *parity bit* pada sisi.
2. Menentukan nilai tiap simpul dengan mencari nilai minimum dari jarak Hamming ditambah nilai simpul sebelumnya dari dua simpul yang mungkin, kecuali untuk simpul-simpul pada kolom pertama.
3. Memilih sebuah simpul dengan nilai minimum pada kolom terakhir dan membuat lintasan dari simpul tersebut ke simpul awal dengan memilih sisi yang memiliki jumlah antara jarak Hamming dengan nilai simpul sebelumnya yang paling minimum.
4. Setiap bit masukan yang direpresentasikan oleh tiap sisi yang dipilih sebagai lintasan merupakan data yang paling mendekati dengan data yang dikirim oleh kode konvolusi.

Contoh hasil akhir dari proses *decode* menggunakan algoritma Viterbi adalah sebagai berikut.



Gambar 8. Hasil Akhir Algoritma Viterbi
sumber :
http://web.mit.edu/6.02/www/f2010/handouts/lectures/L9.pdf

C. Hasil dari Algoritma Viterbi

Hasil yang diterima dari algoritma Viterbi memang belum tentu benar sepenuhnya, karena ide dari penerapan algoritma tersebut adalah mencari kombinasi *input* yang memiliki kemungkinan paling baik dalam merepresentasikan data awal. Kita dapat meningkatkan maupun menurunkan probabilitas tersebut salah satunya dengan cara mengubah besaran nilai K serta banyaknya *parity bit* yang dikirim dalam satu waktu. Semakin banyak *parity bit*, semakin tidak rentan terhadap kerusakan, namun semakin sulit kode tersebut dalam proses *decode*-nya. Cara lain untuk meminimalisasi kerusakan bit pada algoritma Viterbi adalah dengan menerapkan *soft decision coding*. Berbeda dengan *hard decision coding*, yang memproses bit pada bagian *decoder*, *soft decision coding* memproses sinyal kontinu pada bagian *decoder* sehingga *range* nilai dari kerusakan bit akan lebih variatif. Hal ini akan lebih mempermudah penerima dalam mendeteksi banyaknya kerusakan pada bit yang dikirim.

IV. KESIMPULAN

Graf memiliki banyak aplikasi pada cabang ilmu lain, salah satunya telekomunikasi. Pada bidang telekomunikasi, graf salah satunya digunakan dalam proses *decode* sinyal yang di *encode* menggunakan kode konvolusi menggunakan algoritma Viterbi dalam upaya mengurangi kerusakan bit pada data yang dikirim pada sebuah komunikasi digital.

Algoritma Viterbi sendiri merupakan salah satu algoritma yang cukup berhasil dan banyak digunakan pada berbagai aspek dalam telekomunikasi, meskipun masih banyak lagi

algoritma-algoritma lain yang digunakan dalam men-*decode* data pada sistem komunikasi digital.

V. Ucapan Terima Kasih

Puji syukur kehadirat Allah SWT, atas berkah dan rahmat-Nya, sehingga penulis dapat menyelesaikan makalah ini. Terima kasih penulis ucapkan kepada seluruh dosen pengampu mata kuliah matematika diskrit karena telah membimbing penulis selama proses perkuliahan hingga pengerjaan makalah ini. Terima kasih juga penulis ucapkan kepada keluarga penulis yang selalu memberikan dukungan yang besar kepada penulis.

Referensi

[1] http://web.mit.edu/6.02/www/f2010/handouts/lectures/L8.pdf, diakses pada 4 Desember 2019

[2] http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf, diakses pada 4 Desember 2019

[3] http://www.mit.edu/~6.450/handouts/6.450book.pdf, diakses pada 5 Desember 2019

[4] https://repository.unikom.ac.id/43067/1/Jarkom%20-%20Bab%2010.pdf, diakses pada 4 Desember 2019

[5] http://web.mit.edu/6.02/www/f2010/handouts/lectures/L9.pdf, diakses pada 5 Desember 2019

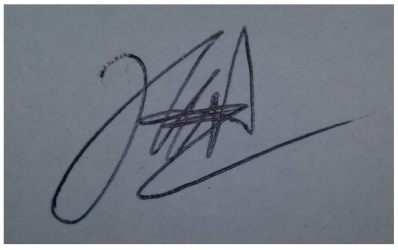
[6] https://www.design-reuse.com/articles/21107/viterbi-algorithm.html, diakses pada 6 Desember 2019

[7] https://www.geeksforgeeks.org/error-detection-in-computer-networks/, diakses pada 5 Desember 2019

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 6 Desember 2019



Jonet Wira Murti 13518083