Perbaikan Internal Blocking Jaringan Interkoneksi Banyak Tingkat Topologi Omega 8x8 dengan Algoritma Look-Ahead

M. Zulfin[1)], Maksum Pinem[2)], Raja Harahap[3)], Fahmi Rinaldi[4)], Muhammad Razali[5)]

[1,2,3)]*Electrical Engineering Department* Universitas Sumatera Utara
[4)]*Nautical Department* Politeknik Adiguna Maritim Indonesia Medan
[5)]Universitas Pembangunan Masyarakat IndonesiaMedan
m.zulfin@usu.ac.id; maksum@usu.ac.id; raja@usu.ac.id;
fahmi.rinaldi@gmail.com; razalialy@gmail.com

Abstrak

Jaringan interkoneksi banyak tingkat (Multistage Interconnection Network/MIN) hingga saat ini digunakan sebagai switching pada sentral-sentral dalam sistem-sistem Telekomunikasi. Disamping itu MIN juga digunakan sebagai switch penghubung antara prosesor dan modul memori pada sistem-sistem Komputer. Penggunaan MIN semakin diminati karena penghematan jumlah crosspoint yang dimilikinya dibandingkan dengan switch matriks konvensional yang jumlah crosspoint-nya lebih banyak. Selain itu MIN mudah dikontrol dan mampu mendukung koneksi input-output dalam skala besar. Namun MIN bersifat internal blocking, sehingga dibutuhkan suatu cara untuk menjadikannya non-blocking atau mengurangi persentase internal blocking yang dimilikinya. Salah satu cara yang dilakukan untuk maksud tersebut adalah menggunakan algoritma. Pada tulisan ini dibahas algoritma Look-Ahead untuk mengurangi internal blocking MIN topologi Omega 8x8. Dari 2 buah contoh yang dianalisis, untuk permutasi dengan koneksi input-output yang uniform tanpa algoritma Look-Ahead diperoleh hasil bahwa internal blocking sebesar 62,5% dan jika menggunakan algoritma Look-Ahead internal blocking turun menjadi menjadi 25%. Sedangkan dengan permutasi yang memiliki koneksi yang non uniform persentase internal blocking-nya menjadi lebih tinggi yaitu sebesar 87,5% jika tanpa algoritma Look-Ahead, sedangkan dengan algoritma Look-Ahead turun menjadi 50%. Jadi algoritma Look-Ahead sangat mengurangi kegagalan koneksi input-output sebuah permutasi atau dengan kata lain mengurangi internal blocking sebuah jaringan MIN secara signifikan.

Kata Kunci: *MIN, Omega, Algoritma,Look-Ahead, Permutasi*

I. PENDAHULUAN

Pada awal perkembangan sistem-sistem Telekomunikasi dan Komputer, *switching* yang digunakan pada sentral-sentral Telepon maupun untuk koneksi paralel prosesor dan memori pada sistem Komputer adalah *switching* topologi matriks. Pada *switching* topologi matriks ini, jika jumlah input dan output sistem *switching* adalah N maka terdapat N^2 *crosspoint* (titik sambung) yang dibutuhkan untuk mewujudkan koneksi input-output yang *non-blocking*. Misalkan jika jumlah pelanggan telepon sebanyak 5000 pelanggan, maka dibutuhkan 25.000.000 *crosspoint* pada sentral *telepon* tersebut. Ini sangat mahal [1].

Dengan bertambahnya pengguna sistem-sistem Telekomunikasi dan jumlah pemrosesan paralel pada sistem-sistem Komputer, topologi matriks sudah tidak lagi menjadi pilihan, meskipun telah difabrikasi sejak lama. Biaya pengadaannya yang sangat mahal telah menjadi penyebab peralihan itu. Sebagai penggantinya para ahli mengusulkan MIN. Sayangnya MIN bersifat *blocking*, sehingga dibutuhkan suatu cara untuk menjadikannya *non-blocking*. Sejumlah cara agar MIN *non-blocking*telah diupayakan para ahli, antara lain: mengatur jumlah elemen switching dan pola interkoneksinya, menambah jumlah tingkat MIN dan mengimplementasikan algoritma tertentu kepada MIN. Charles Clos pada tahun 1953 pertama kali mengusulkan MIN yang *strictly non-bloking* yang artinya jaringan ini benar-benar *non-blocking*, bagaimanapun kondisi jaringan. Tidak pernah ada koneksi yang diblokir meskipun tanpa mengatur ulang jalur koneksi input-outputnya[2]. Penemuan yang luar biasa ini menjadi cikal bakal perkembangan MIN menggantikan topologi matriks.

Pada tahun 1962, Benes melakukan modifikasi terhadap jaringan Clos 3 tingkat . Struktur baru Benes yang unik menghasilkan struktur *switching* yang memiliki jumlah elemen *switching* yang lebih efisien sehingga jumlah *crosspoint* yang dibutuhkan menjadi lebih sedikit dibandingkan dengan *switching* Clos 3 tingkat [3].

Setelah Benes, sejumlah ahli telah mengusulkan algoritma *routing* untuk mengatasi *blocking* pada MIN, di antara yang terkenal adalah: algoritma *looping* [4], algoritma matriks [5], algoritma paralel [6], algoritma *Complete Residue Partition Tree* (CRPT) [7], algoritma *division* [8] dan yang terbaru adalah algoritma dengan matriks*array* [9]. Hampir semua algoritma yang ditawarkan tersebut secara khusus diimplementasikan pada topologi Benes. Untuk mengetahui lebih dalam prinsip-prinsip algoritma

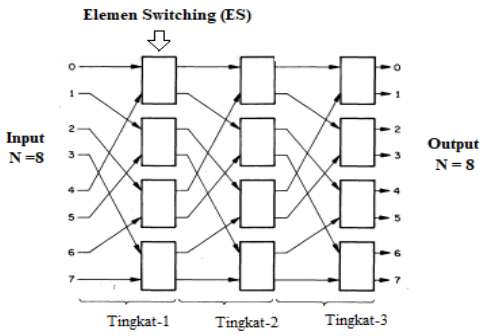
di atas, kami telah melakukan penelitian khusus yang hasilnya dituangkan pada referensi [10][11].

Pada tulisan ini dibahas tentang sebuah algoritma yang dapat mengurangi bahkan menjadikan sebuah *switching* MIN bersifat *non-blocking*. Algoritma tersebut diberi nama Look-Ahead [12]. Pada tulisan ini, algoritma Look-Ahead diimplementasikan kepada sebuah MIN topologi Omega berukuran 8x8. Untuk mengetahui seberapa besar penurunan persentase *internal blocking* koneksi input-output topologi tersebut, penulis menetapkan sejumlah permutasi secara acak yang melibatkan koneksi input-output yang uniform dan non uniform, sehingga pengevaluasian yang dilakukan menjadi realistik.

II. TINJAUAN PUSTAKA

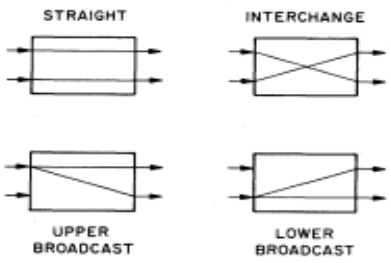
2.1 Jaringan MIN tolopogi Omega

Sebuah jaringan Omega N x N adalah jaringan MIN yang terdiri dari $\log_2 N$ tingkat yang identik dimana N adalah jumlah input dan output jaringan. Pola link antar tingkat yang dimilikinya adalah jaringan interkoneksi *perfect shuffle* dengan jumlah elemen switching setiap tingkat sebanyak N/2 seperti yang ditunjukkan pada Gambar 1 [13].



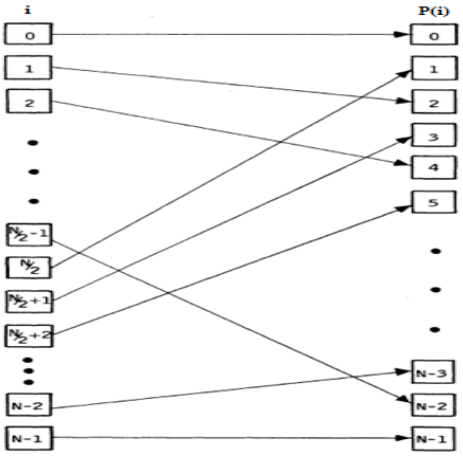
Gambar 1. Jaringan Switching Omega 8xs8

Setiap elemen switching dapat memiliki salah satu dari empat status seperti yang ditunjukkan pada Gambar 2. Artinya, ia dapat mengirim inputnya langsung (*straight through*), menukar input (*interchange*), atau menyiarkannya (*broadcast*) dari input ke kedua output. Dalam hal ini tidak diizinkan kedua input untuk di-*switch* ke *output* yang sama.



Gambar 2. Empat kondisi elemen switching yang diizinkan

Struktur jaringan Omega melibatkan pengocokan yang sempurna yang mengacu kepada pola interkoneksi yang ditunjukkan pada Gambar 3. Di bagian sebelah kiri gambar adalah vektor operan dengan indeks mulai dari 0 hingga N-1, di mana $N= 2^m$ untuk beberapa bilangan bulat m. Operand terhubung ke vektor di sebelah kanan gambar melalui pola interkoneksi tertentu yang dinamakan *perfect shuffle* [14].



Gambar 3. Pola Interkoneksi *Perfect Shuffle*

Dengan mengamati dengan cermat Gambar 3 tampak bahwa indeks di sebelah kiri dipetakan ke indeks di sebelah kanan sesuai dengan permutasi P berikut ini.

$P(i) = 2i$, dimana $0 \leq i \leq N/2 - 1$
$= 2i + 1 - N$, dimana $N/2 \leq i \leq N - 1$ ……(1)

2.2 Algoritma Look-Ahead

Algoritma look ahead yang akan dipaparkan pada tulisan ini adalah algoritma yang mampu untuk mengatasi jumlah *internal blocking* dari tingkat 1 sampai dengan tingkat terakhir MIN dengan cara memberi bobot kepada setiap koneksi yang bertikai [12].

Langkah-langkah algoritma Look-Ahead adalah sebagai berikut:

Langkah ke 1: Menetapkan secara acak permutasi (koneksi input-output) jaringan MIN
Langkah ke 2: Membangun MIN dengan N input/output
Langkah ke 3: Merutekan paket dengan algoritma *self-routing* sesuai dengan permutasi yang ditetapkan
Langkah ke 4 : Membuat tabel nilai pembobotan sesuai rute paket pada langkah ke 2
Langkah ke 5 : Meng-update tabel nilai bobot setiap tingkat mulai dari tingkat 1 dan seterusnya.
Langkah ke 6 : Melakukan tabulasi ulang untuk mengetahui persentase koneksi yang gagal dan sukses

III. ANALISIS

Implementasi Algoritma Look Ahead pada Jaringan MIN Topologi Omega 8x8

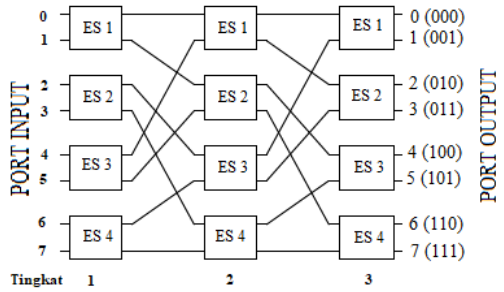
Berikut ini akan dianalisis 2 buah contoh pengimplementasian algoritma Look-Ahead kepada MIN topologi Omega 8x8.

Contoh 1:

Langkah ke 1: Menetapkan permutasi jaringan (P). Permutasi ditetapkan secara acak dengan menganggap bahwa paket-paket yang berasal dari port input terdistribusi secara uniform ke port output.

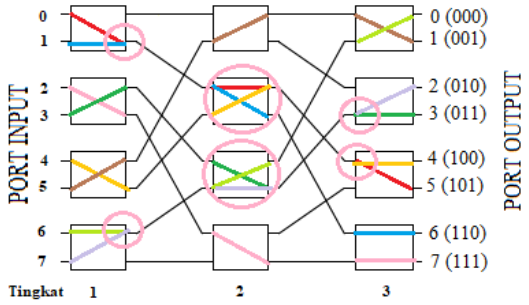
$$P = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & 6 & 7 & 3 & 4 & 1 & 0 & 2 \end{pmatrix} \begin{matrix} \Rightarrow \text{port input} \\ \Rightarrow \text{port output} \end{matrix}$$

Langkah ke 2: Membangun jaringan MIN topologi Omega berukuran 8x8 seperti tampak pada Gambar 4. Jumlah tingkat MIN topologi Omega adalah n = $\log_2$ N dimana N adalah jumlah port input/output. Oleh karena N = 8 maka jumlah tingkat sebanyak 3 dari kiri ke kanan: tingkat 1, tingkat 2 dan tingkat 3. Jumlah elemen switching (ES) jaringan Omega adalah ES = N/2 $\log_2$N. Karena N = 8 maka jumlah ES menjadi 12. Elemen switching diberi nomor urut dari atas ke bawah (ES1, ES2, ES3 dan ES4) pada setiap tingkat.



Gambar 4. Struktur MIN topologi Omega Dasar 8x8

Langkah ke3: Merutekan paket melalui MIN topologi Omega sesuai dengan permutasi P. Perutean dilakukan menggunakan algoritma *self-routing* dengan menyisipkan alamat port output ke *header* paket. Gambar 5 memperlihatkan perutean paket dari port input menuju port output.



Gambar 5. Perutean paket sesuai dengan permutasi P

Langkah ke 4: Membuat tabel nilai pembobotan berdasarkan perutean pada Gambar 5.

Tabel 1. Nilai Bobot Sesuai Perutean pada Gambar 5

Input	Bobot	Kumpulan Input yang bertikai
0	2	{1,4}
1	1	{0}
2	-	{-}
3	1	{7}
4	1	{0}
5	-	{-}
6	1	{7}
7	1	{3, 6}

Langkah ke 5: Memperbaharui nilai bobot dengan mengamati setiap pemblokiran pada setiap tingkat dengan ketentuan sebagai berikut:

a) Input yang lebih tinggi atau lebih rendah diblokir (*fixed blocking*). Pada tulisan ini dipilih input yang lebih rendah diblokir.
b) Input dengan bobot yang lebih besar diblokir (*random blocking*).

Nilai bobot yang telah diperbaharui setelah pemblokiran pada tingkat 1 dapat dilihat pada Tabel 2.

Tabel 2. Nilai bobot yang diperbaharui pada tingkat 1

Input	Bobot	Kumpulan Input yang bertikai
0	x	{x}
1	-	{-}
2	-	{-}
3	1	{7}
4	-	{-}
5	-	{-}
6	-	{-}
7	x	{x}

Keterangan: x = diblokir

Nilai bobot yang telah diperbaharui setelah pemblokiran pada tingkat 2 dapat dilihat pada Tabel 3.

Tabel 3. Nilai bobot yang diperbaharui pada tingkat 2

Input	Bobot	Kumpulan Input yang bertikai
0	x	{x}
1	-	{-}
2	-	{-}
3	-	{-}
4	-	{-}
5	-	{-}
6	-	{-}
7	x	{x}

Oleh karena input 0 dan input 7 sudah terblokir maka pembaharuan nilai bobot pada pertikaian di tingkat 3 sudah tidak diperlukan lagi. Dari Tabel 3 tampak bahwa dengan algoritma Look-Ahead terdapat 2 koneksi input-output yang

diblokir yaitu koneksi 0-5 dan 7-2, sedangkan 6 koneksi lain sukses.

Langkah ke 6: Tabulasikan hasil pembaharuan nilai bobot yang memperlihatkan koneksi yang gagal dan sukses dari permutasi yang telah ditetapkan seperti yang diperlihatkan pada Tabel 4.

Tabel 4. Koneksi Gagal dan Sukses Tanpa dan dengan Algoritma Look-Ahead

Koneksi	Tanpa Algoritma Look-Ahead	Dengan Algoritma Look-Ahead
0-5	Gagal	Gagal
1-6	Gagal	Sukses
2-7	Sukses	Sukses
3-3	Gagal	Sukses
4-4	Gagal	Sukses
5-1	Sukses	Sukses
6-0	Sukses	Sukses
7-2	Gagal	Gagal

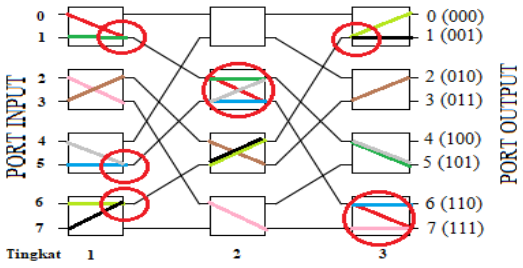
Contoh 2:

Langkah ke 1: Ditetapkan permutasi P yang memiliki koneksi input-output yang bersifat non uniform, sebagai berikut:

$$P = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 7 & 5 & 7 & 2 & 5 & 6 & 0 & 1 \end{pmatrix} \begin{matrix} \Rightarrow \text{port input} \\ \Rightarrow \text{port output} \end{matrix}$$

Langkah ke 2: Membangun MIN topologi Omega berukuran 8x8 yang sama seperti Gambar 4.

Langkah ke 3: Merutekan paket melalui MIN topologi Omega 8x8 sesuai dengan permutasi P seperti tampak pada Gambar 6.



Gambar 6. Perutean paket sesuai dengan permutasi P

Langkah ke 4: Membuat tabel nilai pembobotan berdasarkan perutean pada Gambar 6 seperti yang diperlihatkan pada Tabel 5.

Tabel 5. Nilai Bobot Sesuai Perutean pada Gambar 6

Input	Bobot	Kumpulan Input yang bertikai
0	3	{1,2,5}
1	2	{0,4}
2	1	{0}
3	-	{-}
4	2	{1,5}
5	2	{0,4}
6	1	{7}
7	1	{6}

Langkah ke 5: Memperbaharui nilai bobot dengan mengamati setiap pemblokiran pada setiap tingkat.

Nilai bobot yang telah diperbaharui setelah pemblokiran pada tingkat 1 dapat dilihat pada Tabel 6.

Tabel 6. Nilai bobot yang diperbaharui pada tingkat 1

Input	Bobot	KumpulanInput yang bertikai
0	x	{x}
1	1	{4}
2	1	{0}
3	-	{-}
4	1	{1}
5	x	{x}
6	-	{-}
7	x	{x}

Keterangan: x = diblok

Nilai bobot yang telah diperbaharui setelah pemblokiran pada tingkat 2 dapat dilihat pada Tabel 7.

Tabel 7. Nilai bobot yang diperbaharui pada tingkat 2

Input	Bobot	Kumpulan Input yang bertikai
0	x	{x}
1	-	{-}
2	1	{0}
3	-	{-}
4	x	{x}
5	x	{x}
6	-	{-}
7	x	{x}

Nilai bobot yang telah diperbaharui setelah pemblokiran pada tingkat 3 dapat dilihat pada Tabel 8.

Tabel 8. Nilai bobot yang diperbaharui pada tingkat 3

Input	Bobot	Kumpulan Input yang bertikai
0	x	{x}
1	-	{-}
2	-	{-}
3	-	{-}
4	x	{x}
5	x	{x}
6	-	{-}
7	x	{x}

Langkah ke 6: Hasil pembaharuan nilai bobot yang memperlihatkan koneksi yang gagal dan sukses dari permutasi yang telah ditetapkan seperti yang diperlihatkan pada Tabel 9.

Tabel 9. Koneksi Gagal dan Sukses Tanpa dan dengan Algoritma Look-Ahead

Koneksi	Tanpa Algoritma Look-Ahead	Dengan Algoritma Look-Ahead
0-7	Gagal	Gagal
1-5	Gagal	Sukses
2-7	Gagal	Sukses
3-2	Sukses	Sukses
4-5	Gagal	Gagal
5-6	Gagal	Gagal
6-0	Gagal	Sukses
7-1	Gagal	Gagal

IV. KESIMPULAN

Dari pembahasan yang telah dilakukan terhadap MIN Omega 8x8 diperoleh hasil bahwa jika permutasi hanya mengandung koneksi yang uniform (contoh 1), tanpa mengunakan algoritma Look-Ahead,sebesar 62,5 % koneksi gagal dan 38,5% sukses. Sedangkan jika menggunakan algoritma Look-Ahead, maka25% gagal dan 75% sukses. Sedangkan jika permutasi mengandung koneksi yang nonuniform (contoh 2), diperoleh hasil bahwa tanpa mengunakan algoritma Look-Ahead, 87,5 % koneksi gagal dan 12,5% sukses. Sedangkan jika menggunakan algoritma Look-Ahead, 50% gagal dan 50% sukses. Sehingga dapat disimpulkan bahwa algoritma Look-Ahead dapat menurunkan persentase *internal blocking* MIN. Selain itu diperoleh hasil bahwa persentase perbaikan *internal blocking* dengan algoritma Look-Ahead bergantung kepada permutasi yang diimplementasikan kepada MIN. Dengan kata lain, permutasi yang berbeda akan menghasilkan persentase *internal blocking* yang berbeda.

DAFTAR PUSTAKA

[1] Muhammad Zulfin., et al., 2018, *Cross-Point Comparison of Multistage Non-Blocking Technologies*, International Journal of Engineering Technology, Vol. 7, No. 3.2, pages. 703-708, 2018.

[2] Clos, C., 1953, *A Study of Non-Blocking Switching Networks*, TheBell System Technical Journal.

[3] Benes,V.E., 1962, *On Rearrangeable Three-Stage Connecting Networks*, The Bell System Technical Journal.

[4] D. C. Opfermanand N.T. Tsao-Wu, 1971, *Ona Class Rearrangeable Switching Networks*, Part I: Control Algorithm. Bel System Technical Journal,Vol.50,pp.1,579-1,600.

[5] Amitabha Chakrabarty, Martin Collier and Sourav Mukhopadhyay, 2009, *M Matrix-Based Nonblocking Routing Algorithm for Benes Networks*, Computation World : Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns.

[6] David Nassimi and Sartaj Sahni, 1982, *Parallel Algorithms to Set Up the Benes Permutation Network*, IEEE Trans. Comput., vol.C-31, No.2, pp.148-153,1982.

[7] Lee, Kyungsook Yoon, 1987, *A New Benes Control Algorithm*, IEEE Transactionson Computers, Vol. C-36, No.6.

[8] Abbas Karimi, Kiarash Aghakhani, Seyed Ehsan Manavi, Faraneh Zarafshan and S.A.R Al-Haddad, 2014, *Introduction And Analysis Of Optimal Routing Algorithm In Benes Networks*, International Conferenceon Robot PRIDE 2013-2014-Medical and Rehabilitation Robotics and Instrumentation.

[9] Kiarash Aghakhani and Abbas Karimi, 2016, *A Novel Routing Algorithmin Benes Networks*. International Journal of Educational Advancement, Vol.7 No.1, pp. 168-177.

[10] Muhammad Zulfin, Rahmad Fauzi, Yussa Ananda, Maksum Pinem, Muhammad Razali and S Suherman, 2019, *Performance Comparison of Looping and Input-Output Algorithms on the 8 x 8 Benes Switching Network*, The 3rd International Conferenceon Electrical, Telecommunication and Computer Engineering (ELTICOM).

[11] Muhammad Zulfin, Maksum Pinem, Arbi Divo, Fahmi Rinaldi and Muhammad Razali, 2021, *Performance Comparison of Matrix-Based and Division Algorithms Implemented on The 8x8 Benes Network*, The 5th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM).

[12] Abdulaziz S. Almazyad, 2010, *A New Look-Ahead Algorithm To Improve The Performance Of Omega Networks*, *Mathematical and Computational Applications*, Vol. 15, No. 2, pp. 156-165.

[13] Lawrie, Duncan H., 1975, *Access and Aligment of Data in an Array Processor*, IEEE Trans. Comput. vol c-24, hal 1145-1155.

[14] Stone, Harold S., 1972, *Parallel Processing with the Perfect Shuffle*, IEEE Trans. Comput. vol C - 20. hal 153-161, 1972.