



**ANALISIS KINERJA *TRAFFIC BEHAVIOUR* PADA
MANET DENGAN MENGGUNAKAN NS-2**

SKRIPSI

Oleh

M WAHYU ABDUL GHOFUR

NIM 111910201054

**JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2015**



**ANALISIS KINERJA *TRAFFIC BEHAVIOUR* PADA
MANET DENGAN MENGGUNAKAN NS-2**

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Strata 1 Teknik
dan mencapai gelar Sarjana Teknik

Oleh

**M WAHYU ABDUL GHOFUR
NIM 111910201054**

**JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2015**

PERSEMBAHAN

Puji dan syukur kehadirat Tuhan Yang Maha Esa atas kasih setia-Nya yang telah melimpahkan segala rahmat yang tak ternilai, sehingga bisa terlaksana penyelesaian penelitian yang saya lakukan ini.

Akhirnya, kupersembahkan tugas akhir ini untuk :

1. Kedua Orang tuaku Ibu Sri Suratini dan Bapak Subur Santoso yang telah membesarkan, mendidik, mendoakan tiada henti, memberi motivasi semangat, menitikkan air mata dan memberi kasih sayang yang tak pernah habis serta pengorbanannya selama ini,
2. Saudara Rofi Ana Hayati, Lia Anggraini, dan Siti Chusnul Chotimah yang senantiasa memberi semangat dan doa,
3. Terimakasih Ibu Ike Fibriani S.T.,M.T. dan Bapak Andrita Ceriana Eska S.T.,M.T. selaku dosen pembimbing yang telah membimbing saya menyelesaikan tugas akhir ini,
4. Terimakasih Bapak Dodi Setia Budi S.T.,M.T. dan Bapak Widya Cahyadi S.T.,M.T. selaku dosen penguji yang sudah memberikan saran untuk memperbaiki tugas akhir ini,
5. Terimakasih Bapak Samsul Bachri selaku dosen pembimbing akademik yang selalu membimbing dan memberi motivasi dari awal semester 1,
6. Terimakasih juga kepada Teman-teman Teknik Elektro Universitas Jember angkatan 2011 yang selalu mendampingi dan memberi semangat dimanapun saya berada,
7. Terimakasih juga kepada Keluarga PAMADIKSI Universitas Jember yang telah memberi banyak pengalaman,
8. Terimakasih BIDIKMISI yang telah memberi saya kesempatan untuk merasakan bangku kuliah.
9. Almamater Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

MOTTO

Sesungguhnya bersama kesukaran itu ada keringanan. Karena itu bila kau sudah selesai
(mengerjakan yang lain). Dan berharaplah kepada Tuhanmu.

(terjemahan Surat Al-Insyirahayat6-8)^{*)}

Sesungguhnya Allah tidak mengubah nasib suatu kaum kecuali kaum itu sendiri yang
mengubah apa-apa yang pada diri mereka.

(terjemahan QS. Ar Ra'du ayat 11)^{**)}

Usaha Keras Itu Tak Akan Menghianati.

(Lirik lagu Sonichi (Hari Pertama) JKT48)^{***)}

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : M. Wahyu Abdul Ghofur

NIM : 111910201054

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Analisis Kinerja *Traffic Behaviour* Pada MANET dengan Menggunakan NS-2" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab penuh atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, 28 Agustus 2015

Yang menyatakan

M. Wahyu Abdul Ghofur
NIM111910201054

SKRIPSI

**ANALISIS KINERJA *TRAFFIC BEHAVIOUR* PADA
MANET DENGAN MENGGUNAKAN NS-2**

Oleh

M. Wahyu Abdul Ghofur
NIM 111910201054

Pembimbing

Dosen Pembimbing Utama : Ike Fibriani, S.T.,M.T.
Dosen Pembimbing Anggota : Andrita Ceriana Eska, S.T.,M.T.

PENGESAHAN

Skripsi berjudul “Analisis Kinerja *Traffic Behaviour* Pada MANET dengan Menggunakan NS-2” : M. Wahyu Abdul Ghofur, 111910201054” telah diuji dan disahkan pada :

Hari : Jumat
Tanggal : 28 Agustus 2015
Tempat : Fakultas Teknik Universitas Jember

Tim Penguji

Pembimbing Utama,

Pembimbing Anggota,

Ike Fibriani, S.T.,M.T.
NIP 198002072015042001

Andrita Ceriana Eska, S.T.,M.T.
NRP 760014640

Penguji I

Penguji II,

Dodi Setiabudi, S.T.,M.T.
NIP 198405312008121004

Widya Cahyadi, S.T.,M.T.
NIP 198511102014041001

Mengesahkan
Dekan
Fakultas Teknik
Universitas Jember,

Ir. Widyono Hadi, M.T.
NIP 196104141989021001

Analisis Kinerja Traffic Behaviour Pada MANET dengan Menggunakan NS-2

M Wahyu Abdul Ghofur

Jurusan Teknik Elektro, Fakultas Teknik, Universitas

ABSTRAK

Teknologi MANET terbentuk dari sekumpulan *node* yang menggunakan antarmuka nirkabel untuk melakukan komunikasi antara satu *node* dengan *node* yang lainnya, setiap *node* pada teknologi MANET mempunyai fungsi *routing* untuk mengirimkan paket dari sumber ke tujuan melalui *node* lain. *Routing* diperlukan untuk penentuan *route* terbaik oleh *node* agar paket dari sumber sampai ke *node* tujuan, beberapa jenis *routing* yang digunakan yaitu AODV, DSR, dan DSDV. Selain *routing*, dalam pengiriman paket dari *node* sumber ke *node* tujuan diperlukan suatu sumber *traffic* atau *traffic behaviour*, dan beberapa jenis dari *traffic behaviour* yang digunakan yaitu FTP (*File Transfer Protocol*) dan CBR (*Constant Bit Rate*). Tujuan dari penelitian ini adalah mengetahui kinerja *traffic behaviour* dan juga mengetahui perbandingannya pada teknologi MANET dengan cara mensimulasikannya menggunakan *Network Simulator 2 (NS-2)*. Dalam pengujian penelitian terdapat dua skenario yaitu skenario pertama akan dilakukan perubahan jumlah *node*, yaitu 20, 30, dan 50 *node*, sedangkan skenario kedua akan dilakukan perubahan pada kecepatan *node*, yaitu 5, 10, 15, dan 20 m/s. Untuk penilaian kinerja dan perbandingan menggunakan parameter *Normalized Routing Load (NRL)* dan *Routing Overhead (RO)*. Dari hasil analisis yang telah dilakukan, hasil nilai simulasi yang sudah diperoleh dengan menyimpulkan bahwa *traffic behaviour* jenis FTP merupakan *traffic behaviour* yang memiliki kinerja yang lebih baik dalam pengujian ini, karena memiliki nilai rata-rata terkecil.

Kata kunci : MANET, NS-2, *Routing Protocol*, *Traffic Behaviour*.

Analysis Performance Traffic Behaviour of MANET Using NS-2

M Wahyu Abdul Ghofur

Department of Electrical Enggineering, Faculty of Engineering, University of Jember

ABSTRACT

MANET technology is formed from node which uses wireless interface as communication between node and the other nodes. The use of nodes on MANET technology is sending instruction from the source to the destination by another node. This term is known as routing. Routing is used for determine the best route by node. It has a purpose to ensure that the instruction arrives in the destination. There are 3 types of routing in this researcn namely AODV, DSR, and DSDV. MANET technology also uses traffic behaviour to send the instruction. There are 2 types of traffic behaviour, those are FTP (File Transfer Protocol) and CBR (Constant Bit Rate). The purpose of this research is knowing the performance of Traffic Behavior and figuring its comparison toward MANET technology by using Network Simulator 2 (NS-2). This research uses 2 scenarios for testing method. The first scenario is changing the node total into 20, 30, ad 50 nodes. The second scenario is changing the node speed becomes 5, 10, 15 and 20 m/s. Normalized Routing Load (NRL) dan Routing Overhead (RO) parameter used to evaluate the performace and the ratio. The result of the simulation research shows that FTP traffic behavior has the best performace because it has the smallest average value than the others.

Key word : MANET, Traffic Behaviour, Routing Protocol, NS-2

RINGKASAN

Analisa Kinerja *Traffic Behaviour* Pada MANET dengan Menggunakan NS-2; M. Wahyu Abdul Ghofur, 111910201054; 2015; 95 halaman; Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

Mobile Ad Hoc Network (MANET) terbentuk dari sekumpulan *node* yang menggunakan antarmuka nirkabel untuk melakukan komunikasi antara satu *node* dengan *node* yang lainnya. *Node* pada jaringan MANET tidak hanya berperan sebagai pengirim dan penerima data, namun dapat berperan sebagai penunjang *node* yang lainnya, dan disetiap *node* memiliki kemampuan untuk proses *routing*. *Routing* merupakan penentuan *route* terbaik oleh *node/router* dengan algoritma tertentu agar paket dari sumber sampai ditujuan dengan kecepatan yang optimal. *Routing protocol* yang digunakan adalah AODV, DSR, dan DSDV.

Sumber trafik atau *traffic behaviour* pada MANET dibutuhkan untuk proses pengiriman data antara *node* pengirim dan *node* penerima, dalam hal ini penentuan sumber trafik sangat dibutuhkan dan di dalam penentuan sumber trafik tersebut belum terdapat standar untuk pemilihan tipe aplikasi apa yang harus digunakan dan di implementasikan pada MANET.

Tujuan penelitian ini adalah mengetahui kinerja *traffic behaviour* pada MANET dengan menggunakan NS-2 dan menganalisis hasil perbandingan dari jumlah dan kecepatan *node* terhadap kinerja *traffic behaviour* pada *routing protocol*. *Traffic behaviour* yang digunakan adalah *File Transfer Protocol* (FTP) dan *Constant Bit Rate* (CBR).

Parameter pengujian yang digunakan untuk mengetahui kinerja dan perbandingan tersebut adalah *Routing Overhead* (RO) dan *Normalized Routing Load* (NRL). Analisis yang akan dilakukan adalah dengan melakukan dua skenario, untuk skenario pertama dilakukan perubahan jumlah *node* dari 20 *node* yang mewakili sebagai jumlah *node*

sedikit, 30 *node* yang mewakili sebagai jumlah *node* sedang, dan 50 yang mewakili jumlah *node* banyak dan dengan kecepatan perpindahan *node* sebesar 10 m/s. Untuk skenario kedua dilakukan perubahan kecepatan pada *node* yang terdiri dari 5 m/s, 10 m/s, 15 m/s dan 20 m/s, dengan jumlah banyaknya *node* adalah 25 *node* pada setiap perubahan kecepatan perpindahan *node*. Dan untuk topologi yang digunakan adalah topologi STAR dalam desain jaringan pada MANET di NS2.

Hasil dari kedua skenario pertama dan kedua dengan *traffic behaviour* FTP dan CBR dengan *routing protocol* yang digunakan adalah AODV, DSR, dan DSDV. Dari hasil pengujian yang dilakukan *traffic behaviour* yang memiliki kinerja lebih baik adalah FTP (*File Transfer Protocol*) yang dapat dilihat pada tabel 4.1 sampai 4.8 dengan mengetahui kondisi simulasi yang dilakukan. Dengan mengetahui hasil tersebut, meskipun *traffic behaviour* jenis FTP merupakan jenis yang paling baik namun masih memiliki kelebihan dan kelemahan pada masing-masing *routing protocol* yang digunakan seperti pada *routing protocol* AODV pada skenario pertama *traffic behaviour* FTP ini mempunyai kelebihan yaitu performa yang baik pada efisiensi *Normalized Routing Load* namun pada *Routing Overhead* mengalami peningkatan yang besar ketika perubahan jumlah *node* sehingga efisiensi suatu *routing overhead* berkurang. Dan untuk mengetahui kelebihan dan kekurangan *routing protocol* lainnya dan skenario 2 dapat dilihat pada tabel 4.9 dan tabel 4.10.

DAFTAR ISI

HALAMAN SAMPUL	i
HALAMAN JUDUL	ii
DAFTAR ISI	xii
DAFTAR GAMBAR	xiv
DAFTAR TABEL	xv
BAB 1. PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah	3
1.4 Tujuan Penelitian	3
1.5 Manfaat Penelitian	4
BAB 2. TINJAUAN PUSTAKA	5
2.1 Jaringan Wireless	5
2.1.1 Standarisasi Teknologi <i>Wireless</i>	6
2.1.2 Topologi Jaringan <i>Wireless</i>	7
2.1.2.1 Topologi Jaringan <i>Ad Hoc</i> Pada MANET	8
2.2 <i>Mobile Ad Hoc Network</i> (MANET)	9
2.2.1 Karakteristik MANET	9
2.2.2 Protokol MANET	10
2.2.2.1 <i>Proactive Routing</i>	11
2.2.2.2 <i>Reactive Routing</i>	11
2.2.3 Protokol <i>Routing</i>	12
2.2.3.1 AODV (<i>Ad Hoc On demand Distance Vector</i>)	12
2.2.3.2 DSR (<i>Dynamic Source Routing</i>).....	14
2.2.3.3 DSDV (<i>Destination Sequenced Distance Vector</i>)	16
2.2.4 Pemanfaatan Teknologi MANET.....	18
2.3 Parameter Kualitas <i>Mobile Ad-Hoc Network</i> (MANET).....	19
2.3.1 <i>Routing Overhead</i> (RO).....	20
2.3.2 <i>Normalized Routing Load</i> (NRL)	20

2.4 <i>Network Simulator 2 (NS-2)</i>	20
2.4.1 Konsep Dasar NS-2	21
2.4.2 <i>Output</i> Simulasi NS-2.....	23
BAB 3. METODE PENELITIAN	25
3.1 Blok Diagram Sistem	25
3.2 <i>Flowchart</i> Simulasi MANET	26
3.3 Pemodelan Jaringan MANET di NS2	27
3.4 Parameter-parameter Simulasi	28
3.5 Skenario Penelitian	30
3.5.1 Skenario 1 Perubahan Jumlah <i>Node</i>	30
3.5.2 Skenario 2 Perubahan Kecepatan <i>Node</i>	30
BAB 4. HASIL DAN PEMBAHASAN	31
4.1 `Data Keluaran Simulasi	31
4.2 Analisis Kinerja <i>Traffic Behaviour</i> Terhadap Skenario 1	34
4.2.1 <i>Normalized Routing Load (NRL)</i>	34
4.2.2 <i>Routing Overhead (RO)</i>	40
4.3 Analisis Kinerja <i>Traffic Behaviour</i> Terhadap Skenario 2	45
4.3.1 <i>Normalized Routing Load (NRL)</i>	46
4.3.2 <i>Routing Overhead (RO)</i>	52
4.4 Kelebihan dan Kelemahan <i>Routing Protocol</i> Terhadap FTP	59
BAB 5. PENUTUP	61
5.1 Kesimpulan	61
5.2 Saran	61
DAFTAR PUSTAKA	62
LAMPIRAN	64

DAFTAR GAMBAR

2.1	Pembagian Jaringan <i>Wireless</i> Berdasarkan Jangkauannya	5
2.2	<i>Ad Hoc Wireless Network</i>	7
2.3	Topologi MANET	8
2.4	Mekanisme Penemuan Rute AODV	13
2.5	<i>Data Route Update</i> dan <i>Route Error</i> AODV	14
2.6	<i>Route Discovery</i> Pada <i>Routing DSR</i>	15
2.7	Pesan <i>Error DSR</i>	15
2.8	Komponen Pembangun NS	22
2.9	Arsitektur Dasar dari NS2	22
2.10	Tampilan <i>File Trace</i> Pada NS2	23
2.11	Tampilan <i>File Namtrace</i> Pada NS2.... ..	23
2.12	Tampilan Xgraph pada NS-2	24
3.1	Blok Diagram Sistem	26
3.2	<i>Flowchart</i> Simulasi MANET	27
3.3	Rancangan Jaringan MANET di NS2.	28
4.1	Perintah untuk Menjalankan Simulasi untuk Skenario 1 AODV	31
4.2	Awal <i>File Namtrace</i> Skenario 1 untuk 20 <i>Node</i> Pada AODV FTP	32
4.3	Akhir <i>File Namtrace</i> Skenario 1 untuk 20 <i>Node</i> Pada AODV FTP	32
4.4	Packet drop yang terjadi pada <i>traffic behaviour</i> jenis CBR	33
4.5	<i>File Trace</i> Skenario 1 untuk 20 <i>Node</i> Pada AODV FTP	33
4.6	Grafik Data Rata-rata Nilai NRL dari Perubahan Jumlah <i>Node</i> FTP	38
4.7	Grafik Data Rata-rata Nilai NRL dari Perubahan Jumlah <i>Node</i> CBR	39
4.8	Grafik Data Rata-rata Nilai RO dari Perubahan Jumlah <i>Node</i> FTP	44
4.9	Grafik Data Rata-rata Nilai RO dari Perubahan Jumlah <i>Node</i> CBR	45
4.10	Grafik Data Rata-rata Nilai NRL dari Perubahan Kecepatan <i>Node</i> FTP	50
4.11	Grafik Data Rata-rata Nilai NRL dari Perubahan Kecepatan <i>Node</i> CBR	51
4.12	Grafik Data Rata-rata Nilai RO dari Perubahan Kecepatan <i>Node</i> FTP	56
4.13	Grafik Data Rata-rata Nilai RO dari Perubahan Kecepatan <i>Node</i> CBR	57

DAFTAR TABEL

2.1	Perbedaan Standarisasi 802.11	7
3.1	Rincian Waktu Pelaksanaan Penelitian	25
3.2	Parameter-parameter Simulasi	29
3.3	Skenario 1 Parameter Berdasarkan Perubahan Jumlah <i>Node</i>	31
3.4	Skenario 2 Parameter Berdasarkan Perubahan Kecepatan <i>Node</i>	31
4.1	Data Rata-rata Nilai NRL dari Jumlah <i>Node</i> Terhadap FTP	36
4.2	Data Rata-rata Nilai NRL dari Jumlah <i>Node</i> Terhadap CBR	36
4.3	Data Rata-rata Nilai RO dari Jumlah <i>Node</i> Terhadap FTP.....	41
4.4	Data Rata-rata Nilai RO dari Jumlah <i>Node</i> Terhadap CBR.....	42
4.5	Data Rata-rata Nilai NRL dari Kecepatan <i>Node</i> Terhadap FTP	48
4.6	Data Rata-rata Nilai NRL dari Kecepatan <i>Node</i> Terhadap CBR	48
4.7	Data Rata-rata Nilai RO dari Kecepatan <i>Node</i> Terhadap FTP.....	54
4.8	Data Rata-rata Nilai RO dari Kecepatan <i>Node</i> Terhadap CBR.....	54
4.9	Kelebihan dan Kelemahan pada Skenario 1.....	59
4.10	Kelebihan dan Kelemahan pada Skenario 2.....	60

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Mobile Ad Hoc Network (MANET) terbentuk dari sekumpulan *node* yang menggunakan antarmuka nirkabel untuk melakukan komunikasi antara satu *node* dengan *node* yang lainnya. *Node* pada jaringan MANET tidak hanya berperan sebagai pengirim dan penerima data, namun dapat berperan sebagai penunjang *node* yang lainnya, oleh maka dari itu teknologi MANET sangat bermanfaat bagi dunia perang karena tidak membutuhkan suatu *base station* dan disetiap *node* memiliki kemampuan untuk proses *routing*.

Routing merupakan penentuan *route* terbaik oleh *node/router* dengan algoritma tertentu agar paket dari sumber sampai ditujuan dengan kecepatan yang optimal. *Routing protocol* diperlukan dalam jaringan untuk menunjang proses kirim dan terima paket data antar *node*, namun *routing protocol* yang cocok dengan jaringan MANET yaitu *routing protocol* AODV (*Ad Hoc On demand Distance Vector*) karena *routing* ini bekerja saat ada permintaan dari *source node* untuk mencari tahu jalur-jalur yang tidak ada *loop* dan menemukan jalur terpendek untuk menuju *node* tujuan (Hartati, 2014) dan *routing protocol* DSR (*Dynamic Source Routing*) karena apabila terdapat catatan mengenai rute yang dimaksud, paket akan dikirim melalui rute tersebut, namun apabila tidak ditemukan rute yang diinginkan, proses pencarian rute akan dilakukan serta DSDV (*Destination Sequenced Distance Vector*) merupakan *protocol* yang bersifat *proactive* yang berdasarkan pada algoritma bellman-ford dimana kontribusi algoritma ini adalah untuk mengatasi *routing loop* (Imawan, 2009).

Aliran trafik pada MANET sangat dibutuhkan untuk proses pengiriman data antara *node* pengirim dan *node* penerima, dalam hal ini penentuan aliran trafik sangat dibutuhkan dan di dalam penentuan aliran trafik tersebut belum terdapat standar untuk pemilihan tipe aplikasi apa yang harus digunakan dan di implementasikan pada MANET. Masalah muncul ketika proses penentuan tipe aplikasi atau *traffic behaviour* yang efisien dalam pengiriman paket data dari sumber ke tujuan. Dan permasalahan selanjutnya muncul ketika suatu kualitas

pada kinerja tipe aplikasi atau *traffic behaviour* yang dipilih untuk proses pengiriman dan penerimaan jalur yang digunakan oleh *node – node* tidak sesuai yang diharapkan untuk implementasi yang diinginkan.

Beberapa penelitian sebelumnya, dilakukan penelitian terhadap analisis performansi *routing hybrid wireless mesh protocol* (HWMP) pada *wireless mesh network* (WMN) berdasarkan standar IEEE 802.11s (Muhardafen *at al*, 2012). Parameter yang dilakukan untuk pengujian di penelitian oleh Fenmelin dkk adalah nilai *Normalized Routing Protocol* dan *Routing Overhead*, pada beberapa protokol *routing* dalam jaringan MANET (Imawan, 2009). Dalam penelitian ini menerangkan sistem jaringan MANET berfokus pada pengembangan aplikasi baru MANET seperti kolaborasi komputasi, sistem sharing *file peer to peer*, dan sistem pendidikan. Berkembangannya aplikasi MANET yang beragam mendorong adanya pola trafik yang lebih kompleks. Parameter yang digunakan di penelitian ini adalah PDR, *Average Delay*. Jadi pada kedua penelitian tersebut hanya ditekankan pada *routing* protokol pada MANET.

Parameter yang menentukan efisiensi atau tidaknya pada kinerja *traffic behaviour* yaitu perubahan terhadap jumlah dan kecepatan *node*. Dalam *traffic behaviour* yang sudah digunakan adalah jenis CBR (*Constant Bit Rate*) dengan *transport agent* UDP. Namun dalam hal ini kinerja *traffic behaviour* pada CBR sudah digunakan pada jenis jaringan WMN (*Wireless Mesh Network*) yang cenderung kurang efisiensi (Muhardafen *at al* 2012). Oleh karena itu peneliti melakukan peninjauan kembali dari penelitian sebelumnya dan melakukan penelitian terhadap kinerja *traffic behaviour* pada MANET dengan menggunakan *network simulator 2* (NS-2). Skenario yang dimaksud adalah dalam membuat sebuah perancangan MANET terdapat sebuah bagian yang perlu dibandingkan performansi dari beberapa jenis *traffic behaviour*. Performa *traffic behaviour* tersebut akan diukur berdasarkan *Normalized Routing Load* (NRL) dan *Routing Overhead* (RO) dengan skenario yang berbeda.

1.2 Rumusan Masalah

Dalam penelitian yang diusulkan dalam proposal ini adalah beberapa hal yang menjadi rumusan masalah antara lain :

1. Bagaimana Peformansi kinerja *traffic behaviour* pada *mobile ad hoc network* dengan *routing protocol* AODV, DSR, dan DSDV ?
2. Bagaimana hasil perbandingan dari hubungan jumlah dan kecepatan *node* terhadap kinerja *traffic behaviour* pada *routing protocol* yang digunakan ?

1.3 Batasan Masalah

Dari latar belakang penelitian diatas, agar pembahasan tidak terlalu luas maka diperlukan suatu pembatasan masalah sebagai berikut :

1. Simulasi menggunakan program NS-2 versi 2.31 (NS-2.31), dan untuk *routing protocol* yang digunakan adalah tipe *reactive* (AODV, DSR) dan tipe *proactive* (DSDV), serta *transpont agent* dan *traffic behaviour* yang digunakan TCP (TCPSink) untuk FTP dan UDP (Null) untuk CBR, sedangkan hasil *output* dalam bentuk GNUPLOT.
2. Banyaknya *node* dan kecepatan yang digunakan 20,30 dan 50 *node* serta 5, 10, 15, 20 m/s.
3. Parameter kinerja yang digunakan adalah *Normalized Routing Load* (NRL), *Routing Overhead* (RO).
4. Pergerakan *node* diatur secara manual dan pada setiap skenario mengalami pergerakan yang sama, sedangkan skenario penelitian yang dilakukan hanya fokus pada hubungan jumlah dan kecepatan *node*.

1.4 Tujuan

Dalam penelitian yang diusulkan dalam proposal ini memiliki beberapa tujuan yang diantaranya :

1. Mengetahui kinerja *traffic behaviour* pada *mobile ad-hoc network* dengan menggunakan NS-2.
2. Menganalisis hasil perbandingan dari jumlah dan kecepatan *node* terhadap kinerja *traffic behaviour* pada *routing protocol*.

1.5 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut :

1. Mengetahui *traffic behaviour* yang sesuai dengan *Mobile Ad hoc Network* terhadap performansi atau kinerja yang dihasilkan dengan *routing protocol* AODV, DSR, dan DSDV.
2. Mengetahui *traffic behaviour* yang lebih baik untuk *Mobile Ad hoc Network* terhadap perubahan jumlah dan kecepatan *node* dengan *routing protocol* AODV, DSR, dan DSDV.

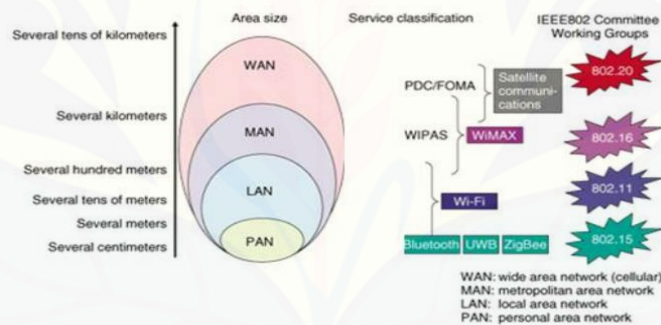


BAB 2. TINJAUAN PUSTAKA

2.1 Jaringan Wireless

Wireless network atau jaringan tanpa kabel adalah salah satu jenis jaringan berdasarkan media komunikasinya, yang memungkinkan perangkat-perangkat didalamnya seperti komputer, hp, dan lain-lain, bisa saling berkomunikasi secara *wireless*/tanpa kabel. *Wireless network* umumnya diimplementasikan menggunakan komunikasi radio. Implementasi ini berada pada *level* lapisan fisik (*physical layer*) dari OSI model.

Pada jaringan *wireless* dapat dibagi berdasarkan jangkauannya yang berfungsi sebagai mekanisme pembawa antara peralatan atau antar peralatan dan jaringan kabel tradisional (jaringan perusahaan dan internet) (Rudi *at al*, 2011:3). Dan Jaringan *wireless* dibagi dalam beberapa kategori, berdasarkan jangkauan area yaitu:



Gambar 2.1. Pembagian Jaringan *Wireless* Berdasarkan Jangkauannya

(Sumber : Rudi *at al*, 2011:4)

a. *Wireless* PAN (WPAN)

Wireless Personal Area Network (WPAN) adalah jaringan *wireless* dengan jangkauan area yang kecil. Contohnya *Bluetooth*, *Infrared*, dan *ZigBee*.

b. *Wireless* LAN (WLAN) / Wifi

Wireless Local Area Network (WLAN) atau biasa disebut *Wifi* memiliki jangkauan yang jauh lebih luas dibanding WPAN. Saat ini WLAN mengalami banyak peningkatan dari segi kecepatan dan luas cakupannya. Awalnya WLAN ditujukan untuk penggunaan perangkat jaringan lokal, namun saat ini lebih banyak digunakan untuk mengakses *internet*.

c. *Wireless* MAN (WMAN)

Wireless Metropolitan Area Network (WMAN) adalah jaringan *wireless network* yang menghubungkan beberapa jaringan WLAN. Contoh teknologi WMAN adalah WiMAX.

d. *Wireless* WAN (WWAN)

Wireless Wide Area Network adalah jaringan *wireless* yang umumnya menjangkau area luas misalnya menghubungkan kantor pusat dan cabang antar provinsi.

e. *Cellular Network*

Cellular Network atau *Mobile Network* adalah jaringan radio terdistribusi yang melayani media komunikasi perangkat *mobile* seperti *handphone*, *pager*, dan lain-lain. Contoh sistem dari *Cellular Network* ini adalah GSM, PCS, dan D-AMPS.

Keunggulan teknologi *wireless* adalah lebih murah dalam biaya pengimplementasian dan perawatan infrastruktur jaringan fleksibilitas. Perangkat yang menggunakan teknologi *wireless* bisa berpindah tempat selama masih didalam *coverage area*, tanpa harus mengurangi fungsionalitas jaringan tersebut. Kekurangan teknologi *wireless* adalah dipengaruhi oleh faktor luar seperti cuaca, hujan, salju dan kabut bisa mempengaruhi penyebaran sinyal bahkan sampai berkurang 50%-nya, halangan seperti pohon, gedung juga bisa mempengaruhinya.

2.1.1 Standarisasi Teknologi *Wireless*

Proyek 802, *protocol* yang dikenal di *wireless local area network* (WLAN) adalah IEEE 802.XX. Arti dari 802.XX merupakan komite yang bergerak dalam standarisasi *Institute of Electrical and Electronics Engineers* (IEEE) tersebut dibentuk pada bulan Februari tahun 1980, sehingga standarisasi diberi nama 802.XX.

Jaringan *wireless* yang populer saat ini adalah *Bluetooth*, *WiFi* dan *Wimax* yang juga standarisasi *wireless*. Secara umum berlaku standarisasi IEEE 802.15, IEEE 802.11 (a,b,g), dan IEEE 802.16. Perbedaan yang paling utama dari

standarisasi diatas adalah kecepatan *transfer* data. Pada standarisasi IEEE 802.11 (a,b,g) yang di implementasikan pada WiFi merupakan sekumpulan standar yang digunakan untuk jaringan lokal nirkabel yang didasari pada spesifikasi IEEE 802.11 (Rudi *at al*, 2011:7). Dalam standarisasi IEEE 802.11 memiliki perbedaan yang diantaranya.

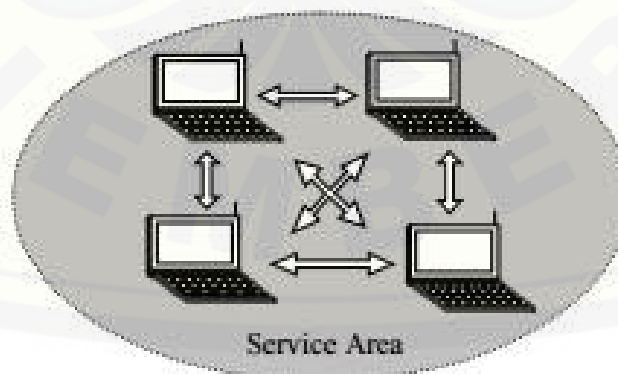
Tabel 2.1. Perbedaan standarisasi 802.11

Perbedaan 802.11 a/b/g/n			
Spesifikasi	Kecepatan	Frekuensi Band	Cocok dengan
802.11a	11 Mbps	5 Ghz	b
802.11b	54 Mbps	2.4 Ghz	a
802.11g	54 Mbps	2.4 Ghz	b, g
802.11n	100 Mbps	2.4 Ghz dan 5 Ghz	b, g, n

(Sumber : Rudi *at al*, 2011:8)

2.1.2 Topologi Jaringan *Wireless*

Jaringan *computer* pada umumnya mempunyai suatu topologi jaringan seperti topologi *bus*, *ring*, *star*, *tree*, *mesh*, dan lainnya. Namun pada jaringan *wireless* ini mempunyai arsitektur topologi tersendiri yaitu topologi *infrastructure* dan topologi *Ad Hoc*. Namun pada penelitian ini akan dibahas tentang topologi yang terdapat dalam jaringan *Ad Hoc*.



Gambar 2.2. *Ad Hoc Wireless Network*

(Sumber : Sun. 2010)

2.1.2.1 Topologi Jaringan *Ad-Hoc* Pada MANET

Pada tiap jaringan yang dibangun pasti menggunakan topologi jaringan tertentu, hal ini penting karena dengan adanya topologi jaringan maka alur-alur komunikasi antara *host* dan *router* pada suatu jaringan dapat diketahui. Jika pada jaringan kabel seperti *local area network* (LAN) memiliki beberapa model topologi yang dapat dipakai, berbeda dengan jaringan *ad hoc* pada MANET yang cenderung tidak memakai topologi yang spesifik. Hal ini dikarenakan pergerakan *node* pada jaringan MANET yang bebas sehingga menyebabkan topologi berubah-ubah sesuai letak tiap *node*.



Gambar 2.3. Topologi Manet

(Sumber : Imawan, 2009)

Tiap *node* pada MANET seperti yang ditunjukkan gambar 1 memiliki jangkauan sinyal tertentu sehingga komunikasi antara dua *node* yang saling berjauhan dapat dilakukan dengan melewati jalur *node* penghubung yang berada di tengah posisi kedua *node* tersebut. Kondisi komunikasi dua *node* inilah yang dikatakan sebagai komunikasi antara *router* dan *host* seperti pada jaringan kabel pada umumnya, karena tiap *node* pada MANET dapat bersifat sebagai *router*

maupun sebagai *host*. Faktor-faktor inilah yang kemudian menjadikan topologi yang dimiliki oleh MANET bersifat dinamis atau berubah-ubah.

2.2 *Mobile Ad Hoc Network* (MANET)

Mobile Ad Hoc Network (MANET) menurut (Sun, 2010) adalah kumpulan dari beberapa *wireless node* yang dapat di *setup* secara dinamis dimana saja dan kapan saja tanpa menggunakan infrastruktur jaringan yang ada. MANET juga merupakan jaringan sementara yang dibentuk oleh beberapa *mobile node* tanpa adanya pusat administrasi dan infrastruktur kabel. Pada MANET, *mobile host* yang terhubung dengan *wireless* dapat bergerak bebas dan juga berperan sebagai *router*.

Menurut (Sun, 2010) terdapat beberapa perbedaan antara jaringan *ad hoc* dengan jaringan yang memiliki infrastruktur, antara lain :

- a. *Peer to Peer*, yaitu komunikasi antara dua *node* dalam satu *hop*.
- b. *Remote to Remote*, yaitu komunikasi antar dua *node* diluar satu *hop*, namun masih tetap mengelola kestabilan rute di antara keduanya.
- c. *Dynamic Traffic*, terjadi ketika *node* bergerak, maka rute harus dikonstruksi ulang. Ini merupakan hasil dari tingkat konektifitas yang rendah.

2.2.1 Karakteristik MANET

Menurut (Prasant *at al.* 2005:4), berdasarkan dokumen *Request for Comments* menjelaskan bahwa terdapat beberapa karakteristik dari *Mobile Ad Hoc Network* (MANET). Disana dijelaskan bahwa MANET terdiri dari *mobile platform* (seperti *router* dan perangkat *wireless*) dalam hal ini disebut dengan “*node*” yang bebas berpindah-pindah ke mana saja. *Node* tersebut bisa saja berada di pesawat, kapal, mobil dan dimana saja.

Setiap *node* dilengkapi dengan *transmitter* dan *receiver wireless* menggunakan antena atau sejenisnya yang bersifat *omnidirectional (broadcast)*, *highly directional (point to point)*, memungkinkan untuk diarahkan, atau kombinasi dari beberapa hal tersebut. *Omnidirectional* maksudnya adalah gelombang radio dipancarkan ke segala arah oleh perangkat *transmitter wireless*.

Sedangkan *highly directional* adalah gelombang dipancarkan ke satu arah tertentu. Selain karakteristik di atas, *Ad hoc Network* juga memiliki beberapa karakteristik yang lebih menonjol, antara lain :

- a. Topologi yang dinamis : *Node* pada MANET memiliki sifat yang dinamis, yaitu dapat berpindah-pindah kemana saja. Maka topologi jaringan yang bentuknya adalah loncatan antara *hop* ke *hop* dapat berubah secara tidak terpolo dan terjadi secara terus menerus tanpa ada ketetapan waktu untuk berpindah. Bisa saja didalam topologi tersebut terdiri dari *node* yang terhubung ke banyak *hop* lainnya, sehingga sangat berpengaruh secara signifikan terhadap susunan topologi jaringan.
- b. Otonomi : Setiap *node* pada MANET berperan sebagai *end user* sekaligus sebagai *router* yang menghitung sendiri *route path* yang selanjutnya akan dipilih.
- c. Keterbatasan *bandwidth* : *Link* pada jaringan *wireless* cenderung memiliki kapasitas yang rendah jika dibandingkan dengan jaringan berkabel. Jadi, kapasitas yang keluar untuk komunikasi *wireless* juga cenderung lebih kecil dari kapasitas maksimum transmisi. Efek yang terjadi pada jaringan yang berkapasitas rendah adalah *congestion* (kemacetan).
- d. Keterbatasan energi : Semua *node* pada MANET bersifat *mobile*, sehingga sangat dipastikan *node* tersebut menggunakan tenaga baterai untuk beroperasi. Sehingga perlu perancangan untuk optimalisasi energi.
- e. Keterbatasan keamanan : Jaringan *wireless* cenderung lebih rentan terhadap keamanan daripada jaringan berkabel. Kegiatan pencurian (*eavesdropping*, *spoofing* dan *denial of service*) harus lebih diperhatikan.

2.2.2 Protokol MANET

Menurut (Cardei. 2004), terdapat berbagai jenis protokol *routing* untuk MANET yang secara keseluruhan dapat dibagi menjadi beberapa kelompok, antara lain adalah *Proactive Routing*, *Reactive Routing*, yang masing-masing kelompok dibagi menjadi beberapa bagian lagi.

2.2.2.1 *Proactive Routing*

Algoritma ini akan mengelola daftar tujuan dan rute terbaru masing-masing dengan cara mendistribusikan *routing table* ke seluruh jaringan, sehingga jalur lalu lintas (*traffic*) akan sering dilalui oleh *routing table* tersebut. Hal ini akan memperlambat aliran data jika terjadi restrukturisasi *routing table*.

Beberapa contoh algoritma *proactive routing* adalah :

1. Babel
2. B.A.T.M.A.N (*Better Approach to Mobile Ad Hoc Network*)
3. DSDV – *Highly Dynamic Destination Sequenced Distance Vector routing protocol*
4. HSR – *Hierarchical State Routing Protocol*
5. IARP – *Intrazone Routing Protocol*
6. LCA – *Linked Cluster Architecture*
7. WAR – *Witness Aided Routing*
8. OLSR – *Optimized Link State Routing Protocol*

2.2.2.2 *Reactive Routing*

Tipe ini akan mencari rute (*on demand*) dengan cara membanjiri jaringan dengan paket *router request*. Sehingga dapat menyebabkan jaringan akan penuh (*clogging*).

Beberapa contoh algoritma *reactive routing* adalah :

1. SENCAS
2. *Reliable Ad Hoc On Demand Distance Vector Routing Protocol*
3. *Ant-Based Routing Algorithm for Mobile Ad Hoc Network*
4. *Admission Control Enabled On Demand Routing (ACOR)*
5. Ariadne
6. *Associativity Based Routing*
7. *Ad Hoc On Demand Distance Vector (AODV)*
8. *Ad Hoc On Demand Multipath Distance Vector*
9. *Backup Source Routing*
10. *Dynamic Source Routing (DSR)*

11. *Flow State in the Dynamic Source Routing*

12. *Dynamic MANET On Demand Routing (DYMO)*

2.2.3 Protokol Routing

Menurut (Imawan. 2009) *protocol routing* adalah standarisasi yang melakukan kontrol terhadap bagaimana sebuah *node* dapat meneruskan paket diantara perangkat komputasi dalam jaringan *mobile ad-hoc network* (MANET).

Dalam jaringan *ad hoc*, setiap *node* tidak mempunyai pengetahuan mengenai topologi jaringan sekitarnya, melainkan *node-node* tersebut harus dicari. Dasar pemikirannya adalah bahwa *node* baru yang masuk akan mendengarkan pesan *broadcast* dari tetangganya. Sebuah *node* akan mempelajari *node* baru didekatnya dan bagaimana cara menjangkau *node* baru tersebut. Pada suatu saat, setiap *node* akan mengetahui tentang *node-node* lain dan mengetahui bagaimana cara menjangkaunya.

Protocol routing layaknya sebuah *router* yang berkomunikasi dengan perangkat lain untuk menyebarkan informasi dan mengizinkan adanya pemilihan rute diantara dua *node* dalam jaringan. Pada jaringan *ad hoc* setiap *node* akan memiliki kemampuan layaknya *router* yang meneruskan pesan antar *node* di sekitarnya. Untuk itu dibutuhkan *protocol routing* untuk membantu tiap-tiap *node* melakukannya.

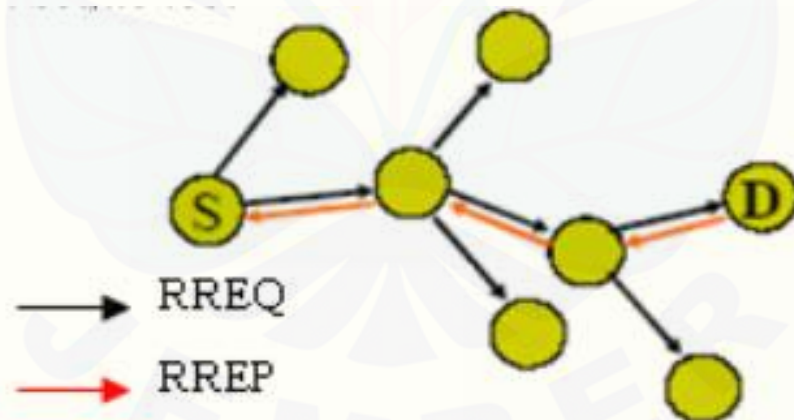
Protokol routing untuk jaringan *ad hoc* tentunya berbeda dengan protokol *routing* yang biasa diimplementasikan pada jaringan kabel. Hal ini disebabkan sifat jaringan *ad hoc* yang dinamis, sehingga memiliki topologi yang berubah-ubah, berbeda dengan jaringan kabel yang cenderung tetap. Dibawah ini beberapa protokol *routing* yang digunakan dalam penelitian ini adalah *Ad Hoc On demand Distance Vector (AODV)*, *Dynamic Source Routing (DSR)*, dan *Destination Sequenced Distance Vector (DSDV)*.

2.2.3.1 AODV (*Ad Hoc On demand Distance Vector*)

Protokol *Ad-hoc On-demand Distance Vector (AODV)* adalah protokol yang mekanismenya merupakan perpaduan antara protokol *DSR (on demand)* dan

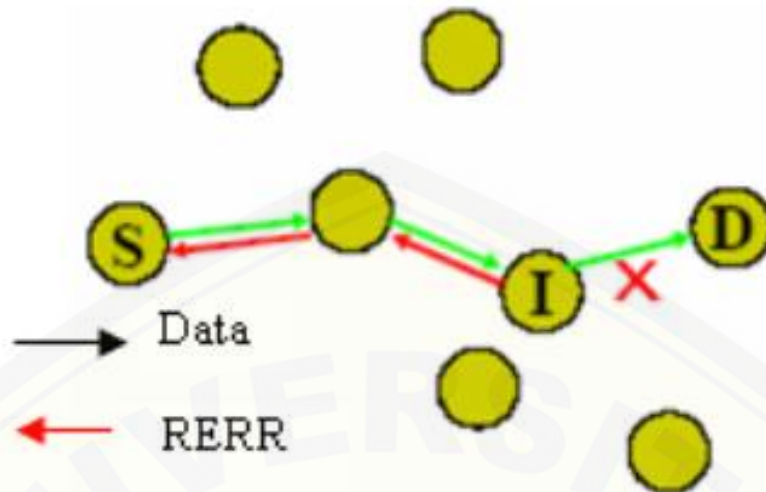
protokol DSDV (*hop by hop*). Protokol ini melakukan mekanisme layaknya pada DSR, yaitu adanya rute pencarian (*Route Discovery*) dan rute pemeliharaan (*Route Maintenance*), dan proses *routing* dilakukan secara *hop by hop* layaknya pada protokol DSDV.

AODV menggunakan tabel *routing* dengan satu *entry* untuk setiap tujuan. Tanpa menggunakan *routing* sumber, AODV mempercayakan pada tabel *routing* untuk menyebarkan *Route Reply* (RREP) kembali ke sumber dan secara sekuensial akan mengarahkan paket data menuju tujuan. AODV juga menggunakan *sequence number* untuk menjaga setiap tujuan agar didapat informasi *routing* yang terbaru dan untuk menghindari *routing loops*. Semua paket yang diarahkan membawa *sequence number* ini. Penemuan jalur (*Path discovery*) atau *Route discovery* diinisiasi dengan menyebarkan *Route Reply* (RREP) (Prasant *at al.* 2005:73), seperti terlihat pada Gambar 2.4. Ketika RREP menjelajahi *node*, ia akan secara otomatis *setup path*. Jika sebuah *node* menerima RREP, maka *node* tersebut akan mengirimkan RREP lagi ke *node* atau *destination sequence number*. Pada proses ini, *node* pertama kali akan mengecek *destination sequence*.



Gambar 2.4. Mekanisme Penemuan Rute AODV

(Sumber: Arifin *at al.* 2010)



Gambar 2.5. Data Route Update dan Route Error AODV

(Sumber: Arifin *at al.* 2010)

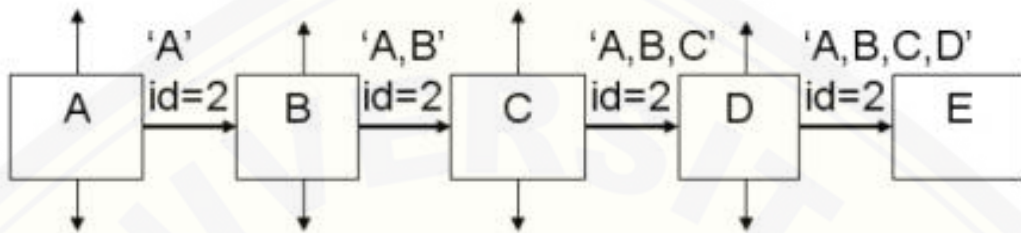
Number pada tabel *routing*, apakah lebih besar dari 1 pada *route request* (RREQ), jika benar, maka *node* akan mengirim RREP. Ketika RREP berjalan kembali ke *source* melalui *path* yang telah di *setup*, ia akan *setup* jalur kedepan dan memperbarui *time out*. Jika sebuah *link* ke *hop* berikutnya tidak dapat dideteksi dengan metode penemuan rute, maka *link* tersebut akan diasumsikan putus dan *Route Error* (RERR) akan disebarkan ke *node neighbour* seperti terlihat pada Gambar 2.5. Dengan demikian sebuah *node* bisa menghentikan pengiriman data melalui rute ini atau meminta rute baru dengan menyebarkan RREQ kembali. (Arifin *at al.* 2010).

2.2.3.2 DSR (Dynamic Source Routing)

Dynamic Source Routing (DSR) protokol adalah sebuah protokol *routing* yang sederhana dan efisien dirancang khusus untuk digunakan dalam jaringan *wireless multihop ad hoc mobile node*. DSR memungkinkan sebuah jaringan bisa mengatur dan konfigurasi sendiri tanpa memerlukan infrastruktur jaringan yang ada atau admin. DSR telah dilakukan oleh berbagai kelompok, dan dicoba di beberapa *testbeds*. Jaringan yang menggunakan protokol DSR telah terhubung ke Internet. DSR dapat *interoperate* dengan *Mobile IP*, dan *node* menggunakan

Mobile IP dan *DSR* telah bisa bermigrasi antara *WLAN*, layanan data seluler dan *DSR* jaringan *mobile ad hoc*.

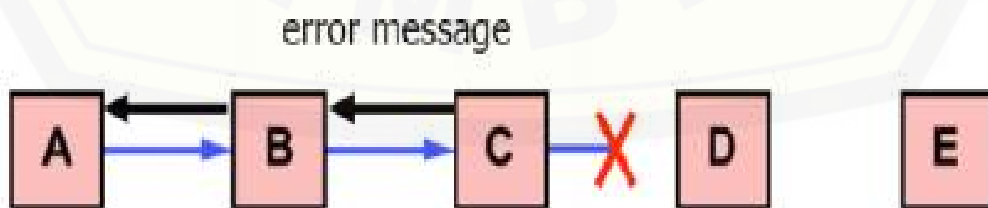
Protokol ini terdiri dari dua mekanisme utama "*Route Discovery*" dan "*Route Maintenance*", yang bekerja sama untuk memungkinkan *node* untuk menemukan dan mempertahankan rute ke tujuan dalam jaringan *ad hoc*.



Gambar 2.6. *Route Discovery* pada *Routing DSR*

Jika *node A* memiliki *Route Cache* rute ke *node E*, maka rute ini segera digunakan. Jika tidak, *protocol discovery route* dimulai:

- Node A* mengirim paket *routerrequest* dengan cara *flooding* pada jaringan.
- Jika *node B* mendapat paket *routerrequest* dari target yang sama (*node A*) atau jika alamat *node B* sudah terdaftar di *record route*, kemudian *node B* membuang *route request* tersebut.
- Jika *node B* adalah target dari *route discovery*, maka *node B* akan mengirim *routerreply* ke *node A*. *Routerreply* berisi daftar dari sumber *node* ke tujuan dengan jarak terpendek. Jika *A* menerima *route reply*, daftar tersebut akan disimpan dan akan dikirim ke *node* berikutnya untuk menuju target.
- Jika *node B* bukan tujuannya, maka *node B* akan mengirim *route request* ke *neighbore node* untuk melanjutkan *routing* nya.



Gambar 2.7. *Pesan Error*

Pada setiap *node* DSR bertanggung jawab untuk mengirimkan dan mengkonfirmasi bahwa *hop* berikutnya dalam sumber *Route* menerima paket. Juga masing-masing paket hanya diteruskan sekali oleh *node* (*hop by hop routing*). Jika sebuah paket tidak dapat diterima oleh sebuah *node*, paket ditransmisikan ulang beberapa kali sampai jumlah maksimum dan konfirmasi diterima dari *hop* berikutnya. Hanya jika hasil retransmisi mengalami kegagalan, pesan *Route Error* dikirim ke inisiator, yang dapat menghapus bahwa Sumber Rute dari *Route Cache*. Jadi inisiator dapat memeriksa *Route Cache* untuk rute lainnya untuk target. Jika tidak ada rute di *cache*, paket *Route Request* dikirimkan lagi (Prasant *at al.* 2005:72).

2.2.3.3 DSDV (*Destination Sequenced Distance Vector*)

DSDV merupakan algoritma *routing protocol ad hoc* proaktif yang didasari pada Bellman–Ford yang pertama kali dikenalkan, kontribusi algoritma ini adalah untuk mengatasi *Routing Loop*. Pada DSDV, digunakan *sequence number* untuk mengirimkan pesan pada jaringan. *Sequence number* dihasilkan juga saat ada perubahan dalam jaringan, hal ini terjadi karena sifat *table routing node* pada jaringan yang menggunakan protokol proaktif yang *update* secara periodik, serta *Triggered update* ulang digunakan oleh *node* untuk mengupdate *node* yang masuk dan keluar dari jaringan.

Dalam metode *routing* DSDV, setiap *node* yang berada dalam jaringan, akan memelihara tabel *routing* ke *node* tetangganya, tabel *routing* yang dimiliki oleh setiap *node* berisi tentang : alamat tujuan *node*, jumlah *hop* yang diperlukan untuk mencapai tujuan, serta *sequenced number*. Jika tabel *routing* dalam satu *node* telah diupdate, maka akan dipilih *rute* untuk mencapai *node* tujuan dengan beberapa pertimbangan sebagai berikut :

- a. Memiliki *sequence number* yang terbaru, hal ini dapat dilihat dari nilai *sequenced number* yang paling besar.
- b. Jika nilai *sequence number* sama, maka akan dilihat nilai metriknya, nilai metrik yang lebih kecil akan dipilih.

2.2.3.3.1 Kelebihan dan Kelemahan *Routing Protocol* DSDV

1. Penerapan Protocol DSDV memiliki kelebihan yaitu :
 - a. DSDV menjamin tidak ada *looping route*.
 - b. DSDV dapat mereduksi masalah *count to infinity*.
 - c. DSDV dapat menghindari trafik lebih dengan kenaikan drastis *update* penuh untuk *dump*.
 - d. DSDV hanya memperbaiki *path* terbaik menuju tujuan, dari sekian banyak *path* ketujuan.

2. Keterbatasan yang dimiliki algoritma *routing protocol* DSDV ini menjadi kelemahannya, yaitu :
 - a. Banyak memakan alokasi *bandwith* untuk *broadcasting* informasi yang tidak terlalu berguna, meskipun tidak ada perubahan yang berarti pada topologi jaringan, hal ini dikarenakan sifatnya yang selalu *update* setiap waktunya.
 - b. DSDV tidak mendukung *multipath routing*.
 - c. Sangat sulit untuk mendeterminasi waktu *delay* saat *broadcasting* informasi ke semua *node*.
 - d. Table *routing* pada *node* di jaringan DSDV sangat sulit untuk di *maintenance* pada jaringan berskala besar, mengingat jumlah *node* yang banyak dan luas jaringan yang semakin diperluas (*scalable*) yang meningkatkan *overhead*.

2.2.3.3.2 Proses Algoritma *Routing Protocol* DSDV

Berikut merupakan proses secara keseluruhan algoritma *routing protocol* DSDV :

- a. Diawal transmisi sebelum dilakukan , tiap *node* memiliki *table* yang berisi data *node-node* dalam jaringan.
- b. Jika terjadi perubahan topologi jaringan setelah paket data dikirimkan, *table routing* akan diperbarui secara periode

- c. Jika tidak ada permasalahan pada topologi jaringan, maka *node* akan dikirimkan setelah *node* sumber dan *node* tujuan di inialisasi, lalu paket akan sampai ke *node* tujuan.

2.2.4 Pemanfaatan Teknologi MANET

Dalam penelitian sebelumnya MANET digunakan pada beberapa teknologi yang terdiri dari WMN (*Wireless Mesh Network*) dan penelitian lainnya diteliti pada kinerja pola-pola trafik dengan beberapa protokol.

Pada penelitian pertama yaitu yang memanfaatkan *Wireless Mesh Network* dengan judul penelitian yaitu “Analisis Performansi *Routing Hybrid Wireless Mesh Protocol* (HWMP) Pada *Wireless Mesh Network* (WMN) Berdasarkan IEEE 802.11s”.

Dalam penelitian ini menerangkan bahwa *wireless mesh network* (WMN) merupakan jaringan komunikasi *wireless* yang terbentuk dari *node* radio dimana minimal terdapat dua atau lebih jalur komunikasi data pada setiap *node*. Didalam WMN terdapat beberapa *node client* yang dapat bergerak dengan kecepatan lambat (seperti kecepatan pejalan kaki) yang memungkinkan jaringan ini menjadi dinamis. Apalagi WMN merupakan jaringan yang *wireless* berarti akan memungkinkan ada *user* dalam jaringan yang terkadang bertambah ataupun berkurang. Dengan adanya perubahan kecepatan dan jumlah *user*, WMN membutuhkan suatu *protocol routing* yang handal untuk mencari jalur yang terbaik. *Protocol routing* yang digunakan adalah HWMP (*Hybrid*) yang merupakan default *protocol routing* dalam WMN, sementara AODV (reaktif), dan DSDV (proaktif) dijadikan sebagai pembanding *protocol routing* dari HWMP. Dengan *protocol routing* ini akan diamati perubahan yang terjadi berdasarkan parameter uji seperti *routing overhead* (RO) dan *normalized routing load* (NRL) yang akan disimulasikan pada *network simulator 2* (NS-2). Hasil nilai simulasi sudah diperoleh dengan menyimpulkan bahwa *protocol routing* DSDV merupakan *protocol routing* yang paling efisien dalam pengujian ini. (Muhardafen *at al*, 2012).

Dan penelitian lainnya yang membahas tentang pola-pola trafik pada beberapa *protocol routing* dalam jaringan MANET, pada penelitian ini sangat mendukung dalam proses penelitian yang akan dilakukan karena pada penelitian ini hampir memiliki kesamaan namun pada penelitian ini berfokus pada pengembangan aplikasi baru MANET seperti kolaborasi *game*, kolaborasi komputasi, sistem perpesanan, sistem *sharing file peer to peer*, dan kolaborasi sistem pendidikan. Berkembangnya aplikasi MANET yang beragam mendorong adanya pola trafik yang lebih kompleks.

Langkah awal yang efektif untuk mendukung beragam aplikasi MANET yang baru, dalam penelitian ini dipelajari pengaruh kinerja dari beragam pola trafik *protocol routing* dalam jaringan MANET, dan menganalisis performa sebagai akibat dari pola-pola trafik pada beberapa *protocol routing* melalui simulasi yang mendalam terhadap jaringan *ad hoc*. (Imawan, 2009).

Jaringan MANET bukan hanya dimanfaatkan untuk penelitian yang sudah dijelaskan diatas, selain dari penelitian tersebut jaringan MANET banyak digunakan di bidang lainnya atau diterapkan pada implementasi yang lain, karena MANET memiliki perkembangan yang cukup pesat sehingga MANET sangat banyak diteliti.

Dari penelitian diatas didapatkan permasalahan yaitu penentuan *routing protocol* yang cocok untuk MANET dan jenis *traffic behaviour* yang paling cocok untuk jaringan MANET dan *routing protocol* yang efisien dalam pengujian parameter *routing overhead* dan *normalized routing load* pada skenario perubahan kecepatan *node* dan perubahan jumlah *node*. Dalam penelitian ini dilakukan dengan maksimal perubahan *node* adalah 50 *node* dan maksimal perubahan kecepatan *node* adalah 20 m/s.

2.3 Parameter Kualitas Mobile Ad Hoc Network (MANET)

Menurut (Sidharta. 2013) dalam mengukur atau mengetahui kualitas pada jaringan *mobile ad-hoc network* ada beberapa parameter yang dapat mempengaruhi *performance* pada kualitas jaringan *ad-hoc*, parameter ini

berfungsi untuk mengetahui kualitas pada jaringan *ad hoc* tersebut, dan parameter – parameter yang mendukung adalah :

- a. *Routing Overhead* (RO)
- b. *Normalized Routing Load* (NRL)

2.3.1 *Routing Overhead* (RO)

Routing Overhead adalah jumlah total *packet routing* yang dikirimkan oleh *routing protocol* dalam setiap pengiriman paket data. Jadi, nilai parameter ini dilihat dari sisi *node* sumber ataupun juga *node intermediate*. Parameter ini digunakan untuk menghitung efisiensi kinerja suatu *routing protocol*. Persamaan 2.1 menunjukkan perhitungan untuk mendapatkan nilai dari *Routing Overhead*.

$$RO = \frac{\sum \text{Paket Routing terkirim}}{\sum \text{Paket Data Terkirim}} \quad (2.1)$$

2.3.2 *Normalized Routing Load* (NRL)

Normalized Routing Load yaitu jumlah *packet routing* yang ditransmisikan pada setiap paket data yang diterima di tujuan. Jadi, nilai parameter ini dilihat dari sisi *node* tujuan. Semakin tinggi nilai perbandingan banyaknya paket *routing* dengan paket data maka kinerja suatu *protocol routing* semakin kurang efisien. Persamaan 2.2 menunjukkan perhitungan untuk mendapatkan nilai dari NRL.

$$NRL = \frac{\sum \text{Paket Routing diterima}}{\sum \text{Paket Data diterima}} \quad (2.2)$$

2.4 *Network Simulator 2* (NS-2)

Network Simulator 2 (NS-2) merupakan perangkat lunak simulator jaringan yang bersifat *even-driven* yang sering digunakan untuk menganalisa kondisi jaringan yang bersifat dinamis selama komunikasi jaringan berlangsung. Simulasi terhadap komunikasi jaringan kabel maupun jaringan *wireless* beserta protokol-protokol yang bekerja didalam jaringan dapat dilakukan pada simulator NS2. Secara umum, NS2 menyediakan fasilitas kepada pengguna untuk menentukan

protokol jaringan yang akan digunakan didalam simulasi jaringan dan mensimulasikan komunikasi didalam jaringan berdasarkan protokol yang telah ditentukan. Karena sifat yang fleksibel dan modular dari simulator NS2, NS2 mulai menjadi sangat populer digunakan sebagai simulator jaringan oleh peneliti di bidang jaringan komputer (Megawan. 2013).

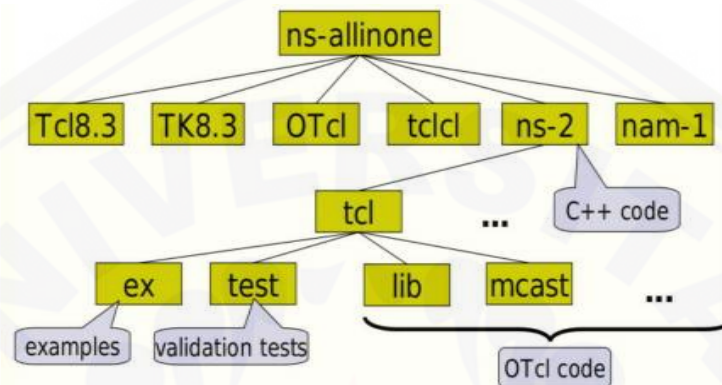
Selain itu juga dapat digunakan untuk mensimulasikan protokol jaringan (TCP/UDP/RTP), *Traffic behaviour* (FTP, Telnet, CBR), *Queue management* (RED, FIFO, CBQ), algoritma *routing unicast* (*Distance Vector*, *Link State*) dan *multicast* (PIM SM, PIM DM, DVRMP, *Shared Tree* dan *Bidirectional Shared Tree*), aplikasi multimedia yang berupa *layered video*, *Quality of Service video-audio* dan *transcoding*. NS juga mengimplementasikan beberapa MAC (IEEE 802.3, 802.11) di berbagai media, misalnya jaringan berkabel (seperti LAN, WAN, *point to point*), jaringan tanpa kabel (seperti *mobile IP*, *wireless LAN*), bahkan simulasi hubungan antar *node* jaringan yang menggunakan media satelit (Aji Sasongko *at al.* 2010).

Komponen utama lainnya pada NS di samping *object* jaringan adalah penjadwal kejadian. Sebuah kejadian dalam NS merupakan suatu paket ID yang unik untuk sebuah paket dengan waktu yang terjadwal dan penunjuk untuk suatu objek yang menangani masalah kejadian. Dalam NS, suatu penjadwal kejadian menjaga jalur dari waktu simulasi dan membawa seluruh kejadian dalam antrian kejadian terjadwal untuk saat ini dengan bantuan komponen jaringan yang sesuai, di mana biasanya hal tersebut merupakan salah satu hal yang disebutkan dalam kejadian, dan membiarkan hal tersebut menjalankan tindakan yang sesuai yang diwakilkan dengan paket yang ditunjuk oleh kejadian. Komponen jaring berkomunikasi dengan salah satu dari berbagai paket yang berlalu, bagaimanapun komponen ini tidak memakan waktu simulasi *actual*.

2.4.1 Konsep Dasar NS2

Pada *Network Simulator* terdapat tampilan atau *display* baik dengan *node* yang bergerak atau *node* yang tidak bergerak. Yang tentunya tidak sama dengan keadaan yang sebenarnya.

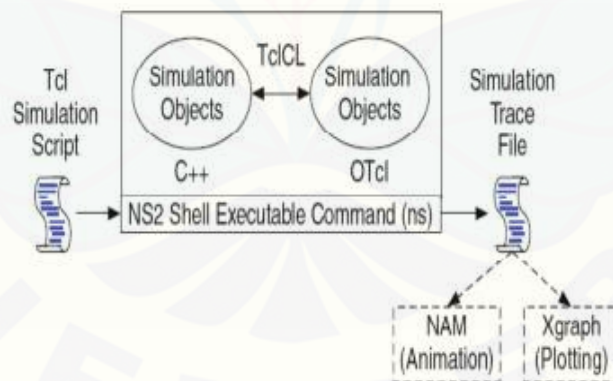
Network Simulator dibangun dengan menggunakan dua bahasa pemrograman, yaitu C++ dan Tcl/Otcl. C++ digunakan untuk *library* yang berisi *event scheduler*. Tcl/Otcl digunakan pada *script* simulasi yang ditulis oleh NS *user* dan pada *library* sebagai *simulator* objek. Otcl juga nantinya berperan sebagai interpreter. Pada gambar 2.8. adalah komponen pembangunan NS.



Gambar 2.8. Komponen Pembangunan NS

(Sumber : Aji Sasongko *at al.* 2010)

Untuk menampilkan hasil simulasi dalam bentuk grafik maupun animasi, biasanya dapat menggunakan *tool* seperti NAM (*Network Animator*) dan Xgraph.



Gambar 2.9. Arsitektur dasar dari NS2

(Sumber : Megawan. 2013)

Bahasa C++ digunakan pada *library* karena C++ mampu mendukung *runtime* simulasi yang cepat, meskipun simulasi melibatkan simulasi jumlah paket dan sumber data dalam jumlah besar.

Bahasa Tcl memberikan respon *runtime* yang lebih lambat daripada C++, namun jika terdapat kesalahan *syntax* dan perubahan *script* berlangsung dengan cepat dan interaktif. *User* dapat mengetahui letak kesalahannya yang dijelaskan pada *console*, sehingga *user* dapat memperbaiki dengan cepat. Karena alasan itulah bahasa ini dipilih untuk digunakan pada skrip simulasi.

2.4.2 Output Simulasi NS2

Pada saat suatu simulasi berakhir, NS membuat satu atau lebih *output* yang berisi detail simulasi jika dideklarasikan pada saat membangun simulasi. Ada tiga jenis *output* NS, yaitu :

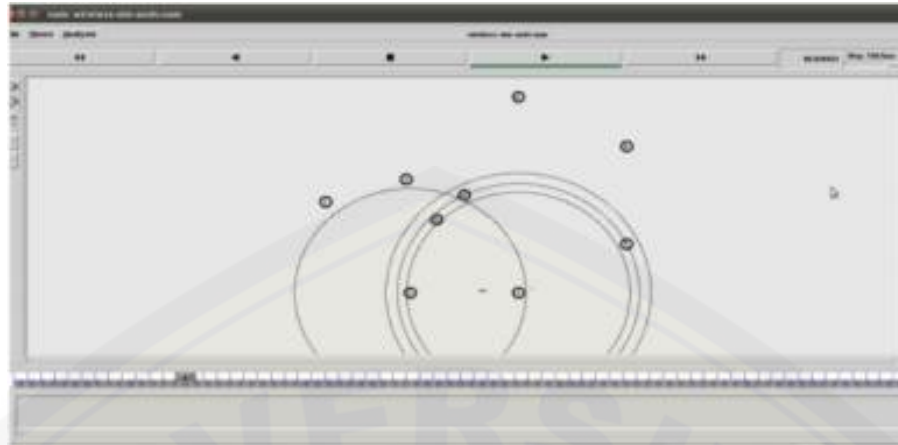
- a. *File trace*, yang digunakan untuk analisa numerik, seperti yang ada pada gambar 2.10.

```
+ 0.110419 1 2 tcp 1040 ----- 2 1.0 4.0 5 12
+ 0.110419 1 2 tcp 1040 ----- 2 1.0 4.0 6 13
- 0.110431 1 2 tcp 1040 ----- 2 1.0 4.0 5 12
- 0.110514 1 2 tcp 1040 ----- 2 1.0 4.0 6 13
r 0.11308 0 2 cbr 1000 ----- 1 0.0 3.0 2 8
+ 0.11308 2 3 cbr 1000 ----- 1 0.0 3.0 2 8
- 0.11308 2 3 cbr 1000 ----- 1 0.0 3.0 2 8
r 0.11316 0 2 cbr 1000 ----- 1 0.0 3.0 3 9
+ 0.11316 2 3 cbr 1000 ----- 1 0.0 3.0 3 9
- 0.113228 2 3 cbr 1000 ----- 1 0.0 3.0 3 9
r 0.115228 2 3 cbr 1000 ----- 1 0.0 3.0 0 6
r 0.115348 1 2 tcp 1040 ----- 2 1.0 4.0 3 10
+ 0.115348 2 4 tcp 1040 ----- 2 1.0 4.0 3 10
- 0.115348 2 4 tcp 1040 ----- 2 1.0 4.0 3 10
r 0.115376 2 3 cbr 1000 ----- 1 0.0 3.0 1 7
r 0.115431 1 2 tcp 1040 ----- 2 1.0 4.0 4 11
```

Gambar 2.10. Tampilan *File Trace* pada NS2

(Sumber : Teerawat *at al*, 2009:31)

- b. *File namtrace*, yang digunakan sebagai *input* tampilan grafis simulasi, seperti pada gambar 2.11.



Gambar 2.11. Tampilan *File namtrace* pada NS2
(Sumber : Kembuan at al 2012)

- c. Xgraph, yang digunakan untuk menampilkan analisa dalam bentuk grafik, seperti yang ada pada Gambar 2.12.



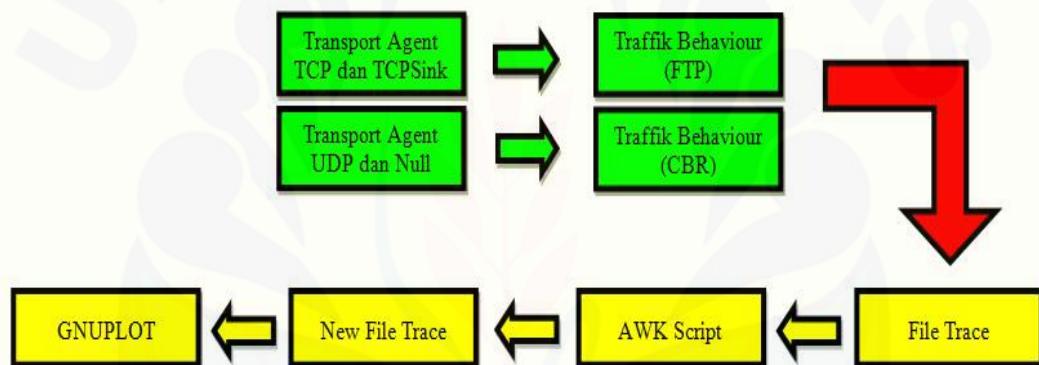
Gambar 2.12. Tampilan Xgraph pada NS2
(Sumber : Nasrul Haqqi. 2013)

BAB 3. METODOLOGI PENELITIAN

Pada bab tiga ini akan dijelaskan mengenai beberapa metodologi peneliti dalam melakukan penelitian ini agar nantinya tidak terjadi beberapa kerancuan dalam melakukan penelitian.

3.1 Blok Diagram Sistem

Pada penelitian ini dilakukan pada teknologi MANET (*Mobile Ad hoc Network*) yang pada simulasi di *Network Simulator 2* (NS 2) memiliki suatu tahapan untuk membuat sebuah *script* yang harus dibuat agar dapat menjalankan simulasi.

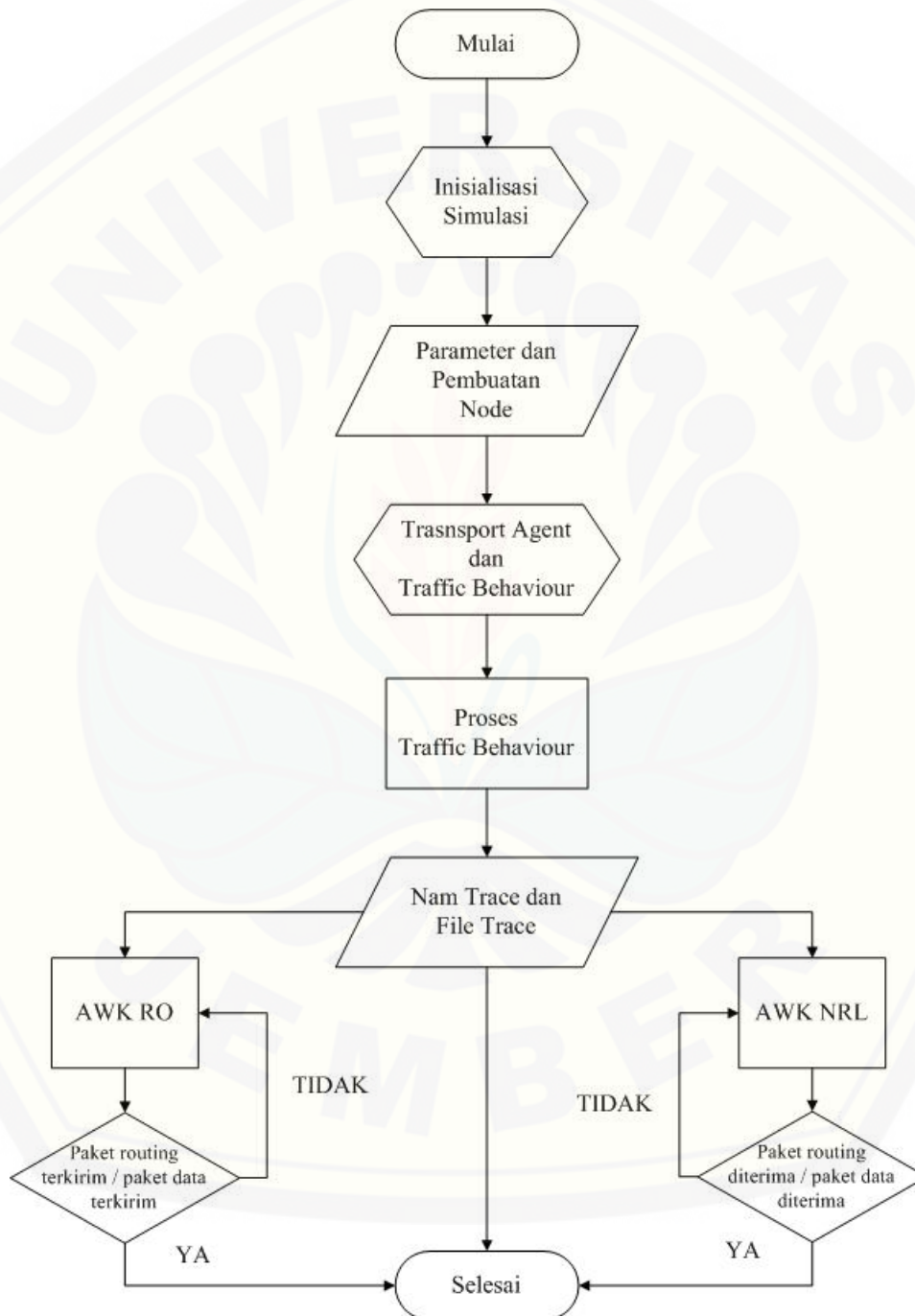


Gambar 3.1. Blok Diagram Sistem

Penentuan *transport* agent yang digunakan adalah TCP dan TCPSink untuk traffic behaviour jenis FTP, serta UDP dan Null untuk *traffic behaviour* jenis CBR, untuk keluaran dari hasil simulasi adalah berupa *file trace* yang berupa catatan kejadian setiap waktu saat simulasi, setelah itu hasil dari *file trace* tersebut di saring dengan *file awk* yang berisi perhitungan untuk NRL dan RO. Dari hasil NRL dan RO tersebut menghasilkan *file trace* baru dan akan dirubah menjadi suatu grafik GNUPLOT dari kumpulan hasil yang didapatkan.

3.2 Flowchart Simulasi MANET

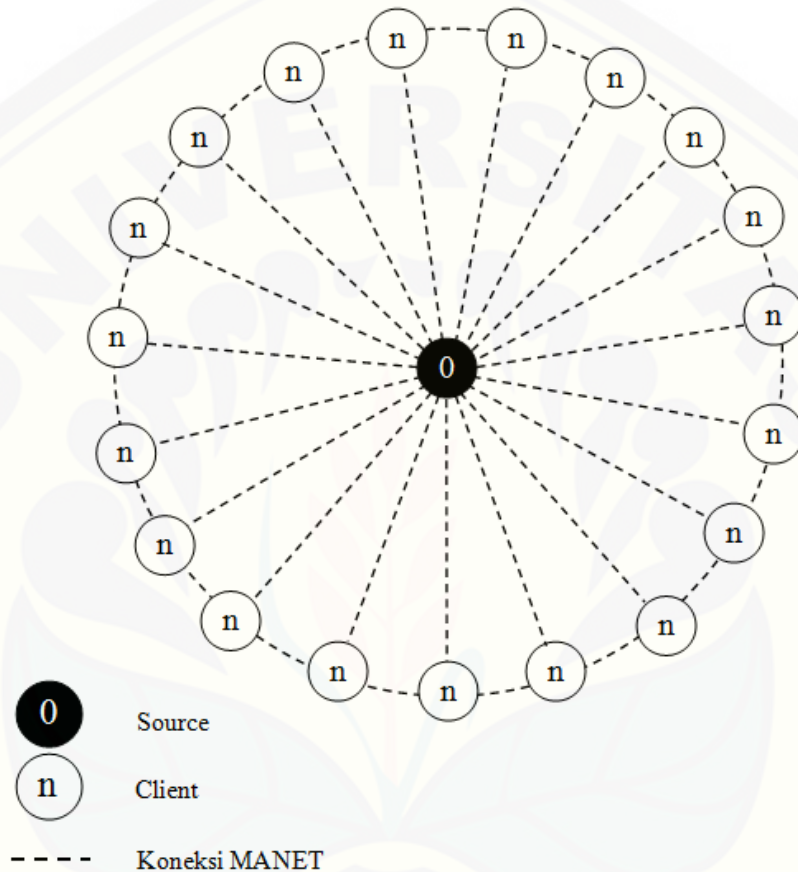
Program simulasi jaringan ini terbagi menjadi beberapa tahap utama, yaitu pengaturan parameter untuk simulasi, inialisasi pembuatan *node* dan pengaturan parameter *node*, pembuatan aliran trafik, akhiri program.



Gambar 3.2. Flowchart Simulasi MANET

3.3 Pemodelan Jaringan MANET di NS2

Adapun rancangan jaringan MANET yang digunakan sebagai objek penelitian ini dalam upaya mengetahui kinerja *traffic behaviour* yang melalui beberapa algoritma protokol. Berikut gambaran umum dari jaringan yang dibentuk.



Gambar 3.3. Rancangan Jaringan MANET di NS2

Angka 0 dengan *background* hitam lingkaran digunakan sebagai *source node*, dan huruf n dengan lingkaran putih digunakan sebagai *client*. Didalam jaringan MANET, *coverage* tidak dibatasi oleh ketersediaan koneksi secara langsung ke *source node* maupun ke *client* lainnya. Disetiap *node user* di MANET tidak hanya beroperasi sebagai *host*, namun juga sebagai *router* untuk meneruskan (*forward*) paket untuk *node* lain yang membutuhkan paket data. Dan pada masing-masing *node* akan berpindah atau bergerak sementara ke tempat tujuan *node* tersebut.

3.4 Parameter-Parameter Simulasi

Parameter-parameter jaringan dibawah ini bersifat konstan dan akan digunakan terus pada setiap pengujian yang dilakukan. Parameter-parameter jaringan yang dimaksud dapat dilihat pada Tabel 3.2.

Tabel 3.2. Parameter-parameter simulasi

Parameter	Value
Simulator	Ns.2.31
Type MAC	Mac/802.11
Model Antena	Omni Antena
Model Propagasi	Two Ray Ground
Network Interface Type	Phy/WirelessPhy
Simulation Time	100 detik
Simulasi Area	X : 1500 dan Y : 1000
Routing Protocol	AODV/DSR/DSDV
Trafik	FTP dan CBR
Maximum Nodes	50 Node
Ukuran Node	40
Maximum Kecepatan Node	20 m/s

3.6.1 Pendefinisian Parameter-Parameter Simulasi

Pendefinisian parameter-parameter simulasi ini bertujuan untuk mendefinisikan nilai dari parameter-parameter yang akan digunakan sekaligus untuk memodelkan karakteristik jaringan yang digunakan pada simulasi. Parameter-parameter tersebut yaitu :

a. Simulator Jaringan

Simulator jaringan yang digunakan pada simulasi adalah *Network Simulator 2* dengan tipe NS2.31.

b. Tipe MAC

Pada simulasi ini tipe MAC yang digunakan adalah Mac/802.11 yang merupakan *mac layer* untuk suatu jaringan *wireless*.

c. Model Antena

Tipe antenna yang digunakan pada simulasi ini adalah *Omniantenna* atau dalam NS2 adalah *Antenna/OmniAntenna* yang merupakan tipe *antenna omni directional*.

d. Model Propagasi

Model propagasi pada simulasi ini adalah *Propagation/TwoRayGround* yang digunakan untuk mensimulasikan jaringan 802.11 pada lingkungan tertutup (jarak transmisi sekitar 25 m).

e. Network Interface Type

Model *network interface type* yang digunakan pada simulasi ini adalah *Phy/WirelessPhy* yang merupakan tipe dari *interface* jaringan.

f. Simulation Time

Waktu simulasi yang digunakan adalah 100 detik.

g. Simulasi Area

Area simulasi yang digunakan adalah 1500 m x 1000 m.

h. Routing Protocol

Routing Protocol yang digunakan pada simulasi ini adalah *Ad hoc On demand Distance Vector* (AODV), *Dynamic Source Routing* (DSR), dan *Destination Sequenced Distance Vector*.

i. Trafik

Jenis trafik yang digunakan pada saat pembuatan aliran trafik adalah *File Transfer Protocol* (FTP) dan *Constant Bit Rate* (CBR).

j. Maximum Nodes

Jumlah maksimum *node* yang digunakan di simulasi ini adalah 50 *node*, dengan pembagian skenario 20 *node*, 30 *node*, dan 50 *node*.

3.5 Skenario Penelitian

Jenis trafik yang digunakan pada simulasi ini menggunakan *File Transfer Protocol* (FTP) dan *Constant Bit Rate* (CBR). Trafik ini digunakan untuk mengetahui jenis trafik apa yang lebih bagus untuk jaringan MANET. Di bawah ini beberapa skenario yang digunakan.

3.5.1 Skenario 1 Perubahan Jumlah *Node*

Dalam skenario ini dicoba untuk melihat bagaimana perubahan parameter berdasarkan pada jumlah *node* yang digunakan yaitu terdiri dari 20 *node* yang mewakili sebagai jumlah *node* sedikit, 30 *node* yang mewakili sebagai jumlah *node* sedang, dan 50 yang mewakili jumlah *node* banyak. Dalam skenario ini pergerakan *node* yang digunakan adalah sebesar 10 m/s dengan luas area masing-masing 1500m x 1000m dan waktu simulasi selama 100 detik.

Tabel 3.3. Skenario 1 parameter berdasarkan perubahan jumlah *node*.

Skenario	Perubahan Jumlah <i>Node</i>	Kecepatan
Skenario 1	20 <i>node</i>	10 m/s
	30 <i>node</i>	
	50 <i>node</i>	

3.5.2 Skenario 2 Perubahan Kecepatan *Node*

Dalam skenario ini dicoba untuk mengetahui bagaimana perubahan parameter berdasarkan pada kecepatan yang terdiri dari 5 m/s, 10 m/s, 15 m/s dan 20 m/s. Dalam jumlah *node* sama yaitu 25 *node* dengan luas area 1500m x1000m dan dalam waktu simulasi selama 100 detik.

Tabel 3.4. Skenario 2 parameter berdasarkan perubahan kecepatan *node*

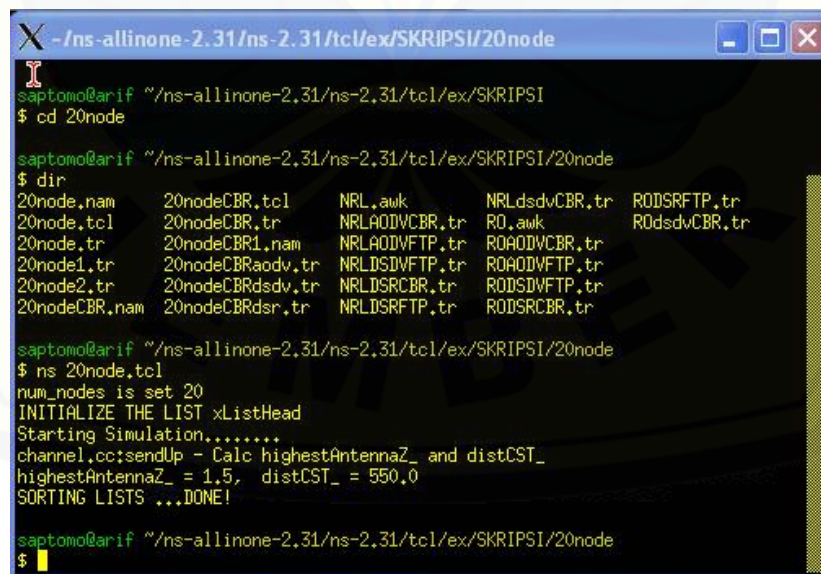
Skenario	Perubahan Kecepatan	Jumlah <i>Node</i>
Skenario 2	5 m/s	25 <i>node</i>
	10 m/s	
	15 m/s	
	20 m/s	

BAB 4. HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas tentang proses analisis yang dilakukan. Hasil simulasi dari program ini akan membahas kinerja dari *traffic behaviour* pada MANET, dalam hal ini jenis *traffic behaviour* yang digunakan adalah FTP (*File Transfer Protocol*) dan CBR (*Constant Bit Rate*). Kinerja dari *traffic behaviour* yang akan dianalisis berupa *Routing Overhead* (RO) dan *Normalized Routing Load* (NRL). Hasil perhitungan kinerja akan diperoleh menggunakan *script awk*.

4.1 Data Keluaran Simulasi

Simulasi MANET pada *traffic behaviour* di NS2 akan menghasilkan keluaran berupa *file namtrace* yang berekstensi “.nam” dan *file trace* yang berekstensi “.tr”, dan *file namtrace* digunakan untuk menginterpretasi hasil *output* secara grafis dan interaktif, digunakan sebuah *tool* seperti NAM (*Network Animator*), dan *file trace* mencatat semua kejadian selama simulasi terjadi, dengan mengolah *file trace* yang ada pada parameter kinerja jaringan akan diperoleh data yang diinginkan. Berikut merupakan tampilan perintah awal untuk menjalankan simulasi, *file namtrace* dan *file trace* yang diperoleh dari hasil simulasi pada skenario pertama dengan *routing protocol* AODV.



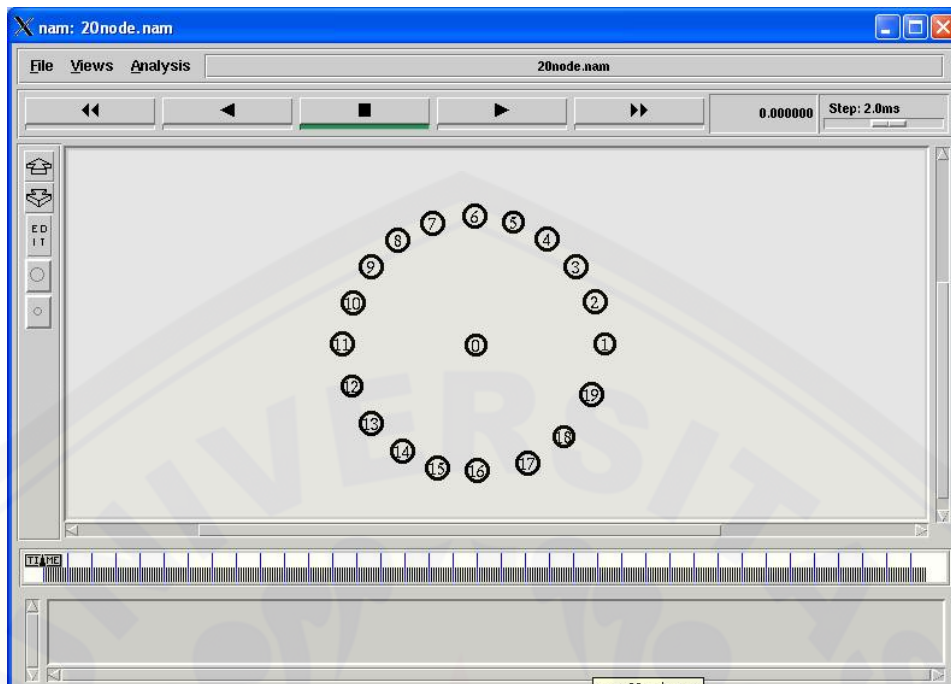
```
X -/ns-allinone-2.31/ns-2.31/tcl/ex/SKRIPSI/20node
saptomo@arif ~/ns-allinone-2.31/ns-2.31/tcl/ex/SKRIPSI
$ cd 20node

saptomo@arif ~/ns-allinone-2.31/ns-2.31/tcl/ex/SKRIPSI/20node
$ dir
20node.nam  20nodeCBR.tcl  NRL.awk  NRLdsvCBR.tr  RODSRFTP.tr
20node.tcl  20nodeCBR.tr  NRLAODVCBR.tr  RO.awk  RODsvCBR.tr
20node.tr  20nodeCBR1.nam  NRLAODVFTP.tr  ROAODVCBR.tr
20node1.tr  20nodeCBRaodv.tr  NRLDSVFTP.tr  ROAODVFTP.tr
20node2.tr  20nodeCBRdsv.tr  NRLSRCBR.tr  RODSVFTP.tr
20nodeCBR.nam  20nodeCBRdsr.tr  NRLSRFTP.tr  RODSRCBR.tr

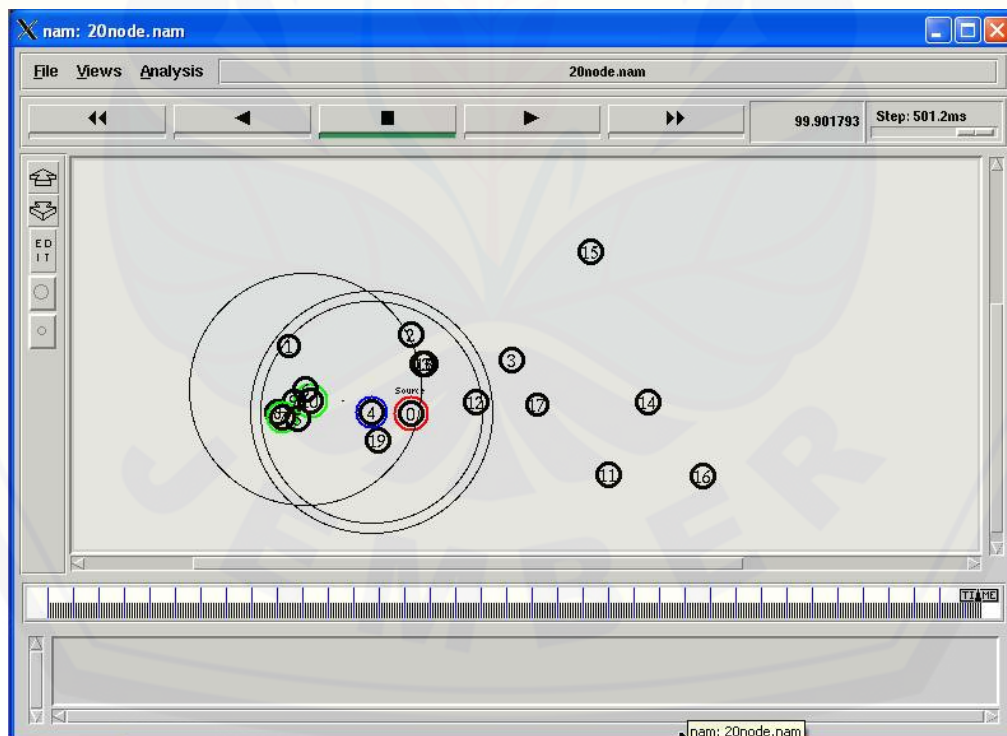
saptomo@arif ~/ns-allinone-2.31/ns-2.31/tcl/ex/SKRIPSI/20node
$ ns 20node.tcl
num_nodes is set 20
INITIALIZE THE LIST xListHead
Starting Simulation.....
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!

saptomo@arif ~/ns-allinone-2.31/ns-2.31/tcl/ex/SKRIPSI/20node
$
```

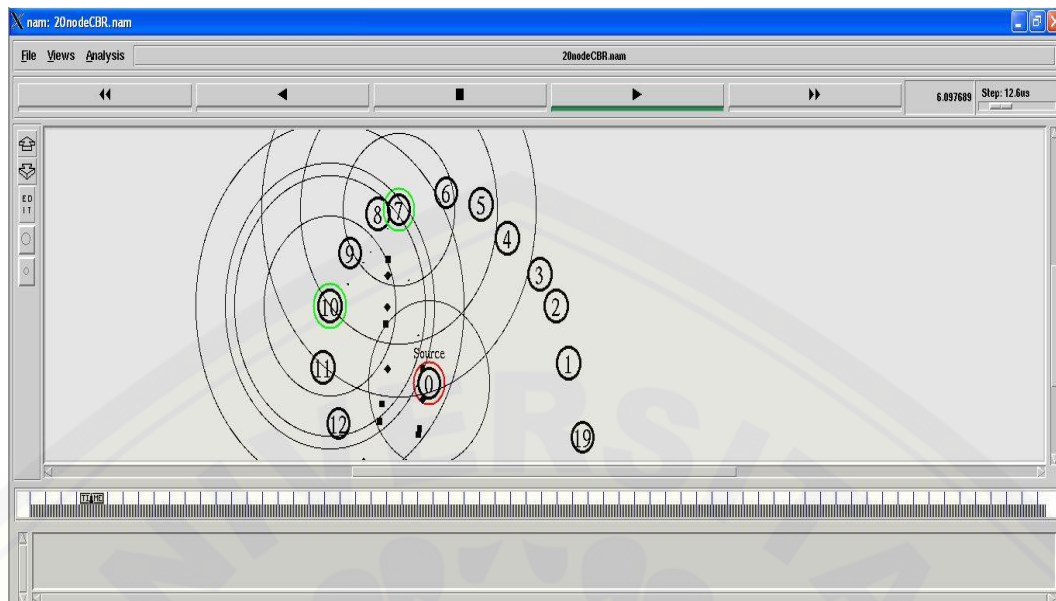
Gambar 4.1. Perintah untuk menjalankan simulasi untuk skenario 1 AODV



Gambar 4.2. Awal *File Namtrace* skenario 1 untuk 20 *node* pada *routing protocol* AODV dengan *traffic behaviour* jenis FTP.



Gambar 4.3. Akhir *File Namtrace* skenario 1 untuk 20 *node* pada *routing protocol* AODV dengan *traffic behaviour* jenis FTP.



Gambar 4.4. *Packet drop* yang terjadi pada *traffic behaviour* jenis CBR

```

r 2.002216491 4 MAC --- 0 ARP 28 [0 ffffffff 7 806] ----- [REQUEST 7/7 0/0]
r 2.002216521 10 MAC --- 0 ARP 28 [0 ffffffff 7 806] ----- [REQUEST 7/7 0/0]
r 2.002216627 0 MAC --- 0 ARP 28 [0 ffffffff 7 806] ----- [REQUEST 7/7 0/0]
s 2.002338893 17 RTR --- 0 AODV 48 [0 ffffffff 0 800] ----- [17:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
s 2.002466491 4 MAC --- 0 AODV 100 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
s 2.002842239 9 RTR --- 0 AODV 48 [0 ffffffff 0 800] ----- [9:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
s 2.002948872 2 RTR --- 0 AODV 48 [0 ffffffff 0 800] ----- [2:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003266708 5 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003266741 3 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003266931 6 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003266964 2 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003267168 7 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003267222 1 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003267284 0 MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003291708 5 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003291741 3 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003291931 6 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003291964 2 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003292168 7 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003292222 1 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
r 2.003292284 0 RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
s 2.003336931 6 MAC --- 0 AODV 100 [0 ffffffff 6 800] ----- [6:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)
s 2.003546781 16 RTR --- 0 AODV 48 [0 ffffffff 0 800] ----- [16:255 -1:255 29 0] [0x2 2 1 [7 0] [0 4]] (REQUEST)

```

Gambar 4.5. *File trace* skenario 1 untuk 20 node pada *routing protocol* AODV dengan *traffic behaviour* jenis FTP.

Dari hasil *file trace* diatas yang ditunjukkan oleh gambar 4.5. dapat diperoleh hasil dari keseluruhan untuk parameter pengujian yang diantaranya *Normalized Routing Load* (NRL) dan *Routing Overhead* (RO) pada skenario satu maupun skenario dua. Kemudian untuk dapat mengetahui besarnya nilai dari *normalized routing load* dan *routing overhead* harus difilter terlebih dahulu dengan bantuan dari *file awk*, setelah mendapatkan *file awk* tersebut dapat

diketikkan perintah pada cygwin dengan mengetikkan perintah untuk NRL perintah yang harus diketikkan adalah

```
awk -f NRL.awk 20node2.tr > NRL20node2.tr
```

Perintah diatas akan secara otomatis memisahkan hasil *normalized routing load*, sedangkan untuk *routing overhead* perintah yang harus diketikkan adalah

```
awk -f RO.awk 20node2.tr > RO20node2.tr
```

Dan perintah diatas akan secara otomatis akan memisahkan hasil dari *routing overhead* dengan sendirinya pada sebuah *file trace*.

4.2 Analisis Kinerja *Traffic Behaviour* Terhadap Skenario 1 (Perubahan Jumlah *Node*)

Dari *file trace* yang diperoleh dapat dihitung parameter-parameter kinerja *traffic behaviour* yang meliputi *normalized routing load* (NRL) dan *routing overhead* (RO), dimana pada perhitungan dari kedua parameter tersebut didapatkan dari pemisahan *file trace* yang memenuhi dari rumus NRL dan RO melalui *file awk* untuk mendapatkannya dan ditampilkan lagi dalam bentuk *file trace* yang sederhana. Dalam skenario ini jumlah perubahan *node* yang diberikan adalah 20 sebagai mewakili *node* sedikit, 30 sebagai mewakili *node* sedang, dan 50 sebagai mewakili *node* banyak, dengan kecepatan perpindahan *node* 10 m/s.

4.2.1 *Normalized Routing Load* (NRL)

Pada parameter ini dilihat dari sisi *node* tujuan, misalnya pada penelitian ini *node* yang menjadi tujuan adalah *node 7*, *node 10*, *node 4*, dan untuk *node* sumber adalah pada *node 0* sesuai dengan perancangan pemodelan jaringan MANET.

Pada NS2 nilai *normalized routing load* ini dihitung dengan menggunakan formula rumus *awk* dengan nama file *NRL.awk*. Kemudian untuk menghitung besarnya nilai dari *normalized routing load* pada *traffic behaviour* jenis FTP dan

CBR dengan menggunakan *routing protocol* AODV, DSDV, dan DSR dapat mengetikkan perintah pada cygwin seperti dibawah ini.

```
awk -f NRL.awk 20node2.tr > NRLAODVFTP.tr
awk -f NRL.awk 20node1.tr > NRLDSRFTP.tr
awk -f NRL.awk 20node0.tr > NRLDSDVFTP.tr
awk -f NRL.awk 20nodeCBRdsr.tr > NRLDSRCBR.tr
awk -f NRL.awk 20nodeCBRdsdv.tr > NRLdsdvCBR.tr
awk -f NRL.awk 20nodeCBRaodv.tr > NRLAODVCBR.tr
awk -f NRL.awk 20node1.tr > NRLDSDVFTP.tr
awk -f NRL.awk 30nodeAODV.tr > NRLaodvFTP.tr
awk -f NRL.awk 30nodeDSDV.tr > NRLdsdvFTP.tr
awk -f NRL.awk 30nodeDSR.tr > NRLdsrFTP.tr
awk -f NRL.awk 30nodeCBRaodv.tr > NRLaodvCBR.tr
awk -f NRL.awk 30nodeCBRdsr.tr > NRLdsrCBR.tr
awk -f NRL.awk 30nodeCBRdsdv.tr > NRLdsrCBR.tr
awk -f NRL.awk 50nodeAODV.tr > NRLaodv.tr
awk -f NRL.awk 50nodeDSDV.tr > NRLdsdv.tr
awk -f NRL.awk 50nodeDSR.tr > NRLdsr.tr
awk -f NRL.awk 50nodeCBRaodv.tr > NRLaodvCBR.tr
awk -f NRL.awk 50nodeCBRdsr.tr > NRLdsrCBR.tr
awk -f NRL.awk 50nodeCBRdsdv.tr > NRLdsdvCBR.tr
```

Setelah selesai melakukan simulasi di NS2 pada skenario satu yang menguji tentang kinerja *traffic behaviour* terhadap perubahan jumlah *node* didapatkan suatu data yang berbentuk *file* “.tr”, yang didalamnya terdapat suatu

nilai dari masing perubahan *node* pada *traffic behaviour* tersebut mulai dari NRLAODV sampai NRLdsvCBR dari masing-masing jumlah *node*. Seperti pada Tabel 4.1, Gambar 4.1. untuk *traffic behaviour* FTP dan Tabel 4.2, Gambar 4.2 untuk CBR berikut.

Tabel 4.1. Data Rata-rata Nilai NRL dari jumlah *node* terhadap FTP

<i>Traffic Behaviour</i>	<i>Jumlah Node</i>	<i>Normalized Routing Load</i>		
		AODV	DSR	DSDV
FTP	20 Node	0.013	0.006	0.043
	30 Node	0.020	0.020	0.090
	50 Node	0.030	0.012	0.180

Tabel 4.2. Data Rata-rata Nilai NRL dari jumlah *node* terhadap CBR

<i>Traffic Behaviour</i>	<i>Jumlah Node</i>	<i>Normalized Routing Load</i>		
		AODV	DSR	DSDV
CBR	20 Node	0.004	0.002	0.013
	30 Node	0.005	0.002	0.025
	50 Node	0.007	0.015	0.052

Dalam skenario satu yang dilakukan pada parameter *Normalized Routing Load* (NRL) terhadap perubahan jumlah *node*, sebagian *node* yang terletak pada area cakupan MANET aktif melakukan pengiriman data. Pengiriman dilakukan hanya pada *node-node* tertentu yaitu dari *node* 0 atau *node source* menuju ke *node* 7 sebagai *node client*, dari *node* 7 ke *node* 10 dan yang terakhir adalah dari *node* 10 ke *node* 4 secara beruntun dalam waktu 100 detik.

Nilai *normalized routing load* untuk keseluruhan *routing protocol* selalu bertambah seiring dengan jumlah *node* yang semakin banyak yang berada dalam MANET, namun pada *routing protocol* DSR pada *traffic behaviour* jenis FTP mengalami penurunan. Meskipun pada *traffic behaviour* jenis CBR pada *routing protocol* jenis AODV mengalami peningkatan nilai yang cukup kecil pada saat

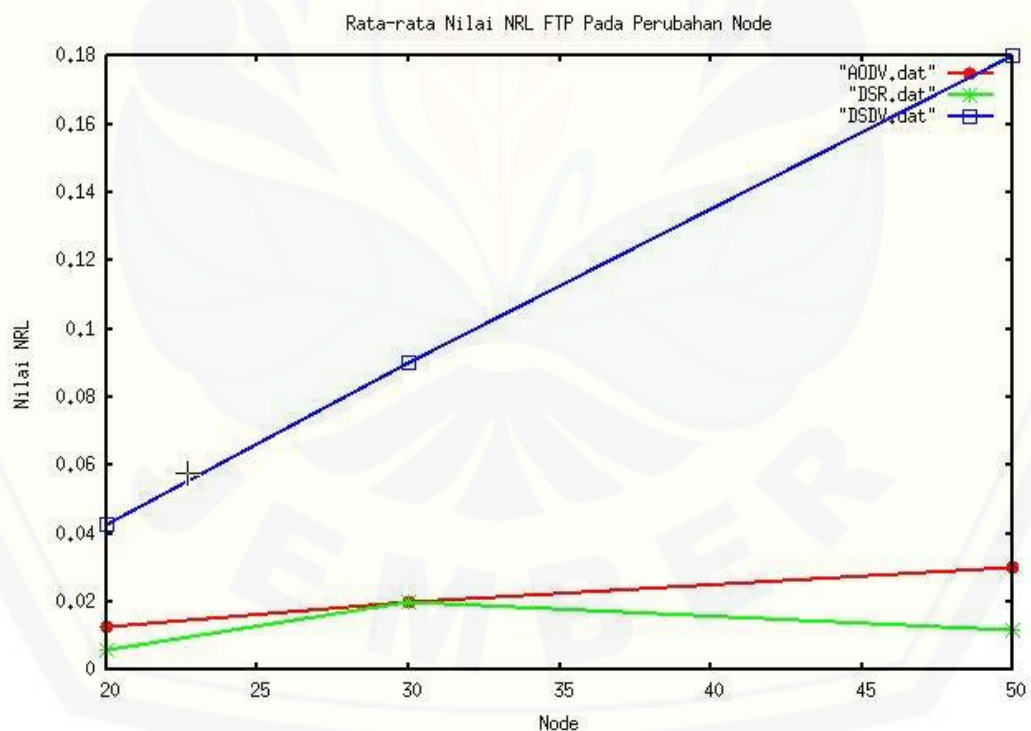
penambahan jumlah *node*, tetapi *routing protocol* yang memiliki nilai NRL terkecil adalah *routing protocol* DSR karena dapat dilihat dari nilai NRL yang dihasilkan baik dari *traffic behaviour* jenis FTP maupun CBR nilai NRL dari DSR yang paling baik yaitu nilai tertingginya adalah 0.015 pada CBR dan pada FTP adalah 0.020.

Untuk perbandingan antara nilai *traffic behaviour* jenis FTP dan CBR yang terbaik untuk di implementasikan pada teknologi MANET untuk efisiensi NRL suatu kinerja *traffic behaviour* adalah pada jenis FTP, karena pada hasil *output file trace* jenis FTP ini sangat minim mengalami paket *drop*. Dengan memiliki nilai terendah saat jumlah *node* maksimum, dimiliki oleh *routing protocol* DSR yaitu 0.012 sedangkan untuk jenis CBR memiliki nilai efisiensi yang cukup bagus untuk NRL yaitu pada 0.015, namun kelemahan pada jenis CBR ini adalah sering mengalami *paket drop* sehingga *traffic behaviour* jenis CBR pada penelitian ini tidak dianjurkan untuk digunakan pada teknologi MANET.

Dalam penabelan dari beberapa *node* yang terdiri dari 20 *node*, 30 *node*, 50 *node*, pengaruh dari perubahan *node* yang mengalami peningkatan nilai NRL dan nilai efisiensi yang paling bagus yaitu DSR, yang kedua adalah AODV, dan yang ketiga DSDV. Kenaikan nilai NRL tersebut dilihat dari kedua tabel 4.1 dan tabel 4.2 diatas menunjukkan bahwa *routing protocol* DSR dengan *traffic behaviour* jenis FTP yang paling bagus digunakan untuk teknologi MANET karena minimnya paket *drop* yang terjadi.

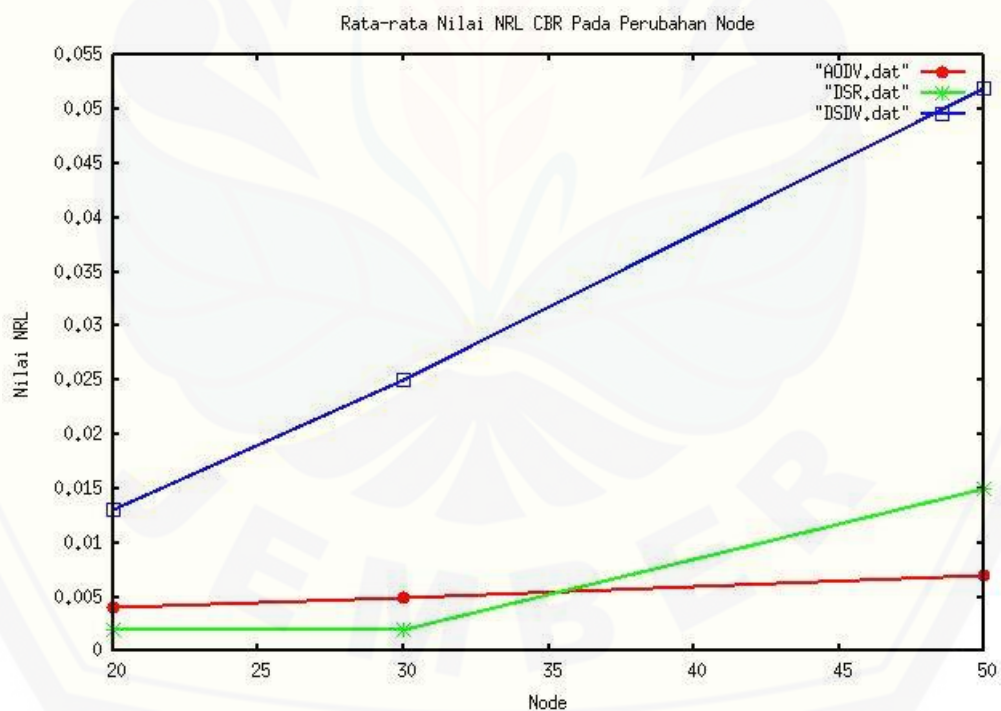
Konsep dari *normalized routing load* (NRL) yaitu semakin tinggi nilai perbandingan banyaknya paket *routing* dengan paket data maka kinerja suatu *protokol routing* semakin kurang efisien. Oleh maka dari itu dalam skenario pertama ini yaitu pada *normalized routing load routing protocol* yang paling efisien pada *node* maksimum adalah pada *routing protocol* DSR dengan nilai 0.012 dengan minim paket *drop* dan *traffic behaviour* yang paling cocok digunakan oleh *routing protocol* ini adalah jenis FTP. Dari data-data dan penjelasan tersebut dapat dilihat melalui grafik *gnuplot*, dengan mengetikkan perintah dibawah ini untuk membuat grafik *gnuplot*.


```
gnuplot  
  
cd 'C:\cygwin\home\saptomo\NRLFTPNODE'  
  
set title "Rata-rata Nilai NRL FTP Pada Perubahan Node"  
  
set xlabel "Node"  
  
set ylabel "Nilai NRL"  
  
set style line 1 lt 1 lw 2 pt 7 ps 2  
set style line 2 lt 2 lw 2 pt 4 ps 2  
set style line 3 lt 3 lw 2 pt 3 ps 2  
  
plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'  
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```



Gambar 4.6. Grafik data rata-rata nilai NRL dari perubahan jumlah *node* terhadap *trafik behaviour* jenis FTP

```
gnuplot
cd 'C:\cygwin\home\saptomo\NRLCBRNODE'
set title "Rata-rata Nilai NRL CBR Pada Perubahan Node"
set xlabel "Node"
set ylabel "Nilai NRL"
set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 4 lw 2 pt 4 ps 2
set style line 3 lt 2 lw 2 pt 3 ps 2
plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```



Gambar 4.7. Grafik data rata-rata nilai NRL dari perubahan jumlah *node* terhadap *traffic behaviour* jenis CBR

Dari grafik tersebut dapat membuktikan data dari tabel 4.1 dan tabel 4.2 bahwa *routing protocol* DSR yang memiliki kinerja yang paling efisien untuk teknologi MANET baik pada *traffic behaviour* jenis FTP namun pada jenis CBR *routing protocol* AODV memiliki nilai efisien yang paling bagus, tetapi kembali lagi pada penjelasan sebelumnya yaitu pada CBR sering terjadi *paket drop* yang terjadi sehingga dapat disimpulkan *traffic behaviour* yang cocok untuk *routing protocol* DSR adalah jenis FTP pada parameter NRL untuk perubahan jumlah *node*.

4.2.2 Routing Overhead

Pada parameter ini dilihat dari sisi *node* sumber, misalnya pada penelitian ini *node* yang menjadi *node* sumber adalah *node* 0, *node* 7, *node* 10. Parameter ini digunakan untuk menghitung efisiensi kinerja suatu *routing protocol*.

Pada NS2 nilai *routing overhead* ini dihitung dengan menggunakan formula rumus awk dengan nama file RO.awk. Kemudian untuk menghitung besarnya nilai dari *routing overhead* pada *traffic behaviour* jenis FTP dan CBR dengan menggunakan *routing protocol* AODV, DSDV, dan DSR dapat menuliskan perintah pada cygwin seperti dibawah ini.

```
awk -f RO.awk 20node2.tr > ROAODVFTP.tr
awk -f RO.awk 20node1.tr > RODSRFTP.tr
awk -f RO.awk 20node0.tr > RODSDVFTP.tr
awk -f RO.awk 20nodeCBRdsr.tr > RODSRCBR.tr
awk -f RO.awk 20nodeCBRdsdv.tr > RODsdvCBR.tr
awk -f RO.awk 20nodeCBRaodv.tr > ROAODVCBR.tr
awk -f RO.awk 20node1.tr > RODSDVFTP.tr
awk -f RO.awk 30nodeAODV.tr > ROaodvFTP.tr
awk -f RO.awk 30nodeDSDV.tr > RODsdvFTP.tr
```

```

awk -f RO.awk 30nodeDSR.tr > ROdsrFTP.tr

awk -f RO.awk 30nodeCBRaodv.tr > ROaodvCBR.tr

awk -f RO.awk 30nodeCBRdsr.tr > ROdsrCBR.tr

awk -f RO.awk 30nodeCBRdsdv.tr > ROdsrCBR.tr

awk -f RO.awk 50nodeAODV.tr > ROaodv.tr

awk -f RO.awk 50nodeDSDV.tr > RODsdv.tr

awk -f RO.awk 50nodeDSR.tr > ROdsr.tr

awk -f RO.awk 50nodeCBRaodv.tr > ROaodvCBR.tr

awk -f RO.awk 50nodeCBRdsr.tr > ROdsrCBR.tr

awk -f RO.awk 50nodeCBRdsdv.tr > RODsdvCBR.tr

```

Setelah selesai melakukan simulasi di NS2 pada skenario satu yang menguji tentang kinerja *traffic behaviour* terhadap perubahan jumlah *node* didapatkan suatu data yang berbentuk *file* “.tr”, yang didalamnya terdapat suatu nilai dari masing perubahan *node* pada *traffic behaviour* tersebut mulai dari ROAODV sampai RODsdvCBR dari masing-masing jumlah *node*. Seperti pada Tabel 4.3, Gambar 4.3. untuk *traffic behaviour* FTP dan Tabel 4.4, Gambar 4.4 untuk CBR berikut.

Tabel 4.3. Data Rata-rata Nilai RO dari jumlah *node* terhadap FTP

<i>Traffic Behaviour</i>	<i>Jumlah Node</i>	<i>Routing Overhead</i>		
		AODV	DSR	DSDV
FTP	20 Node	0.9936	0.9952	0.9949
	30 Node	0.9937	0.9937	0.9927
	50 Node	0.9930	0.9950	0.9956

Tabel 4.4. Data Rata-rata Nilai RO dari jumlah *node* terhadap CBR

<i>Traffic Behaviour</i>	Jumlah Node	<i>Routing Overhead</i>		
		AODV	DSR	DSDV
CBR	20 Node	0.2879	0.2858	0.3395
	30 Node	0.2845	0.3176	0.3525
	50 Node	0.2838	0.9783	0.3332

Dalam skenario satu yang dilakukan pada parameter *Routing Overhead* (RO) terhadap perubahan jumlah *node*, sebagian *node* yang terletak pada area cakupan MANET aktif melakukan pengiriman data. Pengiriman dilakukan hanya pada *node-node* tertentu yaitu dari *node* 0 atau *node source* menuju ke *node* 7 sebagai *node client*, dari *node* 7 ke *node* 10 dan yang terakhir adalah dari *node* 10 ke *node* 4 secara beruntun dalam waktu 100 detik.

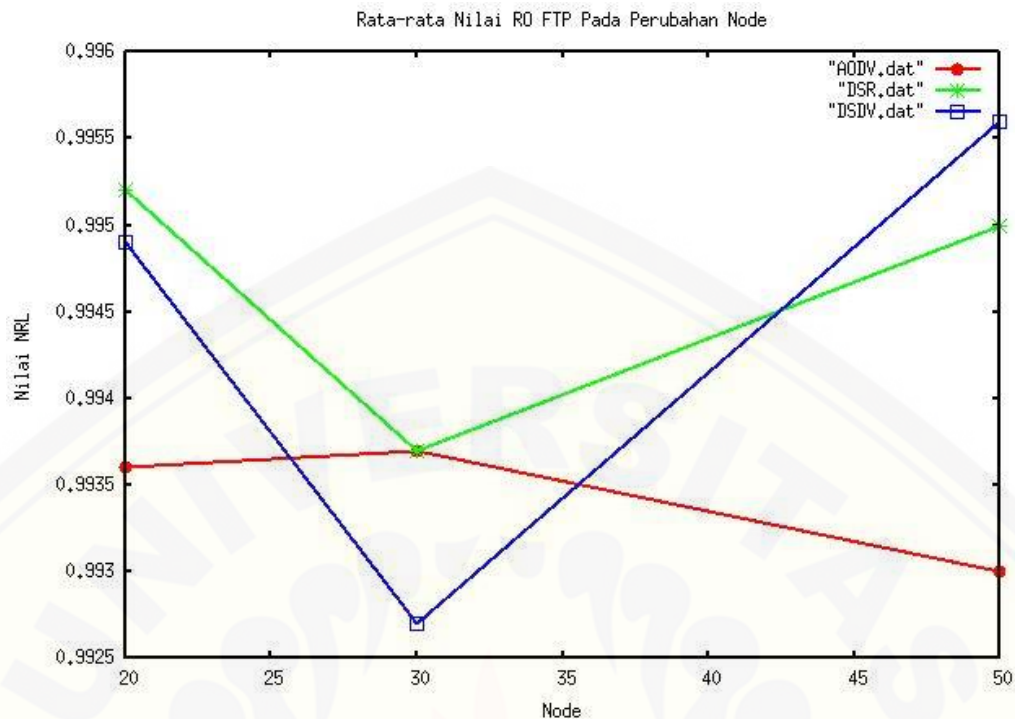
Nilai *routing overhead* untuk ketiga *routing protocol* selalu mengalami penambahan dan juga penurunan pada masing-masing *routing protocol* baik dari *traffic behaviour* jenis FTP maupun CBR. Namun tidak jauh dari parameter NRL bahwa pada jenis CBR sering mengalami paket *drop* sehingga *traffic behaviour* jenis FTP yang paling cocok untuk teknologi MANET meskipun memiliki nilai yang lebih besar dari CBR.

Untuk perbandingan antara nilai *traffic behaviour* jenis FTP dan CBR yang terbaik untuk diimplementasikan pada teknologi MANET untuk efisiensi RO suatu kinerja *traffic behaviour* adalah pada jenis FTP, karena pada hasil *output file trace* jenis FTP ini sangat minim mengalami *paket drop*. Dengan memiliki nilai terendah dimiliki oleh *routing protocol* AODV yaitu 0.9930 sedangkan untuk jenis CBR memiliki nilai efisiensi yang paling rendah untuk RO yaitu pada 0.2838 pada *routing protocol* AODV, namun kelemahan pada jenis CBR ini adalah sering mengalami *paket drop* sehingga *traffic behaviour* jenis CBR pada penelitian ini tidak dianjurkan untuk digunakan pada teknologi MANET. Jadi *routing protocol* yang paling bagus pada parameter RO adalah DSR pada jenis FTP.

Dalam penabelan dari beberapa *node* yang terdiri dari 20 *node*, 30 *node*, 50 *node*, pengaruh dari perubahan *node* yang mengalami peningkatan nilai RO yang paling bagus yaitu AODV, yang kedua adalah DSR, dan yang ketiga DSDV. Kenaikan nilai RO tersebut dilihat dari kedua tabel 4.3 dan tabel 4.4 diatas menunjukkan bahwa *routing protocol* AODV dengan *traffic behaviour* jenis FTP yang paling bagus digunakan untuk teknologi MANET karena minimnya paket *drop* yang terjadi.

Konsep dari *Routing Overhead* (RO) yaitu semakin kecil nilai *routing overhead* suatu *protocol routing*, maka semakin efisien kinerja *protocol routing* tersebut. Oleh maka dari itu dalam skenario pertama ini yaitu pada *routing overhead*, *routing protocol* yang paling efisien adalah pada *routing protocol* DSR dengan nilai 0.9930 dengan minim paket *drop* dan *traffic behaviour* yang paling cocok digunakan oleh *routing protocol* ini adalah jenis FTP. Dari data-data dan penjelasan tersebut dapat dilihat melalui grafik *gnuplot*, dengan mengetikkan perintah dibawah ini untuk membuat grafik *gnuplot*.

```
gnuplot
cd 'C:\cygwin\home\saptomo\ROFTPNODE'
set title "Rata-rata Nilai RO FTP Pada Perubahan Node"
set xlabel "Node"
set ylabel "Nilai NRL"
set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 2 lw 2 pt 4 ps 2
set style line 3 lt 3 lw 2 pt 3 ps 2
plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```

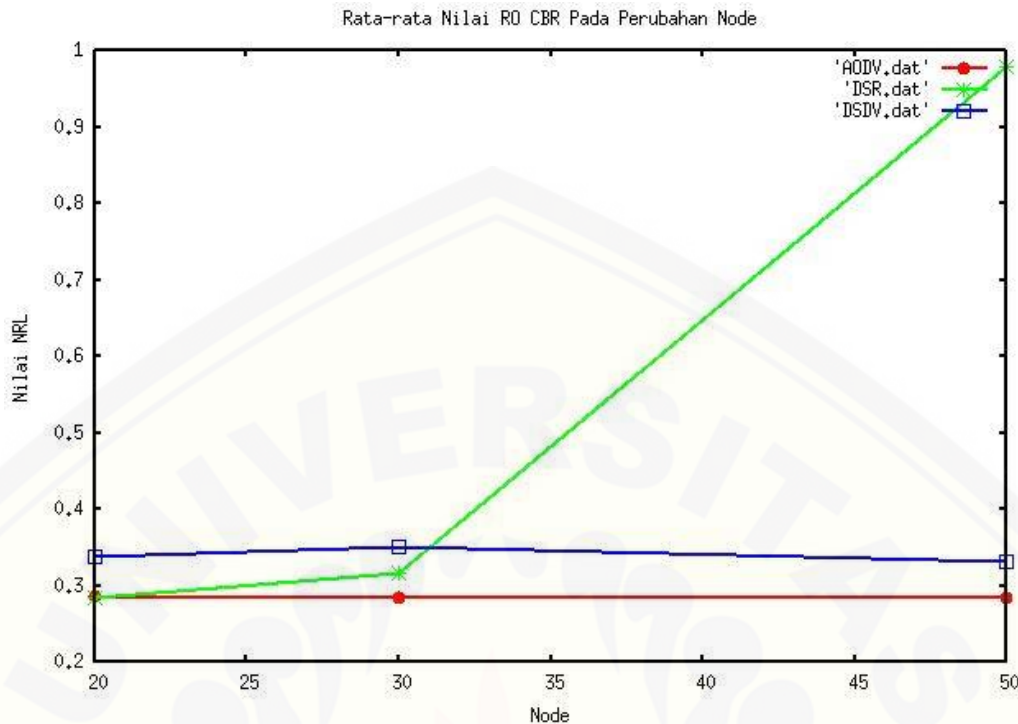


Gambar 4.8. Grafik data rata-rata nilai RO dari perubahan jumlah *node* terhadap *traffic behaviour* jenis FTP

```

gnuplot
cd 'C:\cygwin\home\saptomo\ROCBRNODE'
set title "Rata-rata Nilai RO CBR Pada Perubahan Node"
set xlabel "Node"
set ylabel "Nilai NRL"
set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 4 lw 2 pt 4 ps 2
set style line 3 lt 2 lw 2 pt 3 ps 2
plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3

```



Gambar 4.9. Grafik data rata-rata nilai RO dari perubahan jumlah *node* terhadap *traffic behaviour* jenis CBR

Dari grafik tersebut dapat membuktikan data dari tabel 4.1 dan tabel 4.2 bahwa *routing* AODV yang memiliki kinerja yang paling efisien untuk teknologi MANET baik pada *traffic behaviour* jenis FTP, namun pada jenis CBR *routing protocol* AODV yang paling baik. Namun kembali lagi pada penjelasan sebelumnya yaitu pada CBR sering terjadi paket *drop* yang terjadi sehingga dapat disimpulkan *traffic behaviour* yang cocok untuk *routing protocol* AODV adalah jenis FTP pada parameter RO untuk perubahan jumlah *node*.

4.3 Analisis Performansi *Traffic Behaviour* Terhadap Skenario 2 (Perubahan Kecepatan *Node*)

Dari *file trace* yang diperoleh dapat dihitung parameter-parameter kinerja *traffic behaviour* yang meliputi *normalized routing load* (NRL) dan *routing overhead* (RO), dimana pada perhitungan dari kedua parameter tersebut didapatkan dari pemisahan *file trace* yang memenuhi dari rumus NRL dan RO

melalui file awk untuk mendapatkannya dan ditampilkan lagi dalam bentuk *file trace* yang sederhana. Dalam skenario kedua ini menunjukkan hasil analisis dari kinerja *traffic behaviour* berdasarkan perubahan kecepatan *node*, perubahan kecepatan itu terdiri dari 5 m/s, 10 m/s, 15 m/s, dan 20 m/s dengan total jumlah *node* adalah 25 *node*.

4.3.1 Normalized Routing Load (NRL)

Pada skenario kedua tidak jauh dari skenario satu untuk parameter-parameter yang diukur dalam penelitian ini yaitu untuk yang pertama adalah *normalized routing load* (NRL), Pada parameter ini dilihat dari sisi *node* tujuan, misalnya pada penelitian ini *node* yang menjadi tujuan adalah *node* 7, *node* 10, *node* 4, dan untuk *node* sumber adalah pada *node* 0 sesuai dengan perancangan pemodelan jaringan MANET.

Pada NS2 nilai *normalized routing load* ini dihitung dengan menggunakan formula rumus awk dengan nama *file* NRL.awk. Kemudian untuk menghitung besarnya nilai dari *normalized routing load* pada *traffic behaviour* jenis FTP dan CBR dengan menggunakan *routing protocol* AODV, DSDV, dan DSR dapat menuliskan perintah pada cygwin seperti dibawah ini.

```
awk -f NRL.awk 5msaodv.tr > NRLdsdv.tr
awk -f NRL.awk 5msdsdv.tr > NRLdsdv.tr
awk -f NRL.awk 5msdsr.tr > NRLdsr.tr
awk -f NRL.awk 5msCBRdsr.tr > NRLdsrCBR.tr
awk -f NRL.awk 5msCBRaodv.tr > NRLaodvCBR.tr
awk -f NRL.awk 5msCBRdsdv.tr > NRLdsdvCBR.tr
awk -f NRL.awk 10msaodv.tr > NRLaodv.tr
awk -f NRL.awk 10msdsdv.tr > NRLdsdv.tr
awk -f NRL.awk 10msdsr.tr > NRLdsr.tr
```

```
awk -f NRL.awk 10msCBRdsdv.tr > NRLdsdvCBR.tr  
awk -f NRL.awk 10msCBRaodv.tr > NRLaodvCBR.tr  
awk -f NRL.awk 10msCBRdsr.tr > NRLdsrCBR.tr  
awk -f NRL.awk 15msaodv.tr > NRLaodv.tr  
awk -f NRL.awk 15msdsdv.tr > NRLdsdv.tr  
awk -f NRL.awk 15msdsr.tr > NRLdsr.tr  
awk -f NRL.awk 15msCBRaodv.tr > NRLaodvCBR.tr  
awk -f NRL.awk 15msCBRdsr.tr > NRLdsrCBR.tr  
awk -f NRL.awk 15msCBRdsdv.tr > NRLdsdvCBR.tr  
awk -f NRL.awk 20msaodv.tr > NRLaodv.tr  
awk -f NRL.awk 29msdsdv.tr > NRLdsdv.tr  
awk -f NRL.awk 20msdsr.tr > NRLdsr.tr  
awk -f NRL.awk 20msCBRaodv.tr > NRLaodvCBR.tr  
awk -f NRL.awk 20msCBRdsr.tr > NRLdsrCBR.tr  
awk -f NRL.awk 20msCBRdsdv.tr > NRLdsdvCBR.tr
```

Setelah selesai melakukan simulasi di NS2 pada skenario dua yang menguji tentang kinerja *traffic behaviour* terhadap perubahan kecepatan *node* didapatkan suatu data yang berbentuk *file* “.tr”, yang didalamnya terdapat suatu nilai dari masing perubahan kecepatan *node* pada *traffic behaviour* tersebut mulai dari NRLaodv sampai NRLdsdvCBR dari masing-masing jumlah *node*. Seperti pada Tabel 4.5, Gambar 4.5. untuk *traffic behaviour* FTP dan Tabel 4.6, Gambar 4.6 untuk CBR berikut.

Tabel 4.5. Data Rata-rata Nilai NRL dari kecepatan *node* terhadap FTP

<i>Traffic Behaviour</i>	Kecepatan	<i>Normalizel Routing Load</i>		
		AODV	DSR	DSDV
FTP	5 m/s	0.011	0.010	0.059
	10 m/s	0.012	0.029	0.056
	15 m/s	0.016	0.008	0.052
	20 m/s	0.009	0.005	0.055

Tabel 4.6. Data Rata-rata Nilai NRL dari kecepatan *node* terhadap CBR

<i>Traffic Behaviour</i>	Kecepatan	<i>Normalized Routing Load</i>		
		AODV	DSR	DSDV
CBR	5 m/s	0.006	0.015	0.018
	10 m/s	0.005	0.015	0.016
	15 m/s	0.003	0.015	0.018
	20 m/s	0.004	0.015	0.016

Dalam skenario kedua yang dilakukan pada parameter *Normalized Routing Load* (NRL) terhadap perubahan kecepatan *node*, sebagian *node* yang terletak pada area cakupan MANET aktif melakukan pengiriman data. Pengiriman dilakukan hanya pada *node-node* tertentu yaitu dari *node* 0 atau *node source* menuju ke *node* 7 sebagai *node client*, dari *node* 7 ke *node* 10 dan yang terakhir adalah dari *node* 10 ke *node* 4 secara beruntun dalam waktu 100 detik.

Nilai *normalized routing load* untuk ketiga *routing protocol* memiliki nilai NRL yang *random* terkadang naik dan juga turun, namun pada *routing protocol* DSR pada *traffic behaviour* jenis CBR mengalami nilai yang tetap yaitu 0.015. Meskipun pada *traffic behaviour* jenis CBR pada *routing protocol* jenis DSR tidak sama sekali mengalami perubahan nilai pada saat perubahan kecepatan *node*, *routing protocol* yang paling memiliki nilai NRL terkecil adalah *routing protocol* AODV karena dapat dilihat dari nilai NRL yang dihasilkan baik dari *traffic*

behaviour jenis FTP maupun CBR nilai NRL dari AODV yang paling baik yaitu nilai tertingginya adalah 0.004 pada CBR dan pada FTP dimiliki oleh *routing protocol* DSR yaitu dengan nilai tertinggi 0.005.

Untuk perbandingan antara nilai *traffic behaviour* jenis FTP dan CBR yang terbaik untuk diimplementasikan pada teknologi MANET untuk efisiensi NRL suatu kinerja *traffic behaviour* terhadap perubahan kecepatan *node* adalah pada jenis FTP, karena pada hasil *output file trace* jenis FTP ini sangat minim mengalami paket *drop*. Dengan memiliki nilai terendah dimiliki oleh *routing protocol* DSR yaitu 0.005 sedangkan untuk jenis CBR memiliki nilai efisiensi yang cukup bagus untuk NRL yaitu tidak mengalami perubahan nilai meskipun kecepatan *node* selalu berubah dengan nilai 0.015, namun kelemahan pada jenis CBR ini adalah sering mengalami paket *drop* sehingga *traffic behaviour* jenis CBR pada penelitian ini tidak dianjurkan untuk digunakan pada teknologi MANET.

Dalam penabelan dari beberapa perubahan kecepatan *node* yang terdiri dari 5 m/s, 10 m/s, 15 m/s, dan 20 m/s, pengaruh dari perubahan kecepatan *node* yang menunjukkan nilai NRL yang paling bagus yaitu DSR, yang kedua adalah AODV, dan yang ketiga DSDV. Nilai NRL tersebut dilihat dari kedua tabel 4.5 dan tabel 4.6 diatas menunjukkan bahwa *routing protocol* DSR dengan *traffic behaviour* jenis FTP yang paling bagus digunakan untuk teknologi MANET karena minimnya paket *drop* yang terjadi.

Konsep dari *normalized routing load* (NRL) sama dengan skenario satu yaitu semakin tinggi nilai perbandingan banyaknya paket *routing* dengan paket data maka kinerja suatu *protokol routing* semakin kurang efisien. Oleh maka dari itu dalam skenario kedua ini yaitu pada *normalized routing load routing protocol* yang paling efisien adalah pada *routing protocol* DSR dengan nilai 0.005 dengan minim paket *drop* dan *traffic behaviour* yang paling cocok digunakan oleh *routing protocol* ini adalah jenis FTP. Dari data-data dan penjelasan tersebut dapat dilihat melalui grafik *gnuplot*, dengan mengetikkan perintah dibawah ini untuk membuat grafik *gnuplot*.

```
gnuplot

cd 'C:\cygwin\home\saptomo\NRLFTPS'

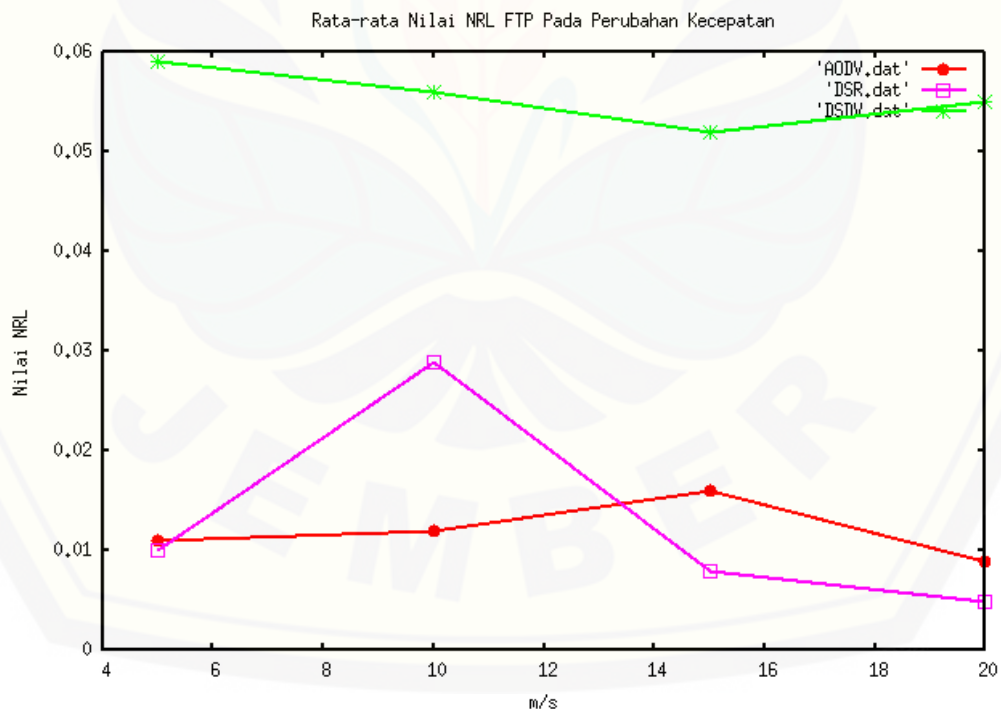
set title "Rata-rata Nilai NRL FTP Pada Perubahan
kecepatan"

set xlabel "m/s"

set ylabel "Nilai NRL"

set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 4 lw 2 pt 4 ps 2
set style line 3 lt 2 lw 2 pt 3 ps 2

plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```



Gambar 4.10. Grafik data rata-rata nilai NRL dari perubahan kecepatan terhadap *traffic behaviour* jenis FTP

```
gnuplot

cd 'C:\cygwin\home\saptomo\NRLCBRPS'

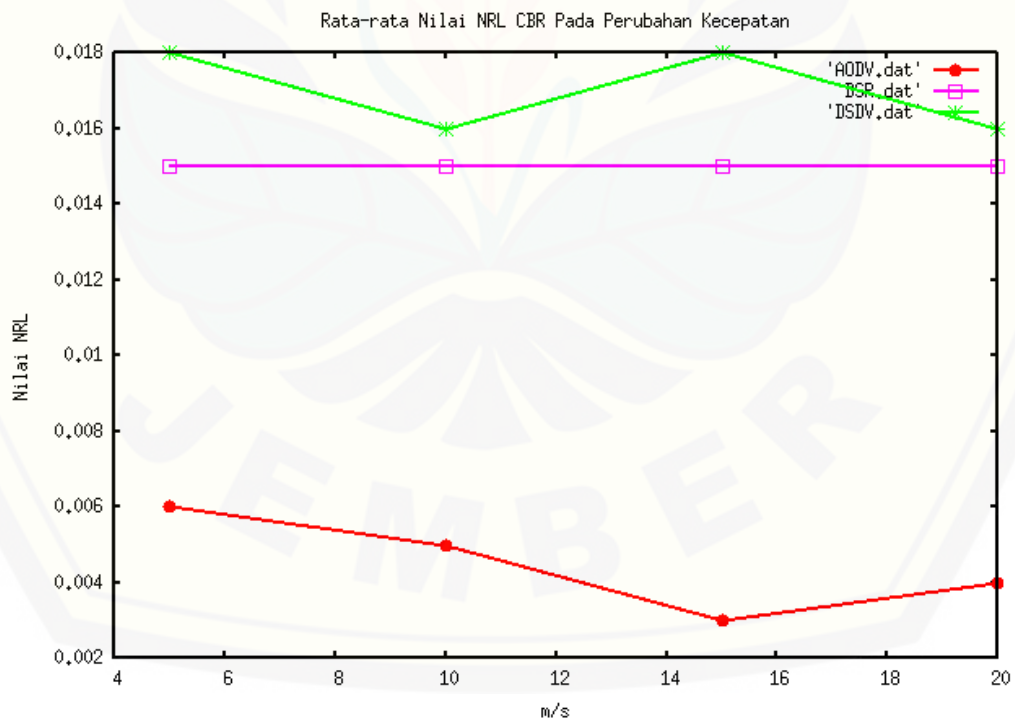
set title "Rata-rata Nilai NRL CBR Pada Perubahan
kecepatan"

set xlabel "m/s"

set ylabel "Nilai NRL"

set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 4 lw 2 pt 4 ps 2
set style line 3 lt 2 lw 2 pt 3 ps 2

plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```



Gambar 4.11. Grafik data rata-rata nilai NRL dari perubahan kecepatan terhadap *traffic behaviour* jenis CBR

Dari grafik tersebut dapat membuktikan data dari tabel 4.5 dan tabel 4.6 bahwa *routing protocol* DSR yang memiliki kinerja yang paling efisien untuk teknologi MANET baik pada *traffic behaviour* jenis FTP, namun pada jenis CBR *routing protocol* AODV yang paling baik. Namun kembali lagi pada penjelasan sebelumnya yaitu pada CBR sering terjadi paket *drop* yang terjadi sehingga dapat disimpulkan *traffic behaviour* yang cocok untuk *routing protocol* DSR adalah jenis FTP pada parameter NRL untuk perubahan kecepatan *node*.

Pada grafik tersebut *routing protocol* DSR mengalami kenaikan nilai NRL secara signifikan, namun pada perubahan kecepatan *node* selanjutnya menunjukkan penurunan sehingga *routing protocol* DSR ini dapat dibuktikan bahwa konsep dari NRL tersebut sangat memenuhi, menunjukkan nilai dari *routing protocol* AODV yang memiliki nilai *Normalized Routing Protocol* yang lebih baik daripada *routing protocol* DSR dan DSDV.

4.3.2 Routing Overhead

Pada skenario kedua tidak jauh dari skenario satu untuk parameter-paramet yang diukur dalam penelitian ini yaitu untuk yang kedua adalah *routing overhead* (RO), Pada parameter ini dilihat dari sisi *node* sumber, misalnya pada penelitian ini *node* yang menjadi sumber adalah *node 0*, *node 7*, *node 10*, sesuai dengan perancangan pemodelan jaringan MANET.

Pada NS2 nilai *routing overhead* ini dihitung dengan menggunakan formula rumus awk dengan nama *file* RO.awk. Kemudian untuk menghitung besarnya nilai dari *routing overhead* pada *traffic behaviour* jenis FTP dan CBR dengan menggunakan *routing protocol* AODV, DSDV, dan DSR dapat menuliskan perintah pada cygwin seperti dibawah ini.

```
awk -f RO.awk 5msaodv.tr > ROdsv.tr
```

```
awk -f RO.awk 5msdsv.tr > ROdsv.tr
```

```
awk -f RO.awk 5msdsr.tr > ROdsr.tr
```

```
awk -f RO.awk 5msCBRdsr.tr > ROdsrCBR.tr
```



```
awk -f RO.awk 5msCBRaadv.tr > ROaadvCBR.tr
awk -f RO.awk 5msCBRdsdv.tr > RODsdvCBR.tr
awk -f RO.awk 10msaadv.tr > ROaadv.tr
awk -f RO.awk 10msdsdv.tr > RODsdv.tr
awk -f RO.awk 10msdsr.tr > RODsr.tr
awk -f RO.awk 10msCBRdsdv.tr > RODsdvCBR.tr
awk -f RO.awk 10msCBRaadv.tr > ROaadvCBR.tr
awk -f RO.awk 10msCBRdsr.tr > RODsrCBR.tr
awk -f RO.awk 15msaadv.tr > ROaadv.tr
awk -f RO.awk 15msdsdv.tr > RODsdv.tr
awk -f RO.awk 15msdsr.tr > RODsr.tr
awk -f RO.awk 15msCBRaadv.tr > ROaadvCBR.tr
awk -f RO.awk 15msCBRdsr.tr > RODsrCBR.tr
awk -f RO.awk 15msCBRdsdv.tr > RODsdvCBR.tr
awk -f RO.awk 20msaadv.tr > ROaadv.tr
awk -f RO.awk 29msdsdv.tr > RODsdv.tr
awk -f RO.awk 20msdsr.tr > RODsr.tr
awk -f RO.awk 20msCBRaadv.tr > ROaadvCBR.tr
awk -f RO.awk 20msCBRdsr.tr > RODsrCBR.tr
awk -f RO.awk 20msCBRdsdv.tr > RODsdvCBR.tr
```

Setelah selesai melakukan simulasi di NS2 pada skenario dua yang menguji tentang kinerja *traffic behaviour* terhadap perubahan kecepatan *node* didapatkan suatu data yang berbentuk *file* “.tr”, yang didalamnya terdapat suatu

nilai dari masing perubahan kecepatan *node* pada *traffic behaviour* tersebut mulai dari NRLaodv sampai NRLdsdvCBR dari masing-masing jumlah *node*. Seperti pada Tabel 4.7, Gambar 4.7. untuk *traffic behaviour* FTP dan Tabel 4.8, Gambar 4.8 untuk CBR berikut.

Tabel 4.7. Data Rata-rata Nilai RO dari kecepatan *node* terhadap FTP

<i>Traffic Behaviour</i>	Kecepatan	<i>Routing Overhead</i>		
		AODV	DSR	DSDV
FTP	5 m/s	0.9947	0.9941	0.9923
	10 m/s	0.9954	0.9911	0.9946
	15 m/s	0.9962	0.9962	0.9958
	20 m/s	0.9962	0.9960	0.9951

Tabel 4.8. Data Rata-rata RO dari kecepatan *node* terhadap CBR

<i>Traffic Behaviour</i>	Kecepatan	<i>Routing Overhead</i>		
		AODV	DSR	DSDV
CBR	5 m/s	0.2935	0.9749	0.3111
	10 m/s	0.2880	0.9680	0.3551
	15 m/s	0.3692	0.9860	0.3786
	20 m/s	0.3655	0.9681	0.3375

Dalam skenario kedua yang dilakukan pada parameter *Routing Overhead* terhadap perubahan kecepatan *node*, sebagian *node* yang terletak pada area cakupan MANET aktif melakukan pengiriman data. Pengiriman dilakukan hanya pada *node-node* tertentu yaitu dari *node* 0 atau *node source* menuju ke *node* 7 sebagai *node client*, dari *node* 7 ke *node* 10 dan yang terakhir adalah dari *node* 10 ke *node* 4 secara beruntun dalam waktu 100 detik.

Nilai *routing overhead* untuk ketiga *routing protocol* memiliki nilai RO yang random terkadang naik dan juga turun, namun pada *routing protocol* DSR

pada *traffic behaviour* jenis CBR memiliki nilai yang hampir sama dengan nilai FTP yaitu berkisar diantara 0.9. Pada *traffic behaviour* jenis CBR pada *routing protocol* jenis DSR pada mengalami perubahan nilai yang mendekati nilai dari *routing protocol* DSR pada *traffic behaviour* jenis FTP pada saat perubahan kecepatan *node*, namun *routing protocol* yang paling memiliki nilai RO terkecil adalah *routing protocol* DSDV karena dapat dilihat dari nilai RO yang dihasilkan baik dari *traffic behaviour* jenis FTP maupun CBR nilai NRL dari AODV yang paling baik yaitu nilai tertingginya adalah 0.3786 pada CBR dan pada FTP dimiliki oleh *routing protocol* DSR yaitu dengan nilai tertinggi 0.9958.

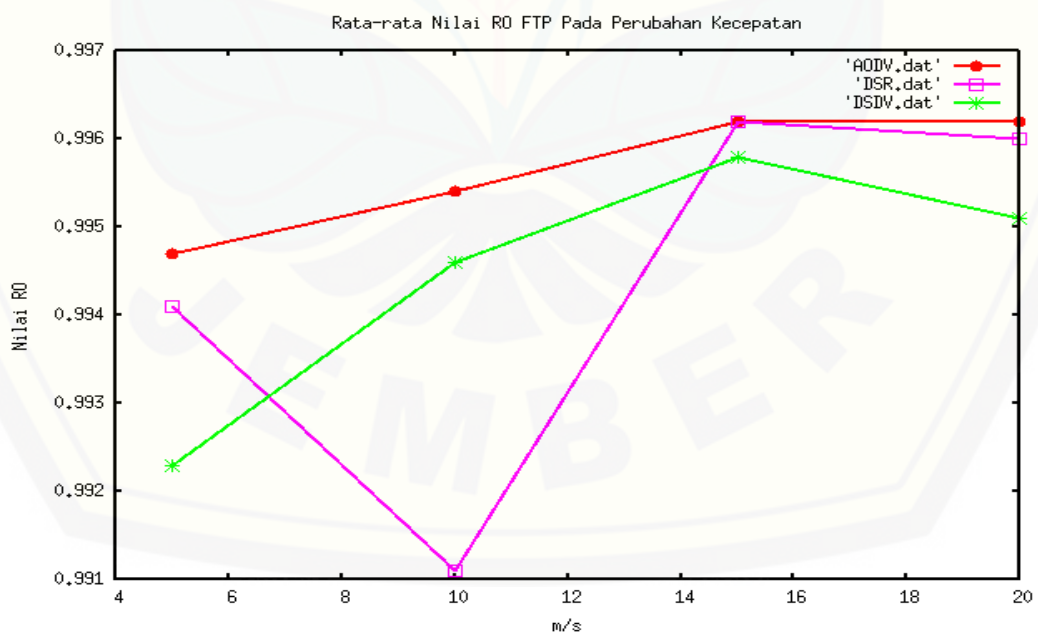
Untuk perbandingan antara nilai *traffic behaviour* jenis FTP dan CBR yang terbaik untuk diimplementasikan pada teknologi MANET untuk efisiensi RO pada suatu kinerja *traffic behaviour* terhadap perubahan kecepatan *node* adalah pada jenis FTP, karena pada hasil *output file trace* jenis FTP ini sangat sedikit mengalami *collusium* atau *paket drop*. Dengan memiliki nilai rata-rata terendah yang dimiliki oleh *routing protocol* DSDV yaitu 0.9958. Dan tersebut menandakan bahwa pada nilai *routing overhead* dalam skenario dua *routing protocol* DSDV dengan *traffic behaviour* FTP yang memiliki kinerja paling efisien.

Dalam penabelan dari beberapa perubahan kecepatan *node* yang terdiri dari 5 m/s, 10 m/s, 15 m/s, dan 20 m/s, pengaruh dari perubahan kecepatan *node* yang menunjukkan nilai RO yang paling bagus yaitu DSDV, yang kedua adalah DSR, dan yang ketiga AODV. Nilai RO tersebut dilihat dari kedua tabel 4.7 dan tabel 4.8 diatas menunjukkan bahwa *routing protocol* DSDV dengan *traffic behaviour* jenis FTP yang paling bagus digunakan untuk teknologi MANET karena minimnya paket *drop* yang terjadi.

Konsep dari *Routing Overhead* (RO) yaitu semakin kecil nilai *routing overhead* suatu *protocol routing*, maka semakin efisien kinerja *protocol routing* tersebut. Oleh maka dari itu dalam skenario kedua ini yaitu pada *routing overhead*, *routing protocol* yang paling efisien adalah pada *routing protocol* DSDV dengan nilai 0.9958 dengan sedikit paket *drop* dan *traffic behaviour* yang paling cocok digunakan oleh *routing protocol* ini adalah jenis FTP. Dari

data-data dan penjelasan tersebut dapat dilihat melalui grafik *gnuplot*, dengan mengetikkan perintah dibawah ini untuk membuat grafik *gnuplot*.

```
gnuplot
cd 'C:\cygwin\home\saptomo\ROFTPS'
set title "Rata-rata Nilai RO FTP Pada Perubahan
kecepatan"
set xlabel "m/s"
set ylabel "Nilai FTP"
set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 4 lw 2 pt 4 ps 2
set style line 3 lt 2 lw 2 pt 3 ps 2
plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```



Gambar 4.12. Grafik data rata-rata nilai RO dari perubahan kecepatan terhadap *traffic behaviour* jenis FTP


```
gnuplot

cd 'C:\cygwin\home\saptomo\ROCBRS'

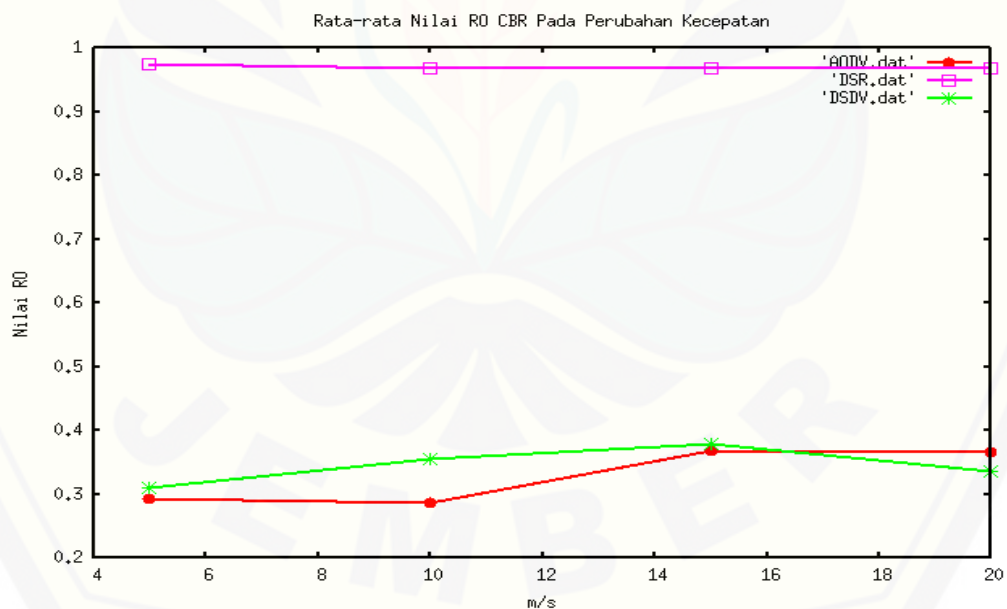
set title "Rata-rata Nilai RO CBR Pada Perubahan
kecepatan"

set xlabel "m/s"

set ylabel "Nilai CBR"

set style line 1 lt 1 lw 2 pt 7 ps 2
set style line 2 lt 4 lw 2 pt 4 ps 2
set style line 3 lt 2 lw 2 pt 3 ps 2

plot 'AODV.dat' with linespoints ls 1, 'DSR.dat'
with linespoints ls 2, 'DSDV.dat' with linespoints ls 3
```



Gambar 4.13. Grafik data rata-rata nilai RO dari perubahan kecepatan terhadap *traffic behaviour* jenis CBR

Dari grafik tersebut dapat membuktikan data dari tabel 4.7 dan tabel 4.8 bahwa *routing protocol* DSDV yang memiliki kinerja yang paling efisien untuk teknologi MANET baik pada *traffic behaviour* jenis FTP, dan pada jenis CBR *routing protocol* DSDV juga yang paling baik. Namun kembali lagi pada penjelasan sebelumnya yaitu pada CBR sering terjadi *collusion* atau paket *drop* yang terjadi sehingga dapat disimpulkan *traffic behaviour* yang cocok untuk *routing protocol* DSDV adalah jenis FTP pada parameter RO untuk perubahan kecepatan *node*.

Pada grafik tersebut *routing protocol* DSR pada *traffic behaviour* jenis CBR mengalami kenaikan nilai RO secara signifikan yaitu berkisar memiliki nilai hampir 1.0 berbeda dengan *routing protocol* lainnya yang hanya berkisar diantara 0.2 sampai 0.4, namun pada *routing protocol* ini tetap menunjukkan efisiensinya yang tidak terlalu memiliki kenaikan nilai pada perubahan perubahan kecepatan *node* selanjutnya.

Dari keseluruhan penelitian didapatkan suatu bukti bahwa *traffic behaviour* jenis FTP memiliki kinerja yang paling efisien daripada CBR, karena *traffic behaviour* jenis FTP ini sangat sedikit bahkan tidak mengalami *packet drop* dari pada *traffic behaviour* jenis CBR dan pengaruh dari perubahan jumlah dan kecepatan *node* pada *traffic behaviour* jenis FTP ini lebih baik digunakan pada MANET daripada CBR dengan melihat kondisi simulasi yang terjadi. Pada skenario pertama dengan kondisi perubahan jumlah *node* *traffic behaviour* jenis FTP dengan *routing protocol* DSR merupakan *traffic behaviour* yang efisien untuk *normalized routing load* dan AODV untuk *routing overhead*, sedangkan pada skenario kedua dengan kondisi perubahan kecepatan *node* *traffic behaviour* jenis FTP dengan *routing protocol* DSR untuk *normalized routing load* dan DSDV untuk *routing overhead*. Selain itu dapat dibuktikan bahwa *routing protocol* DSR dengan *traffic behaviour* jenis FTP memiliki kinerja yang efisien dengan parameter *normalized routing load* sedangkan untuk *routing overhead* dimiliki oleh dua *routing protocol* AODV dan DSDV.

4.4 Kelebihan dan Kelemahan *Routing Protocol* Terhadap FTP (*Traffic Behaviour*).

Dari hasil penelitian terhadap *traffic behaviour* dapat diperoleh kelemahan dan kelebihan dari masing-masing *routing protocol* yang digunakan. Dari penelitian yang didapatkan menunjukkan bahwa *traffic behaviour* yang memiliki kinerja yang paling baik adalah FTP, oleh maka dari itu kelebihan dan kelemahan *routing protocol* terhadap *traffic behaviour* ini dilihat dari sisi *traffic behaviour* jenis FTP. Untuk kelebihan dan kelemahannya tersebut dapat dilihat dibawah ini.

Tabel 4.9. Kelebihan dan Kelemahan pada Skenario 1

No	<i>Routing Protocol</i>	Kelebihan	Kelemahan
1	AODV	Memiliki Nilai RO yang yang lebih baik dibandingkan dengan <i>routing protocol</i> lainnya	Memiliki nilai NRL yang cukup baik pada peningkatan jumlah <i>node</i> tetapi nilai NRL pada <i>routing</i> ini kurang efisien dibandingkan dengan <i>routing</i> DSR.
2	DSR	nilai NRL memiliki performa yang baik karena memenuhi nilai efisiensi parameter tersebut, dengan mengalami penurunan nilai pada peningkatan jumlah <i>node</i>	Mengalami penurunan dan kenaikan pada peningkatan jumlah <i>node</i> yang mengakibatkan ketidak stabilan pada <i>routing</i> ini, sehingga kinerja <i>routing</i> tidak efisien
3	DSDV	Sempat mengalami penurunan nilai RO pada peningkatan jumlah <i>node</i> meskipun nilai akhirnya cukup besar	NRL meningkat cukup tajam seiring dengan peningkatan jumlah <i>node</i> yang diterapkan.

Tabel 4.10. Kelebihan dan Kelemahan pada Skenario 2

No	<i>Routing Protocol</i>	Kelebihan	Kelemahan
1	AODV	Memiliki nilai NRL yang memenuhi konsep dan mengalami kenaikan dan penurunan yang stabil pada peningkatan kecepatan <i>node</i>	Memiliki nilai RO yang sangat tinggi dibandingkan dengan <i>routing</i> lainnya, meskipun mengalami kenaikan nilai yang cukup stabil.
2	DSR	Memiliki nilai NRL yang cukup baik dan memenuhi konsep dari parameter tersebut meskipun sempat memiliki nilai yang cukup tinggi, namun pada peningkatan kecepatan berikutnya mengalami penurunan nilai	Nilai RO yang cukup besar dan mengalami kenaikan yang cukup drastis meskipun sempat mengalami penurunan nilai pada peningkatan kecepatan.
3	DSDV	Memiliki nilai RO yang efisiensi dan performa dari <i>routing</i> yang cukup baik	Meskipun sesuai dengan kriteria parameter NRL namun DSDV pada peningkatan kecepatan <i>node</i> memiliki nilai NRL yang sangat besar.

BAB 5. PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan yang telah dilakukan maka dapat disimpulkan bahwa :

1. Pada skenario pertama dengan kondisi perubahan jumlah *node*, *traffic behaviour* jenis FTP memiliki kinerja yang lebih baik daripada CBR pada *routing protocol* DSR untuk *normalized routing load* dengan nilai rata-rata terkecil yaitu 0.012 dan AODV untuk *routing overhead* dengan nilai rata-rata terkecil 0.9934, sedangkan pada skenario kedua dengan kondisi perubahan kecepatan *node*, *traffic behaviour* yang memiliki kinerja lebih baik adalah FTP daripada CBR, pada *routing protocol* DSR untuk *normalized routing load* dengan nilai rata-rata terkecil 0.013 dan DSDV untuk *routing overhead* dengan nilai rata-rata terkecil 0.9945.
2. Pengaruh penambahan jumlah dan perubahan kecepatan *node* pada *traffic behaviour* jenis FTP lebih baik digunakan pada MANET daripada CBR berdasarkan parameter *routing overhead* dan *normalized routing load* dengan *routing protocol* yang digunakan, dengan pertimbangan *traffic behaviour* jenis CBR saat pengiriman paket data sering terjadi paket *drop* (Dapat dilihat pada tabel 4.1 sampai tabel 4.8.).

5.2 Saran

Beberapa saran yang dapat dipertimbangkan untuk pengembangan pengujian MANET selanjutnya yaitu :

1. Membandingkan dengan *transport agent* yang lain dari TCPSink yaitu TCPReho atau TCPTahoe.
2. Dapat dilakukan penelitian dalam hal efisiensi energi yang dibutuhkan untuk *traffic behaviour* FTP maupun CBR dengan *routing protocol* yang sama pada MANET.
3. Dapat dilakukan penelitian dalam hal pergerakan *node* secara random *node*.

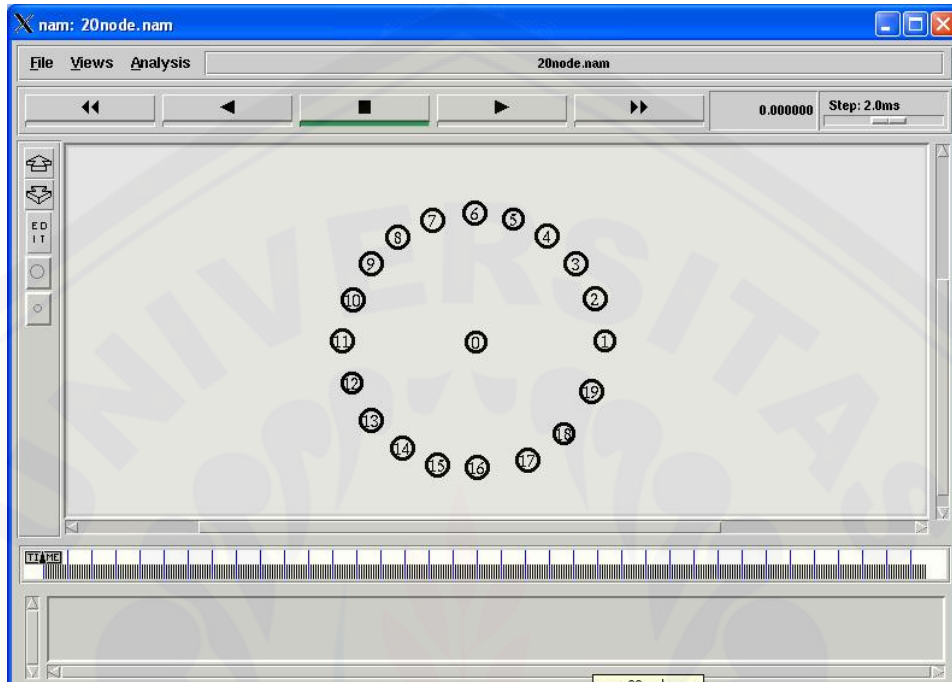
DAFTAR PUSTAKA

- Aji Sasongko Septian, Sukiswo Ajulian Zahra Ajub, 2010, “Analisis Performansi dan Simulasi Protokol ZRP (*Zone Routing Protocol*) Pada MANET (*Mobile Ad Hoc Network*) dengan Menggunakan NS-2”.
- Arifin, M. Zen Samsono, Amron Haryadi, Nopember 2010, “Analisa QoS Protokol *Routing AODV* Pada Jaringan VANET Berbasis Kecepatan Node”, EEPIS.
- Hartati Harahap Evi, 2014, “Analisis Performansi Protokol *AODV (Ad Hoc On demand Distance Vector)* dan *DSR (Dynamic Source Routing)* Terhadap *Active Attack* Pada MANET (*Mobile Ad Hoc Network*) Ditinjau Dari QoS (*Quality Of Service*) Jaringan”.
- Imawan Didik, 2009, “Analisis Kinerja Pola-pola Trafik Pada Beberapa Protokol *Routing* Dalam Jaringan MANET”.
- Kembuan, Widyawan, Suning Kusumawardan Sri, Mei 2012, “Analisis Kinerja *Reactive Routing Protocol* dalam *Mobile Ad-Hoc Network (MANET)* Menggunakan NS-2 (*Network Simulator*), JNTETI, Vol 1, No. 1.
- M. Cardei, I. Cardei, D.-Z. Du, 2004, “*Quality of Service Routing in Mobile Ad Hoc Networks*”. Kluwer Academic Publishers.
- Megawan Sunario, April 2013, “Pengaruh Densitas *Wireless Mobile Node* dan Jumlah *Wireless Mobile Node* Sumber Terhadap *PATH Discovery Time* Pada Protokol *Routing AODV*”. ISSN. 1412-0100, Vol. 14, No. 1.
- Muhardafen Manullang Fenmelin, Dwi Wahyu Cahyani Niken, Suryani Vera, April 2012, “Analisis Performansi *Routing Hybrid Wireless Mesh Protocol (HWMP)* Pada *Wireless Mesh Network (WMP)* Berdasarkan Standar IEEE 802.11s”. ISSN.1412-0100, Vol 13, No. 1.
- Nashrul Haqqi Ma'ruf, Sukiswo, Ajulian Zahra Ajub, 2013, “Analisis Kinerja PUMA Pada MANET Menggunakan NS-2”.
- Pandu Adityas Fitra, Sukiswo, Ajulian Zahra Ajub, 2013, “Analisis MDART Pada MANET Menggunakan NS-2”.

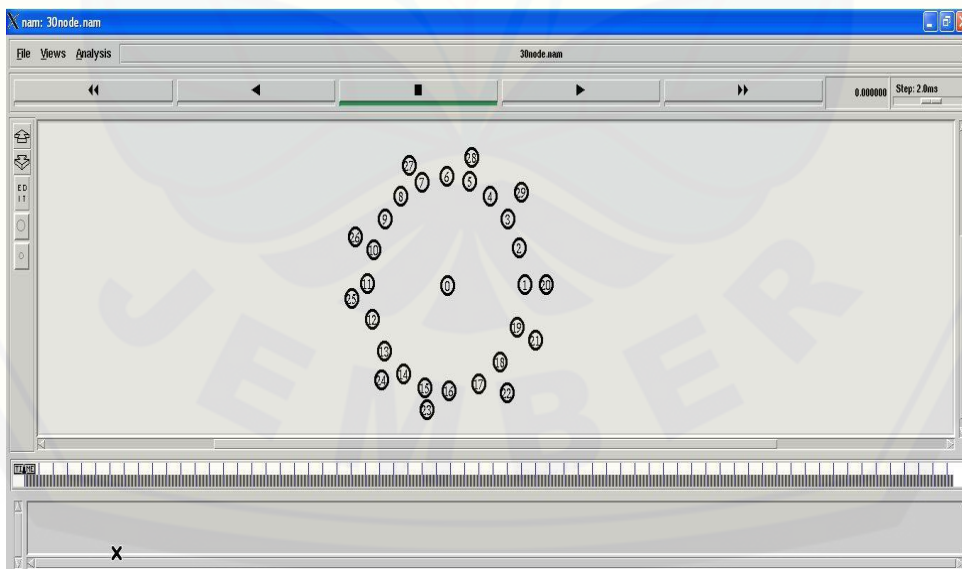
- Prasant Mohapatra, Srikanth V. Khrisnamurthy, 2005, “*Ad Hoc Network Technologies and Protocols*”. Springer
- Rudi Hartono, S.Si, Agus Purnomo, S.Si, Januari 2011, “*Wireless Network 802.11*”. D3 TI FMIPA UNS.
- Sidharta Yonas, Widjaja Damar, 2013, “Perbandingan Unjuk Kerja Protokol *Routing Ad Hoc On-Demand Distance Vector (AODV)* dan *Dynamic Source Routing (DSR)* Pada Jaringan MANET”.
- Sun Jun-Zhao, 2010, “*Mobile Ad Hoc Networking: An Essential Technology for Pervasive Computing*”. University of Oulu.
- Teerawat Issariyakul, Ekram Hossain, 2009, “*Introduction to Network Simulator NS2*”. Springer.
- Wahyu Putri Utami Yuliana, Jusak, Sukmaaji Anjik, “Analisis Perbandingan Unjuk Kerja Algoritma *Congestion Control* Pada TCP TAHOE, RENO, dan SACK (*Selective Acknowledgment*)”.

LAMPIRAN

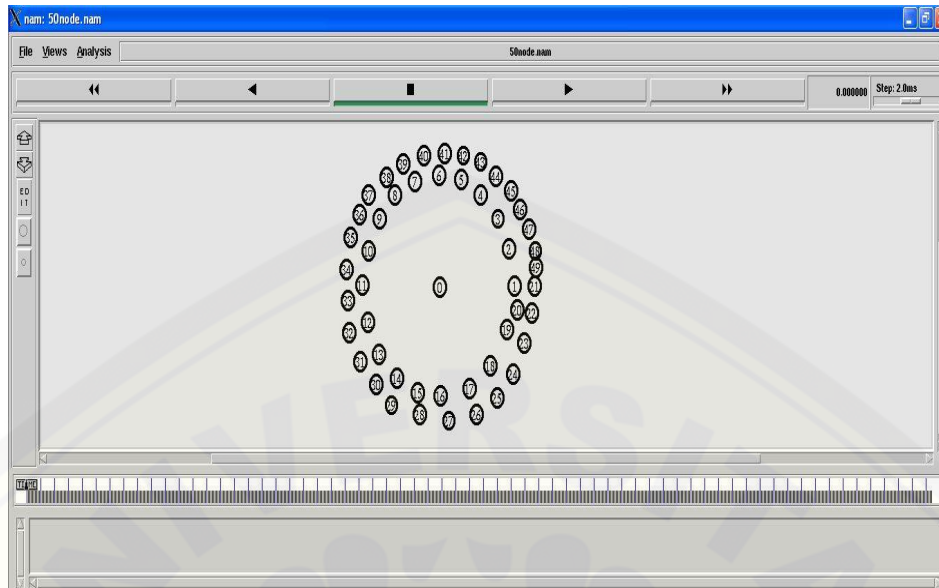
1. Lampiran Simulasi MANET Pada Beberapa Percobaan



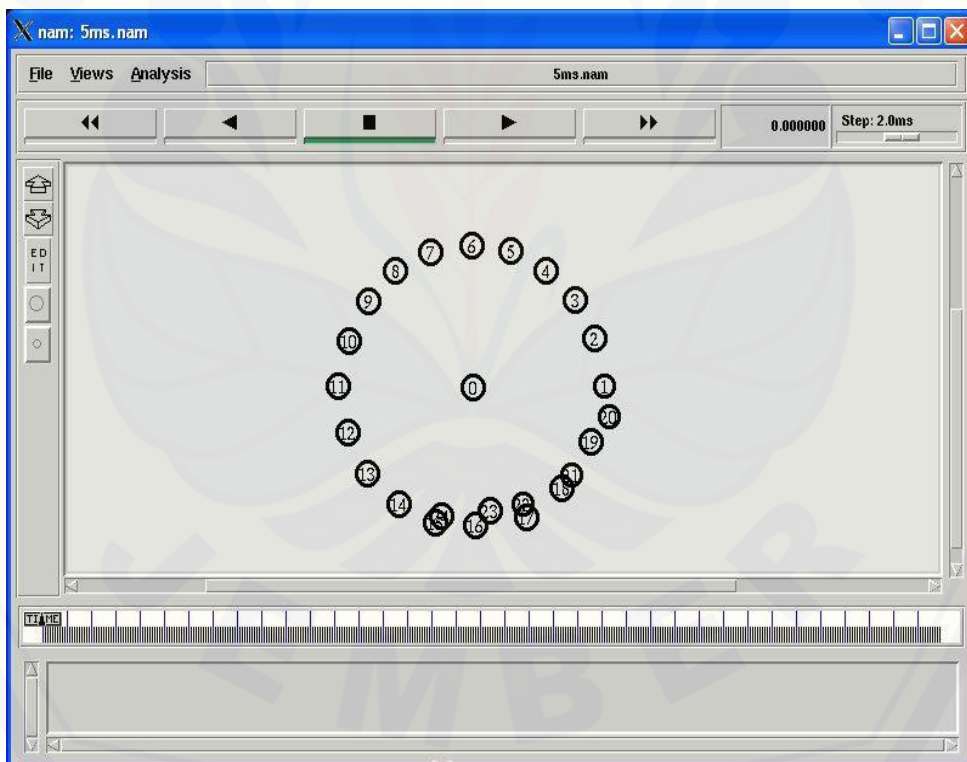
Gambar 1. Posisi Node Pada Jumlah 20 Node



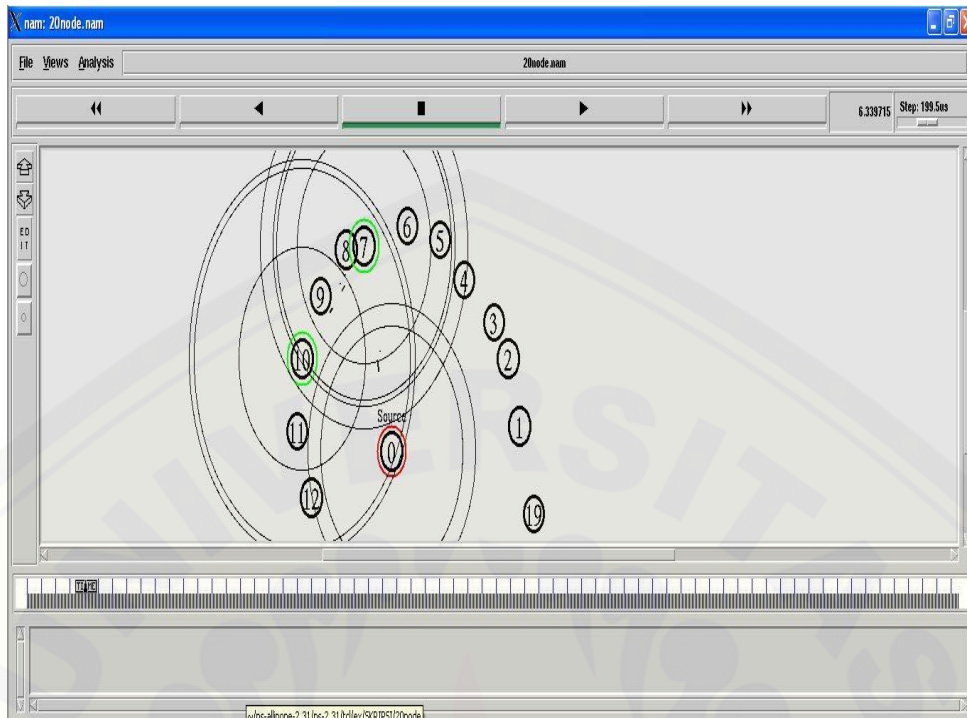
Gambar 2. Posisi Node Pada Perubahan Jumlah 30 Node



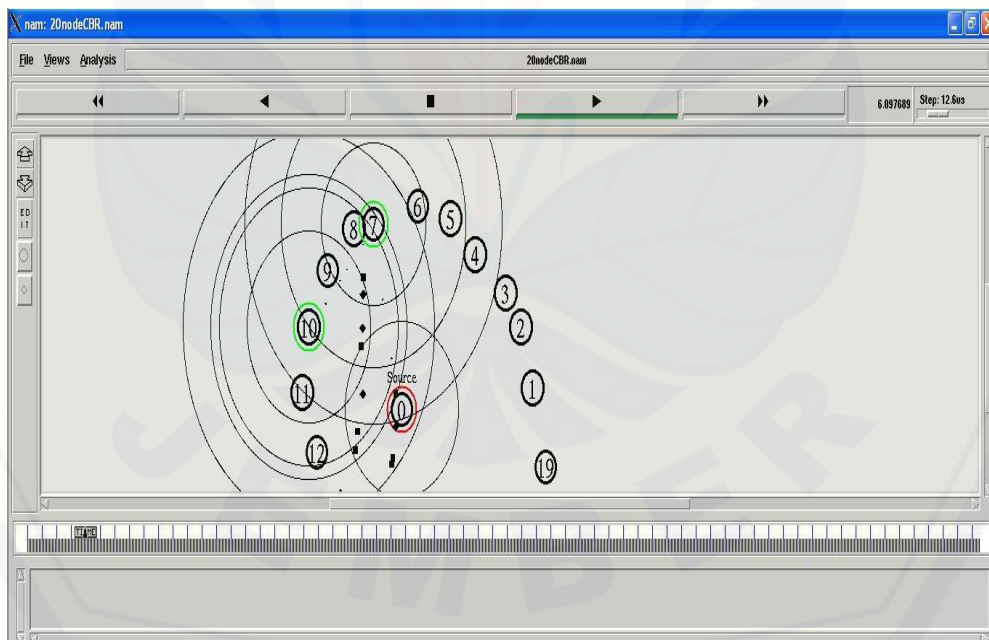
Gambar 3. Posisi Node Pada Jumlah Perubahan 50 Node



Gambar 4. Posisi Node Pada Perubahan Kecepatan.



Gambar 5. Proses Pengiriman Paket Data pada Traffic Behaviour Jenis FTP dengan Jumlah Node 20



Gambar 6. Proses Pengiriman Paket Data pada Traffic Behaviour Jenis CBR dengan Jumlah Node 20

2. List Program tcl Pada Perubahan Jumlah Node FTP

```
# M. Wahyu Abdul Ghofur 111910201054
# Analisis Kinerja Traffic Behaviour Pada MANET Dengan Menggunakan
Ns2
# =====
# Define options
# =====
set val(chan) Channel/WirelessChannel ;# Channel Type
set val(prop) Propagation/TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy ;# network interface type
set val(mac) Mac/802_11 ;# MAC type
set val(ifq) Queue/DropTail/PriQueue ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ;# max packet in ifq
set val(nn) 20 ;# number of mobilenodes
set val(rp) AODV ;# routing protocol
set val(x) 1500
set val(y) 1000
# =====
# Main Program
# =====
# inialisasi variabel global

set ns_ [new Simulator]
#create the nam and trace file:
set tracefd [open 20node2.tr w]
$ns_ trace-all $tracefd
set namtrace [open 20node.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
set topo [new Topography]
```

```
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
set chan_1_ [new $val(chan)]

# Defining Node Configuration
$ns_ node-config -adhocRouting $val(rp) \
    -llType $val(ll) \
    -macType $val(mac) \
    -ifqType $val(ifq) \
    -ifqLen $val(ifqlen) \
    -antType $val(ant) \
    -propType $val(prop) \
    -phyType $val(netif) \
    -topoInstance $topo \
    -agentTrace ON \
    -routerTrace ON \
    -macTrace ON \
    -movementTrace ON \
    -channel $chan_1_

# create nodes
for {set i 0} {$i < $val(nn)} {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0;
}
#Inisialisasi (X,Y, dan Z=0)
#Create 20 nodes
$node_(0) set X_ 599
$node_(0) set Y_ 299
$node_(0) set Z_ 0.0
$ns_ initial_node_pos $node_(0) 40
```

\$node_(1) set X_ 841
\$node_(1) set Y_ 301
\$node_(1) set Z_ 0.0
\$ns_ initial_node_pos \$node_(1) 40
\$node_(2) set X_ 824
\$node_(2) set Y_ 382
\$node_(2) set Z_ 0.0
\$ns_ initial_node_pos \$node_(2) 40
\$node_(3) set X_ 787
\$node_(3) set Y_ 447
\$node_(3) set Z_ 0.0
\$ns_ initial_node_pos \$node_(3) 40
\$node_(4) set X_ 733
\$node_(4) set Y_ 498
\$node_(4) set Z_ 0.0
\$ns_ initial_node_pos \$node_(4) 40

\$node_(5) set X_ 669
\$node_(5) set Y_ 531
\$node_(5) set Z_ 0.0
\$ns_ initial_node_pos \$node_(5) 40
\$node_(6) set X_ 597
\$node_(6) set Y_ 541
\$node_(6) set Z_ 0.0
\$ns_ initial_node_pos \$node_(6) 40
\$node_(7) set X_ 519
\$node_(7) set Y_ 528
\$node_(7) set Z_ 0.0
\$ns_ initial_node_pos \$node_(7) 40
\$node_(8) set X_ 454
\$node_(8) set Y_ 497

\$node_(8) set Z_ 0.0
\$ns_ initial_node_pos \$node_(8) 40
\$node_(9) set X_ 404
\$node_(9) set Y_ 446
\$node_(9) set Z_ 0.0
\$ns_ initial_node_pos \$node_(9) 40
\$node_(10) set X_ 369
\$node_(10) set Y_ 378
\$node_(10) set Z_ 0.0
\$ns_ initial_node_pos \$node_(10) 40
\$node_(11) set X_ 349
\$node_(11) set Y_ 302
\$node_(11) set Z_ 0.0
\$ns_ initial_node_pos \$node_(11) 40
\$node_(12) set X_ 366
\$node_(12) set Y_ 223
\$node_(12) set Z_ 0.0
\$ns_ initial_node_pos \$node_(12) 40
\$node_(13) set X_ 403
\$node_(13) set Y_ 153
\$node_(13) set Z_ 0.0
\$ns_ initial_node_pos \$node_(13) 40
\$node_(14) set X_ 462
\$node_(14) set Y_ 101
\$node_(14) set Z_ 0.0
\$ns_ initial_node_pos \$node_(14) 40
\$node_(15) set X_ 528
\$node_(15) set Y_ 69
\$node_(15) set Z_ 0.0
\$ns_ initial_node_pos \$node_(15) 40
\$node_(16) set X_ 603

```
$node_(16) set Y_ 64
$node_(16) set Z_ 0.0
$ns_ initial_node_pos $node_(16) 40
$node_(17) set X_ 697
$node_(17) set Y_ 78
$node_(17) set Z_ 0.0
$ns_ initial_node_pos $node_(17) 40
$node_(18) set X_ 764
$node_(18) set Y_ 128
$node_(18) set Z_ 0.0
$ns_ initial_node_pos $node_(18) 40
$node_(19) set X_ 817
$node_(19) set Y_ 206
$node_(19) set Z_ 0.0
$ns_ initial_node_pos $node_(19) 40
```

Giving Mobility to nodes

```
$ns_ at 0.20 "$node_(0) setdest 379.0 210.0 10.0"
$ns_ at 1.25 "$node_(1) setdest 159.0 330.0 10.0"
$ns_ at 0.75 "$node_(2) setdest 379.0 349.0 10.0"
$ns_ at 0.75 "$node_(3) setdest 556.0 302.0 10.0"
$ns_ at 0.20 "$node_(4) setdest 309.0 211.0 10.0"
$ns_ at 2.25 "$node_(5) setdest 179.0 200.0 10.0"
$ns_ at 3.25 "$node_(6) setdest 190.0 250.0 10.0"
$ns_ at 2.25 "$node_(7) setdest 150.0 200.0 10.0"
$ns_ at 10.25 "$node_(8) setdest 140.0 210.0 10.0"
$ns_ at 23.25 "$node_(9) setdest 170.0 230.0 10.0"
$ns_ at 12.25 "$node_(10) setdest 200.0 230.0 10.0"
$ns_ at 5.25 "$node_(11) setdest 734.0 100.0 10.0"
$ns_ at 4.25 "$node_(12) setdest 500.0 230.0 10.0"
```



```
$ns_ at 6.25 "$node_(13) setdest 400.0 300.0 10.0"  
$ns_ at 7.25 "$node_(14) setdest 800.0 230.0 10.0"  
$ns_ at 8.25 "$node_(15) setdest 700.0 500.0 10.0"  
$ns_ at 9.25 "$node_(16) setdest 900.0 100.0 10.0"  
$ns_ at 30.25 "$node_(17) setdest 600.0 230.0 10.0"  
$ns_ at 60.25 "$node_(18) setdest 400.0 300.0 10.0"  
$ns_ at 50.25 "$node_(19) setdest 170.0 150.0 10.0"
```

#defining heads

```
$ns_ at 0.0 "$node_(0) label Source"  
$ns_ at 0.0 "$node_(0) shape hexagon "  
$ns_ at 0.0 "$node_(0) add-mark N0 red circle"  
$ns_ at 2.0 "$node_(7) add-mark N0 green circle"  
$ns_ at 3.0 "$node_(10) add-mark N0 green circle"  
$ns_ at 10.2 "$node_(4) add-mark N0 blue circle"
```

#Set a TCP connection between node 0 and node 7

```
set tcp [new Agent/TCP]  
$tcp set class_ 2  
set sink [new Agent/TCPSink]  
$ns_ attach-agent $node_(0) $tcp  
$ns_ attach-agent $node_(7) $sink  
$ns_ connect $tcp $sink  
set ftp [new Application/FTP]  
$ftp attach-agent $tcp  
$ns_ at 2.0 "$ftp start"
```

#Set a TCP connection between node 7 and node 10

```
set tcp [new Agent/TCP]  
$tcp set class_ 2  
set sink [new Agent/TCPSink]
```

```
$ns_ attach-agent $node_(7) $tcp
$ns_ attach-agent $node_(10) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 3.0 "$ftp start"
#Set a TCP connection between node 10 and node 4
set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(10) $tcp
$ns_ attach-agent $node_(4) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 10.5 "$ftp start"

proc stop { } {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
    exec nam 20node.nam &
    exit 0
}
puts "Starting Simulation....."
$ns_ at 100 "stop"
$ns_ run
```

3. List Program Pada Perubahan Kecepatan Node CBR

```
# M. Wahyu Abdul Ghofur 111910201054
# Analisis Kinerja Traffic Behaviour Pada MANET Dengan Menggunakan
Ns2
# =====
# Define options
# =====
set val(chan) Channel/WirelessChannel ;# Channel Type
set val(prop) Propagation/TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy ;# network interface type
set val(mac) Mac/802_11 ;# MAC type
set val(ifq) Queue/DropTail/PriQueue ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ;# max packet in ifq
set val(nn) 25 ;# number of mobilenodes
set val(rp) DSR ;# routing protocol
set val(x) 1500
set val(y) 1000
# =====
# Main Program
# =====
# inialisasi variabel global
set ns_ [new Simulator]
#create the nam and trace file:
set tracefd [open 5msCBRdsr.tr w]
$ns_ trace-all $tracefd
set namtrace [open 5msCBR.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
```

```
create-god $val(nn)
set chan_1_ [new $val(chan)]

# Defining Node Configuration
$ns_ node-config -adhocRouting $val(rp) \
    -llType $val(ll) \
    -macType $val(mac) \
    -ifqType $val(ifq) \
    -ifqLen $val(ifqlen) \
    -antType $val(ant) \
    -propType $val(prop) \
    -phyType $val(netif) \
    -topoInstance $topo \
    -agentTrace ON \
    -routerTrace ON \
    -macTrace ON \
    -movementTrace ON \
    -channel $chan_1_

# create nodes
for {set i 0} {$i < $val(nn)} {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0;
}
#Inisialisasi (X,Y, dan Z=0)
#Create 25 nodes
$node_(0) set X_ 599
$node_(0) set Y_ 299
$node_(0) set Z_ 0.0
$ns_ initial_node_pos $node_(0) 40
$node_(1) set X_ 841
```

\$node_(1) set Y_ 301
\$node_(1) set Z_ 0.0
\$ns_ initial_node_pos \$node_(1) 40
\$node_(2) set X_ 824
\$node_(2) set Y_ 382
\$node_(2) set Z_ 0.0
\$ns_ initial_node_pos \$node_(2) 40
\$node_(3) set X_ 787
\$node_(3) set Y_ 447
\$node_(3) set Z_ 0.0
\$ns_ initial_node_pos \$node_(3) 40
\$node_(4) set X_ 733
\$node_(4) set Y_ 498
\$node_(4) set Z_ 0.0
\$ns_ initial_node_pos \$node_(4) 40
\$node_(5) set X_ 669
\$node_(5) set Y_ 531
\$node_(5) set Z_ 0.0
\$ns_ initial_node_pos \$node_(5) 40
\$node_(6) set X_ 597
\$node_(6) set Y_ 541
\$node_(6) set Z_ 0.0
\$ns_ initial_node_pos \$node_(6) 40
\$node_(7) set X_ 519
\$node_(7) set Y_ 528
\$node_(7) set Z_ 0.0
\$ns_ initial_node_pos \$node_(7) 40
\$node_(8) set X_ 454
\$node_(8) set Y_ 497
\$node_(8) set Z_ 0.0
\$ns_ initial_node_pos \$node_(8) 40

\$node_(9) set X_ 404
\$node_(9) set Y_ 446
\$node_(9) set Z_ 0.0
\$ns_ initial_node_pos \$node_(9) 40
\$node_(10) set X_ 369
\$node_(10) set Y_ 378
\$node_(10) set Z_ 0.0
\$ns_ initial_node_pos \$node_(10) 40
\$node_(11) set X_ 349
\$node_(11) set Y_ 302
\$node_(11) set Z_ 0.0
\$ns_ initial_node_pos \$node_(11) 40
\$node_(12) set X_ 366
\$node_(12) set Y_ 223
\$node_(12) set Z_ 0.0
\$ns_ initial_node_pos \$node_(12) 40
\$node_(13) set X_ 403
\$node_(13) set Y_ 153
\$node_(13) set Z_ 0.0
\$ns_ initial_node_pos \$node_(13) 40
\$node_(14) set X_ 462
\$node_(14) set Y_ 101
\$node_(14) set Z_ 0.0
\$ns_ initial_node_pos \$node_(14) 40
\$node_(15) set X_ 528
\$node_(15) set Y_ 69
\$node_(15) set Z_ 0.0
\$ns_ initial_node_pos \$node_(15) 40
\$node_(16) set X_ 603
\$node_(16) set Y_ 64
\$node_(16) set Z_ 0.0

\$ns_ initial_node_pos \$node_(16) 40

\$node_(17) set X_ 697

\$node_(17) set Y_ 78

\$node_(17) set Z_ 0.0

\$ns_ initial_node_pos \$node_(17) 40

\$node_(18) set X_ 764

\$node_(18) set Y_ 128

\$node_(18) set Z_ 0.0

\$ns_ initial_node_pos \$node_(18) 40

\$node_(19) set X_ 817

\$node_(19) set Y_ 206

\$node_(19) set Z_ 0.0

\$ns_ initial_node_pos \$node_(19) 40

\$node_(20) set X_ 850

\$node_(20) set Y_ 250

\$node_(20) set Z_ 0.0

\$ns_ initial_node_pos \$node_(20) 40

\$node_(21) set X_ 780

\$node_(21) set Y_ 150

\$node_(21) set Z_ 0.0

\$ns_ initial_node_pos \$node_(21) 40

\$node_(22) set X_ 690

\$node_(22) set Y_ 100

\$node_(22) set Z_ 0.0

\$ns_ initial_node_pos \$node_(22) 40

\$node_(23) set X_ 630

\$node_(23) set Y_ 90

\$node_(23) set Z_ 0.0

\$ns_ initial_node_pos \$node_(23) 40

\$node_(24) set X_ 540

\$node_(24) set Y_ 80

```
$node_(24) set Z_ 0.0
```

```
$ns_ initial_node_pos $node_(24) 40
```

```
## Giving Mobility to nodes
```

```
  $ns_ at 0.20 "$node_(0) setdest 379.0 210.0 5.0"
```

```
  $ns_ at 1.25 "$node_(1) setdest 159.0 330.0 5.0"
```

```
  $ns_ at 0.75 "$node_(2) setdest 379.0 349.0 5.0"
```

```
  $ns_ at 0.75 "$node_(3) setdest 556.0 302.0 5.0"
```

```
  $ns_ at 0.20 "$node_(4) setdest 309.0 211.0 5.0"
```

```
  $ns_ at 2.25 "$node_(5) setdest 179.0 200.0 5.0"
```

```
  $ns_ at 3.25 "$node_(6) setdest 190.0 250.0 5.0"
```

```
  $ns_ at 2.25 "$node_(7) setdest 150.0 200.0 5.0"
```

```
  $ns_ at 10.25 "$node_(8) setdest 140.0 210.0 5.0"
```

```
  $ns_ at 23.25 "$node_(9) setdest 170.0 230.0 5.0"
```

```
  $ns_ at 12.25 "$node_(10) setdest 200.0 230.0 5.0"
```

```
  $ns_ at 5.25 "$node_(11) setdest 734.0 100.0 5.0"
```

```
  $ns_ at 4.25 "$node_(12) setdest 500.0 230.0 5.0"
```

```
  $ns_ at 6.25 "$node_(13) setdest 400.0 300.0 5.0"
```

```
  $ns_ at 7.25 "$node_(14) setdest 800.0 230.0 5.0"
```

```
  $ns_ at 8.25 "$node_(15) setdest 700.0 500.0 5.0"
```

```
  $ns_ at 9.25 "$node_(16) setdest 900.0 100.0 5.0"
```

```
  $ns_ at 30.25 "$node_(17) setdest 600.0 230.0 5.0"
```

```
  $ns_ at 60.25 "$node_(18) setdest 400.0 300.0 5.0"
```

```
  $ns_ at 50.25 "$node_(19) setdest 170.0 150.0 5.0"
```

```
  $ns_ at 20.25 "$node_(20) setdest 180.0 100.0 5.0"
```

```
  $ns_ at 30.25 "$node_(21) setdest 599.0 299.0 5.0"
```

```
  $ns_ at 40.25 "$node_(22) setdest 462.0 101.0 5.0"
```

```
  $ns_ at 25.25 "$node_(23) setdest 336.0 223.0 5.0"
```

```
  $ns_ at 35.25 "$node_(24) setdest 404.0 446.0 5.0"
```

```
#defining heads
```

```
$ns_ at 0.0 "$node_(0) label Source"  
$ns_ at 0.0 "$node_(0) shape hexagon "  
$ns_ at 0.0 "$node_(0) add-mark N0 red circle"  
$ns_ at 2.0 "$node_(7) add-mark N0 green circle"  
$ns_ at 3.0 "$node_(10) add-mark N0 green circle"  
$ns_ at 10.5 "$node_(4) add-mark N0 blue circle"  
  
#Set a TCP connection between node 0 and node 7  
set udp [new Agent/UDP]  
$udp set class_ 2  
set null [new Agent/Null]  
$ns_ attach-agent $node_(0) $udp  
$ns_ attach-agent $node_(7) $null  
$ns_ connect $udp $null  
set cbr [new Application/Traffic/CBR]  
$cbr attach-agent $udp  
$ns_ at 2.0 "$cbr start"  
set udp [new Agent/UDP]  
$udp set class_ 2  
set null [new Agent/Null]  
$ns_ attach-agent $node_(7) $udp  
$ns_ attach-agent $node_(10) $null  
$ns_ connect $udp $null  
set cbr [new Application/Traffic/CBR]  
$cbr attach-agent $udp  
$ns_ at 3.0 "$cbr start"  
set udp [new Agent/UDP]  
$udp set class_ 2  
set null [new Agent/Null]  
$ns_ attach-agent $node_(10) $udp  
$ns_ attach-agent $node_(4) $null
```

```
$ns_ connect $udp $null
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$ns_ at 10.5 "$cbr start"

proc stop {} {

global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
    exec nam 5msCBR.nam &
    exit 0
}
puts "Starting Simulation....."
$ns_ at 100 "stop"
$ns_ run
```

4. List Program AWK Routing Overhead

```
BEGIN {
sendpacket = 0;

recvpacket = 0;
}

$0 ~/^s.* AGT/ {sendpacket ++ ; #Calculate the number of
packets sent
}

$0 ~/^r.* AGT/ {recvpacket ++ ; #Calculate the number of
packets received
}

END {
printf "cbr send:%d recv:%d, getRatio:%.4f \n",
sendpacket,recvpacket,(recvpacket/sendpacket);
}
```


5. List Program AWK Normalized Routing Load

```
#####  
#           AWK Script to calculate Normalized Routing Load  
#  
#           Works with AODV, DSDV, DSR and OLSR  
#  
#####  
  
BEGIN{  
  recvd = 0;##### to calculate total number of  
  data packets received  
  rt_pkts = 0;##### to calculate total number of  
  routing packets received  
}  
  
{  
  ##### Check if it is a data packet  
  if (( $1 == "r" ) && ( $7 == "cbr" || $7 == "tcp" ) && ( $4 == "AGT" )) recvd++;  
  
  ##### Check if it is a routing packet  
  if (($1 == "s" || $1 == "f") && $4 == "RTR" && ($7 == "AODV"  
  || $7 == "message" || $7 == "DSR" || $7 == "OLSR")) rt_pkts++;  
}  
  
END{  
  printf("#####\n");  
  printf("\n");  
  printf("           Normalized Routing Load =  
  %.3f\n", rt_pkts/recvd);  
  printf("\n");  
  printf("#####\n");  
}
```