**SIMULASI PROTOKOL MANET DALAM KOMUNIKASI AUDIO PADA DAERAH PEGUNUNGAN MENGGUNAKAN ALGORITMA PA-SHORT**

**UNIVERSITAS UDAYANA**

**KOMPETENSI JARINGAN**

**SKRIPSI**



**I WAYAN ADI SAPUTRA**

**1608561017**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**BUKIT JIMBARAN**

**2020**

# SURAT PERNYATAAN KEASLIAN KARYA ILMIAH

# LEMBAR PENGESAHAN TUGAS AKHIR

Judul : SIMULASI PROTOKOL MANET DALAM KOMUNIKASI

AUDIO PADA DAERAH PEGUNUNGAN MENGGUNAKAN ALGORITMA PA-SHORT

Nama : I Wayan Adi Saputra

NIM : 1608561017

Pembimbing : 1. I Komang Ari Mogi, S.Kom., M.Kom.

2. Luh Arida Ayu Rahning Putri, S.Kom., M.Cs.

ASBTRAK

Judul : SIMULASI PROTOKOL MANET DALAM KOMUNIKASI

AUDIO PADA DAERAH PEGUNUNGAN MENGGUNAKAN ALGORITMA PA-SHORT

Nama : I Wayan Adi Saputra

NIM : 1608561017

Pembimbing : 1. I Komang Ari Mogi, S.Kom., M.Kom.

2. Luh Arida Ayu Rahning Putri, S.Kom., M.Cs.

ASBTRAK

# Kata pengantar

Penelitian dengan judul SIMULASI PROTOKOL MANET DALAM KOMUNIKASI AUDIO PADA DAERAH PEGUNUNGAN MENGGUNAKAN ALGORITMA PA-SHORT ini disusun dalam rangkaian kegiatan pelaksanaan Tugas Akhir di Jurusan Ilmu Komputer FMIPA UNUD. Penelitian ini disusun dengan harapan dapat menjadi pedoman dan arahan dalam melaksanakan penelitian di atas.

Sehubungan dengan telah terselesaikannya penelitian ini, maka diucapkan terima kasih dan penghargaan kepada berbagai pihak yang telah membantu penyusun, antara lain:

1. Bapak I Komang Ari Mogi, S.Kom., M.Kom. sebagai calon Pembimbing I yang telah banyak membantu menyempurnakan penelitian ini;
2. Ibu Luh Arida Ayu Rahning Putri, S.Kom., M.Cs. sebagai calon Pembimbing II yang telah banyak membantu menyempurnakan penelitian ini;
3. Bapak-bapak dan ibu-ibu dosen di Program Studi Teknik Informatika, yang telah meluangkan waktu turut memberikan saran dan masukan dalam penyempurnaan penelitian ini;
4. Kawan-kawan di Program Studi Teknik Informatika yang telah memberikan dukungan moral dalam penyelesaian penelitian ini.

Disadari pula bahwa sudah tentu penelitian tugas akhir ini masih mengandung kelemahan dan kekurangan. Memperhatikan hal ini, maka adanya masukan dan saran – saran penyempurnaan sangat diharapkan.

Bukit Jimbaran,6 September 2019

Penulis

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# DAFTAR LAMPIRAN

# BAB I PENDAHULUAN

1. **Latar Belakang**

Indonesia merupakan merupakan negara berbentuk kepulauan yang tersusun atas beribu-ribu pulau. Secara geografis Indonesia terletak pada wilayah cincin api dunia yang menyebabkan potensi bencana yang sangat besar (Nirwansyah & Nugroho, 2015). Luas daerah Indonesia, menyebabkan Indonesia termasuk dalam daerah yang sangat rawan atas kondisi geografis (Sulistyo, 2016) terhadap bencana, terutama gempa bumi, gunung api (Supriyono, 2014), banjir dan tanah longsor (Supriyono, Citra Sulistyo, & Barchia, 2017a). Kondisi ini juga didukung oleh banyaknya daerah pegunungan di Indonesia, terdapat 30 lebih pegunungan yang terdata di Indonesia. 23 juta penduduk Indonesia adalah masyarakat yang mendiam daerah pegunungan (Sumedi, 2010).

Voice Over Internet Protocols (VoIP) merupakan salah satu teknologi penting yang memungkinkan kita melakukan komunikasi suara melalui koneksi internet. Data menunjukkan bahwa pada per Januari tahun 2019 perkembangan internet di Indonesia mencapai 63% dari total populasi di Indonesia dan menjadi negara dengan perkembangan Internet terbesar per tahun ke-4 (KEMP, 2019). Komunikasi berperan sangat penting dalam kehidupan manusia. Komunikasi digunakan untuk berinteraksi dan menyampaikan informasi. Komunikasi dilakukan dengan beberapa syarat, yakni adanya sumber, tujuan, pesan dan saluran. Perkembangan teknologi mengembangkan proses komunikasi, proses komunikasi pada masa ini, sudah tidak lagi terbatas pada jarak. Perkembangan ini pula menyebabkan banyak pilihan dalam melakukan komunikasi. Salah satu komunikasi yang paling sering dilakukan dengan dukungan teknologi adalah komunikasi audio. Komunikasi audio berkembang menjadi lebih murah dan lebih digunakan, perkembangan komunikasi audio dari menggunakan circuit switching (telepon analog) menjadi packet switching (digital atau VoIP). Kondisi alam dan kurangnya sarana serta prasarana berpengaruh besar terhadap proses evakuasi korban bencana. Pada bencana Gunung Agung tahun 2017, relawan yang berpartisipasi menggunakan komunikasi radio dengan memasang 3 buah unit repeater radio komunikasi. Komunikasi yang dilakukan bertujuan mengedukasi warga terkait bahaya potensi bahaya erupsi pada 28 desa terdampak (Aini, 2017). 28 desa terdampak ini menghandalkan komunikasi radio sehari-hari untuk mengetahui kondisi terkini Gunung Agung. Namun pada tahun 2019, komunikasi radio terancam karena izin dari pemerintah (beritabali, 2019). Pada hasil rapat terakhir, relawan diharuskan untuk membentuk jaringan komunikasi mandiri. Beberapa kasus evakuasi, terdapat permasalahan komunikasi disebabkan karena lokasi bencana di pegunungan atau perbukitan (Simatupang, 2019), dan minimnya jaringan telekomunikasi (Yulistyo, 2012). Hal ini menyebabkan proses evakuasi dan koordinasi antara tim evakuasi terhambat yang menyebabkan terancamnya kondisi terhadap korban bencana yang belum terevakuasi. Perkembangan pada bidang jaringan, dapat membantu pembuatan jaringan mandiri yang dapat mengakomodasi komunikasi audio, teknologi pembuatan jaringan mandiri sering disebut dengan jaringan Ad-Hoc dengan komunikasi audio.

Perkembangan teknologi dewasa ini berjalan dengan sangat pesat. Salah satu hasil dari pengembangan teknologi adalah kemajuan di bidang jaringan dan sensor. Mobile Ad-hoc Network (MANET) merupakan salah satu teknologi perkembangan dari teknologi jaringan komunikasi. MANET memungkinkan perangkan nirkabel memiliki kemampuan untuk berkomunikasi tanpa merujuk pada infrastruktur komunikasi pusat, serta perangkat yang bergerak secara konstan. Berdasarkan kejadian yang telah dipaparkan sebelumnya, diperlukan teknologi komunikasi yang fleksibel dan tidak tergantung dengan jaringan yang terpusat dengan tujuan menghasilkan komunikasi yang dapat diandalkan dalam kondisi seperti kasus diatas.

Sebelumnya, terdapat beberapa penelitian yang pernah dilakukan terkait MANET. Penelitian sebelumnya mengenai peningkatan efisiensi pencarian rute protokol pada MANET menggunakan metode PA-SHORT (Yunia Puspita Wulandari dkk., 2019). Proses pada penelitian ini dilakukan dengan menjalankan kode protokol pada Network Simulator 2, generate berkas trace, dan analisa QoS dengan pemrograman AWK. Perhitungan evaluasi terhadap protokol tergantung dengan QoS yakni throughput, average end-to-end delay, packet delivery ratio, serta routing overhead. Penelitian Yunia Puspita Wulandari dkk menghasilkan performa PA-AOMDV lebih baik daripada AOMDV dengan peningkatan throughput sebanyak 61,64% untuk 50 node dan 45,2% untuk 100 node. Average end-to-end delay juga mengalami penurunan yakni 0,066% pada 50 node dan 0,12% pada 100 node, packet delivery 60,87% pada 50 node 82,02% pada 100 node dan routing overhead berkurang menjadi 67,07% pada 50 node dan 45,36% pada 100 node. Pada penelitian yang penulis akan lakukan menggunakan jenis mobilitas SUMO dan dengan beberapa fitur yang berbeda berdasarkan kondisi evakuasi bencana serta protokol yang berbeda.

Penelitian terkait kedua dilakukan oleh (Ratna, Dewi, Yovita, & Negara, 2018) mengenai algoritma routing MANET-DTN untuk komunikasi pada daerah bencana. Pada penelitian yang dilakukan oleh Dewi menggunakan perbandingan protokol AODV dengan AODV-DTN. DTN atau Delay Tolerant Networks berkeja dengan metode store and forward, hal tersebut mengantisipasi ketika terdapat node yang mati agar pesan dapat tetap diteruskan. Scenario simulasi tergantung pada jumlah node, keceptan simulasi dan ukuran buffer komunikasi. Penelitian dari Dewi meninjau QoS sebagai parameter performa dari protokol routing. Hasil dari penelitian ini diperoleh bahwa protokol AODV-DTN menghasilkan rata-rata QoS yang lebih baik dibandingkan dengan AODV, yakni PDR meningkat sebesar 0,729% namun meningkatkan average delay menjadi 39,39% dan konsumsi energi meningkat 0,161%. Beberapa kekurangan dari penelitian ini adalah konsumsi energi yang meningkat dikarenakan toleransi terhadap gangguan melalui metode DTN, namun memiliki nilai PDR yang tinggi. Pada penelitian yang penulis akan lakukan menggunakan algoritma Path Aware Self-Healing and Optimizing Routing Techniques.

Berdasarkan uraian diatas, akan dilakukan penelitian terkait penerapan MANET dalam komunikasi audio pada daerah dataran tinggi atau pegunungan dengan optimasi menggunakan algoritma Path Aware Self-Healing and Optimizing Routing Techniques. Penelitian dilakukan secara simulasi menggunakan Network Simulator 2, pemodelan lokasi dataran tinggi. Pemodelan lokasi dilakukan dengan menggunakan SUMO (Simulation of Urban Mobility). Untuk komunikasi audio menggunakan pemodelan komunikasi audio pada Network Simulator 2. Pemilihan algoritma PA-SHORT dipertimbangkan karena dapat mengurangi jumlah hop. Tujuan dari algoritma Path Aware SHORT adalah mencoba untuk mempersingkat panjang jalur bila memungkinkan. Rute yang lebih pendek tidak hanya mengurangi latency, tetapi juga meningkatkan throughput.

1. **Rumusan Masalah**

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

* 1. Bagaimana penerapan MANET atau Mobile Ad-hoc Network dalam komunikasi di daerah pegunungan.
  2. Bagaimana kinerja perbandingan dari protokol AODV dan PA-AODV pada komunikasi audio di daerah pegunungan.

1. **Batasan Masalah**

Beberapa Batasan masalah yang akan dijadikan acuan pada penelitian ini sebagai berikut:

1. Simulasi dilakukan terbatas hanya dengan protocol routing pada MANET yakni protokol reaktif (AODV) serta hanya menggunakan algoritma PA-SHORT (Path Aware Self-Healing and Optimizing Routing Techniques).
2. Penelitian akan dilakukan secara simulasi menggunakan program NS2 atau Network Simulator 2.35.
3. Parameter kinerja yang diukur dan diamati adalah berdasarkan MOS dengan E-Model.
4. Model lokasi yang dibangkitkan berdasarkan wilayah Desa Sebudi di Kabupaten Karangasem.
5. Lama waktu trace adalah 120 detik.
6. Model komunikasi audio yang dibangkitkan satu arah.
7. **Tujuan Penelitian**

Berdasarkan rumusan masalah tersebut, tujuan penelitian ini adalah sebagai berikut:

1. Menerapkan metode MANET atau Mobile Ad-hoc Network dalam proses simulasi komunikasi di daerah bencana.
2. Melihat kinerja perbandingan protokol AODV dengan PA-AODV dalam komunikasi audio di daerah pegunungan.
3. **Manfaat Penelitian**

Beberapa manfaat yang dapat diambil dari penelitian ini adalah sebagai berikut:

1. **Metodologi Penelitian**
2. **Desain Penelitian**

Pada penelitian ini menggunakan sebuah simulasi yang merupakan suatu teknik untuk meniru operasi-operasi atau proses-proses yang terjadi dalam suatu sistem dengan bantuan perangkat komputer serta dilandasi dengan beberapa asumsi-asumsi tertentu sehingga sistem tersebut dapat dipelajari secara ilmiah (W. David Kelton, 1991). Pada penelitian ini juga menggunakan metode penelitian eskperimental.

Menurut (Sugiyono, 2010) metode penelitian eksperimen didefinisikan sebagai metode penelitian yang digunakan untuk mencari pengaruh dari perlakukan tertentu terhadap sesuatu dalam kondisi yang dikendalikan. Jadi, melalui penelitian ini dapat disimpulkan bahwa penelitian ini akan mencari hubungan sebab-akibat dari suatu percobaan, dan hasil yang didapatkan akan bervariasi sesuai dengan kondisi yang ditentukan. Maka dari itulah penelitian ini menggunakan metode penelitian *experimental*.

1. **Variabel Penelitian**

Menurut pengertiannya, variable merupakan segala yang berkaitan dengan kondisi, keadaan, factor, perlakuan, atau tindakan yang diperkirakan dapat mempengaruhi hasil penelitian. Variable juga berkaitan langsung untuk mengetahui suati keadaan tertentu dan diharapkan mendapatkan dampat/akibat dari eksperimen sering disebut variable eksperimental (*treatment variable*).

Dalam penelitian ini terdapat variable bebas dan variable terikat, dimana dalam penelitian ini, variable tersebut dijabarkan sebagai berikut:

1. Variable Bebas

Merupakan variable yang mempengaruhi hasil dari variable lain jika variable ini berubah,xxxxxxxxxx

1. Variable terikat

Variable terikat dapat didefinisikan sebagai hasil yang didapatkan dalam penelitian yang dipengaruhi oleh variable bebas. Pada penelitian ini, variable terikat antara lain:

* 1. Delay
  2. Packet Loss
  3. MOS E-Model

1. **Skenario Tahapan Penelitian**

Gambar dibawah menunjukkan bagaimana tahapan-tahapan dalam penelitian. Pada gambar, terdapat alur yang memperlihatkan tahapan dari penelitian, penjelasan mengenai alur sebagai berikut:

1. Pengumpulan data

Tahap ini merupakan proses mengumpulkan data-data dan referensi yang mendukung serta yang dibutuhkan dalam penelitian yang akan dilakukan.

1. Instalasi *Software* Simulasi

Pada tahap ini akan dilakukan proses instalasi *software* simulasi yang digunakan dalam proses penelitian, dimana *software* simulasi yang digunakan adalah Network Simulator 2.35.

1. Memodelkan Lokasi Pegunungan SUMO

Tahap ini memodelkan lokasi pegunungan melalui *openstreetmap* dan SUMO (Simulation of Urban Mobility) sehingga menghasilkan berkas *mobility* yang digunakan pada Network Simulator 2.35.

1. Memodelkan Protokol AODV dan PA-AODV pada Simulator

Pada tahapan ini mulai dilakukan penerapan protokol *routing* AODV dan PA-AODV pada Network Simulator 2.35.

1. Penerapan *Traffic* Komunikasi *Audio* dengan Codec G711 dan G723.1

Tahap ini akan diterapkan *traffic* komunikasi *audio* pada penelitian. Pembangkitan *traffic audio* menggunakan patch Ns2Voip++ dengan Codec G711 dan G723.1 sebagai pembanding, dan decoder H.263.

1. Pembuatan Kode Program

Pada tahapan ini merupakan tahapan pembuatan kode program untuk Protokol *Routing* PA-AODV, kode scenario simulasi, kode pengujian seperti *Delay* dan *Packet Loss* serta kode program perhitungan MOS E-Model.

1. Pengumpulan Data Simulasi

Tahap ini merupakan tahapan pengumpulan data-data hasil simulasi. Pengumpulan data ini akan dilakukan untuk semua scenario yang telah ditentukan.

1. Analisis

Data hasil simulasi yang telah dikumpulkan tersebut akan diamati bagaimana pengaruh variable yang diteliti dengan kualitas komunikasi yang dihasilkan. Data hasil simulasi akan dicatat dan dituangkan kedalam bentuk tabel dan grafik. Dengan membandingkan komunikasi audio menggunakan protokol *routing* AODV dan PA-AODV serta pengaruh kualitas komunikasi berdasarkan Codec G711 dan G723.1.

1. Penarikan kesimpulan dan saran

Setelah dilakukan analisis data yang diperoleh dari proses simulasi, kemudian akan disimpulkan berdasarkan nilai-nilai yang diperoleh dari analisis data simulasi yang telah dilakukan pada penelitian.



1. **Skenario Pengujian**

Pada penelitian ini dilakukan beberapa scenario pengujian untuk mengetahui kualitas komunikasi *audio* dengan menggunakan protokol routing AODV dan PA-AODV serta pengaruh codec dan jumlah node yang terlibat dalam proses simulasi.

Tabelin aja dulu gan

|  |  |
| --- | --- |
| Node | 25, 35, 45, 55, 65, 75 |
| Codec | G711, G723.1 |
| Protokol Routing | AODV, PA-AODV |

# BAB II TINJAUAN PUSTAKA

Pada bagian ini, akan dipaparkan sejumlah tinjauan empiris dan tinjauan teoritis yang dijadikan acuan dalam penelitian menyenai simulasi protokol MANET dalam komunikasi audio di daerah pegunungan yang dilakukan ini.

1. **Tinjauan Empiris**

Penelitian terdahulu yang berkaitan dengan penelitian yang penulis baik dari segi metode maupun jenis penelitian yang dilakukan, akan diperjelas pada tinjauan empiris. Beberapa penelitian terdahulu yang berkaitan yaitu:

1. ***Routing Protocol for MANET: A Literature Survey*** (Muralishankar et al, 2014)

Mobile Ad-hoc Network (MANET) merupakan jaringan yang jaringan yang dapat dibuat tanpa dukungan infrastruktur jaringan. Pada jaringan MANET, node bergerak dengan cepat. Batasan transmisi pada MANET menyebabkan rentangan komunikasi pada setiap node tidak selalu dapat dilakukan secara langsung, sehingga pada MANET, setiap node dapat berperan sebagai router atau bersifat multihop.

1. **Meningkatkan Efisiensi Rute Pada Protokol Routing AOMDV Menggunakan Metode PA-SHORT di Jaringan MANET** (Wulandari, Y. P., Jatmika, A. H., & Bimantoro, F., 2019)

Penelitian ini dilatarbelakangi oleh permasalahan routing pada jaringan MANET, dimana penemuan jalur paling efektif antara node yang terus bergerak. Maka diperlukan protokol routing yang dapat menangani dengan baik segala perubahan tersebut. Wulandari memilih menggunakan AOMDV atau Ad-hoc On-demand Multipath Distance Vector karena AOMDV ketika mengalami kerusakan dalam proses pencarian rute, maka akan menggunakan rute cadangan yang dimilikinya tanpa harus mencari lagi. Selain itu AOMDV menyediakan dua layanan yaitu route discovery dan route maintenance, AOMDV berkeja mencari rute dengan cara memperhitungkan jarak berdasarkan jumlah hop. Perubahan topologi dapat dimanfaatkan untuk mengurangi jumlah hop. Pengurangan jumlah hop dapat meningkatkan throughput dan menurunkan delay. Algoritma Path Aware SHORT digunakan untuk mempersingkat hop sehingga diharapkan dapat meningkatkan QoS.

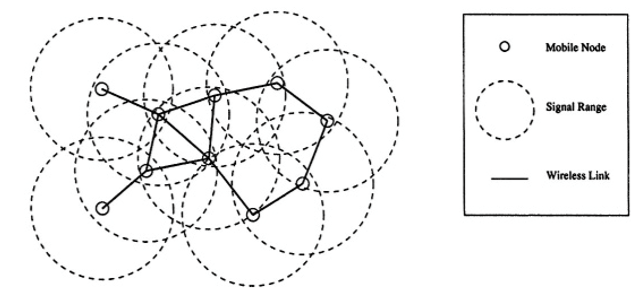
Pada penelitian Wulandari dkk ini mengefisiensikan pencarian rute pada protokol AOMDV menggunakan PA-SHORT sehingga menjadi PA-AOMDV. Menggunakan algoritma Path Aware SHORT mengurangi jumlah hops pada rute dan meningkatkan optimasi dengan monitoring rute serta update mengenai rute terbaik. Pada penelitian ini menggunakan jumlah node sebanyak 50 dan 100 node pada luas area simulasi 500m2 x 500 m2 dan 1000m2 x 1000m2, waktu simulasi 600 detik dengan jenis mobilitas Random Way Point serta model propagasi Two Ray Ground. Penelitian ini menggunakan Network Simulator 2 dengan menggunakan bahasa TCL (Tool Command Language) dalam pembuatan kode AOMDV dan PA-AOMDV. Pada penelitian ini dihasilkan kesimpulan yaitu PA-AOMDV menghasilkan peningkatan sebesar 16.04% throughput pada 50 node dan 11,55% untuk 100 node dengan aera 500 x 500 m2 dan peningkatan sebesar 61,84% pada 50 node dan 45,2% untuk 100 node untuk luas area 1000 x 1000 m2. QoS delay mengalami penurunan pada PA-AOMDV serta peningkatan PDR dan penurunan routing overhead. Kinerja protokol PA-AOMDV meningkatkan optimalisasi routing dengan memantau rute secara bertahap mengalihkan jalur setiap kali short-cut ditemukan.

1. **Analisis Kualitas VoIP pada SCTP Menggunakan ECN dan AQM**
2. ***Performance Analysis of VoIP in Wireless Networks*** ()
3. ***VOIP APPLICATIONS OVER MANET: CODEC PERFORMANCE ENHANCEMENT BY TUNING ROUTING PROTOCOL PARAMETERS***
4. **Tinjauan Teoritis**

Beberapa teori yang dijadikan acuan pada penelitian yang penulis lakukan adalah sebagai berikut:

1. ***Mobile Ad-Hoc Network* (MANET)**

Definisi *Ad-*(Bello, 2013)*Hoc Network* adalah desentralisasi dari jaringan nirkabel*,* disebut dengan *Ad-Hoc­* karena tidak bergantung pada infrastruktur yang sudah ada seperti *router* dalam jaringan kabel maupun *access point* pada jaringan nirkabel. Setiap *node* dapat bertugas untuk menrutekan data kepada *node* lainnya, jadi penentuan *node* yang mana mengirimkan data dibuat secara dinamis berdasarkan konektivitas pada jaringan itu sendiri (Nishani & Biba, 2016). Namun tidak benar-benar dapat dikatakan bahwa MANET tidak dapat memiliki infrastruktur jaringan. Beberapa mediasi antar *node* dapat dipilih untuk bertindak sebagai *base station* dan *node* tetangga. Penerapan MANET sangat diperlukan di daerah seperti jasa penyelamatan darurat, medan perang dan tempat lainnya dimana penyebaran infrastruktur jaringan menjadi sangat sulit. Dalam MANET, setiap *mobile node* bertindak sebagai router serta *base station* untuk menemukan dan mempertahankan rut eke *mobile node* lainnnya untuk berkomunikasi melalui jaringan (Bello, 2013).



Gambar 1. Mobile Ad-Hoc Network  
Sumber: (Gui & Mohapatra, 2008)

Pergerakan dari *node* dapat menyebabkan perubahan topologi yang tidak bisa diprediksikan, karena itu tugas untuk menemukan dan mempertahankan rute merupakan tugas penting dari protokol penemuan rute MANET. Parameter yang mempengaruhi kinerja protokol penentuan rute jaringan MANET adalah beberapa *node* di dalam area, perilaku pergerakan *node* yang mengubah konektivitas (bergerak secara bebas) dan kualitas rute. Penggunaan MANET memberikan beberapa keuntungan, diantaranya adalah (Purba, Primananda, & Amron, 2018):

1. Tidak memerlukan dukungan *backbone* infrastruktur, hal ini menyebabkan lebih mudahnya untuk diimplementasikan dan sangat berguna ketika infrastruktur tidak berfungsi bahkan tidak ada.
2. *Node* yang selalu bergerak atau dinamis dapat mengakses informasi sevara *real-time* ketika berhubungan dengan *node* lainnya. Sehingga pertukaran data dapat menjadi acuan dalam pengambilan keputusan dan segera dilakukan.
3. Fleksibel, karena jaringan bersifat sementara.
4. Dapat dikonfigurasikan dalam bermacam-macam topologi, baik dengan jumlah *node* kecil hingga besar sesuai dengan aplikasi dan instalasi.

Sedangkan kekurangan dari penggunaan MANET sendiri adalah sebagai berikut:

1. Packet loss akan terjadi bila transmisi mengalami kesalahan.
2. Memungkinkan seringnya terjadi konektivitas yang terputus karena tidak selalu berada dalam cakupan area (*network area*).
3. *Bandwidth* komunikasi terbatas.
4. *Lifetime* baterai yang singkat.
5. **Karakteristik *Mobile Ad-Hoc Network***

Berikut ini merupakan karakteristik dari jaringan *Mobile Ad-Hoc Network* (MANET) adalah sebagai berikut (Pathan, 2016):

1. *Mobility*: *node* dapat bergerak cepat dengan penebaran di daerah yang tdak memiliki infrastruktur. Pada MANET dapat memiliki individual random mobility, group mobility, bergerak sepanjang rute yang telah direncanakan sebelumnya dan lain sebagainya. Mobilitas memiliki dampak besar pada pemilihan skema routing dan dapat mempengaruhi kinerja.
2. *Multihopping*: jaringan multihop adalah jaringan di mana jalur dari sumber ke tujuan melintasi beberapa node lainnya. Jaring ad hoc sering menunjukkan banyak hop untuk negosiasi hambatan, penggunaan kembali spektrum, dan konservasi energi.
3. Self-organisasi: jaringan ad hoc secara mandiri dapat menentukan parameter konfigurasi sendiri termasuk: pengalamatan, routing, clustering, identifikasi posisi, kontrol daya, dan lain-lain. Dalam beberapa kasus, node khusus misalnya, mobile backbone nodes dapat mengkoordinasikan gerakan dinamis dan mendistribusikan di wilayah geografis untuk menyediakan cakupan dari daerah yang terputus.
4. ***Ad Hoc On Demand Distance Vector (AODV)***

*Ad-Hoc On Demand Distance Vector* (AODV) merupakan protokol standar IETF (RFC2561), dan protokol routing berbasis topologi yang paling banyak digunakan. AODV adalah protokol routing reaktif yang akan mempelajari topologi berdasarkan pesan yang diteria dari tetangganya yang mengirimkan *Hello Message* secara berkala (Mantoro & Reza, 2017).

Berikut ini merupakan karakteristik dari protokol routing AODV adalah sebaga berikut:

1. Hanya akan mencari rut eke tujuan jika diperlukan.
2. Menggunakan *sequence number* untuk menemukan informasi yang akurat.
3. Hanya menyimpan alamat tujuan berikutnya pada tabel routing.
4. Menggunakan *broadcast Hello Message* untuk menemukan tetangga terdekatnya.

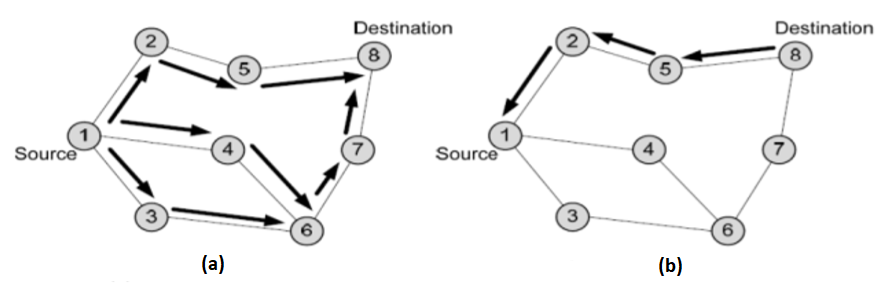
Pada proses pencarian rute, AODV menggunakan 3 buah pesan untuk mengontrol proses pencarian rute sebagai berikut:

1. RREQ: pesan RREQ mengandung id, IP tujuan, *sequence number* IP tujuan, IP sumber dan *sequence number* IP sumber, yang akan disiarkan ke tetangga dan meneruskannya dengan simpul di antaranya.
2. RREP: pesan *unicast* yang diteruskan oleh IP tujuan yang menerima RREQ ke sumber yang mengirim RREQ. Pesan ini mengandung jumlah *hop,* IP sumber, IP rujuan dan *sequence number* IP sumber dan tujuan.
3. RERR: pesan ini akan dikirimkan jika terdapat kesalahan perutean. Informasi yang terkandung dalam pesan ini adalah IP tujuan yang tidak dapat dijangkau dengan *sequence number*nya.

Proses routing atau pencarian rute pada protokol AODV diawali dengan proses *Route Discovery* kemudian setelah rute ditemukan, akan dilakukan proses *Route Maintenance* dan *Routing Table Management*, sebagai berikut (Mantoro & Reza, 2017):

1. ***Route Discovery***

Ketika *node* sumber tidak memiliki informasi perutean untuk ke *node* tujuan dalam tabel peruteannya, *node* sumber akan mulai melakukan proses penemuan rute dengan menyebarkan *Broadcast* RREQ (gambar 1a). Jika ada *node* tetangga yang mendapatkan pesan ini dan memiliki rute ke *node* tujuan, makaa akan membalas dengan paket balasan rute (gambar 1b); jika tidak, *node* tetangga akan menyiarkan ulang paket RREQ hingga mencapai tujuannya. Ketika pesan RREP diterima, maka rute akan dibuat. RREP dapat menerima banyak *node* sumber dengan rute yang berbeda ke *node* tujuan.



Gambar 2. (a) Proses broadcast RREQ (b) Pengiriman pesan RREP ke node sumber.  
Sumber: (Mantoro & Reza, 2017)

1. ***Route Maintenance***

Ketika rute terputus, proses pemeliharaan rute akan dimulai. Proses pemeliharaan rute akan dilakukan jika *node* tidak lagi menerima pesan Helllo dari tetangganya dalam jumlah waktu tertentu dari proses pemanggilan fungsi HELLO. *Node* akan ditandai sebagai rute yang tidak valid dan pesan RERR akan dikirim untuk memberi tahu *node* lainnya tentang informasi kerusakan rute tersebut.

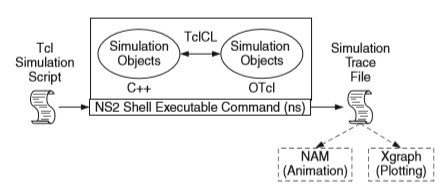
1. ***Routing Table Management***

Manajemen tabel perutean di AODV diperlukan untuk memperbarui informasi tentang tabel perutean AODV. Mengelola informasi tabel perutean di AODV ditangan dengan *sequence number node* tujuan dan salah satu kriteria terpenting adalah harus bebas dari *looping.*

***Isikan proses aodvnya***

1. ***Network Simulator 2 (NS2)***

Network Simulator 2 (versi 2.35) merupakan sebuah alat simulasi event-driven yang berguna untuk mempelajari sifat dinamis dari komunikasi pada jaringan. Simulasi wired maupun wireless serta protokol (misalnya algoritma routing, TCP/UDP) dapat dilakukan menggunakan NS2. NS2 memberikan kebebasan untuk menentukan jenis komunikasi, protokol jaringan dan simulasi yang sesuai.



Gambar 3. Arsitektur Network Simulation 2.  
Sumber: (Issariyakul, Teerawat & Introduction, 2013)

NS2 dapat dijalankan oleh pengguna dengan mengekseskusi “ns” yang mengambil satu argument input, nama dari file Tcl. Dalam kebanyakan kasus, simulasi file Tcl dibuat dan digunakan untuk memplot grafik dan animasi dari komunikasi jaringan.

NS2 menggunakan dua bahasa utama yakni C++ dan Object Tool Command Language (OTCL). C++ berfungsi untuk mendefinisikan backend dari simulasi, OTCL membentuk simulasi dengan membuat dan mengkonfigurasi objek serta penjadwalan diskrit dari event yakni antarmuka. C++ dan OTCL dihubungkan menggunakan TclCL. C++ dan OTCL dipetakan ke objek C++ dan variable dalam domain OTCL yang disebut dengan handles. Handles merupakan sebuah string (misalnya “\_o10”) dalam domain OTCL dan tidak mengandung fungsi apapun. Sedangkan pada C++ fungsi untuk menerima paket didefinisikan dengan class “connector”. Pada domain OTCL handle berfungsi sebagai antarmuka yang berinteraksi dengan pengguna dan objek OTCL lainnya.

Dalam NS2 dapat dibangun banyak class dengan C++. Class pada C++ ini dapat digunakan untuk mengatur simulasi melalui script simulasi Tcl. Setelah simulasi melalui NS2, output yang dihasilkan berbasis teks, untuk merepresentasikan hasil grafis dan interaktif, digunakan alat seperi NAM (Network Animator) dan XGraph (Issariyakul, Teerawat & Introduction, 2013).

1. **Komunikasi *Audio***
2. **Daerah Pegunungan**
3. ***Simulation of Urban Mobility (SUMO)***

SUMO merupakan simulator lalu lintas yang bersifat *open source* dikembangkan di *Institute of Transportation System* Jerman. SUMO adalah simulasi lalu lintas *amicroscopic* yang berarti setiap kendaraan dan pedestrian dimodelkan secara eksplisit (Michael Behrisch, Krajzewicz, & Weber, 2013). Fitur utama dari SUMO termasuk pergerakan kendaraan bebas tabrakann, perbedaan tipe kendaraan, multi jalur dan lain-lain. Dengan mengkombinasikan SUMO dengan *Open Street Map* (OSM) dapat dilakukan simulasi dengan lokasi yang diinginkan sesuai keadaan di dunia nyata. Karena SUMO merupakan murni *traffic generator*, jalur yang dihasilkan ini tidak dapat langsung digunakan pada simulator jaringan seperti NS2 karena memerlukan representasi *road network* (Pradana, Negara, & Dewanta, 2017).

Untuk menghasilkan *road network* atau jalur di SUMO, menggunakan tools pada SUMO seperti *netconvert, plyconvert,* dan *randomTrips.py*. Berikut ini penjelasan dari masing-masing *tools* yang digunakan pada penelitian (Deshmukh & Dorle, 2016):

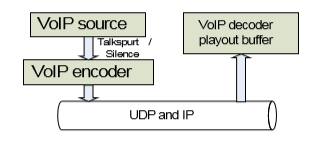
|  |  |
| --- | --- |
| Tools | Fungsi |
| Netconvert | Ekstrak data jaringan dari map OSM dan menghasilkan jaringan jalan yang digunakan di SUMO. Fungsi ini mengidentifikasi *node*, persimpangan dan lainnya. Juga digunakan untuk membangun file *network* pada SUMO |
| Polyconvert | Impor bentuk geometris (polygon - bangunan) dari berbagai sumber dan mengonversinya menjadi representasi yang divisualisasikan dalam SUMO-GUI. |
| randomTrips.py | Digunakan untuk membangkitkan *node*, termasuk posisi dan pergerakan *node* pada *edges.* |
| traceExporter.py | Digunakan untuk menghasilkan *file* yang dapat digunakan diluar ekosistem SUMO. Beberapa opsi memberikan pilihan untuk menghasilkan *file* pergerakan untuk Network Simulator 2/3. |

1. ***Open Street Map* (OSM)**

*Open Street Map* (OSM)adalah proyek *open source* untuk membuat contoh peta dunia yang dapat diubah secara bebas oleh siapa saja. Dua factor pendukung dalam OSM untuk digunakan dan dikembangan adalah kurangnya ketersediaan informasi peta pada sebagian besar wilayah didunia dan gabungan dari alat navigasi yang terjangkau. OSM adalah contoh utama dari informasi geografis yang ditetapkan secara bebas. Yang paling penting adalah peta OSM dapat disimpan di internet dan siapa saja dapat mengkases peta kapan saja secara gratis (Dian Khumara, Fauziyyah, & Kristalina, 2019).

OSM digambarkan sebagai “*Map of Wikipedia*” yang tidak dapat dipisahkan dari ketersediaan mekanisme dimana sukarelawan atau siapa pun dapat berkontribusi secara langsung untuk mengubah atau memperbarui data geografis untuk membuat peta yang lebih akurat, terperinci, dan terkini. Karena keterbukaan akses ini, peta OSM dapat digunakan sebagai pemodelan lokasi pada proses penelitian. Pemodelan lokasi dilakukan dengan mengekstrak peta OSM menghasilkan file berformat osm, kemudian diproses menggunakan fungsi dari SUMO, sehingga menghasilkan file input yang akan menjadi acuan lokasi pada penelitian.

1. ***NS2Voip++***

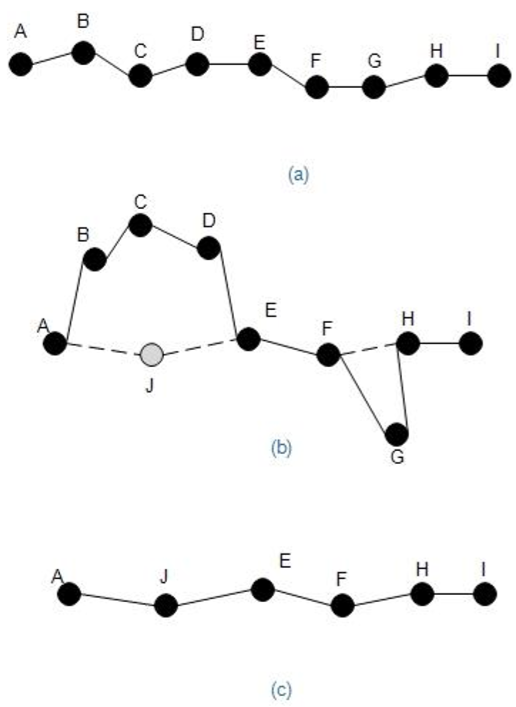
NS2Voip++ merupakan modul pembentukan *traffic* pada Network Simulator 2.35. Modul ini dikembangkan oleh University of Pisa di Italy menyempurnakan modul sebelumnya, yakni Ns2Voip (Andreozzi, Migliorini, Stea, & Vallati, 2010). NS2Voip++ dikembangkan dalam bahasa C++, NS2Voip++ dapat melakukan pembentukan *traffic* komunikasi *audio* dengan jenis codec yang berbeda dan model VAD (Voice Activity Detection). Jenis codec yang tersedia pada NS2Voip++ meliputi G.711, G.723.1, G.729A dan GSM.AMR. VAD (Voice Activity Detection) pada NS2Voip++ berupa *one-to-one, exponential,* dan *Weibull-custom.* VAD (Voice Activity Detection) meliputi periode *talk* dan *silence*.

Sumber gambar (Andreozzi et al., 2010)

*Voip Source* merupakan model penggunaan komunikasi *audio*. *Voip Source* meliputi pengaturan periode *talkspurt* dan *silence.* Ketika proses *talkspurt* *encoder* akan menghasilkan data sesuai dengan codec yang dipilih meliputi besaran paket dan frekuensi pembuatan paket. Di sisi penerima, *decoder* VoIP menerima paket dari jaringan dan menggunakan *buffer playout* untuk mempercepat *playout* mereka (Andreozzi et al., 2010).

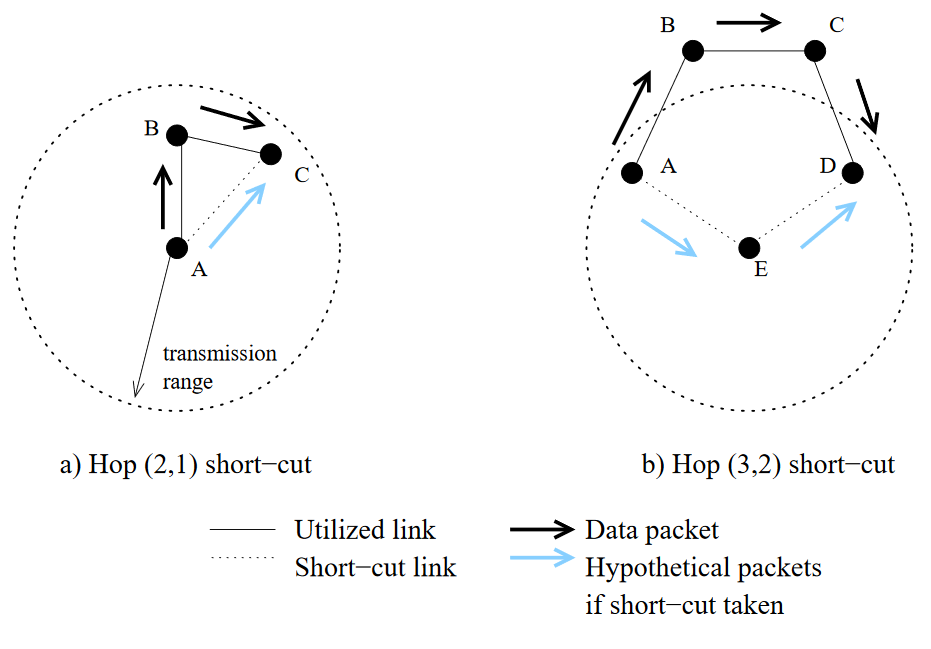
Pada versi terbaru NS2Voip++, pengembang menambahkan *buffer playout* yang lebih realistis dan terkenal, yakni H323 dan eEM (Causal) (Andreozzi et al., 2010). H323 dikenal sebagai *buffer playout* dinamis dan digunakan dalam berbagai perangkat lunak yang terkenal seperti *open source* VoIP Client Ekiga. Dengan mengimport kode asli codec H323 dari aplikasi Ekiga ke dalam modul NS2Voip++.

1. ***Path Aware-SHORT***

Bentuk jalur rute dapat berubah secara signifikan ketika konektivitas utuh karena pergerakan dari *node*. Perubahan bentuk ini dapat menimbulkan jalur yang lebih baik ataupun sebaliknya. Perubahan bentuk dapat dieksploitasi untuk mendapatkan jalur-jalur pengaturan rute yang lebih baik jika dapat menghindari setiap *overhead* yang signifikan (setidaknya menghindari proses-proses penemuan rute ekstra).

Sumber gambar: (Fahriani, Djanali, & Shiddiqi, 2012)

Gambar 6.7 menunjukkan perubahan rute, Gambar xx (a) menunjukkan jalur awal yang ditentukan melalui proses penemuan jalur, dimana jarak antara sumber dan tujuan adalah jarak terpendek menurut jumlah hop, maka sesuai jalur awal, node A membutuhkan 8 hop untuk menuju ke node I. MANET selama perjalanan waktu, mobilitas node dapat membuat jalur rute menjadi berubah, perubahan jalur diperlihatkan pada Gambar xx (b) dengan bentuk baru, node J ada dalam tingkatan transmisi node F, idealnya, jalur terpendek dari A ke node I hanya membutuhkan hop seperti pada Gambar xx (c).

Tujuan utama dari penggunaan PA-SHORT adalah untuk menemukan rute pintas ketika diperlukan atau ketika terjadi perubaha topologi. Scenario dasar dari proses penemuan rute pintas ditujukkan pada Gambar xx. Pada Gambar xx (a) , jaluar A-B-C dapat direduksi menjadi A-C karena C berada dalam jangkauan transmisi *node* A. Pembentukkan jalur pintas tersebut disebut reduksi (2,1). Gambar xx (b) menunjukkan bahwa jalur perutean A-B-C-D dapat disingkat menjadi A-E-D karena E berada dalam jangkauan transmisi A, dan D berada dalam jangkauan transmisi E. Pembentukkan jalur pintas ini disebut reduksi (3,2). Dengan demikian, reduksi (n,2) menyiratkan bahwa n *hop* disepanjang penemuan rute dapat dikurangi hanya menjadi 2 *hop*. Secara umum reduksi (n,k) menyiratkan bahwa n *hop* dapat direduksi menjadi k *hop*, dimana k<n. Untuk menghindari kompleksitas, maka nilai k menjadi 1 atau 2 pada algoritma PA-SHORT. Semakin tinggi perbedaan antara n dan k, maka semakin baik kinerja dari PA-SHORT (Gui & Mohapatra, 2008).

Sumber gambar: (Gui & Mohapatra, 2008)

1. **MOS E-Model**

*Mean Opinion Score* (MOS) merupakan satuan kualitas suara yang biasanya digunakan untuk mengukur kualitas suara yang dihasilkan dari proses komunikasi dalam jaringan. Terdapat 2 standar yang digunakan dalam menentukan kualitas yakni ITU-T p.800 dan E-Model ITU-T G.107. Pada standar ITU-T P.800 kualitas ditentukan secara subjektif menggunakan pendapat orang-perorang yang pengujiannya dapat menggunakan *conversation opinion test* dan *listening test*. Metode ini dirasa kurang efektif untuk menentukan kualitas suara untuk komunikasi audio karena ketidaktepanan nilai dan standar orang yang berbeda-beda (Surya, 2011).

Kemudian ITU menciptakan standar pengukuran, sehingga MOS dapat dilakukan sebagai pengujian yang bersifat objektif dengan menggunakan E-Model yang menghasilkan nilai R factor. Standar baru tersebut tertuang pada ITU-T.G.107. E-Model memetakan metrik jaringan ke estimasi nilai MOS. E-Model menghitung nilai R factor berkisar 0 hingga 100. Variable yang dipertimbangkan dalam pengujian khususnya komunikasi audio adalah Id dan Ie (). R factor diitung dengan rumusan sebagai berikut:

Variabel R pada persamaan X merupakan factor kualitas transmisi. Id merupakan factor penurunan kualitas yang disebabkan oleh delay atau d. Nilai Id dijabarkan pada persamaan berikut:

Variable H merupakan fungsi heavyside ditunjukkan pada persamaan X. Sedangkan Ie merupakan factor penurunan kualitas yang diakibatkan presentasi kehilangan paket. Nilai Ie dijabarkan pada persamaan X.

Setelah nilai R factor ditemukan, maka nilai tersebut akan diestimasikan kedalam MOS (ITU-T.P.800). Hasil estimasi akan dikorelasikan kedalam tingkat kualitas komunikasi. Untuk mengubah R factor kedalam MOS (ITU-T.P.800) menggunakan ketentuan berdasarkan rentang nilai R factor yang ditunjukkan pada persamaan berikut.

Tabel x Mapping MOS dan E-Model (Mohd & Ong, 2007)

|  |  |  |
| --- | --- | --- |
| R | User Satisfaction | MOS |
| 90-100 | Very Satisfied | 4.3-4.5 |
| 80-90 | Satisfied | 4.0-4.3 |
| 70-80 | Some user dissatisfied | 3.6-4.0 |
| 60-70 | Many users dissatisfied | 3.1-3.6 |
| 50-60 | Nearly all user dissatisfied | 2.6-3.1 |
| 0-50 | Not recommended | 1-2.6 |

Untuk mendapatkan perhitunga *Id* dan *Ie* maka diperlukan nilai dari *delay* dan *packet loss* dari komunikasi yang dilakukan. Semakin kecil nilai dari *delay* dan *packet loss* memungkinkan semakin tinggi nilai R yang didapatkan. Melalui nilai *delay* dan *packet loss* juga dapat memperlihatkan kondisi dari kualitas komunikasi yang dilakukan.

**Isikan keterkaitan dengan performance metric**

1. ***End to End Delay***

End-to-end delay merupakan waktu yang dibutuhkan suatu paket ketika dikirim oleh node sumber sampai berhasil diterima oleh node tujuan, rumus seperti pada persamaan x.

(2)

Dimana :

Tr = Waktu penerimaan paket (detik)

Ts = Waktu pengiriman paket(detik)

Berdasarkan kategori, delay dapat diklasifikasikan sebagai berikut:

|  |  |  |
| --- | --- | --- |
| **Kategori delay** | **Besar *delay*** | **Indeks** |
| Sangat bagus | < 150 ms | 4 |
| Bagus | 150 – 300 ms | 3 |
| Sedang | 300 – 450 ms | 2 |
| Jelek | > 450 ms | 1 |

1. ***Packet Loss***

Suatu parameter yang menunjukkan kondisi jumlah total paket yang hilang karena collision dan congestion pada jaringan. Di dalam jaringan, nilai packet loss ini diharapkan memiliki nilai yang minimum. Berikut merupakan rumusan untuk mencari nilai packet loss:

()

*Packet loss* dapat dikategorikan dalam rentangan tabel sebagai berikut:

|  |  |  |
| --- | --- | --- |
| **Kategori Degredesi** | **Paket Loss** | **Indeks** |
| Sangat Bagus | 1 % | 4 |
| Bagus | 3 % | 3 |
| Sedang | 15 % | 2 |
| Jelek | 25 % | 1 |

# BAB III ANALISIS DAN PERANCANGAN SISTEM

1. **Tahapan Analisis**

Dalam pengerjaan penelitian tugas akhir ini dilakukan dalam sebuah simulasi, simulasi pada penelitian ini menggunakan *tools* simulator Network Simulator 2.35 atau NS-2.35. Dalam proses penelitian ini, akan diimplementasikan protokol routing AODV (*Ad hoc On-Demand Distance Vector*) dan penggunaan algoritma Path Aware SHORT pada protokol AODV sehingga menghasilkan protokol *routing* baru, yakni PA-AODV (*Path Aware-Ad hoc On-Demand Distance Vector*). Kedua protokol *routing* tersebut akan digunakan pada simulasi MANET atau Mobile Ad-Hoc Network.

*Node* yang berkomunikasi pada simulasi yang dilakukan akan bergerak berdasarkan kondisi geografis dari pegunungan yang ditentukan. Hal ini dapat dicapai dengan menggunakan data map geografis dari *OpenStreetMap* dan dikonversi menjadi posisi dan pergerakan *node* menggunakan SUMO (*Simulation of Urban Mobility*). Posisi dan rute pergerakan *node* dibuat secara acak mengikuti *egde* yang terbangun dari data map *OpenStreetMap.* Kecepatan dibuat dengan rentangan yang disesuaikan dengan kecepatan pergerakan manusia. Komunikasi yang terjadi antar *node* akan melibatkan satu *node* sumber dengan satu *node* tujuan. Besaran paket komunikasi yang dikirimkan akan tergantung dengan codec yang ditentukan serta waktu paket dikirimkan akan ditentukan melalui parameter simulasi. Jenis paket komunikasi yang digunakan ada paket UDP (*User Datagram Protocol*).

Dalam penelitian ini menggunakan beberapa parameter simulasi, berikut merupakan parameter simulasi yang digunakan:

|  |  |
| --- | --- |
| **Parameter** | **Nilai** |
| Tipe Kanal | *Wireless Channel* |
| Model Propagasi | *TwoRayGround* |
| Tipe *Network Interface* | *WirelessPhy* |
| Tipe MAC | IEEE 802.11 |
| *Max* Paket dalam Antrian | 50 |
| Tipe Paket | UDP |
| Mobilitas *Node* | SUMO |
| Kecepatan *Node* | 0 m/s sampai 7 m/s |

Dari tabel diatas, terdapat beberapa nilai *default* untuk parameter Tipe Kanal, Model Propagasi, Network Interface, Tipe MAC dan *Max* paket dalam antrian.

1. **Perancangan Simulasi**

Dalam penelitian ini terdapat beberapa scenario yang akan dijadikan acuan dalam pengujian dan analisis data hasil simulasi. Scenario diberikan dalam simulasi dengan tujuan untuk mengukur kualitas komunikasi *audio* pada simulasi yang dipenaruhi oleh variable yang berubah ubah. Berikut merupakan scenario yang digunakan dalam proses simulasi:

|  |  |
| --- | --- |
| **Parameter** | **Nilai** |
| Luas Area | 4000m2 |
| Waktu Simulasi | 120 detik |
| Lama Waktu Komunikasi *Audio* | 60 detik |
| Jumlah *Node* | 25, 35, 45, 55, 65, 75 |
| Protokol *Routing* | AODV, PA-AODV |
| Codec *Audio* | G.711, G.723.1 |

Simulasi dilakukan dengan luas wilayah simulasi 4000m2. Setiap *node*, baik itu *node* sumber, *node* penerima dan *node* perantara diletakkan secara acak dalam *edge* dan rentangan luas wilayah tersebut. *Node* sumber merupakan node yang mengirimkan paket, *node* pengantar merupakan *node* yang digunakan untuk meneruskan paket hingga menuju ke *node* tujuan. Sedangkan *node* tujuan merupakan *node* yang menerima paket dari *node* sumber. Jumlah node yang digunakan dalam simulasi adalah 25, 35, 45, 55, 65 dan 75 node. Hal ini dilakukan untuk mendapatkan nilai dari penambahan jumlah node serta acuan dan perbandingan dalam menentukan jumlah *node* yang sesuai ketika diimplementasikan dalam luas wilayah simulasi tertentu.

Setiap *node* diletakkan secara acak pada *edges* serta mobilitas *node* dibuat mengikuti *edges* secara acak dengan kecepatan dalam rentangan 0 m/s hingga 7 m/s. Kondisi *node* yang diletakkan secara acak pada *edges* serta mobilitas *node* dibuat menggunakan randomTrips.py pada SUMO, dimana randomTrips.py ini akan membangkitkan *node* pada *edges* dan memberikan mobilitas pada *nodes* dengan menelusuri *edges* pada data lokasi *OpenStreetMap*. Hasil randomTrips.py akan menghasilkan *file* dengan format xml. *File* xml ini merupakan bagian dari file konfigurasi sumo dengan format sumo.cfg, *file* berformat sumo.cfg ini mengandung parameter lain dari *file net-file, route-files dan additional-files*. Berikut merupakan isi dari *file* konfigurasi sumo.cfg.

<configuration>

<input>

<net-file value='"$HOME/$proj/$netfiles"'/>

<route-files value='"$HOME/$proj/$rout/$i.rou.xml"'/>

<additional-files value='"$HOME/$proj/$poly/$i.poly.xml"'/>

</input>

<time>

<begin value='"0"'/>

<end value='"120"'/>

<step-length value='"0.1"'/>

</time>

</configuration>

Untuk menghasilkan *file* mobilitas untuk Network Simulator 2.35 melalui SUMO, maka setelah *file* konfigurasi SUMO dibuat, maka proses selanjutnya adalah menghasilkan *file* output dari *file* konfigurasi. Melalui perintah sumo dan dengan parameter *file* konfigurasi maka didapatkan *file* output berformat \*.sumo.xml. Format perintah sumo sebagai berikut:

sumo -c <*file* konfigurasi \*.sumo.cfg> --fcd-output <*file* *output* \*.sumo.xml>

Kemudian dengan menggunakan fungsi traceExporter pada SUMO. Fungsi ini akan mengkonversi *file* \*.sumo.xml menjadi *file* dengan format \*.tcl, yang pada Network Simulator 2.35 akan digunakan sebagai *file* mobilitas dengan menggunakan fungsi *source* pada fungsi pemrograman tcl. Untuk parameter mobilitas Network Simulator 2.35, pada perintah traceExporter, didefinisikan tipe *output* dengan –-ns2mobility-output. Berikut merupakan format fungsi dan traceExporter.py

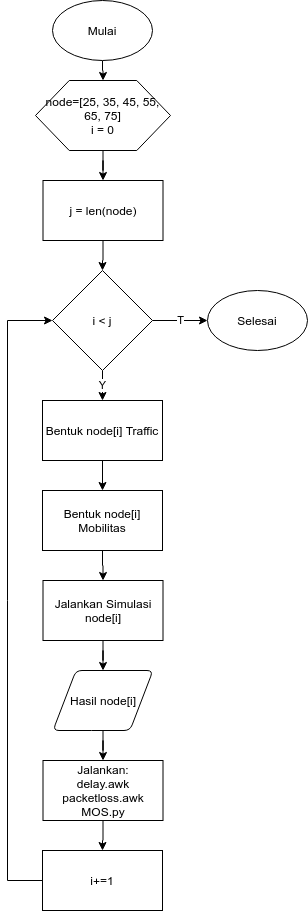
python $SUMO\_HOME/tools/traceExporter.py --fcd-input <file \*.sumo.xml> <tipe *output*> <*output file* \*.tcl>

Selanjutnya adalah pembuatan *traffic* komunikasi *audio*. Pembuatan *traffic* komunikasi *audio* menggunakan modul Ns2Voip++ (<http://cng1.iet.unipi.it/wiki/index.php/Ns2voip%2B%2B>). Ns2Voip++ merupakan sebuah patch yang akan menghasilkan satu *file* berformat tcl sebagai *file* yang menciptakan *traffic* komunikasi *audio.* Pada *file* tcl ini pengaturan beberapa parameter untuk disesuaikan dengan kondisi simulasi. *File* tcl dari Ns2Voip++ akan digunakan sebagai sumber paket komunikasi *audio*, jenis paket dari Ns2Voip++ adalah bertipe UDP (*User Datagram Protocol*). Paket ini akan dikirimkan untuk mensimulasikan komunikasi *audio* di Network Simulator 2.35.

Langkah selanjutnya adalah menjalankan simulasi di Network Simulator 2.35 dengan menjalankan program tcl, simulasi akan menghasilkan *output file trace* dan NAM *file.* Berdasarkan *output* tersebut, akan diketahui kualitas dari komunikasi *audio* yang dijalankan pada simulasi.

1. **Tahapan Kerja Simulasi**

Tahap pengerjaan simulasi ditunjukkan pada diagram alir pada gambar dibawah, berikut penjelas mengenai diagram alir tahapan pengerjaan simulasi:



Jumlah *node* pada proses simulasi mengalami penambahan konstan dari node 25, 35, 45, 55, 65, dan terakhir 75 *node*. Pengerjaan simulasi akan dilakukan sebanyak jumlah *node* pada list.

* + 1. Konfigurasi dan pembentukan *node* pada jaringan *wireless* pada Network Simulator 2.35 dengan penggunaan protokol *routing* AODV dan PA-AODV.
    2. Konfigurasi *traffic* komunikasi *audio* pada *file* generator *traffic*. Proses ini juga akan menentukan *node* pengirim dan penerima.
    3. Menjalankan simulasi untuk mendapatkan *file trace*.
    4. Menjalankan kode perhitungan *delay, packet loss* dan perhitungan MOS untuk mendapatkan nilai kualitas komunikasi *audio* pada simulasi.
    5. Jika nilai i tidak lebih dari j, maka simulasi dilanjutkan pada node indeks i+1.

1. ***Traffic* Komunikasi *Audio***

Traffic komunikasi *audio* dibangun dengan modul pembangkitan *traffic* NS2Voip++. Pembangkitan *traffic* komunikasi *audio* pada proses simulasi mengikuti parameter pada NS2Voip++ sebagai berikut:

|  |  |
| --- | --- |
| Lama Waktu Komunikasi | 60 detik |
| *Talkspurt* | Weibull distribution f(x; 0.412, 0.824) |
| *Silence* | Weibull distribution f(x; 0.899, 1.089) |
| Codec | G.711, G.723.1 |
| Tipe Paket | UDP |

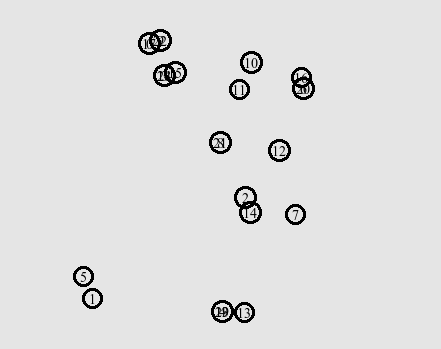
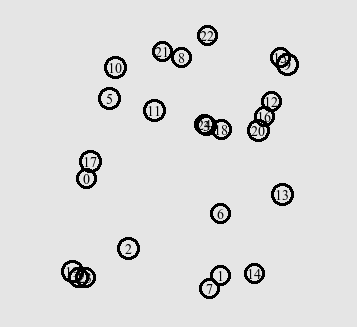
Komunikasi *audio* yang dirancang akan dilakukan selama 60 detik antara penerima dan pengirim. Komunikasi *audio* yang dilakukan merupakan komunikasi satu arah, dengan periode *talkspurt* dan *silence* dibangkitkan dengan distribusi Weibull pada modul NS2Voip++. Ukuran data komunikasi *audio* akan ditentukan berdasarakan ukuran *payload* dari codec yang digunakan. Spesifikasi codec yang digunakan untuk membangkitkan ukuran data atau paket komunikasi *audio* diperlihatkan pada Gambar xx.

Sumber tabel (El Brak, Bouhorma, & Boudhir, 2012).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Codec | Bit rate (kbps) | Sample size (bytes) | Packet per second | Payload size (bytes) |
| G.711 | 64 | 80 | 50 | 160 |
| G.723.1 | 6.3 | 24 | 34 | 20 |

1. **Topologi Jaringan**

Bentuk topologi pada jaringan *ad-hoc* dibuat berdasarkan data geologis pegunungan *openstreetmap.* Peletakan *node* pengirim, penerima dan pengantar diletakkan secara acak pada *edges* yang dihasilkan dari data geologis pegunungan *openstreetmap.* Hasil dari setiap scenario yang dijalankan memiliki pergerakan *node* penerima, pengirim dan pengantar yang berbeda.



Gambar diatas menunjukkan posisi awal *node* yang diletakkan secara acak pada *edges. Node* pada MANET bergerak dan berpindah secara acak mengikuti jalur *edges* pada saat simulasi dijalankan, pada gambar dapat dicontohkan *node* 1 dan *node* 7 bergerak berpindah dan membentuk topologi baru.

1. **Parameter Kinerja**

Pada penelitian ini, yang menjadi tolak ukur kualitas komunikasi *audio* pada simulasi yang dilakukan adalah nilai MOS E-Model. Perhitungan MOS E-Model ini diperoleh dengan beberapa parameter tambahan, yakni *delay* dan *packet loss.* Untuk menghitung *delay* dan *packet loss* dapat dilakukan melalui *file trace*. *File trace* merupakan keluaran yang mencatat segala perilaku *node* pada saat simulasi. Melalui *file* ini perhitungan *delay* dan *packet loss* dapat dilakukan, sehingga mendapatkan nilai MOS E-Model pada akhirnya. Gambar dibawah ini merupakan contoh dari *file trace*

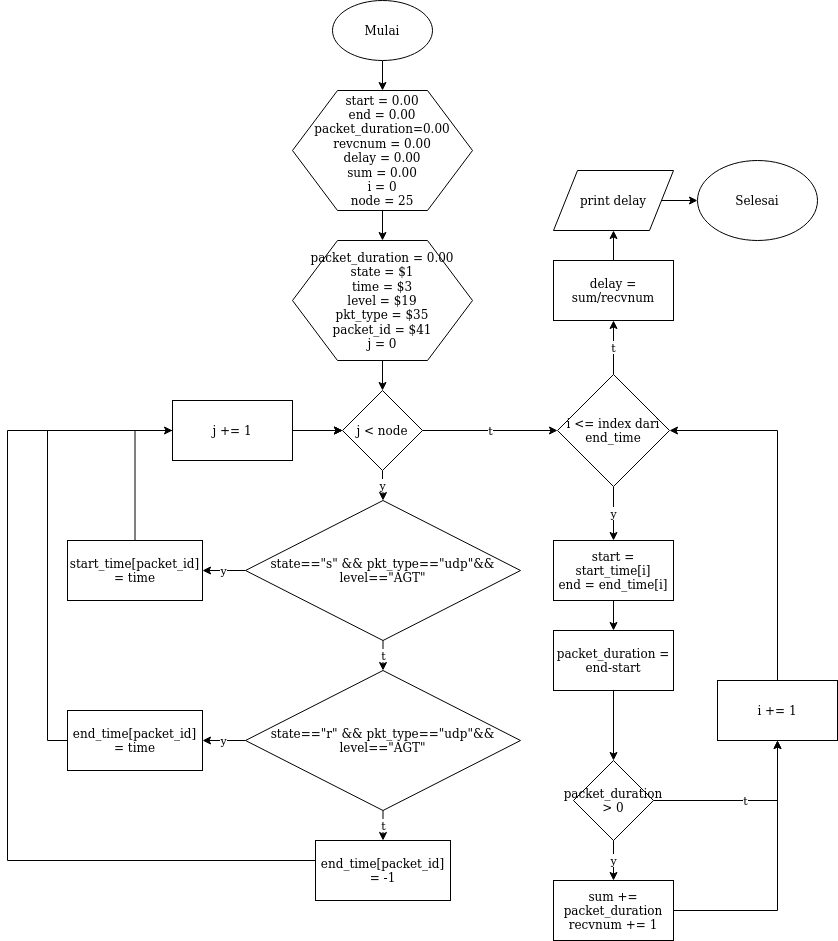
*Gambar file trace.*

|  |  |
| --- | --- |
| 1 | s -t 10.010000000 -Hs 1 -Hd -2 -Ni 1 -Nx 2116.31 -Ny 2677.17 -Nz 0.00 -Ne -1.000000 -Nl AGT -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 1.1 -Id 24.0 -It udp -Il 163 -If 1 -Ii 99 -Iv 32 |
| 2 | s -t 10.033660465 -Hs 7 -Hd -2 -Ni 7 -Nx 2227.70 -Ny 2787.52 -Nz 0.00 -Ne -1.000000 -Nl AGT -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 7.2 -Id 9.0 -It cbr -Il 512 -If 0 -Ii 101 -Iv 32 -Pn cbr -Pi 23 -Pf 0 -Po 0 |
| 3 | r -t 10.034419350 -Hs 10 -Hd -2 -Ni 10 -Nx 2176.96 -Ny 2958.10 -Nz 0.00 -Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md ffffffff -Ms 9 -Mt 800 -Is 9.255 -Id -1.255 -It AODV -Il 48 -If 0 -Ii 0 -Iv 28 -P aodv -Pt 0x2 -Ph 3 -Pb 2 -Pd 24 -Pds 0 -Ps 1 -Pss 6 -Pc REQUEST |

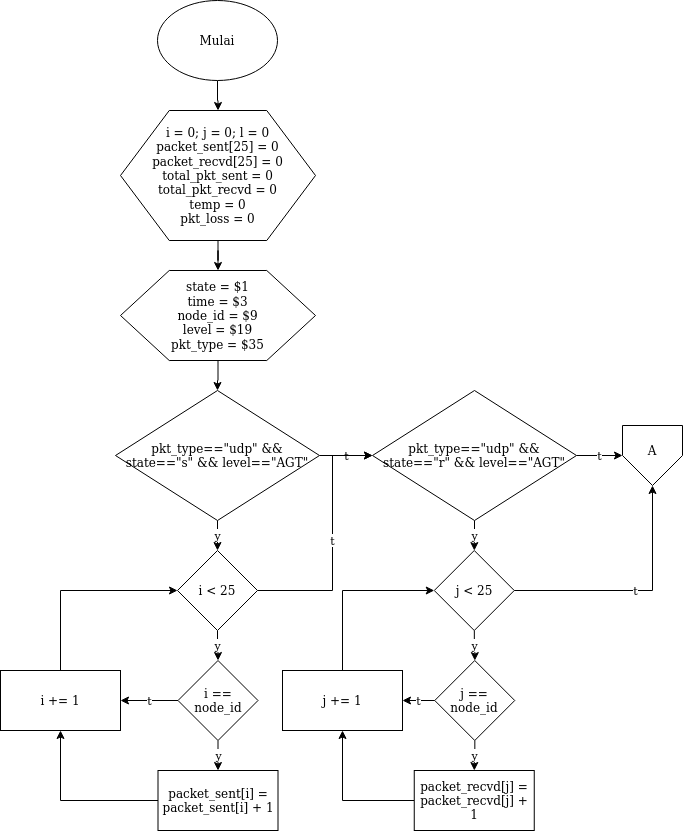
*File trace* mencatat aktivitas yang terjadi pada saat simulasi seperti *send, drop, receive,* waktu, jumlah paket yang dikirimkan. Karena komunikasi yang dilakukan merupakan komunikasi wireless atau nirkabel, maka hal tersebut juga akan mempengaruhi parameter pada *file trace.* Penjelasan mengenai parameter yang terdapat pada *file trace* diperlihatkan pada Gambar xx.

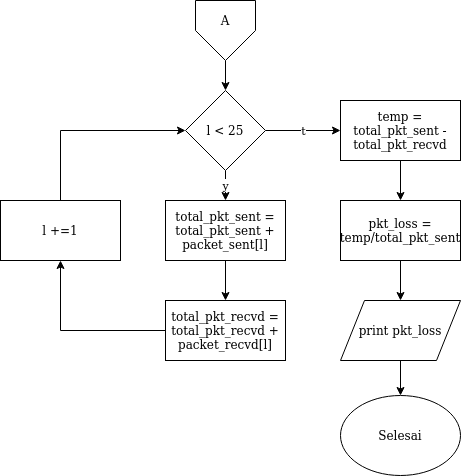
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Event | Singkatan | *Flag* | Tipe | *Value* |
| Koneksi *Wireless* (Nirkabel) | s: send  r: receive  d: drop  f: forward | -t | double | Waktu |
| -Ni | int | ID *Node* |
| -Nx | double | Koordinat X pada *node* |
| -Ny | double | Koordinat Y pada *node* |
| -Nz | double | Koordinat Z pada *node* |
| -Ne | double | *Node* energi |
| -Nl | string | Level *trace* jaringan |
| -Nw | string | Sebab paket *drop* |
| -Hs | int | ID *node* dari *Hop Source* |
| -Hd | int | ID *node* dari *Hop Destination* |
| -Ma | hexadecimal | Durasi |
| -Ms | hexadecimal | Alamat *ethernet* sumber |
| -Md | hexadecimal | Alamat *ethernet* tujuan |
| -Mt | hexadecimal | Tipe *ethernet* |
| -P | string | Tipe paket (arp, dsr, imep, tora, dan lain-lain) |
| -Pn | string | Tipe paket (cbr, tcl) |
| IP *Trace* |  | -Is | int.int | Alamat sumber dan *port* |
| -Id | int.int | Alamat tujuan dan *port* |
| -It | string | Tipe paket |
| -Il | int | Ukuran paket |
| -If | int | ID *flow* |
| -Ii | int | *Unique ID* |
| -Iv | int | Nilai TTL |
| Protokol *Trace* (AODV/PA-AODV) |  | -Pt | hexadecimal | tipe |
| -Ph | int | Jumlah *hop* |
| -Pb | int | *Broadcast* ID |
| -Pd | int | Tujuan |
| -Pds | int | Nomor urut tujuan |
| -Ps | int | Sumber |
| -Pss | int | Nomor urut sumber |
| -Pl | double | *Lifetime* |
| -Pc | string | Operasi (REQUEST, REPLY, ERROR, HELLO) |
| CBR *Trace* |  | -Pi | int | Nomor urut |
| -Pf | int | Berapa kali paket telah di*forward* |
| -Po | int | Jumlah optimal *forward* |

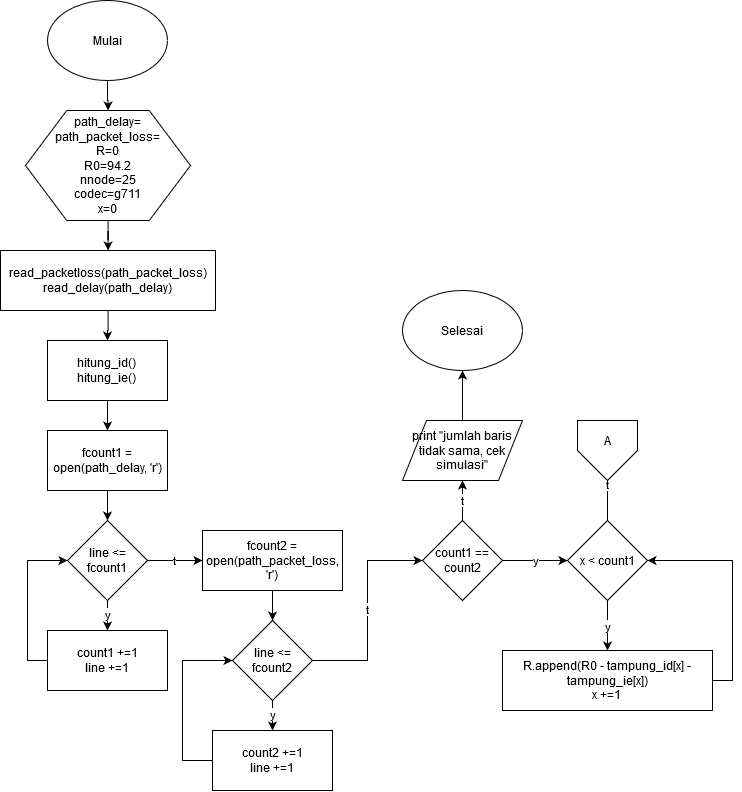
Simulasi yang dilakukan merupakan simulasi dengan perangkat atau *interface* nirkabel (*wireless*)maka *file trace* hasil *trace* akan selalu melibatkan *event* dari Koneksi *Wireless* (nirkabel). Pada baris 1 Gambar xx, merupakan aktivitas dari IP *trace*, membawa tipe paket UDP yang berarti merupakan aktivitas dari komunikasi *audio*. Kemudian pada baris 2 merupakan aktivitas dari pengiriman paket CBR dan baris 3 merupakan aktivitas dari protokol *routing* yang digunakan pada proses simulasi.

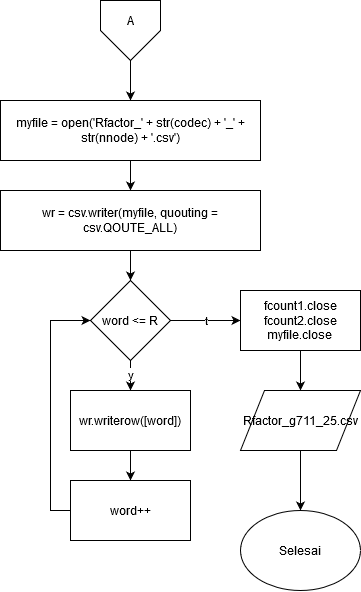
Parameter kinerja Delay

Parameter kinerja paket loss





Parameter kinerja R factor



Parameter kinerja MOS

1. **Tahap Evaluasi**

# BAB IV HASIL DAN PEMBAHASAN

1. **Tahap Implementasi**

Pada penelitian ini, komunikasi *audio* disimulasikan dalam perangkat lunak Network Simulator 2.35. Network Simulator 2.35 digunakan untuk membantu proses simulasi *routing* dan komunikasi *audio.* Protokol *routing* yang digunakan dalam proses simulasi adalah AODV dan PA-AODV sedangkan *traffic audio* yang disimulasikan, dibangkitkan melalui *patch* NS2Voip++. Untuk pembentukkan topologi dan pergerakan *node* menggunakan SUMO dan data geologis pegunungan pada *OpenStreetMap.* Network Simulator 2.35 diimplementasikan pada sistem operasi dengan spesifikasi perangkat sebagai berikut:

|  |  |
| --- | --- |
| Sistem Operasi | Ubuntu 18.04 |
| Prosesor | Intel(R) Core(TM) i7-6700HQ |
| Memory | 8192MB |
| Hardisk | 100GB |

1. **Konfigurasi**

Pada tahap konfigurasi, dilakukan langkah-langkah untuk mengimplementasikan parameter-parameter simulasi serta scenario dalam simulasi. Adapun konfigurasi yang dilakukan adalah sebagai berikut:

1. **Modifikasi PA-AODV**

Pada Network Simulator 2.35, protokol AODV telah disediakan ketika pengguna menginstall Network Simulator 2.35. Pada penelitian ini, selain menggunakan protokol AODV, scenario juga menggunakan protokol PA-AODV. Untuk dapat mengimplementasikan PA-AODV dilakukan dengan memodifikasi file dasar AODV sehingga menjadi PA-AODV. Pada Network Simulator 2.35 diperlukan beberapa modifikasi pada *file* packet.h, ns\_tcl.cc, Makefile.in, priqueue.cc, ns-agent.tcl, ns-lib.tcl, ns-mobilenode.tcl, ns-packet.tcl, cmu-trace.ccdan cmu-trace.h*.* Serta penambahan *folder* pa\_aodv pada direktori ns-2.35 dan *file* pada pa\_aodv pa\_aodv\_logs.cc, pa\_aodv\_packet.h, pa\_aodv\_rqueue.h, pa\_aodv\_rqueue.cc, pa\_aodv\_rtable.h, pa\_rtable.cc, pa\_aodv.hdan pa\_aodv.cc.

1. **Pembentukan Mobilitas**

Proses pembentukan mobilitas menggunakan data geologis dari *OpenStreetMap* kemudian diproses menggunakan SUMO. Berikut merupakan kode pembuatan pergerakan *node* dari data geologis *OpenStreetMap.*

|  |  |
| --- | --- |
| 1 | nnode=(25 35 45 55 65 75) #definisikan jumlah node |

Pembuatan mobilitas untuk *node* diimplementasikan pada kode *bash* yakni generate\_mobility. Pada prosesnya, akan diinisialisasikan terlebih dahulu, *node* yang diakan dibuat pergerakannya. Pada variable nnode mengandung *list* dari *node* yang akan dibangkitkan pergerakannya.

|  |  |
| --- | --- |
| 1 | polyconvert --osm-files osmfiles/sebudi.osm --net-file $netfiles --type-file $SUMO\_HOME/data/typemap/osmPolyconvert.typ.xml -o $poly/$i.poly.xml |

Perintah polyconvert akan membentuk pola geometris – bangunan dari *file* osm. Perintah tersebut akan memerlukan parameter *netfiles* dari proses NETCONVERT dan *file* xml *typemap.* Keluaran perintah ini akan menghasilkan file *additional* pada *folder* $poly dengan nama sesuai jumlah *node* yang dibentuk pada nilai $i.

Gambar xx merupakan pembangkitan pergerakan dan posisi *node* pada *edges* menggunakan fungsi randomTrips.py dari SUMO. Pada baris kedua akan dilakukan pemberian nilai pada variable x dengan 1 per jumlah node yang akan dibentuk. Kemudian pada perintah randomTrips.py beberapa parameter seperti –attributes=”type” mendeklarasikan jenis *node* yang akan dibangkitkan, pada *file* ini dapat diinisialisasikan rentangan kecepatan pergerakan *node*. Parameter -n adalah *netfiles,* -a merupakan *additional file*, *file* dimana rentangan kecepatan pergerakan *node* seperti pada Gambar xx, -r merupakan keluaran dari perintah randomTrips.py, dimana akan menghasilkan *file route* yang berisikan informasi jalur *edges* yang dilalui dari masing-masing *node*, -e merupakan END TIME pembangkitan *node*. -e pada perintah ini diberi nilai 1 detik agar pembuatan *node* dalam 1 detik akan menghasilkan 1/$x *node*.

|  |  |
| --- | --- |
| 1 | x=0 |
| 2 | x=`awk -v var1=1 -v var2=$i 'BEGIN { print ( var1 / var2 ) }'` |
| 3 | python $SUMO\_HOME/tools/randomTrips.py --trip-attributes="type=\"myType\"" -n $netfiles -a $type -r $rout/$i.rou.xml -e 1 -p $x -l #akan membuat 1/$x node dalam 1 detik pada file route. |

Gambar xx merupakan isi dari *file additional* mengenai id dari *node* dan kecepatan maksimum dari pegerakan *node*. Kecepatan pada SUMO didefinisikan dalam satuan meter/second.

|  |  |
| --- | --- |
| 1 | <additional> |
| 2 | <vType id="myType" maxSpeed="7.0" vClass="passenger"/> |
| 3 | </additional> |

Gambar xx merupakan kode pembentukkan *file* berformat \*.sumo.cfg. Isi dari *file* yang terbentuk menggunakan *file* dari proses sebelumnya yakni input *net-files, route-files* dan *additional-files*. Masing-masing parameter memiliki nilai *path* dari masing-masing file tersebut. Kemudian pada *file* konfigurasi SUMO ini juga didefinisikan waktu mulai dan waktu berhenti simulasi. Waktu ini akan menjadi waktu rentangan pergerakan *node*, sehingga nilai *end value* disesuaikan dengan lama waktu simulasi. Pada baris ke 16, perintah sumo -c akan menghasilkan *file* sumo.xml, *file* ini mengandung nilai pergerakan dari *node* setiap 0.10 detik. Parameter yang dibutuhkan dari perintah sumo -c yakni *path* file konfigurasi sumo.cfg, dan *path* untuk *output file*.

|  |  |
| --- | --- |
| 1 | echo "building $i sumo.cfg configuration" |
| 2 | printf "<configuration> |
| 3 | <input> |
| 4 | <net-file value='"$HOME/$proj/$netfiles"'/> |
| 5 | <route-files value='"$HOME/$proj/$rout/$i.rou.xml"'/> |
| 6 | <additional-files value='"$HOME/$proj/$poly/$i.poly.xml"'/> |
| 7 | </input> |
| 8 | <time> |
| 9 | <begin value='"0"'/> |
| 10 | <end value='"120"'/> |
| 11 | <step-length value='"0.1"'/> |
| 12 | </time> |
| 13 | </configuration>" > $sumocfg/$i.sumo.cfg |
| 14 |  |
| 15 | echo "sumo.xml files creating ....." |
| 16 | sumo -c $HOME/$proj/$sumocfg/$i.sumo.cfg --fcd-output $sumoxml/$i.sumo.xml |

Untuk menghasilkan *file* pergerakan yang dapat digunakan pada Network Simulator 2.35, SUMO menyediakan *tools* yakni traceExporter.py. Ini memerlukan parameter input berupa *file* sumo.xml dan dengan parameter –ns2mobility-output dapat menghasilkan *file* dengan format tcl yang dapat digunakan sebagai pergerakan *node* pada Network Simulator 2.35. Gambar xx merupakan penggunaan perintah traceExporter.py pada SUMO.

|  |  |
| --- | --- |
| 1 | Python $SUMO\_HOME/tools/traceExporter.py --fcd-input $sumoxml/$i.sumo.xml --ns2mobility-output $ns2mobility/mobility\_$i.tcl |

Isikan ndak ya file pergerakan .tclnya?

1. **Membentuk Objek Simulasi**

Pada proses simulasi, diperlukan penentuan objek-objek yang akan menjadi tolak ukur untuk menentukan kualitas komunikasi atau jaringan baik itu jumlah *node* yang digunakan terdiri dari *node* sumber, *node* pengantar, *node* tujuan, jenis codec serta protokol *routing* MANET yang digunakan.

|  |  |
| --- | --- |
| 1 | set val(chan) Channel/WirelessChannel |
| 2 | set val(prop) Propagation/TwoRayGround |
| 3 | set val(netif) Phy/WirelessPhy |
| 4 | set val(mac) Mac/802\_11 |
| 5 | set val(ifq) Queue/DropTail/PriQueue |
| 6 | set val(ll) LL |
| 7 | set val(ant) Antenna/OmniAntenna |
| 8 | set val(x) 4594 |
| 9 | set val(y) 4864 |
| 10 | set val(ifqlen) 50 |
| 11 | set val(adhocRouting) $opt(routing) |
| 12 | set val(nn) $opt(nnode) |
| 13 | set val(stop) 120 |
| 14 | set val(vip) "./voip.tcl" |

Gambar xx menjelaskan parameter dari pembentukan jaringan *wireless*. Pada baris 1 menujukkan jenis kanal yang digunakan, baris 2 merupakan tipe radio, baris 3 menunjukkan jenis *interface*, baris 4 merupakan jenis MAC, baris 7 menunjukkan jenis antenna. Nilai val(x) dan val(y) merupakan nilai panjang dan lebar dari luas simulasi, baris 10 menunjukkan maksimum paket dalam antrian, baris 11 menunjukkan jenis protokol *routing* yang digunakan, baris 12 menunjukkan jumlah node, baris 13 merupakan nilai dari lama waktu simulasi. Kemudian val(vip) berisikan nilai *path* menuju file generator *traffic* komunikasi *audio* pada simulasi.

|  |  |
| --- | --- |
| 1 | set topo [new Topography] |
| 2 | $topo load\_flatgrid $val(x) $val(y) |
| 3 | # SET G O D |
| 4 | set god\_ [create-god $val(nn)] |

Gambar xx merupakan *script* untuk tahap pengaturan luas area simulasi. Luas area sudah ditentukan denan parameter val(x) dan val(y), yang ditunjukkan pada Gambar xx. Dilanjutkan dengan konfigurasi tiap-tiap *node*.

|  |  |
| --- | --- |
| 1 | # N O D E Config |
| 2 | $ns\_ node-config -adhocRouting $val(adhocRouting) \ |
| 3 | -llType $val(ll) \ |
| 4 | -macType $val(mac) \ |
| 5 | -ifqType $val(ifq) \ |
| 6 | -ifqLen $val(ifqlen) \ |
| 7 | -antType $val(ant) \ |
| 8 | -propType $val(prop) \ |
| 9 | -phyType $val(netif) \ |
| 10 | -channelType $val(chan) \ |
| 11 | -topoInstance $topo \ |
| 12 | -agentTrace ON \ |
| 13 | -routerTrace ON \ |
| 14 | -macTrace OFF \ |
| 15 | -movementTrace OFF |

Gambar xx menunjukkan pengaturan tiap-tiap *node* yang terbentuk. Pada tahap ini, parameter pada Gambar xx akan digunakan oleh tiap-tiap *node* sesuai dengan banyaknya *node* yang dibentuk.

1. **Pembuatan *File Trace* dan *File NAM***

*File trace* merupakan hasil dari seluruh aktivitas yang terjadi selama proses simulasi, seluruh aktivitas seperti *event,* waktu, besaran paket, *node* sumber, *node* tujuan tercatat dalam *file trace.* Berikut merupakan kode yang digunakan untuk menghasilkan *file trace.*

|  |  |
| --- | --- |
| 1 | #use new trace format |
| 2 | $ns\_ use-newtrace |
| 3 | set tracefile [open "trace/voip-$opt(codec)-$opt(nnode)-T$opt(try)-Node[lindex $vnode1 [expr $opt(try)-1]]-[lindex $vnode2 [expr $opt(try)-1]].tr" w] |
| 4 | $ns\_ trace-all $tracefile |

Gambar xx menunjukkan proses pembuatan *file trace* simulasi. Pada baris 2, merupakan kode untuk menggunakan format *file trace* yang baru. Kemudian pada baris 3, perintah set tracefile akan membuat *file* pada *folder* trace dengan format penamaan file voip-<jenis codec>-<jumlah node>-<replikasi ke>-<node pengirim>-<node tujuan> dengan format .tr. Pada baris 4 perintah untuk melakukan *trace* dan disimpan pada *file* $tracefile.

*File NAM* merupakan hasil dari proses simulasi pada Network Simulator 2.35 yang menampilkan objek-objek simulasi yang digunakan. Hal seperti jumlah *node,* pergerakan *node* serta proses transmisi yang disimulasikan dapat dilihat melalui *file NAM*. Berikut merupakan kode yang digunakan untuk menghasilkan *file NAM.*

|  |  |
| --- | --- |
| 1 | set namtrace [open "nam/voip-$opt(codec)-$opt(nnode)-T$opt(try)-Node[lindex $vnode1 [expr $opt(try)-1]]-[lindex $vnode2 [expr $opt(try)-1]].nam" w] |
| 2 | $ns\_ namtrace-all-wireless $namtrace $val(x) $val(y) |

Gambar xx memperlihatkan kode pembentukan *file NAM* pada simulasi. Baris 1 merupakan inisialisasi *file NAM* pada variable namtrace. Format penamaan *file NAM* sama dengan *file trace*. Kemudian pada baris 2 merupakan proses pembentukan objek-objek simulasi pada *file* namtrace.

Berikut ini merupakan kode yang digunakan untuk mengakhiri keseluruhan aktivitas pada *file trace* dan *file NAM.*

|  |  |
| --- | --- |
| 1 | proc finish {} { |
| 2 | global ns\_ tracefile |
| 3 | $ns\_ flush-trace |
| 4 | close $tracefile |
| 5 | exit 0 |
| 6 | } |

1. **Membentuk *Node***

Posisi dan pergerakan *node* yang sebelumnya telah dibuat menggunakan data *OpenStreetMap* dan SUMO akan digunakan pada *file* tcl. Berikut merupakan kode yang digunakan untuk membangkitkan sejumlah node sesuai scenario simulasi.

|  |  |
| --- | --- |
| 1 | for {set i 0} {$i < $val(nn) } {incr i} { |
| 2 | set node\_($i) [$ns\_ node] |
| 3 | $node\_($i) random-motion 0 |
| 4 | #Without random motion |
| 5 | } |

Gambar diatas merupakan kode pembangkitan *node*, perulangan pembangkitan *node* sejumlah *node* pada scenario.

|  |  |
| --- | --- |
| 1 | # M O B I L I T Y config |
| 2 | if {$opt(nnode) == 25} { |
| 3 | set val(mob) "./osmfiles/mobility/mobility\_25.tcl" |
| 4 | } elseif {$opt(nnode) == 35} { |
| 5 | set val(mob) "./osmfiles/mobility/mobility\_35.tcl" |
| 6 | } elseif {$opt(nnode) == 45} { |
| 7 | set val(mob) "./osmfiles/mobility/mobility\_45.tcl" |
| 8 | } elseif {$opt(nnode) == 55} { |
| 9 | set val(mob) "./osmfiles/mobility/mobility\_55.tcl" |
| 10 | } elseif {$opt(nnode) == 65} { |
| 11 | set val(mob) "./osmfiles/mobility/mobility\_65.tcl" |
| 12 | } elseif {$opt(nnode) == 75} { |
| 13 | set val(mob) "./osmfiles/mobility/mobility\_75.tcl" |
| 14 | } else { |
| 15 | puts "# tidak ada file MOBILITY!!" |
| 16 | exit 0 |
| 17 | } |

Pada Gambar xx, pergerakan *node* akan disesuaikan dengan jumlah *node* yang digunakan dalam simulasi. Jumlah pergerakan *node* sesuai dengan scenario simulasi yang ditentukan pada Gambar xx.

1. **Pembentukan *Traffic* *Audio***

Pembentukan *traffic audio* untuk komunikasi audio dilakukan pada *file* tcl dari patch NS2Voip++. Berikut merupakan kode pembentukan paket *audio* yang dikirimkan pada proses simulasi.

Kode di voip.tcl

|  |  |
| --- | --- |
| 1 | set val(nn) 30 |
| 2 | for {set i 0} {$i < $val(nn)} {incr i} { |
| 3 | set node\_($i) [$ns\_ node] |
| 4 | $ns\_ initial\_node\_pos $node\_($i) 2 |
| 5 | } |
| 6 | puts "Loading connection pattern ..." |
| 7 | source "/root/letakfile/namafile" |

1. **Pembentukan *Background Traffic***

Implementasi pembentukan *background traffic* pada Network Simulator 2.35 menggunakan *file* cbrgen.tcl dari Network Simulator 2.35. Kode pembuatan traffic diimplementasikan dengan *script* *bash*, berikut perintah penggunaan pembentukan *background traffic.*

|  |  |
| --- | --- |
| 1 | ns cbrgen.tcl -type cbr -nn $i -seed 1 -mc 10 -rate 10.0 > traffic-$i.tcl |

Pembentukan *traffic* digunakan untuk merepresentasikan komunikasi yang sebenarnya. *Traffic* yang digunakan berupa paket CBR (Constant Bit Rate) dengan maksimum koneksi antara 10 *node* dengan besaran paket 10.0 bytes. Perintah diatas akan menghasilkan *output* *file* yang berisikan pembentukan komunikasi antar *node* dengan besaran paketnya.

Gambar xx menunjukkan penggunaan *background traffic* yang dibentuk. Penggunaan *background traffic* akan disesuaikan dengan jumlah *node* pada proses simulasi.

|  |  |
| --- | --- |
| 1 | # B G T R A F F I C config |
| 2 | if {$opt(nnode) == 25} { |
| 3 | set val(cp) "./traffic-25.tcl" |
| 4 | } elseif {$opt(nnode) == 35} { |
| 5 | set val(cp) "./traffic-35.tcl" |
| 6 | } elseif {$opt(nnode) == 45} { |
| 7 | set val(cp) "./traffic-45.tcl" |
| 8 | } elseif {$opt(nnode) == 55} { |
| 9 | set val(cp) "./traffic-55.tcl" |
| 10 | } elseif {$opt(nnode) == 65} { |
| 11 | set val(cp) "./traffic-65.tcl" |
| 12 | } elseif {$opt(nnode) == 75} { |
| 13 | set val(cp) "./traffic-75.tcl" |
| 14 | } else { |
| 15 | puts "# tidak ada file traffic!!" |
| 16 | exit 0 |
| 17 | } |

1. **Pembentukan Koneksi**

Pembentukan koneksi

|  |  |
| --- | --- |
| 1 | #set source node dan destination node |
| 2 | if { $opt(nnode) == 50 } { |
| 3 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 4 | set vnode2 {49 48 47 46 45 44 43 42 41 40} |
| 5 | } elseif { $opt(nnode) == 25 } { |
| 6 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 7 | set vnode2 {24 23 22 21 20 19 18 17 16 15} |
| 8 | } elseif { $opt(nnode) == 35 } { |
| 9 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 10 | set vnode2 {34 33 32 31 30 29 28 27 26 25} |
| 11 | } elseif { $opt(nnode) == 45 } { |
| 12 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 13 | set vnode2 {44 43 42 41 40 39 38 37 36 35} |
| 14 | } elseif { $opt(nnode) == 55 } { |
| 15 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 16 | set vnode2 {54 53 52 51 50 49 48 47 46 45} |
| 17 | } elseif { $opt(nnode) == 65 } { |
| 18 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 19 | set vnode2 {64 63 62 61 60 59 58 57 56 55} |
| 20 | } elseif { $opt(nnode) == 75 } { |
| 21 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 22 | set vnode2 {74 73 72 71 70 69 68 67 66 65} |
| 23 | } elseif { $opt(nnode) == 70 } { |
| 24 | set vnode1 {1 2 3 4 5 6 7 8 9 10} |
| 25 | set vnode2 {69 68 67 66 65 64 63 62 61 60} |
| 26 | } else { |
| 27 | set vnode1 {2 6 10 14 18} |
| 28 | set vnode2 {3 7 11 15 19} |
| 29 | } |

1. **Pengujian dan Evaluasi Sistem**