Pencarian Rute Berdasarkan Jumlah Kepadatan Node Menggunakan Protokol Routing DSR Pada Jaringan MANET

(*Route Searching Based on Number of Node Density Using DSR Routing Protocol on MANET Network*)

Andy Hidayat Jatmika [1], Baiq Ashfiatun Nisa' Mulyani[1], Ariyan Zubaidi[1]
[1] Program Studi Teknik Informatika, Universitas Mataram
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: baiqashfiatunnisamulyani@gmail.com, [andy, zubaidi13]@unram.ac.id

Mobile Ad-hoc Network (MANET) is a wireless network that is temporary in nature and consists of a collection of mobile nodes that move freely in any direction. When a mobile node intends to transmit data packets to a specific destination node, it requires routing details retrieved from its routing table. The routing process requires a routing protocol that determines the route to be taken to deliver the data packets. The Dynamic Source Routing (DSR) protocol is used for this purpose. DSR is a reactive protocol that establishes routes on-demand when there is a route request. When a route request is initiated, the DSR protocol sends a route request packet (RREQ) to all its neighboring nodes. The nodes participating in this route discovery process are referred to as intermediate nodes. However, a problem arises in finding a route based on the node density in the DSR routing protocol to reduce the delay caused by route rediscovery when a route breaks. To address this issue, a solution is proposed by calculating the node density level among neighboring nodes to select the intermediate node candidates with the highest node density using the node density algorithm. Consequently, after conducting the tests in this research, it is observed that the implementation of the node density algorithm in the DSR routing protocol improves the performance of the standard DSR routing. The average throughput shows an increase of 0.50%, the packet delivery ratio (PDR) shows an increase of 0.22%, and the end-to-end delay shows a decrease of 0.50%.

***Key words*: Mobile Ad-hoc Network, Node Density, Rute, Routing, Dynamic Source Routing.**

I. PENDAHULUAN

Jaringan *Mobile Ad-Hoc* (MANET) adalah sebuah jaringan nirkabel sementara yang terdiri dari sekelompok *mobile node* yang bergerak bebas di segala arah. Jaringan ini tidak memiliki manajemen terpusat, sehingga setiap simpul *mobile* dapat mengelola dirinya sendiri secara mandiri dan dapat dibangun di mana dan kapan saja jika diperlukan[1].

Ketika sebuah mobile *node* ingin mengirim *packet* data ke *node* tujuan, *node* tersebut memerlukan infomasi *rute* yang ada di *table routing*-nya. Jika informasi *rute* tersebut tersedia, *packet* data dapat langsung dikirim ke *node* tujuan. Namun, jika informasi rute belum tersedia, maka pencarian *rute* perlu dilakukan. Pada proses pencarian *rute* protokol *routing* diperlukan. *Routing* adalah proses menentukan *rute* yang akan digunakan untuk mengirimkan paket data[2].

Salah satu jenis protokol *routing* yang digunakan adalah *Dynamic Source Routing* (DSR). Protokol DSR adalah mekanisme pencarian *rute* yang berguna untuk menemukan dan memilih *rute*. Protokol ini dipilih karena sifatnya yang *reaktif*, di mana *rute* terbentuk hanya saat ada permintaan *rute*. Ketika ada permintaan *rute*, protokol DSR mengirimkan *packet route request* (RREQ) ke semua *node* tetangga nya. Jika informasi tentang *node* tidak ada, *packet* RREQ akan diteruskan oleh *node* tetangga sampai *node* tujuan ditemukan. *Node-node* yang berpartisipasi pada pencarian *rute* ini disebut *node intermediate*. Kelemahan protokol *routing* DSR adalah tidak memperhitungkan *node intermediate* terbaik saat melakukan proses pencarian rute[3]. Protokol *routing* DSR pun tidak melakukan melakukan *broadcast* RREQ[4]. Selain itu, protokol *routing* DSR cenderung mengalami penundaan yang lebih lama saat pertama kali pengiriman data dalam melakukan pencarian *rute* baru[5]. Ketika *rute* telah ditemukan, ada kemungkinan *rute* yang dilewati melalui *node intermediate* tersebut putus, sehingga perlu dilakukan pencarian ulang dengan melakukan *broadcast* pesan RREQ ke *node* tetangga. Jika tingkat kepadatan *node* tetangga tidak terlalu tinggi, pilihan untuk kandidat *node intermediate* akan berkurang, sehingga waktu yang dibutuhkan untuk pencarian rute semakin lama dan *delay* akan semakin tinggi.

Berdasarkan uraian sebelumnya, solusi untuk mengatasi masalah tersebut adalah dengan menghitung tingkat kepadatan *node* tetangga untuk memilih *node intermediate* dengan jumlah kepadatan *node* terbesar[6]. Nilai tingkat kepadatan *node* dapat ditentukan dengan menghitung jumlah *node* tetangga yang berada di *range* komunikasi, kemudian membagi jumlah *node* tetangga dengan jumlah total *node* yang digunakan dalam simulasi. Nilai ini akan menjadi *threshold* atau ambang batas untuk menentukan kandidat *node intermediate* yang memiliki

tingkat kepadatan *node* terbesar dalam proses pencarian rute[6].

Dalam penelitian ini, akan dilakukan modifikasi pada kerangka protokol *routing* DSR dengan menambahkan penghitungan tingkat kepadatan *node* tetangga. Pemilihan tingkat kepadatan *node* bertujuan untuk memilih *node intermediate* yang akan menyebarkan pesan RREQ. Protokol *routing* yang telah dimodifikasi disebut DSR-Node *Density*, dan kinerjanya akan dibandingkan dengan protokol *routing* DSR standar. Sebelum proses *broadcast* pesan RREQ dilakukan, *node* akan menghitung dan menyimpan nilai *threshold* dalam tabel *neighbor*. Nilai *threshold* yang paling tinggi akan menjadi *node intermediate* yang menerima pesan RREQ.

Simulasi jaringan akan dilakukan dengan menggunakan *tools Network* Simulator 2 (NS-2), yang di *install* di sistem operasi Linux Ubuntu dalam lingkungan *VirtualBox*. Parameter pengujian kinerja kedua protokol *routing* yang dibandingkan akan dilihat dari *throughput*, *end-to-end delay*, dan *packet delivery ratio*.

II. Tinjauan Pustaka

Terdapat beberapa penelitian yang mengusulkan modifikasi pada protokol routing AODV, DYMO, dan ZRP dalam rangka meningkatkan kualitas jaringan MANET berdasarkan tingkat kepadatan *node.*

Dalam penelitian [7], sebuah algoritma berdasarkan tingkat kepadatan *node* diusulkan untuk mengurangi *flooding* atau *broadcast message* secara luas saat melakukan pencarian *rute* pada protocol *routing* AODV. Algoritma ini bekerja dengan membandingkan *node density* dengan nilai *threshold* yang telah ditentukan (0.95). Jika *node density* lebih rendah dari 0.95, nilai TTL (*Time to Live*) akan diperiksa. Jika nilai TTL lebih besar dari 7 *hop*, TTL akan ditambahkan dengan nilai TTL saat ini. Selanjutnya, akan diperiksa apakah nilai TTL tersebut melebihi *network* diameter (berjumlah 32 *hops*). Jika iya, *broadcast* dilakukan dengan nilai TTL sebesar 32 *hops*, dan jika tidak, *broadcast* dengan nilai TTL saat ini. Parameter pengujian yang digunakan meliputi *throughput*, *delay*, dan *packet delivery ratio*. Hasil penelitian menunjukkan bahwa protokol *routing* AODV dengan modifikasi dapat mengurangi *flooding broadcast message* sebesar 50%, serta meningkatkan *throughput* dan *packet delivery ratio*, namun memberikan *delay* yang tinggi dibandingkan dengan AODV standar.

Pada penelitian [8] mengusulkan algoritma pencarian *rute* berdasarkan tingkat kepadatan *node*, kali ini diterapkan pada protokol *routing* AODV. Saat ada permintaan *rute*, *node* akan memilih berdasarkan besarnya *link bandwidth node* tetangganya. Pemilihan ini akan digunakan sebagai parameter tingkat kepadatan *node*. Kemudian *node* akan mengirimkan paket *route request* (RREQ) ke *node* tetangga. Jika *rute* belum ditemukan, *node* tetangga akan melakukan hal yang sama untuk meneruskan paket RREQ. Parameter pengujian yang digunakan adalah *packet loss* dan *end-to-end delay*. Hasil penelitian menunjukkan bahwa protokol *routing* AODV dengan modifikasi memberikan nilai *end-to-end delay* yang lebih rendah dan *packet loss* yang lebih sedikit dibandingkan dengan AODV standar.

Pada penelitian [9], menggunakan protokol *routing* DYMO dan DSR yang berjenis *reactive routing*. Pada kedua protokol ini, pembentukan *rute* dari *node* sumber menuju *node* tujuan terjadi hanya saat ada permintaan dari node sumber. Dalam penelitian ini, digunakan Network Simulator 2.35 dengan skenario yang melibatkan variasi tingkat kepadatan *node* dari 50 hingga 200 *node* dengan penambahan sebanyak 10 *node*, serta variasi ukuran paket data sebesar 512 dan 1024 *byte.* Kinerja protokol diukur berdasarkan parameter *throughput*, *end to end delay*, *packet loss*, dan *average* dengan nilai terbaik yang diperoleh secara berturut-turut adalah 101,45 KBps untuk ukuran paket data 512 *byte*, 44,26 ms untuk ukuran paket data 512 *byte*, 2,36% untuk ukuran paket data 512 *byte*, dan 1,72 ms untuk ukuran paket data 512 *byte*. Hasil ini didapatkan pada protokol *routing* DYMO, sehingga dapat disimpulkan bahwa protokol *routing* DYMO menunjukkan kinerja yang lebih baik daripada protokol DSR dalam hal kepadatan *node* dengan variasi ukuran paket yang berbeda.

Pada penelitian [10] mengusulkan memodifikasi pada protokol *routing Zone Routing Protokol* (ZRP) dengan mengubah konsep penentuan zona dari statis menjadi adaptif berdasarkan tingkat kepadatan *node* tetangga pada zona di dalam radius. Penyesuaian zona adaptif ini dilakukan dengan melakukan perhitungan terhadap jumlah *node* tetangga dalam setiap *node*. Jika jumlah *node* tetangga di bawah nilai ambang yang ditetapkan, maka jangkauan akan diperluas satu satuan. Sebaliknya, jika jumlah *node* tetangga melebihi nilai ambang, maka jangkauan akan dikurangi satu satuan. Parameter pengujian meliputi rasio pengiriman paket, penggunaan jalur *routing*, dan keterlambatan pengiriman. Hasil implementasi simulasi menunjukkan peningkatan rata-rata rasio pengiriman paket sebesar 2,21% dan peningkatan rata-rata keterlambatan pengiriman sebesar 3,23%. Penyesuaian zona adaptif pada *protokol routing* ZRP ini dapat diaplikasikan dan berdampak pada kinerja algoritma *routing* ZRP dalam lingkungan MANET.

III. Dasar Teori

A. Jaringan Mobile Ad-Hoc Network (MANET)

Jaringan *Mobile ad-hoc network* (MANET) merupakan jenis jaringan nirkabel yang bersifat sementara dan terdiri dari sekumpulan *node* yang bergerak secara dinamis tanpa membutuhkan infrastruktur jaringan yang sudah ada[10]. Dalam MANET, *node-node* tersebut bergerak secara acak dan bebas, sehingga mengakibatkan perubahan koneksi antar *node*. Jika suatu *node* keluar dari jaringan, jalur komunikasi yang sudah ada akan terputus. Pada saat pengiriman paket data, hal ini menuntut penemuan jalur alternatif dengan segera agar paket dapat

mencapai tujuan yang dituju[11]. Karakteristik jaringan *ad hoc* antara lain[12]: topologi yang dinamis, *bandwidth* terbatas, energi baterai terbatas, dan lemahnya keamanan jaringan secara fisik.

B. Protokol Routing

Protokol merujuk pada seperangkat aturan yang mengatur pertukaran informasi antara komputer melalui media jaringan. Salah satu fungsi utama protokol adalah *routing*, yang melibatkan pengiriman data dari satu lokasi ke lokasi lain hingga mencapai tujuan akhir. Dalam konteks jaringan, *router* digunakan untuk mengarahkan lalu lintas data. *Router* akan memilih jalur terbaik untuk menghubungkan dua host yang ingin berkomunikasi. Dalam jaringan *Mobile ad-hoc network* (MANET), terdapat tiga jenis protokol *routing*, yaitu proaktif, reaktif, dan hibrid[13].

B.1 Protokol Routing Proaktif

Protokol *routing proaktif* adalah jenis protokol routing di mana informasi pada tabel routing harus diperbarui secara berkala. Protokol ini tidak memerlukan proses pencarian *rute* (*route discovery*) karena *rute* tujuan telah tersimpan sebelumnya. Contoh protokol ini adalah FSR, OLSR, DSDV.

B.2 Protokol Routing Reaktif

Protokol *routing reaktif* merupakan salah satu tipe protokol yang melakukan pencarian *rute* hanya ketika *node* sumber memerlukan komunikasi dengan bagian *node* tujuan. Protokol *routing reaktif* melibatkan *fase* pencarian rute, di mana paket permintaan (*request*) akan disebarluaskan (*flooding*) ke seluruh jaringan untuk mencari jalur yang stabil, dan *fase* ini akan berakhir ketika *rute* ditemukan. Protokol *routing* reaktif juga dikenal sebagai *protokol routing on-demand* karena tabel *routing* hanya diperbarui secara berkala ketika ada data yang akan dikirimkan. Contoh protokol ini adalah AODV, DSR, JARR, TORA [14].

B.3 Protokol Routing Hibrid

Protokol *routing hybrid* adalah jenis protokol *routing* yang mendukung *multiple route* (*rute* ganda). Jika suatu paket data gagal dikirim, protokol tidak perlu mencari *rute* baru karena masih ada *rute* lain yang dapat digunakan untuk mengirimkan informasi data. Contoh protokol ini adalah ZRP, HARP, HAODV.

C. DSR

Protokol *Dynamic Source Routing* (DSR) merupakan salah satu tipe dari protokol *routing* reaktif yang aktif saat ada permintaan dari *source node* untuk mengirimkan pesan ke *node* tujuan. DSR menggunakan *cache memory* yang akan menyimpan informasi *routing* dalam jaringan, sehingga dapat untuk memproses pemulihan jaringan saat terjadi perubahan topologi yang mendadak. DSR memulai pencarian *rute* dari *source node* ke *node* tujuan dengan melakukan pemulihan rute atau *route recovery*. Pada tahap tersebut, terjadi aktivitas DSR mentransfer pesan *Route Request* (RREQ) ke semua *node* tetangga dari *source node*. Alamat pengirim dan tujuan pesan akan tersimpan di RREQ. Informasi mengenai jalur menuju pengiriman dalam *cache rauting* akan selalu di perbaharui pada *node* yang menerima RREQ. Ketika *node* tujuan menerima RREQ, *node* tersebut menyampaikan pesan *Route Reply* (RREP) kembali ke *source node*. RREP membagikan informasi lengkap tentang jalur dari *source node* ke *node* tujuan[15].

D. Network Simulator 2 (NS-2)

Network Simulator 2 (*NS-2*) merupakan *software* yang digunakan untuk mereplikasikan dan menggambarkan bagaimana proses komunikasi tersebut terjadi. *Network simulator* dapat melakukan simulasi untuk komunikasi dengan kabel maupun nirkabel.

Output NS-2 berupa *file log* yang berekstensi ".tr". Kemudian dari data *log* ini dapat diolah dan dianalisis menggunakan metode manual atau menggunakan skrip awk yang merupakan file terpisah[12].

E. Network Simulator 2 (NS-2) Trace file

NS-2 *trace file* adalah hasil *output* yang dihasilkan saat menjalankan NS-2, yang memiliki ekstensi file .tr. *Trace File* ini berisi *log* yang mencatat semua pengiriman dan penerimaan paket yang terjadi selama simulasi berlangsung. Di dalam *Trace File* ini terdapat informasi mengenai berbagai jenis paket sesuai dengan jenis protokol *routing* yang digunakan. Untuk mengakses informasi performa masing-masing protokol, setiap baris pada *log* perlu dianalisis secara individual. Sebagai contoh, berikut adalah contoh pengiriman paket data dalam NS-2 *Trace File*:

```
s 26.000000000 _1_ AGT --- 618 cbr 64
[0 0 0 0] -- ----- [1:0 0:0 32 0] [26]
                  0 0
```

Huruf "s" pada *field* pertama merupakan transmisi paket (*send*), *field* kedua berisi waktu transmisi paket pada detik ke 26, *field* ketiga menunjukkan *node* lokasi terjadinya peristiwa yaitu pada *node* 1. *Field* keempat berisi AGT yang mengisyaratkan pengiriman paket data, *field* kelima mencatat peristiwa khusus seperti tabrakan (*collision*), *field* keenam adalah ID unik paket, *field* ketujuh berisi jenis paket yang dikirimkan, yaitu "cbr", dan *field* kedelapan menunjukkan ukuran paket dalam *byte*, yakni 64. Untuk perolehan data paket, formatnya tidak jauh berbeda dengan pengiriman data paket. Perbedaannya terletak pada kolom pertama dengan huruf awalan "r" yang menunjukkan paket diterima. Type *field-field* berikutnya sama persis dengan paket pengiriman. Berikut ini adalah contoh perolehan data paket dalam NS-2 *Trace File* [16]:

```
r 26.011627928 _0_ AGT --- 618 cbr 84
[13a 0 16 800] ------- [1:0 0:0 27 0]
               [26] 5 0
```

IV. METODE PENELITIAN

A. Kebutuhan Perangkat Keras dan Perangkat Lunak

A.1 Hardware

Spesifikasi perangkat keras mengacu pada Laptop yang digunakan oleh peneliti yaitu Asus Vivobook A409F. Spesifikasi komponen pada laptop tersebut telah memenuhi standar minimum untuk digunakan pada penelitian ini.

Spesifikasi perangkat keras yang digunakan pada penelitian ini adalah sebagai berikut:

1) *Central Processing Unit*
   AMD Ryzen 5 5600H 3.30 GHz.
2) *Main Memory*
   8 GB DDR4.
3) *Storage*
   140 GB SSD.

A.2 Software

Adapun kebutuhan *software* yang digunakan pada penelitian ini adalah sebagai berikut:

1) *Network Simulator* 2 (NS-2) versi 2.35

NS-2 menjadi *software* yang digunakan untuk keperluan simulasi dalam bidang protokol, jenis jaringan, elemen jaringan, pemodelan jaringan, dan pemodelan lalu lintas jaringan. NS-2 merupakan perangkat lunak dengan lisensi *open-source* yang berada di bawah GNU *Public License* (GPL), sehingga dapat diunduh secara gratis melalui situs web resminya di https://www.isi.edu/nsnam/ns/.

2) Sistem Operasi Linux Ubuntu versi 14.04 LTS 64 bit

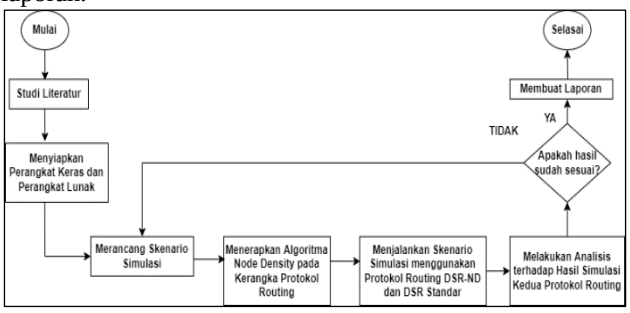
Software yang digunakan untuk menjalankan Network Simulator NS-2.35.

3) *Oracle* VM *Virtual Box* versi 5.2.44

Software virtualisasi yang digunakan untuk menjalankan sistem operasi Linux Ubuntu 14.04 LTS.

B. Alur Penelitian

Alur penelitian yang dilakukan dapat dilihat pada Gambar 2. Diagram alir penelitian diawali dari studi *literature* sampai dengan tahap akhir yaitu membuat laporan.

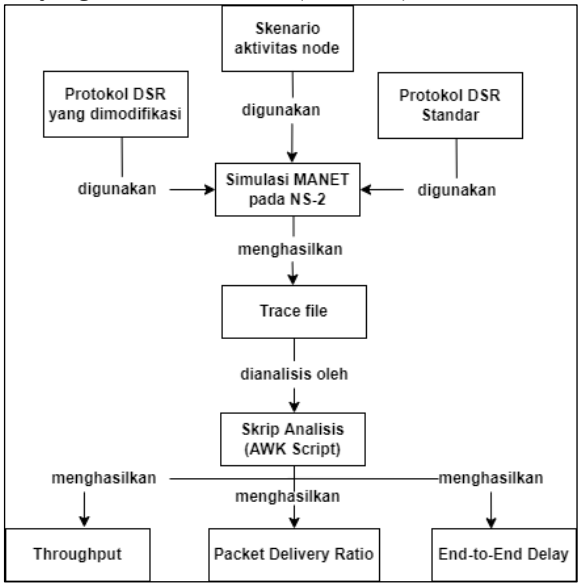


Gambar 1 Diagram Alir Penelitian.

C. Peracangan Simulasi Jaringan

Proses perancangan dimulai dengan membuat skenario aktivitas node menggunakan bahasa pemrograman TCL, termasuk model mobilitas, model komunikasi, kecepatan *node*, dan kapasitas jaringan. Dalam protokol *routing* DSR, modifikasi dilakukan dengan menyisipkan algoritma ND menggunakan bahasa pemrograman C. Simulasi jaringan MANET kemudian dilakukan berdasarkan skenario yang telah dibuat, menggunakan *protokol routing* DSR standar dan protokol routing DSR-ND. Selama simulasi berjalan, kegiatan akan berhenti secara otomatis berdasarkan waktu simulasi yang telah ditentukan.

Setelah simulasi selesai, akan dihasilkan *trace file* dengan ekstensi .tr yang berisi data mentah hasil simulasi jaringan. Data dari *trace file* akan di*filter* menggunakan bahasa pemrograman AWK untuk mendapatkan parameter uji yang diinginkan, seperti *throughput*, *end-to-end delay*, dan *packet delivery ratio*. Langkah terakhir dari simulasi jaringan melibatkan presentasi data dalam bentuk grafik dan tabel parameter uji, serta analisis dan perbandingan kinerja antara protokol routing DSR standar dan protokol DSR yang telah dimodifikasi (DSR-ND).



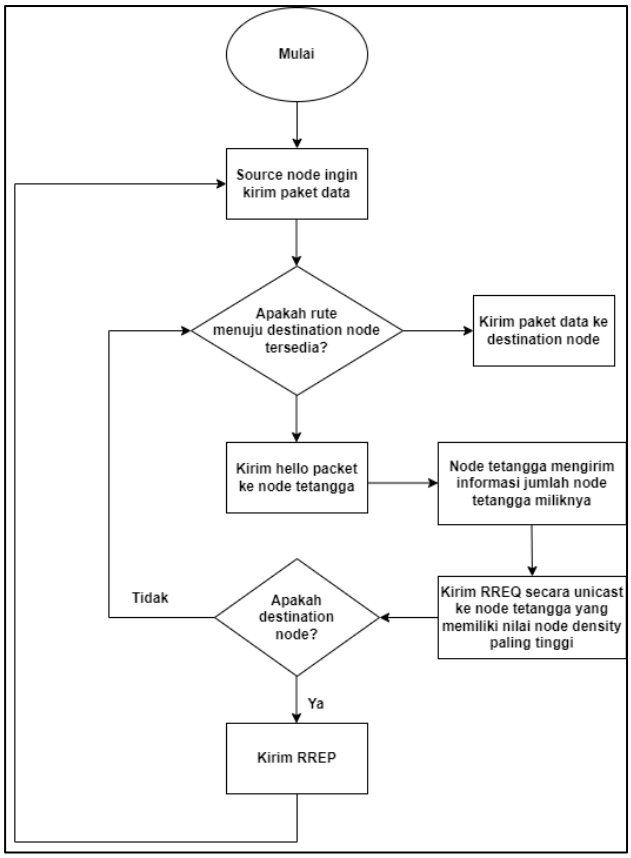
Gambar 2 Tahap perancangan simulasi jaringan.

D. Metode yang Diajukan

Pada penelitian ini, diajukan penggunaan metode *Dynamic Source Routing-Node Density* (DSR-ND) untuk meningkatkan kinerja protokol routing DSR standar dalam pencarian *rute* yang lebih efisien berdasarkan tingkat kepadatan *node* dalam jaringan MANET. Metode yang diusulkan akan menggabungkan protokol *routing* DSR standar dengan metode *Node Density*, sehingga terbentuk protokol DSR-ND.

Dalam metode yang diusulkan, cakupan *range* komunikasi akan ditentukan dengan menghitung tingkat kepadatan *node* sebelum paket data dikirimkan. Algoritma *Node Density* akan melakukan penyesuaian otomatis terhadap cakupan range komunikasi berdasarkan tingkat kepadatan *node*.

Gambar 3 menunjukkan metode alur kerja yang digunakan. *Node* pengirim akan mengirimkan paket hello kepada *node* tetangga untuk mendapatkan informasi tentang jumlah *node* di sekitarnya dan jangkauan komunikasi. *Node* tetangga akan merespons dengan mengirim paket *hello* kembali kepada *node* pengirim, dan informasi tentang semua *node* tetangga akan disimpan dalam tabel *node neighbor*. *Node* pengirim akan menghitung jumlah *node* tetangga yang terhubung untuk setiap *node* dan menghitung kepadatan *node*. Selanjutnya, *node* dengan kepadatan tertinggi akan dipilih sebagai *node* perantara, dan *node* perantara ini akan menerima pesan RREQ dari *node* dengan kepadatan tertinggi.

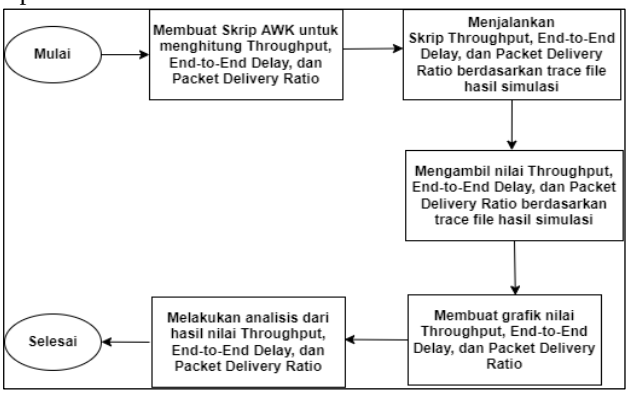


Gambar 3 Alur kerja metode yang digunakan.

E. Cara Analisis

Hasil simulasi menggunakan NS-2 akan menghasilkan *output* dalam bentuk file teks dengan ekstensi .tr, yang disebut sebagai *trace file*, yang berisi informasi tentang aktivitas yang terjadi selama simulasi berlangsung. Dari data yang dihasilkan tersebut, dapat dilakukan analisis kinerja protokol *routing* DSR-ND dengan mengukur beberapa metrik yang relevan. Dalam penelitian ini, metrik yang akan diukur meliputi *throughput*, *end-to-end delay*, dan *packet delivery ratio*. Analisis kinerja akan dilakukan menggunakan *script* AWK yang akan menghasilkan *output* berupa nilai-nilai metrik parameter pengujian. AWK merupakan sebuah bahasa pemrograman yang dirancang khusus untuk pemrosesan teks, dan sering digunakan untuk ekstraksi dan pelaporan data dalam lingkungan mirip sistem operasi Unix.

Gambar 4 merupakan tahapan yang dilakukan untuk menganalisis kinerja protokol *routing* yang diusulkan. Langkah pertama adalah membuat *script* AWK yang akan digunakan untuk menghitung nilai *throughput*, *end-to-end delay*, dan *packet delivery ratio*. Selanjutnya, *script* tersebut akan dijalankan dengan menggunakan terminal pada sistem operasi Linux Ubuntu 14.04, dan akan menghasilkan nilai-nilai parameter pengujian berdasarkan data yang terdapat dalam *trace file* hasil simulasi. Setelah itu, langkah berikutnya adalah mengekstraksi nilai-nilai *throughput*, *end-to-end delay*, dan *packet delivery ratio* dari *trace file* hasil simulasi. Kemudian, menggunakan perangkat lunak Microsoft Excel, nilai-nilai metrik tersebut akan digunakan untuk membuat grafik yang menggambarkan performa *protokol routing* dalam parameter pengujian yang telah ditentukan. Langkah terakhir adalah melakukan analisis terhadap grafik yang telah dibuat untuk menjelaskan hasil penelitian dalam laporan secara naratif.



Gambar 4 Diagram alir tahapan analisis.

F. Parameter Skenario Simulasi

Pada langkah ini, dilakukan pengaturan parameter dan skenario simulasi. Skenario pengujian ditentukan menggunakan versi 2.35 dari simulator NS-2 (Network Simulator 2). Dalam pengujian ini, dua jenis protokol routing yang digunakan adalah DSR standar dan DSR-ND yang telah dimodifikasi. Durasi simulasi yang dipilih adalah 200 detik. Area jaringan simulasi berbentuk persegi dengan ukuran 1500m x 1500m, dan jumlah *node density* yang digunakan adalah 25, 50, dan 100 *node*. Selama simulasi berlangsung, *node-node* tersebut bergerak dengan kecepatan 5m/s. Protokol MAC yang digunakan dalam skenario simulasi didasarkan pada standar IEEE 802.11, dengan penggunaan LAN nirkabel sebagai lapisan MAC. Jenis antena yang digunakan adalah *Omni Antena*, sedangkan tipe kanal yang digunakan adalah *Wireless Channel*. Dalam simulasi ini, jenis mobilitas yang digunakan adalah *Random Waypoint*.

Parameter skenario simulasi yang lebih lengkap dapat dilihat pada Tabel 1.

Tabel 1 Parameter skenario simulasi.

Parameter Skenario	
Tipe Parameter	**Nilai Parameter**
Network Simulator	NS-2 versi 2.35
Routing protocol	DSR-ND dan DSR
Waktu Simulasi	200 detik
Area Simulasi	1500 m x 1500 m
Banyak Node	25, 50, 100
Kecepatan Pergerakan Node	5 m/s
Protokol MAC	IEEE 802.11
Tipe Antena	OmniAntenna
Tipe Kanal	Wireless Channel
Mobility model	Random Waypoint

V. HASIL DAN PEMBAHASAN

A. Implementasi Metode

Pada tahapan ini akan dijelaskan implementasi metode yang diusulkan pada penelitian ini. Implementasi

metode menggunakan bahasa pemrograman C++. Kode implementasi dari routing protokol DSR pada NS-2 versi 2.35 berada pada direktori ns-2.35/dsr. Di dalam direktori tersebut terdapat beberapa file diantaranya seperti `dsragent.h`, `dsragent.cc`, `mobicache.h`, `mobicache.cc`, `cache.h`, `cache.cc` dan sebagainya.

Untuk menghitung jumlah tetangga *node*, dapat dilakukan dengan menghitung *node-node* yang berada dalam jangkauan komunikasi dari *node* yang sedang aktif. Sebelum mengirim paket RREQ, suatu *node* harus mengetahui *node-node* tetangganya dengan mengirim paket Hello. Proses menghitung jumlah *node* tetangga diimplementasikan pada file `dsragent.cc` di dalam fungsi `DSRAgent::recv()`.

Langkah dalam menghitung jumlah *node* tetangga adalah dengan mendeklarasikan variabel bertipe *integer* berupa *array* yang bernama `neighbor_counter` untuk menjadi sebuah penghitung jumlah tetangga, kemudian variabel `neighbor_id` untuk mendapatkan *node-node* mana yang sedang dieksekusi yang berada dalam fungsi `net_id.dump()`.

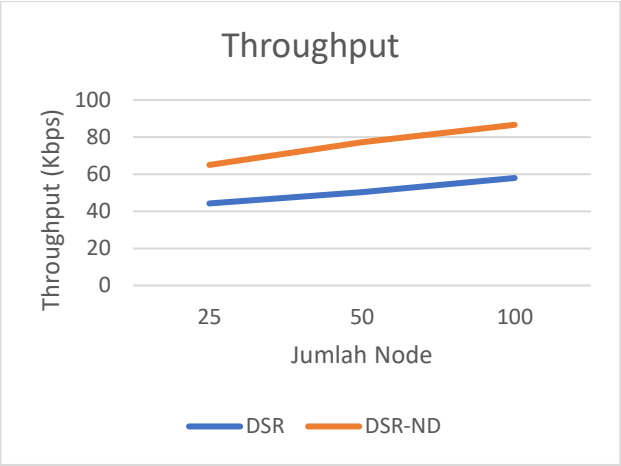
Node yang mengirim paket Hello ke *node* tetangga akan mendapat balasan berupa paket Acknowledgement. Setelah itu akan dihitung jumlah *node* tetangganya. Langkah berikutnya adalah mencari *node* tetangga yang memiliki jumlah *node* tetangga paling besar. *Node* yang memiliki jumlah *node* tetangga paling besar akan dikirimi paket RREQ, begitu seterusnya hingga paket RREQ tiba di *node* tujuan.

Dalam menentukan skenario ada dua hal yang ditentukan yaitu menentukan pola *traffic* yang dilakukan dengan menjalankan file *"cbrgen.tcl* dan menentukan pergerakan node dilakukan dengan menjalankan file *"setdest.tcl"*.

B. Hasil Simulasi

Dalam hasil simulasi, akan ditampilkan hasil uji coba yang diperoleh dari beberapa percobaan simulasi yang dilakukan sebanyak 3 kali pada setiap skenario. Tujuan dari ini adalah untuk memastikan konsistensi dan keakuratan data dalam membandingkan kinerja protokol *routing* DSR standar dengan protokol *routing* DSR yang telah dimodifikasi. Parameter uji yang digunakan untuk menganalisis kinerja protokol *routing* meliputi *throughput*, *packet delivery ratio*, dan *end-to-end delay*. Hasil uji coba ini akan disajikan dalam bentuk tabel. Selanjutnya, analisis grafik hasil simulasi akan menjelaskan tentang hasil-hasil yang diperoleh dari simulasi yang telah dilakukan. Grafik tersebut akan menunjukkan hasil rata-rata dari 3 percobaan yang dilakukan. Untuk mengukur kinerja protocol *routing* DSR standar dan DSR yang telah dimodifikasi, dilakukan uji coba dengan menggukan parameter *throughput*, *end-to-end delay*, dan *packet delivery ratio*.

B.1 Hasil Protokol Routing Terhadap Throughput

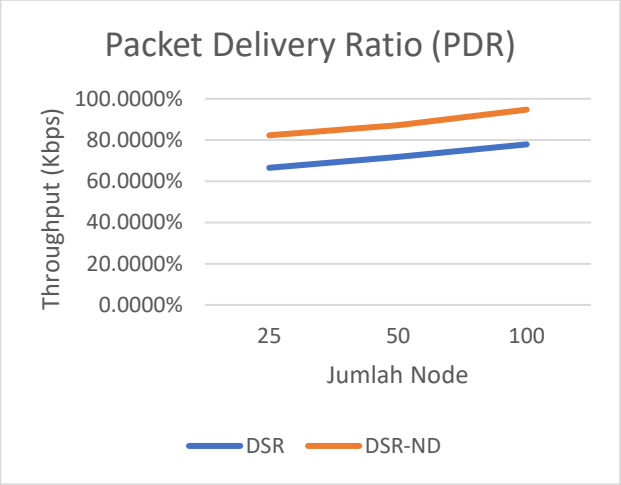


Gambar 5 Hasil rata-rata uji coba *throughput*

Gambar 5 merupakan grafik nilai rata-rata hasil uji coba *throughput* pada percobaan 1, percobaan 2, dan percobaan 3 yang telah dilakukan.

Rata-rata hasil uji coba untuk 25 *node* pada DSR-ND adalah 64.99 Kbps dan DSR Standar 44.23 Kbps. Rata-rata hasil uji coba untuk 50 *node* pada DSR-ND adalah 77.22 Kbps dan DSR Standar 50.24 Kbps. Kemudian rata-rata hasil uji coba untuk 100 *node* pada DSR-ND adalah 86.63 Kbps dan DSR Standar adalah 57.94 Kbps. Nilai throughput dapat dipengaruhi oleh beberapa faktor antara lain jumlah *node* dan kepadatan lalu lintas *node*. Hal tersebut dapat dibuktikan jika jumlah *node* bertambah maka kepadatan lalu lintas meningkat dan mengakibatkan nilai *throughput* tinggi[17]. Pada grafik terlihat bahwa seiring dengan meningkatnya jumlah *node* pada simulasi, maka nilai *throughput* mengalami kenaikan. Protokol DSR-ND memberikan kenaikan nilai *throughput* yang lebih baik karena mampu mengurangi *broadcast* paket sehingga lalu lintas atau beban trafik menjadi lebih sedikit. Rata-rata *throughput* pada protokol *routing* DSR-ND mampu memberikan kenaikan nilai *throughput* sebesar 0.50 %.

B.2 Hasil Protokol Routing Terhadap Packet Delivery Ratio

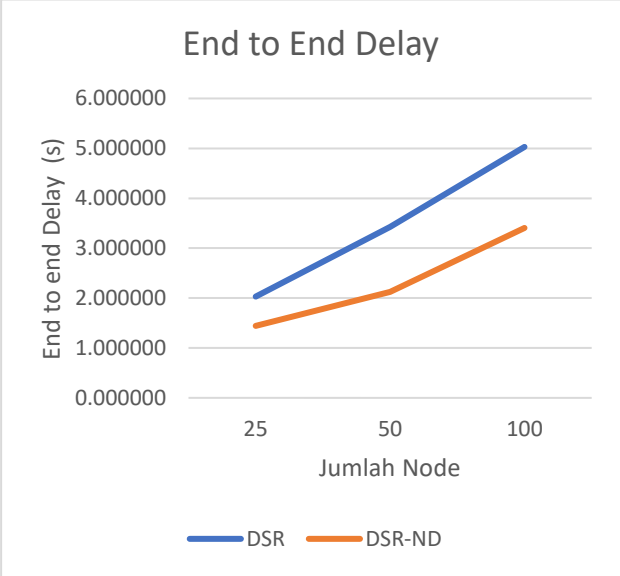


Gambar 6 Hasil rata-rata uji coba PDR

Gambar 6 merupakan grafik nilai rata-rata hasil uji coba *packet delivery ratio* (PDR) pada percobaan 1, percobaan 2, dan percobaan 3 yang telah dilakukan.

Rata-rata hasil uji coba untuk 25 *node* pada DSR-ND adalah 82.2944 % dan DSR Standar 66.5255 %. Rata-rata hasil uji coba untuk 50 *node* pada DSR-ND adalah 87.2081 % dan DSR Standar 71.8278 %. Kemudian rata-rata hasil uji coba untuk 100 *node* pada DSR-ND adalah 94.7366 % dan DSR Standar adalah 77.8851 %. Nilai PDR meningkat seiring bertambahnya jumlah *node* dikarenakan adanya peningkatan jumlah rute yang tersedia antara *node* sumber dan *node* tujuan. Apabila rute utama terputus maka proses pencarian rute baru akan mudah diperoleh karena banyak rute alternatif ditemukan. Selain itu, peningkatan jumlah *node* menyebabkan lebih banyak *node* yang berfungsi sebagai *node* perantara dalam mengirimkan paket data. Protokol routing DSR-ND mampu memberikan nilai PDR yang lebih baik seiring meningkatnya nilai *throughput* sehingga paket-paket yang lewat melalui *link-link node* intermediate pada rute tersebut diterima dengan baik. Rata-rata PDR pada protokol *routing* DSR-ND mampu memberikan kenaikan nilai PDR sebesar 0.22 %.

B.3 Hasil Protokol Routing Terhadap End to End Delay



Gambar 7 Hasil rata-rata uji coba *end to end delay*

Gambar 7 merupakan grafik nilai rata-rata hasil uji coba *end to end delay* pada percobaan 1, percobaan 2, dan percobaan 3 yang telah dilakukan.

Rata-rata hasil uji coba untuk 25 *node* pada DSR-ND adalah 1.44252s dan DSR Standar 2.02974s. Rata-rata hasil uji coba untuk 50 *node* pada DSR-ND adalah 2.12121s dan DSR Standar 3.42498s. Kemudian rata-rata hasil uji coba untuk 100 *node* pada DSR-ND adalah 3.40515s dan DSR Standar adalah 5.02912s. Pada grafik Gambar 4.4 terlihat bahwa seiring dengan meningkatnya jumlah *node* pada simulasi, maka nilai *end-to-end delay* mengalami kenaikan. *End to end delay* merupakan rata-rata jumlah waktu yang didapatkan untuk sebuah paket ke *node* tujuan[18]. Kenainakan nilai *end-to-end delay* disebabkan karena semakin banyak jumlah *node* maka akan bertambah pula jumlah node *intermediate* yang berpartisipasi pada rute tersebut. Protokol DSR-ND mampu memiliki nilai *end-to-end delay* yang kecil dibandingkan dengan protokol DSR standar karena memiliki kelebihan mengurangi broadcast paket dan tidak perlu melakukan pencarian rute dari awal ketika rute terputus. Rata-rata *delay* pada protokol *routing* DSR-ND mampu memberikan penurunan nilai *delay* yang rendah yaitu sebesar 0.50 %.

VI. KESIMPULAN DAN SARAN

A. Kesimpulan

1) Pada hasil uji coba penelitian, penerapan algoritma *node density* pada protokol *routing* DSR mampu meningkatan kinerja protokol *routing* DSR standar. Rata-rata *throughput* menunjukkan hasil peningkatan dengan nilai sebesar 0.50 %, *packet delivery ratio* (PDR) 0.22 % dan *end to end delay* menunjukkan hasil penurunan sebesar 0.50 %.
2) Jumlah *node* yang semakin bertambah memberikan pengaruh juga terhadap algoritma yang diterapkan. Hal ini terlihat dari nilai parameter uji *throughput*, *packet delivery ratio*, dan *end to end delay*. Namun, algoritma *node density* mampu memberikan hasil yang lebih baik dari protokol DSR standar.

B. Saran

1) Melakukan percobaan berikutnya dengan menggunakan algoritma *node density* pada protokol *routing* yang berbeda.
2) Melakukan percobaan berikutnya dengan menggunakan algoritma yang sama pada protokol *routing* yang sama, tetapi dengan tambahan algoritma lain dan variasi parameter simulasi yang berbeda.

REFERENCES

[1] D. Ramphull, A. Mungur, S. Armoogum, and S. Pudaruth, “A Review of Mobile Ad hoc NETwork (MANET) Protocols and their Applications,” in *Proceedings - 5th International Conference on Intelligent Computing and Control Systems, ICICCS 2021*, Institute of Electrical and Electronics Engineers Inc., May 2021, pp. 204–211. doi: 10.1109/ICICCS51141.2021.9432258.

[2] F. Lee, *Mobile Ad-Hoc Networks: Protocol Design*. 2011. Accessed: May 27, 2023. [Online]. Available: http://www.intechopen.com/books/mobile-ad- hoc-networks-protocol-design/routing-in-mobile-ad-hoc-networks

[3] M. R. Pearlman and Z. J. Haas, “Determining The Optimal Configuration for The Zone Routing Protocol,” *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1395–1414, Aug. 1999, doi: 10.1109/49.779922.

[4] S. D. Anggraini, K. Nugroho, and E. F. Cahyadi, "Analisis Perbandingan Performasi Protokol Routing AODV Dan DSR Pada Mobile Ad-Hoc Network (MANET)," *2nd Seminar Nasional IPTEK Terapan (SENIT)*, pp. 212–218, 2017, [Online]. Available: https://www.researchgate.net/publication/318982812

[5] R. A. Putri, J. Jusak, and A. Sukmaaji, "Analisis Perbandingan Kinerja Protokol On-Demand Routing Pada Jaringan Sensor Nirkabel Ad Hoc," *Journal of Control and Network Systems*, vol. 2, no. 1, pp. 16–25, 2013, [Online]. Available: http://jurnal.stikom.edu/index.php/jcone

[6] H. Maghfira, "Implementasi Adaptif Zonasi Pada Zone Routing Protocol Berdasarkan Tingkat Kepadatan Node Tetangga," *Fakultas Teknologi Informasi dan Komunikasi Institut Teknologi Sepuluh Nopember*, 2019, Accessed: May 25, 2023. [Online]. Available: https://repository.its.ac.id/60446/

[7] E. Natsheh and K. Buragga, "Density Based Routing Algorithm for Spare/Dense Topologies in Wireless Mobile Ad-Hoc Networks," *American Journal of Engineering and Applied Sciences*, vol. 3, no. 2, pp. 312–319, Feb. 2010, doi: 10.3844/ajeassp.2010.312.319.

[8] A. Quintero, S. Pierre, and B. Macabéo, "A Routing Protocol Based on Node Density For Ad Hoc Networks," *Ad Hoc Networks*, vol. 2, no. 3, pp. 335–349, Jul. 2004, doi: 10.1016/j.adhoc.2004.03.003.

[9] M. Syaifuddin, P. H. Trisnawan, and R. A. Siregar, "Analisis Pengaruh Kepadatan Node terhadap Kinerja Protokol Routing DYMO dan DSR Pada Mobile Ad-Hoc Network (MANET)," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 4, pp. 3433–3442, 2019, [Online]. Available: http://j-ptiik.ub.ac.id

[10] G. Paliwal and S. Taterh, "A Topology Based Routing Protocols Comparative Analysis for MANETs," *International Journal of Advanced Engineering Research and Science (IJAERS)*, vol. 3, no. 4, 2016, [Online]. Available: www.ijaers.com

[11] M. Nurusshobah, P. Hari Trisnawan, and K. Amron, "Analisis Kinerja Protokol Routing Dynamic MANET On-Demand (DYMO) dan Cluster Based Routing Protocol (CBRP) pada Mobile Ad-Hoc Network (MANET)," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 4, pp. 3563–3572, 2019, [Online]. Available: http://j-ptiik.ub.ac.id

[12] S. F. Ramadhan, M. S. Iqbal, and A. S. Rachman, "Performance of Routing Protocol DSDV, DSR and AODV on Mobile Ad hoc Network with NS-2," *Dielektrika*, vol. 5, no. 2, pp. 133–141, 2018.

[13] A. Dianingrum and I. Nurcahyani, "Analisis Pengaruh Kelas Protokol Routing Terhadap Kinerja Jaringan Mobile Ad Hoc Network (MANET)," Yogyakarta, 2018.

[14] P. D. Pradana, R. M. Negara, and F. Dewanta, "Evaluation of Performance of Routing Protocol DSR and AODV in Network Simulation of Vehicular Ad-Hoc Network (VANET) for Transportation Safety With Study Case of City Car," *e-Proceeding of Engineering*, vol. 4, no. 2, pp. 1996–2004, 2017.

[15] E. H. Harapah, "Performance Analysis AODV(Ad Hoc On Demand Distance Vector) And DSR (Dynamic Source Routing) Protocol to Active Attack I MANET (Mobile Ad Hoc Network) In Term Of Network QOS (Quality Of Service)," *e-Proceeding of Engineering*, vol. 1, no. 1, pp. 118–125, 2014.

[16] W. Febdila, "Studi Kinerja Protokol MAODV dengan Propagasi Model Two Ray Ground dan Free space Pada Jaringan MANET Menggunakan Network Simulator 2 (NS-2)," Surabaya, 2016.

[17] S. P. Waskito, A. H. Jatmika, and A. Zubaidi, "Implementasi Algoritma Pemilihan Node Tetangga Terbaik Pada Protokol Routing DSR di Jaringan MANET," *(Journal of Computer Science and Informatics Engineering*, vol. 6, no. 1, pp. 64–72, 2022, [Online]. Available: http://jcosine.if.unram.ac.id/

[18] L. Ikhwan Rosadi, A. Hidayat Jatmika, and S. Endang Anjarwani, "Pengaruh Penerapan Algoritma Clustering COEEC terhadap Kinerja Protokol Routng DSR di Jaringan MANET," *Jurnal Teknologi Informasi, Komputer, dan Aplikasinya (JTIKA)*, vol. 1, no. 2, 2019, doi: https://doi.org/10.29303/jtika.v1i2.33.