PENERAPAN ALGORITMA SEMUT PADA PROTOKOL *ROUTING* AOMDV UNTUK OPTIMASI PENCARIAN RUTE DI JARINGAN VANET

(Application of Ant Colony Optimization Algorithm to AOMDV Routing Protocol for Optimization of Route Search on VANET)

Ni Desak Ketut Pujika Dewi*, Andy Hidayat Jatmika, Ariyan Zubaidi
Dept Informatics Engineering, Mataram University
Jl. Majapahit 62, Mataram, Lombok NTB, Indonesia
Email: fujikadewi@gmail.com, [andy, ariyan.zubaidi]@unram.ac.id

Abstract

VANET is a derivative of the MANET network. VANET networks are more specifically used to communicate between one vehicle and another vehicle. With the existence of the VANET network, it is expected to be able to improve the safety of drivers on the highway. In the application of the VANET network, a routing protocol is needed. The AOMDV protocol is an example of a routing protocol that can help the performance of a VANET network. The purpose of this research is to optimize the route search in the AOMDV routing protocol. The route selection process in the AOMDV routing protocol is based on the least number of hops, so the route is likely to be disconnected quickly. To overcome the shortcomings of AOMDV routing protocol, the ant colony optimization algorithm is applied. The advantage of an ant colony optimization algorithm is that in conducting a route search, performed by calculating the distance between nodes so that the route is not quickly interrupted. Selection of the route on the ant colony optimization algorithm based on the most pheromones, pheromones are ant footprints. The author modifies some of the frameworks of the AOMDV routing protocol section named AOMDV-ant. In the study conducted 5 experiments. The trial parameters used to assess network performance is throughput, packet delivery ratio (PDR), and average end-to-end delays. From the results of trials that have been carried out on the AOMDV routing protocol using the ant colony optimization algorithm, it can improve the performance of the test throughput parameters of 9.7649 Kbps and the performance of the Packet delivery ratio of 11.2838%. Whereas the average end to end delay parameter can reduce the delay by 13.3093ms.

Keywords: *VANET, AOMDV, Algoritma Semut, Protokol Routing, Pencarian Rute*

**Penulis Korespondensi*

1. PENDAHULUAN

Vehicular Ad Hoc Network (VANET) merupakan salah satu jaringan *wireless*, jaringan ini merupakan turunan dari jaringan MANET. Jaringan VANET adalah suatu jaringan *ad hoc* yang dapat digunakan untuk berkomunikasi antar kendaraan satu dengan kendaraan lainnya [1]. Performa suatu jaringan ditentukan oleh kinerja protokol *routing* yang digunakan, *routing* ialah suatu proses yang dilakukan untuk menetukan rute dari *node* sumber ke *node* tujuan sehingga pada protokol *routing* inilah yang akan bertugas untuk menentukan suatu rute. *Ad Hoc On-Demand Multipath Distance Vector* (AOMDV) merupakan salah satu contoh dari protokol *routing* yang digunakan pada jaringan VANET.

Protokol *routing* AOMDV pada saat melakukan *route discovery* atau proses pencarian rute dari *node* sumber ke *node* tujuan dilakukan dengan mengirimkan paket RREQ (*Route Request*) ke *node* tetangganya. Jika paket RREQ telah diterima oleh *node* tujuan, maka *node* tujuan akan mengirimkan paket RREP (*Route Reply*) ke *node* sumber. Pada protokol *routing* AOMDV setiap melakukan proses RREP akan dipertimbangkan oleh *node* tujuan, sehingga beberapa *path* bisa ditemukan dalam satu proses pencarian rute [2]. Proses pemilihan rute pada protokol *routing* AOMDV dilakukan dengan mempertimbangkan jumlah *hop* pada rute tersebut. Jumlah *hop* yang paling sedikit yang akan digunakan untuk memilih rute pada protokol routing AOMDV. Kekurangan dari protokol *routing* AOMDV yaitu rute yang dipilih tidak memperhitungkan jarak antar *node*, sehingga dapat mengakibatkan rute tersebut memiliki kemungkinan terputus. Untuk mengatasi kekurangan dari protokol *routing* AOMDV tersebut diperlukannya sebuah metode. Algoritma semut merupakan salah satu metode yang digunakan untuk proses pencarian rute yang optimal.

Algoritma semut (*Ant Colony Optimization)* merupakan algoritma yang diambil dari perilaku semut atau koloni semut untuk mencari makanan. Pada algoritma semut memperhitungkan jarak antar *node*, menghitung probabilitas untuk menentukan arah rute yang akan dituju sehingga lintasan semut dapat diketahui. Dalam algoritma semut terdapat istilah feromon yaitu jejak kaki semut yang melintasi rute, koloni semut dapat menemukan rute terpendek dengan mengikuti jejak kaki semut sebelumnya yang telah melintasi rute tersebut. Pada feromon tersebut dilakukan pembaharuan feromon (*update* feromon) untuk mendapatkan rute yang paling sering dilintasi oleh semut. Semakin banyak semut yang melewati rute tersebut maka semakin jelas feromon atau jejak kaki semut pada lintasan tersebut. Tujuan dari penelitian ini yaitu untuk menemukan proses pencarian rute yang optimal dengan menerapkan algoritma semut pada protokol *routing* AOMDV. Rute yang dipilih adalah berdasarkan jumlah feromon yang paling banyak yang dimiliki oleh algoritma semut, bukan lagi berdasarkan jumlah *hop* yang dimiliki oleh protokol *routing* AOMDV. Protokol *routing* yang telah dimodifikasi diberi nama AOMDV-Semut.

Oleh karena itu pada penelitian ini akan membandingkan kinerja dari protokol *routing* AOMDV standar dengan protokol *routing* AOMDV yang telah dimodifikasi dengan algoritma semut (AOMDV-Semut). Pemilihan peta pada penelitian ini yaitu mengambil peta jalan Kota Mataram, daerah tersebut dipilih karena memiliki karakteristik jalan persimpangan yang lebih dari satu. Simulasi dilakukan dengan menggunakan *Network Simulator* 2 (NS-2) versi 2.35, *Java OpenSteetMap Editor* (JOSM) dan *Simulation of Urban Mobility* (SUMO). NS2 merupakan perangkat lunak didesain secara spesifik untuk penelitian dalam bidang jaringan komunikasi komputer yang bersifat *open source*. JOSM merupakan aplikasi yang digunakan untuk meng-*edit* peta yang diperoleh melalui *OpenStreetMap*. SUMO merupakan aplikasi yang bersifat *open source* digunakan untuk membuat skenario *mobilitas* serta dapat diintegrasikan dengan *OpenStreetMap* untuk menetukan lokasi simulasi yang diinginkan. Parameter uji coba yang digunakan untuk menilai kinerja jaringan adalah *throughput*, *Packet Delivery Ratio (PDR)* dan *End To End Delay*.

2. TINJAUAN PUSTAKA

Penelitian yang berjudul "Analisis Performansi *Routing Protocol* OLSR dan AOMDV pada *Vehicular Ad Hoc Network* (VANET)" [2] yaitu melakukan penelitian dengan membandingkan protokol *routing* OLSR dan AOMDV di jaringan VANET. Setelah dilakukan simulasi dan analisa terhadap kedua algoritma *routing protocol* yaitu OLSR dan AOMDV, maka dapat diambil kesimpulan bahwa AOMDV lebih unggul hampir pada semua metrik performansi dengan nilai nilai rata-rata PDR 87.804%, *throughput* 449.565 kbps, *routing overhead* 0.9773, dan NRL 1.1108. Sedangkan pada OLSR memiliki rata-rata PDR 83.539%, *throughput* 427.735 kbps, *routing overhead* 1.4523, NRL 1.7436. Pada metrik performansi *End To End Delay* OLSR lebih unggul dengan memiliki nilai rata-rata 4.765 ms. Sedangkan pada AOMDV memiliki nilai 8.215 ms. Karena AOMDV lebih unggul empat dari lima metrik performansi yang diujikan ini dapat menunjukkan bahwa protokol *routing* AOMDV lebih efisien diterapkan pada jaringan *vehicular ad-hoc network* (VANET) pada kondisi perkotaan.

Penelitian yang berjudul "*Performance Improvement of Dynamic Source Routing (DSR) Protocol using Ant Colony Optimization for Vehicular Ad-hoc Network (VANET)"* [3] yaitu melakukan analisis terhadap protokol *routing* DSR standar dengan protokol *routing* DSR yang telah dimodifikasi dengan menggunakan algoritma semut (DSR-Ant). Prameter uji coba yang dihitung yaitu *Delay*, *jitter*, konsumsi energi, *routing load* dan *Packet Delivery Ratio.* Hasil menunjukan bahwa DSR-Semut lebih unggul dibandingkan DSR standar dilihat dari metrik performasi *Delay* 2,25 ms, jitter 1,2 ms, *routing* load 0,36 Kbps, dan PDR 83,5%. Sedangkan DSR standar hanya unggul pada metrik perfomansi konsumsi energi yaitu 15,75 joule.

Penelitian yang berjudul *"Comparative Analysis of Various Routing Protocols in VANET"* [4] yaitu melakukan penelitian dengan membandingkan kinerja dari protokol *routing* AODV, AOMDV, DSR dan DSDV di jaringan VANET. Parameter uji yang digunakan yaitu *Delay, Packet Delivery Ratio, throughput, normalized routing load, packet loss.* Hasil menunjukan bahwa protokol *routing* AOMDV memiliki kinerja yang baik dilihat dari PDR 98,55498%, *packet loss* 1,4512%, dan NRL 0,5558. DSR unggul pada metrik perfomasni *Delay* dan *throughput* yaitu 3,09288 m/s dan 1833,24 Kbps. Dengan demikian dapat disimpulkan bahwa AOMDV lebih unggul dari hasil uji yang telah dilakukan, dilihat dari hasil uji PDR, *packet loss* dan NRL, sehingga ini dapat menunjukkan bahwa protokol *routing* AOMDV lebih efisien diterapkan pada jaringan *vehicular ad-hoc network* (VANET).

Penelitian yang berjudul "*Ant Colony Optimization based Modified* AOMDV *for Multipath Routing in MANET"* [5] yaitu menerapkan algoritma semut pada

routing protokol AOMDV di jaringan MANET. Pada penelitian ini memodifikasi protokol *routing* AOMDV dengan menggunakan algoritma semut yang dimana menggunakan AODV-Ant sebagai perbandingannya. Hasil menunjukan bahwa protokol *routing* AOMDV-Ant lebih unggul dibandingkan AODV-Ant dilihat dari metrik perfomansi UDP data *transmission analysis,* UDP data *Receives analysis*, *routing overhead, normalized routing load, Packet Delivery Ratio*, dan *no.of dropped data (packets) dengan nilai rata-rata sebesar* 6694 Bps, 6325 Bps, 4088 Bps, 0,65, 94,49%, 364 paket *drop.* Karena AOMDV-Ant unggul dari ke enam parameter uji yang telah dilakukan, dapat ditarik kesimpulan bahwa AOMDV-Ant memiliki kinerja yang lebih baik dibandingkan dengan AODV-Ant.

Penelitian yang berjudul "*AODV Extension using Ant Colony Optimization for Scalable Routing in VANETs*" [6] yaitu melakukan modifikasi protokol *routing* AODV dengan menggunakan algoritma semut. Penerapan optimasi koloni semut ke dalam protokol *routing* AODV digunakan untuk menentukan rute dan melakukan perbaikan jika terjadi kegagalan rute. Modifikasi AODV dengan algoritma semut dapat mengurangi *routing overhead* dan meningkatkan kinerja dengan menghindari kegagalan rute yang sering terjadi.

3. METODE PENELITIAN

3.1 Alur Penelitian

3.1.1 Studi Literatur

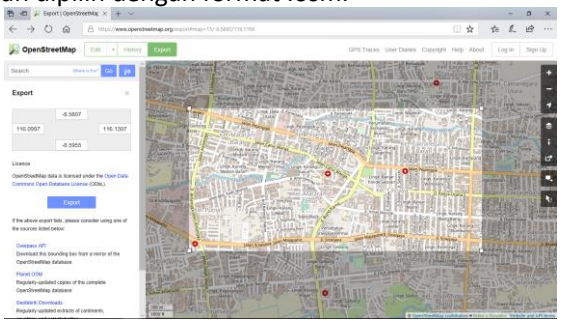
Pada penelitian ini dilakukan pembelajaran terhadap penelitian-penelitian terkait yang telah dilakukan sebelumnya sebagai dasar dalam melakukan penelitian yang akan dilakukan. Sumber penelitian sebelumnya dapat berupa paper, skripsi, tesis, maupun buku yang dapat menunjang serta mempermudah penelitian yang akan dilakukan.

3.1.2 Mempesiapkan Hardware dan Software

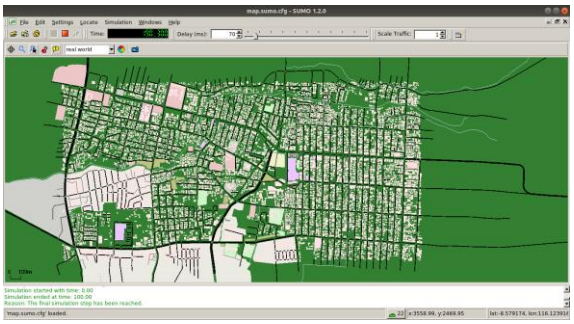
Pada bagian ini, perlu mempersiapkan baik perangkat keras maupun perangkat lunak yang akan digunakan pada penelitian ini. Perangkat keras yang digunakan seperti laptop dan perangkat lunak yang digunakan seperti *Network Simulator* 2 (NS2), *Simulation of Urban Mobility* (SUMO), *Java OpenStreetMap* (JOSM) dan *Microsoft Office Excel*.

3.1.3 Melakukan Pemilihan Peta

Pada tahap ini, peneliti melakukan proses pemilihan peta, penelitian ini menggunakan peta Kota Mataram khususnya jalan yang terhubung antara Jl. Pejanggik, Jl. Catur Warga, Jl. Panca Usaha, Jl. Bung Karno, Jl. Sriwijaya Majapahit. Peta jalan dapat dilihar dari website *http://openstreetmap.org*. Hasil dari proses ini berupa peta daerah Kota Mataram yang telah dipilih dengan format .osm.



Gambar 1. Pemilihan lokasi peta melalui *openstreetmap*



Gambar 2. Simulasi jaringan lalu lintas pada SUMO : *mode real world*

3.1.4 Membuat *Script* Algoritma Semut pada Protokol *Routing* AOMDV

Pada tahap ini, dilakukan modifikasi *script* pada protokol AOMDV dengan menggunakan algoritma semut. Modifikasi ini diharapkan dapat mengoptimalkan kinerja protokol AOMDV dalam melakukan pencarian rute pada jaringan VANET. Modifikasi yang dilakukan yaitu dengan melakukan perubahan pada proses *route discovery* AOMDV.

3.1.5 Membuat *Script* Simulasi VANET

Pada tahap ini dilakukan perancangan simulasi jaringan yaitu pada jaringan VANET. Dalam membuat *script* simulasi jaringan digunakan bahasa pemerograman TCL (*Tool Command Language*). Bahasa pemerograman TCL merupakan bahasa pemerograman yang berdasarkan pada *string.*

3.1.6 *Filtering File Trace*

Pada tahap ini didapatkan hasil simulasi jaringan dengan menggunakan NS-2 dan menghasilkan *file trace* (.tr). Setelah itu dilakukan *filtering* terhadap *file trace* dengan menggunakan bahasa pemrograman

AWK. Kemudian hasil *filtering* dari *file trace* berupa parameter uji. Parameter uji yang digunakan yaitu :

a. *Throughput* : suatu istilah yang mendefinisikan banyaknya bit yang diterima dalam selang waktu tertentu. Secara umum *throughput* dinyatakan dalam persamaan 1 dengan rumus sebagai berikut [7]:

$$\textit{Throughput}\ (\text{Kbps}) = \frac{\sum paket\ data\ diterima}{\sum waktu} \quad (1)$$

b. *Average end-to-end Delay* : waktu yang dibutuhkan untuk melakuka pengiriman paket data dari *node* sumber ke *node* tujuan [8]. *Average end-to-end Delay* dihitung menggunakan persamaan 2 dengan rumus sebagai berikut :

$$\textit{Average Delay}\ (\text{m/s}) = \frac{\text{Total } Delay}{\text{Total paket yang diterima}} \quad (2)$$

c. *Packet Delivery Ratio* (PDR) : perbandingan antara paket data yang berhasil diterima oleh *node* tujuan dengan total paket data yang dikirimkan oleh *node* sumber[8]. PDR dihitung menggunakan persamaan 3 dengan rumus sebagai berikut :

$$\text{PDR}\ (\%) = \frac{\text{Paket data diterima}}{\text{Paket data dikirim}} \quad (3)$$

3.1.7 Membuat Grafik Hasil Simulasi

Pada tahap ini nilai yang didapatkan setelah dilakukan *filtering file trace* kemudian dibuat dalam bentuk grafik agar memudahkan dalam melakukan analisis. Dalam membuat grafik hasil uji coba simulasi dengan mengguanakan *Microsoft Excel,* dimana datanya diperoleh dari proses *filtering* dari *file trace* yang berisikan parameter uji.

3.1.8 Menganalisis Hasil Simulasi

Pada tahap ini dilakukan proses analisis terhadap hasil yang diperoleh dari penelitian yang dilakukan. Hasil yang diperoleh untuk mengetahui mengetahui bagaimana pengaruh algoritma semut pada protokol AOMDV dalam jaringan VANET. *Script* AWK digunakan untuk menganalisis parameter uji kinerja yang menjadi acuan yaitu *throughput, Average End To End Delay* dan *Packet Delivery Ratio.* Jika hasil yang diperoleh tidak sesuai memenuhi kualitas kinerja yang baik yang ditentukan dari parameter uji maka akan dilakukan perancangan simulasi kembali.

3.1.9 Membuat Kesimpulan

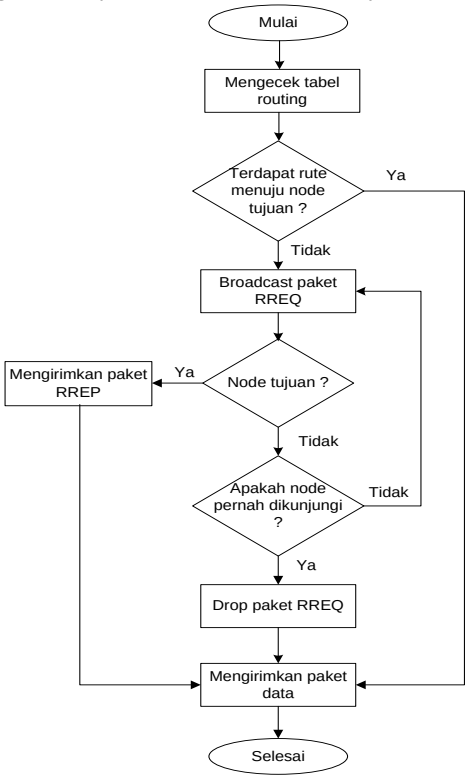
Pada tahap ini akan dilakukan penarikan kesimpulan terhadap penelitian yang telah dilakukan. Penarikan kesimpulan dilakukan agar dapat mengetahui kelebihan dan kekurangan dari suatu penelitian, sehingga kinerja protokol *routing* AOMDV dengan algoritma semut di jaringan VANET dapat diketahui.

3.1.10 Membuat Laporan

Pada tahap ini membuat laporan atau dokumentasi terhadap penelitian yang dilakukan. Dokumentasi laporan ini diharapkan agar dapat membantu dalam penelitian berikutnya yang berkaitan dengan penelitian yang dilakukan.

3.2 Protokol *Routing* AOMDV

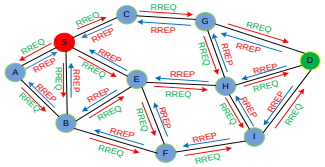
Protokol *routing* AOMDV (*Ad Hoc On-Demand Multipath Distance Vector Routing*) adalah protokol *routing* yang bersifat reaktif yang merupakan pengembangan dari protokol *routing* AODV. AOMDV juga menyediakan dua mekanisme kerja yaitu *route discovery* dan *maintenance*. Pada protokol *routing* AOMDV berbasis vektor dan menggunakan pendekatan *hop-by-hop*. AOMDV juga hanya melakukan pencarian rute ketika dibutuhkan dengan menggunakan prosedur *route discovery*.



Gambar 3. Diagram alir *route discovery* AOMDV

Proses pencarian rute pada protokol *routing* AOMDV ketika *node* sumber memerlukan rute untuk mengirimkan paket ke *node* tujuan. *Node* sumber akan memeriksa pada tabel *routing* terlebih dahulu untuk

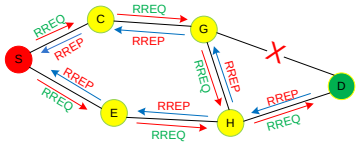
mengirimkan paket data menuju *node* tujuan. Jika rute tersedia maka akan dipilih rute terdekat berdasarkan jumlah *hop* terkecil untuk mengirimkan paket data. Jika tidak ada maka akan dilakukan pencarian rute dengan cara mem-*broadcast* paket RREQ *node* terdekatnya dan melakukan *set up reverse path*. Setiap *node* yang dikunjungi akan diperiksa terlebih dahulu sebelum melanjutkan *broadcast* RREQ. Jika *node* yang dikunjungi merupakan *node* tujuan, maka akan dilakukan pengiriman RREP melalui *reverse path* yang sudah di *set up* sebelumnya. Jika bukan *node* tujuan maka akan dicek kembali apakah *node* tersebut sudah pernah dikunjungi sebelumnya. Jika pernah dikunjungi, maka paket RREQ akan di *drop.* Jika belum pernah dikunjungi, maka proses *broadcast* RREQ akan dilanjutkan sampai *node* tujuan ditemukan.



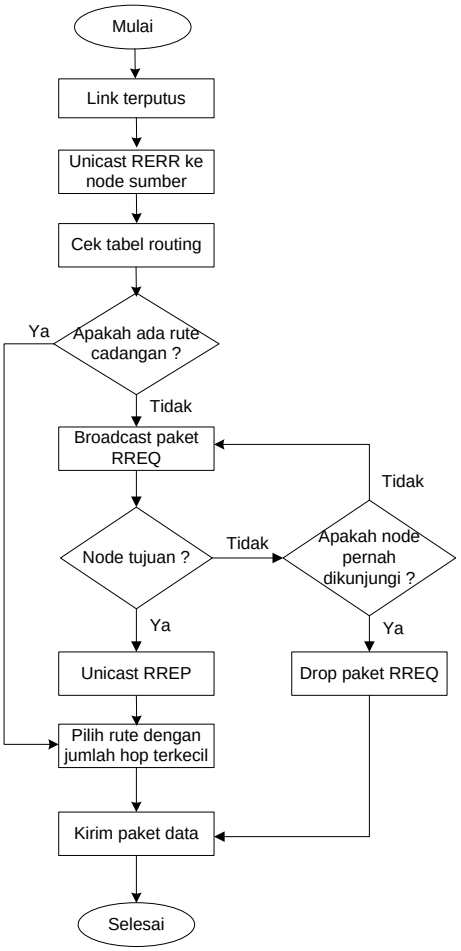
Gambar 4. Ilustrasi *route discovery* AOMDV

Proses pencarian rute pada protokol *routing* AOMDV. *Node* S akan mengirimkan paket data ke *node* D, karena pada tabel *routing node* S tidak terdapat rute cadangan ke *node* D, maka *node* harus mem-*broadcast* paket RREQ ke *node-node* disekitarnya untuk menemukan rute menuju *node* D. *Node* S mem-*broadcast* paket RREQ ke *node* A, B, C dan E, kemudian *node* A, B, C, E juga akan mem-*broadcast* paket RREQ ke *node-node* tetangga mereka masing-masing hingga sampai di *node* D. *Node-node* yang berada diantara *node* S dan *node* D merupakan *intermediate node* yang akan melakukan *set up reverse path* untuk mengingat *node-node* yang telah dilewati untuk mencapai *node* D. Selanjutnya, *node* D akan mengirimkan paket RREP sebagai balasan dari paket RREQ ke *node* S secara *unicast* melalui masing-masing rute. Dengan demikian, diperoleh empat rute, yakni S → A → B → F → I → D, S→B→F→I→D, S→C→G→D dan S→E→H→D. Rute yang terpilih adalah rute yang memiliki jumlah *hop* terkecil, yakni rute S→C→G→D dan S→E→H→D. Sedangkan, rute yang tidak terpilih akan dijadikan rute cadangan.

Gambar 5. ilustrasi *route maintenance* AOMDV



Dimana pada saat terjadi kegagalan rute, maka akan dikirim paket *Route Error* (RERR), jika terdapat jalur yang rusak, protokol AOMDV akan memilih rute cadangan (*multipath*) apabila ada, kemudian *node* sumber akan kembali melakukan pencarian rute.



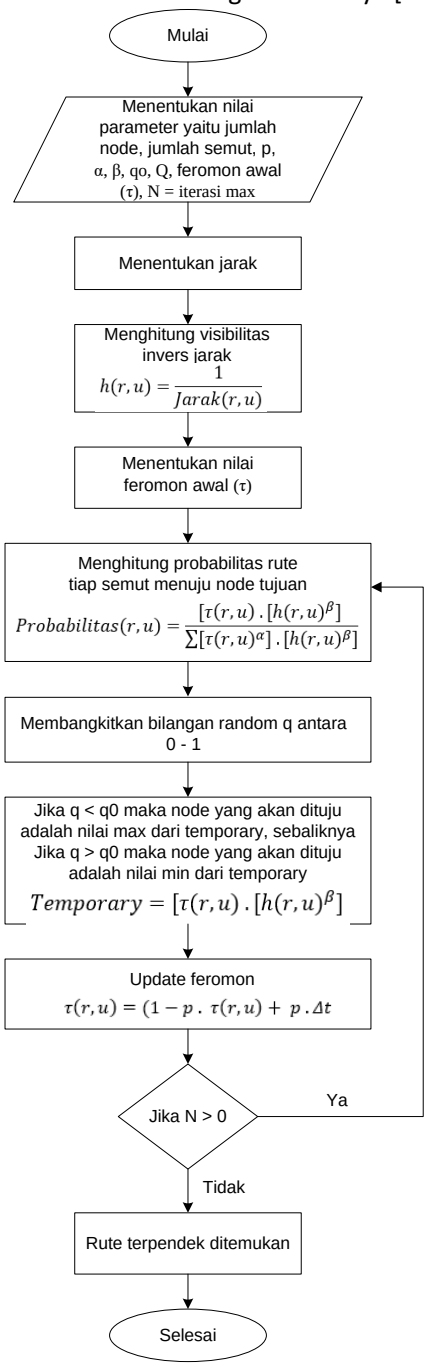
Gambar 6. Diagram alir *route maintenance* AOMDV

Proses route maintenance dari ptotokol *routing* AOMDV. *Route maintenance* merupakan proses pemeliharaan rute. *Node* akan mengirimkan RERR (*Route Error*) ke *node* sumber secara *unicast* jika link yang menghubungkan antara *node* sumber dan *node* tetangganya terputus. Kemudian tabel *routing node* sumber akan di cek, apakah terdapat rute cadangan ke *node* tujuan yang sama. Jika ada, maka rute dengan jumlah *hop* terkecil akan dipilih dan paket data dapat langsung dikirimkan. Namun, apabila tidak terdapat rute cadangan, maka akan dilakukan proses *route discovery.*

3.3 Algoritma Semut

Algoritma semut merupakan algoritma yang terinspirasi oleh perilaku semut dalam menemukan rute dari sarang menuju sumber makanan. Semut mampu mengindra lingkungannya yang kompleks mencari makanan dan kemudian kembali ke sarangnya dengan meninggalkan zat feromon pada jalur-jalur

yang mereka lalui. Dengan adanya feromon inilah semut dapat menemukan jalur tercepat menuju sumber makanan. Feromon merupakan zat kimia yang berasal dari kelenjar endokrin dan digunakan oleh makhluk hidup untuk mengenali sejenisnya (satu spesies). Pada semut meninggalkan jejak kaki untuk sebagai penanda untuk mengingat jalan pulang maupun berkomunikasi dengan koloninya [10].



Gambar 7. Diagram alir proses algoritma semut

Pada Gambar 7 merupakan proses kerja dari algoritma semut. Adapun tahap-tahap yang dilakukan yaitu:

1. Tahap pertama yaitu menentukan atau menginisialisasikan nilai parameter awal, parameter-parameter tersebut adalah [7] [9]:
   a. Banyak *node* atau jumlah *node*
   b. Banyak semut
   c. Tetapan siklus-semut (Q)
   d. Tetapan pengendali intensitas jejak semut (α), nilai α ≥ 0
   e. Tetapan pengendali visibilitas (β), nilai β ≥ 0
   f. Tetapan penguapan jejak semut (ρ), nilai ρ harus > 0 dan < 1 untuk mencegah jejak feromone yang tak terhingga.
   g. Parameter perbandingan eksploitasi terhadap eksplorasi (q0)
   h. Nilai feromon awal (τ)
2. Kemudian memperhitungkan koordinat (x,y) atau jarak antar *node* untuk mengetahui jarak *node* satu dengan *node* lainnya.
3. Setelah semut di inisialisaikan kemudian semut ditempatkan pada *node* pertama tertentu secara acak. Semut ini disebut paket *forward ant*, karena akan menelusuri rute hingga ke *node* tujuan. Ketika sudah menemukan *node* tujuan, semut ini akan kembali melewati rute yang sama. Pada kondisi ini semut tersebut disebut *backward ant*.
4. Selanjutnya yaitu menghitung *invers* jarak atau *visibilitas* antar *node* yang akan dilakukan pengisian kedalam tabu *list* yang berisikan informasi dari *visibilitas* antar *node*. Dapat dilihat pada persamaan 4 yaitu sebagai berikut [9]:

$$h(r,u) = \frac{1}{Jarak(r,u)} \quad (4)$$

5. Menentukan nilai feromon awal dan memasukkannya kedalam tabu list. Nilai dari semua feromon (τ) pada awal perhitungan ditetapkan dengan angka awal yang sangat kecil [8], misal 0.001.
6. Selanjutnya menghitung probabilitas rute tiap semut menuju *node* tujuan. Koloni semut yang sudah terdistribusi ke sejumlah atau setiap *node*, akan mulai melakukan perjalanan dari *node* pertama masing-masing sebagai *node* asal dan salah satu *node* lainnya sebagai *node* tujuan. Untuk menentukan *node* tujuan dapat dilihat pada persamaan 5 yaitu sebagai berikut [9]:

$$Prob(r,u) = \frac{[\tau(r,u).[h(r,u)^{\beta}]}{\sum[\tau(r,u)^{\alpha}].[h(r,u)^{\beta}]} \quad (5)$$

7. Membangkitkan bilangan *random* q yaitu koloni semut akan melanjutkan perjalanan dengan memilih salah satu dari *node* - *node* yang tidak terdapat pada tabu *list* sebagai *node* tujuan selanjutnya. Perjalanan koloni semut berlangsung

terus menerus sampai semua *node* satu persatu dikunjungi atau telah menempati tabu *list*.

8. Membandingkan nilai dari bilangan *random* q dengan bilangan q0 yang telah ditetapkan diawal. Jika q>q0 maka *node* selanjutnya yang dipilih adalah nilai *temporary* yang paling kecil, jika q<q0 maka *node* selanjutnya yang dipilih adalah nilai *temporary* yang paling besar. Dapat dilihat pada persamaan 6 yaitu sebagai berikut [9]:

$$Temporary = [\tau(r,u) \,.\, [h(r,u)^{\beta}] \quad (6)$$

9. Setelah rute dari semut 1 hingga ke-n dari siklus pertama ditemukan, maka dilakukan pembaharuan feromon atau *update* feromon. Koloni semut akan meninggalkan jejak-jejak kaki pada lintasan antar *node* yang dilaluinya. Adanya penguapan dan perbedaan jumlah semut yang lewat, menyebabkan kemungkinan terjadinya perubahan harga intensitas jejak kaki semut antar *node*. Dapat dilihat pada persamaan 7 yaitu sebagai berikut [9]:

$$\tau(r,u) = (1 - p \,.\, \tau(r,u) + p \,.\, \Delta t \quad (7)$$

Δt adalah perubahan harga intensitas jejak kaki semut antar *node* setiap semut yang dihitung berdasarkan persamaan 8 [9] :

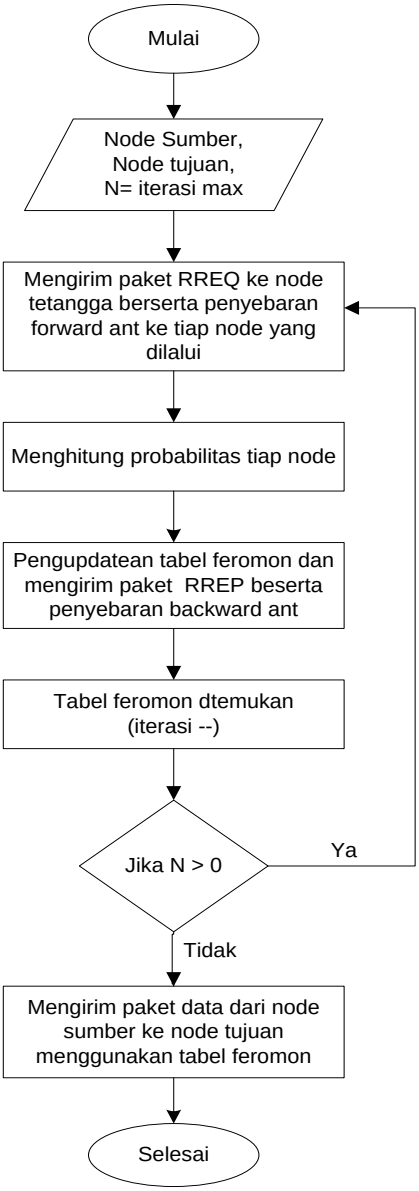
$$\Delta t = \frac{Q}{Lk} \quad (8)$$

Lk merupakan panjang jarak dari rute yang telah ditemukan.

10. Lakukan ulang langkah 6 hingga iterasi atau perulangan selesai, sehingga rute terpendek dapat ditemukan dengan menilai dari banyak feromon yang dilalui semut pada rute tersebut.

3.4 Penerapan Algoritma Semut pada Protokol *Routing* AOMDV

Pada protokol *routing* AOMDV mempunyai tujuan yang sama dengan algoritma semut yaitu sama-sama mencari rute yang paling optimal dari *node* sumber ke *node* tujuan. Pada algoritma semut memiliki dua mekanisme kerja yaitu *forward ant* dan *backward ant. Forward ant* sama halnya dengan paket RREQ pada protokol *routing* AOMDV dan *backward ant* sama halnya dengan paket RREP pada protokol *routing* AOMDV. Penerapan pada *routing* jaringan, sarang semut merupakan *node* sumber dan sumber makanan adalah *node* tujuan.



Gambar 8. Diagram alir proses AOMDV-Semut

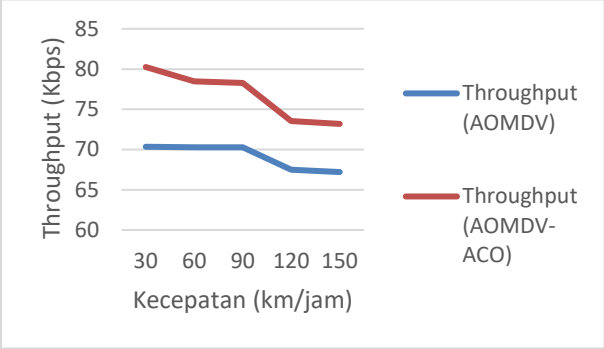
4. HASIL DAN PEMBAHASAN

Berdasarkan ujicoba yang dilakukan pada penelitain ini untuk menilai kinerja dari protocol routing AOMDV yang digunakan dibutukannya suatu parameter ujicoba Berikut ini merupakan analisis dari parameter ujicoba yang digunakan pada penelitan :

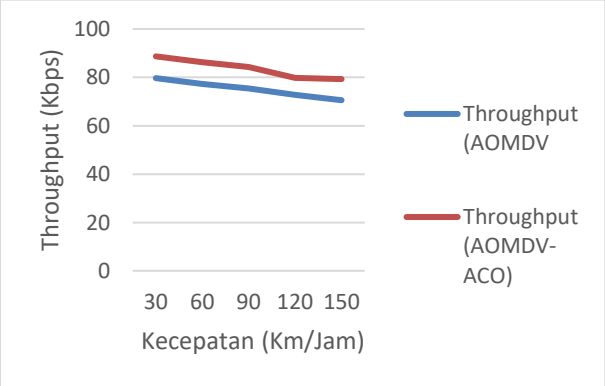
4.1 Analisis *Throughput*

Throughput merupakan ukuran kecepatan yang diterima dari *node* sumber ke *node* tujuan. *Throughput* dapat dikatakan baik dilihat dari kinerja pengiriman data yang tinggi dari *node* sumber ke *node* tujuan. Berdasarkan seluruh ujicoba yang telah dilakukan pada percobaan AOMDV standar, dapat diketahui bahwa

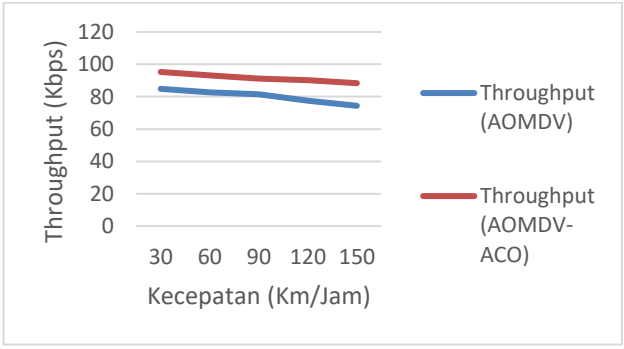
pada parameter *throughput* nilai yang di peroleh relative tinggi, Hal ini dikarenakan sifat reaktif yang dimiliki oleh protokol AOMDV, sehingga pencarian rute hanya dilakukan ketika rute dari *node* sumber ke *node* tujuan tidak terdapat di dalam tabel *routing*. Adanya penambahan jumlah menyebabkan nilai *throughput* pada protokol AOMDV mengalami peningkatan, hal ini disebabkan semakin banyak *node* yang terdapat pada lingkungan simulasi maka lingkungan simulasi akan semakin padat sehingga semakin kecil kemungkinan terjadinya link terputus pada jalur komunikasi dan menyebabkan daya tahan link menjadi lebih lama. Kondisi ini bebanding terbalik dengan adanya penambahan kecepatan *node*, semakin tinggi mobilitas *node* maka semakin rendah nilai *throughput* yang dihasilkan, karena semakin tinggi mobilitas maka setiap *node* maka pergerakan *node* akan semakin cepat, sehingga link yang sebelumnya telah terbentuk akan semakin cepat putus, sehingga untuk dapat mengirimkan paket dari sumber ke tujuan diperlukan proses broadcast kembali.



Gambar 9. Grafik *Throughput* 50 *Node*



Gambar 10. Grafik *Throughput* 70 *Node*

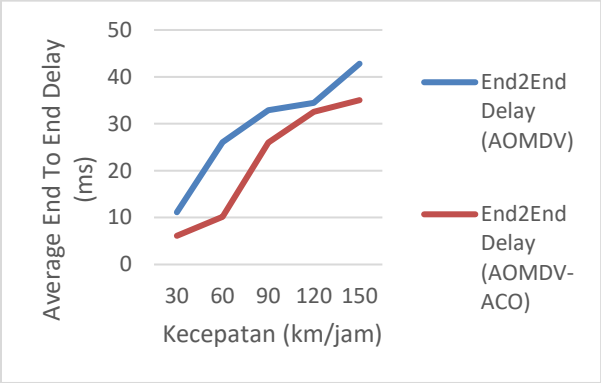


Gambar 11. Grafik *Throughput* 100 *Node*

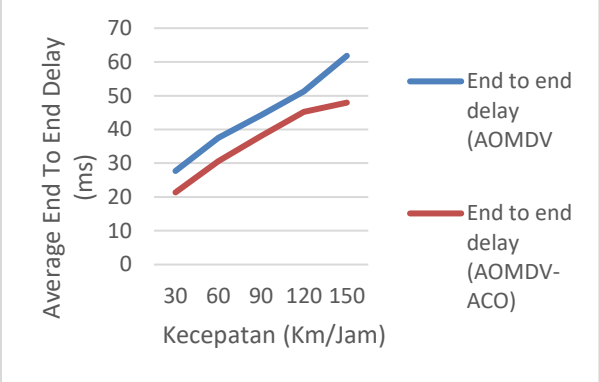
Perbandingan nilai *throughput* pada *node* 50, 70 dan 100 yaitu berdasarkan grafik Gambar 9, Gambar 10 dan Gambar 11. Pada percobaan tersebut menunjukan hasil bahwa protocol routing AOMDV yang telah dimodifikasi dengan algoritma semut lebih baik dibandingkan AOMDV standar. Dapat dilihat dari nilai nilai rata-rata yang diperoleh *throughput* pada protokol *routing* AOMDV tanpa modifikasi dengan 50 *node* yaitu sebesar 69.11256Kbps. Untuk nilai rata-rata *throughput* pada protokol *routing* AOMDV-semut dengan 50 *node* yaitu sebesar 76.7503Kbps. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut pada parameter *throughput* pada 50 *node* yaitu sebesar 7.63774Kbps. Sedangkan nilai rata-rata *throughput* protokol *routing* AOMDV standar pada 70 *node* yaitu sebesar 75.1384Kbps. Untuk nilai rata-rata *throughput* protokol *routing* AOMDV-semut pada 70 *node* yaitu sebesar 83.6689Kbps. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut pada 70 *node* yaitu sebesar 10.2366Kbps. Kemudian nilai rata-rata *throughput* protokol AOMDV standar pada 100 *node* yaitu sebesar 80.20694Kbps. Dan nilai rata-rata *throughput* protokol *routing* AOMDV-semut pada 100 *node* yaitu sebesar 91.62742Kbps. Adapun selisih rata-rata yang diperoleh pada parameter *throughput* antara AOMDV standar dengan AOMDV-Semut pada 100 *node* yaitu sebesar 11.42048Kbps. Sehingga jika dirata-ratakan secara keseluruhan AOMDV semut mampu meningkatkan nilai *throughput* sebesar 9.7649 Kbps.

4.2 Analisis *Average End-to-End Delay*

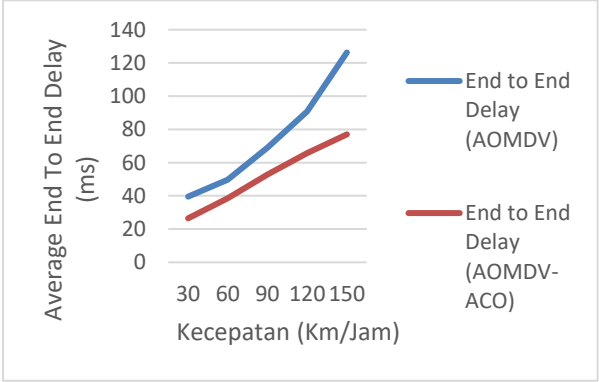
Average End To End Delay merupakan waktu yang diperlukan untuk mengirimkan paket dari *node* sumber menuju *node* tujuan. Semakin sedikit waktu yang dibutuhkan maka semakin bagus kinerja dari protocol *routing* tersebut.



Gambar 12. Grafik *Delay* 50 *Node*



Gambar 13. Grafik *Delay* 70 *Node*



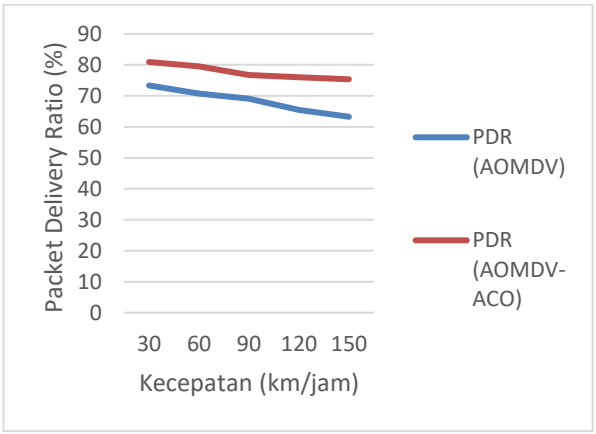
Gambar 14. Grafik *Delay* 100 *Node*

Perbandingan nilai *average end to end delay* pada *node* 50, 70 dan 100 yaitu berdasarkan grafik Gambar 12, Gambar 13 dan Gambar 14. Dapat diperoleh nilai rata-rata *end to end delay* pada AOMDV standar dengan 50 *node* yaitu sebesar 29.48072 ms. Sedangkan nilai rata-rata *end to end delay* pada AOMDV semut dengan 50 *node* yaitu sebesar 21.980328ms. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut dengan 50 *node* pada parameter *end to end delay* sebesar 7.500392ms. Dan nilai rata-rata *end to end delay* pada AOMDV standar dengan 70 *node* yaitu sebesar 44.52014 ms sedangkan nilai rata-rata *end to end delay* pada AOMDV semut dengan 70 *node* yaitu sebesar 36.65096ms. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut dengan 70 *node* pada parameter *end to end delay* sebesar 9.443016ms. Dan nilai rata-rata *end to end delay* pada AOMDV standar dengan 100 *node* yaitu sebesar 75.06622 ms. Sedangkan nilai rata-rata *end to end delay* pada AOMDV semut dengan 100 *node* yaitu sebesar 52.0818ms. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut dengan 100 *node* pada parameter *end to end delay* sebesar 22.98442ms. Hal ini menunjukkan bahwa implementasi algoritma semut pada protokol AOMDV menyebabkan penurunan nilai *end to end delay* pada protocol *routing* AOMDV standar. Sehingga dari hasil ujicoba yang telah dilakukan pada protokol *routing* AOMDV dengan menggunakan algoritma semut untuk parameter *average end to end delay* mampu menurunkan *delay* sebesar 13.3093 ms.

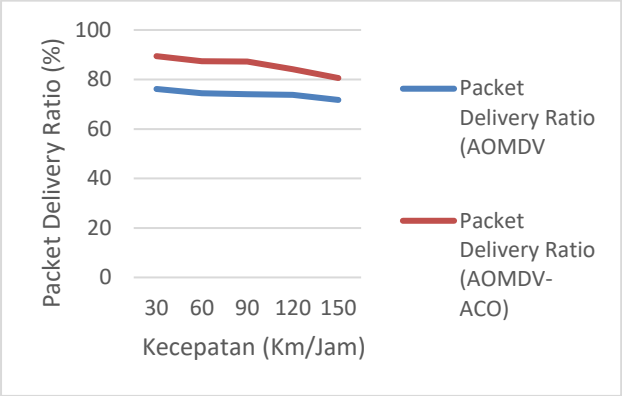
Pada parameter *End to End Delay* yaitu waktu yang dibutuhkan dalam pengiriman data dari *node* sumber ke *node* tujuan. Dalam proses pengiriman data, ada banyak faktor yang mengakibatkan terjadinya *Delay*. Peningkatan *Delay* dapat terjadi ketika jumlah komunikasi bertambah banyak (kepadatan kendaraan bertambah). Semakin padatnya kendaraan (*node*) dapat menyebabakn semakin banyak *node* perantara yang terpakai untuk mengirimkan layanan dari sumber ke tujuan, sehingga paket-paket yang dikirimkan harus melalui buffer dan menyebabkan waktu pengiriman semakin lama.

4.3 Analisis *Packet Delivery Ratio*

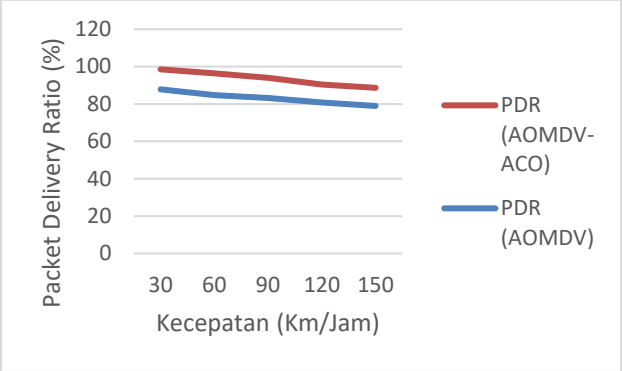
Packet Delivery Ratio merupakan perbandingan antara paket yang berhasil di kirim oleh *node* sumber dan di terima oleh *node* tujuan. Berikut grafik kualitas *Packet Delivery Ratio* dari percobaan yang telah dilakukan :



Gambar 15. Grafik PDR 50 *Node*



Gambar 16. Grafik PDR 70 *Node*



Gambar 17. Grafik PDR 100 *Node*

Perbandingan nilai *Packet Delivery Ratio* pada *node* 50, 70 dan 100 yaitu berdasarkan grafik Gambar 15, Gambar 16 dan Gambar 17. Dapat di peroleh jumlah rata-rata *Packet delivery ratio* pada AOMDV standar dengan 50 *node* yaitu sebesar 68.3622%. Sedangkan jumlah rata-rata *Packet delivery ratio* pada AOMDV semut dengan 50 *node* yaitu sebesar 77.6981%. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut pada parameter *Packet delivery ratio* dengan 50 *node* yaitu sebesar 9.3359%. Dan jumlah rata-rata *Packet delivery ratio* pada AOMDV standar dengan 70 *node* yaitu sebesar 74.03446%. Sedangkan jumlah rata-rata *Packet delivery ratio* AOMDV semut dengan 70 *node* yaitu sebesar 85.7396%. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut dengan 70 *node* pada parameter *Packet delivery ratio* sebesar 14.046168%. Dan jumlah rata-rata *Packet delivery ratio* pada AOMDV standar dengan 100 *node* yaitu sebesar 83.1059%. jumlah rata-rata *Packet delivery ratio* pada AOMDV semut dengan 100 *node* yaitu sebesar 93.57522%. Adapun selisih rata-rata yang diperoleh antara AOMDV standar dengan AOMDV-Semut pada parameter *Packet delivery ratio* dengan 100 *node* yaitu sebesar 10.46932%. Hal ini menunjukkan bahwa implementasi algoritma semut pada protokol AOMDV menyebabkan peningkatan persentase *Packet delivery ratio*. Dari hasil ujicoba yang telah dilakukan pada protokol *routing* AOMDV dengan menggunakan algoritma semut mampu meningkatkan kinerja dari parameter ujicoba *Packet delivery ratio* sebesar 11.2838%.

Pada parameter *Packet Delivery Ratio* (PDR) menunjukkan keberhasilan protokol dalam mengirim data, tinggi nilai *Packet Delivery Ratio* salah satunya disebabkan oleh berhasilnya sebuah protokol dalam melakukan pencarian dan pemeliharaan rutenya. Semakin padat jumlah *node* dapat diperoleh hasil yang semakin tinggi. Semakin tingginya kecepatan pergerakan *node* mengakibatkan semakin rendahnya *Packet Delivery Ratio* pada protokol AOMDV. Hal ini membuktikan bahwa semakin padatnya lingkungan simulasi yang diikuti dengan semakin lambatnya kendaraan bergerak menyebabkan rute yang terbentuk tidak cepat terputus, sehingga protokol tidak sering melakukan pembaharuan rute dan semakin sedikit terjadinya *link failure* maka semakin besar nilai *Packet Delivery Ratio* pada suatu protokol.

Berdasarkan seluruh uji coba yang dilakukan, didapatkan hasil yang menunjukkan bahwa implementasi algoritma semut pada protokol AOMDV mampu menurunkan waktu *End to End Delay* pada proses pengiriman data. Dan mampu meningkatkan nilai *throughput* dan persentase *Packet Delivery Ratio*. Sehingga dapat disimpulkan bahwa algoritma Semut mampu meningkatkan kinerja protokol AOMDV dengan sangat baik.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasi percobaan dan analisa yang telah dilakukan, maka dapat ditarik kesimpulan sebagai berikut :

1. Pada uji coba AOMDV dengan jumlah *node* yang semakin banyak (50 *node*, 70 *node* 100 *node*) didapatkan hasil yang menunjukan bahwa semakin banyak jumlah *node*, maka nilai parameter *throughput*, *Packet delivery ratio* yang diperoleh semakin meningkat hal ini di sebabkan karena semakin padatnya lingkungan simulasi sehingga link tidak cepat terputus.
2. Pada *avegare end to end delay* semakin besar jumlah *node* maka semakin besar nilai yang di peroleh karena banyaknya *node* yang harus di lewati untuk mengirimkan paket data.

3. Pada penelitian dilakukan 5 kali percobaan. Hasil menunjukkan bahwa uji coba yang telah dilakukan pada protocol *routing* AOMDV dengan menggunakan algoritma semut dapat meninggkatkan kinerja dari parameter *throughput* sebesar sebesar 9.7649 Kbps.
4. Untuk parameter dari *Packet delivery ratio* dengan menerapkan algoritma semut pada protocol *routing* AOMDV menunjukan hasil peningkatan dengan nilai sebesar 11.2838%.
5. Sedangkan untuk parameter *average end to end delay* mampu menurunkan *delay* pada protocol *routing* AOMDV dengan nilai sebesar 13.3093 ms.
6. Penentuan parameter, kecepatan *node*, serta jumlah *node* serta jumlah semut pada algoritma semut yang digunakan, dapat mempengaruhi hasil optimasi pencarian rute pada protokol *routing* AOMDV.

5.2 Saran

Berdasarkan kesimpulan di atas, maka peneliti dapat memberikan saran-saran sebagai berikut:

1. Mengimplementasikan algoritma optimasi lainnya pada protokol *routing* AOMDV
2. Melakukan penelitian dengan menggunakan protokol *routing* yang berbeda pada algoritma semut.
3. Melakukan penelitian yang sama dengan mengubah parameter-parameter penelitian.
4. Berdasarkan kendala yang dialami penulis, penelitian dapat dilakukan dengan mengembangkan sendiri algoritma optimasi, sehingga dapat digunakan sesuai dengan kebutuhan dan dapat dikostumasi.
5. Untuk penelitian selanjutnya dapat digunakan whitebox testing untuk menjabarkan efisiensi rute yang di hasilkan.

DAFTAR PUSTAKA

[1] F. Nutrihadi, R. Anggoro and R. M. Ijtihadie, "Studi Kinerja VANET Scenario Generators: SUMO dan Vanet Mobisim untuk Implementasi *Routing* Protocol AODV menggunakan Network Simulator 2 (NS-2)," *Jurnal Teknik ITS ,* vol. 5, no. 1, pp. A19-A24, 2016.

[2] R. Anisia, R. Munadi and R. M. Negara, "AnalisisPerformansi *Routing* Protocol OLSR dan AOMDV pada Vehicular Ad Hoc Network (VANET)," *Jurnal Nasional Teknik Elektro ,* vol. 5, no. 1, pp. 87-97, 2016.

[3] A. Deshmukh and S. Dorle, "Performance Improvement of Dynamic Source *Routing* (DSR) Protocol using Ant Colony Optimization for Vehicular Ad-hoc Network (VANet)," *International Journal of Scientific Research,* vol. 5, no. 1, pp. 171-173, 2016.

[4] S. Singh, P. Kumari and S. Agrawal, "Comparative Analysis of Various Routing Protocols in VANET," International Conference on Advanced Computing & Communication Technologies, pp. 315-319, 2015.

[5] C. Kanani and A. Sinhal, "Ant Colony Optimization based Modified AOMDV for Multipath Routing in MANET," International Journal of Computer Applications, vol. 82, no. 10, pp. 14-19, 2013.

[6] R. Chauhan and A. Dahiya, "AODV Extension using Ant Colony Optimization for Scalable Routing in VANETs," Journal of Emerging Trends in Computing and Information Sciences, vol. 3, no. 2, pp. 241-244, 2012.

[7] V. L. Ratrindra, Analisis Performansi Destination Sequenced Distance Vector (DSDV) dan Zone Routing Protocol (ZRP) Berbasis Algoritma Ant pada Jaringan Mobile Ad Hoc, Bandung: Institut Teknologi Telkom, 2010.

[8] H. E. Wahanani, "Kinerja Protokol DSR Pada Jaringan MANET dengan Metode Node Disjoint And Alternative Multipath Routing," in Seminar Nasional Teknik Informatika (SANTIKA) , Universitas Pembangunan Nasional “Veteran” Jawa Timur , 2013.

[9] R. Amalia, "Pencarian Jalur Terpendek Menggunakan Ant Colony System (Kasus: Pariwisata Kota Bogor)," Universitas Indraprasta PGRI, pp. 290-304, 2015.

[10] Karjono, Moedjiono and D. Kurniawan, "Ant Colony Optimization," *Jurnal TICOM,* vol. 4, no. 3, pp. 119-125, 2016.