Simulasi dan Pengkajian Performa *Vehicular Ad Hoc Network*

Aletheia Anggelia Tonoro[1], Hartanto Kusuma Wardana[2], Saptadi Nugroho[3]

Program Studi Sistem Komputer
Fakultas Teknik Elektronika dan Komputer
Universitas Kristen Satya Wacana, Salatiga
[1]aletheiaanggelia@gmail.com, [2]Hartanto.Kusuma@staff.uksw.edu,
[3]saptadi_nugroho@yahoo.com

Ringkasan

Meningkatnya, tingkat kecelakaan dan kemacetan di jalan raya, dan berkembangnya teknologi informasi dengan menggunakan *wireless*, menghadirkan teknologi *Wireless Access for Vehicular Environment* (WAVE) sebagai standart komunikasi kendaraan. Salah satu, perkembangan WAVE adalah *Vehicular ad hoc networks* (VANET). Teknologi VANET memungkinkan sebuah perangkat komunikasi dapat berkomunikasi secara langsung dengan perangkat lain dalam posisi bergerak misalnya mobil. Meskipun VANET dapat membantu menyelesaikan permasalahan lalu lintas seperti kecelakaan, dan kemacetan, tapi untuk membangun infrastruktur jaringan VANET tidaklah mudah dan membutuhkan biaya yang cukup besar. Untuk itu, muncullah *network simulator* seperti VEINS, SUMO dan OMNET++ yang dapat membantu mensimulasikan jaringan VANET tanpa harus mengeluarkan biaya untuk membangun infrastrukturnya. Karena itu, pada akan dilakukan simulasi jaringan VANET menggunakan VEINS, SUMO dan OMNET++. Dalam pengujian unjuk kerja VANET digunakan 3 jenis *routing* yaitu *Optimized Link State* (OLSR), *Ad hoc on Demand Distance Vector Routing* (AODV) dan *Dynamic* Manet *on Demand* (DYMO) dengan *delay*, *throughput* dan *packet delivery ratio* sebagai parameter pengujian.

Kata kunci : WAVE, VANET, VEINS, SUMO, OMNET++, OLSR, AODV dan DYMO

1.`Latar Belakang

Meningkatnya jumlah kendaraan per tahun menimbulkan banyak permasalahan bukan saja polusi tapi juga kemacetan hingga kecelakaan lalu lintas yang terus bertambah setiap tahunnya. Sehingga, diperlukan suatu sistem yang dapat membantu mengurangi kemacetan dan kecelakaan seperti *Intelligent Transportation System* (ITS). ITS adalah transportasi cerdas yang menggabungkan antara sistem transportasi dengan teknologi informasi demi meningkatkan aksesibilitas dan efisiensi serta keamanan transportasi.

Hadirnya ITS diharapkan mampu memberikan informasi secara *real time* kepada pengguna jalan berkaitan dengan situasi jalan yang akan dilalui. Misalnya, jalan yang akan digunakan macet, maka akan disediakan alternatif jalan lain yang tidak mengalami kemacetan. Selain memberikan alternatif jalan dan menghindari macet, ITS pun memberikan informasi mengenai kondisi kendaraan yang ada di sekitar, sehingga dapat membantu dan menghindarkan pengguna dari kecelakaan. Salah satu, kecanggihan dari

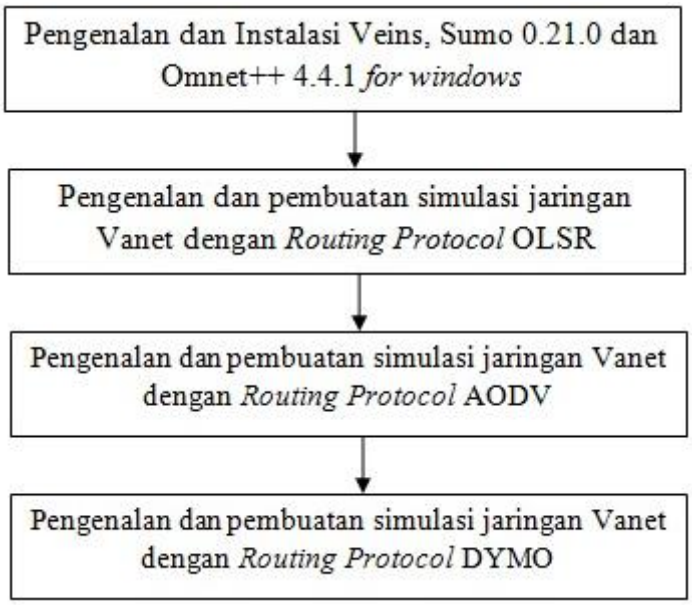
ITS dan sampai saat ini masih terus dikembangkan adalah *Vehicular Ad Hoc Network* (VANET).

Walaupun VANET sangat membantu untuk mengurangi kecelakaan dan kemacetan tapi pembangunan infrastruktur untuk sistem VANET tidaklah mudah, karena dibutuhkan biaya yang besar dalam pengembangan dan pengujiannya sehingga sampai saat ini belum ada negara yang benar-benar menerapkan VANET. Agar penelitian dibidang VANET tidak terhenti, maka dilakukan pemodelan jaringan VANET dalam bentuk simulasi. Keuntungan dari melakukan simulasi terlebih dahulu adalah kebebasan untuk memodelkan dan mengevaluasi rancangan tanpa harus membangun jaringan fisiknya.

Selain akan melakukan simulasi VANET, akan dilakukan juga analisis *performance* protokol *routing Optimized Link State* (OLSR), *Ad hoc on Demand Distance Vector Routing* (AODV) dan *Dynamic* Manet *on Demand* (DYMO), sehingga mahasiswa dapat melakukan analisa dari hasil yang didapatkan. Untuk melakukan simulasi dibutuhkan sebuah *network simulator* yang dapat mensimulasikan kerja VANET. Salah satu *network simulator* yang dapat mensimulasikan VANET dengan baik yaitu *Vehicle in Network Simulation* (VEINS).

2. Perancangan Sistem

Simulasi jaringan VANET, dibagi menjadi 4 langkah seperti yang terlihat di Gambar 1.



Gambar 1. Diagram Alir Simulasi jaringan VANET

2.1. Simulasi Jaringan VANET

Simulasi jaringan VANET menggunakan *Vehicle In Network Simulation* (VEINS) yang dibangun oleh 2 simulator yaitu OMNET++ dan SUMO [1]. OMNET++ adalah sebuah *event-based network simulator*. OMNET++ didesain untuk komunikasi wireless yang menggunakan pendekatan arsitektur berbasis komponen dan menggunakan bahasa pemrograman C++ serta Tcl/Tk untuk implementasi model dan simulasi[2], sedangkan

Simulation Of Mobility (SUMO) adalah *road traffic simulator*[3]. Berikut adalah pengaturan dari simulasi VANET pada omnet++:

Tabel 1. Pengaturan Simulasi Jaringan VANET.

Atribut	Nilai
Waktu Simulasi	100 s
Ukuran Jaringan	1000m x 1000m
Bit rate	54Mbps
Protokol *Routing*	OLSR, AODV, DYMO

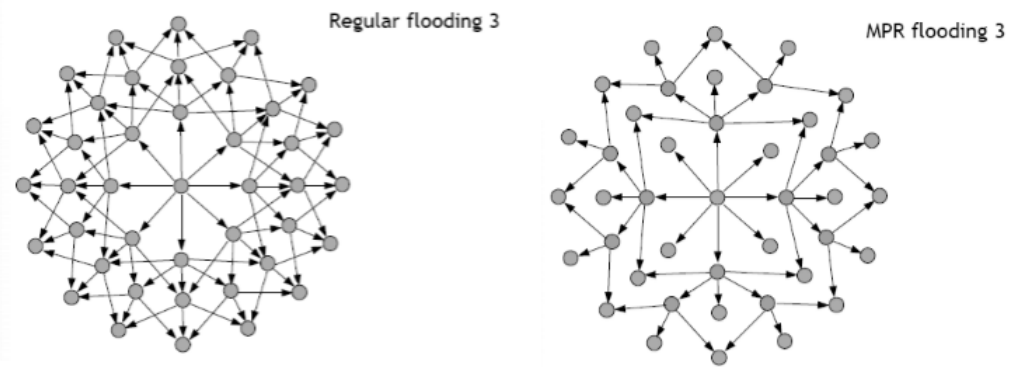
2.2. Pengujian Unjuk Kerja VANET

Pengujian unjuk kerja dari VANET menggunakan tiga jenis protokol *routing* yaitu *Optimized Link State* (OLSR), *Ad hoc on Demand Distance Vector Routing* (AODV) dan *Dynamic* Manet *on Demand* (DYMO) dengan membandingkan hasil keluaran *delay, throughput* dan *packet delivery ratio*.

2.2.1. OLSR

OLSR adalah merupakan pengoptimalan dari protokol *link state*. OLSR menggunakan konsep *multipoint relays* (MPR) untuk menyebarkan dan memperbaharui informasi kepada *node*. Di dalam OLSR hanya *node* yang dipilih sebagai *node* MPR saja yang dapat meneruskan pesan yang diterima sehingga dapat mempercepat waktu pengiriman informasi dan mengurangi kemungkinan *node* yang sama menerima pesan yang sama juga.

Dua pesan dalam *routing* OLSR adalah *hello* dan *topology control* (TC). Pesan *hello* bertugas untuk menemukan informasi kondisi *link* dan *node* tetangga. Pesan *hello* juga akan memilih MPR *selector set* yang bertugas untuk memilih *node* tetangga yang bertindak sebagai *node* MPR. Melalui pesan *hello* ini, *node* pengirim dapat menentukan *node* MPR-nya. Pesan *hello* hanya dikirim sejauh 1 *hop*, sedangkan pesan TC dikirim secara *broadcast* ke seluruh jaringan. Manfaat dari pesan TC yaitu untuk menyebarkan informasi tentang *node* tetangga yang telah ditetapkan sebagai MPR tak terkecuali MPR *selector*. TC disebarkan secara periodik dan hanya *node* MPR yang dapat meneruskan pesan TC [4][5].



Gambar 2. *Flooding* biasa vs *Flooding* MPR [5]

Pada gambar 2 digambarkan perbedaan mendasar dari *flooding* biasa dan *flooding* MPR. Pada *flooding* biasa semua *node* akan melakukan *broadcast* keseluruhan jaringan

sehingga memungkinkan *node* yang sama menerima pesan yang sama lebih dari satu kali sedangkan *flooding* MPR akan memilih *node* MPR yang bertugas untuk meneruskan pesan ke *node* yang lain sedangkan *node* yang bukan merupakan *node* MPR tidak akan meneruskan pesan ke jaringan yang lain. Sehingga, meminimalkan kemungkinan *node* yang sama mendapatkan pesan yang sama berulang kali. Berdasarkan hal tersebut, OLSR disebut sebagai pembaharuan dari versi *link state*.

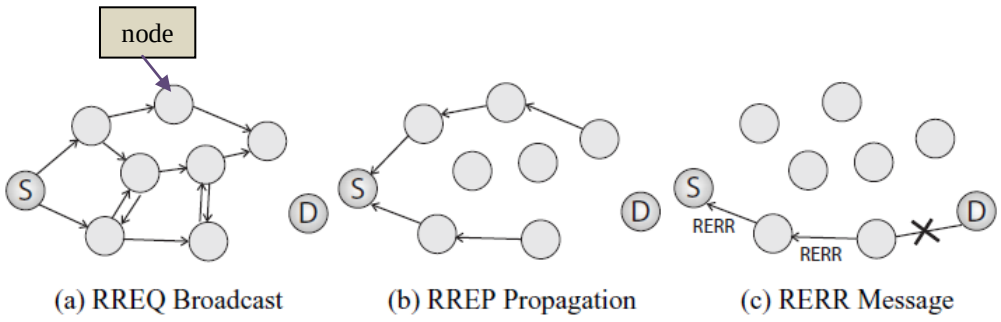
2.2.2. AODV

AODV adalah protokol *routing* yang membangun rute antara *node* pengirim dan *node*. AODV dapat digunakan pada *multicast* atau *unicast routing*. *Multicast routing* adalah pengiriman pesan dari satu jaringan ke lebih dari satu penerima sedangkan *unicast routing* adalah pengiriman pesan dari satu jaringan ke satu penerima saja. Pesan-pesan yang digunakan dalam AODV adalah *route request* (RREQ), *route reply* (RREP) dan *route error* (RERR). Pesan-pesan tersebut dikirim menggunakan pengalamatan IP.

Gambar 3a merupakan proses dari pencarian rute pengiriman yaitu dengan mengirimkan pesan RREQ ke semua *node* yang ada di sekitarnya, yang disimbolkan dengan tanda panah yang ditujukan ke arah semua *node*.

Node tetangga yang menerima RREQ akan mengirim pesan balasan kepada sumber berupa RREP jika *node* tersebut adalah penerima atau memiliki rute ke penerima seperti pada gambar 3b. *Node* yang mengetahui rute ke penerima disebut *node* penghubung. Selama rute terbentuk, setiap *node* dalam jaringan memantau kondisi *link* di depannya untuk mengantisipasi adanya kerusakan.

Apabila sebuah rute mengalami kerusakan atau terputus, maka *node* yang terhubung pada *link* tersebut akan memberitahukan ke seluruh *node* bahwa rute tersebut rusak seperti pada gambar 3c. Kemudian *node* yang bersangkutan akan menyebarkan RERR ke seluruh *node* tetangga hingga ke pengirim. RERR mengindikasikan bahwa penerima tidak dapat dicapai melalui rute yang rusak. Oleh karena itu pengirim harus menyebarkan RREQ kembali [4][6].



Gambar 3. Algoritma AODV. (a) RREQ Broadcast. (b) RREP Propagation. (c) RERR Message[6]

2.2.3. DYMO

DYMO merupakan penerus dari AODV atau bisa dikatakan sebagai pembaharuan dari *routing* AODV. DYMO memiliki desain yang sederhana serta gampang diimplementasikan. Operasi dasar dari DYMO adalah *route discovery* dan *route maintance*.

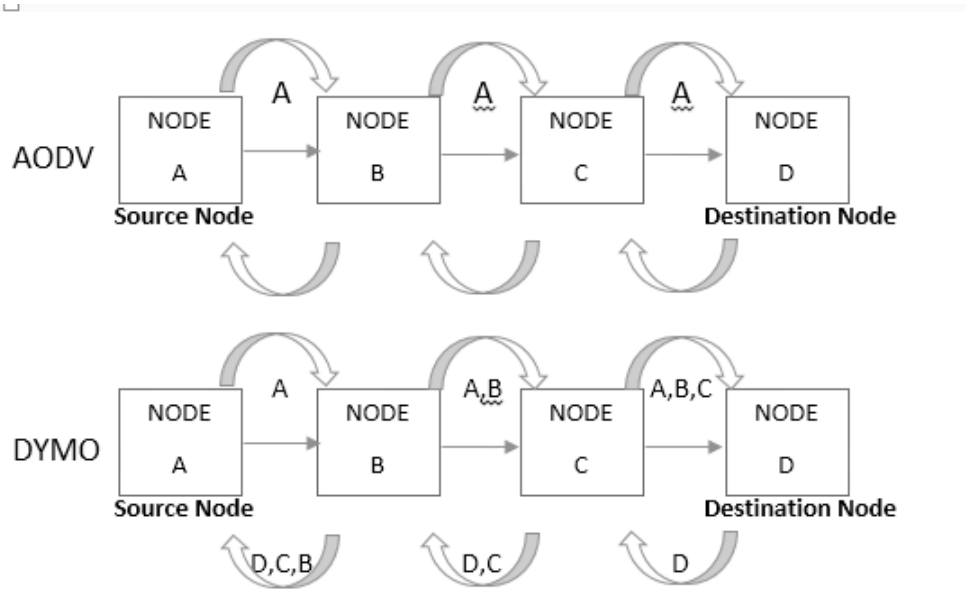
Cara kerja dari DYMO hampir sama dengan AODV yaitu, jika terdapat permintaan pengiriman pesan maka, untuk mengetahui rute ke *node* yang dituju akan diinisialisasi operasi *route discovery* dan sebuah pesan RREQ dikirimkan ke jaringan. Tiap *node* yang

berpartisipasi dalam penyebaran pesan dan menyimpan sumber rute, sama halnya dengan AODV.

Ketika *node* penerima sudah menerima RREQ maka *node* tersebut akan mengirimkan RREP ke *node* pengirim sebagai tanda bahwa pesan yang dikirimkan telah diterima. Setiap *node* yang menerima RREP akan membuat rute ke *node* pengirim sampai RREP diterima di *node* pengirim [6].

Perbedaan terbesar pada DYMO dan AODV adalah yang ditampilkan pada gambar 4. Pada AODV dalam proses pengiriman pesan, yang akan disimpan hanyalah alamat dari *source node*. Sehingga, bisa dilihat pada gambar 4 mulai dari *node* A sampai dengan *node* D sebagai tujuan, *node* yang berpartisipasi akan mengabaikan *node* apa saja yang ikut serta dalam pengiriman pesan tersebut dan hanya menyimpan *source node* yang dalam gambar 4 adalah *node* A. Rute *end to end* diantara pengirim dan penerima terbentuk setelah adanya RREP dari *destination node*.

Berbeda dengan AODV, dalam DYMO ketika proses pencarian rute dilakukan, *node* yang berpartisipasi dalam meneruskan pesan dari *source node* akan menyimpan *node* apa saja yang ikut berpartisipasi dalam pengiriman. Pada gambar 4 di bagian DYMO, dapat dilihat ketika *node* A sebagai *source node* mengirimkan pesan ke *node* B maka akan disimpan A sebagai *node* sebelumnya yang mengirimkan pesan, dan selanjutnya ketika *node* B meneruskan pesan ke *node* C maka yang disimpan tidak saja A sebagai *source node* tapi juga *node* sebelumnya yaitu B, sehingga ketika sampai di *node* C yang tersimpan adalah A,B. Proses akan berlangsung seperti itu terus sampai pesan yang dikirimkan sampai ke tujuan. Proses rute *end to end* antara pengirim dan penerima akan langsung terbentuk ketika pesan sampai pada penerima bukan ketika dikirimkan pesan RREP dari penerima. Hal ini, membuat DYMO dianggap lebih baik daripada AODV karena ketika terjadi kegagalan dalam pengiriman pesan, maka rute yang gagal tersebut langsung tidak akan digunakan lagi.



Gambar 4. AODV vs DYMO[6]

2.3. Perhitungan *delay*, *throughput* dan *packet delivery ratio*

Dalam membandingkan unjuk kerja dari *routing* OLSR, AODV dan DYMO, digunakan parameter *delay, throughput* dan *packet delivery ratio*.

Delay adalah waktu yang diperlukan suatu paket data dari *source node* hingga *destination node* [7]. Untuk mendapatkan nilai *delay* digunakan rumus sebagai berikut,

$$Delay = waktu\ paket\ diterima - waktu\ paket\ dikirimkan \quad (1)$$

Throughput adalah kecepatan pengiriman data sampai di tujuan[7]. Rumus mencari *throughput* adalah sebagai berikut,

$$\text{Throughput} = \frac{ukuran\ \ data\ \ yang\ \ diterima}{waktu\ \ pengiriman\ \ \ data} \quad (2)$$

Packet delivery ratio adalah perbandingan antara data yang diterima dan dikirim. Rumus untuk mencari PDR adalah sebagai berikut,

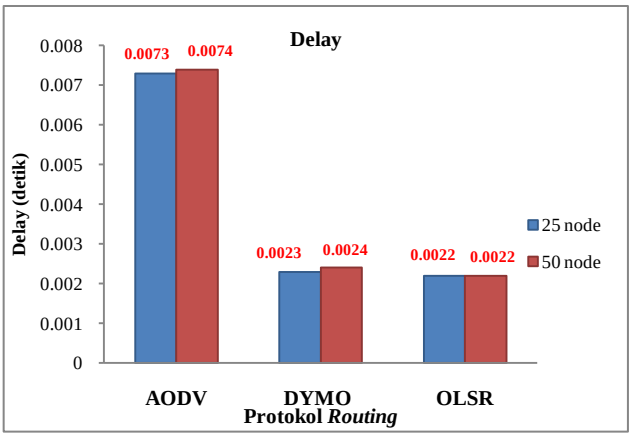
$$\text{Packet delivery ratio} = \frac{paket\ \ yang\ \ diterima}{pak\,et\ \ yang\ \ dikirim} \times 100\% \quad (3)$$

3. Pengujian dan Analisis

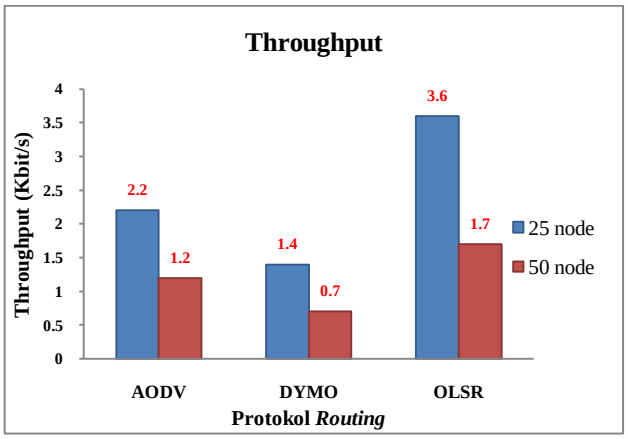
Pada simulasi jaringan VANET digunakan 25 node dan 50 *node* yang masing-masing akan dilakukan pengujian terhadap *delay, throughput* dan *packet delivery ratio*. Walaupun ketiga parameter ini bisa didapatkan dengan melakukan perhitungan secara teoritis tapi, dengan menggunakan OMNET++ dapat memudahkan dalam memperoleh hasilnya karena data-data yang dibutuhkan sudah diolah oleh OMNET++ . Hasil pengujian yang telah dilakukan ditunjukkan pada Gambar 5 – 7.

Dari hasil pada gambar 5, 6 dan 7 didapatkan analisis sebagai berikut,

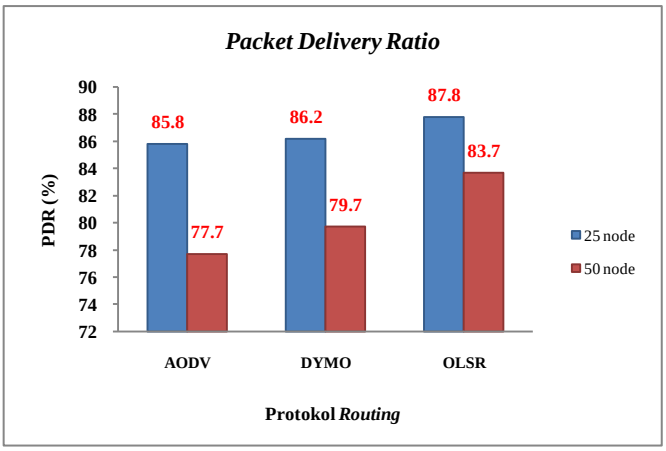
1. OLSR memiliki nilai *delay* paling kecil yaitu sebesar 0,0022, *throughput* 3,6 Kbit/s dan 1,7 Kbit/s dan *packet delivery ratio* sebesar 87,8 % dan 83,7%. Dari hasil ini didapatkan bahwa unjuk kerja dari OLSR lebih baik dibanding yang lain karena OLSR melakukan pengiriman dengan menggunakan rute terpendek dan memiliki 2 jenis pesan yaitu TC dan *hello* yang memaksimalkan kerja dari OLSR seperti menemukan *node* yang ada disekitar dan menentukan MPR. Dengan adanya MPR maka, hanya *node* MPR saja yang akan mengirimkan pesan sehingga tidak adanya penerimaan pesan ganda. Selain itu, karena merupakan mencari rute terpendek sehingga waktu pengiriman lebih cepat dan ketika terjadi kegagalan maka, proses pengiriman kembali pesan bisa lebih cepat dibanding yang lain.
2. DYMO dan AODV bekerja *hop by hop*. Setiap kali, melakukan pengiriman, maka pesan akan dikirimkan ke *node* terdekat lalu *node* tersebut akan mencari *node* terdekat untuk dikirmkan pesan tersebut, sehingga membutuhkan waktu yang lama. Hanya saja pada DYMO proses pengiriman lebih cepat daripada AODV karena DYMO menyimpan rute pengiriman sehingga dapat mempercepat waktu pengiriman dan jika terjadi kegagalan pengiriman maka DYMO akan mengirim pesan kembali dan menghapus rute yang gagal tersebut. Hal ini, meminimalkan pengulangan pengiriman terhadap rute yang gagal.



Gambar 5. Hasil *delay*



Gambar 6. Hasil *Throughput*



Gambar 7. Hasil packet delivery ratio

4. Kesimpulan

Berdasarkan pengujian dengan tiga protokol *routing* yang digunakan dalam artikel ini, didapatkan bahwa *routing* OLSR memiliki unjuk kerja yang paling baik dibandingkan dengan AODV dan DYMO dilihat dari *delay, throughput* dan PDR.

Dalam hal unjuk kerja, DYMO yang merupakan perbaikan dari protokol *routing* AODV memiliki unjuk kerja yang lebih baik daripada AODV. Tapi untuk pengujian *packet delivery ratio*, unjuk kerja antara DYMO dan AODV hampir sama, karena sama-sama menggunakan pencarian rute *hop by hop* yang membutuhkan waktu yang lebih lama untuk beradaptasi dengan bertambahnya jumlah *node* yang ada.

Daftar Pustaka

[1] C. Sommer, *Vehicle in Network Simulation* [Online], http://veins.car2x.org/, diakses pada tanggal 16 Mei 2014.

[2] A. Varga, *OMNET++* [Online], http://omnetpp.org, diakses tanggal 15 Mei 2014.

[3] R. Hilbrich, "Simulation of Urban Mobility," [Online], http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/, diakses pada tanggal 17 Mei 2014.

[4] S. Basagni [Ed], M. Conti, S. Giordano, I. Stojmenovic, *Mobile Ad hoc Networking*, Canada, 2004.

[5] A. Tonnesen, *Mobile Ad-Hoc Networks* [Online], www.olsr.org, diakses pada tanggal 31 Mei 2014.

[6] I. Khan, *Performance Evaluation Of Ad Hoc Routing Protocols For Vehicular Ad Hoc Network*, Mohammad Ali Jinah University, 2009.

[7] B. Forouzan, *Data Communications and Networking*, Fourth edition