Analisis Performansi *ProPHETv2 Routing* Berbasis *Vehicular Delay-Tolerant Network* pada Daerah *Rural*

Gumilar Hadi Prabowo[1], Rendy Munadi [2], Leanna Vidya Yovita [3]

S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Telkom University[1]
gumilarhadiprabowo@students.telkomuniversity.ac.id
Fakultas Teknik Elektro, Telkom University[2]
Fakultas Teknik Elektro, Telkom University[3]

Abstrak

Internet dan kemajuan teknologi komunikasi telah mengubah secara drastis cara orang bekerja dengan komputer. Pada implementasinya, internet menggunakan sebuah protokol yaitu *Transmission Control Protocol*/*Internet Protocol* (TCP/IP), dimana protokol ini tidak dapat bekerja bila terjadi *delay* yang cukup lama seperti yang terjadi pada daerah rural. *Vehicular Delay-Tolerant Network* (VDTN) berusaha untuk memperbaiki masalah tersebut dengan cara menyimpan paket (*Store*) bila belum memungkinkan terjadinya penerusan paket (*Packet Forwarding*). VDTN menganggap bahwa kendaraan sebagai sebuah *node* dan menggunakan paradigma *store-carry-forward* untuk proses komunikasinya, serta secara simultan memberikan informasi kepada *node* lain. *ProPHETv2* merupakan salah satu algoritma *routing* di VDTN yang mengguanakan informasi tersebut dan menambahkan parameter *contact duration* untuk menilai probabilitas pengiriman dalam membuat keputusan *Forwarding Packet*. Dengan menggunakan ONE (*Opportunistic Network Environment*) Simulator 1.5.0 RC2 akan dianalisis performa algoritma *ProPHETv2* dengan menggunakan beberapa parameter yaitu *Delivery Probability, Overhead Ratio,* dan *Average Latency*, serta menggunakan OpenJUMP 1.8.0 untuk *map processing*. Hasil simulasi yang diperoleh bahwa nilai maksimum *delivery probability* adalah sebesar 0.9618 untuk perubahan ukuran *buffer* dan 0.9541 untuk perubahan kecepatan *node.* Nilai maksimum *overhead ratio* sebesar 63.4736 untuk perubahan ukuran *buffer* dan 59.6612 untuk perubahan kecepatan *node.* Dan nilai maksimum *average latency* sebesar 1011.958 untuk perubahan ukuran *buffer* dan 1848.503 untuk perubahan kecepatan *node.*
Kata Kunci: delay, *ProPHETv2*, rural, VDTN.

1. Pendahuluan

Teknologi jaringan komputer yang dimanfaatkan ketika dalam kondisi normal adalah jaringan bersifat tetap (*fixed*) menggunakan infrastruktur sebagai pendukung. Infrastruktur adalah jaringan yang menggunakan perangkat tetap (*acces point*, router) dan bisa dikategorikan dalam kabel (*wire*) atau nirkabel (*wireless*). Media transmisi kabel merupakan teknologi yang pertama ditemukan untuk menyampaikan data antar perangkat. Sedangkan untuk media transmisi nirkabel merupakan terobosan yang memanfaatkan gelombang elektromagnetik sebagai media untuk mengirim data melalui udara (Schiller, 2003)

Realitas dari penggunaan jaringan yang menggunakan media transmisi nirkabel adalah bahwa kebiasaan pengguna tidak lagi seperti ketika menggunakan perangkat dalam jaringan kabel. Pengguna akan berpindah tempat (*mobile*) dan jaringan nirkabel mampu mengatasinya selama perangkat masih dalam jaringan.

Sejauh ini, infrastuktur jaringan baik kabel atau nirkabel, masih dibutuhkan dalam kondisi normal. Tantangan yang selanjutnya muncul adalah bagaimana jika infrastruktur tidak lagi tersedia dan komunikasi harus tetap berjalan? Tentunya dalam hal ini mobilitas pengguna masih diperhitungkan. Jika masih menggunakan kabel tentunya hal tersebut tidak lagi menjadi masalah, lalu bagaimana jika media transmisi berbasis nirkabel dan perangkat yang *mobile* mengikuti penggunanya?

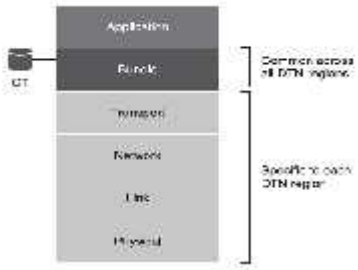
Untuk mengatasi permasalahan yang ada, terdapat inovasi berupa jaringan *ad hoc* dimana infrastuktur jaringan sudah tidak lagi dibutuhkan dan menggunakan media transmisi nirkabel dalam proses komunikasi. Tantangan selanjutnya adalah mobilitas pengguna dan perangkatnya (*node*) yang harus diatasi. Pada kondisi ini disebut dengan *Mobile Ad Hoc Network* (MANET). Menurut Zhang (2014), strategi yang digunakan dalam MANET adalah melibatkan setiap *node* agar mampu berperan sebagai pengirim pesan, router (*relay*), maupun penerima pesan.

Jaringan MANET dapat terpenuhi jika antar node masih terhubung atau koneksi tiap *node* terjamin tanpa adanya gangguan. Akan tetapi kondisi sebenarnya adalah bahwa setiap *node* terkadang terlepas/keluar jangkauan transmisi *node* lain sehingga terjadi *link failure*

(Vahdat, 2015), seperti halnya yang terjadi di daerah dengan konektivitas yang rendah (*rural*). Permasalahan ini memerlukan strategi untuk menyempurnakan MANET. Kini berkembang jaringan yang disebut dengan *Delay-Tolerant Network* (DTN). Hal yang membedakan dengan MANET adalah jaringan ini mentolerir adanya jeda waktu sampainya data (*delay*) ke penerima/tujuan. Kunci keberhasilan arsitektur jaringan DTN terletak pada protokol *routing* yang akan melakukan *forwarding message* atau menyampaiakan pesan dengan berbagai macam metode atau strategi pendekatan (Cabacas, 2010).

Arsitektur jaringan pada DTN berbeda dengan arsitektur jaringan yang lainnya. Pada jaringan lain dapat memastikan bahwa pesan dapat sampai ke tujuan karena koneksi dari *source* ke *destination* terjamin. DTN memiliki sifat yang berbeda dimana jalur antara *source* ke *destination* tidak tersedia karena jangkauan dan bergeraknya tiap *node* serta koneksi jaringan yang bersifat *intermittern* (kadang terputus dan terhubung kembali). DTN memperkenalkan *layer* baru untuk menangani kondisi jaringan yang disebut dengan *bundle layer*.

Bundle layer mengimplementasikan mekanisme *store-carry-and-forward* dimana setiap *node* dapat *store* (menyimpan) dan *carry* (membawa) pesan dalam *buffer*-nya (memori) serta dapat meneruskan pesan tersebut ke *node* lainnya yang terkoneksi (Puri, 2013). Berikut adalah arsitektur yang dipakai dalam jaringan DTN.



Gambar 1. Arsitektur DTN
(Advanced in Delay-Tolerant Network(DTNs), pp. 64-74)

Metode-metode penanganan pesan dalam DTN yaitu :

1.1. *Message-ferry-based*

Dalam metode ini, sistem biasanya menggunakan *node* lain sebagai pembawa pesan untuk disampaikan ke tujuan. Cara ini bertujuan untuk meningkatkan *delivery probability* dengan menggunakan tahap *store* (menyimpan) dan kemudian *carry* (membawa) pesan sampai dengan bertemu ke tujuan dan mengirimkannya.

1.2. *Opportunity-based*

Skema ini setiap *node* mem-*forward* pesan secara acak (*random*) dari hop ke hop sampai ke akhir tujuannya tetapi tidak menggaransi pesan dapat tersampaikan. Pada umumnya pesan akan ditukarkan hanya ketika dua node bertemu pada lokasi yang sama dan copy *message* yang sama membanjiri jaringan untuk meningkatkan jumlah pesan terkirim.

1.3. *Prediction-based*

Pada skema *prediction-based* ini, protokol *routing* menentukan *relay* (perantara) dengan mengestimasi *node* yang dapat dipercara menyampaikan pesan ke tujuan.

Pada penelitian kali ini digunakan teknologi *Vehicular Delay-Tolerant Network* (VDTN), dimana teknologi ini diterapkan pada komunikasi antar kendaraan bergerak yang tidak ada koneksi *end-to-end* secara langsung antara pengirim dan penerima. Kendaraan digunakan sebagai pembawa data informasi yang akan diteruskan ke kendaraan lain yang dikenal dengan istilah *forwarding*. Setiap kendaraan memiliki radius cakupan yang memungkinkan melakukan *forwarding* jika suatu kendaraan yang lain telah memasuki area cakupannya. Data dari pengirim akan disimpan ke dalam *buffer* dari *router* yang terpasang pada kendaraan. Selanjutnya, kendaraan akan bergerak dan meneruskan paket ke kendaraan yang lainnya sampai tujuan. Pada VDTN kendaraan dianalogikan sebagai sebuah *node* yang memiliki kemampuan *store-carry-and-forward* (Niyato, 2009).

Metode yang selalu digunakan adalah pada setiap node, ketika menerima pesan maka pesan tersebut disimpan dan dibawa sampai menemukan *node* lain untuk diperikan *copy*-nya. Protokol *routing ProPHET* (*Probabilistic Routing Protocol using History of Encounters and Transitivity*) merupakan protokol probabilistik berdasarkan metrik probabilitas bertemu dengan *node* yang ditemui serta *transitivity*-nya. Sedangkan untuk *transitivity* adalah kondisi dimana sebuah *node* dapat menjadi *relay* atau *node* perantara untuk menyampaikan pesan dari *node* lain (Lindgren, 2003).

Grasic (2011) menyatakan bahwa *ProPHET* merupakan protokol *routing* yang menggunakan probabilitas dalam proses pengirimannya dan mampu mengurangi penggunaan sumber daya pada jaringan yang terjadi pada *Epidemic Routing*. Namun, *ProPHET* memiliki suatu kelemahan yaitu ketika dua *node* saling bertemu dengan interval waktu yang singkat, maka akan menyebabkan probabilitas antara dua *node* tersebut meningkat sehingga akan mendistorsi pola mobilitas yang ada. Lahirlah *ProPHETv2* untuk mengatasi

permasalahan tersebut dengan menggunakan waktu *inter-meeting* atau *contact duration*.

Dalam *ProPHETv2*, ketika *node* A dan *node* B saling berhadapan, probabilitas pengiriman keduanya P(A,B) dihitung menggunakan faktor skala P_{enc}, sebagai berikut.

$$P(A,B) = P(A,B)_{old} + (1 - P(A,B)_{old}) \times P_{enc}$$

$$P_{enc} = P_{max} \times (Intvl / I_{typ})$$

Dimana $P_{max} \in [0,1]$ merupakan batas atas dari P_{enc}. Intvl adalah waktu *inter-meeting* dan Ityp adalah ambang batas waktu *inter-meeting*. Dalam penelitian kali ini, $Intvl/I_{typ}$ diganti menggunakan parameter *d* untuk mempertimbangkan *contact duration*, persamaannya menjadi :

$$P_{enc} = P_{max} \times d$$

Dimana *d* merupakan perhitungan *contact duration* dan durasi tersebut digunakan untuk mengirimkan pesan dari *buffer*. Persamaannya adalah.

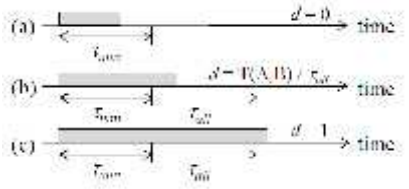
$$d = \begin{cases} 0 & ,T(A,B) \leq \tau_{min} \\ T(A,B)/\tau_{min} & ,\tau_{min} \leq \tau_{all} \\ 1 & ,T(A,B) \geq \tau_{all} \end{cases}$$

Dimana T(A,B) adalah *contact duration* yang diharapkan antara *node* A dan *node* B. Dengan T(A,B) dapat dihitung sebagai berikut.

$$T(A,B) = \alpha T_{enc}(A,B) + (1-\alpha)T(a,b)_{old}$$

Dimana $\alpha \in [0,1]$ adalah faktor beban dan $T_{enc}(A,B)$ adalah keadaan *contact duration* saat ini antara *node* A dan *node* B.

Sementara itu, *contact duration* yang diperlukan dalam pengiriman pesan di *buffer* ada tiga jenis. Pada Gambar 2(a), probabilitas pengiriman tidak perlu diperbaharui dengan *d* = 0 ketika *contact duration* lebih pendek dari τ_{min}, yang merupakan durasi waktu yang diperlukan untuk mengirimkan pesan minimum. Tetapi jika *contact duration* lebih panjang dari τ_{all}, merupakan durasi yang diperlukan untuk mengirimkan semua pesan dari *buffer*, probabilitas pengiriman akan diperbaharui menjadi *d* = 1, seperti yang terlihat dari Gambar 2(c). Jika tidak, *d* ditentukan dengan menggunakan T(A,B) / τ_{all}, seperti terlihat pada Gambar 2(b).



Gambar 2. Tiga tipe contact duration (The Evolution of a DTN Routing Protocol - ProPHETv2, 2011)

Ketika sebuah *node* memiliki probabilitas bertemu dengan *node* lain tinggi maka ia akan dipercaya untuk menjadi *relay* atau menyampaikan pesan ke tujuan dan selanjutnya pesan dititipkan ke *node* tersebut. *Node* yang populer karena dipercaya untuk menyampaikan pesan oleh *node* lain disebut juga dengan nama *hub-node*. Tentu saja *hub-node* akan memiliki beban kerja yang lebih tinggi dibandingkan dengan *node* lainnya. Beban kerja dari *hub-node* akan berdampak pada *resource* yang dimiliki karena menangani banyak pesan dari berbagai *node*.

Dalam penelitian kali ini, akan dilakukan analisis performansi *routing ProPHETv2* dan terhadap kaitan antar parameter kinerja (*delivery ratio*, *overhead ratio*, dan *average latency*) pada daerah rural. Sehingga dapat diketahui bagaimana performansi *ProPHETv2* di daerah *rural* dan menjadi pertimbangan ketika diimplementasikan di dunia nyata.

2. Metode Penelitian

2.1 Pembuatan Peta

Pada penelitian ini akan dirancang rute kendaraan yang akan dilewati oleh *mobile node*. Rute kendaraan diolah sebisa mungkin sesuai dengan kondisi aslinya. Pengolahan peta akan dibagi dalam beberapa blok sebagai berikut.

Gambar 3. Blok pengolahan peta

2.1.1. Blok *OpenStreetMap*

Pada blok ini akan dilakukan pengambilan peta pada *website* www.openstreetmap.org. Pada *OpenStreetMap* tidak bias mengambil jalur *end-to-end* yang diinginkan melainkan harus mengolahnya pada perangkat lunak berbasis GIS. *File* yang didapat dari *OpenStreetMap* bertipe* .OSM. Ada beberapa proses untuk mendapatkan *file* dari *OpenStreetMap* yakni menentukan lokasi, memperkecil area sesuai kebutuhan dan mengeksportnya ke dalam OSM file.

Gambar 4. Pengambilan area pada OpenStreetMap

Pada Gambar 4 menunjukkan bahwa rute lokasi penelitian adalah daerah Pangalengan, Jawa Barat, setelah menentukan area lokasi penentuan kemudian memperkecil luas area pada peta yang dipilih selanjutnya diekspor ke dalam file bertipe *.OSM.

2.1.2. Blok OSM2WKT.jar

Pada blok ini akan dilakukan konversi OSM file yang sebelumnya diperoleh dari *OpenStreetMap* akan dikonversi ke dalam *Well Known Text* (WKT) file yang dilanjutnya akan diolah pada perangkat lunak *OpenJUMP*. Adapaun cara penggunaan *tools* ini adalah dengan menuju direktori dari OSM2WKT.jar ini disimpan, memindahkan peta yang akan dikonversi ke dalam direktori OSM2WKT.jar dan melakukan perintah pada terminal "java -jar OSM2WKT.jar nama file". Tools ini hanya berjalan pada system operasi linux.

Gambar 5. Koversi peta menggunakan OSM2WKT.jar

Pada Gambar 5 menunjukkan bahwa sebelum melakukan konversi terlebih dahulu masuk dalam direktori OSM2WKT, kemudian dalam folder tersebut sudah terdapat peta yang siap dikonversi yaitu pangalengan.osm kemudian menjalankan perintah pada terminal Ubuntu.

2.1.3. *OpenJUMP*

Pada blok ini akan dilakukan pengolahan peta berbasis WKT file. Setelah mendapatkan file bertipe* .WKT dari OSM2WKT.jar maka peta tersebut diolah sehingga menghasilkan rute kendaraan yang dibutuhkan dalam simulasi.

Gambar 6. Peta keseluruhan pada OpenJUMP

Gambar 7. Peta rute kendaraan

Pada Gambar 6 terlihat bahwa peta keseluruhan hasil konversi dari OSM2WKT.jar yang kemudian diolah sehingga menjadi seperti Gambar 7 yakni peta rute kendaraan yang akan disimulasikan pada ONE Simulator.

2.2 Skenario dan Parameter Simulasi

Pada penelitian ini akan dibuat beberapa skenario untuk melihat kinerja jaringan VDTN dengan melihat performansi protokol routing. Seperti sudah dijelaskan sebelumnya, protokol routing yang digunakan adalah *ProPHETv2* dan *ProPHET* sebagai pembandingnya. Adapun skenario dan parameter simulasi yang digunakan pada penelitian ini adalah sebagai berikut.

2.2.1. Skenario Variasi Buffer

Pada skenario kali ini akan dilakukan perubahan terhadap ukuran *buffer* dari tiap *node* yang ada. Semakin besar ukuran *buffer* yang dimiliki maka akan semakin besar pula media penyimpanan pesan yang akan dikirimkan. Ukuran *buffer* yang digunakan dimulai dari 2.5 hingga 80 *Megabyte*.

Tabel 1: Skenario simulasi dengan perubahan ukuran *buffer*

Parameter	Nilai
Jumlah *node*	50 nodes
Ukuran *buffer* (Mbyte)	2.5, 5, 10, 20, 40, 80
Ukuran pesan (Kbyte)	100
Kecepatan *node* (m/s)	10 - 15

2.2.2. Skenario Kecepatan Node

Perubahan pada skenario kali ini terjadi pada kecepatan dari tiap *node* yang ada. Kecepatan tiap *node* akan dirubah dimulai dari 0.01 – 0.1 m/s hingga 15 - 20 m/s. Kecepatan tiap *node* akan berpengaruh terhadap *contact duration* yang terjadi antara satu *node* dengan *node* yang lainnya.

Tabel 2: Skenario simulasi dengan perubahan kecepatan *node*

Parameter	Nilai
Jumlah *node*	50 nodes
Ukuran *buffer* (Mbyte)	20
Ukuran pesan (Kbyte)	100
Kecepatan *node* (m/s)	0.01 – 0.1, 0.1 – 2, 2 - 5, 5 - 10, 10 - 15, 15 - 20

2.2.3. Parameter Simulasi

Pada penelitian yang dilaksanakan, telah ditentukan parameter-parameter dari simulasi yang bersifat tetap dan dipakai dengan nilai sama pada simulasi berbeda. Parameter-parameter tersebut adalah.

Tabel 3: Parameter Simulasi

Parameter	Nilai
Lokasi	Pangalengan, Jawa Barat
Luas Area (km)	5 x 5
Model Pergerakan	*Map Based Movement*
Waktu Simulasi	43200 detik (12 jam)
Kecepatan Pengiriman Data	9 Mbps
Antarmuka	Wifi 802.11p
Pola Pancar	*Omnidirectional*
Cakupan Area	300 m

Protokol *Routing*	*ProPHETv2, ProPHET*
Bundle Interval	10-20 detik

3. Hasil dan Pembahasan

3.1. Analisis Perubahan Ukuran *Buffer* dan Kecepatan *Node* terhadap *Delivery Probability*

Delivery probability merupakan perbandingan antara jumlah pesan yang terkirim dengan total pesan yang dibentuk. Parameter ini merepresentasikan keberhasilan pengiriman paket selama transmisi data dalam jaringan.

$$\text{Delivery Probability} = \frac{\text{Total Delivered Message}}{\text{Total Generated Message}}$$

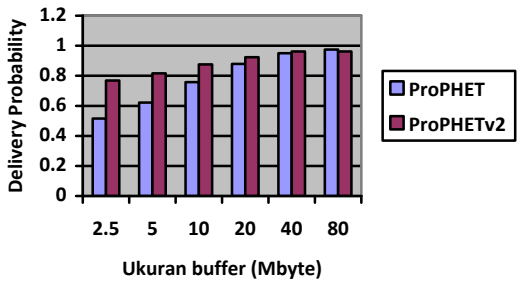
Tabel 4 dan 5 menunjukkan hasil simulasi *delivery probability* dari *ProPHETv2* dan *ProPHET*.

		Delivery Probability	
		ProPHET	*ProPHETv2*
Ukuran *Buffer* (Mbyte)	2.5	0.5159	0.7682
	5	0.622	0.8162
	10	0.7578	0.8753
	20	0.8787	0.924
	40	0.9507	0.9618
	80	0.9743	0.9618

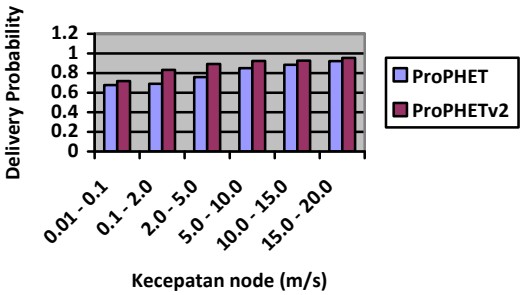
Tabel 4: Hasil simulasi *delivery probability* berdasarkan ukuran *buffer*

		Delivery Probability	
		ProPHET	*ProPHETv2*
Kecepatan Node (m/s)	0.01 – 0.1	0.6774	0.718
	0.1 – 2	0.6902	0.8328
	2 – 5	0.7591	0.8929
	5 – 10	0.85	0.9226
	10 – 15	0.8838	0.926
	15 – 20	0.922	0.9541

Tabel 5: Hasil simulasi *delivery probability* berdasarkan kecepatan *node*



Gambar 8. Delivery Probability berdasarkan ukuran buffer



Gambar 9. Delivery Probability berdasarkan kecepatan node

Terlihat dari Gambar 8, menunjukkan bahwa semakin besar ukuran *buffer* yang dimiliki maka akan semakin tinggi nilai *delivery probability*. Dari Gambar 9, dapat diketahui bahwa semakin besar kecepatan *node* maka akan semakin besar pula nilai delivery *probability*-nya. Hal ini terjadi di *ProPHET* dan *ProPHETv2*. Semakin besar nilai *delivery probability* maka akan semakin bagus performansi dari protokol *routing*. Secara garis keseluruhan performansi *ProPHETv2* lebih baik dibanding dengan *ProPHET*. Pada *ProPHETv2* memiliki maksimum *delivery probability* sebesar 0.9618 untuk perubahan ukuran *buffer* dan 0.9541 untuk perubahan kecepatan *node*, sedangkan *ProPHET* memiliki *delivery probability* maksimum sebesar 0.9743 untuk perubahan ukuran *buffer* dan 0.8838 untuk perubahan kecepatan *node*.

3.2. Analisis Perubahan Ukuran *Buffer* dan Kecepatan *Node* terhadap *Overhead Ratio*

Overhead Ratio merupakan jumlah salinan pesan/*copy message* yang dibuat dengan pesan yang terkirim. Banyaknya *copy message* akan berpengaruh terhadap *overhead ratio* karena dilihat dari perilaku protokol *routing*. Apabila terlalu banyak *copy message* yang berada dalam jaringan maka hal itu mengakibatkan penggunaan *resource* tiap *node* juga meningkat.

$$\text{Overhead Ratio} = \frac{\text{Number of Relayed Message} - \text{Number of Delivered Message}}{\text{Number of Relayed Message}}$$

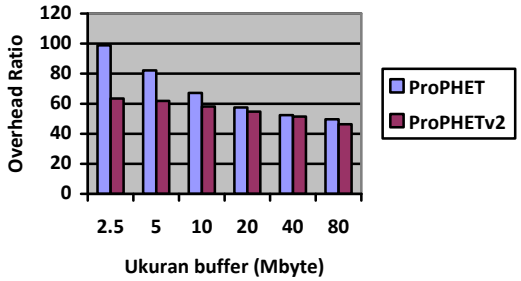
Tabel 6 dan 7 menunjukkan hasil simulasi *Overhead Ratio* dari *ProPHETv2* dan *ProPHET*.

Tabel 6: Hasil simulasi *overhead ratio* berdasarkan ukuran *buffer*

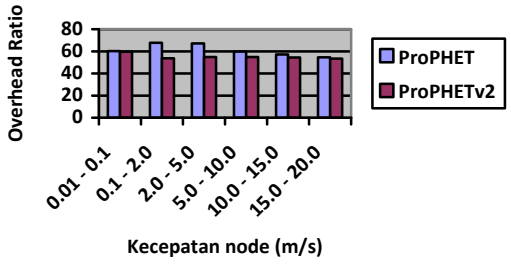
		Overhead Ratio	
		ProPHET	*ProPHETv2*
Ukuran *Buffer* (Mbyte)	2.5	98.8107	63.4736
	5	82.1907	61.8386
	10	67.1271	58.1277
	20	57.4752	54.7119
	40	52.4485	51.4159
	80	49.6033	46.3395

Tabel 7: Hasil simulasi *overhead ratio* berdasarkan kecepatan *node*

		Overhead Ratio	
		ProPHET	*ProPHETv2*
Kecepatan Node (m/s)	0.01 – 0.1	60.2015	59.6612
	0.1 – 2	67.7655	53.7748
	2 – 5	67.1771	54.9198
	5 – 10	59.7846	54.8689
	10 – 15	57.1869	54.4517
	15 – 20	54.5962	53.3892



Gambar 10. Overhead Ratio berdasarkan ukuran buffer



Gambar 11. Overhead Ratio berdasarkan kecepatan node

Terlihat dari Gambar 10, menunjukkan bahwa semakin besar ukuran *buffer* yang dimiliki maka akan semakin rendah nilai *overhead ratio.* Dari Gambar 11, dapat diketahui bahwa semakin besar kecepatan *node* maka akan relatif semakin rendah pula nilai *overhead ratio*-nya. Hal ini terjadi di *ProPHET* dan *ProPHETv2*. Semakin rendah nilai *overhead ratio* maka akan semakin bagus performansi dari protokol *routing.* Secara garis keseluruhan performansi *ProPHETv2* lebih baik dibanding dengan *ProPHET.* Pada *ProPHETv2* memiliki maksimum *overhead ratio* sebesar 63.4736 untuk perubahan ukuran *buffer* dan 59.6612 untuk perubahan kecepatan *node*, sedangkan *ProPHET* memiliki *overhead ratio* maksimum sebesar 98.8107 untuk perubahan ukuran *buffer* dan 67.7655 untuk perubahan kecepatan *node*.

3.3. Analisis Perubahan Ukuran *Buffer* dan Kecepatan *Node* terhadap *Average Latency*

Parameter *Average Latency* digunakan untuk mengetahui waktu yang dibutuhkan pesan dari *source* (asal) untuk sampai ke *destination* (tujuan).

$$Average\ latency = \frac{Sum\ of\ Latency\ of\ Delivered\ Messages}{Total\ Generated\ New\ Messages}$$

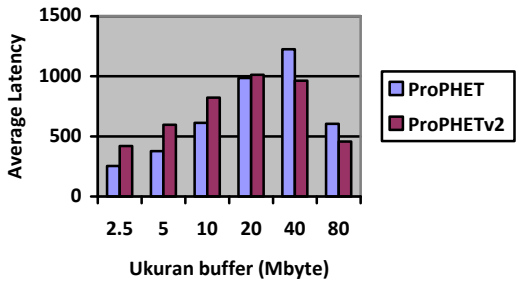
Tabel 8 dan 9 menunjukkan hasil simulasi *Average Latency* dari *ProPHETv2* dan *ProPHET.*

Tabel 8: Hasil simulasi *average latency* berdasarkan ukuran *buffer*

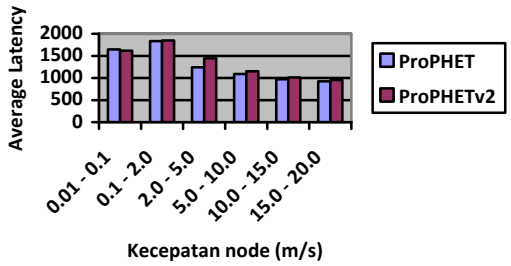
		Average Latency	
		ProPHET	*ProPHETv2*
Ukuran *Buffer* (Mbyte)	2.5	253.3091	419.2625
	5	377.3764	597.1734
	10	612.6848	823.0031
	20	985.8643	1011.958
	40	1225.272	963.53
	80	604.9865	457.5663

Tabel 9: Hasil simulasi *average latency* berdasarkan kecepatan *node*

		Average Latency	
		ProPHET	*ProPHETv2*
Kecepatan Node (m/s)	0.01 – 0.1	1647.569	1615.906
	0.1 – 2	1834.926	1848.503
	2 – 5	1242.75	1444.482
	5 – 10	1090.0759	1151.781
	10 – 15	972.3563	1012.455
	15 – 20	925.1173	952.9065



Gambar 12. Average Latency berdasarkan ukuran buffer



Gambar 13. Average Latency berdasarkan kecepatan node

Terlihat dari Gambar 12, menunjukkan bahwa semakin besar ukuran *buffer* yang dimiliki maka akan relatif semakin tinggi nilai *average latency*. Dari Gambar 13, dapat diketahui bahwa semakin besar kecepatan *node* maka akan relatif semakin rendah pula nilai *average latency*-nya. Hal ini terjadi di *ProPHET* dan *ProPHETv2*. Semakin rendah nilai *average latency* maka akan semakin bagus performansi dari protokol *routing.* Pada *ProPHETv2* memiliki maksimum *average latency* sebesar 63.4736 untuk perubahan ukuran *buffer* dan 59.6612 untuk perubahan kecepatan *node*, sedangkan *ProPHET* memiliki *average latency* maksimum sebesar 1225.272 untuk perubahan ukuran *buffer* dan 1834.926 untuk perubahan kecepatan *node*.

4. Kesimpulan

Berdasarkan hasil simulasi dan analisis menunjukkan bahwa dari sisi *delivery probability*, *overhead ratio*, dan *average latency*, *ProPHETv2* memiliki nilai yang lebih baik dibanding dengan *ProPHET* pada implementasi di daerah *rural*. Namun, besarnya *node* yang digunakan dalam simulasi berpengaruh terhadap hasil yang diperoleh. Oleh karena itu, merujuk pada penelitian M. Zhang (2008), penelitian ini menetapkan jumlah *node* yang digunakan pada daerah rural yaitu 50 *node*.

Perubahan yang dilakukan pada ukuran *buffer* akan berbanding lurus terhadap *delivery probability* dan *average latency*, tetapi akan berbanding terbalik terhadap *overhead ratio* yang dihasilkan. Sedangkan pada perubahan kecepatan *node*, hanya berbanding lurus terhadap *delivery probability* dan akan berbanding terbalik terhadap *overhead ratio* dan *average latency*.

Ucapan Terima Kasih

Terima kasih kepada Tuhan Yang Maha Esa dan keluarga yang senantiasa mendoakan penulis. Terima kasih pula penulis sampaikan kepada dosen pembimbing dan teman-teman seperjuangan yang selalu mendukung hingga penyelesaian penelitian ini.

Daftar Pustaka

Cabacas, R. A., Nakamura, H. & Ra, I-H., "Energy Consumption Analysis of Delay Tolerant Network Routing Protocols." *International Joutnal of Software Engineering and Its Applications*, vol. 8, no. 2, pp. 1-10, 2010.

D. Niyato, P. Wnag & M. Teo, " Performance Analysis of The Vehicular Delay Tolerant Network," *IEEE Communications Society Subject Matter Expert for Publication in the WCNC*, 2009.

M. Zhang & R.S. Wolff, "Routing Protocols for Vehicular Ad Hoc Networks in Rural Area," 2008.

Puri, P., Singh, M.P. (2013) "A Survey Paper on Routing in Delay-Tolerant Netwokrs", *International Conference on Information Systems and Computer Network*, pp. 2015-220.

S. Grasic, E. Davies, A. Lindgren & A. Doria, "The Evolution of a DTN Routing Protocol – ProPHETv2," 2011.

Schiller, J,. " Mobile Communication", Great Britain: Biddles, 2003.

Vahdat, Amin & Becker, David. "Epidemic Routing for Partially-connected Ad Hoc Network". Technical Report CS-200006. Duke University, April 2000. Diakses dari http://ieeexplore.ieee.org/Xplore/, diunduh pada Oktober 2016.

Zhang, Z. (2014) "Energy Consumption Analysis of Delay Tolerant Network Routing Protocols", *International Journal of Software Engineering and Its Applications*, 8 (2), pp. 1-10.