TESIS – IF 185401

PENGEMBANGAN *GREEDY PERIMETER STATELESS ROUTING* (GPSR) DENGAN KONSEP *OVERLAY NETWORK PADA* VANETS

RIZKY FENALDO MAULANA
05111850010019

Dosen Pembimbing
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Prof. Ir. Supeno Djanali, M.Sc., Ph.D.

Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
2020

(Halaman ini sengaja dikosongkan)

TESIS – IF 185401

PENGEMBANGAN *GREEDY PERIMETER STATELESS ROUTING* (GPSR) DENGAN KONSEP *OVERLAY NETWORK PADA* VANETS

RIZKY FENALDO MAULANA

05111850010019

Dosen Pembimbing

Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.

Prof. Ir. Supeno Djanali, M.Sc., Ph.D.

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

2020

(Halaman ini sengaja dikosongkan

LEMBAR PENGESAHAN TESIS

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom)
di
Institut Teknologi Sepuluh Nopember
Oleh:

RIZKY FENALDO MAULANA
NRP: 05111850010019

Tanggal Ujian: 17 Januari 2020
Periode Wisuda: Maret 2020

Disetujui oleh:
Pembimbing:

1. Dr.Eng. Radityo Anggoro, S.Kom., M.Sc. .....
   NIP: 19841016 200812 1 002

2. Prof. Ir. Supeno Djanali, M.Sc., Ph.D. .....
   NIP: 19480619 197301 1 001

Penguji:

1. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D .....
   NIP: 19810620 200501 1 003

2. Royyana Muslim Ijtihadie, S.Kom., M.Kom., Ph.D. .....
   NIP: 19770824 200604 1 001

3. Tohari Ahmad, S.Kom., MIT., Ph.D. .....
   NIP: 19750525 200312 1 002

Kepala Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas

Dr. Eng. Chastine Fatichah, S.Kom., M.Kom
NIP: 19751220 200112 2 002

(Halaman ini sengaja dikosongkan)

Pengembangan *Greedy Perimeter Stateless Routing* (GPSR) dengan Konsep *Overlay Network* pada VANETs

Nama Mahasiswa	:	Rizky Fenaldo Maulana
NRP	:	05111850010019
Pembimbing	:	Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
		Prof. Ir. Supeno Djanali, M.Sc., Ph.D.

ABSTRAK

Salah satu teknologi komunikasi nirkabel terkini adalah *Vehicular Ad hoc Networks* (VANETs). VANETs merupakan pengembangan dari *Mobile Ad hoc Networks* (MANETs). Salah satu protokol *routing* yang dinilai memberikan solusi terhadap permasalahan komunikasi di lingkungan VANETs adalah *Greedy Perimeter Stateless Routing Protocol* (GPSR). GPSR merupakan protokol *routing* berbasis posisi tiap *node* dimana rute pengiriman ditentukan dengan posisi *node* tujuan. Namun, protokol ini memiliki kelemahan ketika *node* penerus paket mengalami keadaan *local maximum*. Keadaan *local maximum* adalah *node* penerus paket tidak dapat mengirim paket ke *node* selanjutnya dikarenakan tidak ada *node* di sekitar yang memiliki posisi terdekat dengan *node* tujuan atau semua *node terdekat* sudah pernah menerima paket tersebut.

Pada penelitian ini, peneliti mengembangkan protokol GPSR tradisional dengan konsep *overlay network*. *Overlay network* diterapkan dengan menggunakan *route discovery* milik protokol *routing dynamic source routing* (DSR) sebelum paket pertama dikirim. Hasil rute dari proses *route discovery* akan menjadi acuan pencarian *virtual anchor point* (VAP). VAP merupakan representasi dari *overlay network* dan berguna untuk mengganti acuan posisi node tujuan dalam metode *greedy forwarding* untuk menghindari pemilihan *node* yang mengalami keadaan *local maximum*. Selain VAP, peneliti telah menerapkan metode area optimum dalam pemilihan node penerus paket agar tidak memilih *node* penerus paket di luar batas transmisi *node* pengirim.

Berdasarkan hasil uji coba, pencarian VAP memerlukan waktu 0.12 detik hingga 0.23 detik serta GPSR yang dimodifikasi pada skenario *real* berhasil meningkatkan nilai rata-rata PDR sebesar 72% dibandingkan dengan GPSR tradisional. Kenaikan nilai PDR yang signifikan dikarenakan GPSR tradisional tidak mampu menghindari area *void* pada skenario *real*. Pada analisa nilai rata-rata *end to end delay*, GPSR modifikasi mengalami peningkatan sebesar 1118% pada skenario *real*. Peningkatan nilai rata-rata *end to end delay* yang signifikan dikarenakan formula penghitungan nilai rata-rata *end to end delay* hanya menghitung paket yang terkirim, sedangkan jumlah paket yang terkirim pada GPSR tradisional kurang dari GPSR modifikasi. Pada analisa nilai rata-rata *routing overhead* (RO), GPSR modifikasi mengalami peningkatan sebesar 0.6% pada skenario *real*. peningkatan nilai rata-rata RO dikarenakan GPSR modifikasi

melakukan pencarian VAP sebelum melakukan pengiriman paket. Pencarian VAP dengan metode *route discovery* menambahkan jumlah paket pada RO.

Kata Kunci: *GPSR, greedy forwarding, local maximum, overlay network, Virtual anchor point.*

Improvement of *Greedy Perimeter Stateless Routing* (GPSR) with the *Overlay Network* Concept on VANETs

By	:	Rizky Fenaldo Maulana
Student Identity Number	:	05111850010019
Supervisor	:	Dr. Eng. Radityo Anggoro, S.Kom., M.Sc. Prof. Ir. Supeno Djanali, M.Sc., Ph.D.

ABSTRACT

One of the latest wireless communication technologies is Vehicular Ad hoc Networks (VANETs). VANETs are a development of Mobile Ad hoc Networks (MANETs). One routing protocol that is considered to provide solutions to communication problems in the VANETs environment is the Greedy Perimeter Stateless Routing Protocol (GPSR). GPSR is a position-based routing protocol for each node where the sending route is determined by the position of the destination node. However, this protocol has a weakness when the packet forwarding node experiences a local maximum state. The local maximum is a condition where the forwarding node cannot send the packet to the next node because there are no nearby nodes that are closest to the destination node or all the closest nodes have ever received the packet.

In this research, researchers developed the traditional GPSR protocol with the concept of overlay networks. The overlay networks concept is applied using route discovery belonging to the dynamic source routing (DSR) routing protocol before the first packet is sent. The results of the route from the route discovery process will become a reference to search for virtual anchor points (VAP). VAP is a representation of overlay networks and is useful for changing the reference position of the destination node in the greedy forwarding method to avoid selecting nodes that experience maximum local conditions. In addition to VAP, researchers have applied the optimum area method in selecting packet forwarding nodes so that they do not select packet forwarding nodes outside the sending node sending boundary.

Based on the results of simulation, the VAP search takes 0.12 seconds to 0.23 seconds and the modified GPSR in the real scenario succeeded in increasing the average PDR value by 72% compared to traditional GPSR. A significant increase in PDR value due to traditional GPSR is not able to avoid the void area in the real scenario. In the analysis of the average value of end to end delay, modified GPSR has increased by 1118% in the real scenario. The significant increase in the average value of end to end delay is because the formula for calculating the average value of end to end delay only counts the packets sent, while the number of packets sent on traditional GPSR is less than the modified GPSR. In the analysis of the average value of routing overhead (RO), modified GPSR has increased by 0.6% in the real scenario. the increase in the average value of RO is because the modified

GPSR does a VAP search before sending packages. VAP search with the route discovery method adds the number of packets to RO.

Keywords: *GPSR, overlay network, greedy forwarding, local maximum, Virtual anchor point*

KATA PENGANTAR

Puji syukur kepada Allah Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul "Pengembangan *Greedy Perimeter Stateless Routing* (GPSR) dengan Konsep Overlay Network pada VANETS".

Pengerjaan tesis ini menjadi sebuah sarana untuk penulis memperdalam ilmu yang telah didapatkan di Institut Teknologi Sepuluh Nopember Surabaya, khususnya dalam disiplin ilmu Informatika. Harapan penulis semoga apa yang tertulis di dalam buku tesis ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini. Terselesaikannya buku tesis ini tidak terlepas dari bantuan dan dukungan semua pihak. Pada kesempatan kali ini penulis ingin mengucapkan terima kasih kepada:

1. Allah SWT atas semua rahmat yang diberikan sehingga penulis dapat menyelesaikan Tugas Akhir ini.
2. Bapak Slamet Efendy Bhaskara dan Ibu Indah Isnaini selaku kedua orang tua penulis atas segala dukungan berupa motivasi serta doa sehingga penulis dapat mengerjakan Tugas Akhir ini.
3. Arifian Remianto, Kristiana dan Almh. Dian Septiayu Fendini selaku kakak penulis atas segala dukungan yang telah diberikan sehingga penulis tetap semangat dalam mengerjakan Tugas Akhir ini.
4. Bapak Dr. Eng. Radityo Anggoro, S.Kom., M.Sc., dan Bapak Prof. Ir. Supeno Djanali, M.Sc., Ph.D. selaku dosen pembimbing, atas arahan dan bantuannya dalam pengerjaan Tugas Akhir ini.
5. Sahabat Warkop x Jojoran serta teman – teman S1 TC ITS angkatan 2014 dan S2 TC ITS angkatan 2018 yang selama ini sudah membantu penulis dalam menyelesaikan Tugas Akhir ini
6. Kepada teman-teman SIMPANSE yang selama ini sudah menyemangati Penulis dalam menjalankan kuliah di ITS.
7. Juga tidak lupa kepada semua pihak yang belum sempat disebutkan satu per satu yang telah membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa masih terdapat kekurangan, kesalahan, maupun kelalaian yang telah penulis lakukan dalam penelitian ini. Oleh karena itu, saran dan kritik yang membangun sangat dibutuhkan untuk penyempurnaan tesis ini.

Surabaya, 28 Januari 2020

Rizky Fenaldo Maulana

DAFTAR ISI

DAFTAR GAMBAR

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR LAMPIRAN

(Halaman ini sengaja dikosongkan)

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Perkembangan infrastruktur transportasi darat yang signifikan serta mobilitas manusia yang selalu bergerak berbanding lurus dengan perkembangan teknologi komunikasi. Salah satu perkembangan komunikasi yang berkembang adalah *Intelligent Transportation System* (ITS). Dimana ITS memanfaatkan komunikasi, jaringan dan teknologi informasi untuk meningkatkan mobilitas, kualitas, kenyamanan dan keamanan di kota yang menerapkan konsep *smart cities*. Pada perkembangan ITS, *Vehicular Ad hoc Networks* (VANETs) dianggap sebagai inti dari semua aplikasi komunikasi dan sebagai topik menarik bagi para peneliti dari sektor industri dan akademisi di seluruh dunia. VANETs memiliki potensi untuk meningkatkan keselamatan kendaraan di jalan, efisiensi lalu lintas dan kenyamanan bagi para pengendara (Jindal dan Bedi, 2016).

VANETs memiliki beberapa karakteristik unik seperti mobilitas kendaraan yang tinggi, jarak transmisi yang terbatas, kepadatan lalu lintas, tidak ada batasan energi dan posisi kendaraan yang dapat diprediksi. Dengan karakteristik terebut, banyak peneliti mengembangkan berbagai protokol *routing* untuk meningkatkan kinerja komunikasi pada lingkungan VANETs. Algoritma *routing* pada VANETs dapat dikategorikan menjadi 5 kategori, *topology-based routing*, *position-based (geography-based) routing*, *traffic-aware routing*, *cluster-based routing* dan *broadcast-based (multicast, geocast) routing.*

Dalam perkembangannya, kategori *routing* pada VANETs berdasarkan posisi tiap *node* atau yang sering disebut dengan *position-based hybrid routing protocol* merupakan protokol *routing* terbaik dikarenakan sebelum melakukan pengiriman, node sumber dapat mengetahui dan memprediksi posisi, kecepatan dan informasi jalan node lain sehingga dapat dipastikan paket yang dikirim oleh *node* sumber dapat diterima oleh *node* tujuan dan begitu pun sebaliknya. Dibalik keuntungannya, protokol *routing* berbasis posisi memiliki kelemahan ketika *node*

mengalami masalah *local maximum*, dimana *node* tidak dapat mengirim paket ke *node* selanjutnya karena tidak ada *node* di sekitar yang memiliki posisi terdekat dengan *node* tujuan atau semua *node* di sekitar sudah pernah menerima paket tersebut (Liu *et al.*, 2016).

Greedy Perimeter Stateless Routing Protocol (GPSR) merupakan salah satu protokol *routing* berdasarkan posisi tiap *node* yang ditujukan untuk lingkungan *VANETs*. *GPSR* menggunakan metode *greedy forwarding* dalam melakukan pengiriman maupun penerusan paket. Metode ini bekerja dengan aturan bahwa *node* selanjutnya akan dipilih berdasarkan jarak terdekat dengan *node* tujuan, namun hal ini dapat menyebabkan keadaan *local maximum* seperti pada protokol berdasarkan posisi lainnya. Selain permasalahan *local maximum* pada *greedy forwarding*, pemilihan *node* selanjutnya dengan berdasarkan jarak terdekat dengan *node* tujuan dapat menyebabkan kegagalan pengiriman paket dikarenakan transmisi yang terbatas.

Permasalahan pada metode *greedy forwarding* dapat diselesaikan dengan kombinasi dengan metode terdahulu dari beberapa protokol yang telah diusulkan. Protokol *routing* berdasarkan *anchor* dapat menjadi solusi terjadinya keadaan *local maximum*. Pengiriman paket akan memanfaatkan *anchor* atau sebuah *node* yang dapat bersifat statis maupun dinamis sebagai acuan pengiriman paket. Pada protokol *routing Anchor-based Street and Traffic Aware Routing* (A-STAR) membuat *node anchor* berdasarkan lalu lintas yang didapatkan dari informasi peta secara statis dan dinamis. Peta statis digunakan untuk mengetahui rute *node* di daerah perkotaan dan peta dinamis digunakan untuk mengukur informasi lalu lintas terbaru. Kedua peta tersebut menghasilkan *anchor* sebagai acuan rute pengiriman paket. Ketika rute tersebut mencapai keadaan *local maximum*, rute tersebut akan ditandai sebagai *out of services*, lantas pada tahap ini komunikasi tidak dilakukan, lalu A-STAR membangun *anchor* jalur baru untuk komunikasi (Balasubramani, Karthikeyan dan Deepalakshmi, 2015).

Pada protokol *Greedy Perimeter Coordinating Routing* (GPCR) akan meletakan node *anchor* dengan posisi statis pada tiap persimpangan. Paket akan diteruskan ke

paket ke *node anchor* daripada mengirimnya melintasi persimpangan (Abbasi, Khan dan Ali, 2018). Metode ini dapat mengurangi terjadinya keadaan *local maximum*, namun rute yang terbuat akan lebih panjang daripada metode *greedy forwarding* pada protokol *routing* GPSR.

Pengembangan selanjutnya dari A-STAR dan GPSR adalah *Greedy Traffic Aware Routing* (GyTAR). Tidak seperti A-STAR dan GPCR yang memilih node *anchor* pada persimpangan secara statis, GyTAR mampu memilih persimpangan yang akan dilewati secara dinamis (Abbasi, Khan dan Ali, 2018). Pemilihan persimpangan berdasarkan faktor skor persimpangan maksimum (Balasubramani, Karthikeyan dan Deepalakshmi, 2015). Kelebihan pada protokol ini adalah protokol yang adaptif dalam lingkungan dengan mobilitas yang tinggi, perubahan topologi yang cepat dan sering terjadinya fragmentasi jaringan.

Permasalahan selanjutnya dari metode *greedy forwarding* adalah permasalahan transmisi yang terbatas. Pada penelitian Yang *et al*., memberikan pernyataan bahwa dalam *greedy forwarding* pada protokol *GPSR, node* sumber akan mengirimkan paket kepada *node* yang memiliki posisi terdekat dengan node tujuan tanpa memprediksi arah dan kecepatan *node* tersebut, sehingga sering kali *node* perantara tersebut telah keluar dari jangkauan radius node sumber dan menyebabkan kegagalan pengiriman paket (Yang *et al.*, 2018). Solusi yang ditawarkan adalah dengan menambahkan 2 parameter pada *greedy forwarding* yaitu durasi komunikasi kumulatif (T) dan area komunikasi optimum (Q) yang didapatkan dengan menghitung jarak antara *node* sumber, *node* perantara dan *node* tujuan dengan persamaan *Euclidean Distance*. Area komunikasi optimum didapatkan dari irisan dua lingkaran dari lingkaran transmisi node sumber dan lingkaran node tujuan dengan *node* selanjutnya yang memiliki jarak terdekat dengan node tujuan pada transmisi node sumber. Selanjutnya *node* selanjutnya akan dipilih berdasarkan komunikasi kumulatif dan berada pada radius optimum.

Pemilihan *node* selanjutnya pada radius optimum dapat memanfaatkan metode *Preferred Group Broadcast* (PGB). Metode PGB adalah metode *broadcasting* untuk memilih *node* mana yang berhak untuk melakukan penerusan paket. Metode

ini berguna untuk mengurasi *broadcast packet* dalam proses pencarian rute dan menjaga kestabilan rute serta memiliki kemampuan untuk memperbaiki rute yang ada (Boussoufa-Lahlah, Semchedine dan Bouallouche-Medjkoune, 2018). Metode PGB memilih *node* selanjutnya yang berhak meneruskan paket berdasarkan kekuatan sinyal transmisi.

Protokol *GPSR* memiliki metode *recovery*, yaitu *perimeter forwarding* dengan pengiriman berdasarkan arah jarum jam, sehingga tidak terjadi paket yang hilang dalam topologi. Metode ini dianggap kurang efektif karena dapat meningkatkan *delay* pengiriman paket dan meningkatkan jumlah *routing* paket dalam pengiriman serta rute yang dibentuk tidak efektif (Khunt, Kodinariya dan Sharma, 2016). Keadaan *local maximum* pada metode *greedy forwarding* tidak dapat dihindarkan sehingga penggunaan *perimeter forwarding* dapat terjadi berbanding lurus dengan terjadinya keadaan *local maximum*.

Untuk menghindari keadaan *local maximum* dapat menggunakan konsep *overlay network*. Konsep ini diterapkan pada protokol *Landmark Overlays for Urban Vehicular Routing Environment* (LOUVRE). LOUVRE menerapkan *routing overlay geo-proactive* dimana urutan *node overlay* ditentukan secara apriori (Khan, 2017). LOUVRE membangun *overlay network* dengan membuat *overlay link* dengan memeriksa apakah kepadatan *node* lebih tinggi dari ambang batas atau tidak (Dora, Kumar dan Joshi, 2016). O*verlay link* dibangun tanpa mempertimbangkan distribusi spasial temporal *node* tersebut. Kepadatan *node* didapatkan dengan menghitung jumlah *node* pada suatu segmen, lantas disimpan pada daftar kepadatan *node* dan akan dikirim ke semua *node tetangga*. Saat *node* menerima pesan tersebut, *node* tidak hanya menambahkan *node ID* sumber paket tersebut, tetapi juga memperbarui informasi kepadatan (Wang, Luo dan Hu, 2017).

Sedangkan pada protokol *Anchor-based Connectivity Aware Routing* (ACAR) (Pete dan Jaini, 2015) menggabungkan metode *anchor based* dan konsep *overlay network*. Protokol ini menggunakan *Route Request Beacon Message* (RRBM) dan *Route Reply Message* (RRM) untuk membangun rute komunikasi dengan

memanfaatkan *overlay network*. Rute yang dihasilkan dari RRBM dan RRM akan dijadikan rute pengiriman paket.

Dari evaluasi uji performa beberapa literatur *routing protocol* yang menerapkan konsep *overlay network* pada *greedy forwarding* menghasilkan performa yang lebih baik dari pada *routing protocol* tradisional. Peningkatan performa terjadi pada *routing protocol* ACAR (Pete dan Jaini, 2015), A-STAR (Kaur dan Meenakshi, 2018), GPCR (Sangari, 2014) dan GyTAR (Abdalla, 2017). Hal ini dikarenakan acuan dari metode *greedy forwarding* yang semula menggunakan posisi *node* tujuan diganti dengan titik dari representasi *overlay network*. Meski meningkatkan delay pada pengiriman paket, namun jumlah paket yang terkirim lebih banyak dari pada *routing protokol* yang menggunakan metode konvensional.

Pada pengembangan GPSR yang telah dilakukan didapati metode *greedy forwarding* tidak dapat menghindari keadaan *local maximum* dan metode *perimeter forwarding* sebagai mode *recovery* `pada *GPSR* memiliki beberapa kekurangan. Salah satu cara yang dapat digunakan untuk meningkatkan metode kinerja *greedy forwarding* yaitu dengan menerapkan konsep *overlay network* yang berkembang pada beberapa protokol *routing protocol* dan modifikasi dari GPSR. Konsep *overlay network* dianggap memiliki dampak signifikan dalam meningkatkan kinerja protokol *routing* serta dapat menghindari keadaan *local maximum*. Hal ini didapati pada beberapa pengembangan GPSR, yakni GyTAR dan GPCR. Metode yang diusulkan adalah modifikasi *greedy forwarding* pada protokol *GPSR* dengan acuan *Virtual Anchor Point* (VAP) dan pemilihan *node* selanjutnya menggunakan metode PGB berdasarkan posisi area optimum untuk menghindari *node* yang berada di garis terluar jangkauan *node* sumber dan kekuatan sinyal transmisi. VAP adalah sebuah representasi dari *overlay network* berupa titik maya dimana terjadi perpotongan antara *node* dengan posisi non-linear. VAP didapatkan dengan melakukan proses *route discovery* milik protokol dynamic source routing (DSR) sebelum mengirimkan paket pertama. Dengan metode ini diharapkan metode *greedy forwarding* tidak mengalami kondisi *local maximum* sehingga dapat meningkatkan kinerja protokol *GPSR*.

1.2 Perumusan Masalah

Rumusan Masalah yang diangkat dalam penelitian ini adalah sebagai berikut:

1. Bagaimana merancang modifikasi metode *greedy forwarding* dengan penerapan *virtual anchor point* (VAP) sebagai representasi dari konsep *overlay network* untuk menghindari keadaan *local maximum*?
2. Bagaimana pengaruh penerapan VAP sebagai representasi dari konsep *overlay network* terhadap kinerja protokol GPSR?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dalam pembuatan penelitian ini adalah untuk meningkatkan kinerja *greedy forwarding* pada protokol GPSR yang mampu menghindari keadaan *local maximum* dengan konsep *overlay network*. Berkaitan dengan hal tersebut, diharapkan metode pengembangan *greedy forwarding* pada protokol GPSR dapat meningkatkan nilai rata-rata pengiriman paket.

1.4 Manfaat Penelitian

Manfaat penelitian ini adalah memberikan metode *greedy forwarding* yang mampu menghindari keadaan *local maximum* dengan penerapan *virtual anchor point* (VAP) sebagai representasi dari konsep *overlay network* dan ditandai dengan meningkatnya nilai hasil pengukuran menggunakan matriks *Packet Delivery Ratio*, *End to End Delay* dan *routing overhead*.

1.5 Kontribusi Masalah

Kondisi terkini dari protokol GPSR yang ditujukan untuk lingkungan VANETs masih belum maksimal dalam melakukan proses penerusan paket. Penggunaan metode *greedy forwarding* tidak dapat dilakukan apabila *node* dalam keadaan *local maximum*, sehingga proses penerusan paket akan digantikan dengan metode *perimeter*, hal ini menyebabkan jumlah penerusan paket dan *delay* paket

bertambah. Penelitian ini mengembangkan metode *greedy forwarding* dengan penerapan *virtual anchor point* (VAP) sebagai representasi dari konsep *overlay network* untuk mengurangi masalah tersebut. Kontribusi dari penelitian ini adalah mengembangkan metode *greedy forwarding* pada protokol GPSR dengan penerapan VAP sebagai representasi dari konsep *overlay network* untuk menghindari keadaan *local maximum* pada *node* penerus.

1.6 Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Protokol yang digunakan adalah *Greedy Perimeter Stateless Routing* (GPSR).
2. Implementasi dan uji coba menggunakan simulator *Simulation of Urban Mobility* (SUMO) dan NS-2.35.
3. Simulasi dilakukan menggunakan peta berbentuk *grid* dengan ukuran 600 meter x 800 meter dan peta real, yaitu peta perumahan wisma permai Mulyosari Surabaya dengan ukuran 700 m x 700m.
4. Peta *grid* dan peta *real* memiliki area *void* di antara *node* sumber dan *node* tujuan.
5. Simulasi menggunakan *node* dengan jumlah 50, 100 dan 150.
6. Posisi *node* pengirim dan *node* penerima bersifat statis.

(Halaman ini sengaja dikosongkan)

BAB 2
KAJIAN PUSTAKA

2.1 Dasar Teori

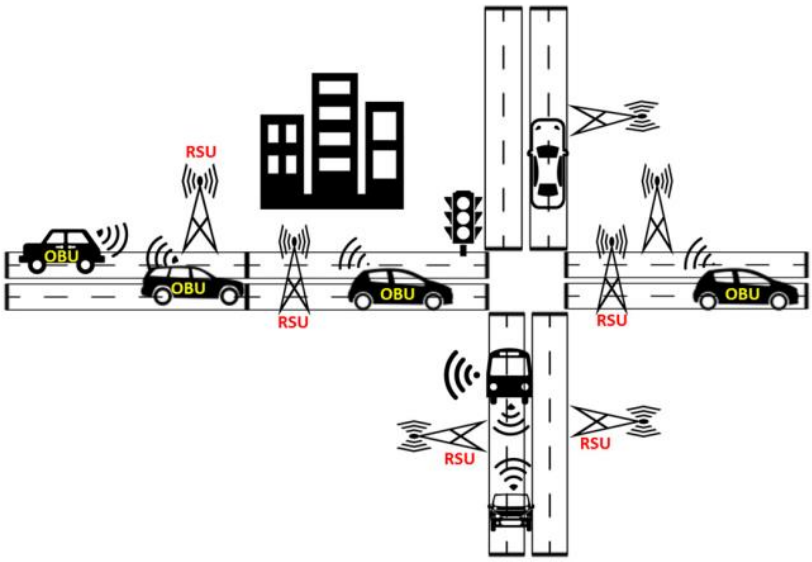
2.1.1 *Vehicular Ad hoc Networks* (VANETs)

Vehicular Ad hoc Networks (VANETs) adalah pengembangan dari *Mobile Ad hoc Networks* (MANETs) dimana setiap *node* digantikan dengan kendaraan. VANETs merupakan komponen utama dari *Intelligent Transportation System* (ITS) yang diharapkan dapat menyediakan zona keselamatan dan kenyamanan bagi pengguna jalan (Devangavi dan Gupta, 2017). Selain pada layanan keamanan, VANETs dapat digunakan pada layanan lain, seperti navigasi yang disempurnakan, pembayaran tol otomatis, manajemen lalu lintas, layanan berbasis lokasi dan mengunduh dari internet (Jindal dan Bedi, 2016).

VANETs terdiri atas kendaraan yang dilengkapi dengan perangkat *Global Positioning System* (GPS), perangkat komunikasi nirkabel *IEEE* 802.11p atau *Wireless Access for Vehicular Environment* (WAVE) serta peta digital. Perangkat *IEEE* 802.11p dan WAVE membentuk standar *Dedicated Short Range Communication* (DSRC) untuk komunikasi antar kendaraan pada VANETs. Standar WAVE mendeskripsikan seperangkat standar *IEEE* 1609.x (.1 / .2 / .3 / .4) yang digunakan pada *MAC layer* (*layer* 2) dan *Network Layer* (*layer* 3) dari model *OSI Layer*. Pada *Physical Layer* (*layer* 1), standar *IEEE* 802.11p digunakan. DSRC ditujukan untuk menyediakan transfer data yang tinggi dengan latensi komunikasi yang rendah di zona komunikasi kecil sehingga dianggap sebagai standar yang tepat untuk komunikasi nirkabel pada jaringan VANETs. DSRC menawarkan *bandwidth* secara teori sebesar 6 hingga 27 Mbps, jangkauan nirkabel antara 300 meter sampai 1000 meter dan mendukung kecepatan kendaraan melebihi 200 Km / jam (Boussoufa-Lahlah, Semchedine dan Bouallouche-Medjkoune, 2018).

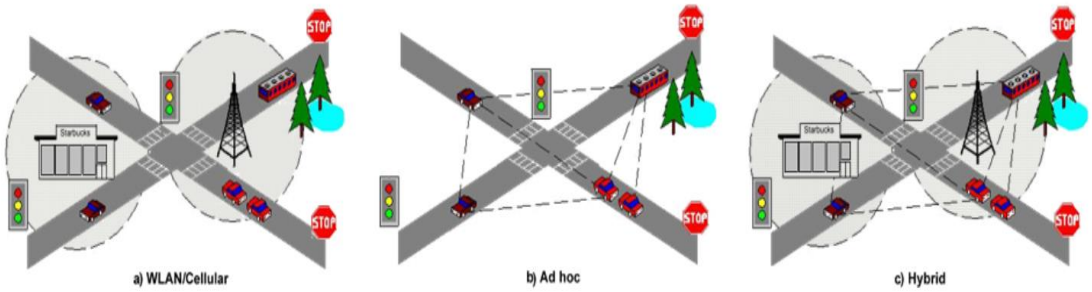
2.1.1.1 Arsitektur VANETs

VANETs mengikuti prinsip komunikasi dan arsitektur yang sama dengan MANETs dalam hal *self-management*, *self-origination* dan *shared radio*. Komunikasi dalam VANETs dapat diklasifikasikan menjadi 2 kelas utama, yaitu komunikasi *Vehicle-to-Vehicle* (V2V) dan komunikasi *Vehicle-to-Infrastructure* (V2I) (Abdel-Halim dan Fahmy, 2018). Pada komunikasi V2V, kendaraan dapat berkomunikasi secara langsung satu sama lain, sedangkan pada komunikasi V2I, kendaraan dapat berkomunikasi dengan infrastruktur tidak bergerak di sisi jalan.



Gambar 2.1 Arsitektur VANETs (Hasrouny *et al.*, 2017)

Komponen utama dari VANETs adalah *Application Unit* (AU), *On Board Unit* (OBU) dan *Road Side Unit* (RSU) (Sangari, 2014). Pada Gambar 2.1 menggambar peletakan komponen pada arsitektur VANETs. RSU terletak pada posisi yang statis, digunakan untuk menjadi letak layanan aplikasi dan menerapkan SDRC. OBU adalah perangkat komunikasi yang menggunakan layanan aplikasi dan terletak pada setiap kendaraan. Sedangkan AU adalah aplikasi dari jaringan tersebut dan dapat terletak terdapat pada setiap kendaraan maupun pada RSU.



Gambar 2.2 Tipe Arsitektur Pada VANETs (Tufail dan Dawood, 2016)

Pada Gambar 2.2 menunjukkan tipe dari arsitektur VANETs, diklasifikasikan menjadi 3 kelas, yaitu (a) *Pure Cellular*, (b) *Pure Ad hoc* dan (c) *Hybrid* (Tufail dan Dawood, 2016). Pada kelas arsitektur jaringan *Pure Cellular*, setiap kendaraan memiliki akses internet melalui *access point* atau *cellular tower* terdekat sehingga memudahkan komunikasi antar kendaraan. Sedangkan pada kelas arsitektur jaringan *Pure Ad hoc*, setiap kendaraan dapat berkomunikasi secara langsung melalui OBU. Karena OBU memiliki jarak transmisi yang terbatas, komunikasi dibantu oleh RSU. Selanjutnya, pada kelas arsitektur jaringan *Hybrid*, merupakan kombinasi dari *pure cellular* dan *pure Ad hoc* dimana setiap kendaraan dapat berkomunikasi secara fleksibel dengan kendaraan lain melalui internet maupun jaringan *Ad hoc*.

2.1.1.2 Karakteristik VANETs

VANETs memiliki beberapa karakteristik yang menguntungkan untuk membantu kinerja protokol *routing* antara lain kendaraan memiliki daya komputasi yang tinggi, pergerakan kendaraan dibatasi oleh besar dan panjang jalan yang telah ditentukan sebelumnya, tidak ada batasan energi, posisi kendaraan dapat diprediksi dengan layanan GPS dan kendaraan dapat mengakses infrastruktur RSU yang tersedia (Saleh, Gamel dan Abo-Al-Ez, 2017).

Selain beberapa karakteristik yang menguntungkan, VANETs memiliki beberapa karakteristik unik yang harus dipenuhi oleh protokol *routing*, antara lain mobilitas kendaraan yang tinggi, jarak transmisi yang terbatas (Nebbou dan Lehsaini, 2018). kepadatan lalu lintas dan adanya penghalang sinyal seperti pohon, gedung dan gunung (Rehan *et al.*, 2015).

2.1.1.3 Model Mobilitas

Dengan membuat model mobilitas yang realistis untuk lingkungan VANETs harus mempertimbangkan semua karakteristik yang dimiliki VANETs. Berdasarkan hal tersebut, tipe mobilitas dapat diklasifikasikan menjadi 2 kelas, yaitu makroskopis dan mikroskopis (Abdel-Halim dan Fahmy, 2018). Kelas makroskopis menyertakan semua karakteristik yang berdampak pada pola kendaraan seperti posisi awal dan tujuan, rambu lalu lintas serta penghalang yang ada. Sedangkan kelas mikroskopis menyertakan semua karakteristik yang terkait dengan perilaku kendaraan seperti profil pengemudi dan profil kendaraan.

2.1.1.4 Tipe Pesan

Pada lingkungan VANETs, terdapat 4 pesan dasar yang dapat dipertukarkan antara kendaraan dan infrastruktur, antara lain pesan darurat, pesan pribadi, pesan *routing* dan pesan multimedia dan informasi (Jindal dan Bedi, 2016). Pesan darurat dapat berupa semua jenis kejadian pada lalu lintas, seperti kecelakaan, mobil ambulans, mobil pemadam kembaran, mobil polisi, peringatan perbaikan jalan, kemacetan dan sebagainya. Pesan pribadi digunakan untuk menukar informasi terkait profil pengemudi dan kendaraan. Pesan *routing* digunakan oleh protokol routing untuk saling berbagi informasi terkait kecepatan, posisi, arah, identitas kendaraan dan sebagainya. Sedangkan pesan multimedia dan informasi sebagai pembagi informasi terkait layanan atau sumber daya yang tersedia baik di pinggir jalan maupun layanan oleh kendaraan lain, pesan ini dapat berupa informasi titik makanan terdekat, pengisian bensin dan sebagainya.

2.1.1.5 Protokol *Routing* pada VANETs

Routing pada VANETs adalah metode untuk menemukan jalur terpendek antara *node* sumber dan *node* tujuan dan meneruskan paket secara tepat waktu (Ullah *et al.*, 2019). Tugas berat yang harus dilakukan oleh algoritma *routing* adalah mendapatkan rute dengan delay yang minimum dengan penggunaan sumber daya jaringan yang minimum (Goel, Sharma dan Dhyani, 2017). Protokol *routing* pada *VANETs* dibagi menjadi 5 kategori (Liu *et al.*, 2016), yaitu *topology-based routing*, *position-based (geography-based) routing*, *traffic-aware routing*, *cluster-based routing* dan *broadcast-based (multicast, geocast) routing.* Pada perkembangannya,

Position-based routing diperkenalkan sebagai solusi yang tepat untuk lingkungan dinamis seperti VANETs dikarenakan proses *routing* tanpa jalur dan hanya mempertahankan informasi lokasi geografis dari *node* tetangga daripada mempertahankan informasi *routing*.

2.1.2 *Greedy Perimeter Stateless Routing* (GPSR)

Greedy perimeter stateless routing protocol (GPSR) merupakan salah satu protokol *routing proactive* yang bekerja berdasarkan posisi dari setiap *node* (Yang, Yu dan Zeng, 2017). Protokol GPSR diusulkan pertama kali pada tahun 2000 untuk jaringan datagram nirkabel (Qureshi, Abdullah dan Lloret, 2016). Dikarenakan GPSR bekerja berdasarkan posisi setiap *node*, maka GPSR merupakan protokol yang cocok diimplementasikan dalam lingkungan VANETs. Algoritma GPSR disusun atas tiga komponen utama, yaitu *beacon message*, *greedy forwarding* dan *perimeter forwarding*.

GPSR memiliki beberapa kelebihan dan kekurangan apabila diimplementasikan pada lingkungan VANETs yang memiliki karakteristik yang unik. Berbeda pada protokol *proactive* atau *reactive* lainnya seperti *Ad hoc on Demand Distance Vector* (AODV), *Destination Sequenced Distance Vector* (DSDV) maupun *Dynamic Source Routing* (DSR), GPSR memiliki kelebihan mampu mendapatkan informasi pergerakan setiap *node* dengan bantuan GPS dan dapat menghindari terjadinya *flooding message* pada jaringan (Houssaini *et al.*, 2016). Namun, GPSR juga memiliki beberapa kekurangan, yaitu GPSR tidak memperhatikan kecepatan dan arah kendaraan (Silva, Reza dan Oliveira, 2018). Kecepatan *node* yang tinggi dan arah yang tidak diperkirakan memungkinkan terjadinya paket los karena *node* selanjutnya sudah tidak berada dalam jarak transmisi *node* pengirim. GPSR menggunakan metode antrean *First In First Out* (FIFO) dalam menerima paket. Metode ini memungkinkan terjadinya peningkatan *buffer* pada *node* ketika beberapa *node* melakukan *broadcast* ke *intermediate node* yang sama secara berturut turut. Hal ini dapat menyebabkan *network congestion* dan *delay* yang tinggi

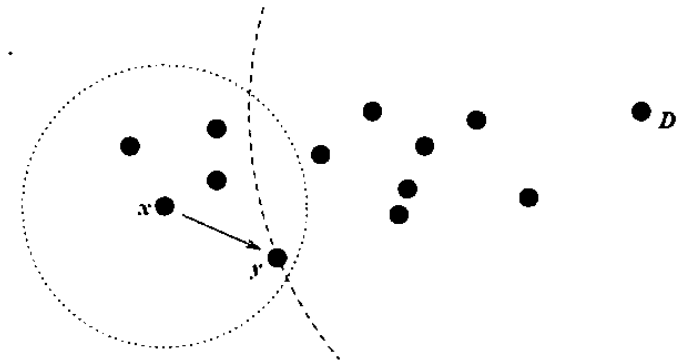
karena kecepatan proses paket lebih lambat dari pada kecepatan paket yang diterima (Hu *et al.*, 2015)

2.1.2.1 *Beacon Message*

Untuk mengetahui posisi setiap *node*, GPSR menggunakan *beacon message* dalam interval waktu yang telah ditentukan. *Beacon message* akan dikirimkan setiap interval waktu oleh setiap *node* ke semua *node* tetangga atau *node* selanjutnya yang memiliki posisi pada radius transmisi node tersebut. Pesan ini digunakan untuk memperbarui *neighbor table* untuk membantu keputusan dalam melakukan pengiriman paket (Patel, 2017). Selain berisi informasi posisi *node*, *beacon message* juga dapat mengirimkan informasi arah dan kecepatan setiap *node* (Nebbou dan Lehsaini, 2018).

2.1.2.2 Greedy Forwarding

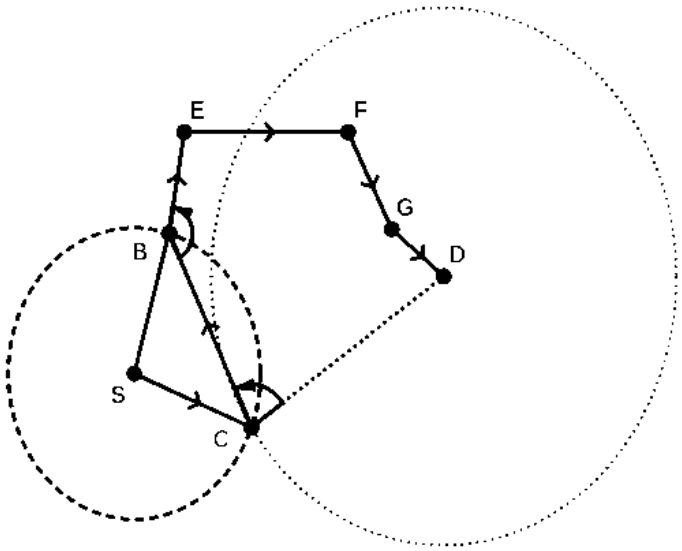
Greedy Forwarding merupakan salah satu inti dari kinerja GPSR. Pada *Greedy Forwarding*, GPSR akan menentukan *node* selanjutnya yang memiliki jarak terdekat dengan *node* tujuan dalam pengiriman paket (Anggoro, Husni dan Bastian, 2018). Metode ini akan dilakukan hingga paket sampai pada *node* tujuan dengan menggunakan referensi *neighbor table* yang selalu diperbarui oleh hasil dari *beacon message*.



Gambar 2.3 Simulasi *Greedy Forwarding*

Pada Gambar 2.3 menggambarkan terjadinya proses *greedy forwarding*. Ketika n*ode* x akan mengirimkan paket ke *node* D, maka GPSR akan memprioritaskan penggunaan metode *greedy forwarding* daripada *perimeter forwarding*. *Node* x akan mencari *node* selanjutnya yang memiliki posisi terdekat

dengan *node* D. Setelah melakukan proses penghitungan jarak dan tanpa memperhitungkan arah dan kecepatan *node* selanjutnya maka *node* x akan memilih *node* y sebagai *intermediate node* untuk meneruskan paket ke *node* D.



Gambar 2.4 Contoh skenario simulasi *GPSR*

Greedy forwarding tidak selalu sukses meneruskan paket ke *node* tujuan. Pada Gambar 2.4 menggambarkan *node* s sebagai pengirim dan *node* D sebagai tujuan. *Node* S akan mengirimkan paket ke *node* D, bila menggunakan *greedy forwarding*, maka node S akan memilih *node* C sebagai *intermediate node*. Namun, ketika akan meneruskan paket, *node* C tidak dapat meneruskan ke *node* D karena tidak ada *node* selanjutnya yang memiliki posisi terdekat dengan *node* D selain *node* C sendiri, maka keadaan ini disebut dengan *local maximum* (Silva, Reza dan Oliveira, 2018).

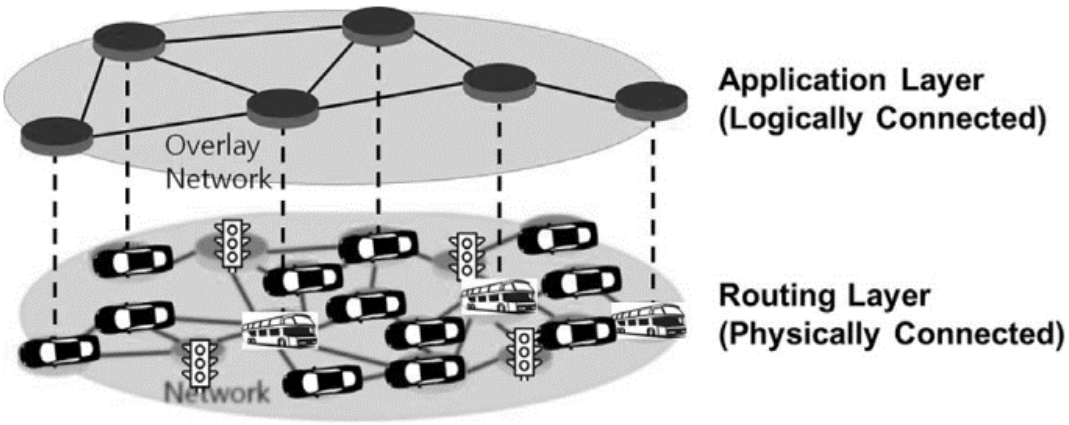
2.1.2.3 *Perimeter Forwarding*

Perimeter forwarding akan digunakan apabila pada proses *greedy forwarding* gagal atau terjadi *local maximum* (Yang, Yu dan Zeng, 2017). *Perimeter forwarding* menggunakan aturan *right hand* dalam menentukan *node* selanjutnya untuk meneruskan paket (Patel, 2017). Aturan *right hand* adalah suatu algoritma dimana pemilihan *node* selanjutnya mengikuti arah jarum jam. Pada Gambar 2.4 menggambarkan terjadinya *local maximum* pada *node* C. *Node* C akan melakukan *perimeter forwarding* dan memilih *node* B sebagai *node* selanjutnya berdasarkan aturan *right hand*. Pada proses penerusan paket hingga *node* D sebagai tujuan,

semua *node* akan menggunakan *perimeter forwarding*. *Perimeter Forwarding* memiliki beberapa kelemahan, antara lain rute yang dihasilkan lebih jauh, tidak efisien dan memiliki delay yang tinggi (Khunt, Kodinariya dan Sharma, 2016).

2.1.3 *Overlay Network*

Overlay Network dibangun diatas jaringan lain yang menggunakan kumpulan *node* sebagai representasinya (Singh, 2015). Jaringan lain yang dimaksud adalah jaringan yang terhubung secara fisik seperti peta kota, rute bus dan sebagainya. Dalam jaringan fisik tersebut, persimpangan adalah tempat penting dimana keputusan akan dibuat.



Gambar 2.5 *Overlay Network* dalam *OSI Layer*

Pada Gambar 2.5 Mendeskripsikan bahwa *overlay network* berada pada *application layer*, dimana *layer* tersebut berada tepat diatas *routing layer*. Konsep ini dapat diimplementasikan pada lingkungan VANETs dengan penggunaan *overlay network* sebagai pemilihan persimpangan yang akan digunakan sebagai acuan atau *node relay* dengan dapat memperhatikan kepadatan lalu lintas, peta kota, informasi rute dan sebagainya (Jang, Ahn dan Han, 2017). Dalam penentuan persimpangan, *overlay network* harus memperhatikan bagaimana konektivitas antar *node*. Pada lingkungan VANETs, semakin banyak kendaraan di jalan,

konektivitasnya semakin baik, dengan asumsi kendaraan didistribusikan secara seragam (Wang, Luo dan Hu, 2017).

Dalam proses pencarian rute dengan konsep *overlay network* harus memperhatikan metode *broadcasting* yang akan digunakan. Pemilihan *node* selanjutnya terdekat akan meningkatkan *hop* transmisi, sedangkan memilih *node* selanjutnya dengan posisi lebih jauh akan mengakibatkan *unstable link* karena pergerakan *node* atau gangguan sinyal (Boussoufa-Lahlah, Semchedine dan Bouallouche-Medjkoune, 2018). Permasalahan ini dapat dijawab dengan mengimplementasikan metode *Preferred Group Broadcast* (PGB) dimana pemilihan *node* dengan jarak moderat sebagai *node* selanjutnya untuk menyediakan *node relay* yang stabil untuk algoritma *routing*.

2.2 Studi Literatur

2.2.1 *Overlay Network* pada protokol *routing*

Landmark Overlays for Urban Vehicular Routing Environment (LOUVRE) merupakan salah satu protokol *routing* yang menggunakan konsep *overlay network*. LOUVRE menerapkan *routing overlay geo-proactive* dimana urutan *node overlay* ditentukan secara apriori (Khan, 2017). LOUVRE membangun *overlay network* dengan membuat *overlay link* dengan memeriksa apakah kepadatan *node* lebih tinggi dari ambang batas atau tidak (Dora, Kumar dan Joshi, 2016). O*verlay link* dibangun tanpa mempertimbangkan distribusi spasial temporal *node* tersebut. Kepadatan *node* didapatkan dengan menghitung jumlah *node* pada suatu segmen, lantas disimpan pada daftar kepadatan *node* dan akan dikirim ke semua *node tetangga*. Saat *node* menerima pesan tersebut, *node* tidak hanya menambahkan *node ID* sumber paket tersebut, tetapi juga memperbarui informasi kepadatan (Wang, Luo dan Hu, 2017).

Dengan metode ini, saat pencarian rute dilakukan maka sebagian besar rute akan menggunakan *overlay link* yang sama sehingga protokol ini menjamin optimalisasi rute global dan mengurangi penundaan untuk membuat rute *overlay*. Kelemahan

pada protokol ini adalah skalabilitas dan kegagalan pengiriman ketika kepadatan tidak dapat dideteksi oleh *node* mana pun.

Anchor-based Connectivity Aware Routing (ACAR) merupakan penelitian lanjut dari *Connectivity Aware routing* (CAR) (Pete dan Jaini, 2015). Protokol ini menggunakan *Route Request Beacon Message* (RRBM) dan *Route Reply Message* (RRM) untuk membangun rute komunikasi dengan memanfaatkan *overlay network*. *Node* sumber, *node* tujuan, maupun *node* perantara yang berada pada *anchor* atau persimpangan akan memberitahu *node* tetangga tentang perubahan lokasi dikarenakan perubahan arah dan kecepatan pada *node*. Protokol ACAR mampu mengurangi jumlah hop dengan menggunakan metode *greedy forwarding*.

2.2.2 Anchor Based Routing Protocol

Anchor Based Routing Protocol adalah kelompok protokol *routing* yang memanfaatkan *anchor* atau sebuah node yang dapat bersifat statis maupun dinamis sebagai acuan pengiriman paket. *Anchor-based Street and Traffic Aware Routing* (A-STAR) merupakan salah satu protokol *routing* berdasarkan posisi *node*. Protokol ini menggabungkan informasi posisi dan informasi peta, dapat mengevaluasi kemampuan komunikasi jaringan setiap jalan dengan jumlah *node* di jalan (Qin dan Yu, 2017). Pencarian rute dilakukan dengan kesadaran lalu lintas berdasarkan informasi peta secara statis dan dinamis. Peta statis digunakan untuk mengetahui rute *node* di daerah perkotaan dan peta dinamis digunakan untuk mengukur informasi lalu lintas terbaru. Kedua peta tersebut menghasilkan *anchor* sebagai acuan pengiriman paket.

Jalur optimal dipilih menggunakan algoritma Dijkstra. Protokol A-STAR mengusulkan teknik baru untuk proses pemulihan, yaitu ketika jalan mencapai keadaan *local maximum*, sementara itu ditandai sebagai *out of services*, lantas pada tahap ini komunikasi tidak dilakukan, lalu A-STAR membangun *anchor* jalur baru untuk komunikasi (Balasubramani, Karthikeyan dan Deepalakshmi, 2015).

Greedy Perimeter Coordinating Routing (GPCR) merupakan protokol *routing* yang ditujukan untuk lingkungan kota. Gagasan utama dari protokol ini adalah

untuk mengambil keuntungan dari grafik planar alami tentang jalan dan persimpangan (Oubbati *et al.*, 2017). Protokol ini terdiri dari dua bagian, yaitu metode *greedy forwarding* yang terbatas, yang meneruskan pesan ke *node* di persimpangan dan metode *recovery*, yang didasarkan pada topologi jalan dan persimpangan dunia nyata (Abdalla Ahmed *et al.*, 2017).

Metode *greedy forwarding* yang terbatas mengikat *node* pembawa paket untuk meneruskan paket ke *node anchor* yang terletak di setiap persimpangan daripada mengirimnya melintasi persimpangan (Abbasi, Khan dan Ali, 2018). Setelah paket mencapai *node anchor*, paket ini akan ditransmisikan dengan metode *greedy forwarding* yang terbatas mencapai tujuan. Namun, jalan berikutnya yang akan diambil ditentukan tanpa mempertimbangkan apakah ada cukup banyak *node* di jalan (Lahlah *et al.*, 2016). *Node anchor* akan mengalami kondisi *local maximum* ketika tidak dapat meneruskan paket. Metode *recovery* yang ditawarkan adalah dengan menggunakan *perimeter forwarding*. GPCR tidak mempertimbangkan kasus kepadatan kendaraan rendah saat melakukan *routing* dan tidak mempertimbangkan status tautan saat meneruskan paket-paket yang dapat mengakibatkan hilangnya paket yang berlebihan

Greedy Traffic Aware Routing (GyTAR) merupakan salah satu protokol *routing* berdasarkan lokasi yang dirancang khusus untuk daerah perkotaan. Tidak seperti A-STAR dan GPCR yang memilih *node anchor* pada persimpangan secara statis, GyTAR mampu memilih persimpangan yang akan dilewati secara dinamis (Abbasi, Khan dan Ali, 2018). Pemilihan persimpangan berdasarkan faktor skor persimpangan maksimum S_j (Balasubramani, Karthikeyan dan Deepalakshmi, 2015) dengan persamaan sebagai berikut:

$$S_j = \alpha * T_j + \beta * D_j \tag{2.1}$$

dengan:

S_j = Skor persimpangan maksimum

α = Nilai bobot dari faktor pertama

β = Nilai bobot dari faktor kedua

T_j = Kepadatan lalu lintas

D_j = Jarak metrik kurva

Setelah mendapat kan bobot persimpangan, maka *node* akan melakukan penerusan paket dengan metode *improved greedy forwarding* dimana *node* pengirim memperkirakan lokasi saat ini dari tetangga mereka, sebelum memilih penerusan berikutnya dan menggunakan informasi vektor kecepatan yang dipertukarkan dalam pesan *beacon*. Ketika metode *improved greedy forwarding* mengalami kondisi *local maximum* maka metode recovery yang ditawarkan oleh GyTAR adalah dengan menggunakan metode *carry-and-forwards* (Alsharif, 2017). Kelebihan pada protokol ini adalah protokol yang adaptif dalam lingkungan dengan mobilitas yang tinggi, perubahan topologi yang cepat dan sering terjadinya fragmentasi jaringan. Sedangkan kelemahan dalam protokol ini adalah tidak mampu menghindari jalan yang kosong dan kinerja protokol tergantung pada kinerja RSU dikarenakan protokol mengasumsikan bahwa informasi jumlah *node* di suatu jalan akan diberikan oleh RSU (Abdalla, 2017).

2.2.3 *Preferred Group Broadcast* (PGB)

Metode PGB adalah metode *broadcasting* untuk memilih *node* mana yang berhak untuk melakukan penerusan paket. Metode ini berguna untuk mengurasi *broadcast packet* dalam proses pencarian rute dan menjaga kestabilan rute serta memiliki kemampuan untuk memperbaiki rute yang ada (Boussoufa-Lahlah, Semchedine dan Bouallouche-Medjkoune, 2018). Metode PGB memilih calon *node* yang berhak meneruskan paket berdasarkan kekuatan sinyal transmisi calon *node* tersebut dan mengklasifikasikan *node* di sekitarnya menjadi 3, yaitu PG, *In Group* dan *Out Group* (Khan, 2017). PG adalah kelompok *node* yang memiliki nilai kekuatan sinyal yang disukai, *In Group* adalah kelompok *node* yang memiliki sinyal lebih kuat daripada PG sedangkan *Out Group* adalah kelompok *node* yang memiliki sinyal lebih lemah daripada PG.

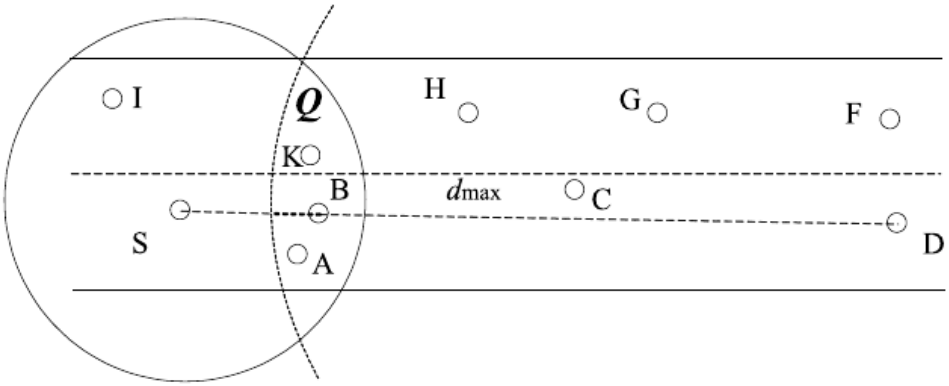
Kelemahan dari protokol PGB adalah masalah pada area dengan kepadatan kendaraan rendah seperti jalan raya. Lebih jauh, dalam PGB satu *node* diizinkan untuk disiarkan dan karena PG tidak selalu merupakan *node* yang membuat kemajuan paling maju ke tujuan, penemuan rute mungkin memakan waktu lebih lama dari sebelumnya. Kerugian lain adalah siaran tak dapat dihentikan jika grup ditemukan kosong sebagaimana dibahas bahwa PGB bekerja lebih buruk di jaringan yang jarang (jalan raya). Duplikasi paket dapat terjadi ketika dua *node* dalam siaran PG pada saat yang sama. Itu juga menciptakan overhead yang sama seperti protokol DSR ketika berurusan dengan duplikasi siaran untuk menambah pendahulu ke dalam paket asli.

Metode PGB diterapkan pada protokol *routing Connectivity Aware Routing* (CAR). Pada protokol ini, PGB dikombinasikan dengan metode *Advance Greedy Forwarding* (AGF) serta metode *guards* yang berfungsi untuk menjaga dan memperbarui informasi kendaraan dalam suatu zona (Liu *et al.*, 2016). Terdapat 2 jenis *guards*, yaitu *standing guards* dan *travelling guards* (Goel, Sharma dan Dhyani, 2017). *Standing guards* merepresentasikan informasi keadaan sementara pada suatu area tertentu, sedangkan *travelling guards* merepresentasikan informasi kecepatan, posisi dan radius.

CAR bekerja dengan cara *node* sumber melakukan *broadcast* pesan *request* ke semua tetangganya untuk menemukan rute ke tujuan. Untuk memperkirakan konektivitasnya, setiap *node* yang melakukan penerusan paket ke tujuan akan memperbarui jumlah hop, rata-rata dan jumlah minum tetangga. Setelah paket *request* sampai pada *node* tujuan, maka *node* tujuan akan menentukan rute optimal berdasarkan tingkat konektivitas rute dan akan mengirimkan paket *reply* ke *node* sumber. Meskipun CAR menangani masalah konektivitas, informasi yang dikumpulkan tentang jumlah *node* tidak dapat memastikan konektivitas di segmen jalan individu di sepanjang jalur rute karena konektivitas tergantung pada jumlah *node* dan topologinya (Goudarzi, Asgari dan Al-Raweshidy, 2019).

2.2.4 Pengembangan Radius Optimum pada Metode *Greedy Forwarding*

Pada penelitian Yang et al., *greedy forwarding* yang diterapkan pada GPSR, *node* sumber selalu memilih *node* yang memiliki posisi terdekat dengan *node* tujuan dalam daftar tetangga sebagai hop berikutnya (Yang *et al.*, 2018). Tetapi, sering kali *node* yang dipilih adalah *node* yang berposisi di lingkar terluar dari radius *node* sumber. Posisi dari *node* tetangga sangat mudah berubah ubah karena *node* bergerak dengan mobilitas yang tinggi di lingkungan VANETs. Hal ini dapat menyebabkan kegagalan pengiriman paket dikarenakan node sudah bergerak menjauhi node sumber dan keluar dari radius node sumber tersebut sehingga berdampak terjadinya pengiriman ulang yang menurunkan kinerja jaringan secara keseluruhan.



Gambar 2.6 Pencarian area komunikasi optimum ($\mathcal{Q}$)

Solusi yang diberikan adalah dengan menambahkan 2 parameter pada proses *greedy forwarding*, yaitu durasi komunikasi kumulatif (T) dalam satuan detik dan area komunikasi Optimum ($\mathcal{Q}$). Pada Gambar 2.6 terdapat lingkaran kecil yang merepresentasikan radius dari *node* sumber, yakni *node* S yang memiliki posisi koordinat (x_S, y_S). *Node* S mencoba mengirim paket ke *node* tujuan, yakni *node* D yang memiliki posisi koordinat (x_D, y_D). Pada proses *greedy forwarding*, maka *node* S akan memilih *node* yang memiliki jarak terdekat ke *node* D, yaitu *node* B dengan posisi koordinat (x_B, y_B), maka dapat dihitung jarak antar node:

$$d_{BD} = \sqrt{(x_D - x_B)^2 + (y_D - y_B)^2} \tag{2.2}$$

dengan:

d_{BD} = jarak antara *node* B dengan *node* D

x_D, y_D = titik koordinat *node* D

x_B, y_B = titik koordinat *node* B

$$d_{SB} = \sqrt{(x_S - x_B)^2 + (y_S - y_B)^2} \tag{2.3}$$

dengan:

d_{SD} = jarak antara *node* S dengan *node* D

x_S, y_S = titik koordinat *node* S

x_B, y_B = titik koordinat *node* B

$$d_{max} = d_{BD} + \lambda * d_{SB} \tag{2.4}$$

dengan:

d_{max} = jarak maksimum

d_{BD} = jarak antara *node* B dengan *node* D

d_{SD} = jarak antara *node* B dengan *node* D

$\lambda \in [0, 1]$

Penentuan nilai λ akan mempengaruhi ukuran $\mathcal{Q}$. Ketika λ terlalu besar, $\mathcal{Q}$ akan menjadi lebih besar, maka simpul di dekat S lebih mudah dipilih sebagai hop berikutnya di $\mathcal{Q}$, tetapi jumlah hop untuk menuju *node* D dapat meningkat. Ketika λ terlalu kecil, $\mathcal{Q}$ akan menjadi lebih kecil, maka *node* di dekat *node* D lebih mudah dipilih sebagai lompatan berikutnya di $\mathcal{Q}$, jarak dari *node* S ke *node* yang dipilih bias menghasilkan *delay* yang lebih lama dan stabilitas *link* bisa menjadi lebih buruk, menyebabkan meningkatnya jumlah paket yang hilang. Dengan melakukan beberapa percobaan, persamaan ini memiliki kinerja yang layak dalam *greedy forwarding* ketika λ diatur ke 0,3.

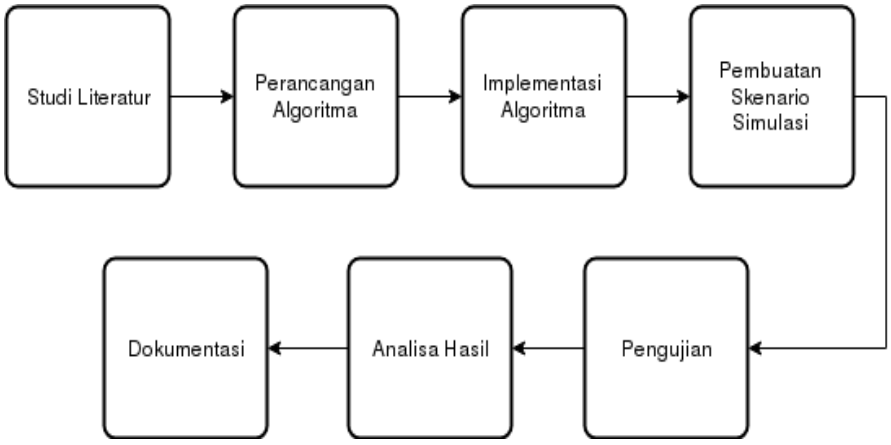
Area yang beririsan dari dua lingkaran, lingkaran *node* D sebagai pusat dan d_{max} sebagai jari-jari, dan lingkaran lainnya dengan *node* S sebagai pusat dan jarak komunikasi maksimum sebagai jari-jari, didefinisikan sebagai area komunikasi optimum. Masing-masing *node* dalam $\mathcal{Q}$ tidak hanya dekat dengan *node* tujuan

tetapi juga dalam jangkauan komunikasi *node* S, dan cocok untuk dipilih sebagai *node* berikutnya dari *node* S.

BAB 3
METODE PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian dibutuhkan agar tujuan yang diharapkan dari penelitian ini dapat tercapai. Langkah awal dari penelitian ini adalah studi literatur seperti yang ditunjukkan pada Gambar 3.1 Studi literatur digunakan untuk mempelajari masalah yang ditemukan dan penelitian saat ini tentang masalah tersebut. Langkah selanjutnya adalah perancangan dan penerapan (implementasi) algoritma. Dalam penelitian ini, algoritma yang dirancang adalah modifikasi pada *greedy forwarding* dengan melakukan pencarian *virtual anchor point* (VAP) terlebih dahulu sebagai representasi dari konsep *overlay network* dan dari VAP akan menjadi salah satu acuan pemilihan *node* selanjutnya selain radius optimal yang dimiliki oleh *node* pengirim. Perihal tentang penerapan, algoritma yang telah dirancang akan diterapkan pada lingkungan VANETs. Tahap pengujian dilakukan dengan simulasi pada lingkungan peta berbentuk *grid* dan menggunakan peta *real,* yaitu peta kota perumahan wisma permai Mulyosari Surabaya yang telah dibuat terlebih dahulu. Hasil dari tahap pengujian kemudian akan dianalisis untuk mendapatkan kesimpulan dari algoritma yang telah dirancang dan dibandingkan dengan algoritma pembanding yang akan ditulis dalam dokumentasi berupa buku tesis.



Gambar 3.1 Tahapan Penelitian

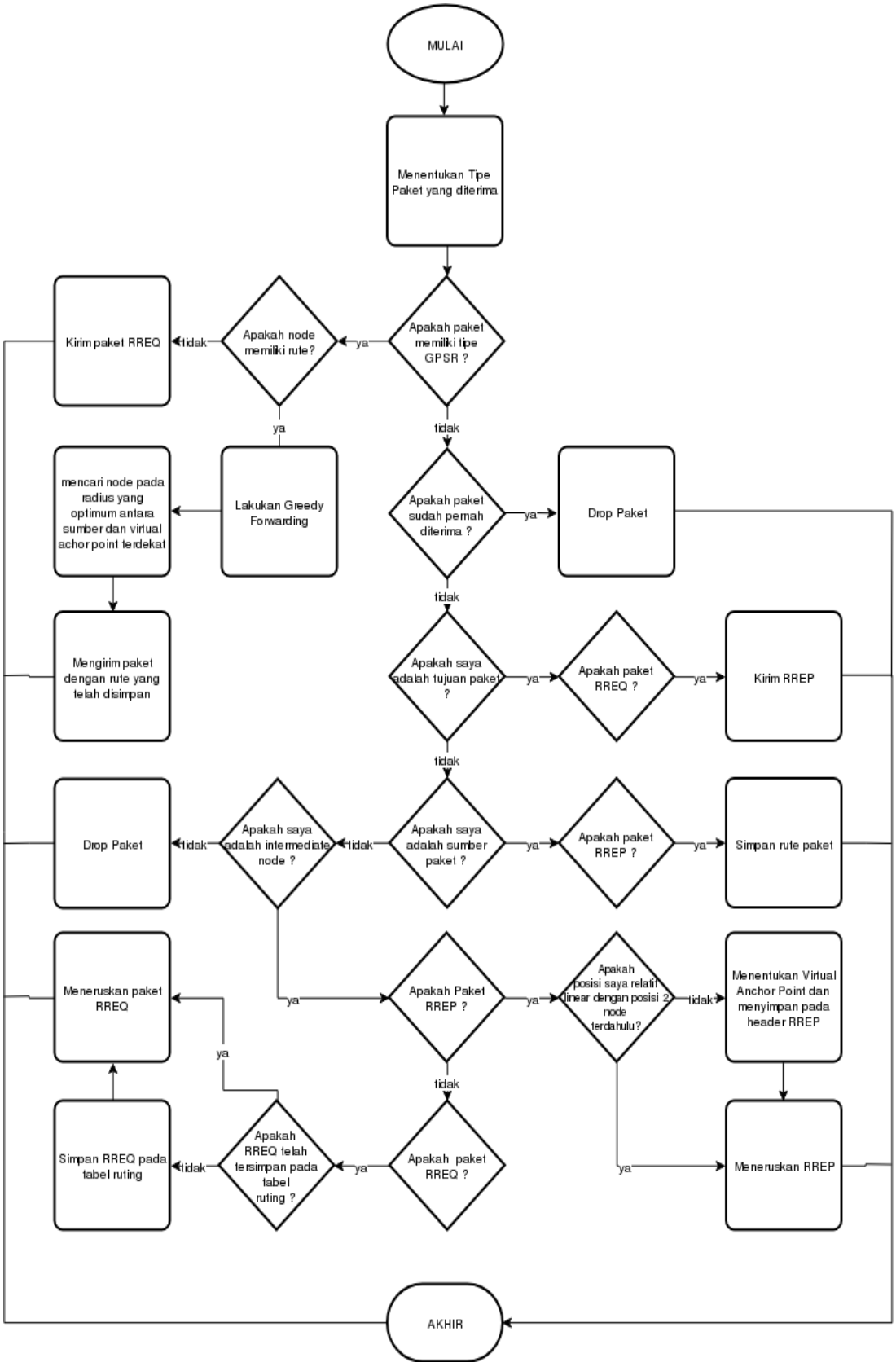
3.2 Hasil Kajian Tentang *GPSR* dan *Overlay Network*

Penelitian diawali dengan melakukan kajian yang berkaitan dengan topik penelitian yang diajukan. Referensi yang digunakan dalam penelitian ini berasal dari jurnal, konferensi dan buku yang berkaitan dengan penerapan konsep *overlay network* pada protokol *routing* dan penelitian terbaru protokol GPSR. Berdasarkan studi literatur yang telah dilakukan, didapatkan informasi sebagai berikut:

1. Konsep *overlay network* telah diterapkan pada beberapa protokol *routing*, yaitu LOUVRE dan ACAR.
2. Penerapan konsep *overlay network* pada protokol *routing* mampu meningkatkan kinerja protokol *routing* tersebut. Hal ini dapat terjadi karena penerusan paket berupa implementasi dari konsep *Preferred Group Broadcast* (PGB) dan pencarian rute berdasarkan tingkat konektivitas dari suatu kelompok *node*.
3. Metode *greedy forwarding* pada protokol GPSR memiliki kelemahan jika menemui keadaan *local maximum*.
4. Pemilihan *node* selanjutnya pada metode *greedy forwarding* memiliki kemungkinan terjadinya *packet loss* karena metode ini memilih *node* yang memiliki jarak terdekat dengan *node* tujuan tanpa memperhatikan kecepatan dan arah *node* yang akan dipilih.
5. Perbaikan kinerja GPSR pada penelitian terbaru menghasilkan protokol baru GPCR dan GyTAR dimana *greedy forwarding* melakukan penentuan penerusan paket pada persimpangan.

3.3 Perancangan Algoritma

Alur perancangan pengiriman data secara keseluruhan dapat dilihat pada Gambar 3.2. Alur tersebut merupakan modifikasi terhadap alur standar pengiriman data pada protokol GPSR. Terdapat 2 perubahan yang dilakukan yaitu adanya proses pencarian *virtual anchor point* (VAP) yang dilakukan sebelum pengiriman paket dan modifikasi *greedy forwarding* dengan acuan radius optimal dan VAP dalam pemilihan *node* selanjutnya. Secara lebih rinci kedua proses tersebut dijelaskan pada sub-sub bab di bawah ini:



Gambar 3.2 Diagram alur pengiriman paket

3.3.1 Perancangan Algoritma Pencarian *Virtual Anchor Point*

Virtual Anchor Point (VAP) adalah sebuah representasi dari konsep *overlay network* dimana berupa titik maya yang didapatkan dari proses pencarian rute yang diinisiasi sebelum pengiriman paket dilakukan. Proses tersebut pada layer aplikasi jaringan. VAP akan digunakan sebagai acuan dalam memilih *node* selanjutnya oleh metode *greedy forwarding*. Pada proses awal, *node* sumber akan mengirimkan paket *request* menuju *node* tujuan dengan konsep *route discovery* milik protokol *routing dynamic source routing* (DSR), yaitu berdasarkan konektivitas *node* pada lingkungan simulasi. Setelah paket *request* sampai pada *node* tujuan, maka *node* tujuan akan mengirimkan paket reply ke *node* sumber, paket ini bertugas membentuk rute VAP, syarat terbentuknya VAP adalah sebagai berikut:

1. Posisi *node* dengan 2 *node* sebelumnya relatif tidak linear.
2. Apabila posisi *node* dengan 2 dalam keadaan ketidaksamaan segitiga maka *virtual anchor point* didapatkan dengan mencari titik tengah antara *node* dengan *node* kedua sebelumnya.

Formula (3.1) merupakan formula ketidaksamaan segitiga. Perhitungan diawali dengan mendapatkan jarak antara *node* dengan dua *node* sebelumnya serta jarak kedua *node* sebelumnya.

$$r \leq p + q \tag{3.1}$$

dengan:

r = jarak antara *node* dengan *node* kedua sebelumnya

p = jarak antara *node* dengan *node* sebelumnya

q = jarak antara kedua *node* sebelumnya.

Formula jarak menggunakan *euclidean distance* dan ditunjukkan pada (3.5). Apabila jarak *node* dengan *node* kedua sebelumnya lebih kecil daripada jumlah jarak *node* dengan *node* sebelumnya dan jarak kedua *node* sebelumnya, maka dapat diidentifikasikan bahwa *node* tersebut mengalami ketidaksamaan segitiga. Setelah itu, maka akan melakukan perhitungan koordinat VAP menggunakan rumus titik tengah antara 2 titik dan ditunjukkan pada (3.3).

$$d(x,y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (3.2)$$

dengan:

X_1 = titik koordinat X *node* pertama

X_2 = titik koordinat X *node* kedua

Y_1 = titik koordinat Y *node* pertama

Y_2 = titik koordinat Y *node* kedua

$$VAP = (\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}) \quad (3.3)$$

dengan:

X_1 = titik koordinat X *node* pertama

X_2 = titik koordinat X *node* kedua

Y_1 = titik koordinat Y *node* pertama

Y_2 = titik koordinat Y *node* kedua

3.3.2 Perancangan Algoritma Pencarian Radius Optimum

Berdasarkan *virtual anchor point* yang didapatkan sebelumnya, maka saat melakukan *greedy forwarding*, *node* tujuan tidak menjadi acuan arah pengiriman paket, namun *node* sumber atau *node* perantara akan memilih *node* yang memiliki jarak terdekat dengan *virtual anchor point* terdekat yang berada pada radius optimal. Radius optimal (d_{max}) didapatkan dengan persamaan sebagai berikut:

$$d_{max} = d_{PVap} + \lambda * d_{SP} \quad (3.4)$$

dengan:

d_{max} = Radius optimal

d_{PVap} = jarak antara *node* perantara dengan *virtual anchor point* terdekat

Λ $\in$ [0, 1]

d_{SP} = jarak antara *node* sumber dengan *node* perantara

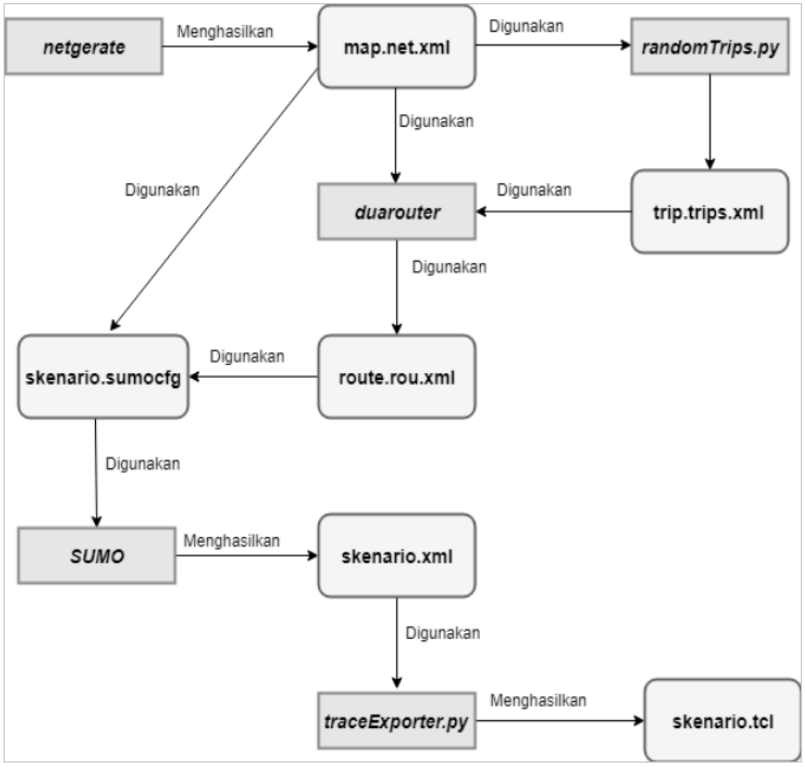
Penghitungan jarak menggunakan persamaan *euclidean distance* (3.3) karena pada lingkungan VANETs penelitian ini tidak memperhitungkan adanya halangan (*obstacle*).

3.4 Perancangan Skenario Mobilitas

Pada penelitian ini skenario mobilitas akan menggunakan 2 bentuk peta, yaitu peta berbentuk *grid* dan peta *real*. Skenario mobilitas dengan varian peta *grid* dan *real* dianggap merepresentasikan lingkungan VANETs. Penjelasan kedua skenario mobilitas adalah sebagai berikut:

3.4.1 Skenario Mobilitas *grid*

Perancangan skenario mobilitas *grid* diawali dengan menentukan luas area simulasi yang dibutuhkan. Dari luas area tersebut ditentukan panjang dan lebar area. Selanjutnya dari panjang dan lebar area, ditentukan banyaknya persimpangan atau perpotongan jalan yang dibutuhkan, sehingga dapat diketahui jumlah petak. Setelah jumlah petak diketahui, dapat menentukan panjang dan lebar setiap petak.



Gambar 3.3 Alur perancangan skenario mobilitas *grid*

Peta grid yang telah ditentukan luas areanya tersebut kemudian akan mengikuti alur perancangan skenario mobilitas grid yang digambarkan pada Gambar 3.3. Langkah pertama adalah membuat peta dengan area yang ditentukan sebelumnya menggunakan *tools Simulation of Urban Mobility* (SUMO) yaitu *netgenerate*. Selain titik persimpangan dan panjang tiap petak *grid*, dibutuhkan juga pengaturan jumlah *node* dan kecepatan maksimal *node* dalam pembuatan peta *grid*. Peta *grid* yang dihasilkan oleh *netgenerate* akan memiliki ekstensi .net.xml. Peta grid ini kemudian digunakan untuk membuat pergerakan node dengan *tools* SUMO yaitu *RandomTrips* dan *duarouter*.

Skenario mobilitas *grid* dihasilkan dengan menggabungkan file peta *grid* dan file pergerakan *node* yang telah dibuat sebelumnya, akan menghasilkan file dengan ekstensi .xml. Selanjutnya, untuk dapat menerapkannya pada NS-2.35 file skenario mobilitas *grid* yang memiliki ekstensi .xml dikonversi ke dalam bentuk *file* .tcl. Konversi ini dilakukan menggunakan tools *traceExporter*. Hasilnya berupa file yang berisi mobilitas dari setiap *node* (mobility.tcl) dan informasi *lifetime* dari setiap *node* (activity.tcl).

3.4.2 Skenario Mobilitas *real*

Perancangan skenario mobilitas *real* diawali dengan memilih area yang akan dijadikan simulasi. Pada penelitian ini, digunakan peta dari *OpenStreetMap* untuk mengambil area yang dijadikan model simulasi. Area yang diambil adalah perumahan wisma permai daerah Mulyosari kota Surabaya. Setelah memilih area, unduh dengan menggunakan fitur *export* dari *OpenStreetMap*. Peta hasil *export* dari *OpenStreetMap* ini memiliki ekstensi .osm.

Setelah mendapatkan peta area yang dijadikan simulasi, peta tersebut dikonversi ke dalam bentuk *file* dengan ekstensi .net.xml menggunakan tools *SUMO* yaitu *netconvert*. Tahap berikutnya memiliki tahapan yang sama seperti tahapan ketika merancang skenario mobilitas grid yang ditunjukkan pada Gambar 3.3, yaitu membuat pergerakan *node* menggunakan *RandomTrips* dan *duarouter*. Kemudian gabungkan *file* peta *real* yang sudah dikonversi ke dalam *file* dengan ekstensi .net.xml dan file pergerakan *node* yang sudah di buat sebelumnya. Hasil dari

penggabungan tersebut merupakan *file* skenario yang memiliki ekstensi .xml. *File* yang dihasilkan tersebut dikonversi ke dalam bentuk *file* .tcl agar dapat diterapkan pada NS-2.35. Alur pembuatan skenario *real* hampir sama dengan alur pembuatan skenario *grid*. Namun peta map.net.xml dihasilkan dari peta OSM yang telah dikonversi. Proses selanjutnya sama dengan pembuatan skenario *grid*.

3.5 Implementasi

Simulasi *VANETs* pada NS-2.35 dilakukan dengan menggabungkan file skenario yang telah dibuat menggunakan *Simulation of Urban Mobility* (*SUMO*) dan skrip *file Tool Command Language* (*TCL*) yang berisikan konfigurasi lingkungan simulasi. Parameter dalam simulasi ditunjukkan pada Tabel 3.1.

Tabel 3.1 Parameter Simulasi

Parameter	Spesifikasi
Protokol *Routing*	GPSR tradisional, GPSR modifikasi
MAC Protocol	IEEE 802.11p
Area Simulasi	600 m x 800 m skenario *grid* 700 m x 700 m skenario *real*
Jumlah Kendaraan	50, 100, 150
Radius Transmisi	200m
Kecepatan Maksimal	55 Km/jam
Node pengirim dan *node* tujuan	Posisi statis
Agen	*Constant Bit Rate* (*CBR*)
Ukuran Paket	512 Bytes
Packet Rate	2 kb/s

3.6 Pengujian

Untuk melakukan pengujian dibutuhkan sebuah skrip *file Tool Command Language* (*TCL*) yang berisi deskripsi dari lingkungan simulasi untuk simulasi *VANETs* pada perangkat lunak NS-2.35. File tersebut berisikan pengaturan untuk setiap *node* dan beberapa *Event* yang perlu diatur agar berjalan. Pada setiap skenario

lingkungan simulasi, akan di *generate* skenario mobilitas sebanyak 10 kali. Pengujian menggunakan protokol GPSR tradisional dan GPSR yang telah dimodifikasi.

3.7 Analisa Hasil

Analisa hasil dilakukan dengan mengolah hasil *trace file* simulasi menggunakan *script awk*. Dimana matriks yang digunakan adalah sebagai berikut:

3.7.1 Analisa *overlay network*

Pada analisa ini akan menjelaskan letak *virtual anchor point* (VAP) yang didapatkan dari proses *route discovery* di layer aplikasi jaringan. Analisa *overlay network* meliputi *plotting* VAP pada skenario *grid* dan *real*, faktor yang mempengaruhi letak VAP dan perbandingan letak VAP di skenario dengan hasil PDR yang berbeda.

3.7.2 *Packet Delivery Ratio*

Packet delivery ratio (*PDR*) merupakan perbandingan antara jumlah paket yang diterima dengan jumlah paket yang dikirimkan. Persamaan (3.5) merupakan perhitungan *PDR* dimana nilai *PDR* dinyatakan dalam bentuk persentase.

$$PDR = \frac{received}{sent} * 100 \tag{3.5}$$

dengan:

PDR = *Packet Delivery Ratio*

$sent$ = $jumlah\ paket\ yang\ dikirim$

$received$ = $jumlah\ paket\ yang\ terima$

Packet delivery ratio dapat menunjukkan keberhasilan paket yang dikirimkan. Semakin tinggi *packet delivery ratio*, artinya semakin berhasil pengiriman paket yang dilakukan.

3.7.3 *Average End to End Delay*

Rata-rata *end-to-end delay* merupakan rata-rata dari *delay* atau waktu yang dibutuhkan tiap paket untuk sampai ke *node* tujuan dalam satuan detik. *Delay* tiap paket didapatkan dari rentang waktu antara *node* asal mengirimkan paket dan *node* tujuan menerima paket. Dari *delay* tiap paket tersebut semua dijumlahkan dan dibagi dengan jumlah paket yang berhasil diterima, maka akan didapatkan rata-rata *end-to-end delay*, yang dapat dihitung dengan persamaan (3.6).

$$E2E = \frac{\sum_{m=1}^{recvnum} CBRRecvTime - CBRSentTime}{recvnum} \tag{3.6}$$

dengan:

$E2E$ = *End-to-End Delay*

$CBRRecvTime$ = Waktu *node* asal mengirimkan paket

$CBRSentTime$ = Waktu *node* tujuan menerima paket

$recvnum$ = Jumlah paket yang berhasil diterima

Besar *end to end delay* dapat menunjukkan tingkat konektivitas dan efisiensi pada sebuah rute yang digunakan. Semakin besar *end to end delay* maka rute yang dipilih merupakan rute yang tidak efisien dan memiliki kemungkinan memiliki konektivitas yang buruk atau terlalu padat.

3.7.4 *Routing Overhead*

Routing Overhead (*RO*) adalah jumlah paket kontrol *routing* yang ditransmisikan per data paket yang terkirim ke tujuan selama simulasi terjadi. *RO* dihitung berdasarkan jumlah paket *routing* yang ditransmisikan. Baris yang mengandung *routing overhead* pada *trace file* ditandai dengan paket yang memiliki tipe *send* (s) / *forward* (f) dan terdapat *header* paket dari protokol. Perhitungan *routing overhead* dapat dilihat pada persamaan (3.4)

$$RO = sentRTR + forwardedRTR \tag{3.7}$$

dengan:

RO = *Routing Overhead*

$sentRTR$ = Jumlah paket *routing* yang dikirim

$forwardedRTR$ = Jumlah paket *routing* yang diteruskan

Analisa menggunakan *RO* mempunyai fungsi yang sama dengan *Average End to End Delay*. Besaran RO menunjukkan tingkat konektivitas dan efisiensi sebuah rute. Semakin besar *RO* maka dapat dipastikan bahwa rute tersebut tidak efisien dan terdapat kemungkinan menggunakan node dengan konektivitas yang padat

3.8 Dokumentasi dan Jadwal Penelitian

Proses dokumentasi digunakan untuk penulisan laporan hasil penelitian yang dilakukan. Setiap tahapan dalam proses penelitian juga disertakan dalam laporan yang ditulis. Kegiatan dokumentasi ini akan dicantumkan pada jadwal penelitian yang direncanakan mulai bulan Juli 2019 hingga Nopember 2019 yang secara rinci terlihat pada Tabel 3.2.

Tabel 3.2 Jadwal kegiatan penelitian

No.	Kegiatan	Bulan																			
		Juli				Agustus				September				Oktober				Nopember			
1.	Studi Literatur																				
2.	Analisa dan Perancangan																				
3.	Implementasi																				
4.	Pengujian dan Evaluasi																				
5.	Penyusunan Buku Tesis																				

(Halaman ini sengaja dikosongkan)

BAB 4
HASIL DAN PEMBAHASAN

4.1 Implementasi Sistem

Spesifikasi perangkat keras yang digunakan untuk penelitian ini dapat dilihat pada Tabel 4.1 Lingkungan Simulasi.

Tabel 4.1 Lingkungan Simulasi

Komponen	Spesifikasi
Sistem Operasi	LXC Ubuntu 16.04 64 bit gcc 5.4
CPU	Intel Core i7-5500U @ 2,4 GHz x 8
Memori	8 GB PC3-12800 DDR3
Penyimpanan	500 GB

Sedangkan perangkat lunak yang digunakan dalam penelitian ini adalah:

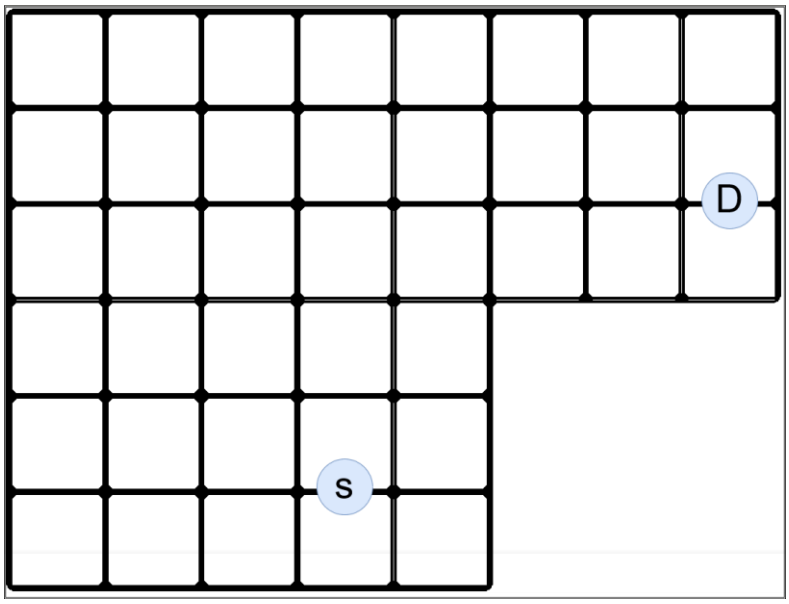
1. *Simulation of Urban Mobility* (SUMO) versi 1.4.0 untuk pembuatan skenario mobilitas baik peta grid maupun peta real pada VANETs
2. *Java OpenStreetMap Editor* (JOSM) versi 15553 untuk penyuntingan peta kota yang didapatkan dari OpenStreetMap.
3. Network Simulator (NS) versi 2.35 dengan gcc versi 5.4 untuk melakukan simulasi routing protokol pada peta yang telah dibuat oleh SUMO.
4. Visual Studio Code versi 1.40 sebagai editor kode dalam proses perancangan algoritma.

4.2 Analisa Hasil

Analisa hasil penelitian ini dibagi menjadi dua, yaitu hasil simulasi menggunakan skenario mobilitas *grid* dan hasil simulasi menggunakan skenario mobilitas *real*. Setiap hasil simulasi mencakup pembahasan tentang analisa implementasi *overlay network* serta *routing* metrik yang dibandingkan antara lain PDR (*Packet Delivery Ratio*), RO (*Routing Overhead*) dan E2E (*end to end delay*).

4.2.1 Hasil uji coba skenario *grid*

Uji coba skenario *grid* dilakukan sebanyak 10 kali dengan skenario mobilitas *random*. Peta *grid* yang digunakan memiliki luas area 600 meter x 800 meter dengan memiliki area *void* sebesar 300 meter x 300 meter. Jumlah *node* yang digunakan pada uji coba sebanyak 50 *node* sebagai representasi area dengan jumlah kepadatan yang jarang, 100 *node* sebagai representasi area dengan jumlah kepadatan yang sedang dan 150 *node* sebagai representasi area dengan jumlah kepadatan yang tinggi. *Node* sumber dan *node* pengirim diletakan secara statis dan tidak bergerak. Setiap *node* memiliki kecepatan maksimum 55 Km/jam. Rancangan simulasi peta *grid* dapat dilihat pada Gambar 4.1 Rancangan peta *grid*.



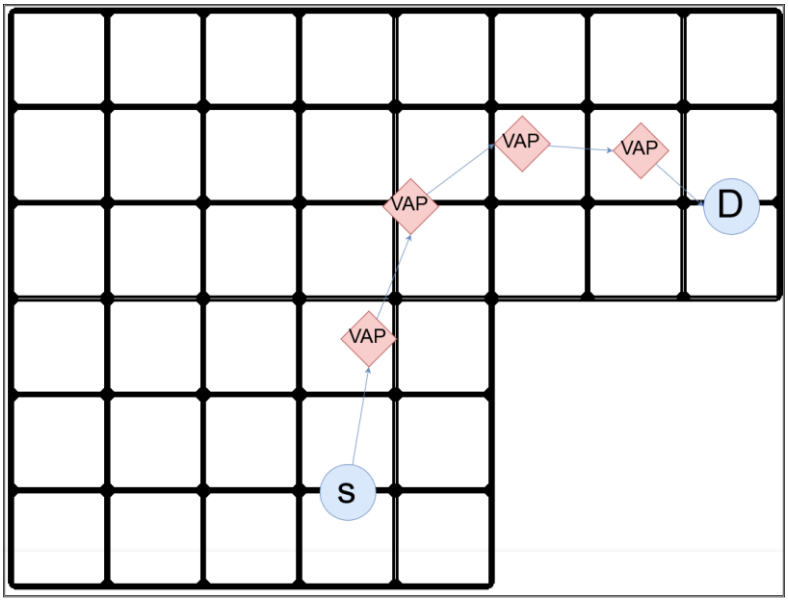
Gambar 4.1 Rancangan peta *grid*

4.2.1.1 Analisa implementasi *overlay network*

Protokol GPSR modifikasi melakukan *pencarian virtual anchor point (VAP)* sebelum melakukan pengiriman paket, VAP tersebut merupakan representasi dari konsep *overlay network*. VAP berupa koordinat dan akan digunakan sebagai acuan pengiriman paket menggantikan acuan posisi *node* destinasi, proses ini dinamakan *preferred group broadcast* (PGB). Pemilihan VAP sebagai acuan akan berganti mengikuti jumlah hop yang telah dilewati. Ketika semua VAP telah digunakan, maka protokol akan memilih posisi *node* destinasi menjadi titik acuan.

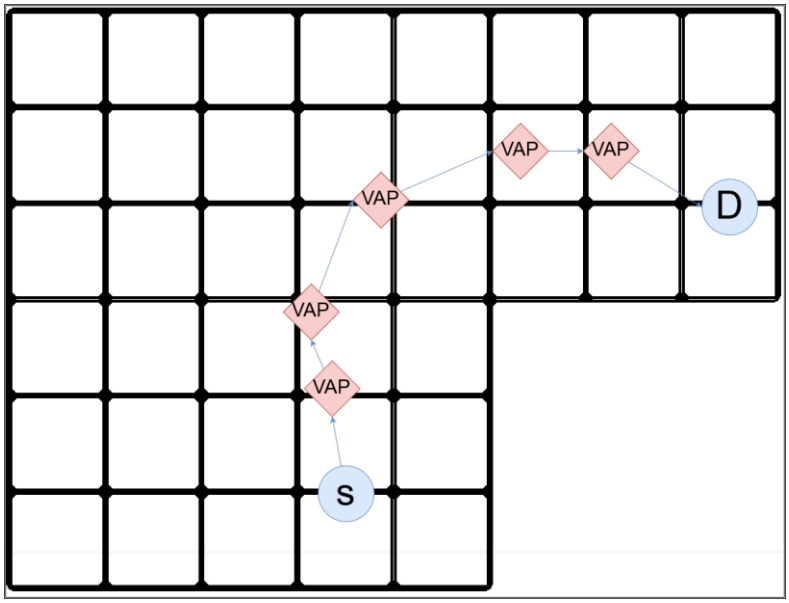
Pencarian VAP menggunakan metode *route discovery* milik protokol *dynamic source routing* (DSR) pada layer aplikasi jaringan. Proses ini berjalan sebelum melakukan pengiriman paket pertama, yaitu pada detik 10, setelah proses broadcast *beacon message* selesai dilakukan. Pada skenario *grid* 50 *node*, VAP didapatkan dalam rata-rata waktu 0.12 detik, sedangkan pada skenario *grid* 100 *node*, VAP didapatkan dalam rata-rata waktu 0.16 detik dan pada skenario *grid* 150 *node*, VAP didapatkan dalam rata-rata waktu 0.21 detik. Hal ini dapat disimpulkan bahwa kenaikan waktu yang dibutuhkan dalam pencarian VAP berbanding lurus dengan kenaikan jumlah *node* pada area simulasi.

PGB pada GPSR modifikasi akan ditambahkan dengan pencarian area optimum *node* pengirim, dimana VAP menjadi salah satu faktor dalam penghitungan area optimum. Hasil dari pencarian PGB akan menghasilkan *node* selanjutnya yang diasumsikan tidak mengalami keadaan *local maximum*. Selain kebutuhan dalam penghitungan area optimum, VAP juga mampu mendapatkan rute yang berubah rubah menyesuaikan pergerakan *node* yang memiliki mobilitas tinggi. Rute yang dihasilkan akan memiliki bentuk relatif sama dengan rute sebelumnya namun dengan *node* terpilih yang berbeda.



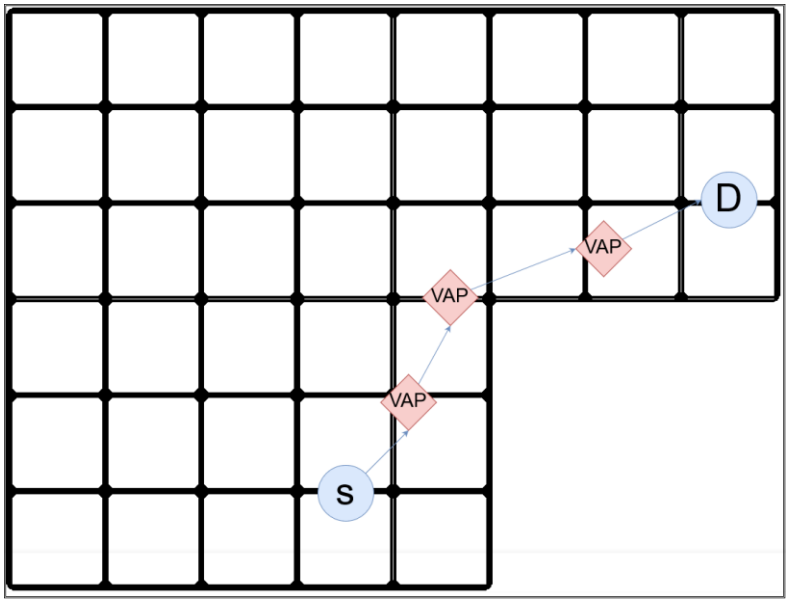
Gambar 4.2 Hasil pencarian VAP pada skenario *grid* 50 *node* PDR rendah

Gambar 4.2 menunjukkan letak VAP pada skenario *grid* 50 *node* yang menghasilkan nilai rata-rata *packet delivery ratio* (PDR) yang rendah. Terdapat 4 VAP yang terletak secara tidak beraturan berdasarkan rute yang didapatkan dari proses *route discovery* sebelumnya. Jumlah dan letak VAP dipengaruhi faktor posisi dan kepadatan *node* pada area simulasi. Skenario ini sukses mengirimkan 5 paket dari 181 paket yang dikirimkan. Nilai rata-rata PDR yang rendah dikarenakan posisi VAP tidak dapat menjawab permasalahan pergerakan *node* yang tinggi pada area simulasi.



Gambar 4.3 Hasil pencarian VAP pada skenario *grid* 50 *node* PDR sedang

Gambar 4.3 menunjukkan letak VAP pada skenario *grid* 50 *node* yang menghasilkan nilai rata-rata *packet delivery ratio* (PDR) yang sedang. Terdapat 5 VAP yang terletak secara tidak beraturan berdasarkan rute yang didapatkan dari proses *route discovery* sebelumnya. Skenario ini sukses mengirimkan 72 paket dari 185 paket yang dikirimkan. Jika dibandingkan dengan Gambar 4.2, jumlah VAP mengalami kenaikan.

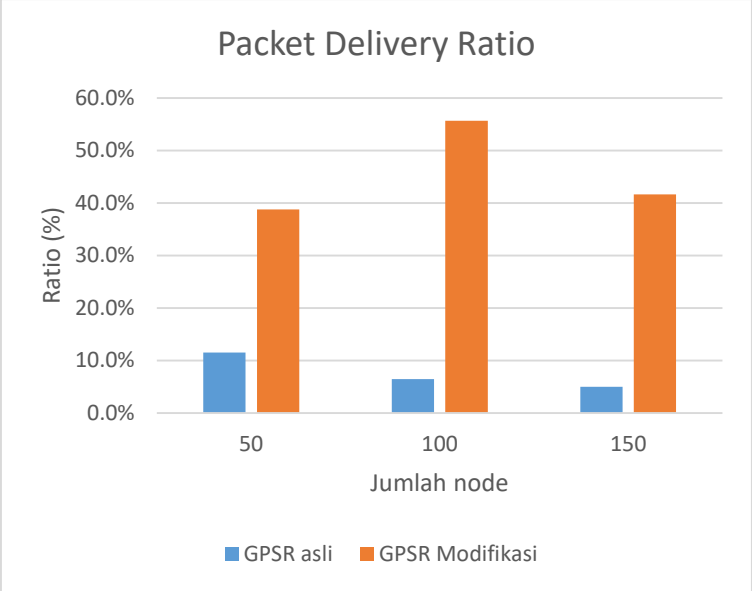


Gambar 4.4 Hasil pencarian VAP pada skenario *grid* 50 *node* PDR tinggi

Sedangkan pada Gambar 4.4 menunjukkan letak VAP pada skenario *grid* 50 *node* yang menghasilkan nilai rata-rata *packet delivery ratio* (PDR) yang tinggi. Terdapat 3 VAP yang terletak secara tidak beraturan berdasarkan rute yang didapatkan dari proses *route discovery* sebelumnya. Skenario ini sukses mengirimkan 136 paket dari 185 paket yang dikirimkan. Bila dibandingkan dengan Gambar 4.2 dan Gambar 4.3 jumlah VAP tidak berpengaruh dengan nilai PDR. Berdasarkan analisa diatas, dapat ditarik kesimpulan bahwa jumlah VAP tidak mempengaruhi nilai rata-rata PDR pada skenario *grid* GPSR modifikasi dikarenakan pergerakan *node* yang cepat dan kepadatan *node* yang berbeda di setiap skenario yang dijalankan.

Apabila dibandingkan dengan GPSR tradisional, maka VAP menyebabkan kenaikan nilai rata-rata PDR yang signifikan, kenaikan nilai rata-rata *end to end delay* yang tinggi serta mengalami sedikit kenaikan *routing overhead* (RO). Kenaikan *end to end delay* yang tinggi disebabkan karena GPSR modifikasi memilih rute mengikuti titik VAP dengan hop yang lebih besar daripada GPSR tradisional. Pada nilai RO mengalami kenaikan dikarenakan GPSR modifikasi melakukan proses route discovery, dimana pada GPSR tradisional tidak ada.

4.2.1.2 Packet Delivery Ratio (PDR)



Gambar 4.5 Grafik PDR skenario *grid*

Gambar 4.5 merupakan grafik perbandingan PDR antara protokol GPSR tradisional dengan protokol GPSR modifikasi. Hasil perhitungan rata-rata dan selisih nilai PDR dari skenario *grid* dengan penambahan jumlah node dapat dilihat pada Tabel 4.2.

Tabel 4.2 Hasil perhitungan rata-rata PDR (%) skenario grid

Jumlah Node	PDR GPSR		selisih
	tradisional	modifikasi	
50	11.6	38.8	27.2
100	6.5	55.7	49.2
150	5.0	41.6	36.6

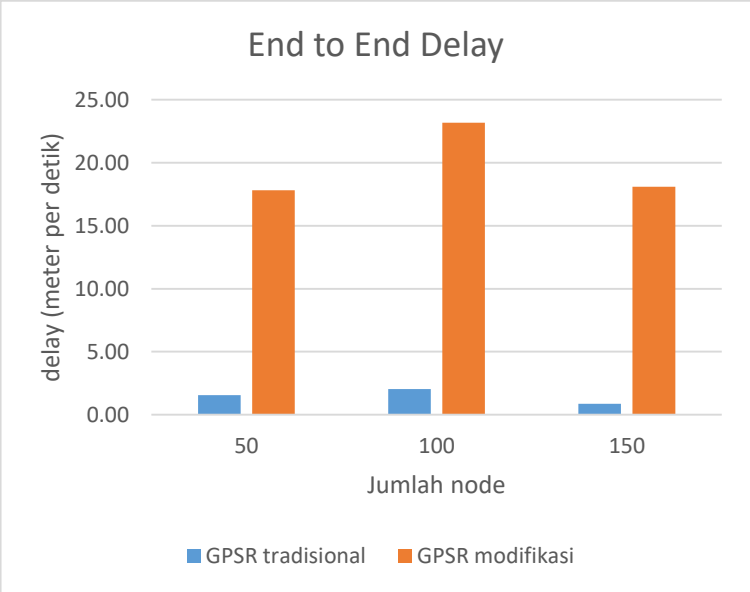
Berdasarkan Gambar 4.5 dapat dilihat bahwa PDR protokol GPSR modifikasi memiliki performa yang lebih baik dibandingkan dengan protokol GPSR tradisional dengan peningkatan performa yang signifikan. Hal ini disebabkan adanya area *void* yang tidak terdapat *node*. Walaupun jarak transmisi tiap node adalah 200 meter, namun metode GPSR tradisional yang menerapkan posisi tujuan menjadi acuan dalam pemilihan *node* selanjutnya menyebabkan seringnya paket

yang gagal terkirim. Kegagalan paket terkirim pada GPSR tradisional disebabkan karena *node* yang bertugas meneruskan paket mengalami kondisi *local maximum*. GPSR modifikasi dapat menghindari *node* dengan keadaan *local maximum* yang memiliki posisi di dekat area *void* dalam pemilihan *node* selanjutnya.

Tabel 4.2 menunjukkan bahwa GPSR tradisional dan GPSR modifikasi memiliki selisih PDR yang tinggi. Pada lingkungan dengan kepadatan yang rendah atau 50 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata PDR hingga 27.2 atau naik 234% dibandingkan dengan nilai rata-rata PDR GPSR tradisional. Sedangkan pada lingkungan dengan kepadatan yang sedang atau 100 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata PDR hingga 49.2 atau naik 757% dibandingkan dengan nilai rata-rata PDR GPSR tradisional dan pada lingkungan dengan kepadatan yang tinggi atau 150 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata PDR hingga 36.6 atau naik 732% dibandingkan dengan nilai rata-rata PDR GPSR tradisional.

Dari analisa tersebut, maka dapat disimpulkan bahwa GPSR modifikasi memiliki nilai PDR yang lebih tinggi dari pada GPSR tradisional baik di lingkungan dengan kepadatan rendah, kepadatan sedang ataupun kepadatan tinggi.

4.2.1.3 Average End to End Delay



Gambar 4.6 Grafik rata-rata *end to end delay* skenario *grid*

Gambar 4.6 menunjukkan grafik perbandingan rata-rata *end to end delay* pada protokol GPSR tradisional dan GPSR modifikasi dari skenario *grid*, dimana protokol GPSR modifikasi memiliki *delay* yang lebih tinggi. Hasil perhitungan rata-rata dan selisih nilai *end to end delay* dari skenario *grid* dengan penambahan jumlah node dapat dilihat pada Tabel 4.3.

Tabel 4.3 Hasil perhitungan rata-rata *end to end delay* (meter per detik) skenario *grid*

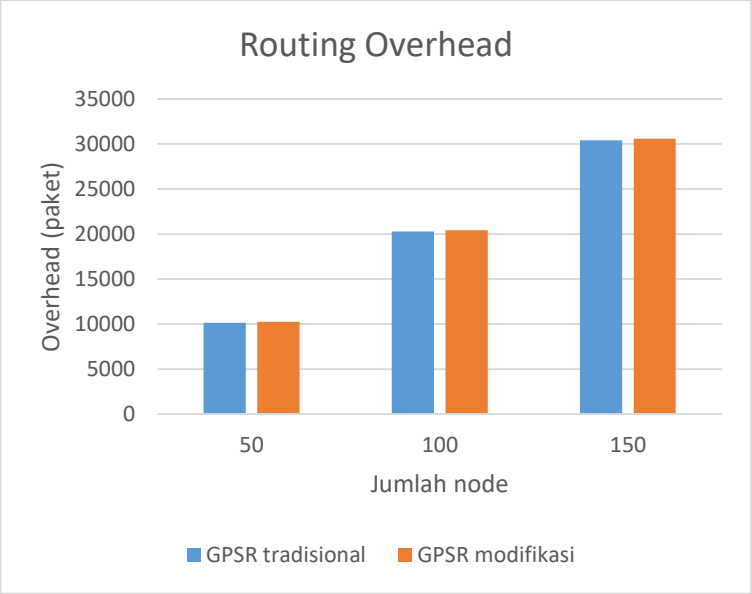
Jumlah Node	End to End Delay GPSR		selisih
	tradisional	modifikasi	
50	1.55	17.82	16.27
100	2.03	23.17	21.14
150	0.88	18.08	17.20

Tabel 4.3 menunjukkan bahwa GPSR tradisional dan GPSR modifikasi memiliki selisih *end to end delay* yang tinggi. Pada lingkungan dengan kepadatan yang rendah atau 50 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata *end to end delay* hingga 16.27 atau naik 1049% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional. Sedangkan pada lingkungan dengan kepadatan yang sedang atau 100 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata *end to end delay* hingga 21.14 atau naik 1041% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional dan pada lingkungan dengan kepadatan yang tinggi atau 150 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata *end to end delay* hingga 17.20 meter per detik atau naik 1954% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional.

Dari analisa tersebut, maka dapat disimpulkan bahwa GPSR modifikasi memiliki nilai *end to end delay* yang lebih tinggi daripada GPSR modifikasi di semua lingkungan dengan kepadatan rendah, sedang ataupun tinggi. Perbedaan yang signifikan ini disebabkan karena pada GPSR tradisional memiliki jumlah paket terkirim cukup rendah, semakin tinggi jumlah *node* maka nilai PDR akan semakin rendah dan nilai *end to end delay* semakin kecil. Pernyataan tersebut berbanding terbalik dengan GPSR modifikasi dimana protokol berhasil

mengirimkan paket yang relatif tinggi dibandingkan GPSR tradisional, sehingga semakin tinggi nilai jumlah *node* maka nilai PDR dan *end to end delay* semakin tinggi.

4.2.1.4 Routing Overhead (RO)



Gambar 4.7 Grafik rata-rata RO skenario *grid*

Gambar 4.7 menunjukkan grafik perhitungan rata-rata *routing* overhead pada skenario *grid* dimana nilai RO linear naik mengikuti naiknya jumlah *node* pada simulasi baik pada GPSR tradisional maupun GPSR modifikasi. Hal ini disebabkan pada GPSR tradisional maupun GPSR modifikasi tidak adanya metode perbaikan rute yang gagal sehingga tidak berdampak pada kenaikan jumlah RO.

Tabel 4.4 Hasil perhitungan rata-rata RO (paket) skenario *grid*

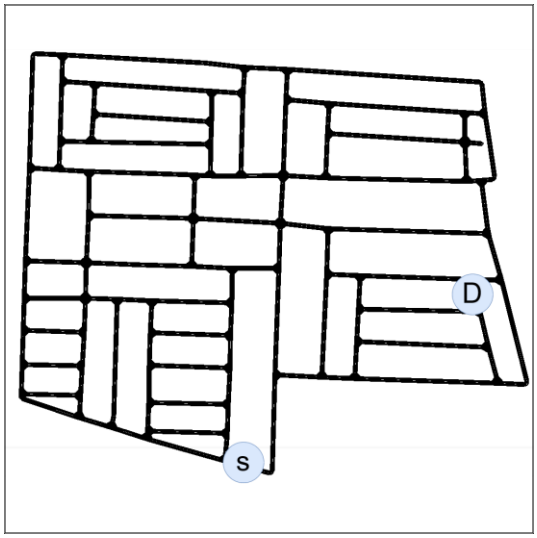
Jumlah Node	RO GPSR		selisih
	tradisional	modifikasi	
50	10154	10232	78
100	20273	20398	125
150	30395	30554	159

Tabel 4.4 menunjukkan bahwa GPSR tradisional dan GPSR modifikasi memiliki selisih RO yang kecil. Pada lingkungan dengan kepadatan yang rendah atau 50 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata RO hingga 78 atau naik 0.7% dibandingkan dengan nilai rata-rata RO GPSR tradisional. Sedangkan pada lingkungan dengan kepadatan yang sedang atau 100 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata RO hingga 125 atau naik 0.6% dibandingkan dengan nilai rata-rata RO GPSR tradisional dan pada lingkungan dengan kepadatan yang tinggi atau 150 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata RO hingga 159 atau naik 0.5% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional.

Dari analisa tersebut, maka dapat disimpulkan bahwa GPSR modifikasi mengalami kenaikan nilai RO yang tidak signifikan dibandingkan dengan GPSR modifikasi di semua lingkungan dengan kepadatan rendah, sedang ataupun tinggi. Hal ini disebabkan GPSR modifikasi memiliki metode pencarian *Virtual Anchor Point* (VAP) sebelum melakukan pengiriman paket pertama. Metode tersebut membutuhkan paket *Route Request* (RREQ) dan *Route Reply* (RREP). Kedua tipe paket tersebut termasuk dalam perhitungan RO.

4.2.2 Hasil uji coba skenario *real*

Uji coba skenario *real* dilakukan sebanyak 10 kali dengan skenario mobilitas *random*. Peta *real* yang digunakan adalah peta perumahan wisma permai daerah Mulyosari Surabaya yang memiliki luas area 700 meter x 700 meter dengan memiliki area *void* yaitu bagian pojok kanan bawah. Jumlah *node* yang digunakan pada uji coba sebanyak 50 *node* sebagai representasi area dengan jumlah kepadatan yang jarang, 100 *node* sebagai representasi area dengan jumlah kepadatan yang sedang dan 150 *node* sebagai representasi area dengan jumlah kepadatan yang tinggi. *Node* sumber dan *node* pengirim diletakan secara statis dan tidak bergerak. Setiap *node* memiliki kecepatan maksimum 55 Km/jam. Rancangan simulasi peta *real* dapat dilihat pada Gambar 4.8 Rancangan peta *real*.



Gambar 4.8 Rancangan peta *real*

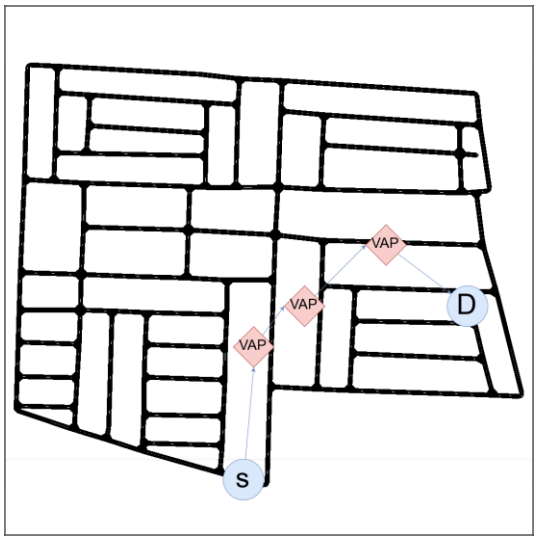
4.2.2.1 Analisa *overlay network*

Overlay network pada skenario *real* memiliki mekanisme yang sama dengan skenario *grid*, namun skenario *real* dengan bentuk jalan yang lebih variatif pada skenario *grid*. Hal ini dapat mempengaruhi pencarian VAP dikarenakan persebaran posisi *node* lebih variatif mengikuti jalan pada peta.

Pencarian VAP menggunakan metode *route discovery* milik protokol *dynamic source routing* (DSR) pada layer aplikasi jaringan. Proses ini berjalan sebelum melakukan pengiriman paket pertama, yaitu pada detik 10, setelah proses broadcast *beacon message* selesai dilakukan. Pada skenario *grid* 50 *node*, VAP didapatkan dalam rata-rata waktu 0.14 detik, sedangkan pada skenario *grid* 100 *node*, VAP didapatkan dalam rata-rata waktu 0.18 detik dan pada skenario *grid* 150 *node*, VAP didapatkan dalam rata-rata waktu 0.23 detik. Hal ini dapat disimpulkan bahwa kenaikan waktu yang dibutuhkan dalam pencarian VAP berbanding lurus dengan kenaikan jumlah *node* pada area simulasi.

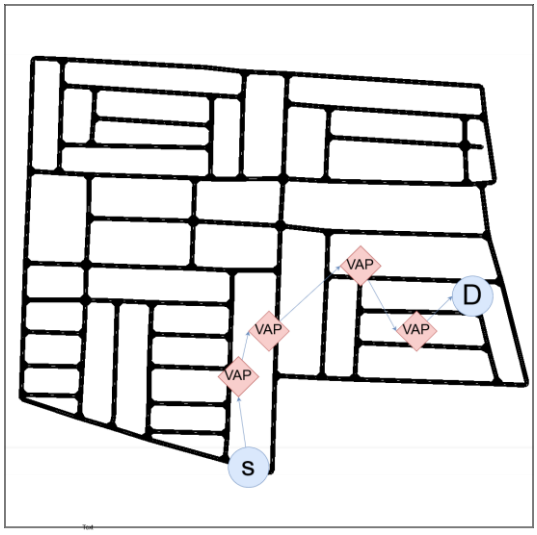
GPSR modifikasi akan menggunakan metode *Preferred Group Broadcast* (PGB) dengan modifikasi pencarian area optimum *node* pengirim. VAP menjadi salah satu faktor dalam penghitungan area optimum. Hasil dari pencarian PGB akan menghasilkan *node* selanjutnya yang diasumsikan tidak mengalami keadaan *local maximum*. Selain kebutuhan dalam penghitungan area optimum, VAP juga mampu

mendapatkan rute yang berubah rubah menyesuaikan pergerakan *node* yang memiliki mobilitas tinggi. Rute yang dihasilkan akan memiliki bentuk relatif sama dengan rute sebelumnya namun dengan *node* terpilih yang berbeda.



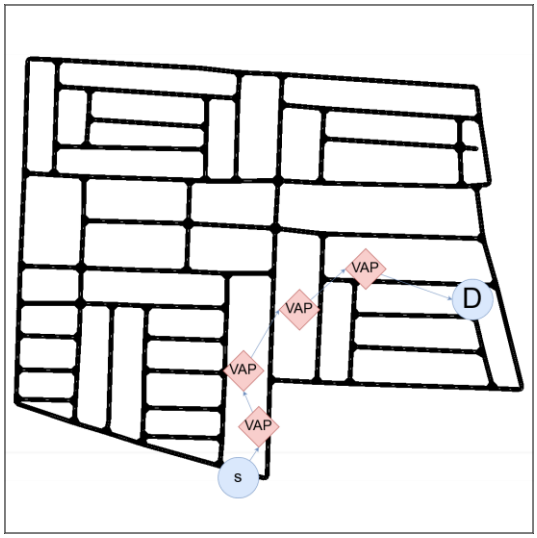
Gambar 4.9 Hasil pencarian VAP pada skenario *real* 150 *node* PDR rendah

Gambar 4.9 menunjukkan letak VAP pada skenario *real* 150 *node* yang menghasilkan nilai rata-rata *packet delivery ratio* (PDR) yang rendah. Terdapat 3 VAP yang terletak secara tidak beraturan berdasarkan rute yang didapatkan dari proses *route discovery* sebelumnya. Skenario ini sukses mengirimkan 1 paket dari 185 paket yang dikirimkan.



Gambar 4.10 Hasil pencarian VAP pada skenario *real* 150 *node* PDR sedang

Gambar 4.10 menunjukkan letak VAP pada skenario *real* 150 *node* yang menghasilkan nilai rata-rata *packet delivery ratio* (PDR) yang sedang. Terdapat 4 VAP yang terletak secara tidak beraturan berdasarkan rute yang didapatkan dari proses *route discovery* sebelumnya. Skenario ini sukses mengirimkan 104 paket dari 184 paket yang dikirimkan.



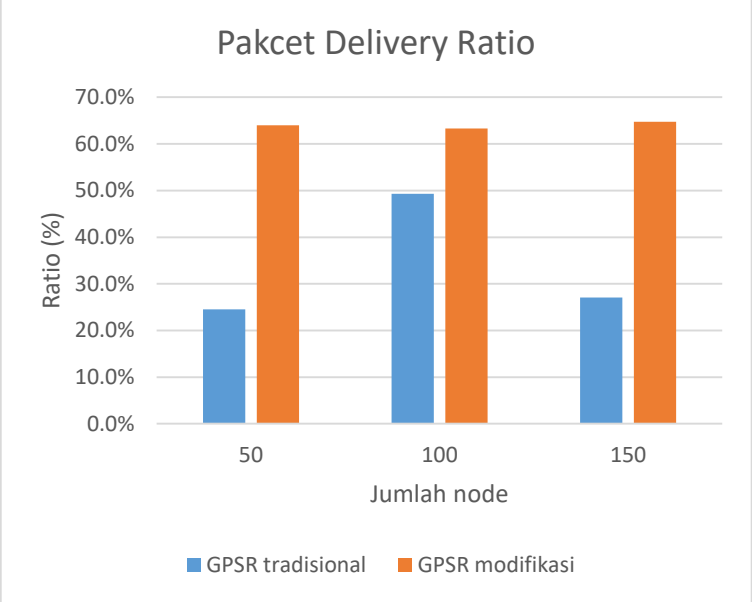
Gambar 4.11 Hasil pencarian VAP pada skenario *real* 150 *node* PDR tinggi

Sedangkan pada Gambar 4.11 menunjukkan letak VAP pada skenario *real* 150 *node* yang menghasilkan nilai rata-rata *packet delivery ratio* (PDR) yang tinggi. Terdapat 4 VAP yang terletak secara tidak beraturan berdasarkan rute yang didapatkan dari proses *route discovery* sebelumnya. Skenario ini sukses mengirimkan 189 paket dari 190 paket yang dikirimkan. Bila dibandingkan dengan Gambar 4.9 dan Gambar 4.10 jumlah VAP tidak berpengaruh dengan nilai PDR. Berdasarkan analisa diatas, dapat ditarik kesimpulan bahwa jumlah VAP tidak mempengaruhi nilai rata-rata PDR pada skenario *grid* dikarenakan pergerakan *node* yang cepat dan kepadatan *node* yang berbeda di setiap skenario yang dijalankan.

Apabila dibandingkan dengan GPSR tradisional, maka VAP menyebabkan kenaikan nilai rata-rata PDR yang signifikan, kenaikan nilai rata-rata *end to end delay* yang tinggi serta mengalami sedikit kenaikan *routing overhead* (RO). Kenaikan *end to end delay* yang tinggi disebabkan karena GPSR modifikasi memilih rute mengikuti titik VAP dengan hop yang lebih besar daripada GPSR

tradisional. Pada nilai RO mengalami kenaikan dikarenakan GPSR modifikasi melakukan proses route discovery, dimana pada GPSR tradisional tidak ada.

4.2.2.2 Packet Delivery Ratio (PDR)



Gambar 4.12 Grafik PDR skenario *real*

Gambar 4.12 merupakan grafik perbandingan PDR antara protokol GPSR tradisional dengan protokol GPSR modifikasi. Hasil perhitungan rata-rata dan selisih nilai PDR dari skenario *real* dengan penambahan jumlah node dapat dilihat pada Tabel 4.5.

Tabel 4.5 Hasil perhitungan rata-rata PDR (%) skenario *real*

Jumlah Node	PDR GPSR		selisih
	tradisional	modifikasi	
50	24.5	64.0	39.5
100	49.3	63.3	14
150	27.1	64.7	7.6

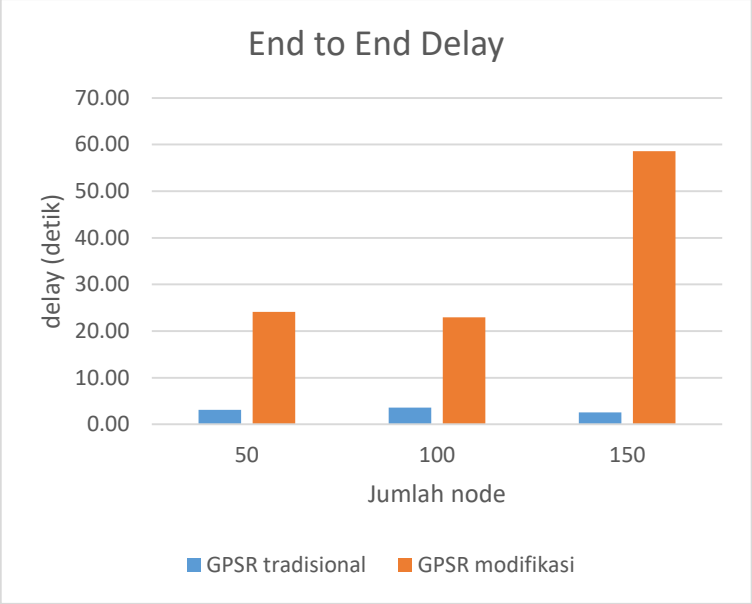
Berdasarkan Gambar 4.12 dapat dilihat bahwa PDR protokol GPSR modifikasi memiliki performa yang lebih baik dibandingkan dengan protokol

GPSR tradisional dengan peningkatan performa yang signifikan. Hal ini disebabkan adanya area *void* yang tidak terdapat *node*. Walaupun jarak transmisi tiap node adalah 200 meter, namun metode GPSR tradisional yang menerapkan posisi tujuan menjadi acuan dalam pemilihan *node* selanjutnya menyebabkan seringnya paket yang gagal terkirim.

Kegagalan paket terkirim pada GPSR tradisional disebabkan karena *node* yang bertugas meneruskan paket mengalami kondisi *local maximum*. GPSR modifikasi dapat menghindari *node* dengan keadaan *local maximum* yang memiliki posisi di dekat area *void* dalam pemilihan *node* selanjutnya. Tabel 4.5 menunjukkan bahwa GPSR tradisional dan GPSR modifikasi memiliki selisih PDR yang tinggi. Pada lingkungan dengan kepadatan yang rendah atau 50 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata PDR hingga 39.5 atau naik 161% dibandingkan dengan nilai rata-rata PDR GPSR tradisional.

Sedangkan pada lingkungan dengan kepadatan yang sedang atau 100 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata PDR hingga 14 atau naik 28% dibandingkan dengan nilai rata-rata PDR GPSR tradisional dan pada lingkungan dengan kepadatan yang tinggi atau 150 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata PDR hingga 7.6 atau naik 28% dibandingkan dengan nilai rata-rata PDR GPSR tradisional. Dari analisa tersebut, maka dapat disimpulkan bahwa GPSR modifikasi memiliki nilai PDR yang lebih tinggi dari pada GPSR tradisional baik di lingkungan dengan kepadatan rendah, kepadatan sedang ataupun kepadatan tinggi.

4.2.2.3 Average End to End Delay



Gambar 4.13 Grafik rata-rata *end to end delay* skenario *real*

Gambar 4.13 menunjukkan grafik perbandingan rata-rata *end to end delay* pada protokol GPSR tradisional dan GPSR modifikasi dari skenario *real*, dimana protokol GPSR modifikasi memiliki *delay* yang lebih tinggi. Hasil perhitungan rata-rata dan selisih nilai *end to end delay* dari skenario *real* dengan penambahan jumlah node dapat dilihat pada Tabel 4.6.

Tabel 4.6 Hasil perhitungan rata-rata *end to end delay* (detik) skenario *real*

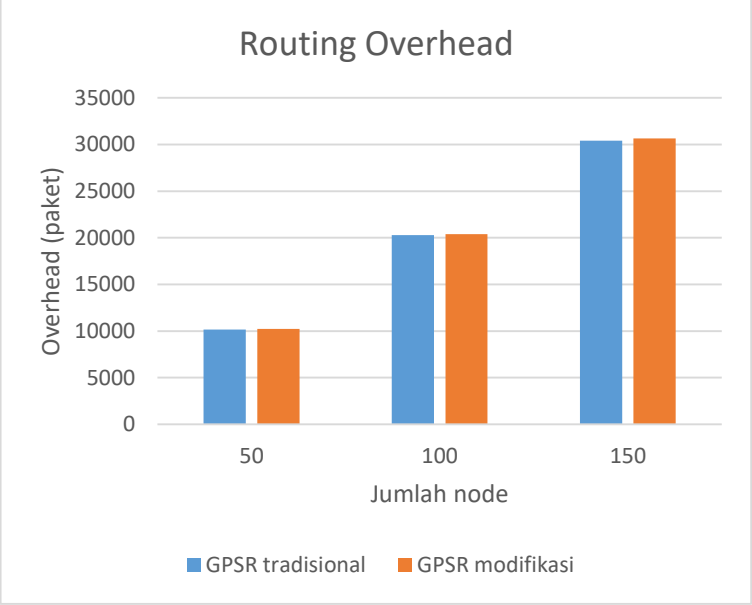
Jumlah Node	End to End Delay GPSR		selisih
	tradisional	modifikasi	
50	3.13	24.11	20.98
100	3.63	22.99	19.36
150	2.60	58.59	55.99

Tabel 4.6 menunjukkan bahwa GPSR tradisional dan GPSR modifikasi memiliki selisih *end to end delay* yang tinggi. Pada lingkungan dengan kepadatan yang rendah atau 50 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata *end to end delay* hingga 20.98 atau naik 670% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional. Sedangkan pada lingkungan dengan kepadatan yang

sedang atau 100 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata *end to end delay* hingga 19.36 atau naik 533% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional dan pada lingkungan dengan kepadatan yang tinggi atau 150 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata *end to end delay* hingga 55.99 meter per detik atau naik 2153% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional. Dari analisa tersebut, maka dapat disimpulkan bahwa GPSR modifikasi memiliki nilai *end to end delay* yang lebih tinggi daripada GPSR modifikasi di semua lingkungan dengan kepadatan rendah, sedang ataupun tinggi.

Perbedaan yang signifikan ini disebabkan karena pada GPSR tradisional memiliki jumlah paket terkirim cukup rendah, semakin tinggi jumlah *node* maka nilai PDR akan semakin rendah nilai *end to end delay* semakin tinggi. Sedangkan pada GPSR modifikasi dimana protokol berhasil mengirimkan paket yang relatif tinggi dibandingkan GPSR tradisional, sehingga semakin tinggi nilai jumlah *node* maka nilai PDR semakin tinggi dan nilai rata-rata *end to end delay* semakin tinggi.

4.2.2.4 Routing Overhead (RO)



Gambar 4.14 Grafik rata-rata RO skenario *real*

Gambar 4.14 menunjukkan grafik perhitungan rata-rata *routing* overhead pada skenario *grid* dimana nilai RO linear naik mengikuti naiknya jumlah *node*

pada simulasi baik pada GPSR tradisional maupun GPSR modifikasi. Hal ini disebabkan pada GPSR tradisional maupun GPSR modifikasi tidak adanya metode perbaikan rute yang gagal sehingga tidak berdampak pada kenaikan jumlah RO.

Tabel 4.7 Hasil perhitungan rata-rata RO (paket) skenario *real*

Jumlah Node	RO GPSR		selisih
	tradisional	modifikasi	
50	10161	10239	78
100	20284	20395	111
150	30406	30593	187

Tabel 4.7 menunjukkan bahwa GPSR tradisional dan GPSR modifikasi memiliki selisih RO yang kecil. Pada lingkungan dengan kepadatan yang rendah atau 50 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata RO hingga 78 atau naik 0.7% dibandingkan dengan nilai rata-rata RO GPSR tradisional. Sedangkan pada lingkungan dengan kepadatan yang sedang atau 100 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata RO hingga 111 atau naik 0.5% dibandingkan dengan nilai rata-rata RO GPSR tradisional dan pada lingkungan dengan kepadatan yang tinggi atau 150 *node*, GPSR modifikasi mendapatkan selisih nilai rata-rata RO hingga 187 atau naik 0.6% dibandingkan dengan nilai rata-rata *end to end delay* GPSR tradisional. Dari analisa tersebut, maka dapat disimpulkan bahwa GPSR modifikasi mengalami kenaikan nilai RO yang tidak signifikan dibandingkan dengan GPSR modifikasi di semua lingkungan dengan kepadatan rendah, sedang ataupun tinggi. Hal ini disebabkan GPSR modifikasi memiliki metode pencarian *Virtual Anchor Point* (VAP) sebelum melakukan pengiriman paket pertama. Metode tersebut membutuhkan paket *Route Request* (RREQ) dan *Route Reply* (RREP). Kedua tipe paket tersebut termasuk dalam perhitungan RO.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pengujian dan analisis yang telah dilakukan menghasilkan beberapa kesimpulan penelitian, yaitu:

1. Metode *greedy forwarding* pada GPSR dimodifikasi dengan melakukan perubahan acuan posisi destinasi dengan koordinat *virtual anchor point* (VAP). VAP merupakan representasi dari *overlay network* dimana VAP didapatkan dari proses *route discovery* yang dilakukan sebelum pengiriman paket.
2. Berdasarkan uji coba yang dilakukan pada skenario *real*, pencarian VAP memerlukan waktu 0.12 detik hingga 0.23 detik serta performa GPSR modifikasi dibandingkan dengan GPSR tradisional mengalami peningkatan nilai rata-rata *packet delivery ratio* (PDR) sebesar 72%, tetapi terjadi peningkatan pada nilai rata-rata *end to end delay* sebesar 1118% dan peningkatan nilai rata-rata *routing overhead* (RO) sebesar 0.6%.

5.2 Saran

Saran yang dapat diberikan dari hasil uji coba dan evaluasi adalah sebagai berikut:

1. Studi lebih lanjut mengenai formula pencarian *virtual anchor point* (VAP) yang bersifat dinamis. Dengan penelitian tersebut diharapkan posisi VAP dapat dibuat adaptif menyesuaikan tingkat kepadatan dan mobilitas *node*.
2. Penambahan faktor kecepatan dan arah pada pemilihan *node* penerus paket pada area optimum yang didapatkan dikarenakan dengan mobilitas *node* yang tinggi pada VANETs, *node* penerus paket dapat keluar dari area optimum sebelum meneruskan paket.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

Abbasi, I., Khan, A. dan Ali, S. (2018) "Dynamic Multiple Junction Selection Based Routing Protocol for VANETs in City Environment," *Applied Sciences*, 8(5), hal. 687. doi: 10.3390/app8050687.

Abdalla, A. M. (2017) "Performance evaluation for a unicast Non Delay Tolerant position based routing protocols in VANETs," *International Journal of Scientific Research and Management*, 5(12), hal. 7751–7757. doi: 10.18535/ijsrm/v5i12.22.

Abdalla Ahmed, A. I. *et al.* (2017) "Intersection-based Distance and Traffic-Aware Routing (IDTAR) protocol for smart vehicular communication," *2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC 2017*. IEEE, hal. 489–493. doi: 10.1109/IWCMC.2017.7986334.

Abdel-Halim, I. T. dan Fahmy, H. M. A. (2018) "Prediction-based protocols for vehicular Ad Hoc Networks: Survey and taxonomy," *Computer Networks*. Elsevier B.V., hal. 34–50. doi: 10.1016/j.comnet.2017.10.009.

Alsharif, N. (2017) *Connectivity-Aware Routing in Vehicular Ad Hoc Networks*. University of Waterloo. Tersedia pada: https://uwspace.uwaterloo.ca/handle/10012/11666.

Anggoro, R., Husni, M. dan Bastian, R. (2018) "Source route implementation using intersection node on GPSR protocol to increase VANET's data transmission reliability," *Proceeding - ICAMIMIA 2017: International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation*. IEEE, hal. 356–358. doi: 10.1109/ICAMIMIA.2017.8387618.

Balasubramani, Karthikeyan, L. dan Deepalakshmi, V. (2015) "Comparison study on non-delay tolerant routing protocols in vehicular networks," *Procedia Computer Science*. Elsevier Masson SAS, 50, hal. 252–257. doi: 10.1016/j.procs.2015.04.052.

Boussoufa-Lahlah, S., Semchedine, F. dan Bouallouche-Medjkoune, L. (2018) "Geographic routing protocols for Vehicular Ad hoc NETworks (VANETs): A

survey," *Vehicular Communications*. Elsevier Inc., 11, hal. 20–31. doi: 10.1016/j.vehcom.2018.01.006.

Devangavi, A. D. dan Gupta, D. R. (2017) "Routing Protocols in VANET - A Survey," in *International Conference On Smart Technology for Smart Nation*. IEEE, hal. 163–167.

Dora, D. P., Kumar, S. dan Joshi, M. (2016) "Impact of traffic signal on connectivity in intersection based connectivity aware geocast routing (IB-CAGR) in VANETs," *3rd International Conference on Signal Processing and Integrated Networks, SPIN 2016*. IEEE, hal. 4–8. doi: 10.1109/SPIN.2016.7566652.

Goel, N., Sharma, G. dan Dhyani, I. (2017) "A study of position based VANET routing protocols," *Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2016*. IEEE, hal. 655–660. doi: 10.1109/CCAA.2016.7813803.

Goudarzi, F., Asgari, H. dan Al-Raweshidy, H. S. (2019) "Traffic-aware VANET routing for city environments-a protocol based on ant colony optimization," *IEEE Systems Journal*. IEEE, 13(1), hal. 571–581. doi: 10.1109/JSYST.2018.2806996.

Hasrouny, H. *et al.* (2017) "VANet security challenges and solutions: A survey," *Vehicular Communications*. Elsevier Inc., 7, hal. 7–20. doi: 10.1016/j.vehcom.2017.01.002.

Houssaini, Z. S. *et al.* (2016) "Improvement of GPSR protocol by using future position estimation of participating nodes in vehicular ad-hoc Networks," *Proceedings - 2016 International Conference on Wireless Networks and Mobile Communications, WINCOM 2016: Green Communications and Networking*, hal. 87–94. doi: 10.1109/WINCOM.2016.7777196.

Hu, T. *et al.* (2015) "An enhanced GPSR routing protocol based on the buffer length of nodes for the congestion problem in VANETs," *10th International Conference on Computer Science and Education, ICCSE 2015*. IEEE, (Iccse), hal. 416–419. doi: 10.1109/ICCSE.2015.7250281.

Jang, J., Ahn, T. dan Han, J. (2017) "A new application-layer overlay platform for

better connected vehicles," *International Journal of Distributed Sensor Networks*, 13(11). doi: 10.1177/1550147717742072.

Jindal, V. dan Bedi, P. (2016) "Vehicular Ad-Hoc Networks: Introduction, Standards, Routing Protocols and Challenges," *International Journal of Computer Science Issues*, 13(2), hal. 44–55. doi: 10.20943/01201602.4455.

Kaur, H. dan Meenakshi (2018) "Analysis of VANET geographic routing protocols on real city map," *RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings*, 2018-Janua, hal. 895–899. doi: 10.1109/RTEICT.2017.8256727.

Khan, S. (2017) *Geometry-Predicting Communication Protocols for Car2X Applications*. Carl von Ossietzky University.

Khunt, N. H., Kodinariya, T. M. dan Sharma, S. S. (2016) "An Enhance Approach of Route Selection Technique of GPSR using Multi-Path Mechanism over MANET," 6(5), hal. 4721–4723. doi: 10.4010/2016.1173.

Lahlah, S. B. *et al.* (2016) "PSCAR: a proactive-optimal-path selection with coordinator agents assisted routing for vehicular ad hoc networks," *International Journal of High Performance Computing and Networking*, 11(2), hal. 129. doi: 10.1504/ijhpcn.2018.10010949.

Liu, J. *et al.* (2016) "A survey on position-based routing for vehicular ad hoc networks," *Telecommunication Systems*. Springer US, 62(1), hal. 15–30. doi: 10.1007/s11235-015-9979-7.

Nebbou, T. dan Lehsaini, M. (2018) "Greedy curvemetric-based routing protocol for VANETs," *2018 International Conference on Selected Topics in Mobile and Wireless Networking, MoWNeT 2018*. IEEE, hal. 1–6. doi: 10.1109/MoWNet.2018.8428952.

Oubbati, O. S. *et al.* (2017) "Intelligent UAV-assisted routing protocol for urban VANETs," *Computer Communications*, 107, hal. 93–111. doi: 10.1016/j.comcom.2017.04.001.

Patel, H. (2017) “Enhanced Route Selection Approach of Gpsr Using Multipath Forwarding in Vanet,” *International Journal of Advanced Research in Computer Science*, 8(8), hal. 140–145. doi: 10.26483/ijarcs.v8i8.4772.

Pete, B. dan Jaini, P. (2015) “Continuous connectivity aware routing in VANET using hybrid protocol,” *2nd International Conference on Electronics and Communication Systems, ICECS 2015*. IEEE, (Icecs), hal. 223–226. doi: 10.1109/ECS.2015.7124897.

Qin, H. dan Yu, C. (2017) “A road network connectivity aware routing protocol for Vehicular Ad Hoc Networks,” *2017 IEEE International Conference on Vehicular Electronics and Safety, ICVES 2017*. IEEE, hal. 57–62. doi: 10.1109/ICVES.2017.7991901.

Qureshi, K. N., Abdullah, A. H. dan Lloret, J. (2016) “Road Perception Based Geographical Routing Protocol for Vehicular Ad Hoc Networks,” *International Journal of Distributed Sensor Networks*, 2016. doi: 10.1155/2016/2617480.

Rehan, M. *et al.* (2015) “Vehicular Ad-hoc Networks for Smart Cities,” in Laouiti, A. (ed.) *Vehicular Ad-hoc Networks for Smart Cities, Advances in Intelligent Systems and Computing*. Singaporte: Springer Science+Business Media Singapore, hal. 75–84. doi: 10.1007/978-981-287-158-9.

Saleh, A. I., Gamel, S. A. dan Abo-Al-Ez, K. M. (2017) “A Reliable Routing Protocol for Vehicular Ad hoc Networks,” *Computers and Electrical Engineering*. Elsevier Ltd, 64, hal. 473–495. doi: 10.1016/j.compeleceng.2016.11.011.

Sangari, M. S. (2014) “A Comprehensive Survey on Efficient Routing Protocols And Simulation Tools For VANET,” *International Journal of Computer Science and Information Technologies*, 5(3), hal. 2729–2737.

Silva, A., Reza, K. M. N. dan Oliveira, A. (2018) “An Adaptive GPSR Routing Protocol for VANETs,” *Proceedings of the International Symposium on Wireless Communication Systems*. IEEE, 2018-Augus, hal. 1–6. doi: 10.1109/ISWCS.2018.8491075.

Singh, M. (2015) “Non-DTN Geographic Unicast Routing Protocol for VANET :

State of the," *International Journal of Current Engineering and Technology*, 5(5), hal. 3418–3425.

Tufail, R. M. dan Dawood, T. B. E. (2016) *Review of Routing Protocols in VANETs A Project Submitted in Partial Fulfillment of the Requirements for Degree of*. University of Victoria.

Ullah, A. *et al.* (2019) "Advances in Position Based Routing Towards ITS Enabled FoG-Oriented VANET-A Survey," *IEEE Transactions on Intelligent Transportation Systems*. IEEE, PP, hal. 1–13. doi: 10.1109/tits.2019.2893067.

Wang, W., Luo, T. dan Hu, Y. (2017) "Landmark-based routing using real-time urban traffic information in VANET," *2016 2nd IEEE International Conference on Computer and Communications, ICCC 2016 - Proceedings*. IEEE, hal. 2193–2197. doi: 10.1109/CompComm.2016.7925089.

Yang, H., Yu, M. dan Zeng, X. (2017) "Link available time prediction based GPSR for vehicular ad hoc networks," *Proceedings of the 2017 IEEE 14th International Conference on Networking, Sensing and Control, ICNSC 2017*. IEEE, hal. 293–298. doi: 10.1109/ICNSC.2017.8000107.

Yang, X. *et al.* (2018) "Improvement of GPSR Protocol in Vehicular Ad Hoc Network," *IEEE Access*, 6, hal. 39515–39524. doi: 10.1109/ACCESS.2018.2853112.

(Halaman ini sengaja dikosongkan)

LAMPIRAN

Lampiran 1. Modifikasi *file* gpsr.h

```
...
#include "gpsr_rtable.h"
#include "gpsr_vtable.h"
...
class GPSRAgent : public Agent {
private:
  ...
  double myx;
  double myy;
  double myprevx;
  double myprevy;
  int n=0;
  double totalgeospeed[500];
  double avg_speed;

  FILE *fp;
  FILE *fq;
  gpsr_rtable * rt;
  gpsr_vtable * vt;
  int rq_id;
  bool valid_route;
  nsaddr_t src;
  nsaddr_t dst;
  double thePath[16][2];
  int routeLength;
  ...
  bool checkRequest(Packet * );
  void sendReply(Packet * );
  void createVapRoute(Packet * );
  ...
};
```

Lampiran 2. Modifikasi fungsi GPSRAgent::GetLocation pada *file* gpsr.cc

```
void GPSRAgent::GetLocation(double *x, double *y){
  *x=node_->X();
  *y=node_->Y();
}
```

Lampiran 3. Modifikasi fungsi GPSRAgent:: GPSRAgent pada *file* gpsr.cc

```
GPSRAgent::GPSRAgent():
Agent(PT_GPSR), hello_timer_(this), query_timer_(this), my_id_(-
1),
my_x_(0.0), my_y_(0.0),recv_counter_(0), query_counter_(0),
query_period_(INFINITE_DELAY)
{
  bind("planar_type_", &planar_type_);
  bind("hello_period_", &hello_period_);

  sink_list_ = new Sinks();
  nblist_ = new GPSRNeighbors();

  /* modifikasi */
  rt= new gpsr_rtable();
  vt= new gpsr_vtable();
  src= NULL;
  dst= NULL;
  rq_id= 1;
  valid_route= 0;
  routeLength= 0;
  thePath[0][0]= NULL;
  /* modifikasi */

  for(int i=0; i<5; i++) randSend_.reset_next_substream();
}
```

Lampiran 4. Modifikasi fungsi GPSRAgent::turnon pada *file* gpsr.cc

```
void GPSRAgent::turnon(){
  getLoc();

  /* modifikasi */
  double avg_speed = calculate_speed((double)node_->speed());
  nblist_->myinfo(my_id_, my_x_, my_y_,avg_speed);
  myprevx=node_->X();
  myprevy=node_->Y();
  /* modifikasi */

  hello_timer_.resched(randSend_.uniform(0.0, 0.5));
}
```

Lampiran 5. Modifikasi fungsi GPSRAgent::hellomsg pada *file* gpsr.cc

```
void GPSRAgent::hellomsg(){
  ...
 /* modifikasi */
  double avg_speed = calculate_speed((double)node_->speed());
  nblist_->myinfo(my_id_, my_x_, my_y_,avg_speed);

  float temp_prevx = myprevx;
  float temp_prevy = myprevy;

  ghh->avg_speed = (float)avg_speed;
  ghh->prevx = temp_prevx;
  ghh->prevy = temp_prevy;
 /* modifikasi */
 ...
}
```

Lampiran 6. Modifikasi fungsi GPSRAgent::recvHello pada *file* gpsr.cc

```
void GPSRAgent::recvHello(Packet *p){

  struct hdr_cmn *cmh = HDR_CMN(p);
  struct hdr_gpsr_hello *ghh = HDR_GPSR_HELLO(p);

 /* modifikasi */
  nblist_->newNB(cmh->last_hop_, (double)ghh->x_, (double)ghh-
 >y_, (double)ghh->prevx, (double)ghh->prevy, (double)ghh-
 >avg_speed);
 /* modifikasi */
}
```

Lampiran 7. Modifikasi fungsi GPSRAgent::forwardData pada *file* gpsr.cc

```
void GPSRAgent::forwardData(Packet *p){
  ...
  /* modifikasi */
  getLoc();
  double avg_speed = calculate_speed((double)node_->speed());
  nblist_->myinfo(my_id_, my_x_, my_y_,avg_speed);
  /* modifikasi */
```

```
  if(cmh->direction() == hdr_cmn::UP && ((nsaddr_t)iph-
>daddr() == IP_BROADCAST || iph->daddr() == my_id_)){
    sinkRecv(p);
    printf("receive\n");
    port_dmux_->recv(p, 0);
    return;
  }
  else {
    struct hdr_gpsr_data *gdh=HDR_GPSR_DATA(p);

    /* modifikasi */
    struct hdr_overlay_data * op= HDR_OVERLAY_DATA(p);
    MobileNode *thisNode= (MobileNode *)(Node::get_node_by_addres
s(my_id_));

    // jika node bukan source
    if (iph->saddr() != my_id_) {
      #ifdef DEBUG_ALDO
        fp= fopen ("debug.txt", "a");
        double distance= nblist_->getdis(thisNode->X(), thisNode-
>Y(), gdh->dx_, gdh->dy_);
        fprintf (fp, "%f distance= %f\t", CURRTIME, distance);
        fprintf (fp, "index : %d\n", op->index);
        fclose (fp);
      #endif

      // change source to this node
      if (op->index < op->length){
        gdh->sx_= my_x_;
        gdh->sy_= my_y_;

        gdh->dx_= op->path[op->index][0];
        gdh->dy_= op->path[op->index][1];

        // increase index vap
        op->index++;
      }
      else {
        gdh->sx_= my_x_;
        gdh->sy_= my_y_;

        MobileNode * dstNode= (MobileNode * )(Node::get_node_by_a
ddress(iph->daddr()));
          gdh->dx_= dstNode->X();
```

```
        gdh->dy_= dstNode->Y();
      }
    }
    /* modifikasi */


    double dx = gdh->dx_;
    double dy = gdh->dy_;

    // destination
    nsaddr_t dst_id = iph->daddr();

    nsaddr_t nexthop;
    if(gdh->mode_ == GPSR_MODE_GF){
      nexthop = nblist_->gf_nexthop(dx, dy, dst_id);

      /* modifikasi */
      MobileNode * nextNode= (MobileNode * )(Node::get_node_by_ad
dress(nexthop));
      gdh->dx_= nextNode->X();
      gdh->dy_= nextNode->Y();
      /* modifikasi */

      // perimeter
      if(nexthop == -1){
        nexthop = nblist_->peri_nexthop(planar_type_, -1,
              gdh->sx_, gdh->sy_,
              gdh->dx_, gdh->dy_);
        gdh->sx_ = my_x_;
        gdh->sy_ = my_y_;
        gdh->mode_ = GPSR_MODE_PERI;
      }
    }
    // perimeter
    else {
      double sddis = nblist_->getdis(gdh->sx_, gdh->sy_, gdh-
>dx_, gdh->dy_);
      double mydis = nblist_->getdis(my_x_, my_y_, gdh->dx_, gdh-
>dy_);
      if(mydis < sddis){
        //switch back to greedy forwarding mode
        nexthop = nblist_->gf_nexthop(dx, dy, dst_id);
        gdh->mode_ = GPSR_MODE_GF;
```

```
        if(nexthop == -1){
          nexthop = nblist_->peri_nexthop(planar_type_, -1,
                gdh->sx_, gdh->sy_,
                gdh->dx_, gdh->dy_);
          gdh->sx_ = my_x_;
          gdh->sy_ = my_y_;
          gdh->mode_ = GPSR_MODE_PERI;
        }
      }
      else{ //still perimeter routing mode
        nexthop = nblist_->peri_nexthop(planar_type_, cmh-
>last_hop_,
              gdh->sx_, gdh->sy_, gdh->dx_, gdh->dy_);
      }
    }

    cmh->direction() = hdr_cmn::DOWN;
    cmh->addr_type() = NS_AF_INET;
    cmh->last_hop_ = my_id_;
    cmh->next_hop_ = nexthop;
    send(p, 0);
  }
}
```

Lampiran 8. Penambahan fungsi GPSRAgent::checkRequest pada *file* gpsr.cc

```
bool GPSRAgent::checkRequest(Packet * p) {
  struct hdr_ip * iph= HDR_IP(p);
  struct hdr_overlay_data * op= HDR_OVERLAY_DATA(p);

  if (rt->rt_getrid(iph->saddr()) >= op->rq_id) return true;

  if (op->rq_ttl== 0) return true;

  if (iph->saddr()== my_id_) return true;

  for (int i= 0; i < op->length; i++) {
    if (op->nodePath[i]== my_id_) return true;
  }
  return false; //Request OK
}
```

Lampiran 9. Penambahan fungsi GPSRAgent::sendReply pada *file* gpsr.cc

```
void GPSRAgent::sendReply(Packet * p) {
  struct hdr_ip * iph= HDR_IP(p);
  struct hdr_overlay_data * op= HDR_OVERLAY_DATA(p);
  //Create new RREP packet
  Packet * rp= allocpkt();
  struct hdr_cmn * rcmh= HDR_CMN(rp);
  struct hdr_ip * riph= HDR_IP(rp);
  struct hdr_overlay_data * roverlay= HDR_OVERLAY_DATA(rp);
  riph->saddr()= my_id_;
  riph->sport()= RT_PORT;
  riph->daddr()= iph->saddr();
  riph->dport()= RT_PORT;
  riph->ttl()= 255;

  // OVERLAY header
  roverlay->pValid= 1;
  roverlay->repValid= 1;
  roverlay->rp_id= op->rq_id;
  roverlay->type_= GPSRTYPE_ROUTEDISC;
  roverlay->length= op->length;
  roverlay->pathLen= 0;
  int param= 0;
  for (int i= roverlay->length; i >= 0; i--) {
    roverlay->nodePath[param]= op->nodePath[i];
    param++;
  }
  // insert Loc of Destination
  MobileNode * tempNode= (MobileNode * )(Node::get_node_by_addres
s(my_id_));
  roverlay->path[roverlay->pathLen][0]= tempNode->X();
  roverlay->path[roverlay->pathLen][1]= tempNode->Y();
  roverlay->overlayNode[roverlay->pathLen]= my_id_;
  roverlay->pathLen++;

  roverlay->index= 1;
  rcmh->next_hop_= roverlay->nodePath[roverlay->index];
  rcmh->last_hop_= my_id_;
  rcmh->addr_type_= NS_AF_INET;
  rcmh->ptype()= PT_GPSR;
  rcmh->size()= IP_HDR_LEN + roverlay->size();
  rcmh->direction()= hdr_cmn::DOWN;
  Scheduler::instance().schedule(this, rp, 0.010);
}
```

Lampiran 10. Penambahan fungsi GPSRAgent::createVapRoute *file* gpsr.cc

```
void GPSRAgent::createVapRoute(Packet * p){
  struct hdr_overlay_data * op= HDR_OVERLAY_DATA(p);
  src= op->nodePath[op->length];
  dst= op->nodePath[0];
  double tmp_vroute[16][2];
  int tmp_index=0;
  for(int i=0; i < op->pathLen + 1; i++){
    if (i - 2 < 0) continue;
    // ujung source
    if (i== op->pathLen){
      MobileNode * thisNode= (MobileNode * )(Node::get_node_by_ad
dress(src));
      double src_x= thisNode->X();
      double src_y= thisNode->Y();
      // jarak node a ke node b
      double p= nblist_->getdis(op->path[i-2][0], op->path[i-
2][1], op->path[i-1][0], op->path[i-1][1]);
      // jarak node b ke node c
      double q= nblist_->getdis(op->path[i-1][0], op->path[i-
1][1], src_x, src_y);
      // jarak node a ke node c
      double r= nblist_->getdis(op->path[i-2][0], op->path[i-
2][1], src_x, src_y);
      // triangle
      if (r <= p + q) {
        tmp_vroute[tmp_index][0]= (op->path[i-2][0] + src_x) / 2;
        tmp_vroute[tmp_index][1]= (op->path[i-2][1] + src_y) / 2;
        tmp_index++;
      }
    }
    else {
      // jarak node a ke node b
      double p= nblist_->getdis(op->path[i-2][0], op->path[i-
2][1], op->path[i-1][0], op->path[i-1][1]);

      // jarak node b ke node c
      double q= nblist_->getdis(op->path[i-1][0], op->path[i-
1][1], op->path[i][0], op->path[i][1]);

      // jarak node a ke node c
      double r= nblist_->getdis(op->path[i-2][0], op->path[i-
2][1], op->path[i][0], op->path[i][1]);
```

```
      if (r <= p + q) {
        tmp_vroute[tmp_index][0]= (op->path[i-2][0] + op-
>path[i][0]) / 2;
        tmp_vroute[tmp_index][1]= (op->path[i-2][1] + op-
>path[i][1]) / 2;
        tmp_index++;
      }
    }
  }

  for (int j= tmp_index - 1; j >= 0; j--){
    vt->vt_add(src, dst, tmp_vroute[j][0], tmp_vroute[j][1]);
  }
  for (int j= 0; j < tmp_index; j++){
    vt->vt_add(dst, src, tmp_vroute[j][0], tmp_vroute[j][1]);
  }

  routeLength = vt->vt_count(src, dst);
  for (int i= 0; i < routeLength; i++) {
    thePath[i][0]=  vt->vt_getXbyIndex(src, dst, i);
    thePath[i][1]=  vt->vt_getYbyIndex(src, dst, i);
  }
}
```

Lampiran 11. Modifikasi fungsi GPSRAgent::recv pada *file* gpsr.cc

```
void GPSRAgent::recv(Packet *p, Handler *h){
  struct hdr_cmn *cmh = HDR_CMN(p);
  struct hdr_ip *iph = HDR_IP(p);
  getLoc();

  /*modifikasi*/
  struct hdr_overlay_data * op= HDR_OVERLAY_DATA(p);

  if (op->pValid == 0) { // overlay header not valid
    if (iph->saddr()== my_id_) { // I'm the Source
      if (valid_route== 1) {
        if (src != iph->saddr() && dst != iph->daddr()) {
          valid_route= 0;
        }
      }
      if (valid_route== 0) { // Route Discovery
        op->pValid= 1; // overlay header valid
```

```
        // header RREQ
        op->reqValid= 1;
        op->rq_id=    rq_id++;
        op->rq_ttl=   20;
        op->length=   0;
        op->index=    0;
        op->nodePath[op->length]= my_id_;
        op->length++;
        // type
        op->type_= GPSRTYPE_ROUTEDISC;
        cmh->next_hop_=   IP_BROADCAST;
        cmh->last_hop_=   my_id_;
        cmh->addr_type_=  NS_AF_INET;
        cmh->ptype()=     PT_GPSR;
        cmh->size()=      IP_HDR_LEN + op->size();
        cmh->direction()= hdr_cmn::DOWN;
        cmh->num_forwards()= 0;

        iph->saddr()= my_id_;
        iph->sport()= RT_PORT;
        iph->daddr()= iph->daddr();
        iph->dport()= RT_PORT;
      }
      else { // forward data
        if (iph->saddr()== my_id_) {
          if (cmh->num_forwards()== 0) {
            struct hdr_gpsr_data * gdh= HDR_GPSR_DATA(p);
            cmh->size() += IP_HDR_LEN + gdh->size() + op->size();
            //overlay data
            op->pValid= 0;
            op->length= routeLength;
            op->index= 0;
            for (int i= 0; i < routeLength; i++) {
              for (int j= 0; j < 2; j++) {
                op->path[i][j]= thePath[i][j];}
            }
            //gpsr data
            gdh->type_= GPSRTYPE_DATA;
            gdh->mode_= GPSR_MODE_GF;
            gdh->sx_= (float) my_x_;
            gdh->sy_= (float) my_y_;
            gdh->dx_= (float) op->path[op->index][0];
            gdh->dy_= (float) op->path[op->index][1];
            gdh->ts_= (float) GPSR_CURRENT;
            gdh->indexvap= 0;
```

```
          }
          // routing loop
          else if (cmh->num_forwards() > 0) {
            if (cmh->ptype() != PT_GPSR)
             drop(p, DROP_RTR_ROUTE_LOOP);
            else Packet::free(p);
            return;
          }
        }
      }
    }
  }
  else if (op->pValid== 1) { // overlay header valid
    if (iph->daddr()== my_id_) { // I'm not the Destination
      if (op->reqValid) { // RREQ
        op->nodePath[op->length]= my_id_;
        sendReply(p);
        goto done;
      }
      else if (op->repValid) { // RREP
        if (valid_route== 1) {
          goto done;
        }
        MobileNode * prevNode= (MobileNode * )(Node::get_node_by_
address(cmh->last_hop_));
        MobileNode * thisNode= (MobileNode * )(Node::get_node_by_
address(my_id_));
        // check intersection
        if (prevNode->X() != thisNode->X() && prevNode-
>Y() != thisNode->Y()) {
          // check Overlay Node list
          if (op->overlayNode[op->pathLen - 1] != cmh-
>last_hop_) {
            op->path[op->pathLen][0]= prevNode->X();
            op->path[op->pathLen][1]= prevNode->Y();
            op->overlayNode[op->pathLen]= cmh->last_hop_;
          }
        }
        else {op->pathLen--;}
        int temp= 0;
        if (op->pathLen== 0) {
          thePath[op->pathLen][0]= op->path[op->pathLen][0];
          thePath[op->pathLen][1]= op->path[op->pathLen][1];
          temp++;
        }
```

```
        else {
          for (int i= op->pathLen - 1; i >= 0; i--) {
            for (int j= 0; j < 2; j++) {
              thePath[temp][j]= op->path[i][j];
            }
            temp++;
          }
        }
        routeLength= temp;
        src= op->nodePath[op->length];
        dst= op->nodePath[0];
        valid_route= 1;

        createVapRoute(p);
        goto done;
      }
    }
    // I'm not the Destination
    else { //forward data
      if (op->reqValid) { // RREQ
        cmh->num_forwards() ++;
        op->rq_ttl--;
        if (checkRequest(p)) {
          Packet::free(p);
          goto done;
        }
        MobileNode * lastNode= (MobileNode * )(Node::get_node_by_
address(cmh->last_hop_));
        nblist_->newNB(cmh->last_hop_, lastNode->X(), lastNode-
>Y(),0,0,0);
       rt->rt_add(iph->saddr(), op->rq_id);
        //update RREQ header
        op->nodePath[op->length]= my_id_;
        op->length++;
        cmh->direction()= hdr_cmn::DOWN;
        cmh->addr_type()= NS_AF_INET;
        cmh->last_hop_=   my_id_;
        cmh->next_hop_=   IP_BROADCAST;
      }
      else if (op->repValid) { // RREP
        MobileNode * prevNode= (MobileNode * )(Node::get_node_by_
address(cmh->last_hop_));
        MobileNode * thisNode= (MobileNode * )(Node::get_node_by_
address(my_id_));
```

```
        if (prevNode->X() != thisNode->X() && prevNode-
>Y() != thisNode->Y()) {
          if (op->overlayNode[op->pathLen - 1]== cmh-
>last_hop_) {
            op->path[op->pathLen][0]= thisNode->X();
            op->path[op->pathLen][1]= thisNode->Y();
            op->overlayNode[op->pathLen]= my_id_;
            op->pathLen++;
          }
          else {
            op->path[op->pathLen][0]= prevNode->X();
            op->path[op->pathLen][1]= prevNode->Y();
            op->overlayNode[op->pathLen]= cmh->last_hop_;
            op->pathLen++;
            op->path[op->pathLen][0]= thisNode->X();
            op->path[op->pathLen][1]= thisNode->Y();
            op->overlayNode[op->pathLen]= my_id_;
            op->pathLen++;
          }
        }
        cmh->num_forwards() ++;
        cmh->addr_type()= NS_AF_INET;
        cmh->last_hop_= my_id_;
        cmh->next_hop_= op->nodePath[op->index];
        op->index++;
        cmh->direction()= hdr_cmn::DOWN;
      }
    }
  }
  if(cmh->ptype() == PT_GPSR){
    struct hdr_gpsr *gh = HDR_GPSR(p);
    switch(gh->type_){
    case GPSRTYPE_HELLO:
      recvHello(p);
      break;
    case GPSRTYPE_QUERY:
      recvQuery(p);
      break;
    case GPSRTYPE_ROUTEDISC:
      send(p, 0);
      break;
    default:
      exit(1);
    }
  }
```

```
    else {
    iph->ttl_--;
    if(iph->ttl_ == 0){
      drop(p, DROP_RTR_TTL);
      return;
    }
    forwardData(p);
  }
  done:
    p= 0;
  return;
}
```

Lampiran 12. Modifikasi pada *file* gpsr_neighbor.h

```
...
struct gpsr_neighbor {
  ...
  double myprevx;
  double myprevy;
  double avg_speed;
  ...
};
...
struct candidatenexthop {
  nsaddr_t id_;
  double dif_speed;
  double dt1;
  double dt2;
  double tempdis;
};
class GPSRNeighbors {
private:
  ...
  double myprevx;
  double myprevy;
  double avg_speed;

  void swap(struct candidatenexthop *, struct candidatenexthop *)
;
  void bubbleSort(struct candidatenexthop [], int);
  void bubbleSort_tempdis(struct candidatenexthop [], int);
  ...
};
```

Lampiran 13. Modifikasi fungsi GPSRNeighbors::newNB pada gpsr_neighbor.cc

```
void
GPSRNeighbors::newNB(nsaddr_t nid, double nx, double ny,double pr
evx, double prevy, double avg_speed){
  struct gpsr_neighbor *temp = getnb(nid);
  if(temp==NULL){ //it is a new neighbor
    ...
    temp->myprevx = prevx;
    temp->myprevy = prevy;
    temp->avg_speed = avg_speed;
    ...
  }
  else { //it is a already known neighbor
    ...
    temp->myprevx = prevx;
    temp->myprevy = prevy;
    temp->avg_speed = avg_speed;
  }
}
```

Lampiran 14. Penambahan fungsi GPSRNeighbors::swap pada gpsr_neighbor.cc

```
void
GPSRNeighbors::swap(struct candidatenexthop *xp, struct candidate
nexthop *yp) {
  struct candidatenexthop temp = *xp;
  *xp = *yp;
  *yp = temp;
}
```

Lampiran 15. Penambahan fungsi GPSRNeighbors::bublesort gpsr_neighbor.cc

```
void
GPSRNeighbors::bubbleSort(struct candidatenexthop arr[], int n){
  int i, j;
  for (i = 0; i < n-1; i++){
     for (j = 0; j < n-i-1; j++){
      if (arr[j].tempdis > arr[j+1].tempdis){
        swap(&arr[j], &arr[j+1]);}
    }
  }
}
```

Lampiran 16. Modifikasi fungsi GPSRNeighbors::gf_nexthop gpsr_neighbor.cc

```
nsaddr_t
GPSRNeighbors::gf_nexthop(double dx, double dy, nsaddr_t dst_id){
  struct gpsr_neighbor *temp = head_;
  //initializing the minimal distance as my distance to sink
  double mindis =getdis(my_x_, my_y_, dx, dy);
  nsaddr_t nexthop = -1; //the nexthop result

  struct candidatenexthop allowednode[100];
  int counterallowednode = 0;
  nsaddr_t temp_nexthop = -1; // id node acuan
  double tempX, tempY; // menyimpan lokasi node acuan
  while(temp){
    double tempdis = getdis(temp->x_, temp->y_, dx, dy);
    if(tempdis < mindis){
      mindis = tempdis;
      nexthop = temp->id_; //SET NEXTHOP

      temp_nexthop = temp->id_;
      tempX = temp->x_;
      tempY = temp->y_;
    }
    temp = temp->next_;
  }
  if(temp_nexthop == dst_id) return nexthop;
  else {
    // MENGHITUNG dSB dan dBD
    double dSB = getdis(my_x_, my_y_, tempX, tempY);
    double dBD = getdis(tempX, tempY, dx, dy);

    // MENGHITUNG dmax
    double dmax = dBD + (0.3 * dSB);
    double dmax2 = dmax * dmax;

    temp = head_;
    while(temp){
      double hasil = ((temp->x_-dx)*(temp->x_-dx)) + ((temp->y_-
dy)*(temp->y_-dy));
      //titik ada dalam lingkaran
    if(hasil <= dmax2){
        double dt1 = getdis(temp->myprevx, temp-
>myprevy, dx, dy);
        double dt2 = getdis(temp->x_, temp->y_, dx, dy);
        double diffspeed = avg_speed - temp->avg_speed;
        double tempdis = getdis(temp->x_, temp->y_, dx, dy);
```

```
        if(diffspeed < 0){
          diffspeed = diffspeed * -1;
        }

        allowednode[counterallowednode].id_ = temp->id_;
        allowednode[counterallowednode].dt1 = dt1;
        allowednode[counterallowednode].dt2 = dt2;
        allowednode[counterallowednode].dif_speed = diffspeed;
        allowednode[counterallowednode].tempdis = tempdis;
        counterallowednode++;
      }
      temp = temp->next_;
    }

    bubbleSort(allowednode, counterallowednode);
    int nodefound = 0;

    for (int i = 0; i < counterallowednode; i++)
    {
      if (allowednode[i].dt2 <= allowednode[i].dt1)
      {
        nexthop = allowednode[i].id_;
        nodefound = 1;
        break;
      }
    }

    if(nodefound == 0){
      bubbleSort_tempdis(allowednode, counterallowednode);
      nexthop = allowednode[0].id_;
    }
    return nexthop;
  }
}
```

Lampiran 17. Modifikasi fungsi GPSRNeighbors::newNB gpsr_neighbor.cc

```
...
# define GPSRTYPE_ROUTEDISC 0x03 //Route Discovery
# define HDR_OVERLAY_DATA(p)((struct hdr_overlay_data * ) hdr_gps
r::access(p))
...
```

```
struct hdr_gpsr_hello {
  ...
  float avg_speed;
  float prevx;
  float prevy;
  ...
};
struct hdr_gpsr_data {
  ...
  int indexvap;
  ...
};

struct hdr_overlay_data {
  u_int8_t type_;
  int pValid; //packet overlay valid
  //RREQ PACKET
  int reqValid; //is RREQ
  int rq_id; //Request ID
  int rq_ttl;
  //RREP PACKET
  int repValid; //is RREP
  int rp_id; //Reply ID
  nsaddr_t nodePath[16]; //Path of node
  nsaddr_t overlayNode[16]; //list of Overlay Node;
  double path[16][2]; //loc X,Y node
  int pathLen; //length path
  int length; //length nodePath
  int index;
  inline int size() {
    int sz =
      9 * sizeof(int) +
      (16 * 2 + 1) * sizeof(u_int8_t) +
      16 * 2 * sizeof(double);
    return sz;
  }
};

union hdr_all_gpsr {
  ...
  hdr_overlay_data overlay;
  ...
};
```

Lampiran 18. Penambahan *file* gpsr_rtable.h

```
#ifndef __gpsr_rtable_h__
#define __gpsr_rtable_h__

#include <config.h>


/*
  Route Table Entry
*/

struct gpsr_rt_entry {
  nsaddr_t rt_id;
  int rt_reqno;
};

//Route Table

class gpsr_rtable {
  struct gpsr_rt_entry * rt;
  int len;
  public:
    gpsr_rtable();
  int rt_lookup(nsaddr_t id);
  int rt_getrid(nsaddr_t id);
  void rt_add(nsaddr_t id, int reqno);
};

#endif /* _gpsr__rtable_h__ */
```

Lampiran 19. Penambahan *file* gpsr_rtable.cc

```
#include <gpsr/gpsr_rtable.h>
gpsr_rtable::gpsr_rtable() {
  rt = new gpsr_rt_entry[128];
  len = 0;
}
int gpsr_rtable::rt_lookup(nsaddr_t id) {
  for (int i = 0; i < len; i++) {
    if (rt[i].rt_id == id) return i;
  }
  return len;
}
```

```
int gpsr_rtable::rt_getrid(nsaddr_t id) {
  int param = rt_lookup(id);
  if (param >= len) return 0;
  return rt[param].rt_reqno;
}

void gpsr_rtable::rt_add(nsaddr_t id, int reqno) {
  int param = rt_lookup(id);
  if (param < len) rt[param].rt_reqno = reqno;

  rt[len].rt_id = id;
  rt[len].rt_reqno = reqno;
  len++;
}
```

Lampiran 20. Penambahan *file* gpsr_vtable.h

```
#ifndef __gpsr_vtable_h__
#define __gpsr_vtable_h__
#include <config.h>

struct gpsr_vt_entry {
  int vt_id;
  nsaddr_t vt_src;
  nsaddr_t vt_dst;
  int vt_index;
  float vt_x;
  float vt_y;
};
class gpsr_vtable {
  struct gpsr_vt_entry * vt;
  int len;
  public:
    gpsr_vtable();
  int vt_count(nsaddr_t src, nsaddr_t dst);
  int vt_lookup(nsaddr_t src, nsaddr_t dst, int index);
  void vt_add(nsaddr_t src, nsaddr_t dst, float x, float y);
  float vt_getXbyIndex(nsaddr_t src, nsaddr_t dst, int index);
  float vt_getYbyIndex(nsaddr_t src, nsaddr_t dst, int index);
  int vt_getlen();
};
#endif /* _gpsr__vtable_h__ */
```

Lampiran 21. Penambahan *file* gpsr_vtable.cc

```
#include <gpsr/gpsr_vtable.h>
gpsr_vtable::gpsr_vtable() {
  vt = new gpsr_vt_entry[128];
  len = 0;
}
int gpsr_vtable::vt_count(nsaddr_t src, nsaddr_t dst) {
  int result=0;
  for (int i = 0; i < len; i++) {
    if ( (vt[i].vt_src == src) && (vt[i].vt_dst  == dst) ) {
      result++;}}
  return result;
}

int gpsr_vtable::vt_lookup(nsaddr_t src, nsaddr_t dst, int indeks
){
  for (int i = 0; i < len; i++) {
    if ( (vt[i].vt_src == src) && (vt[i].vt_dst==dst) && (vt[i].v
t_index==index) ) {
      return i;}}
  return -1;
}
float gpsr_vtable::vt_getXbyIndex(nsaddr_t src, nsaddr_t dst, int
 index){
  int id= vt_lookup(src, dst, index);
  if (id > -1) return vt[id].vt_x;
  else return -1.0;
}
float gpsr_vtable::vt_getYbyIndex(nsaddr_t src, nsaddr_t dst, int
 index){
  int id= vt_lookup(src, dst, index);
  if (id > -1) return vt[id].vt_y;
  else return -1.0;
}
int gpsr_vtable::vt_getlen(){return len;}
void gpsr_vtable::vt_add(nsaddr_t src,nsaddr_t dst,float x,float
y){
  vt[len].vt_id=    len;
  vt[len].vt_src=   src;
  vt[len].vt_dst=   dst;
  vt[len].vt_index= vt_count(src, dst);
  vt[len].vt_x =    x;
  vt[len].vt_y =    y;
  len++;
}
```

Lampiran 22. Script awk untuk evaluasi hasil simulasi

```
BEGIN {
    sendLine = 0;
    recvLine = 0;
    fowardLine = 0;
    TC =0;
    rt_pkts=0;
    rt_send=0;
    rt_forward=0;
}
$0 ~/^s.* AGT/ {sendLine ++ ;}
$0 ~/^r.* AGT/ {recvLine ++ ;}
$0 ~/^f.* RTR/ {fowardLine ++ ;}
{
    if($4 == "AGT" && $1 == "s" && seqno < $6) {
     seqno = $6;
    }

    #end-to-end delay
    if($4 == "AGT" && $1 == "s") {
        start_time[$6] = $2;
        hc[$6]=0;
    }
    else if(($7 == "cbr") && ($1 == "r")) {
        end_time[$6] = $2;
        hc[$6]++;
    }
    else if($1 == "D" && $7 == "cbr") {
        end_time[$6] = -1;
        # hc[$6] = -1;
    }

    if (($1 == "s" || $1 == "f") && ($4 == "RTR") && ($7 == "gpsr
")) {
        rt_pkts++;
    }
    if (($1 == "s") && ($4 == "RTR") && ($7 == "gpsr")) {
        rt_send++;
    }
    if (($1 == "s") && ($4 == "RTR") && ($7 == "gpsr") && ($3 !=
"_98_")) {
        rt_forward++;
    }
```

```
    if (($1 == "s") && ($7 == "cbr")) {
        rt_tcppktno++;
    }
}

END {
    sum_hc=0;
    index_hc=0;
    for(i=0; i<=seqno; i++) {
        if(end_time[i] > 0) {
            delay[i] = end_time[i] - start_time[i];
            count++;
        }
        else {
            delay[i] = -1;
        }

        if (hc[i] > 0){
            sum_hc+= hc[i];
            index_hc++;
        }
    }

    for(i=0; i<=seqno; i++) {
        if(delay[i] > 0) {
            n_to_n_delay = n_to_n_delay + delay[i];
        }
    }
    n_to_n_delay = n_to_n_delay/count;

    if ( index_hc > 0 ){
        # hop_count = sum_hc / sendLine;
        hop_count = sum_hc / index_hc;
    }
    else {
        hop_count = 0;
    }
    printf  sendLine","recvLine",%.4f,"(sendLine-
recvLine)","n_to_n_delay * 1000","rt_pkts","rt_send","rt_forward"
,%.0f\n", (recvLine/sendLine), hop_count;
}
```

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Rizky Fenaldo Maulana, lahir di Denpasar, 14 Juli 1997. Penulis adalah anak keempat dari empat bersaudara. Penulis menempuh pendidikan sekolah dasar di SDN Dinoyo 2 Malang lalu melanjutkan pendidikan sekolah menengah pertama di SMPN 4 Malang dan penulis menempuh pendidikan menengah atas di SMA Negeri 1 Malang. Selanjutnya penulis melanjutkan pendidikan sarjana dan pascasarjana di Departemen Teknik Informatika, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember Surabaya. Selama kuliah, penulis aktif dalam berbagai organisasi baik tingkat jurusan maupun institut.

Saat pendidikan sarjana, penulis mengambil bidang minat Arsitektur dan Jaringan Komputer (AJK). Sebagai mahasiswa, penulis berperan aktif dalam beberapa organisasi kampus seperti staf Dalam Negeri Himpunan Mahasiswa Teknik-Computer (HMTC) ITS, staf ahli Himpunan Mahasiswa Teknik-Computer (HMTC) ITS , staf Kementerian Pemuda dan Kebangsaan BEM ITS dan Gerigi 2015. Selain itu, penulis juga menjadi staf REEVA SCHEMATICS 2015 dan koordinator 1 REEVA pada acara SCHEMATICS 2016. Penulis pernah melakukan kerja praktik di PT. Telekomunikasi Indonesia Juni – Agustus 2017 dan membuat aplikasi berbasis web Monitoring FO Migrations.

Dalam menyelesaikan pendidikan pascasarjana, penulis mengambil bidang minat Komputasi Berbasis Jaringan (KBJ). Penulis juga mendapatkan kesempatan sebagai asisten dosen pada beberapa mata kuliah bidang minat KBJ baik S1 maupun S2, antara lain jaringan nirkabel (S1 – 2018 Gasal), desain audit jaringan (S2 – 2019 gasal), jaringan nirkabel (S1 – 2018 Genap) dan topik khusus (S1 – 2018 Genap, 2019 Gasal). Penulis juga berkesempatan sebagai asisten pembimbing tugas akhir bidang minat KBJ oleh pembimbing Dr. Eng Radityo Anggoro. S.Kom., M.Sc. Penulis dapat dihubungi melalui nomor *handphone*: 081230777099 atau *email*: rizkyfenaldoo@gmail.com.

(Halaman ini sengaja dikosongkan)