PENGARUH PENERAPAN ALGORITMA *CLUSTERING* COEEC TERHADAP KINERJA PROTOKOL *ROUTING* DSR DI JARINGAN MANET

(Effect of COEEC Clustering Algorithm on DSR Routing Protocols Performance in MANET)

Lalu Ikhwan Rosadi, Andy Hidayat Jatmika*, Sri Endang Anjarwani
Dept Informatics Engineering, Mataram University
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: laluikhwan17@gmail.com,[andy, endang]@unram.ac.id

Abstract

MANET can be used as an alternative communication solution because it is mobile, meaning that it does not have a fixed infrastructure. MANET can be applied to earthquake disaster areas. Manet can use routing protocols as communication between nodes, such as the DSR routing protocol. DSR routing protocol works when there is a request for sending packets from the source node to the destination node using two main mechanisms namely, route discovery and route maintenance. Route discovery occurs when the process of sending data from the source node to the destination node by performing a flooding mechanism to the neighboring node. Flooding mechanism can cause an increase in routing overhead on the DSR routing protocol. To overcome these problems is clustering by utilizing the Overhead Energy Efficient Cluster (COEEC) algorithm on the DSR routing protocol. The results of the addition of clusters in the DSR routing protocol with the COEEC algorithm can increase throughput performance by 43% and 31%, then an increase in the average end to end delay of 0.005% and 0.01%, then the packet delivery ratio parameter increases by 41% and 27%, then an increase in the average residual energy by 15% and 17%, and can suppress routing overhead values by 87% and 81%.

Keywords: *MANET, Flooding, DSR, COEEC, Routing Overhead*

**Penulis Korespondensi*

1. PENDAHULUAN

Mobile Ad Hoc Network (MANET) merupakan sekumpulan *node* yang bergerak bebas dan berdiri sendiri dimana topologinya sering berubah. Setiap terjadinya perubahan topologi atau ukuran jaringan, maka dapat mempengaruhi kinerja dari protokol *routing*. MANET dapat diterapkan pada kasus bencana alam seperti gempa bumi dengan menggunakan protokol *routing* sebagai komunikasi antar *node*. Salah satu contoh protokol *routing* yaitu *Dynamic Source Routing* (DSR).

DSR merupakan protokol *routing* yang mulai bekerja saat terjadi permintaan dari *source node* untuk mencari tahu jalur yang digunakan dalam mengirim pesan ke *node* tujuan. Dalam melakukan tugasnya, DSR menyebarkan pesan *Route Request* (RREQ) ke semua *node* tetangga dari *source node*. Setelah *node* tujuan menerima RREQ, maka dilakukan kembali pengiriman *Route Reply* (RREP) yang menandakan jalur dari *source node* ke *destination node* telah ditemukan. Untuk mencari jalur yang dituju, terlebih dahulu melakukan mekanisme *flooding*. Mekanisme *flooding* merupakan proses membanjiri (*flooding*) paket RREQ ke seluruh jaringan pada fase *route discovery*. Ketika terjadi mekanisme *flooding* yang berlebihan, maka dapat menyebabkan meningkatnya *routing overhead* pada protokol *routing* DSR [1].

Untuk mengatasi terjadinya peningkatan *routing overhead*, maka perlu dilakukan pengelompokan terh-adap *node*. Pengelompokan (*clustering*) merupakan solusi terbaik untuk mengurangi paket-paket *routing flooding* yang dapat meningkatkan *routing overhead* di jaringan MANET [2].

Clustering mengacu pada teknik di mana MANET dibagi menjadi kelompok virtual yang berbeda. Hal ini memungkinkan jaringan menjadi mudah dikelola yang memungkinkan koneksi cepat dan juga pengelolaan *routing* dan topologi MANET menjadi lebih baik [3]. Algoritma yang digunakan pada penelitian yang

dilakukan adalah *Algoritma Control Overhead Energy Efficient Cluster* (COEEC). Algoritma COEEC bekerja dengan membandingkan jumlah energi yang dimiliki setiap *node*. *Node* yang memiliki jumlah energi maksimum akan menjadi *cluster head* dan *node-node* yang berada satu *range* dengan *cluster head* menjadi *cluster member* [4].

Penelitian yang diusulkan untuk mengoptimalkan kinerja protokol *routing* DSR dengan memanfaatkan algoritma *Control Overhead Energy Efficient Cluster* (COEEC) untuk meminimalkan *routing overhead (RO)* ketika terjadi mekanisme *flooding* yang berlebihan pada fase *route discovery*. Pengukuran kinerja protokol *routing* dilihat dari *RO*, *throughput* , *delay*, *packet delivery ratio (PDR),* dan *average residual energy* pada skenario yang berbeda.

2. TINJAUAN PUSTAKA

Penelitian ini melakukan pengujian dengan menggunakan protokol EAODV. Protokol yang diusulkan yaitu AODV menggunakan metode *cluster*ing dengan menambahkan *black hole attack* yang bertujuan untuk mengetahui efisiensi energi pada protokol *routing Ad hoc On-Demand Distance Vector* (AODV) menggunakan metode *cluster*ing. Dari pengujian yang telah dilakukan didapatkan kesimpulan bahwa protokol *routing* EAODV lebih besar dalam konsumsi energi dari pada protokol *routing* yang diusulkan. Untuk performansi kinerja, protokol *routing* AODV dengan *cluster*ing, memiliki kualitas kinerja yang lebih baik dengan meningkatnya nilai *throughput* dan *PDR*, serta *delay* dan konsumsi energi yang lebih sedikit. [5]

Pada penelitian ini dilakukan perbandingan protokol AODV standar dengan protokol *routing* AODV dengan mengusulkan penambahan *cluster*ing atau disebut Cluster-AODV yang bertujuan untuk mengurangi tingkat *overhead* pada protokol *routing* Ad *hoc On-Demand Distance Vector* (AODV) dengan mengusulkan penambahan *cluster*ing. Dari hasil pengujian didapat kesimpulan bahwa penambahan metode *cluster*ing dapat mengurangi *routing overhead* karena sifatnya yang terlokalisasi, terdistribusi dan reaktif. [6]

Pada penelitian ini dilakukan perbandingan protokol *Ad hoc On-Demand Distance Vector* (AODV) yang menggunakan metode *safe route* berdasarkan nomor urut dan protokol *routing* AODV yang menggunakan *cluster*ing yang bertujuan untuk mendeteksi adanya *black hole attack* dan menghapus *node* jahat yang dapat meningkatkan *packet drop ratio*. Dari hasil simulasi yang telah dilakukan metode yang diusulkan berhasil meningkatkan tingkat deteksi *node* jahat yang mengarah pada peningkatan kinerja jaringan, dan sekaligus terjadi penurunan tingkat *packet drop ratio*. [7]

Pada penelitian terkait, dilakukan perbandingan kinerja dua protokol yaitu *Ad Hoc On-Demand Multipath Distance Vector* (AOMDV) dan *AOMDV-Low Energy Adaptive Clustering Hierarchy* (LEACH) yang bertujuan untuk mengukur efisiensi energi untuk memperpanjang umur jaringan. Dalam penelitian ini mengusulkan pendekatan berbasis *cluster* untuk menyediakan komunikasi yang efisien. Hasil yang di dapat untuk parameter *routing overhead* protokol LEACH dengan AOMDV dapat meminimalisasi paket-paket *routing flooding* dalam jaringan dinamis. Dalam jaringan ini kinerja *throughput* dari protokol *routing* LEACH jauh lebih baik dari pada protokol *routing* standar. Alasan utama yaitu komunikasi yang terjadi antar *node* lebih baik dalam *cluster*. Kesimpulan akhir yang didapat, konsumsi energi protokol *routing* AOMDV-LEACH lebih baik dari AOMDV standar. [8]

Pada penelitian ini dilakukan pengujian terhadap protokol *routing Cluster Based Routing Protocol* (CBRP) dengan algoritma COEEC. Pengujian yang dilakukan mencangkup parameter *normalized routing overhead*, *throughput* , *average energy consumption*, dan *end to end delay*. Parameter pengujian menunjukkan bahwa penambahan algoritma COEEC pada CBRP lebih baik dibandingkan dengan protokol *routing* CBRP standar dalam memilih *cluster head*. Hasil dari penelitian yang sudah dilakukan menunjukkan bahwa algoritma COEEC memberikan kinerja yang lebih baik. [4]

3. METODE PENELITIAN

3.1 Alur Penelitian

3.1.1 Studi Literatur

Peneliti melakukan riset terkait topik dalam penelitian yang dilakukan. Hal tersebut sebagai dasar dalam melakukan penelitian. Sumber-sumber referensi untuk topik penelitian didapatkan melalui jurnal-jurnal dalam penelitian sebelumnya, buku penunjang yang terkait dalam penelitian dan berbagai sumber dari internet.

3.1.2 Menerapkan Algoritma COEEC pada protokol DSR

Pada bagian ini, peneliti melakukan proses penerapan algoritma COEEC pada protokol *routing* DSR dengan menambahkan *cluster formation* untuk melakukan *cluster*ing pada *node* dan *cluster*

maintenance untuk melakukan pemeliharan *cluster* yang telah dibentuk.

3.1.3 Perancangan Skenario Simulasi CDSR dan DSR

Pada tahap ini, peneliti melakukan proses penerapan *script* simulasi pada dua protokol yaitu protokol DSR standar dan protokol CDSR untuk mengetahui pengaruh *cluster*ing terhadap kinerja protokol DSR. *Script* simulasi dibuat untuk mengatur formasi *cluster*, jumlah *node*, luas area simulasi, dan waktu simulasi yang digunakan.

3.1.4 Menjalankan Simulasi

Pada tahap ini, peneliti melakukan proses menjalankan simulasi untuk mengetahui perubahan kinerja yang terjadi pada protokol *routing* DSR standar dengan protokol *routing* CDSR dengan mendapatkan *output* berupa *file trace* dari hasil *running*.

3.1.5 Melakukan *Filter File Trace*

Dalam simulasi yang dilakukan didapatkan *output* berupa *file trace* yang kemudian dilakukan *filter* untuk memperoleh hasil kinerja dari masing-masing protokol *routing*. Dalam melakukan *filter terhadap file trace* maka dibutuhkan *file* AWK yang berisi *script* untuk *filter* yang sesuai dengan parameter uji yang digunakan yaitu :

a. *Routing overhead* : perbandingan antara banyaknya paket *routing* dalam waktu tertentu terhadap banyaknya paket yang diterima. *Routing overhead* dihitung menggunakan persamaan 1.

$$Routing\ Overhead = \frac{\sum paket\ routing}{\sum paket\ diterima} \quad (1)$$

b. *Throughput* : kecepatan (rate) transfer paket data yang diterima selama waktu yang telah ditentukan [9]. *Throughput* dihitung menggunakan persamaan 2 .

$$Throughput = \frac{\sum paket\ data\ diterima}{\sum waktu}\ \text{Kbps} \quad (2)$$

c. *Delay* : waktu yang dibutuhkan saat terjadi pengiriman paket yang melewati suatu rute dari *node* sumber menuju *node* tujuan [10]. *Average end-to-end delay* dihitung menggunakan persamaan 3.

$$AVG\ Delay = \frac{\sum_{j\leq dikirim}^{i=0} t_{terima[i]} - t_{dikirim[i]}}{paket\ yang\ terkirim}\ \text{ms} \quad (3)$$

d. PDR : perbandingan antara paket yang diterima oleh *node* tujuan dengan total paket yang dikirim oleh *node* asal [9]. PDR dihitung menggunakan persamaan 4.

$$\text{PDR} = \frac{\sum paket\ diterima}{\sum paket\ dikirim} \text{ x } 100\% \quad (4)$$

e. *Average Residual Energy* (ARE) *:* rata-rata sisa energi yang dhasilkan dari energi awal semua *node* (EA) dikurangi dengan total energi yang dikonsumsi (EK) dibagi dengan jumlah *node* (N). ARE dapat dihitung menggunakan persamaan 5.

$$\text{ARE} = \frac{\sum_{i-0}^{N} EA - \sum_{i=0}^{N} EK}{N}\ joules \quad (5)$$

3.1.6 Analisis Hasil *Filter*

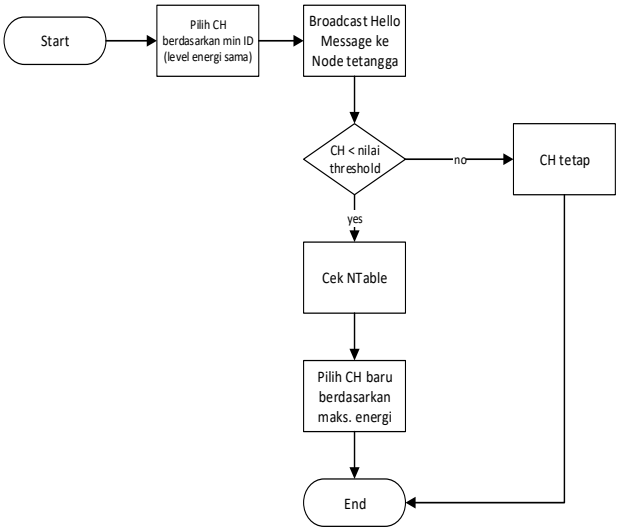
Pada bagian ini, peneliti melakukan analisis terhadap hasil *filter* yang didapatkan untuk dijadikan sebagai suatu kesimpulan.

3.1.7 Pembuatan Laporan Akhir

Pada tahap ini dilakukan dokumentasi oleh peneliti secara menyeluruh melalui pembuatan laporan akhir. Hasil dari pembuatan laporan akhir ini dapat dijadikan sebagai acuan untuk penelitian selanjutnya.

3.2 Mekanisme *Control Overhead Energy Efficient Cluster* (COOEC)

Clustering merupakan proses membagi jaringan menjadi substruktur yang di kelompokkan ke dalam kelompok geografis terdistribusi yang berbeda. *Clustering* berfokus dalam membagi jaringan menjadi *cluster* dan memilih *node* tertentu sebagai *cluster head*. Dalam penelitian ini pemilihan *cluster head* menggunakan *Control Overhead Energy Efficient Cluster* (COEEC). Pemilihan *cluster head* dengan COOEC dilakukan untuk mengurangi tingkat *overhead* saat terjadi paket *flooding* yang berlebihan. *Pada Control Overhead Energy Efficient Cluster* terdapat dua fase yaitu *cluster formation* dan *cluster maintenance* [4].

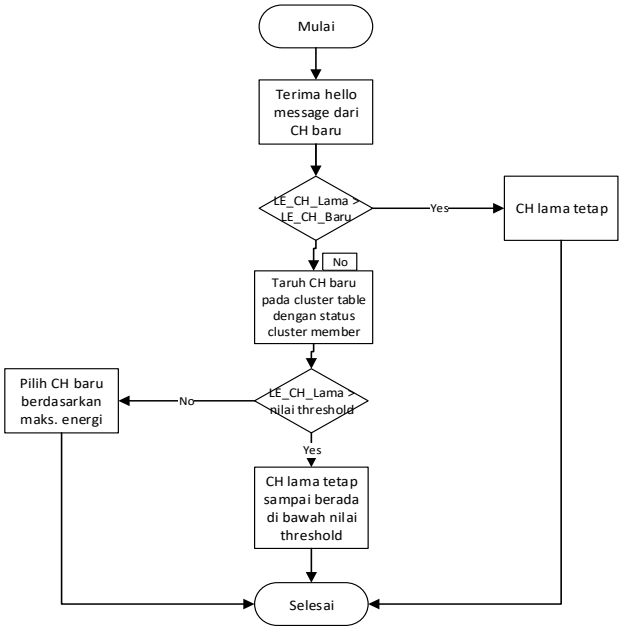


Gambar 1. *Cluster formation* COEEC

cluster formation merupakan proses awal dalam membentuk *cluster* dan melakukan pemilihan *cluster head*. Di dalam MANET semua *node* memiliki *neighbour table* yang memberikan informasi mengenai semua *node* yang terhubung ke *node* tersebut. Pada tahap ini setiap *node* melakukan *broadcast Hello Message* ke tetangganya. Setelah menerima *Hello Message*, setiap *node* memperbaharui informasi *Neighbour Table* (NTable). Pada COEEC pemilihan *cluster head* baru berdasarkan energi maksimum.

Pada Gambar 1 merupakan alur *cluster formation* pada algoritma COEEC. Tahap awal pemilihan *cluster head* dengan menggunakan ID terkecil sebagai *cluster head*. Selang beberapa waktu, pemilihan *cluster head* baru dilakukan dengan membandingkan jumlah energy dari setiap *node* dengan mengacu pada NTable. *Node* yang memiliki jumlah energy maksimum akan terpilih sebagai *cluster head* yang baru.

Selanjutnya *cluster maintenance* dilakukan sebagai upaya untuk menghindari pemilihan *cluster head* saat level energi berada pada ambang batas minimum (*threshold*).

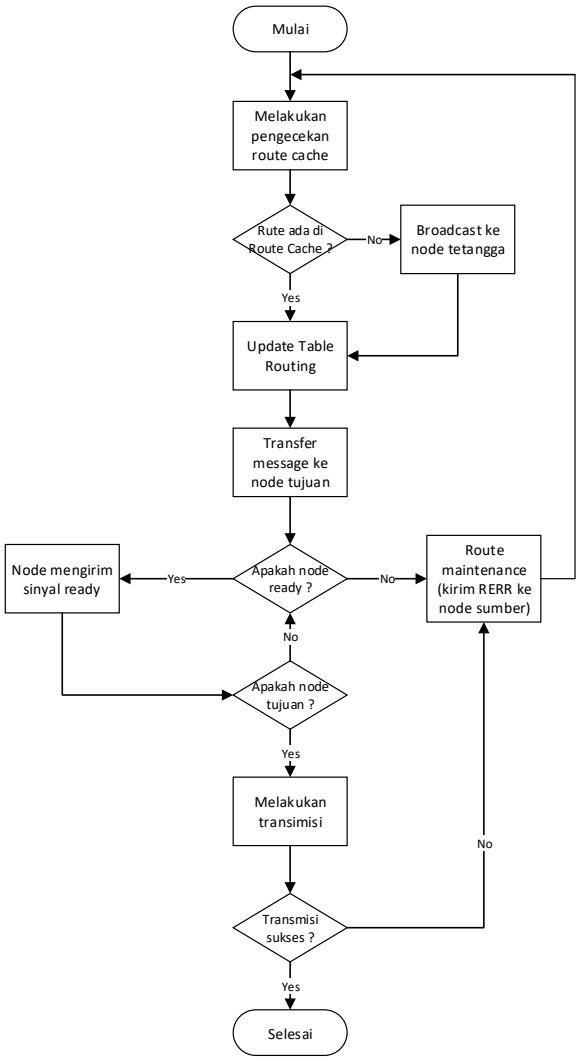


Gambar 2. *Cluster maintenace* COEEC

Cluster maintenance dilakukan saat terdapat *node* baru masuk ke dalam *cluster*. Saat terdapat *node* baru masuk dalam *cluster* maka dilihat tingkat energinya. Jika tingkat energinya lebih kecil dari *cluster head* maka akan menjadi *node* biasa pada *cluster*, tetapi jika tingkat energinya lebih besar dari *cluster head* maka terlebih dahulu dilakukan pengecekan terhadap *cluster head* dengan membandingkan tingkat energi *cluster head* dengan nilai *threshold*. Nilai *threshold* berfungsi untuk menjaga *cluster head* tetap hidup sampai nilai energinya turun pada ambang batas yang telah ditentukan. Jika nilai *threshold* lebih besar dari tingkat energi *cluster head* awal maka *node* baru akan menjadi *cluster head* yang baru dan jika nilai *threshold* lebih kecil maka tidak dilakukan pemilihan *cluster head* yang baru. *Block diagram cluster maintenance* sebagai berikut :

3.3 Proses *Route Discovery* dan *Route Maintenance* DSR

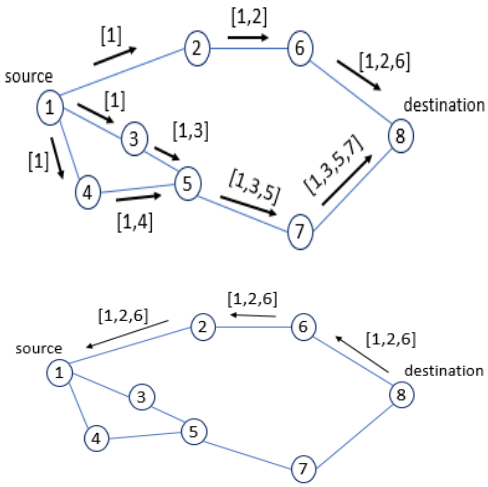
Alur proses *route discovery* dan *route maintenance* protokol DSR sebagai berikut :



Gambar 3. Diagram alur protokol *routing* DSR

DSR merupakan suatu protokol *routing* bersifat reaktif ketika terjadi permintaan rute dari *node* asal ke *node* tujuan. DSR memakai dua mekanisme yang digunakan untuk memastikan rute tetap stabil, yaitu *route discovery* dan *route maintenance*. Pada DSR juga terdapat *route cache* yang berfungsi sebagai penyimpan informasi setiap rute yang telah dituju, sehingga saat akan melakukan pengiriman message ke *node* tujuan terlebih dahulu dilakukan pengecekan *route cache*, jika rute yang diinginkan tidak terdapat pada *route cache* maka akan dilakukan *route discovery*. *Route discovery* merupakan proses pencarian rute.

Dari Gambar 3 dapat diketahui bahwa ketika *node* sumber memerlukan rute untuk mengirim paket data ke *node* tujuan, *node* sumber akan memeriksa *route cache* terlebih dahulu, apakah rute yang diinginkan untuk mengirimkan paket data ke *node* tujuan tersedia. Jika tersedia, maka dilakukan *update table routing* dengan mencocokan rute yang sudah tersedia. Jika tidak ada maka akan dilakukan pencarian rute dengan cara melakukan *broadcast route request* (RREQ) ke setiap *node* tetangga dalam satu *range*. Setiap *node* yang dikunjungi terlebih dahulu akan diperiksa sebelum melanjutkan *broadcast* RREQ, apakah *node* siap mengirimkan paket data. Jika *node* yang dikunjungi mengalami kegagalan dalam melakukan pengiriman paket data, maka akan dilakukan *route maintenance* dengan mengirimkan *route error* (RERR) ke *node* sumber dan menghapus rute. Jika *node* yang dikunjungi merupakan *node* tujuan, maka akan dilakukan pengiriman *route reply* (RREP) dan memulai transmisi. Jika bukan *node* tujuan, maka akan dicek kembali apakah *node* siap dalam mengirimkan paket. Jika sukses melakukan transmisi, maka pencarian rute sudah berakhir, sebaliknya jika gagal maka akan dilakukan *route maintenance* dan kembali melakukan *route discovery*. Proses *route discovery* dapat dilihat pada Gambar 4.

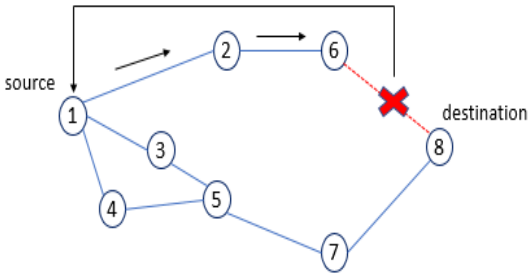


Gambar 4. Proses *route discovery* protokol *routing* DSR

Gambar 4 menunjukkan proses *route discovery* yang terjadi pada protokol *routing* DSR. *Node* 1 akan melakukan pengiriman paket data ke *node* 8, karena pada tabel *routing node* 8 tidak terdapat rute cadangan ke *node* D, maka *node* 1 harus mem-broadcast paket RREQ ke *node-node* tetangganya untuk menemukan rute menuju *node* 8. *Node* 1 melakukan *broadcast* paket RREQ ke *node* 2, 3, dan 4, kemudian *node* 2, 3, dan 4 juga akan mem-broadcast paket RREQ ke *node-node* tetangga mereka masing-masing hingga sampai di *node* 8. *Node-node* yang dikunjungi akan menyimpan informasi dari *node* sebelumnya sampai berhasil ke *node* tujuan yaitu *node* 8. Selanjutnya, *node* D akan mengirimkan paket RREP sebagai balasan dari paket RREQ ke *node* S secara *unicast* melalui masing-masing rute. Dengan demikian diperoleh 3 rute, yaitu 8-6-2-1, 8-7-3-5-1, dan 8-7-5-4-1. Rute yang dipilih

adalah rute yang memiliki jumlah *hop* terkecil, yakni rute 1-2-6-8. Sedangkan, rute yang tidak terpilih akan dijadikan rute cadangan sebagai solusi ketika terjadinya perubahan topologi dan dilakukannya *route maintenance*.

Route maintenance merupakan proses pemeliharaan rute. Suatu *node* akan melakukan pengiriman paket *route request* (RERR) ke *node* sumber. Jika *link* yang menghubungkan pengiriman paket ke *node* tetangganya terputus, maka akan menyebabkan kerusakan rute. Jika terjadi kerusakan rute, maka rute cadangan yang tersimpan pada tabel *routing* dapat digunakan untuk melakukan proses pengiriman paket tanpa harus melakukan proses *route discovery*. Namun, jika semua rute cadangan rusak, maka harus dilakukan proses *route discovery* ulang.

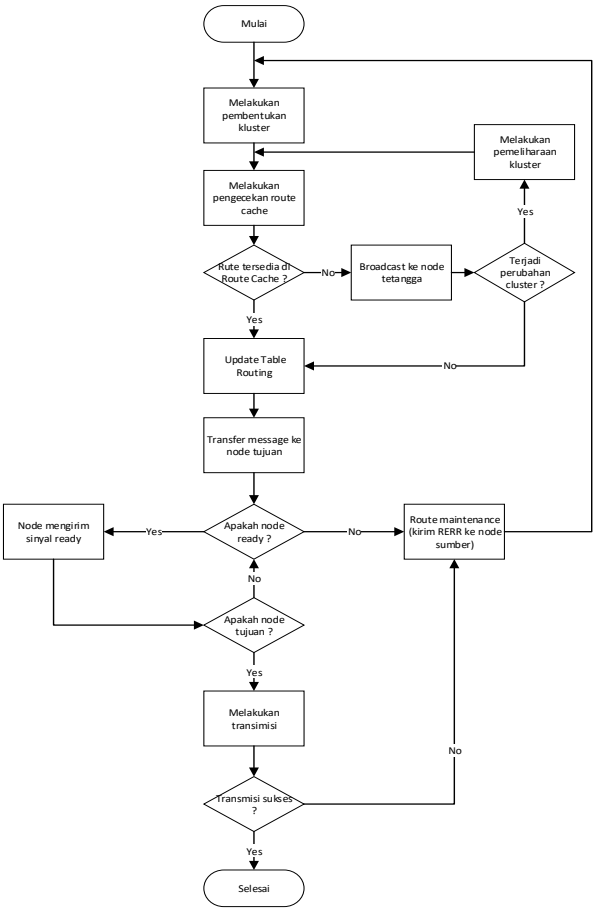


Gambar 5. Proses *route maintenance* protokol *routing* DSR

Gambar 5 menunjukkan terjadinya kerusakan *link* yang menghubungkan *node* 7 dan *node* 8, sehingga *node* 7 harus mengirimkan paket *route error* (RRER) ke *node* 1. Setelah *node* 1 menerima paket *route error* (RRER), maka proses komunikasi akan dialihkan menggunakan rute cadangan yang tersimpan pada tabel *routing node* 1. Jika tidak terdapat rute cadangan, maka *node* 1 harus melakukan *route discovery* ulang.

3.4 Modifikasi protokol *routing* DSR menggunakan algoritma COEEC

Alur modifikasi protokol *routing* DSR menggunakan algoritma COEEC sebagai berikut :



Gambar 6. Modifikasi DSR dengan algoritma COEEC

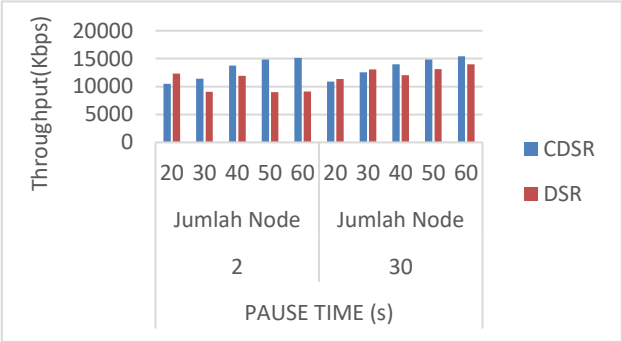
Gambar 6 menunjukkan modifikasi dari protokol *routing* DSR dengan penambahan algoritma COEEC yang berupa *cluster formation* dan *cluster maintenance*. *cluster formation* ditempatkan diawal untuk melakukan pembentukan *cluster* dan memilih *cluster head*. Setelah melakukan pemilihan *cluster head* maka dapat melakukan langkah-langkah pengiriman data ke *node* tujuan. Kemudian *cluster maintenance* ditempatkan setelah melakukan broadcast ke setiap *node* tetangga. Penempatan tersebut untuk mengantisipasi jika terjadi perubahan dalam *cluster* seperti terdapat *node* baru yang masuk atau terdapat *node* yang keluar dari *cluster*, sehingga *cluster* akan tetap dalam keadaan stabil. Jika tidak terjadi perubahan dalam *cluster* maka dilanjutkan ke langkah-langkah pengiriman *node*.

4. HASIL DAN PEMBAHASAN

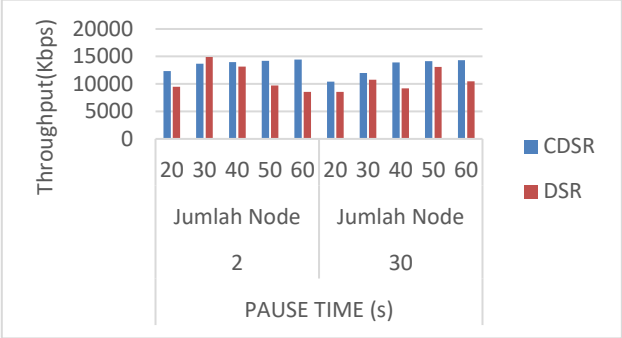
4.1 Analisis *Throughput*

Throughput merupakan ukuran kecepatan dari transfer paket data yang diterima oleh *node* tujuan. Pengukuran kinerja terhadap *throughput* yang baik adalah kecepatan transfer paket yang tinggi dari *node*

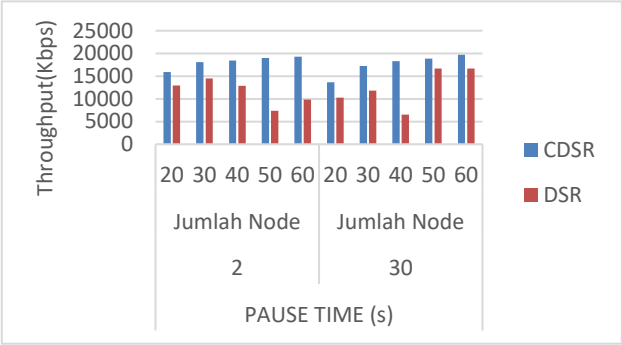
sumber ke *node* tujuan. *Throughput* akan berpengaruh ketika paket RREQ yang di kirimkan melalui banyak *node* atau sedikit untuk sampai ke *node* tujuan. Ketika banyak *node* yang dilewati saat pengiriman paket RREQ maka dapat menyebabkan penurunan kecepatan, sebaliknya jika *node* yang di lewati lebih sedikit maka dapat meningkatkan nilai *throughput*.



(a)



(b)



(c)

Gambar 7. Perbandingan nilai *throughput* dengan 3 kali percobaan. (a) percobaan 1; (b) percobaan 2; (c) percobaan 3.

Berdasarkan grafik Gambar 7, pada percobaan pertama dengan *pause time* 2 protokol *routing* CDSR terus mengalami peningkatan nilai *throughput* seiring dengan bertambahnya jumlah *node* yang digunakan. Protokol *routing* DSR hanya unggul ketika jumlah *node* 20 dan 30. Hal tersebut disebabkan karena pada jumlah *node* yang lebih sedikit dapat menyebabkan tidak stabilnya pengiriman paket yang dipengaruhi oleh jumlah *hop* yang dilewati saat pengiriman paket RREQ dilakukan. Rata-rata nilai *throughput* yang dihasilkan oleh protokol *routing* CDSR sebesar 13124,13 Kbps dan protokol *routing* DSR sebesar 10284,58 Kbps pada *pause time* 2. Kemudian ketika menggunakan *pause time* 30, rata-rata kecepatan pengiriman paket yang dilakukan oleh protokol *routing* CDSR masih lebih baik dengan nilai yang didapatkan sebesar 13544,19 Kbps dan protokol *routing* DSR sebesar 12709,26 Kbps.

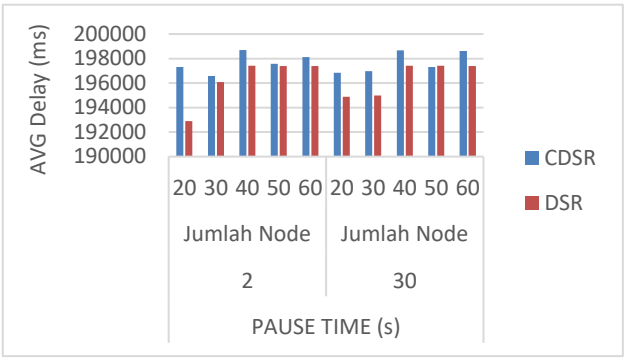
Pada percobaan kedua, protokol *routing* CDSR memiliki kualitas *throughput* yang lebih baik dari protokol *routing* DSR. Protokol *routing* CDSR memiliki nilai rata-rata kecepatan pengiriman paket sebesar 13731,39 Kbps dan protokol *routing* DSR sebesar 11184,64 Kbps pada *pause time* 2. Protokol *routing* DSR hanya unggul ketika jumlah *node* 30, hal ini disebabkan karena pada jumlah *node* tersebut DSR masih mampu dengan baik untuk melakukan pengiriman paket RREQ karena *hop node* yang dilewati masih sedikit. Pada *pause time* 30, protokol *routing* CDSR memiliki kualitas yang lebih baik dibandingkan dengan protokol *routing* DSR pada jumlah *node* 20, 30, 40, 50, dan 60. Nilai rata-rata yang dihasilkan oleh protokol *routing* CDSR mencapai 12977,41 Kbps dan protokol *routing* DSR mencapai 10415,36 Kbps.

Pada percobaan ketiga, protokol *routing* CDSR masih lebih baik dari protokol *routing* DSR pada jumlah *node* 20, 30, 40, 50, dan 60. Protokol *routing* CDSR pada *pause time* 2 memiliki nilai rata-rata yang mencapai 18128,47 Kbps dan protokol *routing* DSR mencapai 11482,40 Kbps. Dari grafik yang disajikan, pada *pause time* 2 protokol *routing* CDSR selalu unggul dari DSR. Selanjutnya pada *pause time* 30, protokol *routing* CDSR masih lebih baik dari protokol *routing* DSR. Nilai rata-rata yang didapatkan protokol *routing* CDSR dapat mencapai 17540,11 Kbps dan protokol *routing* DSR hanya mencapai 12375,18 Kbps.

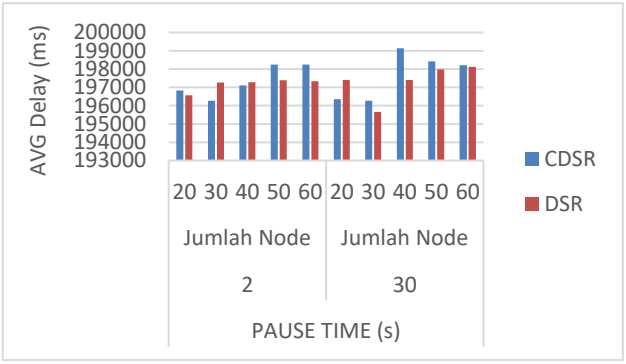
Dari sepuluh percobaan yang telah dilakukan dengan menggunakan jumlah *node* yang berbeda. Protokol *routing* CDSR lebih bagus dalam hal meningkatkan nilai *throughput* dari pada protokol *routing* DSR. Semakin banyak *node* yang terlibat, protokol *routing* CDSR mampu mengatasi penurunan nilai *throughput* . Hal tersebut terjadi karena protokol *routing* CDSR melakukan *cluster*ing terlebih dahulu yang berakibat berkurangnya proses pengiriman RREQ ke semua *node* yang dapat menurunkan kualitas *throughput* . Selanjutnya protokol *routing* DSR dapat unggul pada *node* yang lebih sedikit.

4.2 Analisis *Average End-to-End Delay*

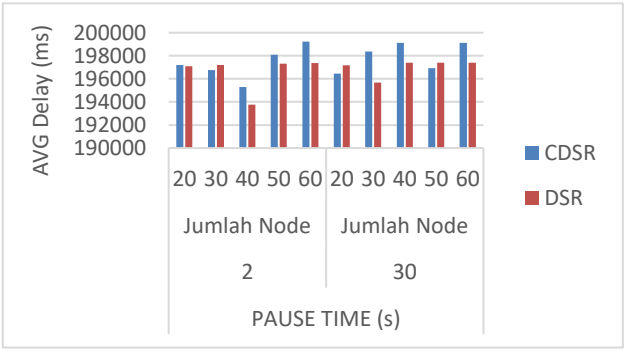
Average end to end delay merupakan rata-rata jumlah waktu yang didapatkan untuk sebuah paket ke *node* tujuan. Pengukuran kinerja *average end to end delay* yang baik adalah waktu mengirimkan paket lebih sedikit.



(a)



(b)



(c)

Gambar 8. Perbandingan kualitas *average end to end delay*. (a) percobaan 1; (b) percobaan 2;(c) percobaan 3.

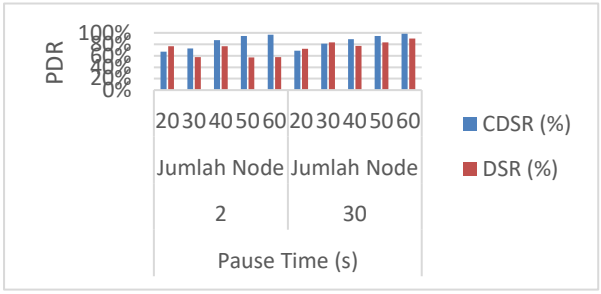
Berdasarkan grafik pada Gambar 8, protokol *routing* CDSR kurang baik dalam masalah *average end to end delay*. Dari percobaan pertama yang telah dilakukan, nilai rata-rata *average end to end delay* dari protokol *routing* CDSR mencapai 197652,4 ms dan protokol *routing* DSR mencapai 192230,2 ms pada *pause time* 2. Selanjutnya pada *pause time* 30 , nilai rata-rata dari protokol *routing* CDSR mencapai 197679,8 ms dan protokol *routing* DSR mencapai 196420 ms. Sesuai dengan konsep *average end to end delay* bahwa nilai *average end to end delay* yang lebih sedikit menunjukkan waktu pengiriman paket yang terbaik. Sehingga protokol *routing* DSR unggul dalam kinerja *average end to end delay*.

Pada percobaan kedua dan ketiga , protokol *routing* DSR masih lebih baik dalam hal waktu untuk pengiriman paket. Pada percobaan kedua, nilai rata-rata yang didapatkan DSR mencapai 197164,4 ms dan CDSR mencapai 197338,6 ms pada *pause time* 2. Selanjutnya pada *pause time* 30, nilai rata-rata yang didapatkan protokol *routing* DSR mencapai 197314,4 s dan CDSR mencapai 197678,8 ms. Untuk percobaan ketiga, protokol *routing* DSR tetap unggul dari protokol *routing* CDSR. Nilai rata-rata dari protokol *routing* DSR mencapai 196549,6 ms dan protokol *routing* CDSR mendapat nilai 197313,8 ms pada *pause time* 2. Selanjutnya pada *pause time* 30, nilai rata-rata protokol *routing* DSR mencapai 197012 ms dan protokol *routing* CDSR mencapai nilai 198000,6 ms.

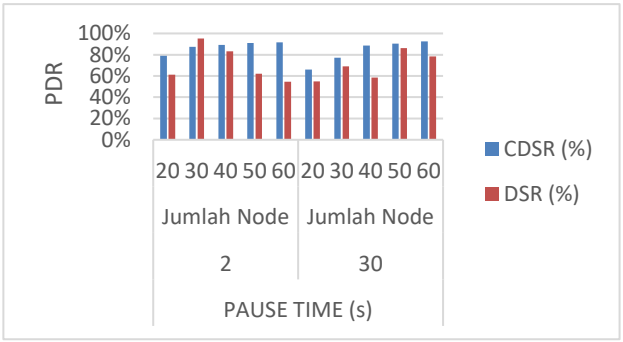
Dari sepuluh percobaan yang telah dilakukan, DSR memiliki nilai *delay* yang lebih baik dari protokol *routing* CDSR. Hal ini disebabkan karena protokol *routing* CDSR melakukan proses *cluster*ing terlebih dahulu sehingga menghambat waktu dalam proses pengiriman paket RREQ. Sedangkan, protokol *routing* DSR langsung melakukan pengiriman paket RREQ ke *node-node* tetangga.

4.3 Analisis *Packet Delivery Ratio*

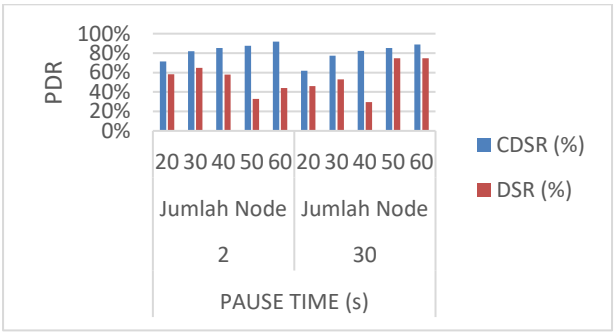
Pada subbab ini akan dijelaskan analisis hasil PDR dari penelitian ini. Gambar 9 merupakan hasil grafiknya Berdasarkan hasil grafik Gambar 9, pada percobaan pertama protokol *routing* CDSR lebih baik dalam jumlah penerimaan paket oleh *node* tujuan. Nilai rata-rata yang didapatkan protokol CDSR mencapai 84%, sedangkan protokol DSR hanya mencapai 65% pada pause time 2. Untuk pause time 30, nilai rata-rata dari protokol routing CDSR mencapai 86%, sedangkan protokol DSR mencapai 81%.



(a)



(b)



(c)

Gambar 9. Perbandingan kualitas *packet delivery ratio*. (a) percobaan 1; (b) percobaan 2; .(c) percobaan 3.

Pada percobaan kedua, protokol *routing* CDSR lebih baik dari protokol *routing* DSR. Pada *pause time* 2, nilai rata-rata yang didapatkan oleh protokol *routing* CDSR mencapai 88%, sedangkan protokol *routing* DSR mencapai 71%. Pada *pause time* 30, protokol *routing* CDSR tetap unggul dengan nilai rata-rata mencapai 83% dan protokol *routing* DSR hanya mencapai 70%.

Pada percobaan ketiga, protokol *routing* CDSR masih tetap lebih baik dari protokol *routing* DSR. Nilai rata-rata yang didapatkan oleh protokol *routing* CDSR mencapai 84%, sedangkan protokol *routing* DSR hanya mencapai 52% pada *pause time* 2. Untuk *pause time* 30, nilai rata-rata yang didapatkna protokol *routing* CDSR mencapai 79% dan protokol DSR mencapai 56%.

Dari sepuluh percobaan yang telah dilakukan, Protokol *routing* CDSR berhasil menghasilkan kualitas *packet delivery ratio* yang diakibatkan oleh terjadinya proses *cluster*ing dalam pengiriman paket. Proses *cluster*ing yang dilakukan menyebabkan paket tidak banyak singgah ke *node* yang tidak seharusnya menjadi jalur untuk menuju ke *node* tujuan, melainkan kepala *cluster* dapat langsung berkomunikasi dengan kepala *cluster* yang lain untuk mengetahui *node* tujuan. Sehingga paket yang dikirimkan masih tetap lebih baik dibandingkan dengan protokol *routing* DSR yang melakukan pengiriman paket ke semua *node* tetangga sampai menemukan *node* tujuan.

4.4 Analisis *Routing Overhead*

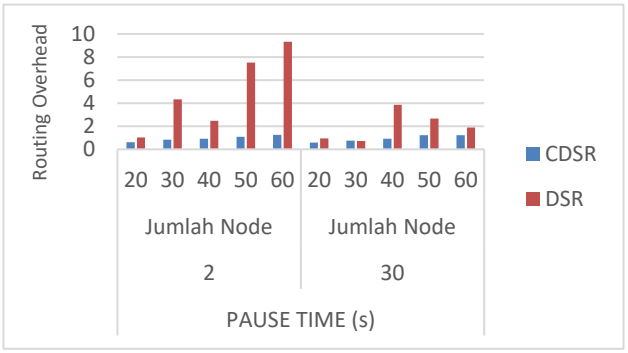
Kualitas kinerja dari *routing overhead* yang baik adalah nilai yang didapatkan oleh protokol *routing* terjadi pada titik minimum. Berikut grafik *Routing Overhead* dari hasil percobaan yang telah dilakukan.

Berdasarkan grafik Gambar 10, pada percobaan pertama kualitas *routing overhead* dari protokol *routing* CDSR lebih baik dari protokol *routing* DSR. Pada *pause time* 2, nilai rata-rata protokol *routing* CDSR mencapai 0,9384 dan protokol *routing* DSR mencapai 4,9328. Selanjutnya pada *pause time* 30, nilai rata-rata yang didapatkan protokol *routing* CDSR mencapai 0,935 dan protokol *routing* DSR mencapai 2,0166.

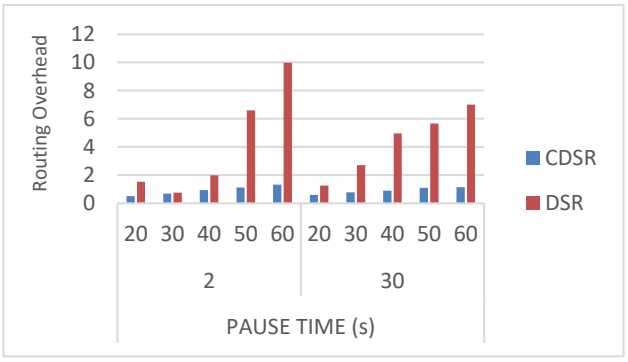
Pada percobaan kedua, kualitas *routing overhead* protokol CDSR lebih baik dari DSR. Nilai rata-rata protokol *routing* CDSR dapat mencapai 0,9138 dan protokol DSR mencapai 4,171 pada *pause time* 2. Selanjutnya pada *pause time* 30, nilai rata-rata protokol *routing* CDSR mencapai 0,9054 dan protokol *routing* DSR mencapai 4,3226.

Pada percobaan ketiga, protokol *routing* CDSR masih lebih baik dalam kualitas *routing overhead* dari protokol *routing* DSR. Nilai rata-rata protokol *routing* CDSR dapat mencapai 0,7668 dan protokol *routing* DSR mencapai 6,372 pada *pause time* 2. Selanjutnya pada *pause time* 30, nilai rata-rata protokol *routing* CDSR mencapai 0,7184 dan protokol *routing* DSR mencapai 3,9162.

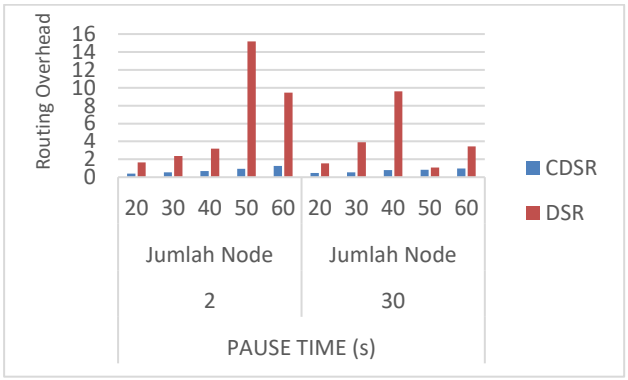
Dari sepuluh percobaan yang telah dilakukan, kualitas *routing overhead* protokol *routing* CDSR lebih baik dari protokol *routing* DSR. Protokol *routing* CDSR dapat mengurangi banyaknya rute yang dilewati untuk mengirim paket dari *node* sumber ke *node* tujuan. Sedangkan, protokol *routing* DSR banyak melakukan kunjungan ke *node* lain, sehingga banyak rute yang dilewati untuk sampai ke *node* tujuan yang dapat menyebabkan peningkatan nilai *routing overhead*.



(a)



(b)



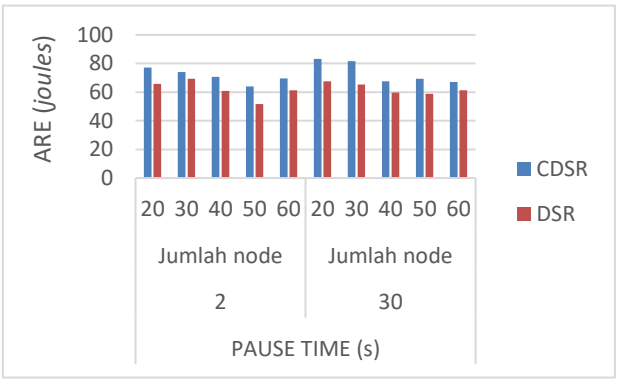
(c)

Gambar 10. Perbandingan kualitas *routing overhead*. (a) percobaan 1; (b) percobaan 2;(c) percobaan 3.

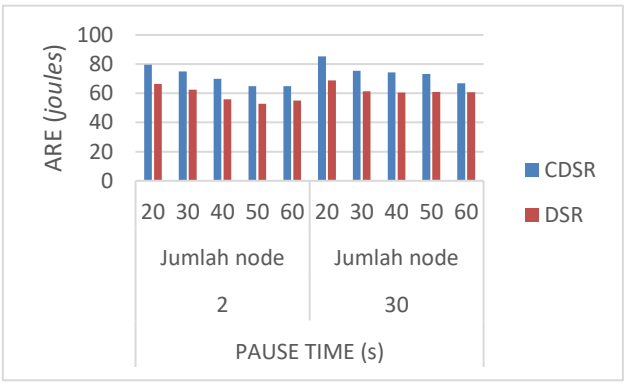
4.5 Analisis *Average Residual Energy*

Kualitas kinerja average residual energy yang baik adalah yang masih memiliki nilai rata-rata sisa energi terbesar. Hasil percobaan dapat dilihat pada grafik Gambar 11.

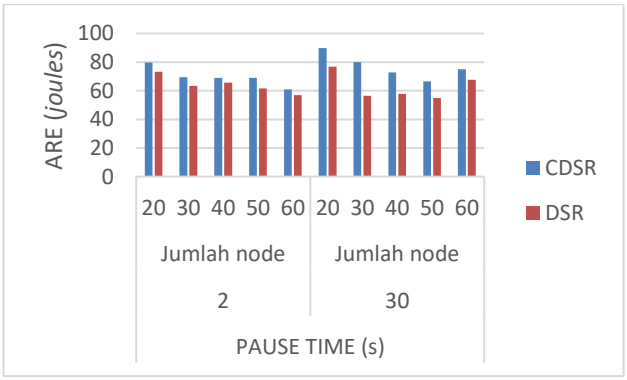
Berdasarkan hasil grafik Gambar 11 , pada percobaan 1 rata-rata sisa energi protokol routing sebesar 71 *joules* dan 73,8 *joules*, sedangkan DSR hanya mencapai 61,8 *joules* dan 62,6 *joules* untuk pause time 2 dan 30. Selanjutnya pada percobaan 7 rata-rata sisa energi yang didapat sebesar 70,8 *joules* dan 75 *joules*, sedangkan DSR hanya mencapai 58,5 *joules* dan 62,5 *joules* untuk *pause time* 2 dan 30. Dan yang terakhir pada percobaan 10 rata-rata sisa energi yang didapat sebesar 69,6 *joules* dan 76,8 *joules*, sedangkan DSR hanya mencapai 64,2 *joules* dan 62,7 *joules* untuk *pause time* 2 dan 30.



(a)



(b)



(c)

Gambar 11 Perbandingan kualitas average residual energy. (a) percobaan 1; (b) percobaan 7;(c) percobaan 10.

Dari 10 percobaan yang telah dilakukan, dari sisi kualitas *average residual energy* protokol routing CDSR lebih baik dari protokol routing DSR. Protokol *routing*

CDSR membagi *node* menjadi berkelompok dan memiliki kepala kluster yang mengkoordinasi *node-node* tetangga, sehingga energi yang akan dikeluarkan bisa lebih sedikit dan bisa menambah *lifetime* dari *node-node* yang lain. Berbeda dengan protokol *routing* DSR yang melakukan penyebaran paket pada semua *node*, sehingga membuat energi yang dikeluarkan lebih besar.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

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

1. Algoritma COEEC yang ditambahkan pada protokol *routing* CDSR berhasil meningkatkan kualitas kinerja *throughput* dari pada protokol *routing* DSR. Protokol *routing* CDSR mampu meningkatkan kinerja *throughput* sebesar 43% pada *pause time* 2 dan 31% pada *pause time* 30.
2. Protokol *routing* CDSR masih sedikit kurang baik dalam menekan peningkatan *average end to end delay* dari pada protokol *routing* DSR. Hal ini disebabkan karena tahap awal dari protokol *routing* CDSR yang melakukan *cluster formation* untuk menentukan kepala *cluster*, sehingga menambah delay yang terjadi pada protokol *routing* CDSR sebesar 0,005% dan 0,01% pada *pause time* 2 dan 30.
3. Kualitas kinerja *packet delivery ratio* dari protokol *routing* CDSR lebih baik dari pada protokol *routing* DSR. Hal tersebut disebabkan karena saat proses pengiriman paket RREQ, tidak banyak *node* yang terlibat akibat dari proses *cluster* yang dilakukan. Peningkatan kinerja *packet delivery ratio* pada protokol *routing* CDSR sebesar 41% untuk *pause time* 2 dan 27% pada *pause time* 30.
4. Kualitas kinerja *routing overhead* berhasil diminimalkan dengan penggunaan protokol *routing* CDSR. Penggunaan *cluster*ing dalam protokol *routing* CDSR dapat menekan mekanisme *flooding* yang berlebihan, sehingga bisa mendapatkan kualitas *routing overhead* yang bagus. Protokol *routing* CDSR dapat menekan peningkatan *routing overhead* sebesar 87% pada *pause time* 2 dan 81% untuk *pause time* 30.
5. Kualitas kinerja *average residual energy* pada protokol *routing* CDSR lebih baik dari protokol routing DSR dengan meningkatkan *residual energy* sebanyak 15% untuk *pause time* 2 dan 17 % untuk *pause time* 30.

5.2 Saran

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

1. Melakukan penelitian untuk memperbaiki kualitas kinerja *average end to end delay*.
2. Melakukan penelitian dengan menggunakan protokol *routing* yang berbeda.
3. Melakukan penelitian yang sama dengan mengubah parameter-parameter penelitian.

DAFTAR PUSTAKA

[1] A. Barve, A. Kini, O. Ekbote, and J. Abraham, "Optimization of DSR Routing Protocol in MANET using Passive Clustering," *IEEE 2nd Int. Conf. Commun. Control Intell. Syst.*, pp. 23–27, 2016.

[2] S. Bhadoria, L. Nishad, and L. Tongue, "Review on Cluster-head Election Mechanism for Clustering Based Routing in Mobile Ad-Hoc Network," *Int. J. Sci. Res. Publ.*, vol. 4, no. 7, pp. 1–3, 2014.

[3] P. Chatterjee and N. Agarwal, "Energy Aware, Scalable, K-Hop Based Cluster Formation In MANET," *Durgapur Natl. Inst. Technol.*, pp. 1–49, 2011.

[4] Ankita Singh Kushwah & Nitin Manjhi, "Control Overhead Energy Efficient Cluster Based Routing Algorithm in MANET," *Int. J. Comput. Sci. Eng. Inf. Technol. Res.*, vol. 4, no. 4, pp. 21–32, 2014.

[5] R. Singh and A. K. Dogra, "Performance Evaluation of Energy Efficient Modified AODV Using Clustering Method in MANET," *Int. J. Comput. Sci. Mob. Comput.*, vol. 4, no. 5, pp. 670–675, 2015.

[6] G. Kaushik and S. Goyal, "An Clustering based AODV approach for MANET," *Assoc. Comput. Electron. Electr. Eng.*, pp. 901–904, 2013.

[7] G. Singh and G. Singh, "Detection and Prevention Of Black Hole Using Clustering In MANET Using Ns2," *Int. J. Eng. Comput. Sci.*, vol. 3, no. 8, pp. 7420–7430, 2014.

[8] J. Chandel and N. Kaur, "Energy Consumption Optimization using Clustering in Mobile Ad-Hoc Network," *Int. J. Comput. Appl.*, vol. 168, no. 12, pp. 11–16, 2017.

[9] G. G. Sitompul, R. M. Negara, and D. D. Sanjoyo, "Analisis Performansi Protocol Routing AODV dan FSR (Studi Kasus : Skenario Jalan Raya)," *e-Proceeding Eng.*, vol. 5, no. 1, pp. 267–274, 2018.

[10] H. Akbar, A. H. Jatmika, and M. A. Albar, "Analisis Pengaruh Metode LET pada Protokol Routing Proaktif dan Reaktif di Jaringan MANET (Effect of the LET Method on Proactive and Reactive Routing Protocols in MANET)," *J. Comput. Sci. Informatics Eng.*, vol. 2, no. 2, pp. 120–126, 2018.