



**ANALISIS PERBANDINGAN KONSUMSI ENERGI DAN MASA
HIDUP JARINGAN PADA PROTOKOL LEACH, HEED, DAN
PEGASIS DI *WIRELESS SENSOR NETWORK***

SKRIPSI

Oleh

Muhammad Arief Wahyudi

NIM 141910201067

**JURUSAN TEKNIK ELEKTRO STRATA 1
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2018**



**ANALISIS PERBANDINGAN KONSUMSI ENERGI DAN MASA HIDUP
JARINGAN PADA PROTOKOL LEACH, HEED, DAN PEGASIS DI
WIRELESS SENSOR NETWORK**

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Strata 1 Teknik Elektro
dan mencapai gelar Sarjana Teknik

Oleh

Muhammad Arief Wahyudi

NIM 141910201067

**PROGRAM STUDI STRATA 1 TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2018**

PERSEMBAHAN

Alhamdulillah, atas berkat rahmat Allah Subhanahu Wa Ta'ala serta kasih sayang-Nya yang melimpah kepada seluruh alam serta kepada hamba-hamba-Nya, sehingga saya bisa menyelesaikan penelitian ini.

Akhirnya, saya persembahkan skripsi ini kepada.

1. Kedua orang tua, Ibu Sri Wachyuni dan Bapak Amin Taksir;
2. Kakak Ayu Purnamawati dan Adik Muhammad Kadarisman;
3. Guru-guru dan dosen sejak taman kanak-kanak hingga perguruan tinggi;
4. Almamater tercinta, Jurusan Teknik Elektro Universitas Jember;
5. Serta seluruh teman-teman yang penulis kenal dan teman-teman yang membaca skripsi ini.

MOTO

Sesungguhnya Allah tidak mengubah nasib suatu kaum kecuali kaum itu sendiri yang mengubah apa-apa yang pada diri mereka.

(terjemahan QS. Ar Ra'du ayat 11)*)

Satu-satunya cara melakukan sebuah pekerjaan yang luar biasa adalah dengan mencintai apa yang saat ini tengah dikerjakan.

(Steve Jobs)

Mimpi boleh melangit, tapi kaki harus membumi.

(Ridwan Kamil)

*) Departemen Agama Republik Indonesia. 1998. *Al Qur'an dan Terjemahannya*. Semarang: PT Kumudasmoro Grafindo

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Muhammad Arief Wahyudi

NIM : 141910201067

Menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul “Analisis Perbandingan Konsumsi Energi dan Masa Hidup Jaringan pada Protokol LEACH, HEED, dan PEGASIS di *Wireless Sensor Network*” adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab penuh atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, 25 April 2018

Yang menyatakan

Muhammad Arief Wahyudi
NIM 141910201067

SKRIPSI

**ANALISIS PERBANDINGAN KONSUMSI ENERGI DAN MASA HIDUP
JARINGAN PADA PROTOKOL LEACH, HEED, DAN PEGASIS
DI WIRELESS SENSOR NETWORK**

Oleh

Muhammad Arief Wahyudi
NIM 141910201067

Pembimbing :

Dosen Pembimbing Utama : Widya Cahyadi, S.T., M.T.

Dosen Pembimbing Anggota : Catur Suko Sarwono, S.T., M.Si.

PENGESAHAN

Skripsi berjudul "Analisis Perbandingan Konsumsi Energi dan Masa Hidup Jaringan pada Protokol LEACH, HEED, dan PEGASIS di *Wireless Sensor Network*" telah diuji dan disahkan oleh Fakultas Teknik Universitas Jember pada :

Hari : Rabu
Tanggal : 23 Mei 2018
Tempat : Fakultas Teknik Universitas Jember

Tim penguji,

Ketua,

Sekretaris,

Widya Cahyadi, S.T., M.T.

NIP 198511102014041001

Catur Suko Sarwono, S.T., M.Si.

NIP 196801191997021001

Anggota I,

Anggota II,

Andrita Ceriana Eska, S.T., M.T.

NRP 760014640

Alfredo Bayu Satriya, S.T., M.T.

NIP 198905192015041001

Mengesahkan
Dekan Fakultas Teknik

Dr. Ir. Entin Hidayah, M.U.M.

NIP 196612151995032001

Analisis Perbandingan Konsumsi Energi dan Masa Hidup Jaringan pada Protokol
LEACH, HEED, dan PEGASIS di *Wireless Sensor Network*

Muhammad Arief Wahyudi

Jurusan Teknik Elektro, Fakultas Teknik, Universitas Jember

ABSTRAK

Wireless Sensor Network atau WSN adalah sekumpulan *node* yang berupa sensor yang membentuk sebuah jaringan secara *wireless*. WSN berfungsi untuk mengumpulkan data tersebar yang diperoleh dari sensor yang kemudian akan dikirimkan melalui komunikasi Ad-Hoc menuju *Base Station* untuk diolah lebih lanjut. Dalam pengimplementasiannya, WSN memiliki beberapa masalah yang terjadi. Salah satunya adalah konsumsi energi dan masa hidup *node* sensor, dimana *node* sensor menggunakan energi hanya dari baterai untuk operasinya. Padahal, pada pengaplikasiannya pengambilan data tidak dilakukan sekali. Sehingga, perlunya penghematan energi untuk mengurangi konsumsi energi dan jumlah energi yang cukup agar *node* sensor mempunyai masa hidup yang lebih lama. Salah satu solusi untuk menanggulangi masalah tersebut yaitu merancang protokol dari WSN agar *node* sensor dapat mengirimkan data ke *Base Station* dengan efektif. Dalam tugas akhir ini, dilakukan simulasi menggunakan tiga buah protokol, yaitu *Low Energi Adaptive Clustering Hierarchy* (LEACH), *Hybrid Energy Efficient Distribute* (HEED), dan *Power-Efficient Gathering in Sensor Information Systems* (PEGASIS). Protokol LEACH dan HEED menggunakan metode *clustering* sedangkan PEGASIS menggunakan metode rantai. Analisis yang dilakukan pada ketiga protokol yaitu konsumsi energi dan masa hidup jaringan dengan menggunakan parameter yang sama dari *Zigbee Xbee Pro* dan disimulasikan pada Matlab R2013a.

Kata Kunci: *WSN, Ad-Hoc, Low Energi Adaptive Clustering Hierarchy (LEACH), Hybrid Energy Efficient Distribute (HEED), Power-Efficient Gathering in Sensor Information Systems (PEGASIS)*

A Comparative Analysis of Energy Consumption and Network Lifetime of LEACH, HEED, and PEGASIS Protocols in Wireless Sensor Network

Muhammad Arief Wahyudi

Department of Electrical Engineering, Faculty of Engineering, University of Jember

ABSTRACT

Wireless Sensor Network or WSN is a collection of nodes in the form of sensors which establish a network wirelessly. The function of WSN is to collect scattered data that is obtained from sensors which will be transmitted via Ad-Hoc communication to Base Station for further processing. In implementing it, WSN has several problems that occur. One of the problems is the energy consumption and lifetime of the sensor node which the sensor node uses energy only from battery for its operation. In fact, on its application the data is not taken once. Thus, it needs energy savings to reduce energy consumption and the amount of enough energy to make the sensor nodes has a longer lifetime. One of solutions to overcome the problem is to design a protocol from WSN so that sensor nodes can transmit data to Base Station effectively. In this final project, simulation was done using three protocols, namely Low Energy Adaptive Clustering Hierarchy (LEACH), Hybrid Energy Efficient Distribute (HEED), and Power-Efficient Gathering in Sensor Information Systems (PEGASIS). The LEACH and HEED protocols used clustering method while PEGASIS used chain method. The analysis performed on these three protocols was energy consumption and network lifespan using the same parameters of Zigbee Xbee Pro and simulated on Matlab R2013a.

Keywords: *WSN, Ad-Hoc, Low Energi Adaptive Clustering Hierarchy (LEACH), Hybrid Energy Efficient Distribute (HEED), Power-Efficient Gathering in Sensor Information Systems (PEGASIS)*

RINGKASAN

Analisis Perbandingan Konsumsi Energi dan Masa Hidup Jaringan pada Protokol LEACH, HEED, dan PEGASIS di *Wireless Sensor Network*; Muhammad Arief Wahyudi, 141910201067; 2018; 115 halaman; Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

Jaringan WSN terdiri dari sejumlah *sensor node* yang diatur sedemikian rupa sehingga menjadi sebuah jaringan dimana antar *sensor node* bisa bekerja sama satu sama lain. Fungsi utama dari WSN adalah mengumpulkan data tersebar melalui jaringan sensor yang kemudian dikirimkan dengan jaringan *wireless* menuju *Base Station* (BS) untuk diolah lebih lanjut. Terdapat beberapa masalah dalam jaringan WSN, salah satunya tentang konsumsi energi dan masa hidup jaringan. Salah satu cara yang dapat dilakukan untuk mengatasi hal tersebut dengan merancang protokol jaringan WSN.

Pada penelitian ini menggunakan tiga buah protokol jaringan WSN, yaitu *Low Energi Adaptive Clustering Hierarchy* (LEACH), *Hybrid Energy Efficient Distribute* (HEED), dan *Power-Efficient Gathering in Sensor Information Systems* (PEGASIS). Dari ketiga protokol tersebut akan dibandingkan konsumsi energi dan masa hidup jaringan yang paling efektif.

Penelitian ini memiliki beberapa tahapan, yaitu pertama menentukan parameter sesuai dengan *Zigbee X-bee Pro*. Kemudian, merancang algoritma dari tiga buah protokol dengan menggunakan parameter yang telah ditentukan sebelumnya dan dilakukan dua kali pengujian yaitu perubahan jumlah *node* dan perubahan energi awal. Untuk jumlah *node* yang digunakan yaitu 10, 30, 50, 70, dan 100 *node* dengan energi awal 0,25 *joule*. Sedangkan untuk perubahan energi awal menggunakan 0,1; 0,3; 0,5; 0,7; dan 0,9 *joule* dengan jumlah *node* sebanyak 100 *node*. Setelah algoritma tersusun, dilakukan simulasi menggunakan *software* Matlab R2013a dan akan didapatkan hasil perbandingan konsumsi energi serta masa hidup jaringan untuk selanjutnya dianalisis. Berdasarkan hasil analisis, didapatkan kelebihan dan kekurangan dari tiga buah protokol tersebut.

Dari hasil simulasi yang dilakukan, pada saat protokol LEACH dibandingkan dengan protokol HEED, protokol HEED lebih unggul. Hal ini dibuktikan dengan konsumsi energi yang lebih sedikit dan jumlah *node* yang mati lebih sedikit jika dibandingkan dengan protokol LEACH. Pada saat menggunakan 100 *node*, rata-rata energi sisa tiap *node* 0,0025 *joule* pada protokol LEACH dengan 99 *node* mati, sedangkan pada protokol HEED 0,0705 *joule* dengan 43 *node* mati. Pada saat protokol LEACH dibandingkan dengan protokol PEGASIS, protokol PEGASIS lebih unggul. Hal ini dibuktikan dengan konsumsi energi yang lebih sedikit dan jumlah *node* yang mati lebih sedikit jika dibandingkan dengan protokol LEACH. Pada saat menggunakan 100 *node*, rata-rata energi sisa tiap *node* 0,0097 *joule* pada protokol LEACH dengan 66 *node* mati, sedangkan pada protokol PEGASIS 0,2060 *joule* dengan tidak ada *node* yang mati. Pada saat protokol HEED dibandingkan dengan protokol PEGASIS, protokol PEGASIS lebih unggul. Hal ini dibuktikan dengan konsumsi energi yang lebih sedikit dan jumlah *node* yang mati lebih sedikit jika dibandingkan dengan protokol HEED. Pada saat menggunakan 100 *node*, rata-rata energi sisa tiap *node* 0,0725 *joule* pada protokol HEED dengan 44 *node* mati, sedangkan pada protokol PEGASIS 0,2040 *joule* dengan tidak ada *node* yang mati.

Kesimpulan penelitian ini, protokol PEGASIS memiliki konsumsi energi yang paling hemat dan masa hidup jaringan yang paling baik dari ketiga protokol yang dianalisis. Sedangkan protokol HEED masih lebih baik daripada protokol LEACH meskipun menggunakan teknik yang sama yaitu teknik *clustering*.

PRAKATA

Puji syukur kehadirat Allah SWT yang maha kuasa atas segalanya, karena dengan ridho, hidayah dan petunjuk-Nya, penulis dapat menyelesaikan skripsi ini. Selama penyusunan skripsi ini penulis mendapat bantuan berbagai pihak yang turut memberikan bantuan berupa motivasi, inspirasi, bimbingan, doa, fasilitas dan dukungan lainnya yang membantu memperlancar pengerjaan skripsi ini. Untuk itu penulis mengucapkan terimakasih kepada.

1. Ibu Dr. Ir. Entin Hidayah, M.U.M., Selaku Dekan Fakultas Teknik Universitas Jember;
2. Bapak Dr. Bambang Srikaloko, S.T., M.T., Selaku Ketua Jurusan Teknik Elektro Universitas Jember dan dosen pembimbing akademik yang selalu membimbing dan memberi motivasi dari awal semester;
3. Bapak Widya Cahyadi, S.T., M.T. dan Catur Suko Sarwono, S.T., M.Si. selaku dosen pembimbing yang telah membimbing menyelesaikan tugas akhir ini;
4. Bapak Andrita Ceriana Eska, S.T., M.T. dan Bapak Alfredo Bayu Satriya, S.T., M.T. selaku dosen penguji yang sudah memberikan saran untuk memperbaiki tugas akhir ini;
5. Kedua Orang tua saya Ibu Sri Wachyuni dan Bapak Amin Taksir, yang telah membesarkan, mendidik, mendoakan tiada henti, memberi motivasi semangat, dan memberi kasih sayang yang tak pernah habis serta pengorbanannya selama hingga detik ini;
6. Kakakku Ayu Purnamawati yang selalu memberi semangat, dukungan, dan telah menjadi inspirasi dan adikku Muhammad Kadarisman yang telah menjadi teman di keluarga dan selalu memberi semangat;
7. Dwie Putri Donnaro, Dwi Sukma Aji, Hanifatuz Sa'diyah, dan teman-teman konsentrasi Elektronika Telekomunikasi yang telah menemani pada akhir-akhir masa perkuliahan dan selalu memberi inspirasi, serta teman-temanku

KETEK UJ 2014 yang telah menjadi keluarga selama menempuh kuliah di jurusan tercinta Teknik Elektro;

8. Hadid, Tjoy, Kaped, Fayer, Ujek, Adel, Nganjuk, Aji, Nuha, Irfan, dan Rosi yang telah rela untuk menjadi tempat keluh kesah baik suka maupun duka selama menjalankan kuliah semester akhir dan selama proses skripsi ini;
9. Rifqi, Alysa, Rijal, dan keluarga besar Laboratorium Listrik Dasar yang telah menemani serta memberi pengetahuan dan pengalaman baru selama menjadi asisten laboratorium;
10. Keluarga KKN Reguler 26 Desa Pace, Mbak Sulis, Zaida Nyai, Heni, Karom, Kholifah, Rio, Ayunda, dan Ulum. Terimakasih atas doa dan semangat yang diberikan, terimakasih untuk nama kedua atas semua perhatian, kesabaran, pengorbanan, kasih sayang, kenangan, canda tawa, semangat, dukungan, motivasi serta menjadi tempat keluh kesah. Semoga kita semua bisa meraih kesuksesan dan kebahagiaan di masa depan;
11. Keluarga Kontrakan Jegger (Firman, Waldhy, Apek, Aam, Amin, dan Muji) yang setia menemani dalam suka dan duka selama kuliah di Jember;
12. Serta semua pihak yang tidak dapat disebutkan satu per satu, yang telah mendukung dalam penyelesaian skripsi ini.

Semoga skripsi ini dapat bermanfaat dalam mengembangkan ilmu pengetahuan khususnya untuk disiplin ilmu teknik elektro. Kritik dan saran yang membangun diharapkan terus mengalir untuk lebih menyempurnakan skripsi ini dan dapat dikembangkan untuk penelitian selanjutnya;

Jember, 25 April 2018

Penulis

DAFTAR ISI

	Halaman
HALAMAN JUDUL	i
HALAMAN PERSEMBAHAN	ii
HALAMAN MOTO	iii
HALAMAN PERNYATAAN	iv
HALAMAN PEMBIMBING SKRIPSI	v
HALAMAN PENGESAHAN	vi
ABSTRAK	vii
ABSTRACT	viii
RINGKASAN	ix
PRAKATA	xi
DAFTAR ISI	xiii
DAFTAR GAMBAR	xv
DAFTAR TABEL	xvi
DAFTAR NOTASI	xviii
DAFTAR SINGKATAN	xix
BAB 1. PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah	3
1.4 Tujuan Penelitian	3
1.5 Manfaat Penelitian	4
BAB 2. TINJAUAN PUSTAKA	5
2.1 Wireless Sensor Network (WSN)	5
2.1.1 Pengertian WSN	5
2.1.2 Arsitektur WSN	7
2.1.3 Komunikasi pada WSN	9
2.1.4 Permasalahan pada WSN	10

2.1.5 Topologi WSN	11
2.2 Hierarchical Routing Protocol	14
2.3 Low Energy Adaptive Clustering Hierarchy (LEACH)	15
2.4 Hybrid Energy-Efficient Distributed (HEED).....	19
2.5 Power-Efficient Gathering in Sensor Information Systems.....	22
2.6 Parameter Kerja	26
2.7 MATLAB (Matrix Laboratory)	27
BAB 3. METODOLOGI PENELITIAN	28
3.1 Waktu dan Tempat Penelitian	28
3.2 Alat dan Bahan	29
3.3 Metodologi Pelaksanaan Penelitian	29
3.4.1 LEACH	31
3.4.2 HEED	32
3.4.3 PEGASIS	33
3.4 Parameter Pengujian	34
3.5 Skenario Pengujian	35
BAB 4. HASIL DAN PEMBAHASAN	36
4.1 Analisis Perbandingan Protokol LEACH dengan HEED	36
4.1.1 Pengujian Variasi Jumlah <i>Node</i>	36
4.1.2 Pengujian Variasi Energi Awal	41
4.2 Analisis Perbandingan Protokol LEACH dengan PEGASIS	45
4.2.1 Pengujian Variasi Jumlah <i>Node</i>	45
4.2.2 Pengujian Variasi Energi Awal	50
4.3 Analisis Perbandingan Protokol HEED dengan PEGASIS	53
4.3.1 Pengujian Variasi Jumlah <i>Node</i>	53
4.3.2 Pengujian Variasi Energi Awal	59
BAB 5. PENUTUP	63
5.1 Kesimpulan	63
5.2 Saran	64
DAFTAR PUSTAKA	65
LAMPIRAN	67

DAFTAR TABEL

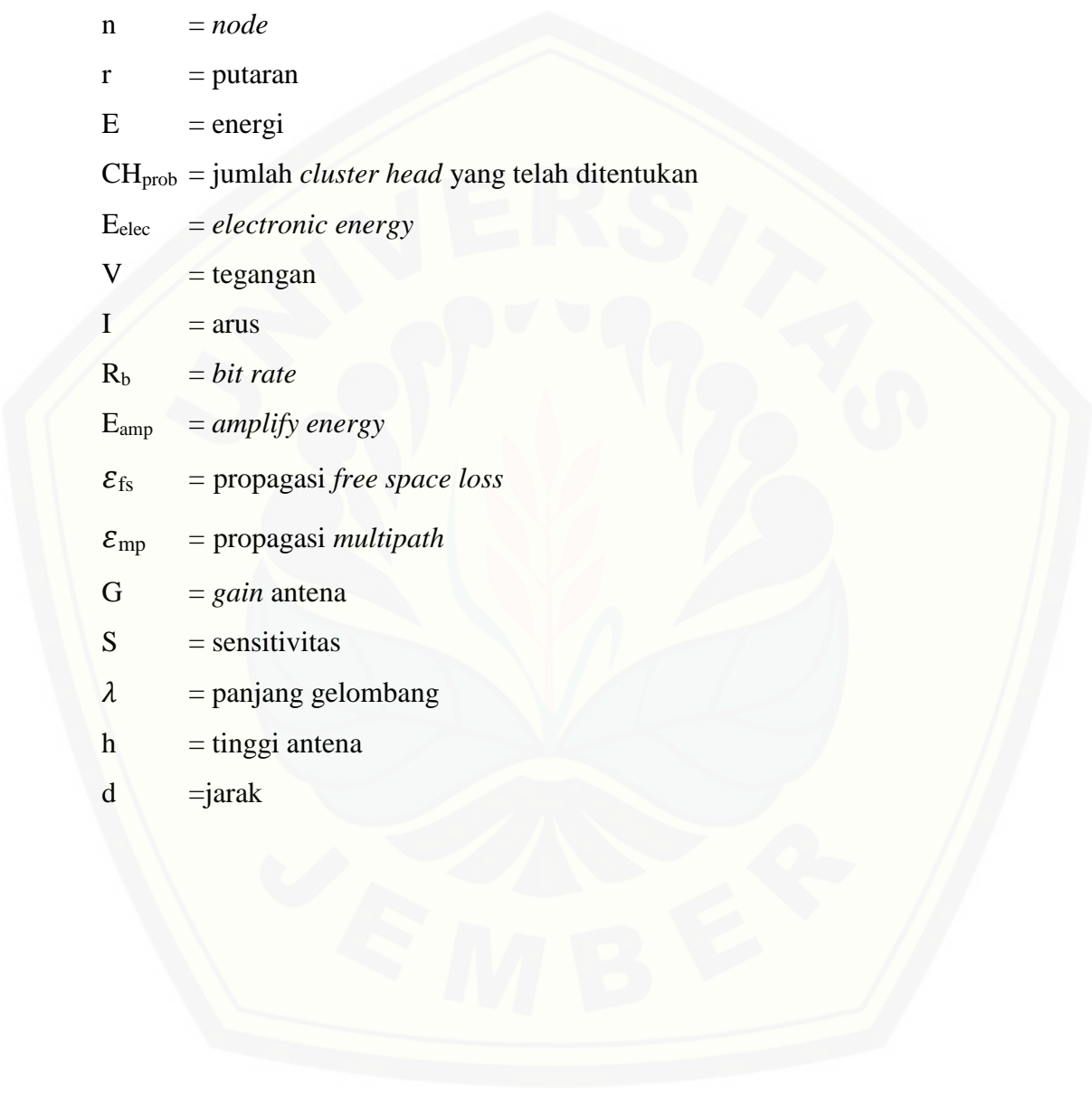
	Halaman
Tabel 3.1 Jadwal Pelaksanaan Penelitian	28
Tabel 3.2 Parameter Simulasi	34
Tabel 3.3 Skenario Pengubahan Jumlah <i>Node</i>	36
Tabel 3.4 Skenario Pengubahan Energi Awal	36
Tabel 4.1 Rata-rata Energi Sisa pada Variasi Jumlah <i>Node</i>	37
Tabel 4.2 Kondisi <i>Node</i> Setelah Simulasi	39
Tabel 4.3 Rata-rata Energi Sisa pada Variasi Energi Awal	41
Tabel 4.4 Kondisi <i>Node</i> Setelah Simulasi	43
Tabel 4.5 Rata-rata Energi Sisa pada Variasi Jumlah <i>Node</i>	45
Tabel 4.6 Kondisi <i>Node</i> Setelah Simulasi	47
Tabel 4.7 Rata-rata Energi Sisa pada Variasi Energi Awal	50
Tabel 4.8 Kondisi <i>Node</i> Setelah Simulasi	51
Tabel 4.9 Rata-rata Energi Sisa pada Variasi Jumlah <i>Node</i>	53
Tabel 4.10 Kondisi <i>Node</i> Setelah Simulasi	56
Tabel 4.11 Rata-rata Energi Sisa pada Variasi Energi Awal	59
Tabel 4.12 Kondisi <i>Node</i> Setelah Simulasi	60

DAFTAR GAMBAR

	Halaman
Gambar 2.1 <i>Wireless Sensor Networks</i>	6
Gambar 2.2 <i>Arsitektur Wireless Sensor Network</i>	8
Gambar 2.3 <i>Komunikasi Single-hop dan Multi-hop pada WSN</i>	9
Gambar 2.4 <i>Hierarchical Clustering</i>	12
Gambar 2.5 <i>Aktuator Sensor Node</i>	13
Gambar 2.6 <i>Komponen node pada WSN</i>	14
Gambar 2.7 <i>Hierarchical Routing Protocol</i>	15
Gambar 2.8 <i>LEACH network model</i>	16
Gambar 2.9 <i>LEACH phases</i>	17
Gambar 2.10 <i>Hybrid Energy-Efficient Distributed (HEED)</i>	20
Gambar 2.11 <i>LEACH and HEED clustering protocols</i>	21
Gambar 2.12 <i>Struktur rantai pada PEGASIS</i>	22
Gambar 2.13 <i>Chain-based data gathering and aggregation scheme</i>	24
Gambar 2.14 <i>Logo MATLAB</i>	27
Gambar 3.1 <i>Tahapan Pengumpulan Data</i>	30
Gambar 3.2 <i>Tahapan protokol LEACH</i>	31
Gambar 3.3 <i>Tahapan protokol HEED</i>	32
Gambar 3.4 <i>Tahapan protokol PEGASIS</i>	33
Gambar 4.1 <i>Grafik Perbandingan Rata-rata Energi Sisa Skenario 10 dan 100 Node</i>	38
Gambar 4.2 <i>Perbandingan Simulasi LEACH dengan HEED (10 Node)</i>	40
Gambar 4.3 <i>Perbandingan Simulasi LEACH dengan HEED (100 Node)</i>	40
Gambar 4.4 <i>Grafik Perbandingan Rata-rata Energi Sisa Skenario 0,1 dan 0,9 joule</i>	42
Gambar 4.5 <i>Perbandingan Simulasi LEACH dengan HEED (0,9 joule)</i>	44
Gambar 4.6 <i>Grafik Perbandingan Rata-rata Energi Sisa Skenario 10 dan 100 Node</i>	46
Gambar 4.7 <i>Perbandingan Simulasi LEACH dengan PEGASIS (10 Node)</i>	48

Gambar 4.8 Perbandingan Simulasi LEACH dengan PEGASIS (100 <i>Node</i>).....	49
Gambar 4.9 Grafik Perbandingan Rata-rata Energi Sisa Skenario 0,1 dan 0,9 <i>joule</i>	51
Gambar 4.10 Perbandingan Simulasi LEACH dengan PEGASIS (0,9 <i>joule</i>) ...	52
Gambar 4.11 Grafik Perbandingan Rata-rata Energi Sisa Skenario 10 dan 100 <i>Node</i>	55
Gambar 4.12 Perbandingan Simulasi HEED dengan PEGASIS (10 <i>Node</i>)	57
Gambar 4.13 Perbandingan Simulasi HEED dengan PEGASIS (100 <i>Node</i>)	58
Gambar 4.14 Grafik Perbandingan Rata-rata Energi Sisa Skenario 0,1 dan 0,9 <i>joule</i>	60
Gambar 4.15 Perbandingan Simulasi HEED dengan PEGASIS (0,9 <i>joule</i>)	61

DAFTAR NOTASI



T	= <i>Threshold</i>
P	= probabilitas
n	= <i>node</i>
r	= putaran
E	= energi
CH _{prob}	= jumlah <i>cluster head</i> yang telah ditentukan
E _{elec}	= <i>electronic energy</i>
V	= tegangan
I	= arus
R _b	= <i>bit rate</i>
E _{amp}	= <i>amplify energy</i>
ε _{fs}	= propagasi <i>free space loss</i>
ε _{mp}	= propagasi <i>multipath</i>
G	= <i>gain</i> antena
S	= sensitivitas
λ	= panjang gelombang
h	= tinggi antena
d	= jarak

DAFTAR SINGKATAN

WSN	= <i>Wireless Sensor Network</i>
BS	= <i>Base Station</i>
LEACH	= <i>Low Energy Adaptive Clustering Hierarchy</i>
HEED	= <i>Hybrid Energy-Efficient Distributed</i>
PEGASIS	= <i>Power Efficient Gathering in Sensor Information System</i>
MATLAB	= <i>Matrix Laboratory</i>
GPS	= <i>Global Positioning System</i>
IEEE	= <i>Institute of Electrical and Electronics Engineers</i>
HRP	= <i>Hierarchical Routing Protocol</i>
CH	= <i>Cluster Head</i>
CDMA	= <i>Code Division Multiple Access</i>

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Wireless Sensor Network (WSN) adalah sebuah teknologi jaringan nirkabel yang memberikan banyak manfaat bagi kehidupan sehari-hari. Jaringan WSN terdiri dari sejumlah *sensor node* yang diatur sedemikian rupa sehingga menjadi sebuah jaringan dimana antar *sensor node* bisa bekerja sama satu sama lain. Fungsi utama dari WSN adalah mengumpulkan data tersebar melalui jaringan sensor yang kemudian dikirimkan dengan jaringan *wireless* menuju *Base Station* (BS) untuk diolah lebih lanjut.

Pada *Wireless Sensor Network*, *sensor node* memegang peranan penting karena data diperoleh dan dikirimkan menggunakan *node* ini. Masing-masing *sensor node* mempunyai keterbatasan dalam beberapa hal. Salah satunya masalah utama dalam implementasi WSN adalah konsumsi energi dan masa hidup jaringan. Konsumsi energi dan masa hidup jaringan terbatas dikarenakan catu daya *sensor node* hanya disuplai oleh baterai untuk operasinya. Padahal, penggunaan energi yang berulang-ulang akan menjadi kendala karena membutuhkan tambahan biaya dan waktu serta mempunyai tingkat kesulitan yang tinggi apabila sensor berada di tempat yang sulit dijangkau manusia. Karena itu, banyak penelitian terkait WSN dilakukan salah satunya difokuskan untuk mengatasi kekurangan energi dan meningkatkan masa hidup jaringan.

Algoritma pertama yang dikembangkan adalah LEACH (*low energy adaptive clustering hierarchy*). LEACH melakukan rotasi terhadap *cluster head* yang pemilihannya dilakukan secara acak. Karena sifatnya yang acak distribusi *cluster head* menjadi tidak seragam, menyebabkan beberapa *nodes* berada pada jarak yang terlalu jauh dengan *cluster head* (Darbi, Rahayu, & Sari, 2014).

Teknik *clustering* adalah pendekatan kontrol topologi yang efektif. HEED (*Hybrid Energy-Efficient Distributed*) adalah protokol *clustering* terdistribusi, yang melibatkan pengelompokan *node* ke dalam *cluster* dan memilih *cluster head* secara periodik sehingga anggota *cluster head* dapat berkomunikasi dengan *cluster head* mereka dan *cluster heads* akan mengirimkan data gabungan yang diterima dari

anggotanya ke *base station*. *Cluster head* bertanggung jawab untuk koordinasi antar *node* dalam kelompok dan kumpulan data mereka (koordinasi intra *cluster*) dan komunikasi satu sama lain dan dengan pengamat eksternal atas nama *cluster* mereka (komunikasi antar *cluster*). HEED menggunakan radius *cluster* sehingga tidak ada anggota *cluster* yang jauh dari CH. Hal ini merupakan pengembangan dari protokol LEACH. Namun, protokol HEED menggunakan CH yang sama secara terus menerus dan akan menyebabkan CH mengeluarkan energi lebih banyak dari *node* yang lain (Galshetwar & Jeyakumar, 2014).

PEGASIS (*Power Efficient Gathering in Sensor Information System*) adalah *routing protocol* yang menggunakan pendekatan berbasis rantai. Protokol ini menggunakan algoritma mulai dari simpul terjauh dan semua *node sensor* membentuk rantai seperti struktur. PEGASIS berkerja dengan prinsip bahwa setiap *node* akan mengirimkan dan menerima dari tetangga terdekatnya, dimana semakin dekat jarak pengiriman maka semakin kecil energi yang dikeluarkan. Ada pemimpin dalam jaringan yang mendistribusikan beban energi antar *node*, namun hanya ada satu pemimpin yang mengirimkan data ke *Base Station* (Bansal, Kundu, & Kaur, 2014).

Dari tiga penelitian sebelumnya, terdapat kelebihan dari masing-masing penelitian tersebut. Masing-masing protokol juga memiliki kelebihan yang berbeda-beda. Sehingga, pada penelitian kali ini akan menggabungkan skenario yang dilakukan pada ketiga penelitian sebelumnya yaitu perubahan jumlah *node*, perubahan variasi energi awal, serta pengaruh posisi *base station*. Dimana protokol yang digunakan adalah LEACH, HEED, dan PEGASIS. Alasan digunakan tiga protokol tersebut karena HEED merupakan pengembangan dari LEACH sehingga dapat dibandingkan, serta PEGASIS yang memiliki protokol yang berbeda dari keduanya sehingga dapat dibandingkan antara LEACH dengan PEGASIS serta HEED dengan PEGASIS, sehingga diharapkan didapatkan perbandingan yang terbaik antara ketiga protokol tersebut.

Berdasarkan paparan di atas, pada tugas akhir ini akan dikaji perbandingan konsumsi energi dan masa hidup jaringan dari ketiga *clustering protocol* tersebut untuk mengetahui protokol yang terbaik, dengan melakukan perbandingan variasi

jumlah *node*, penambahan energi awal, dan pengaruh posisi *base station*. Sehingga judul dari tugas akhir ini adalah “Analisis Perbandingan Konsumsi Energi dan Masa Hidup Jaringan pada Protokol LEACH, HEED, dan PEGASIS di *Wireless Sensor Network*”.

1.2 Rumusan Masalah

Berdasarkan permasalahan yang telah dijelaskan pada latar belakang, maka rumusan masalah yang diangkat sebagai berikut:

1. Bagaimana perbandingan tingkat konsumsi energi dan masa hidup jaringan pada protokol LEACH dengan HEED?
2. Bagaimana perbandingan tingkat konsumsi energi dan masa hidup jaringan pada protokol LEACH dengan PEGASIS?
3. Bagaimana perbandingan tingkat konsumsi energi dan masa hidup jaringan pada protokol HEED dengan PEGASIS?

1.3 Batasan Masalah

Penelitian ini berisikan tentang batasan-batasan yang terdapat pada perancangan dan simulasi yaitu:

1. Simulasi menggunakan *software* MATLAB.
2. Percobaan dilakukan pada jaringan *wireless* yang statis dengan sekumpulan *node* yang diletakkan secara acak.
3. Simulasi menggunakan sekumpulan *node* homogen yang mempunyai energi awal yang sama.
4. Semua *node* pada simulasi dirancang mempunyai parameter Zigbee jenis Xbee Pro S1 (IEEE 802.15.4).

1.4 Tujuan Penelitian

Adapun tujuan yang diharapkan dalam pelaksanaan penelitian ini sebagai berikut:

1. Menganalisis perbandingan tingkat konsumsi energi dan masa hidup jaringan pada protokol LEACH dengan HEED.

2. Menganalisis perbandingan tingkat konsumsi energi dan masa hidup jaringan pada protokol LEACH dengan PEGASIS.
3. Menganalisis perbandingan tingkat konsumsi energi dan masa hidup jaringan pada protokol HEED dengan PEGASIS.

1.5 Manfaat Penelitian

Berdasarkan penelitian yang dilakukan, diharapkan dapat memberikan manfaat yaitu:

1. Mengetahui perbandingan konsumsi energi dari protokol LEACH, HEED, dan PEGASIS pada *Wireless Sensor Network*.
2. Mengetahui perbandingan masa hidup jaringan dari protokol LEACH, HEED, dan PEGASIS pada *Wireless Sensor Network*.

BAB 2. TINJAUAN PUSTAKA

Pada bab tinjauan pustaka ini dilakukan sebagai daftar acuan dalam melakukan penelitian tugas akhir yang saat ini dikerjakan. Tinjauan ini dilakukan guna dapat memperkaya wawasan dan mengembangkan ilmu pengetahuan yang telah ada, sehingga bidang keilmuan yang diteliti akan mengalami perkembangan.

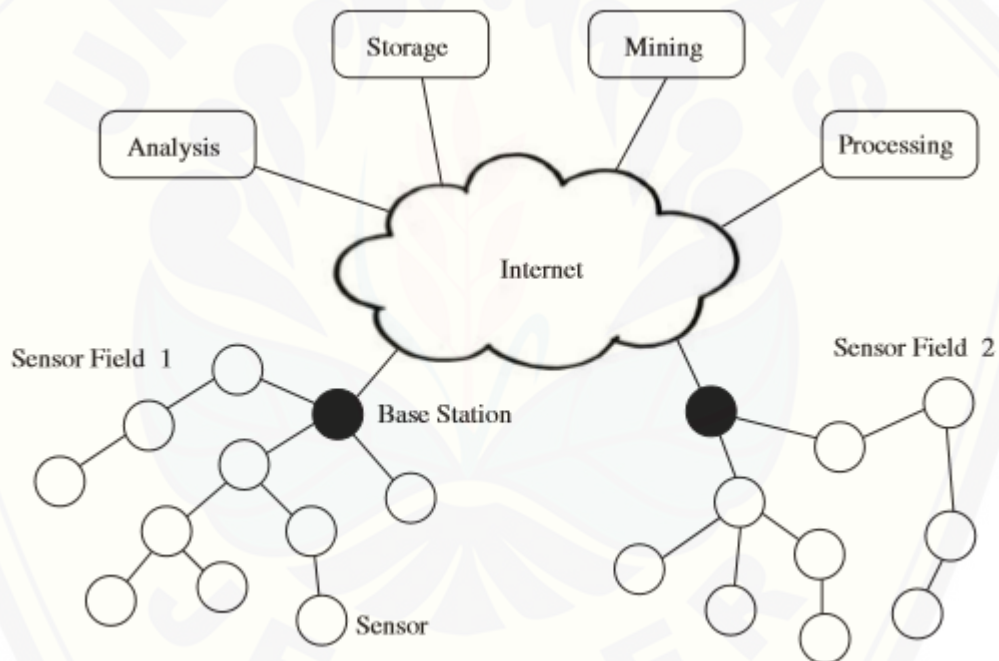
2.1 *Wireless Sensor Network* (WSN)

2.1.1 Pengertian WSN

Kemajuan terbaru dalam teknologi sensor dan komunikasi nirkabel bersamaan dengan perkembangan mikroelektronika telah menyediakan jenis jaringan komunikasi baru yang terbuat dari perangkat sensor nirkabel terintegrasi bertenaga baterai. Jaringan Sensor Nirkabel (*Wireless Sensor Networks* / WSN), seperti namanya, adalah jaringan nirkabel yang dibangun sendiri dan tanpa infrastruktur yang terbuat dari perangkat kecil yang dilengkapi dengan sensor khusus dan *transceiver* nirkabel. Tujuan utama WSN adalah mengumpulkan data dari lingkungan dan mengirimkannya ke situs pelaporan dimana data dapat diamati dan dianalisis. Perangkat sensor nirkabel juga merespon permintaan yang dikirim dari "situs kontrol" untuk melakukan instruksi tertentu atau memberikan contoh penginderaan. Akhirnya, perangkat sensor nirkabel bisa dilengkapi dengan aktuator untuk "bertindak" pada kondisi tertentu. Jaringan ini kadang-kadang lebih spesifik disebut *Wireless Sensor and Actuator Networks* (Labrador & Wightman, 2009).

Sementara banyak sensor terhubung ke pengendali dan stasiun pengolahan secara langsung (misal, menggunakan *Local Area Networks*), semakin banyak sensor mengkomunikasikan data yang dikumpulkan secara nirkabel ke stasiun pemrosesan terpusat. Hal ini penting karena banyak aplikasi jaringan memerlukan ratusan atau ribuan *sensor nodes*, sering ditempatkan di daerah terpencil dan tidak terjangkau. Oleh karena itu, sensor nirkabel tidak hanya memiliki komponen penginderaan, tapi juga kemampuan pemrosesan, komunikasi, dan penyimpanan on-board. Dengan perangkat tambahan ini, *sensor node* seringkali tidak hanya bertanggung jawab untuk pengumpulan data, namun juga untuk analisis jaringan,

korelasi, dan perpaduan data sensor dan data dari *sensor node* lainnya. Ketika banyak sensor secara kooperatif memonitor lingkungan fisik yang besar, mereka membentuk jaringan sensor nirkabel (*Wireless Sensor Network / WSN*). *Sensor node* berkomunikasi tidak hanya satu sama lain, tetapi juga dengan *Base Station* (BS) menggunakan radio nirkabel mereka, yang memungkinkan mereka untuk menyebarkan *sensor data* mereka ke pemrosesan, visualisasi, analisis, dan sistem penyimpanan jarak jauh. Sebagai contoh, Gambar 2.1 menunjukkan dua bidang sensor yang memantau dua wilayah geografis yang berbeda dan terhubung ke Internet dengan menggunakan *base station* mereka (Dargie & Poellabauer, 2010).



Gambar 2.1 *Wireless Sensor Networks*
(Dargie & Poellabauer, 2010)

Kemampuan *sensor node* dalam WSN dapat sangat bervariasi, yaitu *sensor node* sederhana dapat memantau satu fenomena fisik, sementara perangkat yang lebih kompleks dapat menggabungkan banyak teknik penginderaan yang berbeda (misalnya, akustik, optik, magnetik). Mereka juga dapat berbeda dalam kemampuan komunikasi mereka, misalnya menggunakan teknologi ultrasonografi,

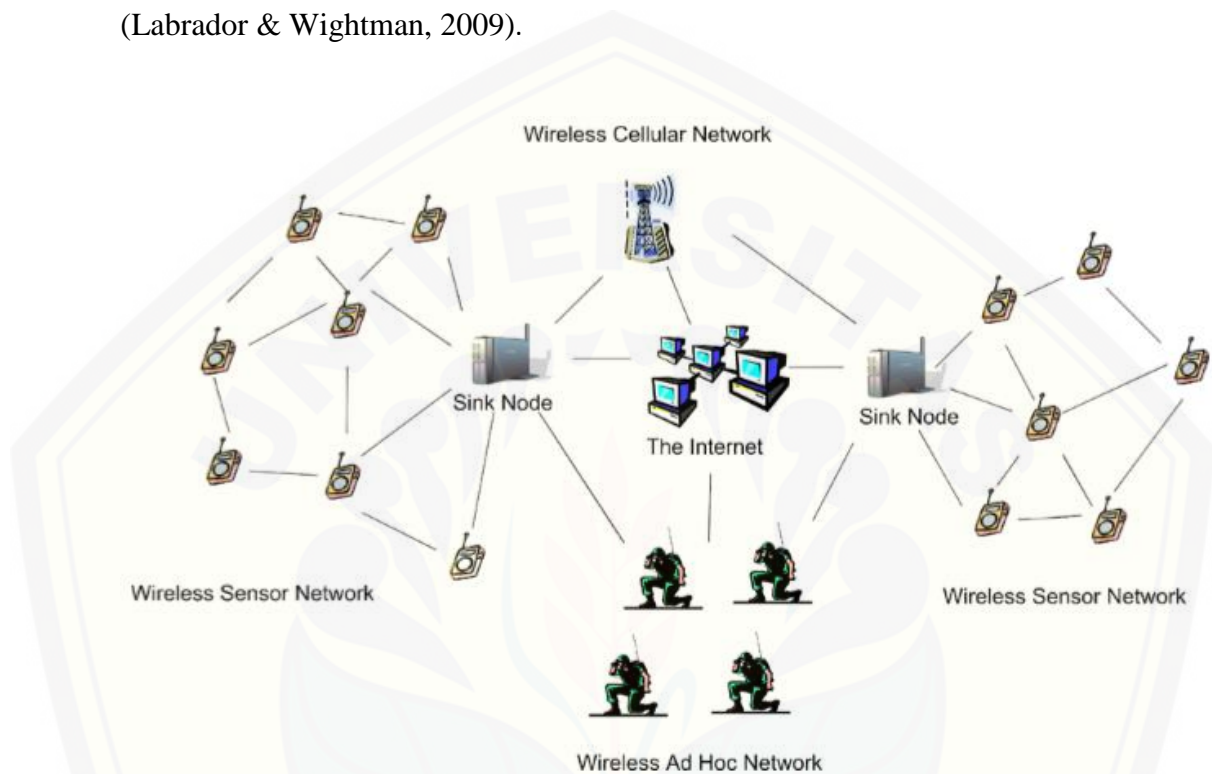
inframerah, atau frekuensi radio dengan kecepatan data dan *latency* yang bervariasi. Meskipun sensor sederhana hanya dapat mengumpulkan dan mengkomunikasikan informasi tentang lingkungan yang diamati, perangkat yang lebih hebat (yaitu, perangkat dengan kapasitas pemrosesan, energi, dan penyimpanan yang besar) juga dapat melakukan fungsi pemrosesan dan agregasi yang ekstensif. Perangkat semacam itu sering memikul tanggung jawab tambahan di WSN, misalnya, mereka dapat membentuk *backbones* komunikasi yang dapat digunakan oleh perangkat sensor terbatas sumber daya lainnya untuk mencapai *base station*. Akhirnya, beberapa perangkat mungkin memiliki akses ke teknologi pendukung tambahan, misalnya, *Global Positioning System (GPS) receiver*, yang memungkinkan mereka menentukan posisi mereka secara akurat. Namun, sistem seperti itu sering mengkonsumsi terlalu banyak energi agar layak untuk *sensor node* berbiaya rendah dan rendah daya (Dargie & Poellabauer, 2010).

2.1.2 Arsitektur WSN

Arsitektur *wireless sensor network* telah berevolusi dari waktu ke waktu dan terus berkembang seiring perangkat baru dan kemampuan tersedia. Awalnya, WSN terdiri dari topologi yang terbuat dari perangkat sensor nirkabel homogen yang mengukur satu variabel. Sebagian besar jaringan ini terdiri dari beberapa perangkat sensor nirkabel (tidak banyak) yang tersebar di seluruh area yang diminati dan satu *sink node* yang menerima semua data dan menghubungkan WSN dengan jaringan lain sehingga data WSN dapat tersedia di lokasi terpencil yang diminati untuk monitoring dan analisa.

Kemudian, aplikasi yang lebih baru membutuhkan jumlah *node*, ratusan, atau bahkan ribuan yang jumlahnya lebih besar. Aplikasi ini membutuhkan arsitektur baru untuk transmisi *wireless sensor data* yang efisien. Dengan demikian, topologi *layered* atau *clustered* dirancang yang mencakup beberapa *sink*. Selanjutnya, jaringan besar ini membutuhkan konektivitas yang berbeda dengan dunia luar, sehingga interkoneksi WSN dengan Internet, jaringan pribadi, jaringan seluler, jaringan *ad hoc* nirkabel, dan sebagainya, merupakan sebuah kebutuhan. Tentunya hal ini memicu upaya besar dalam perancangan dan implementasi

protokol komunikasi untuk memasukkan kemampuan baru ini. Gambar 2.2 menunjukkan dua *flat small wireless sensor networks* dengan satu *sink node* yang terhubung ke jaringan *ad hoc* nirkabel, jaringan seluler dan internet, yang pada saat bersamaan dapat berfungsi sebagai jaringan interkonek di antara keduanya (Labrador & Wightman, 2009).

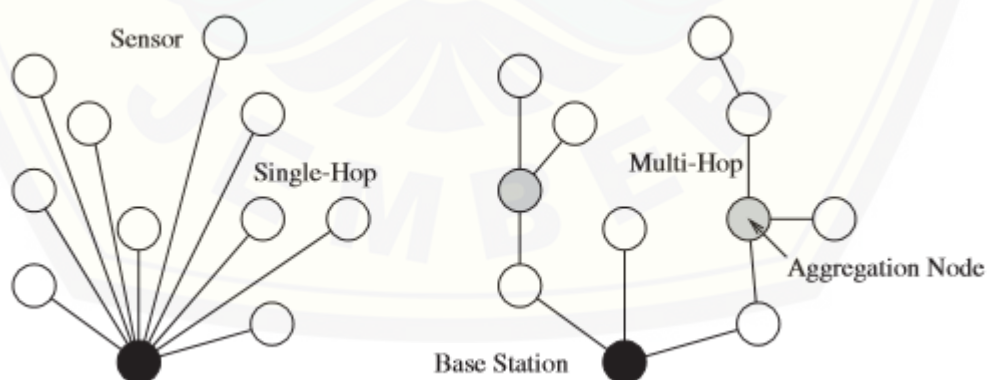


Gambar 2.2 Arsitektur *Wireless Sensor Network*
(Labrador & Wightman, 2009)

Penggabungan beberapa *sinks* mengembangkan arsitektur jaringan untuk menyertakan perangkat heterogen. WSN sekarang mampu mengukur variabel yang berbeda, dan mampu memanfaatkan kemampuan perangkat yang lebih bertenaga untuk melakukan fungsi menarik yang kompleks atau bahkan menghemat energi tambahan. Sebagai contoh, perangkat yang lebih kuat telah diberi peran *cluster head* untuk menggabungkan dan mengirimkan data atas nama *node* lain yang kurang kuat (Labrador & Wightman, 2009).

2.1.3 Komunikasi pada WSN

Standar IEEE 802.11 diperkenalkan pada tahun 1997 dan merupakan teknologi jaringan nirkabel yang paling umum untuk sistem *mobile*. Ini menggunakan pita frekuensi yang berbeda, misalnya pita 2,4 GHz yang digunakan oleh IEEE 802.11b dan IEEE 802.11g, sedangkan protokol IEEE 802.11a menggunakan pita frekuensi 5 GHz. IEEE 802.11 sering digunakan di *wireless sensor network* dan masih dapat ditemukan di jaringan saat ini saat permintaan *bandwidth* tinggi (misal, untuk sensor multimedia). Namun, *overhead* energi tinggi dari jaringan berbasis IEEE 802.11 membuat standar ini tidak sesuai untuk jaringan sensor berdaya rendah. Persyaratan *typical data rate* dalam jaringan sensor sebanding dengan *bandwidth* yang disediakan oleh modem *dial-up*, oleh karena itu kecepatan data yang diberikan oleh IEEE 802.11 biasanya jauh lebih tinggi daripada yang dibutuhkan. Hal ini menyebabkan berkembangnya berbagai protokol yang lebih memuaskan kebutuhan jaringan akan konsumsi daya rendah dan kecepatan data yang rendah. Misalnya, protokol IEEE 802.15.4 (Gutierrez et al., 2001) telah dirancang secara khusus untuk komunikasi jarak pendek di jaringan sensor berdaya rendah dan didukung oleh sebagian besar simpul sensor akademis dan komersial (Dargie & Poellabauer, 2010).



Gambar 2.3 Komunikasi *Single-hop* dan *Multi-hop* pada

Wireless Sensor Networks

(Dargie & Poellabauer, 2010)

Ketika rentang transmisi radio dari semua *sensor node* cukup besar dan sensor dapat mentransmisikan datanya langsung ke *base station*, mereka dapat membentuk topologi bintang seperti yang ditunjukkan di sebelah kiri pada Gambar 2.3. Dalam topologi ini, setiap *sensor node* berkomunikasi langsung dengan *base station* menggunakan satu *hop*. Namun, jaringan sensor sering kali mencakup area geografis yang luas dan daya transmisi radio harus dijaga seminimal mungkin untuk menghemat energi. Oleh karena itu, komunikasi *multi-hop* adalah kasus yang lebih umum untuk jaringan sensor (ditunjukkan di sebelah kanan pada Gambar 2.3). Dalam topologi *mesh* ini, *sensor node* tidak hanya menangkap dan menyebarkan data mereka sendiri, namun juga berfungsi sebagai *relay* untuk *sensor node* lainnya, yaitu mereka harus berkolaborasi untuk menyebarkan *sensor data* ke *base station*. Masalah *routing* ini, yaitu tugas untuk menemukan jalur *multi-hop* dari *sensor node* ke *base station*, merupakan salah satu tantangan yang paling penting dan mendapat perhatian besar dari komunitas riset. Ketika sebuah *node* berfungsi sebagai *relay* untuk beberapa rute, seringkali ada kesempatan untuk menganalisis dan memproses *sensor data* pra-jaringan, yang dapat menyebabkan penghapusan informasi berlebihan atau kumpulan data yang mungkin lebih kecil dari data asli.

2.1.4 Permasalahan pada WSN

Terdapat beberapa manfaat yang dapat diperoleh dari penggunaan dan perkembangan *Wireless Sensor Network* (WSN). Akan tetapi, terdapat juga permasalahan yang timbul dari pengimplementasian WSN. Permasalahan yang muncul pada WSN antara lain (Mukarromah, 2014):

- a. Masalah yang paling sering timbul yaitu penggunaan energi yang besar pada pengoperasian WSN, sementara pada jaringan tersebut daya yang diperoleh berasal dari *power supply* berupa baterai. Daya yang ada pada baterai sangat terbatas sehingga sangat dibutuhkan efisiensi energi pada jaringan WSN ini.
- b. Masalah yang kedua adalah terbatasnya memori. Sensor yang digunakan merupakan sensor yang sangat kecil dengan memori dan tempat penyimpanan yang juga kecil. Biasanya sensor tidak mempunyai ruang yang cukup untuk menjalankan aplikasi yang kompleks dalam algoritmanya.

- c. Masalah selanjutnya, komunikasi yang tidak handal merupakan salah satu masalah penting pada keamanan jaringan WSN. Secara normal, *packet-based* dari jaringan sensor berdasarkan protokol *connectionless* yang sudah dapat dipastikan tidak dapat diandalkan.
- d. Dan yang terakhir pengoperasian jaringan tanpa pengawasan. Sebagian besar pengoperasian WSN merupakan pengoperasian jarak jauh, sehingga keamanan pada jaringan akan susah untuk diawasi.

2.1.5 Topologi WSN

Berikut ini adalah struktur - struktur jaringan *Wireless Sensor Network* (Rijal, Kristalina, & Santoso):

a. *Flat-based*

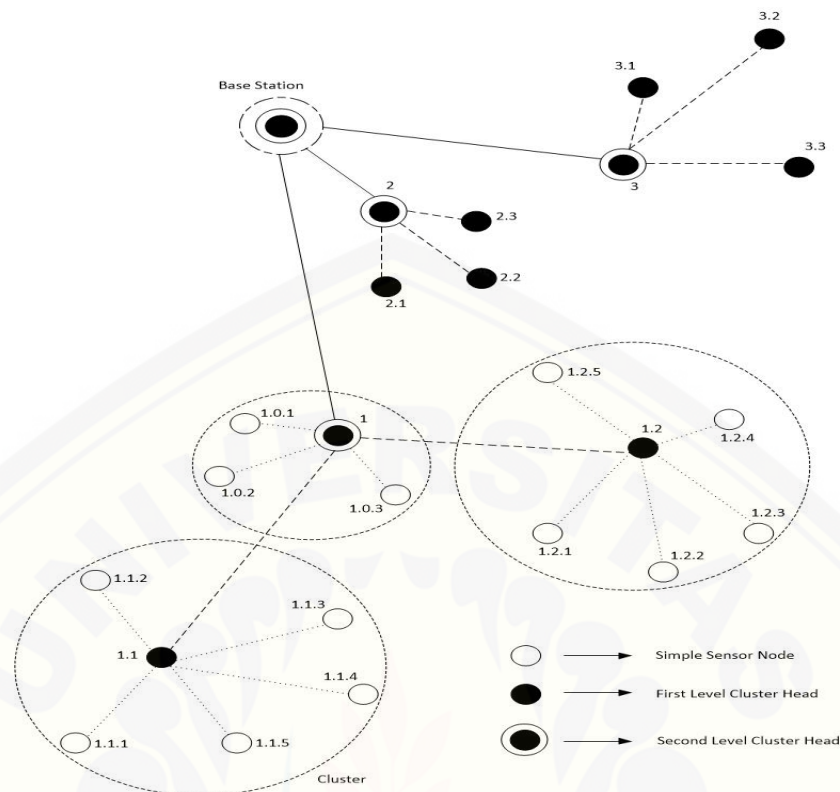
Flat routing protocol mendistribusikan informasi yang diperlukan untuk setiap *sensor node* yang terjangkau dalam jaringan sensor. Semua *node* memainkan peran yang sama dan tidak ada sama sekali hirarki. Tidak ada upaya dilakukan untuk mengatur jaringan atau trafik, hanya untuk menemukan rute terbaik dengan lompatan-lompatan (*hop*) ke tujuan dengan jalan manapun.

b. *Location-based*

Sebagian besar protokol *routing* untuk jaringan sensor memerlukan informasi lokasi untuk *sensor node*. Dalam kebanyakan kasus, informasi lokasi dibutuhkan untuk menghitung jarak antara dua *node* tertentu sehingga konsumsi energi dapat diperkirakan. Hal ini dikarenakan tidak ada skema pengalamatan untuk jaringan sensor seperti alamat IP.

c. *Hierarchical-based*

Kelas ini menetapkan *routing* protokol untuk mencoba menghemat energi dengan mengatur *node* dalam *cluster*. *Node-node* dalam *cluster* mengirimkan data ke *Cluster Head*, dan *Cluster Head* inilah yang meneruskan data ke *Base Station*. *Clustering* yang baik memainkan peran penting dalam skalabilitas jaringan serta penghematan energi.



Gambar 2.4 *Hierarchical Clustering*
(Gupta & Nayyar, 2014)

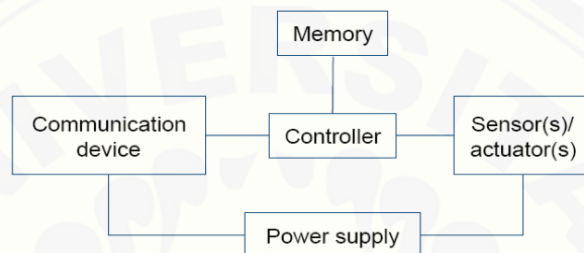
Secara luas, *Wireless Sensor Network* terdiri dari beberapa bagian, yang secara dominan terdiri dari *node-node* yang dapat membentuk sebuah topologi. Berikut ini adalah komponen-komponen yang membentuk WSN (Aufar, 2017):

1. *Sensor nodes*

Sensor node merupakan perangkat yang memiliki kemampuan untuk merutekan data (lingkungan atau objek yang dipantau) yang dikumpulkan ke *sensor node* lain yang berdekatan. *Sensor node* memiliki beberapa komponen sebagai berikut :

- a) *Processor* atau *controller* : sebagai otak dari *node* itu sendiri yang mempunyai beberapa fungsi, seperti mode aktif, *sleep*, *idle*.
- b) *Power source* : sebagai sumber tenaga, yang berasal dari baterai AA, baterai koin (jam) dan panel surya.

- c) *Memory* : sebagai tempat penyimpanan kode program dan *in-memory buffering*.
- d) *Communication device* (radio) : sebagai pentransmisian data yang dipunya ke tempat penyimpanan data yang lain, dan sebagai komunikasi antar *node*.
- e) *Sensor* : sebagai pendeteksi keadaan lapangan (temperatur, suara, getaran, dan kelembapan).



Gambar 2.5 Aktuator Sensor Node

(Aufar, 2017)

2. *Sink nodes*

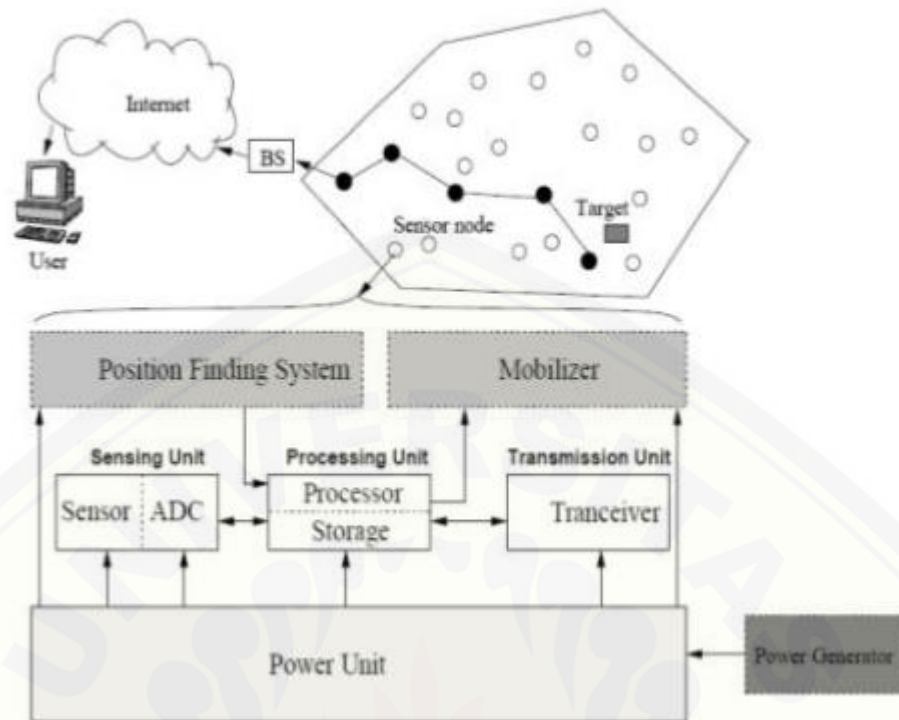
Sink node adalah perangkat yang mempunyai fungsi mengumpulkan data yang didapatkan dari *sensor node*. Kemudian, data tersebut akan diteruskan ke perangkat atau sistem lain, seperti ke *database server* untuk penyimpanan. Selain untuk mengumpulkan data dari *sensor node*, *sink node* juga berfungsi sebagai penyebar paket dari perangkat ke *wireless sensor network*.

3. Internet atau *database*

Internet atau *database* berfungsi untuk pencatat atau penyimpan data yang didapatkan oleh *sensor node* dan juga sebagai gerbang konektivitas yang menghubungkan *user* dengan *node*.

4. *User*

User adalah *client* yang berisi perangkat untuk keperluan visualisasi atau analisis data dan sebagai pengatur *wireless sensor network*.

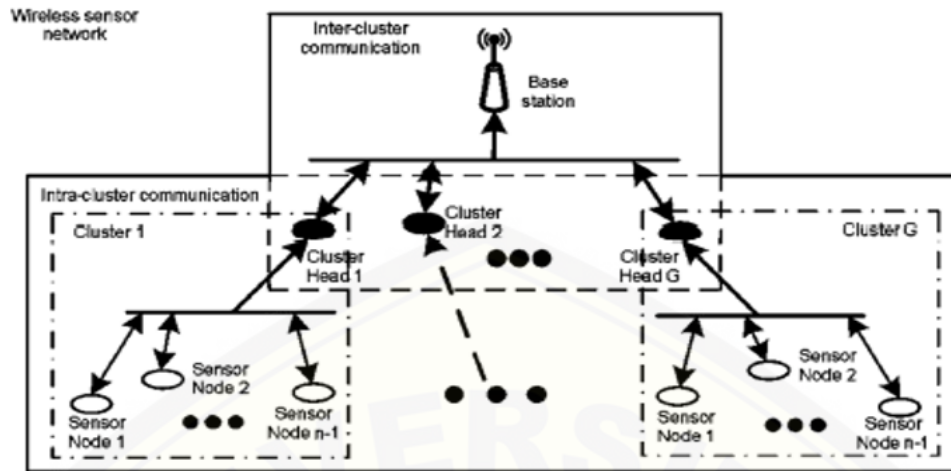


Gambar 2.6 Komponen *node* pada WSN

(Aufar, 2017)

2.2 Hierarchical Routing Protocol

Pada arsitektur hirarki, beberapa *node* bertanggung jawab untuk melakukan pengiriman dengan energi yang lebih tinggi disaat *node* lain melakukan tugas biasa. Algoritma *power-aware* digunakan untuk memilih *node* yang harus melalui persyaratan untuk meneruskan data dari *node* biasa ke *Base Station*. Arsitektur *hierarchical-based* pada WSN dapat ditunjukkan seperti pada Gambar 2.7 (Prabowo, 2014).



Gambar 2.7 Hierarchical Routing Protocol

(Prabowo, 2014)

HRP dapat dikategorikan menjadi 2 tipe berdasarkan manajemen topologinya, yaitu *cluster-based* HRP, contohnya adalah LEACH dan HEED, serta *chain-based* HRP, contohnya adalah PEGASIS. Pada *cluster-based* HRP, *sensor node* dikelompokkan ke dalam *Cluster Head* (CH). CH bertindak sebagai *node* perantara antara anggota *cluster* dengan BS.

2.3 Low Energy Adaptive Clustering Hierarchy (LEACH)

Low Energy Adaptive Clustering Hierarchy (LEACH) adalah algoritma *routing* yang dirancang untuk mengumpulkan dan mengirimkan data ke *sink data*, biasanya sebuah *base station*. Tujuan utama LEACH adalah (Sohraby, Minoli, & Znati, 2007):

- Perpanjangan masa pakai jaringan
- Mengurangi konsumsi energi oleh masing-masing *sensor node* jaringan
- Penggunaan agregasi data untuk mengurangi jumlah pesan komunikasi

Untuk mencapai tujuan ini, LEACH mengadopsi pendekatan hierarkis untuk mengatur jaringan menjadi satu set *cluster*. Setiap *cluster* dikelola oleh *cluster head*. *Cluster head* bertanggung jawab untuk melakukan banyak tugas. Tugas pertama terdiri dari kumpulan data berkala dari anggota *cluster*. Setelah

mengumpulkan data, *cluster head* mengumpulkannya dalam upaya menghilangkan redundansi di antara nilai yang berkorelasi. Tugas kedua dari *cluster head* adalah mentransmisikan data gabungan secara langsung ke *base station*. Transmisi data gabungan dicapai melalui *single hop*. Model jaringan yang digunakan oleh LEACH digambarkan pada Gambar 5.8. Tugas ketiga dari *cluster head* adalah membuat jadwal berbasis TDMA dimana masing-masing *node* dari *cluster* diberi slot waktu yang dapat digunakan untuk transmisi. *Cluster head* menginformasikan jadwal ke anggota *cluster* melalui *broadcasting*. Untuk mengurangi kemungkinan terjadinya tabrakan antar sensor di dalam dan di luar *cluster*, LEACH *node* menggunakan skema pembagian beberapa skema berbasis akses untuk komunikasi.

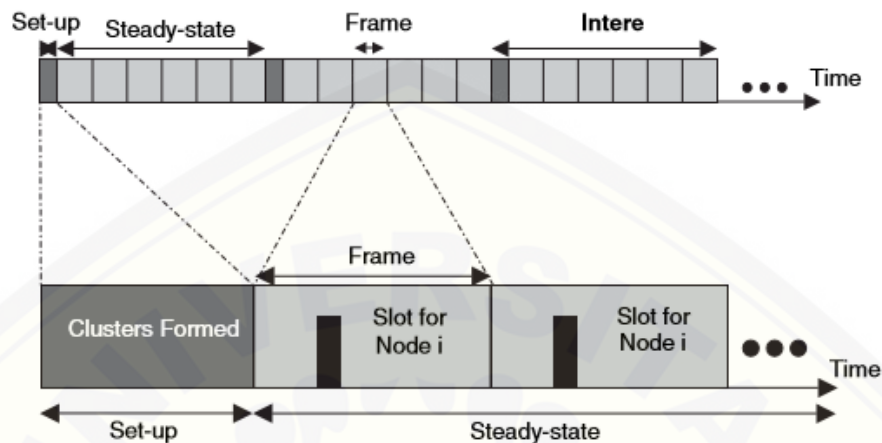


Gambar 2.8 LEACH network model

(Sohraby, Minoli, & Znati, 2007)

Operasi dasar LEACH diatur dalam dua tahap yang berbeda. Fase ini diilustrasikan pada Gambar 2.9. Fase pertama, *the setup phase*, terdiri dari dua tahap, seleksi *cluster-head* dan pembentukan *cluster*. Tahap kedua, *the steady-state phase*, berfokus pada pengumpulan data, penggabungan, dan pengiriman ke *base*

station. Durasi *the setup phase* diasumsikan relatif lebih singkat daripada *the steady-state phase* untuk meminimalkan *overhead* protokol.



Gambar 2.9 LEACH phases
(Sohraby, Minoli, & Znati, 2007)

Pada awal tahap *setup*, putaran seleksi *cluster-head* dimulai. Proses seleksi *cluster-head* memastikan bahwa peran ini berputar di antara *sensor nodes*, sehingga mendistribusikan konsumsi energi secara merata ke semua *node* jaringan. Untuk menentukan apakah gilirannya menjadi *cluster head*, sebuah *node*, n , menghasilkan bilangan acak, v , antara 0 dan 1 dan membandingkannya dengan seleksi *threshold cluster-head*, $T(n)$. *Node* menjadi *cluster head* jika nilai yang dihasilkannya, v , kurang dari $T(n)$. Seleksi *threshold cluster-head* dirancang untuk memastikan dengan probabilitas tinggi bahwa fraksi *node* yang telah ditentukan sebelumnya, P , dipilih sebagai *cluster head* pada setiap putaran. Selanjutnya, batas *threshold* memastikan bahwa *node* yang terlayani dalam putaran $1/P$ terakhir tidak dipilih pada putaran saat ini. Untuk memenuhi persyaratan ini, *threshold* $T(n)$ dari *node* yang bersaing n dapat dinyatakan sebagai berikut :

$$T(n) = \begin{cases} \frac{P}{1 - P \times (r \bmod \frac{1}{P})} & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases} \quad (2.1)$$

Variabel G mewakili himpunan *node* yang belum dipilih menjadi *cluster head* dalam putaran $1/P$ terakhir, dan r menunjukkan putaran saat ini. Parameter

yang ditentukan, P , mewakili probabilitas *cluster-head*. Jelas bahwa jika sebuah *node* berfungsi sebagai *cluster head* dalam putaran I/P terakhir, maka tidak akan terpilih dalam putaran ini.

Pada selesainya proses seleksi *cluster-head*, setiap *node* yang terpilih menjadi *cluster head* menginformasikan peran barunya ke jaringan lainnya. Setelah menerima informasi dari *cluster-head*, setiap *node* yang tersisa memilih *cluster* untuk bergabung. Kriteria pemilihan mungkin didasarkan pada kekuatan sinyal yang diterima, di antara faktor lainnya. *Node* kemudian menginformasikan *cluster head* pilihan mereka tentang keinginan mereka untuk menjadi anggota *cluster*.

Setelah pembentukan *cluster*, masing-masing *cluster head* membuat dan mendistribusikan jadwal CDMA, yang menentukan slot waktu yang dialokasikan untuk setiap anggota *cluster*. Setiap *cluster head* juga memilih kode CDMA, yang kemudian didistribusikan ke semua anggota *cluster*. Kode dipilih dengan hati-hati sehingga mengurangi interferensi *intercluster*. Penyelesaian fase *setup* menandai dimulainya fase *steady-state*. Selama fase ini, *node* mengumpulkan informasi dan menggunakan slot yang dialokasikan untuk mengirimkan data yang dikumpulkan ke *cluster head*. Pengumpulan data ini dilakukan secara berkala.

Hasil simulasi menunjukkan bahwa LEACH mencapai penghematan energi yang signifikan. Penghematan ini terutama bergantung pada rasio pengumpulan data yang dicapai oleh *cluster head*. Meskipun demikian, LEACH mengalami beberapa kekurangan. Asumsi bahwa semua *node* dapat mencapai *base station* dalam satu *hop* mungkin tidak realistis, karena kemampuan dan cadangan energi dari *node* dapat bervariasi dari waktu ke waktu dari satu *node* ke *node* lainnya. Selanjutnya, panjang periode *steady-state* sangat penting untuk mencapai pengurangan energi yang diperlukan untuk mengimbangi overhead yang disebabkan oleh proses pemilihan *cluster*. Periode *steady-state* pendek meningkatkan *overhead* protokol, sedangkan periode yang panjang dapat menyebabkan penipisan energi *cluster head*. Beberapa algoritma telah diusulkan untuk mengatasi kekurangan ini. Protokol LEACH (XLEACH) yang diperluas mempertimbangkan tingkat energi *node* dalam proses seleksi *cluster-head*. Pemilihan *threshold cluster head* yang dihasilkan, $T(n)$, digunakan oleh n untuk

menentukan apakah akan menjadi *cluster head* di putaran saat ini didefinisikan sebagai

$$T(n) = \frac{P}{1-P(r \bmod(1/P))} \left[\frac{E_{n,current}}{E_{n,max}} + (r_{n,s} \bmod 1/P) \left(1 - \frac{E_{n,current}}{E_{n,max}} \right) \right] \quad (2.2)$$

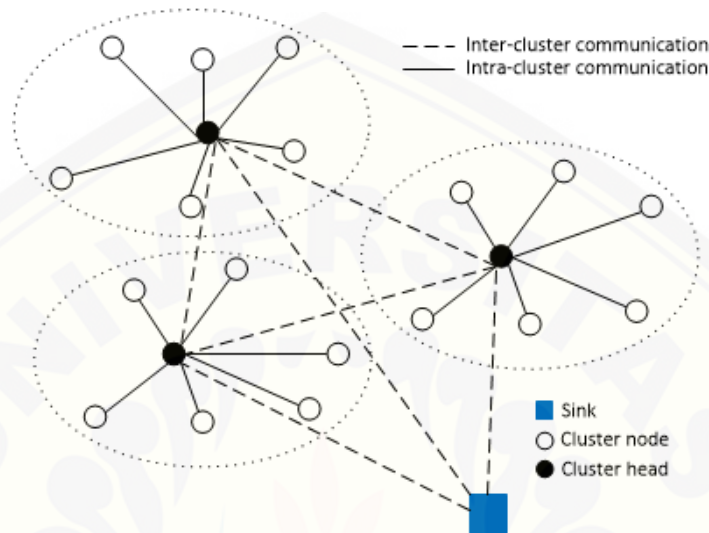
Dalam persamaan ini, $E_{n,current}$ adalah energi saat ini, dan $E_{n,max}$ adalah energi awal dari *sensor node*. Variabel $r_{n,s}$ adalah jumlah putaran berturut-turut dimana sebuah *node* belum menjadi *cluster head*. Bila nilai $r_{n,s}$ mendekati $1/P$, *threshold* $T(n)$ diatur ulang ke nilai yang dimilikinya sebelum dimasukkannya energi yang tersisa ke persamaan *threshold*. Selain itu, $r_{n,s}$ diatur ke 0 ketika sebuah *node* menjadi *cluster head*.

LEACH menampilkan beberapa properti yang memungkinkan protokol mengurangi konsumsi energi. Kebutuhan energi di LEACH didistribusikan ke semua *sensor node*, karena mereka menganggap peran *cluster head* dalam mode *round-robin* berdasarkan energi residu mereka. LEACH adalah algoritma yang benar-benar terdistribusi, tidak memerlukan informasi kontrol dari *base station*. Pengelolaan *cluster* dicapai secara lokal, yang menghapus kebutuhan akan pengetahuan jaringan global. Selanjutnya, pengumpulan data oleh *cluster* juga memberikan kontribusi besar terhadap penghematan energi, karena *node* tidak lagi diminta untuk mengirimkan informasinya langsung ke *sink*. Telah ditunjukkan dengan menggunakan simulasi bahwa LEACH mengungguli protokol *routing* konvensional, termasuk transmisi langsung dan *routing multihop*, *routing* transmisi energi yang minimum, dan algoritma *routing* berbasis *cluster* statis (Sohraby, Minoli, & Znati, 2007).

2.4 Hybrid Energy-Efficient Distributed (HEED)

Hybrid Energy-Efficient Distributed (HEED), *hybrid*, hemat energi, menggunakan pendekatan *clustering* terdistribusi untuk *ad hoc sensor networks*. HEED mengatasi keterbatasan LEACH yang memungkinkan *cluster head* lebih dari satu *hop* dari *sink node*. Ini menjamin konektivitas *intercluster*, dan karena itu masalah skalabilitas LEACH dipecahkan dengan membangun struktur lapisan yang

lebih tinggi yang mampu mengarahkan data dari *cluster* ke *cluster* sampai ke *sink*. Struktur lapisan yang lebih tinggi ini dapat memanfaatkan skema *routing*, sehingga akhirnya data mencapai *sink node*.



Gambar 2.10 *Hybrid Energy-Efficient Distributed (HEED)*
(Labrador & Wightman, 2009)

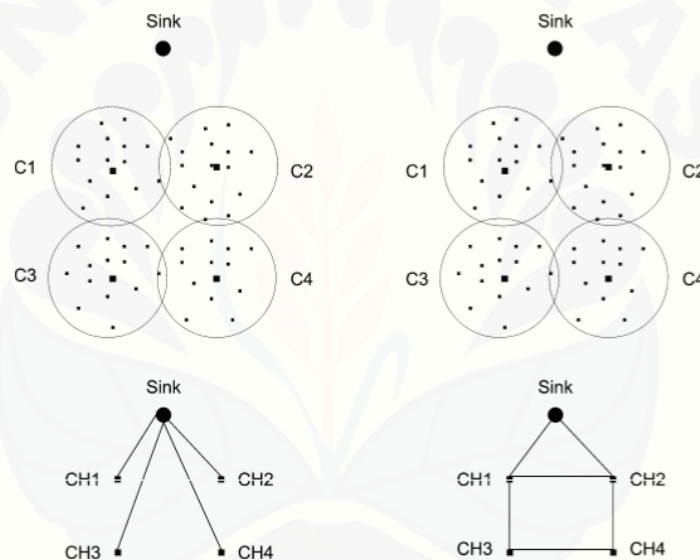
Seperti dalam LEACH, HEED memilih *cluster head* dan mengatur *cluster* secara berkala. Salah satu pilihan di HEED adalah tersedianya *node* dengan beberapa level daya, dimana *node* menggunakan tingkat transmisi yang lebih kecil, sebanding dengan sel persegi dengan ukuran c , untuk menciptakan *cluster* dan meningkatkan *spatial reuse*, dan rentang transmisi yang lebih tinggi $(1 + \sqrt{5})c$ untuk menjamin konektivitas *intercluster*. Berdasarkan hal tersebut, tingkat transmisi *intracluster* menentukan jumlah *cluster* dalam jaringan.

Karena tujuan utama HEED adalah untuk memperpanjang masa pakai jaringan, metrik pemilihan *cluster head* terutama didasarkan pada energi residu setiap *node*. *Node* awalnya menghitung probabilitas mereka menjadi *cluster head*, CH_{prob} , menggunakan persamaan

$$CH_{prob} = C_{prob} \times \frac{E_{residual}}{E_{max}} \quad (2.3)$$

dimana C_{prob} adalah jumlah *cluster head* yang telah ditentukan sebelumnya dalam jaringan (seperti LEACH), $E_{residual}$ adalah perkiraan sisa energi residu *node* dan E_{max}

adalah nilai energi referensi yang sesuai dengan jumlah energi yang dimiliki *node* saat terisi penuh. Dengan probabilitas ini, *node* terpilih menjadi bagian dari rangkaian *cluster heads* yang mungkin, S_{CH} . Kemudian, *node* memilih *cluster heads* mereka dari himpunan S_{CH} sesuai dengan fungsi biaya, yang bisa berbeda sesuai dengan tujuan perancang jaringan atau operator jaringan. Biaya dapat didasarkan pada derajat *node* dari *nodes* dan jumlah daya yang akan digunakan oleh semua *node* dalam *cluster* jika *node* tertentu dipilih sebagai *cluster head*, jika tujuannya adalah untuk mendistribusikan muatan secara merata; atau bisa jadi kebalikan dari tingkat *node* jika *cluster* padat diinginkan.



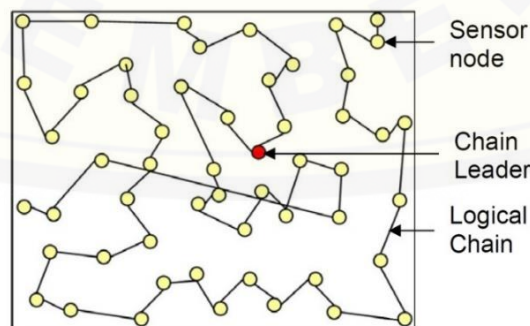
Gambar 2.11 LEACH and HEED clustering protocols
(Labrador & Wightman, 2009)

Gambar 2.11 menunjukkan protokol LEACH dan HEED dan jenis topologi lapisan yang lebih tinggi yang masing-masing dapat dibuat. Dengan metrik pengelompokan berbasis energi dan kemampuan komunikasi *intercluster*, HEED dapat menggabungkan *node* dengan cakupan yang berbeda dan skala lebih baik daripada LEACH. Baik LEACH dan HEED memiliki pesan $O(n)$ dan kompleksitas komputasional (Labrador & Wightman, 2009).

2.5 Power-Efficient Gathering in Sensor Information Systems (PEGASIS)

Power-Efficient Gathering in Sensor Information Systems (PEGASIS) dan perluasannya, PEGASIS hirarkis, adalah keluarga protokol *routing* dan pengumpulan informasi untuk WSN. Tujuan utama PEGASIS ada dua. Pertama, protokol ini bertujuan untuk memperpanjang masa pakai jaringan dengan mencapai tingkat efisiensi energi yang tinggi dan konsumsi energi yang seragam di semua *node* jaringan. Kedua, protokol ini berusaha untuk mengurangi penundaan data yang masuk dalam perjalanan ke *sink* (Sohraby, Minoli, & Znati, 2007).

Model jaringan yang dipertimbangkan oleh PEGASIS mengasumsikan kumpulan *node* homogen yang ditempatkan di wilayah geografis. *Node* diasumsikan memiliki pengetahuan global tentang posisi sensor lainnya. Selanjutnya, mereka memiliki kemampuan untuk mengendalikan kekuatan mereka untuk menutupi rentang *arbitrary*. *Node* juga bisa dilengkapi dengan *transceiver* radio berkemampuan CDMA. Tanggung jawab *node* adalah mengumpulkan dan mengirimkan data ke *sink*, biasanya sebuah *base station* nirkabel. Tujuannya adalah untuk mengembangkan struktur *routing* dan skema pengumpulan untuk mengurangi konsumsi energi dan memberikan data agregat ke *base station* dengan penundaan minimal sambil menyeimbangkan konsumsi energi di antara *sensor node*. Berlawanan dengan protokol lain, yang bergantung pada struktur pohon atau organisasi hirarkis *cluster* berbasis jaringan untuk pengumpulan dan penyebaran data, PEGASIS menggunakan struktur rantai.



Gambar 2.12 Struktur rantai pada PEGASIS

(NCBI, 2012)

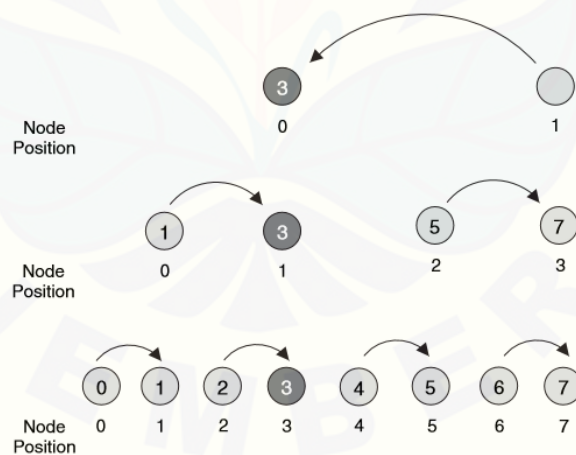
Berdasarkan struktur ini, *node* berkomunikasi dengan tetangga terdekat mereka. Konstruksi rantai dimulai dengan *node* terjauh dari *sink*. *Node* jaringan ditambahkan ke rantai secara progresif, mulai dari tetangga terdekat hingga *node* akhir. *Node* yang saat ini berada di luar rantai ditambahkan ke rantai dengan cara *greedy*, tetangga terdekat ke *node* teratas di rantai pertama, sampai semua *node* disertakan. Untuk menentukan tetangga terdekat, sebuah *node* menggunakan kekuatan sinyal untuk mengukur jarak ke semua *node* tetangganya. Dengan menggunakan informasi ini, *node* menyesuaikan kekuatan sinyal sehingga hanya *node* terdekat yang bisa didengar.

Sebuah *node* dalam rantai dipilih untuk menjadi pemimpin rantai. Tanggung jawabnya adalah mentransmisikan data gabungan ke *base station*. Peran pemimpin rantai bergeser dalam memosisikan rantai setelah setiap putaran. Putaran dapat dikelola oleh *data sink*, dan transisi dari satu putaran ke putaran berikutnya oleh *data sink*. Rotasi peran kepemimpinan di antara simpul rantai memastikan rata-rata konsumsi energi seimbang di antara semua simpul jaringan. Perlu dicatat, bagaimanapun, bahwa *nodes* yang mengasumsikan peran kepemimpinan rantai mungkin lebih bertanggung jawab daripada *data sink*. *Node* tersebut mungkin diperlukan untuk mentransmisikan dengan daya tinggi untuk mencapai *base station*.

Pengumpulan data di PEGASIS dicapai di sepanjang rantai. Dalam bentuknya yang paling sederhana, proses pengumpulan dapat dilakukan secara berurutan sebagai berikut. Pertama, pemimpin rantai mengeluarkan tanda ke *node* terakhir di ujung kanan rantai. Setelah menerima tanda, *node* akhir mentransmisikan datanya ke tetangga hilirnya di rantai menuju pemimpin. *Node* tetangga menggabungkan data dan mentransmisikannya ke tetangga hilirnya. Proses ini berlanjut sampai data gabungan mencapai pemimpin. Setelah menerima data dari sisi kanan rantai, pemimpin mengeluarkan tanda ke ujung kiri rantai, dan proses pengumpulan yang sama dilakukan sampai data mencapai pemimpin. Setelah menerima data dari kedua sisi rantai, pemimpin mengumpulkan data dan mengirimkannya ke *data sink*. Meskipun sederhana, skema pengumpulan berurutan

dapat mengakibatkan penundaan yang lama sebelum data gabungan dikirim ke *base station*. Skema sekuensial semacam itu mungkin diperlukan jika transmisi simultan secara bersamaan tidak dapat dilakukan tanpa gangguan sinyal.

Pendekatan potensial untuk mengurangi penundaan yang dibutuhkan untuk mengirimkan data gabungan ke *sink* adalah dengan menggunakan penggabungan data paralel sepanjang rantai. Tingkat paralelisme yang tinggi dapat dicapai jika node sensor dilengkapi dengan *transceiver* berkemampuan CDMA. Kemampuan tambahan untuk melakukan transmisi *arbitrarily close* tanpa gangguan dapat digunakan untuk "overlay" struktur hirarki ke rantai dan menggunakan struktur tertanam untuk melakukan pengumpulan data. Pada setiap putaran, *node* pada tingkat hirarki tertentu mengirimkan ke tetangga dekat di tingkat atas hirarki. Proses ini berlanjut sampai data gabungan mencapai pemimpin di tingkat atas hirarki. Yang terakhir mentransmisikan gabungan data terakhir ke *base station*.



Gambar 2.13 *Chain-based data gathering and aggregation scheme*

(Sohraby, Minoli, & Znati, 2007)

Untuk menggambarkan pendekatan berbasis rantai, perhatikan contoh yang digambarkan pada Gambar 2.13. Dalam contoh ini diasumsikan bahwa semua *node* memiliki pengetahuan global tentang jaringan dan menggunakan algoritma

greedy untuk membangun rantai. Selanjutnya, diasumsikan bahwa *node* bergiliran mentransmisikan ke *base station* sehingga *node* $i \bmod N$, di mana N mewakili jumlah total *node*, bertanggung jawab untuk mentransmisikan data gabungan ke *base station* di putaran i . Berdasarkan tugas ini, *node* 3, pada posisi 3 dalam rantai, adalah pemimpin dalam putaran 3. Semua *node* dalam posisi genap harus mengirim datanya ke tetangga mereka ke kanan. Pada tingkat berikutnya, *node* 3 tetap dalam posisi ganjil. Akibatnya, semua *node* dalam posisi genap mengumpulkan data mereka dan mengirimkannya ke tetangga kanan mereka. Pada tingkat ketiga, *node* 3 tidak lagi dalam posisi ganjil. *Node* 7, satu-satunya simpul di samping *node* 3 naik ke tingkat ini, menggabungkan data dan mengirimnya ke *node* 3. *Node* 3, pada gilirannya, menggabungkan data yang diterima dengan datanya sendiri dan mengirimkannya ke *base station*.

Pendekatan biner berbasis rantai mengarah pada pengurangan energi yang signifikan, karena *node* beroperasi dengan cara yang sangat paralel. Selanjutnya, karena struktur hierarkis dan struktur seperti *tree* seimbang, skema ini menjamin bahwa setelah langkah-langkah $\log_2 N$, data gabungan sampai pada pemimpin. Skema agregasi biner berbasis rantai telah digunakan di PEGASIS sebagai alternatif untuk mencapai tingkat paralelisme yang tinggi. Dengan *sensor node* berkemampuan CDMA, telah ditunjukkan bahwa skema ini bekerja paling baik sehubungan dengan produk *delay* energi yang dibutuhkan per putaran pengumpulan data, sebuah metrik yang menyeimbangkan biaya energi dan penundaan.

Skema sekuensial dan skema paralel berbasis CDMA merupakan dua titik akhir dari spektrum desain. Skema ketiga, yang tidak memerlukan *node transceiver* untuk dilengkapi kemampuan CDMA, menyerang keseimbangan antara dua skema ekstrim dan mencapai beberapa tingkat paralelisme. Ide dasar dari skema ini adalah membatasi transmisi simultan ke *node* yang dipisahkan secara spasial. Berdasarkan batasan ini, hirarki PEGASIS menciptakan hirarki tiga tingkat di mana jumlah *node* jaringan dibagi menjadi tiga kelompok. Data digabungkan secara bersamaan dalam setiap kelompok dan dipertukarkan antar kelompok. Data dikumpulkan akhirnya mencapai pemimpin, yang mengirimkannya ke *data sink*. Perlu dicatat bahwa transmisi simultan harus dijadwalkan secara hati-hati untuk menghindari gangguan.

Selanjutnya, hierarki tiga tingkat harus direstrukturisasi dengan benar untuk memungkinkan rotasi kepemimpinan di antara *node* kelompok.

Hasil simulasi perpanjangan hirarki PEGASIS menunjukkan peningkatan yang cukup besar atas skema seperti LEACH. Selanjutnya, skema hirarkis telah terbukti mengungguli skema PEGASIS asli dengan faktor 60 (Sohraby, Minoli, & Znati, 2007).

2.6 Parameter Kerja

a. *Electronic Energy* (E_{elec})

Energi yang digunakan untuk mengoperasikan sirkuit elektronik pada sisi pengirim maupun penerima dapat dihitung dengan menggunakan persamaan (Darbi, Rahayu, & Sari, 2014)

$$E_{elec} = \frac{V \times I}{R_b} \quad (2.4)$$

Dimana V merupakan tegangan dengan I sebagai arus pancar pada *electronic energy transceiver* dan arus terima pada *electronic energy receiver*. Sedangkan R_b adalah *bit rate*.

b. *Amplify Energy* (E_{amp})

Gelombang elektromagnetik dapat dipropagasikan dengan dua cara pada medium udara yaitu, secara *free space loss* dan *multipath*. Untuk menghitung nilai ϵ_{fs} dan ϵ_{mp} , dilakukan konversi untuk *gain* (G) dan sensitivitas (S). λ adalah panjang gelombang dan h merupakan tinggi antena. Kondisi LOS (*Line of Sight*) terjadi apabila di antara pengirim dan penerima tidak terdapat penghalang. (Darbi, Rahayu, & Sari, 2014)

$$\epsilon_{fs} = \frac{S \times (4\pi)^2}{R_b \times G_t \times G_r \times \lambda^2} \quad (2.5)$$

Sedangkan apabila ada penghalang, maka pola yang digunakan adalah *multipath*

$$\epsilon_{mp} = \frac{S}{R_b \times G_t \times G_r \times h_t^2 \times h_r^2} \quad (2.6)$$

Untuk membedakan penggunaan ϵ_{fs} dan ϵ_{mp} pada simulasi, digunakan

perhitungan jarak pengirim dan penerima

$$d_c = \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}} \quad (2.7)$$

Jika jaraknya lebih kecil atau sama dengan d_c ($d \leq d_c$) sinyal dipropagasikan *free space loss*, sebaliknya dipropagasikan secara *multipath* ($d > d_c$).

2.7 MATLAB (*Matrix Laboratory*)

MATLAB (*Matrix Laboratory*) adalah sebuah lingkungan komputasi numerikal dan bahasa pemrograman komputer generasi keempat. Dikembangkan oleh The MathWorks, MATLAB memungkinkan manipulasi matriks, pem-plot-an fungsi dan data, implementasi algoritma, pembuatan antarmuka pengguna, dan peng-antarmuka-an dengan program dalam bahasa lainnya. Meskipun hanya bernuansa numerik, sebuah kotak kakas (*toolbox*) yang menggunakan mesin simbolik MuPAD, memungkinkan akses terhadap kemampuan aljabar komputer. Sebuah paket tambahan, Simulink, menambahkan simulasi grafis multiranah dan Desain Berdasar-Model untuk sistem terlekat dan dinamik (Wikipedia, 2017).



Gambar 2.14 Logo MATLAB
(Wikipedia, 2017)

BAB 3. METODOLOGI PENELITIAN

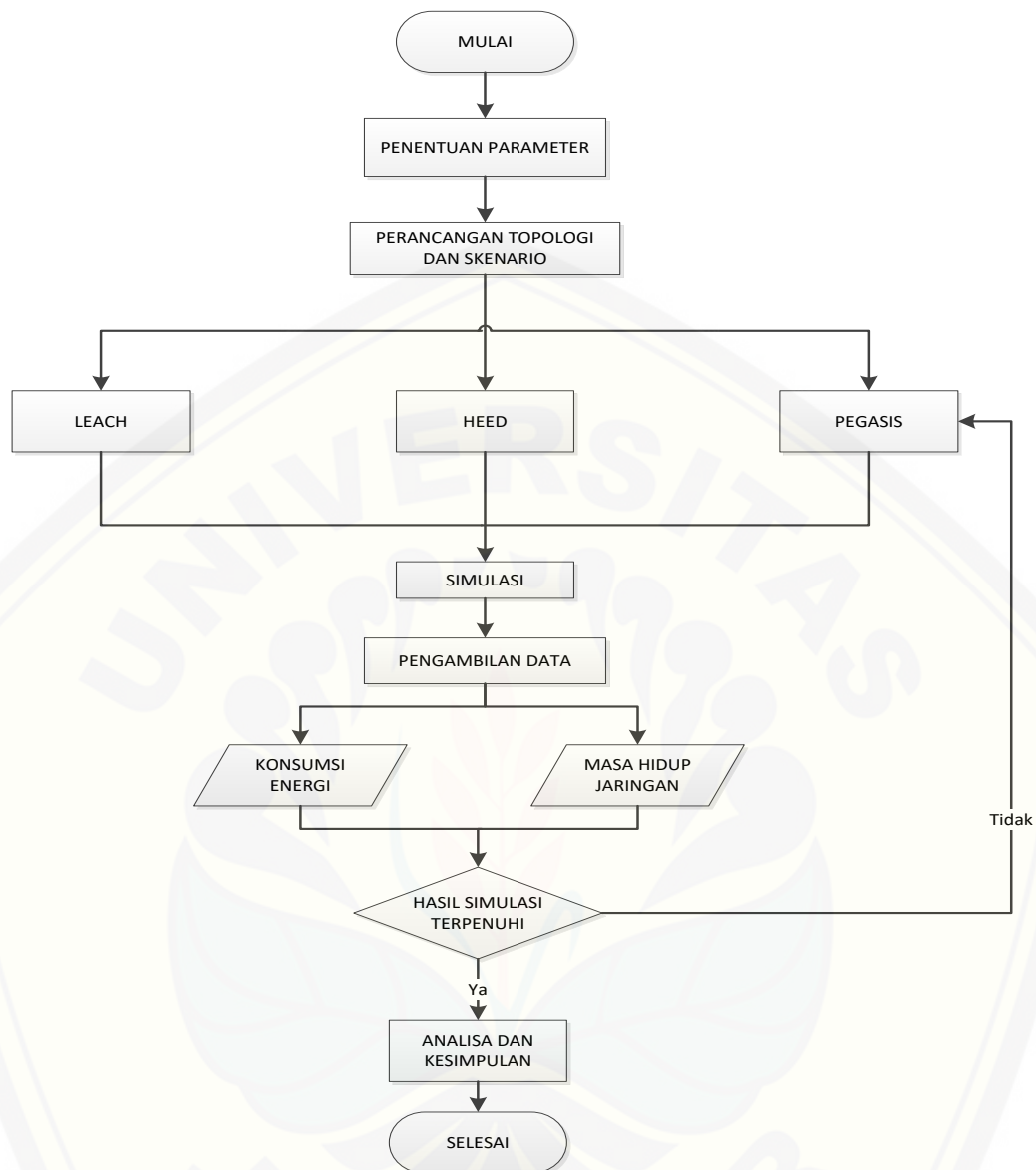
Pada penelitian ini akan dilakukan analisis konsumsi energi dan masa hidup jaringan pada *Wireless Sensor Network* (WSN) menggunakan beberapa protokol. Protokol yang digunakan antara lain, *Low Energy Adaptive Clustering Hierarchy* (LEACH), *Hybrid Energy-Efficient Distributed* (HEED), dan *Power-Efficient Gathering in Sensor Information Systems* (PEGASIS). Untuk menguji performansi dari LEACH, HEED, dan PEGASIS, akan dilakukan kombinasi dari jumlah *node* yang digunakan serta energi awal yang diberikan pada tiap *node*.

3.1 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan di Fakultas Teknik, Universitas Jember yang beralamat di Jalan Kalimantan No. 37, Kampus Tegalboto, Jember 68121. Waktu penelitian selama lebih kurang 5 bulan, dimulai pada bulan November 2017.

Tabel 3.1 Jadwal Pelaksanaan Penelitian

No	Kegiatan	Bulan/Minggu																			
		Bulan ke-1				Bulan ke-2				Bulan ke-3				Bulan ke-4				Bulan ke-5			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Studi literatur																				
2	Perancangan algoritma protokol LEACH, HEED, dan PEGASIS																				
3	Simulasi dan pengujian algoritma																				
4	Pengujian simulasi dengan																				



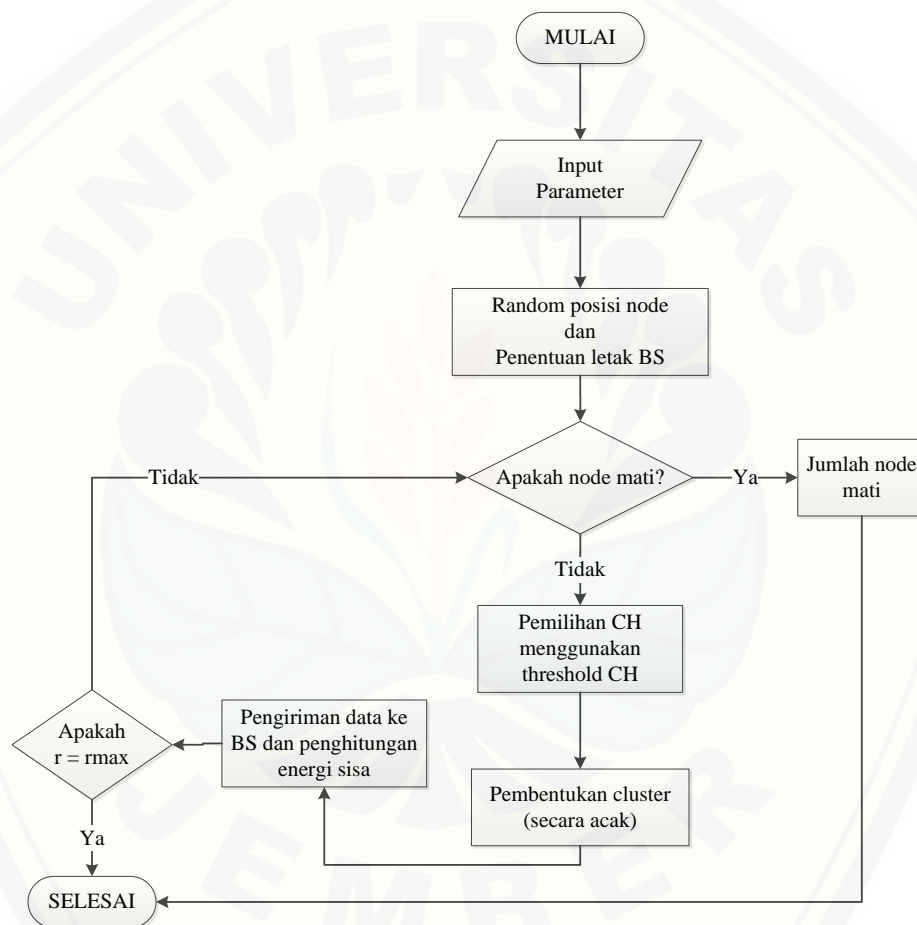
Gambar 3.1 Tahapan Pengumpulan Data

Berdasarkan tahapan di atas, setelah dilakukan perancangan menggunakan protokol LEACH, HEED, dan PEGASIS pada *Wireless Sensor Network*, selanjutnya dilakukan skenario pengujian. Pada skenario tersebut, parameter yang akan diubah yaitu jumlah *node* dan energi awal. Kemudian dilakukan simulasi dan dilihat apakah simulasi yang dijalankan telah sesuai harapan, yaitu banyaknya *round*, data telah terkirim ke *Base Station*, dan diketahui energi yang digunakan serta sisa energinya. Apabila simulasi tidak sesuai, maka simulasi diulang lagi.

Tahapan terakhir adalah pengambilan data untuk dianalisis konsumsi energi serta masa hidup jaringan dari hasil simulasi yang telah dijalankan.

3.3.1 LEACH

Berikut adalah proses penentuan *Cluster Head* pada protokol LEACH yang ditunjukkan oleh *flowchart* di bawah ini.



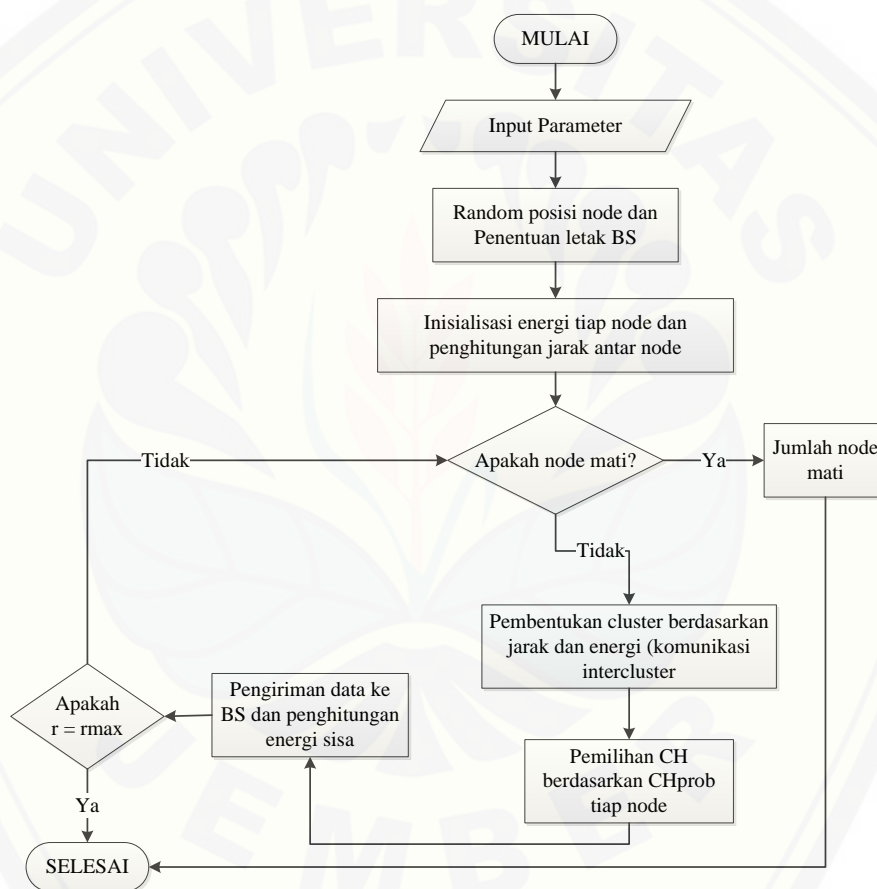
Gambar 3.2 Tahapan protokol LEACH

Berdasarkan tahapan pada diagram alir, dimulai dengan menentukan *cluster* dan *cluster head* yang diinginkan dan masa aktif *node* selama menjadi *cluster head*. Setiap *node* memutuskan menjadi *cluster head* atau tidak selama sesi tersebut berlangsung. Pengambilan keputusan dilakukan oleh *node n* yang memilih angka acak di antara 0 sampai 1. Apabila angka tersebut kurang dari batas

threshold, maka *node* akan menjadi *cluster head* selama sesi tersebut. Kemudian, anggota *cluster* ditentukan berdasarkan jarak dari setiap *cluster head*, dan akan bergabung dengan *cluster head* yang memiliki jarak paling minimum.

3.3.2 HEED

Berikut adalah penentuan *Cluster Head* pada protokol HEED, yang ditunjukkan oleh *flowchart* dibawah ini.



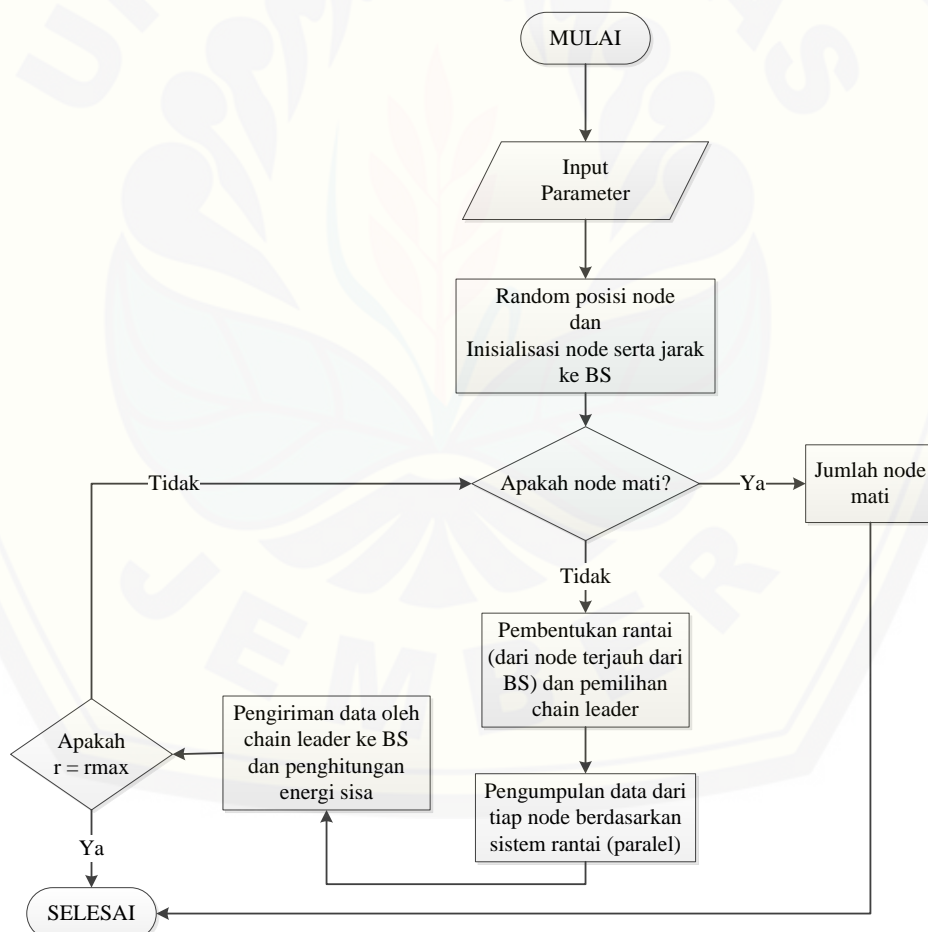
Gambar 3.3 Tahapan protokol HEED

Berdasarkan tahapan pada diagram alir, dimulai dengan menetapkan persentase awal *cluster head* antara semua sensor. Nilai persentase ini, CH_{prob} , digunakan untuk *broadcast cluster head* awal ke sensor lainnya. CH_{prob} tidak boleh kurang dari *threshold* P_{min} , yang berbanding terbalik dengan E_{max} . Bagian utama dari algoritma terdiri dari sejumlah iterasi (konstan). Setiap *sensor node* melewati

iterasi ini sampai menemukan *cluster head* yang dapat *sensor node* tersebut kirimkan dengan daya transmisi paling rendah. Apabila *sensor node* ini mendengar dari *non-cluster head*, *sensor node* memilih dirinya untuk menjadi *cluster head* dan kemudian mengirimkan pesan *broadcast* ke tetangganya untuk menginformasikan perubahan status. Akhirnya, setiap sensor menggandakan nilai CH_{prob} dan menuju ke tahap iterasi berikutnya. Hal ini berhenti ketika CH_{prob} mencapai nilai 1.

3.3.3 PEGASIS

Berikut adalah penentuan *Cluster Head* pada protokol PEGASIS, yang ditunjukkan oleh *flowchart* dibawah ini.



Gambar 3.4 Tahapan protokol PEGASIS

Berdasarkan tahapan di atas, pemimpin rantai berada di urutan hirarki paling atas. Data dikirimkan dari *node* paling ujung ke tetangga hilirnya berdasarkan aturan *node i mod N*. Kemudian apabila telah mencapai pemimpin rantai data dikirimkan ke *Base Station*.

3.4 Parameter Pengujian

Simulasi *Wireless Sensor Network* ini menggunakan *sensor node* yang homogen dengan *node* yang dirancang dalam keadaan tidak bergerak atau statis. Setiap *node* memiliki energi awal yang sama. Seluruh *sensor node* akan disebar secara acak di dalam area berbentuk persegi dengan luas $100 \times 100 \text{ m}^2$. Kemudian, dimasukkan nilai E_{tx} , E_{rx} yang didapatkan dari parameter sensor pada Tabel 3.2 didapat dari *datasheet* Xbee Pro S1.

Tabel 3.2 Parameter Simulasi

Parameter	Nilai
Luas Wilayah	$100 \times 100 \text{ m}^2$
Letak <i>Sink Node</i>	$50 \times 50 \text{ m}^2$
Energi Awal	0,25 <i>Joule</i>
Frekuensi	2,4 Ghz
Sensitifitas	-100 dBm
<i>Bit Rate</i>	250 Kbps
Arus Pancar	250 mA
Arus Terima	55 mA
Tegangan	3,3 V
<i>Gain</i>	2 dBi
Tinggi Antena	0,5 m

a. *Electronic Energy* (E_{elec})

Electronic energy dapat dihitung dengan menggunakan persamaan (2.4). Sehingga didapatkan perhitungan sebagai berikut :

$$E_{elec Transmit} = \frac{3,3 \times 250 \times 10^{-3}}{250 \times 10^3} = 3,3 \mu J/bit$$

$$E_{elec Receive} = \frac{3,3 \times 55 \times 10^{-3}}{250 \times 10^3} = 0,7 \mu J/bit$$

b. *Amplify Energy* (E_{amp})

Untuk menghitung nilai ϵ_{fs} dan ϵ_{mp} , dapat menggunakan persamaan (2.5) dan (6). Sehingga didapatkan perhitungan sebagai berikut :

$$\epsilon_{fs} = \frac{0,1 \times 10^{-12} \times (4\pi)^2}{250 \times 10^3 \times 1,6 \times 1,6 \times 0,125^2}$$

$$\epsilon_{fs} = 1,5 \times 10^{-3} pJ/bit/m^2$$

$$\epsilon_{mp} = \frac{0,1 \times 10^{-12}}{250 \times 10^3 \times 1,6 \times 1,6 \times 0,5^2 \times 0,5^2}$$

$$\epsilon_{mp} = 2,5 \times 10^{-6} pJ/bit/m^4$$

Untuk membedakan penggunaan ϵ_{fs} dan ϵ_{mp} pada simulasi, digunakan perhitungan jarak dengan rumus (2.7) :

$$d_c = \sqrt{\frac{1,5 \times 10^{-15}}{2,5 \times 10^{-18}}} = 10\sqrt{6}$$

3.5 Skenario Pengujian

Simulasi *Wireless Sensor Network* ini menggunakan *sensor node* yang homogen dengan *node* yang dirancang dalam keadaan tidak bergerak atau statis. Setiap *node* memiliki energi awal yang sama. Penomoran *sensor node* dimulai dari 0 sampai dengan N , sedangkan untuk penomoran *base station* adalah $N+1$. Topologi yang digunakan adalah *hierarchical* yaitu dengan metode *cluster-based* untuk LEACH dan HEED, serta *chain-based* pada PEGASIS.

Dalam penelitian ini, digunakan dua buah skenario yaitu perubahan jumlah *node* dan perubahan energi awal. Parameter skenario perubahan *node* dapat dilihat pada Tabel 3.3, sedangkan parameter skenario perubahan energi terdapat pada Tabel 3.4.

Tabel 3.3 Skenario Perubahan Jumlah *Node*

Parameter	Nilai
Luas Area	100 x 100 m ²
Letak <i>Base Station</i>	50 x 50 m ²
Energi Awal	0,25 <i>Joule</i>
Probabilitas	0,1
Jumlah <i>Node</i>	10 <i>node</i> 30 <i>node</i> 50 <i>node</i> 70 <i>node</i> 100 <i>node</i>

Tabel 3.4 Skenario Perubahan Energi Awal

Parameter	Nilai
Luas Area	100 x 100 m ²
Letak <i>Base Station</i>	50 x 50 m ²
Energi Awal	0,1 <i>Joule</i> 0,3 <i>Joule</i> 0,5 <i>Joule</i> 0,7 <i>Joule</i> 0,9 <i>Joule</i>
Probabilitas	0,1
Jumlah <i>Node</i>	100 <i>node</i>

a. Pengujian Konsumsi Energi

Pengujian konsumsi energi akan dilakukan selama 300 *round*. Pada setiap selesai 60 *round*, akan diambil data sisa energi dari setiap *node*. Sehingga, nantinya akan diperoleh perbandingan antara tiga buah protokol yang digunakan, protokol manakah yang memiliki konsumsi energi paling sedikit pada tiap *round*.

b. Pengujian Masa Hidup Jaringan

Pengujian masa hidup jaringan sama dengan pengujian konsumsi energi, baik dalam variasi jumlah *node* maupun variasi energi awal yang digunakan. *Round* yang digunakan juga 300 *round*. Pada akhir *round*, dilihat berapa jumlah *node* yang mati dan pada *round* keberapa *node* pertama kali mati.

BAB 5. PENUTUP

5.1 Kesimpulan

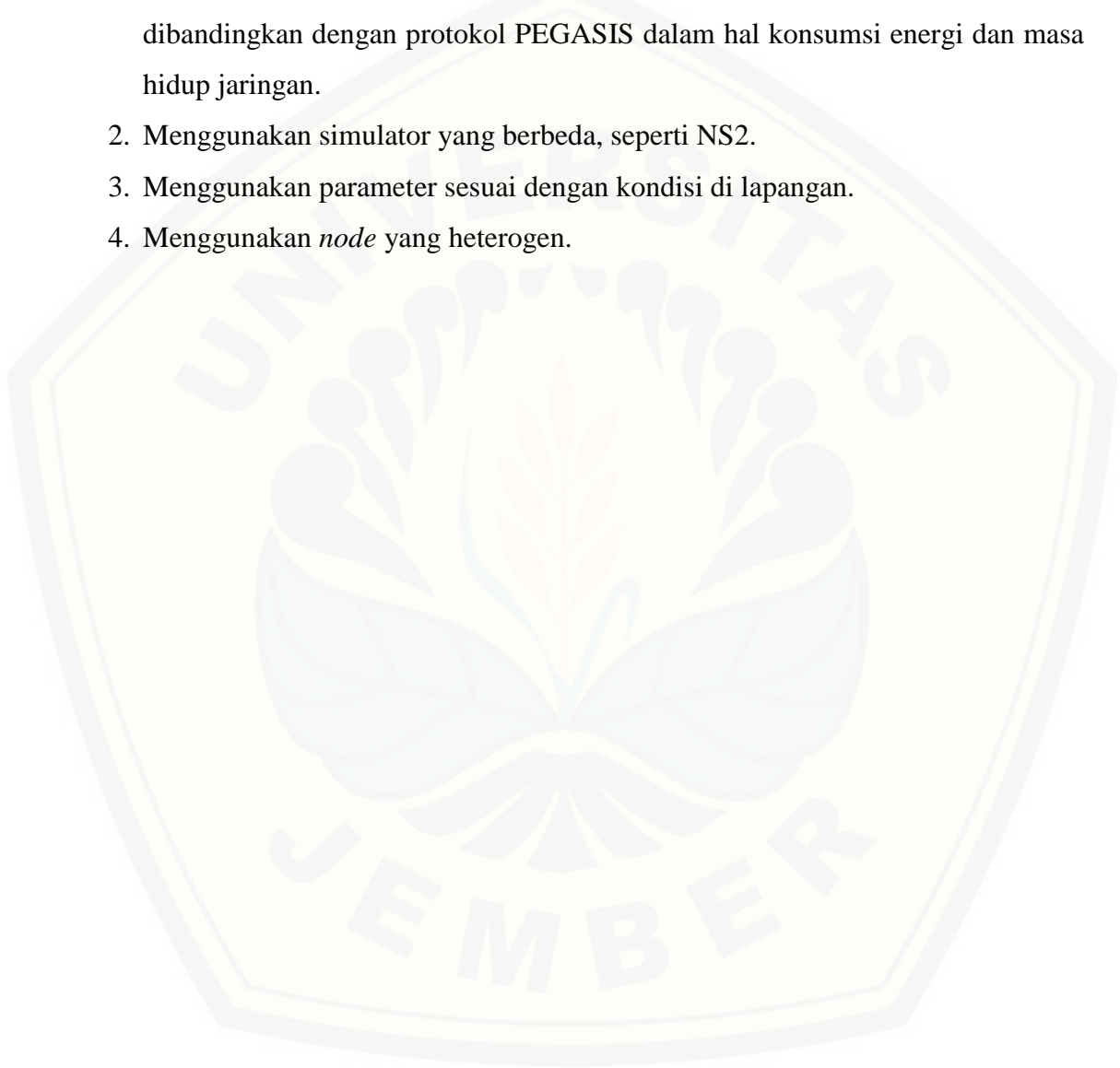
Berdasarkan analisis dari data hasil simulasi protokol pada *Wireless Sensor Network*, maka dapat ditarik beberapa kesimpulan sebagai berikut:

1. Pada protokol LEACH, konsumsi energi dan masa hidup jaringannya paling tidak optimal di antara ketiga protokol yang digunakan. Hal ini dapat terjadi karena pada protokol ini, *cluster head* dipilih secara acak berdasarkan nilai *threshold*. Sehingga, pada setiap *round* terdapat *cluster head* yang berbeda dengan jumlah yang berbeda. Alasan yang kedua, tidak ada batasan jarak *cluster*, sehingga anggota *cluster* dapat terpilih secara acak walaupun jaraknya terlalu jauh dari *cluster head*. Sehingga, energi yang digunakan semakin besar dalam mengirimkan data dan menyebabkan kematian *node* yang lebih banyak.
2. Protokol HEED lebih baik daripada protokol LEACH walaupun menggunakan teknik yang sama yaitu *clustering*. Konsumsi energi dan masa hidup jaringan pada protokol ini lebih baik karena menggunakan radius *cluster*. Radius *cluster* merupakan jarak maksimal antara *cluster head* dengan anggota *cluster*. Dengan adanya radius tersebut, *cluster* tidak seacak pada protokol LEACH karena terdapat *cost intracluster*. Sehingga, konsumsi energinya lebih hemat dan meminimalisir kematian *node*.
3. Penggunaan teknik rantai dapat dikatakan paling efektif apabila dibandingkan dengan teknik *clustering*. Pada saat menggunakan protokol PEGASIS, konsumsi energi yang digunakan sangat hemat dan kematian *node* yang dialami sangat sedikit. Pada protokol ini, semua *node* menjadi satu rantai dan terdapat satu pemimpin rantai. Sehingga, *node* paling ujung mengirimkan data ke *node* tetangga hingga sampai ke pemimpin rantai untuk dikirimkan ke *Base Station*. Hal ini dapat menghemat energi yang dikeluarkan, karena data dikirimkan ke tetangga yang paling dekat. Jadi, semakin banyak jumlah *node* yang digunakan maka semakin hemat energi yang digunakan, karena jarak antar *node* semakin rapat.

5.2 Saran

Dari hasil analisis perbandingan ketiga protokol yaitu LEACH, HEED, dan PEGASIS yang telah dilakukan, terdapat beberapa saran sehingga penelitian ini dapat dikembangkan.

1. Menggunakan protokol yang berbeda, terutama dengan teknik rantai agar dapat dibandingkan dengan protokol PEGASIS dalam hal konsumsi energi dan masa hidup jaringan.
2. Menggunakan simulator yang berbeda, seperti NS2.
3. Menggunakan parameter sesuai dengan kondisi di lapangan.
4. Menggunakan *node* yang heterogen.



DAFTAR PUSTAKA

- Aufar, M. B. (2017). *ANALISIS SIMULASI ROUTING PROTOKOL HIERARKIAL LEACH DAN PEGASIS PADA WIRELESS SENSOR NETWORK*. Bandung: Telkom University.
- Bansal, P., Kundu, P., & Kaur, P. (2014). Comparison of LEACH and PEGASIS Hierarchical Routing Protocol in Wireless Sensor Networks. *Int. J. of Recent Trends in Engineering & Technology*.
- Darbi, A., Rahayu, Y., & Sari, L. O. (2014). Analisa Efisiensi Energi Algoritma Routing Low-Energy Adaptive Clustering Hierarchy (LEACH) pada Wireless Sensor Network Menggunakan MATLAB. *Jom FTEKNIK*.
- Dargie, W., & Poellabauer, C. (2010). *Fundamentals of Wireless Sensor Networks*. United Kingdom: John Wiley & Sons Ltd.
- Galshetwar, V. M., & Jeyakumar, A. (2014). Energy Efficient and Reliable Clustering Algorithms HEED and ADCP of Wireless Sensor Networks : A Comparative Study. *International Conference on Communication and Signal Processing*.
- Gupta, A., & Nayyar, A. (2014). A Comprehensive Review of Cluster-Based Energy Efficient Routing Protocols in Wireless Sensor Networks . *International Journal of Research in Computer and Communication Technology*.
- Homaei, M. H., Amiri, M., Habibi, M., & Pirtaj, R. (2015, 7). *WSNLAB.IR*. Retrieved from <http://wsnlab.org/downloads/>
- Labrador, M. A., & Wightman, P. M. (2009). *Topology Control in Wireless Sensor Network*. USA: Springer Science and Business Media.
- Mukarromah, A. M. (2014). *Analisa Konsumsi Daya pada Wireless Sensor Network Menggunakan Algoritma LEACH*. Bandung: Telkom University.
- NCBI. (2012, November). Retrieved from <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3522945/>
- Prabowo, S. (2014). *Enhanced SIMplified Hybrid Energy-Efficient Distributed Clustering for Wireless Sensor Network*. Bandung: Telkom University.
- Rijal, A. B., Kristalina, P., & Santoso, T. (n.d.). *Simulasi Komunikasi Multihop pada Jaringan Sensor Nirkabel Menggunakan Algoritma HLEACH*. Surabaya: ITS.
- Sohraby, K., Minoli, D., & Znati, T. (2007). *Wireless Sensor Networks : Technology, Protocols, and Applications*. Canada: John Wiley & Sons, Inc.

Wikipedia. (2017, Januari 22). *Wikipedia*. Retrieved from Wikipedia Ensiklopedia Bebas:

<https://id.wikipedia.org/w/index.php?title=MATLAB&action=history>

Wulansari, F. (2016). *Analisis Performansi Jaringan Mpls-Te Fast Reroute Menggunakan Metode Qos Diffserv Dengan Openimscore Sebagai Server Penyedia Layanan*. Bandung: Telkom University.

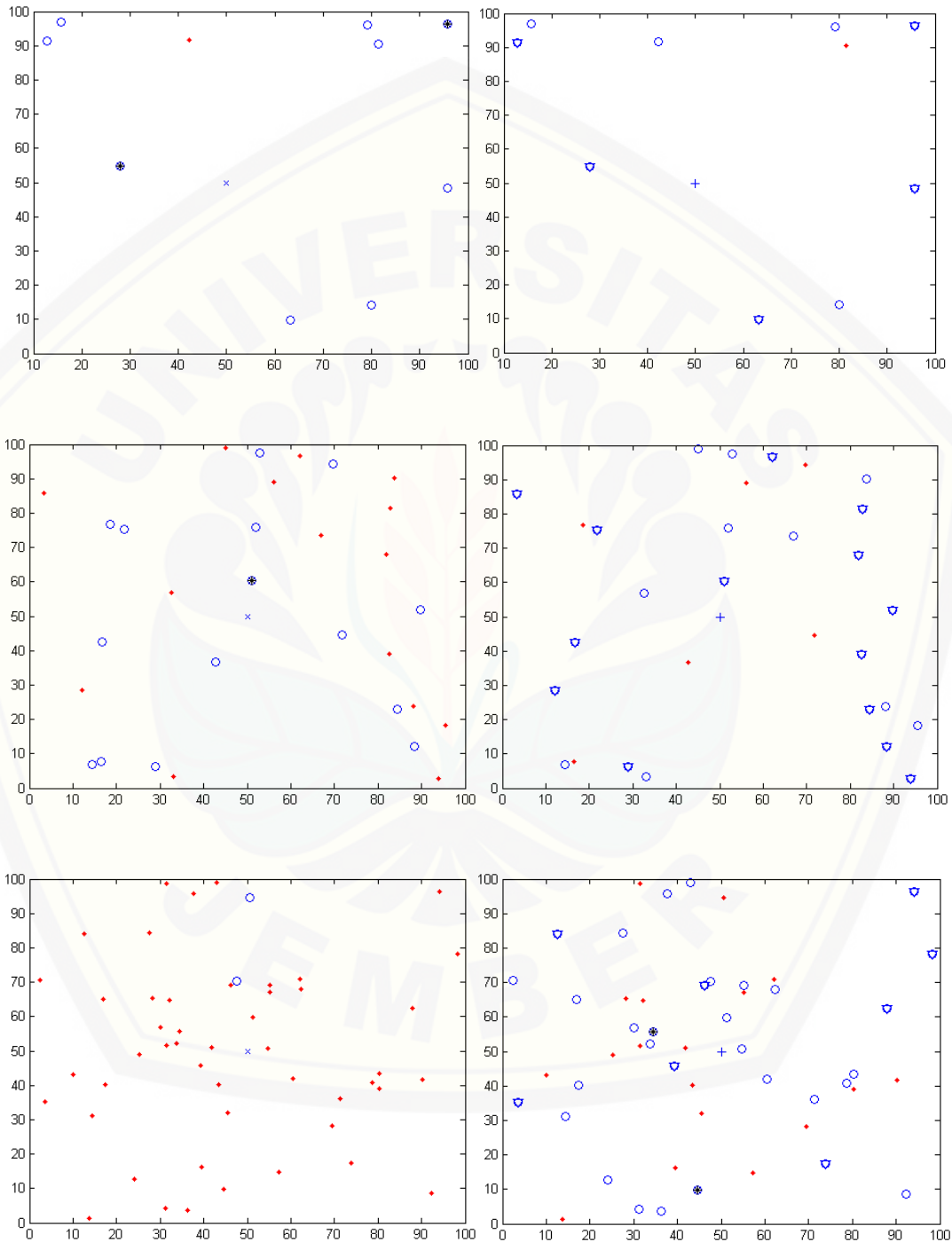


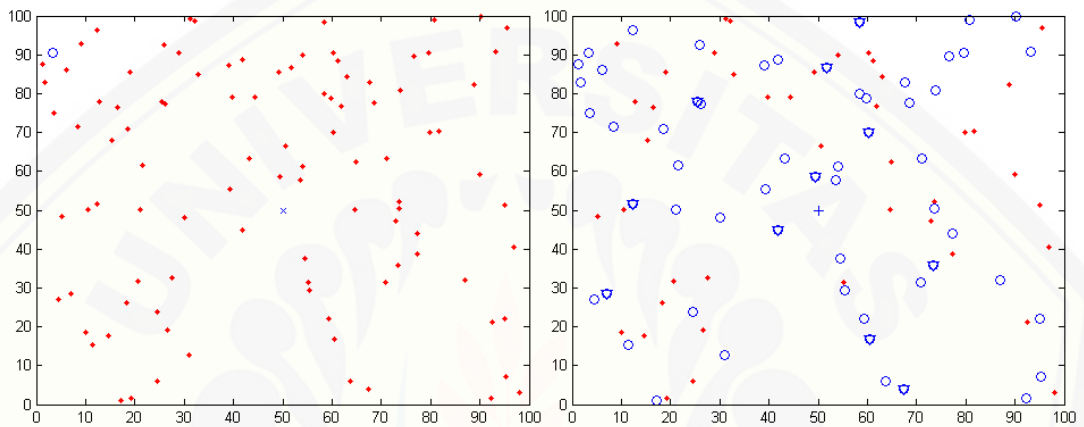
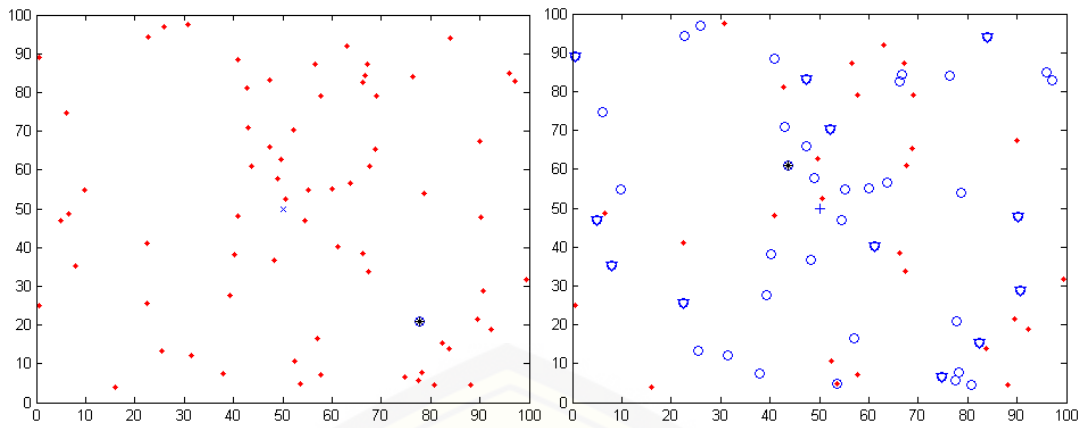
LAMPIRAN

A. Hasil Simulasi

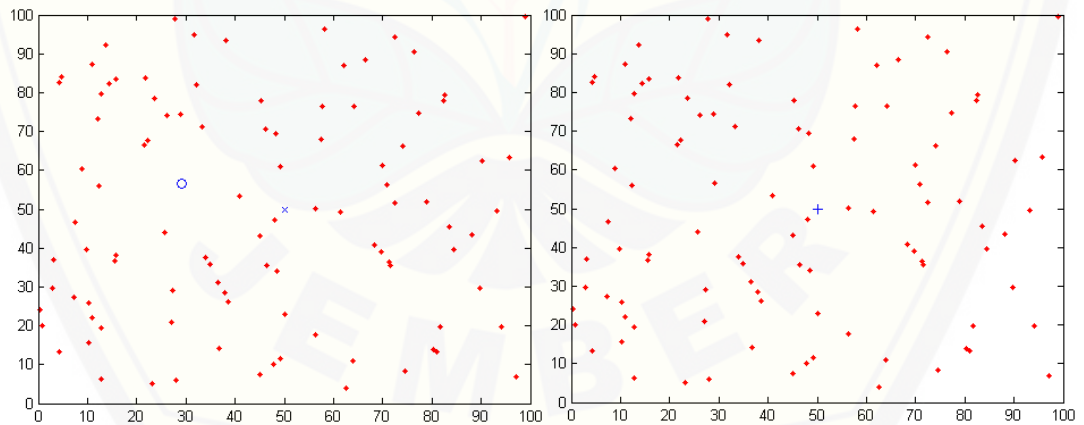
1. LEACH dan HEED

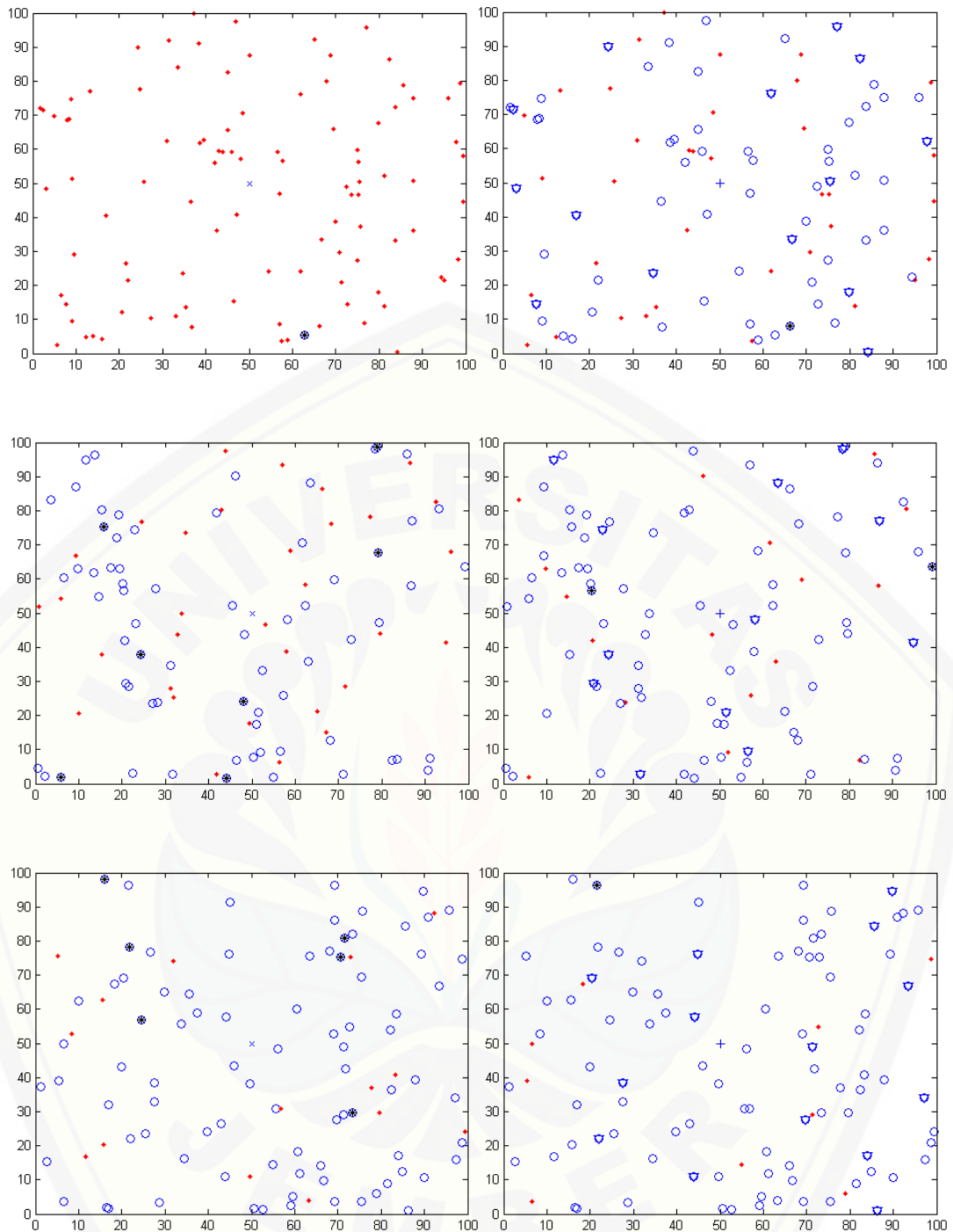
a) Variasi Jumlah *Node* (10, 30, 50, 70, dan 100)

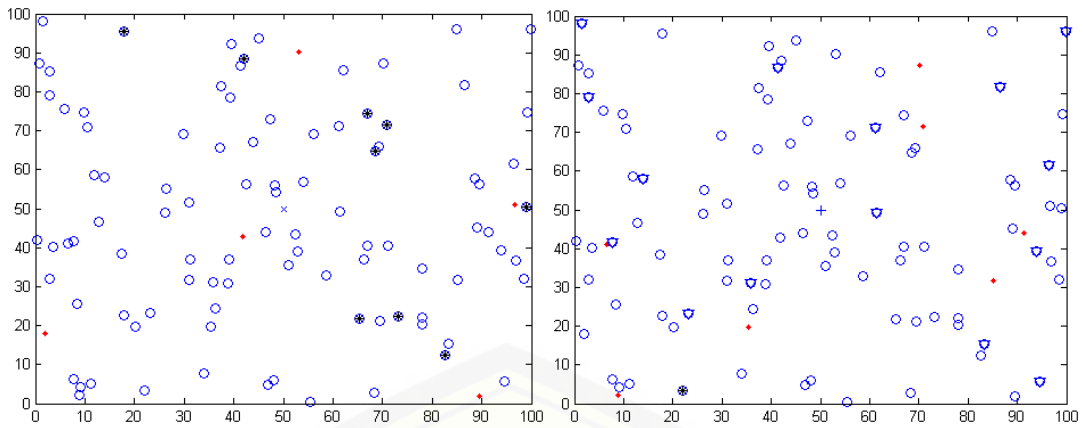




b) Variasi Energi Awal (0,1; 0,3; 0,5; 0,7; dan 0,9)

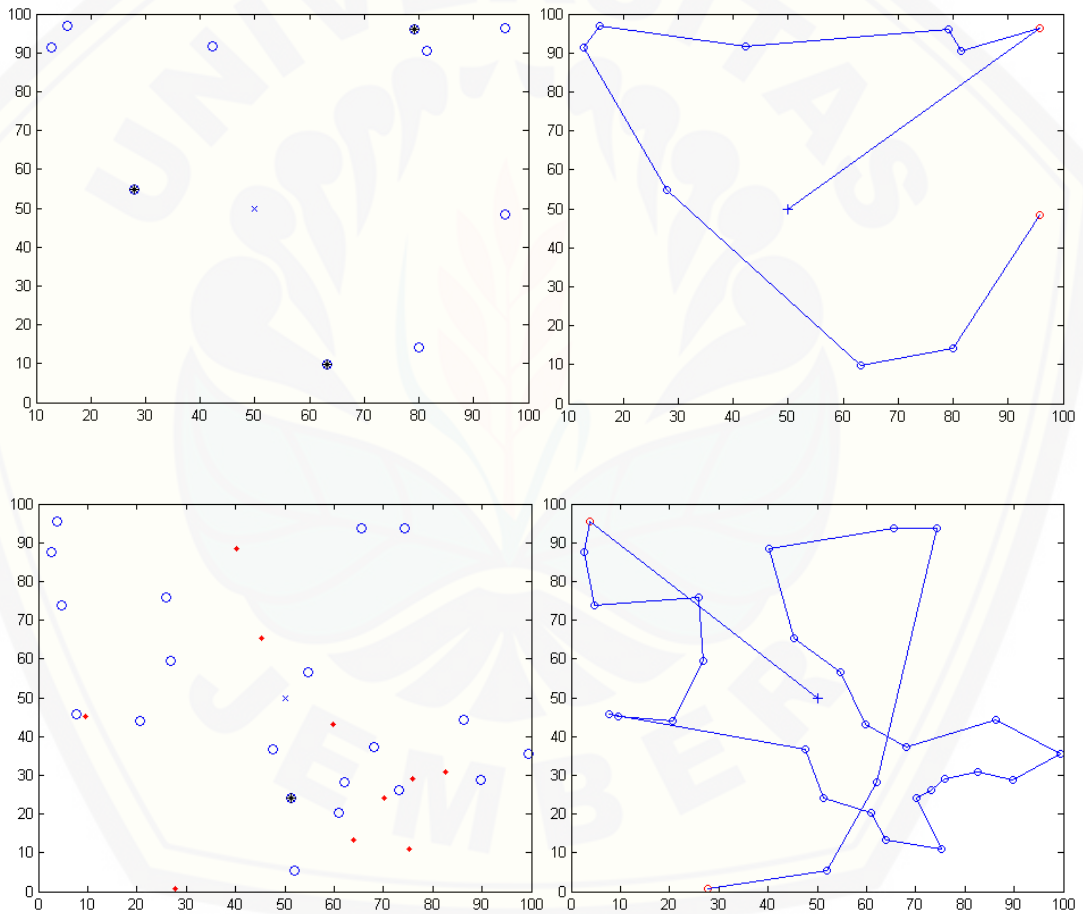


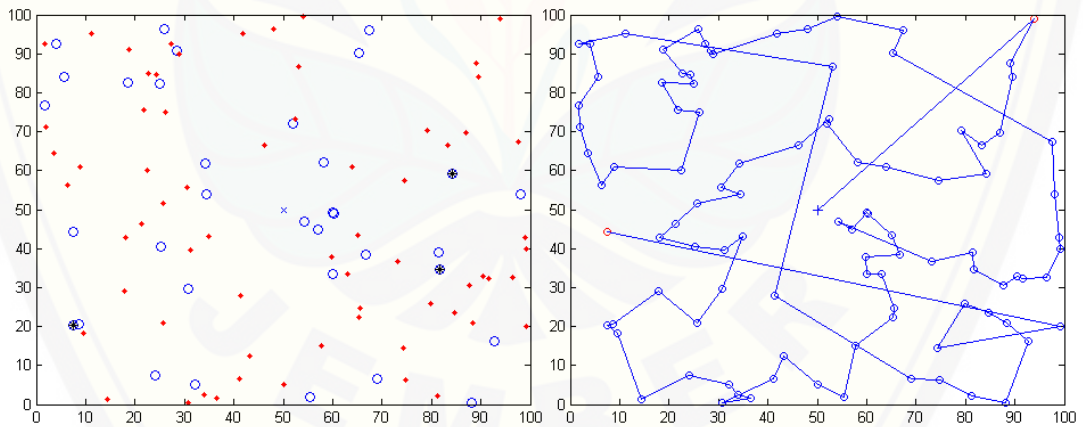
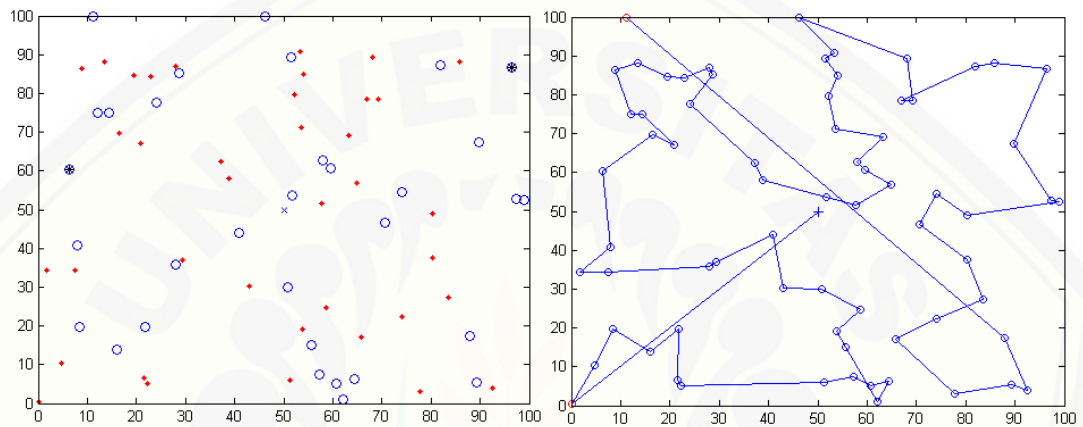
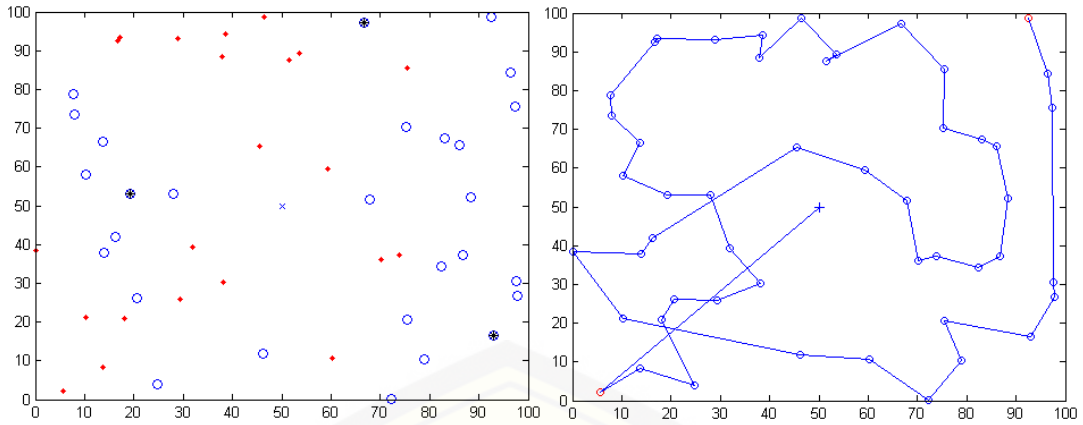




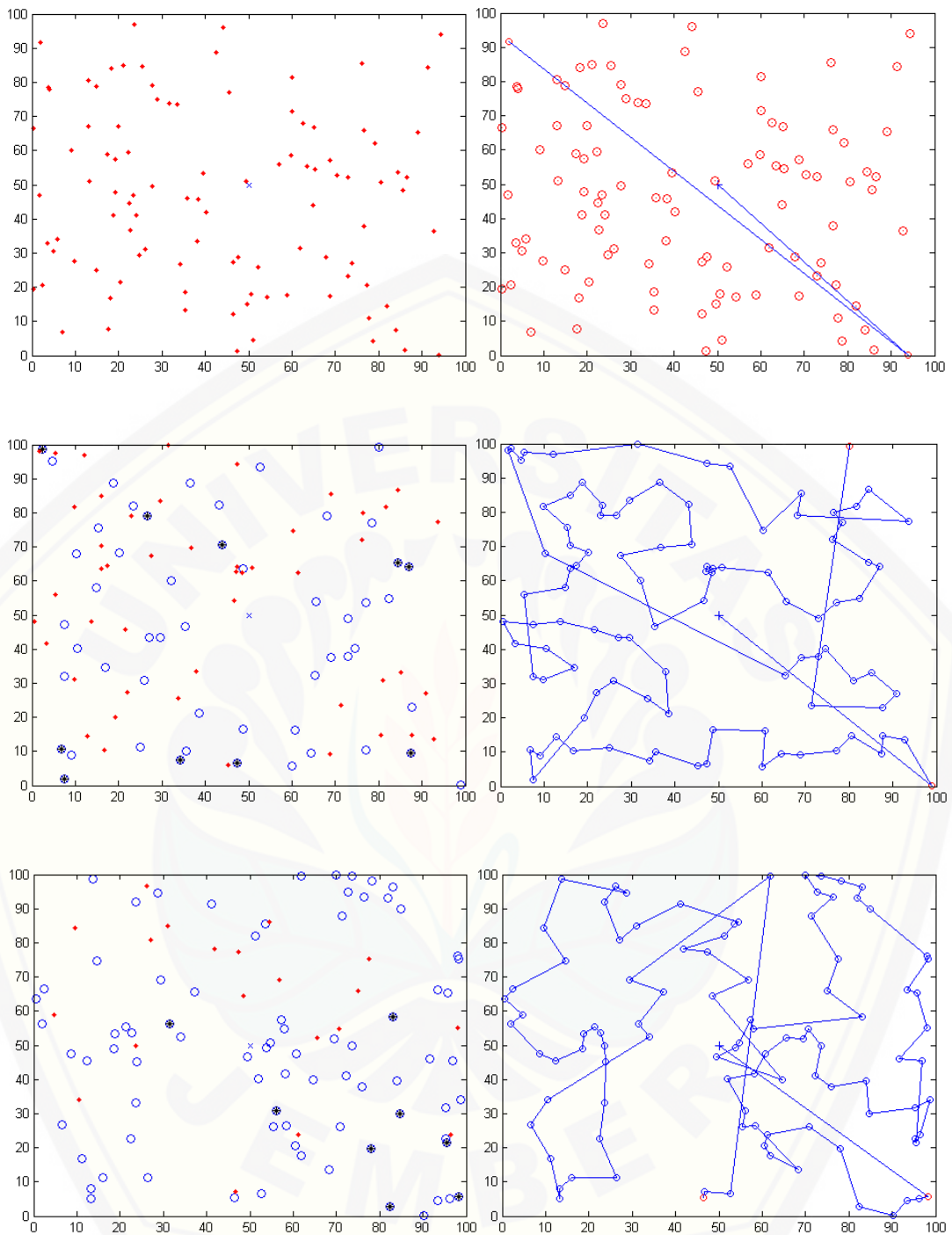
2. LEACH dan PEGASIS

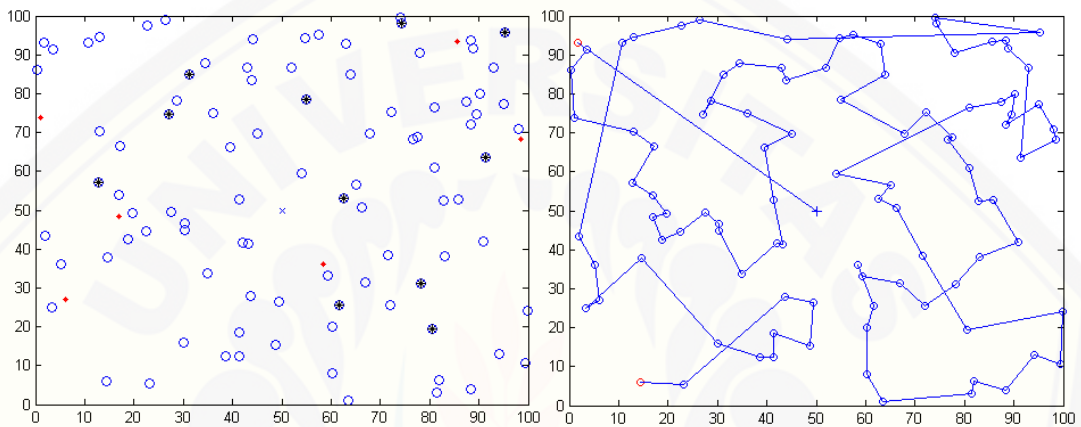
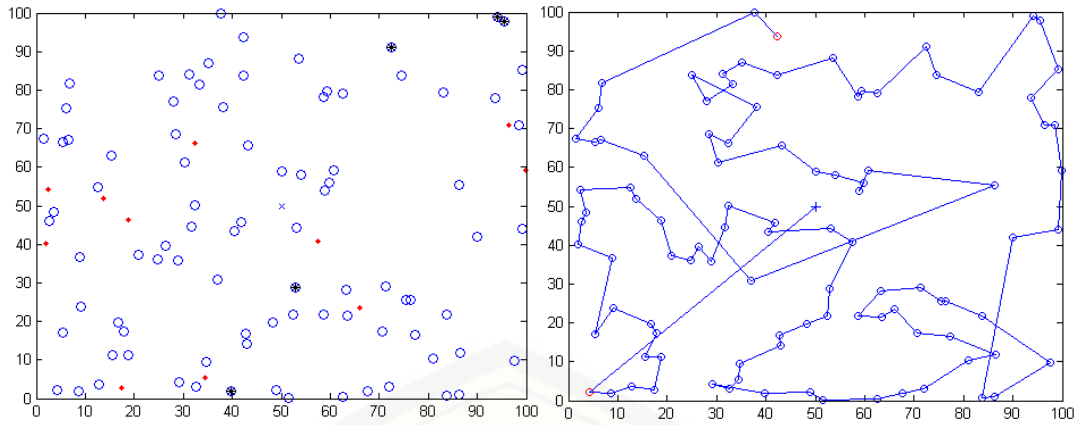
a) Variasi Jumlah *Node* (10, 30, 50, 70, dan 100)





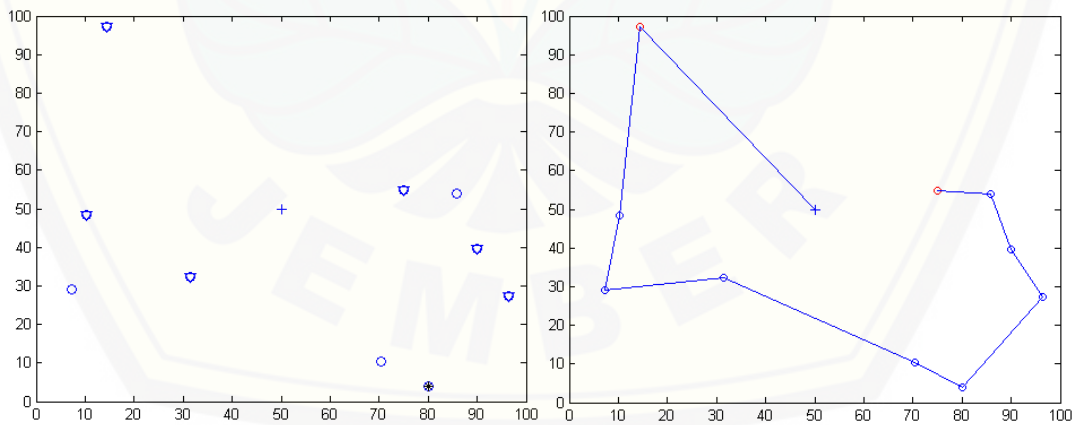
b) Variasi Energi Awal (0,1; 0,3; 0,5; 0,7; dan 0,9)

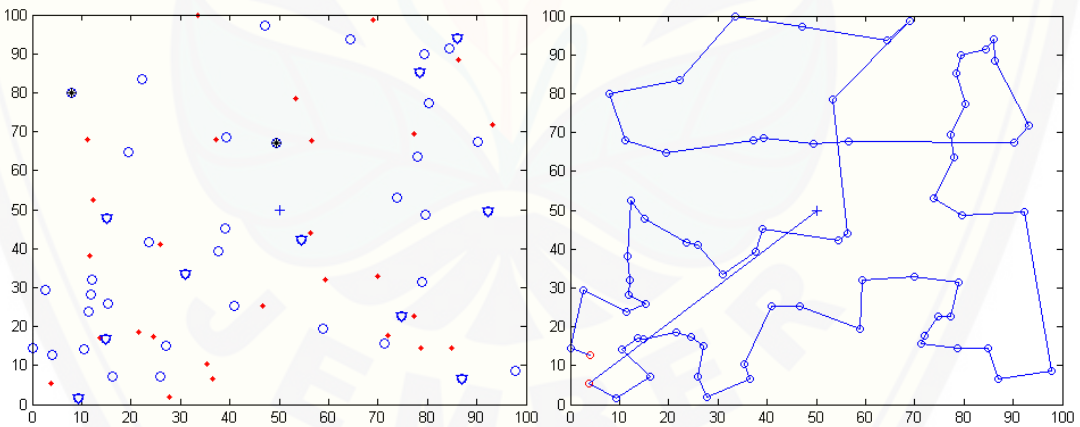
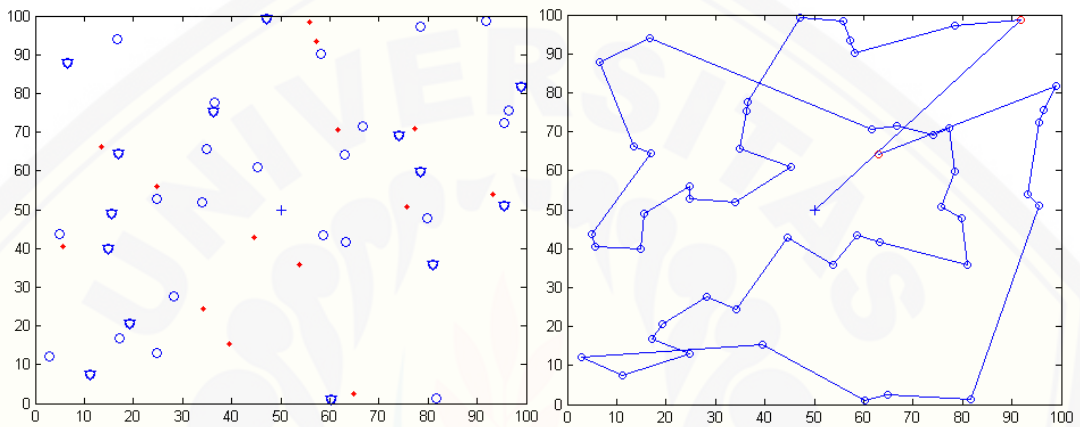
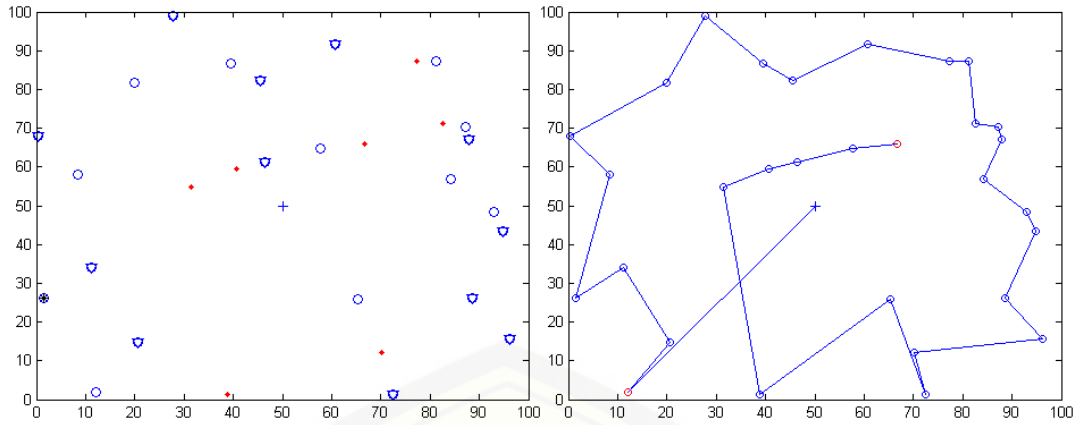


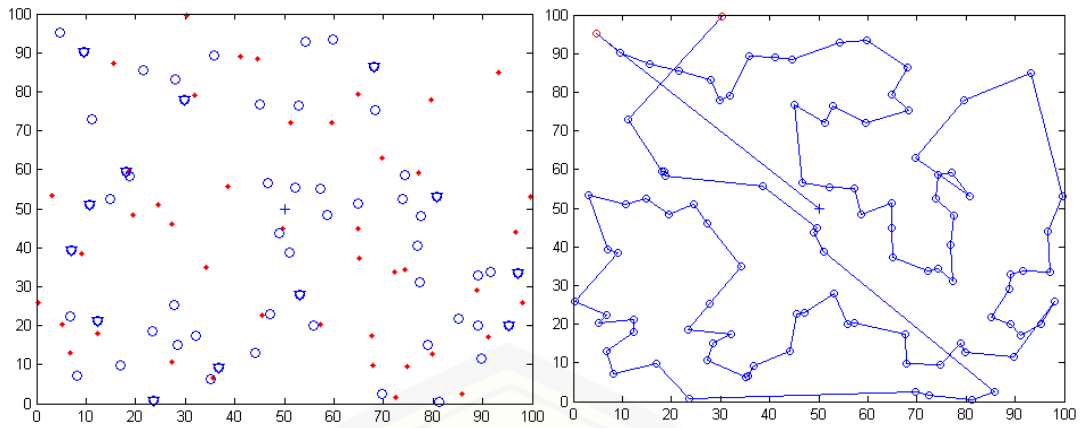


3. HEED dan PEGASIS

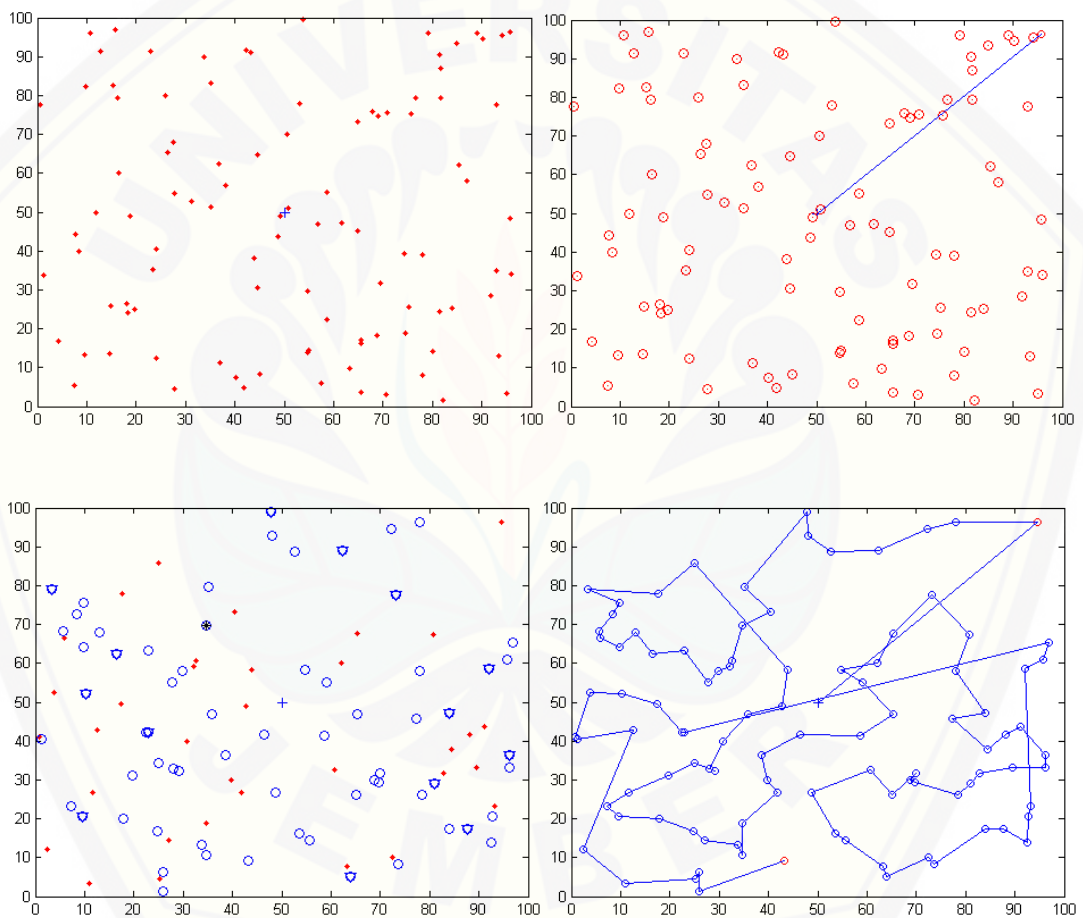
a) Variasi Jumlah *Node* (10, 30, 50, 70, dan 100)

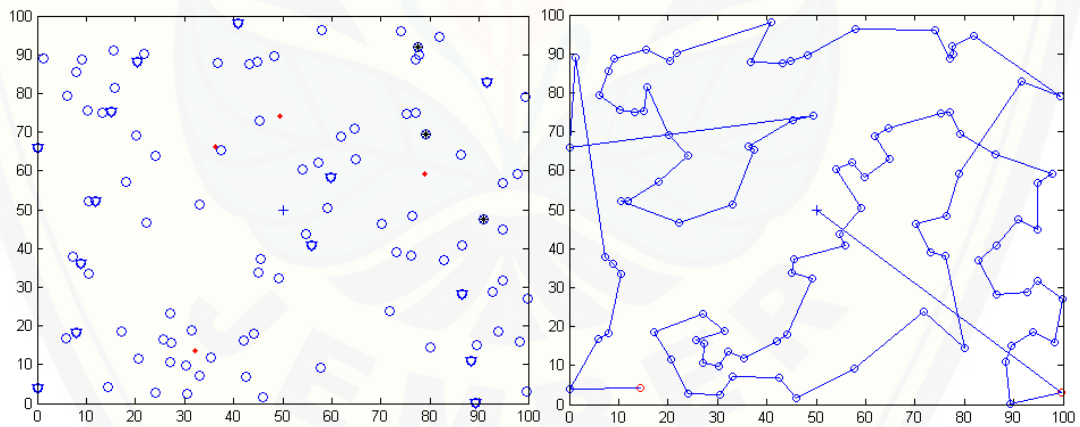
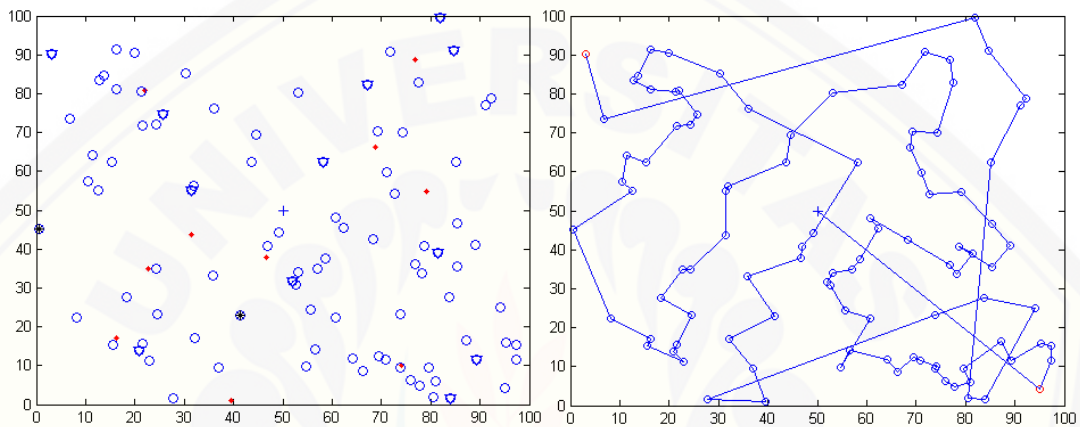
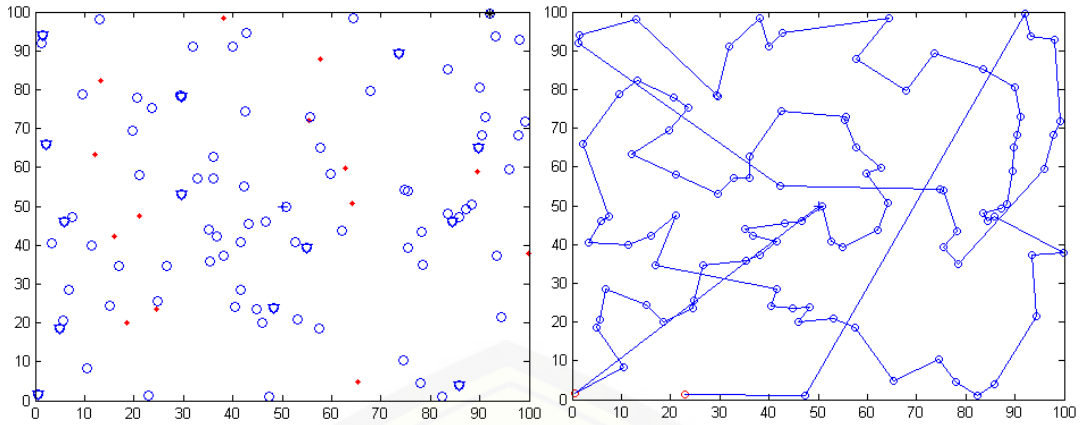






b) Variasi Energi Awal (0,1; 0,3; 0,5; 0,7; dan 0,9)





B. Listing Program

1. LEACH

```

clear;
clc;
clf;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%
xm=100;
ym=100;
sink.x=0.50*xm;
sink.y=0.50*ym;
n=100;
kbytes=100;
bbytes=10;
p=0.1;
Eo=0.25; %energi awal
ETX=33*0.0000001;
ERX=7*0.0000001;
Efs=1.5*0.001*0.000000000001;
Emp=2.5*0.000001*0.000000000001;
EDA=5*0.000000001;
rmax=300; %round
do=sqrt(Efs/Emp);

%random Sensor Network
figure(1);
for i=1:1:n
    S(i).xd=rand(1,1)*xm;
    XR(i)=S(i).xd;
    S(i).yd=rand(1,1)*ym;
    YR(i)=S(i).yd;
    S(i).G=0;
    S(i).type='N';
    S(i).E=Eo;

    plot(S(i).xd,S(i).yd,'o');
    hold on;
end

%penentuan letak BS
S(n+1).xd=sink.x;
S(n+1).yd=sink.y;
plot(S(n+1).xd,S(n+1).yd,'x');

figure(1);
countCHs=0;
rcountCHs=0;
cluster=0;
countCHs;

Dtotal=zeros(1,rmax);
data=zeros(1,rmax);
loss=zeros(1,rmax);
Etotal=zeros(1,rmax);

```



```

Energy=zeros (rmax+1,n);
Energy (1, :)=Eo;

for r=1:1:rmax
    r

    if(mod(r, round(1/p) )==0)
        for i=1:1:n
            S(i).G=0;

        end
    end

    hold off;

    dead=0;

    figure(1);

    for i=1:1:n
        %pengecekan node yang mati
        if (S(i).E<=0)
            plot(S(i).xd,S(i).yd, 'red .');
            dead=dead+1;
            S(i).type='D';
            ronde(r).S(i).type='D';
            hold on;

        end
        if S(i).E>0
            S(i).type='N';
            ronde(r).S(i).type='N'
            plot(S(i).xd,S(i).yd, 'o');
            hold on;

        end
    end
    plot(S(n+1).xd,S(n+1).yd, 'x');

    TOTAL(r).DEAD=dead;

    countCHs=0;
    cluster=0;
    for i=1:1:n
        if(S(i).E>0)
            temp_rand=rand;
            if ( (S(i).G)<=0)

                %pemilihan CH
                if (temp_rand<= (p/(1-p*mod(r, round(1/p)))))
                    countCHs=countCHs+1;

                    S(i).type= 'C';
                    ronde(r).S(i).type= 'C';
                    S(i).G=round(1/p)-1;
                    ronde(r).S(i).G=round(1/p)-1;
                    cluster=cluster+1;
                    C(cluster).xd=S(i).xd;
                    ronde(r).C(cluster).xd=S(i).xd;
                    C(cluster).yd=S(i).yd;
                end
            end
        end
    end
end

```

```

ronde(r).C(cluster).yd=S(i).yd;
plot(S(i).xd,S(i).yd,'k*');

distance=sqrt( (S(i).xd-(S(n+1).xd) )^2 +(S(i).yd-
(S(n+1).yd) )^2 );
C(cluster).distance=distance;
ronde(r).C(cluster).distance=distance;
C(cluster).id=i;
ronde(r).C(cluster).id=i;
C(cluster).Einit=S(i).E;
ronde(r).C(cluster).Einit=S(i).E;

for j=1:n
    if (j~=i)
        dist = sqrt ( (S(i).xd-(S(j).xd) )^2 + (S(i).yd-
(S(j).yd) )^2 );
        if (dist>do)
            ETBroad=ETX*bbytes + Emp*bbytes*(dist* dist *
dist * dist);
        end
        if(dist<=do)
            ETBroad=ETX*bbytes + Efs*bbytes*( dist * dist);
        end
        S(i).E = S(i).E -ETBroad;
        if (S(i).E<=0)
            ronde(r).S(i).type='D';
        end

        ERBroad =( ERX *bbytes );
        S(j).E = S(j).E -ERBroad;
        if (S(j).E<=0)
            ronde(r).S(j).type='D';
        end
    end
end
end
end
end
cluster;
TOTAL(r).CLUSTER=cluster;

%pembentukan cluster
for i=1:1:n
    if ( S(i).type=='N' && S(i).E>0 )
        if (cluster>=1)
            min_dis=9999;
            min_dis_cluster=1;
            for c=1:1:cluster
                temp=min(min_dis,sqrt( (S(i).xd-C(c).xd)^2 + (S(i).yd-
C(c).yd)^2 ) );

                if ( temp<min_dis )
                    min_dis=temp;
                    min_dis_cluster=c;
                end
            end
        end

%pengiriman data dan penghitungan energi sisa
min_dis;

```

```

        min_dis_cluster=c;
        if (min_dis>do)
            S(i).E=S(i).E- ( ETX*(kbytes) + Emp*kbytes*( min_dis *
min_dis * min_dis * min_dis));
            ronde(r).S(i).E=S(i).E- ( ETX*kbytes + Emp*kbytes*( min_dis
* min_dis * min_dis * min_dis));
        end
        if(min_dis<=do)
            S(i).E=S(i).E- ( ETX*(kbytes) + Efs*kbytes*( min_dis *
min_dis));
            ronde(r).S(i).E=S(i).E- ( ETX*kbytes + Efs*kbytes*( min_dis
* min_dis));
        end
        if(S(i).E<=0)
            S(i).type='D';
        end

        if (min_dis>0)
            S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E- (
(ERX + EDA)*kbytes );
            ronde(r).S(C(min_dis_cluster).id).E =
S(C(min_dis_cluster).id).E- ( (ERX + EDA)*kbytes );
        end
        if(S(C(min_dis_cluster).id).E<=0)

            plot(S(C(min_dis_cluster).id).xd,S(C(min_dis_cluster).
id).yd, 'red .' );
            dead=dead+1;
            S(C(min_dis_cluster).id).type= 'D';
            ronde(r).S(C(min_dis_cluster).id).type='D';
            hold on;
        end

        if (C(min_dis_cluster).distance>do)
            ch2bs=( (ETX+EDA)*kbytes + Emp*kbytes*(
C(min_dis_cluster).distance )^4);
        end
        if(C(min_dis_cluster).distance<=do)
            ch2bs=( (ETX+EDA)*kbytes + Efs*kbytes*(
C(min_dis_cluster).distance )^2);
        end

        S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E -
ch2bs;
        ronde(r).S(C(min_dis_cluster).id).E =
S(C(min_dis_cluster).id).E - ch2bs;
        S(i).min_dis=min_dis;
        ronde(r).S(i).min_dis=min_dis;
        S(i).min_dis_cluster=min_dis_cluster;
        ronde(r).S(i).min_dis_cluster=min_dis_cluster;

        if (S(C(min_dis_cluster).id).E<=0)
            S(C(min_dis_cluster).id).type='D';
            ronde(r).S(C(min_dis_cluster).id).type='D';
        end
    else
        min_dis=sqrt( (S(i).xd-S(n+1).xd)^2 + (S(i).yd-S(n+1).yd)^2 );
    end
end

```

```

    if (min_dis>do)
        n2bs=( (ETX+EDA)*kbytes + Emp*kbytes*( min_dis )^4);
    end

    if(min_dis<=do)
        n2bs=( (ETX+EDA)*kbytes + Efs*kbytes*( min_dis )^2);
    end

    S(i).E = S(i).E - n2bs;
    ronde(r).S(i).E = S(i).E - n2bs;
    S(i).min_dis=min_dis;
    ronde(r).S(i).min_dis=min_dis;
    S(i).min_dis_cluster=n+1;
    ronde(r).S(i).min_dis_cluster=n+1;

    if(S(i).E<=0)
        S(i).type='D';
        ronde(r).S(i).type='D';
    end
else
    if( S(i).type=='C' && S(i).E>0 )
        min_dis=sqrt( (S(i).xd-S(n+1).xd)^2 + (S(i).yd- S(n+1).yd)^2 );

        if (min_dis>do)
            n2bs=( (ETX)*kbytes + Emp*kbytes*( min_dis )^4);
        end

        if(min_dis<=do)
            n2bs=( (ETX)*kbytes + Efs*kbytes*( min_dis )^2);
        end

        S(i).E = S(i).E - n2bs;
        ronde(r).S(i).E = S(i).E - n2bs;
        S(i).min_dis=min_dis;
        ronde(r).S(i).min_dis=min_dis;
        S(i).min_dis_cluster=n+1;
        ronde(r).S(i).min_dis_cluster=n+1;

        if (S(i).E<=0)

            S(i).type='D';
            ronde(r).S(i).type='D';
        end
    end
end
if(S(i).E>=0)
    Energy(r+1,i)=S(i).E;
    Etotal(r)=Etotal(r)+S(i).E;
end
end

hold on;
countCHs;
TOTAL(r).CH=countCHs;
Dtotal(r)=TOTAL(r).DEAD;
rcountCHs=rcountCHs+countCHs;
end

```

2. HEED

```

clear;
clc;
clf;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
xm=100;
ym=100;
sink.x=0.5*xm;
sink.y=0.5*ym;
n = 100;
ETX=33*0.0000001;
ERX=7*0.0000001;
Efs=1.5*0.001*0.000000000001;
Emp=2.5*0.000001*0.000000000001;
EDA=5*0.000000001;
do=sqrt(Efs/Emp);
rmax = 300; %Round
InitEn=0.25; %energi awal
MaxEn=InitEn;
Tr=50; %radius cluster
kbytes=100;
bbytes=10;
Cprob=0.1;
NON_CH = 0;
TENTATIVE_CH = 1;
FINAL_CH = 2;

%random Sensor Network
figure(1);
for i=1:1:n
    Node.x(i)=rand(1,1)*xm;
    Node.y(i)=rand(1,1)*ym;
    Node.type(i)= 'N';
    plot(Node.x(i),Node.y(i), 'o');
    hold on;
end

%penentuan letak BS
Node.x(n+1)=sink.x;
Node.y(n+1)=sink.y;
plot(Node.x(n+1),Node.y(n+1), 'x');
figure(1);

Node.EnNode=zeros(1,n)+InitEn;

%inisialisasi energi tiap node
Energy=zeros(rmax+1,n);

Node.Isstop=zeros(1,n);
Dtotal=zeros(1,rmax);
data=zeros(1,rmax);
loss=zeros(1,rmax);
Etotal=zeros(1,rmax);

```

```

%penghitungan jarak antar node
rcountCHs=0;
dist=0;
for i=1:n
    count = 0 ;
    for j=1:n
        if (j~=i)
            dist = sqrt((Node.x(i)-Node.x(j)).^2)+(Node.y(i)-
Node.y(j)).^2); %distance^2
            if dist < sqrt(Tr^2+Tr^2)
                count=count+1;
                Node.Nbr(i,count)=j;
            end
        end
    end
    if (j==n)
        Node.NumNbr(i) = count ;
    end
end
end

Energy(1,:)=InitEn;

for r= 1:1:rmax
    r

    cluster=0;
    Node.csize=zeros(1,n);
    Node.c_cost=zeros(1,n);

    Node.IsClusterHeads=zeros(1,n);
    Node.CHprob=zeros(1,n)+Cprob;
    Node.CHprev=zeros(1,n);

    Node.tent_CH=zeros(1,n)+NON_CH;
    Node.tent_CH_Cost=ones(1,n)+9999;

    Node.ListtentCH=zeros(n);
    Node.ListtentCH_Cost=zeros(n)+9999;

    Node.ListfinalCH=zeros(n);
    Node.ListfinalCH_Cost=zeros(n)+9999;

    Node.my_finalCH=zeros(1,n);
    Node.my_final_CH_Cost=ones(1,n)+9999;

    Node.CHprob=max((Cprob.*((Node.EnNode)./MaxEn)),0.05);

    dead=0;
    hold off;
    figure(1);

    %pengecekan node yang mati
    for i =1:n
        if (Node.EnNode(i)<=0)
            plot(Node.x(i), Node.y(i), 'red. ');
            dead=dead+1;
            Node.type(i) = 'D';
            ronde(r).N(i).type= 'D';
            Node.Isstop(i)=1;
        end
    end
end

```

```

        hold on;
    end

    if Node.EnNode(i)>0
        Node.type(i)= 'N';
        ronde(r).N(i).type= 'N';
        plot(Node.x(i),Node.y(i), 'o');
        hold on;
    end
end
plot(sink.x,sink.y, '+');

cluster=0;

%pemilihan CH dan pembentukan cluster
for i =1:n
    if Node.Isstop(i)==0
        while Node.CHprev(i)<1;
            if sum(Node.ListfinalCH(i,:))~=0
                for j=1:n
                    if Node.ListfinalCH(i,j) ~=0
                        if Node.my_final_CH_Cost(i) >
Node.ListfinalCH_Cost(i,j)
                            Node.my_finalCH(i)= Node.ListfinalCH(i,j);
Node.my_final_CH_Cost(i)=Node.ListfinalCH_Cost(i,j);
                                end
                            end
                        end
                        Node.CHprev(i)=1;

                    elseif sum(Node.ListtentCH(i,:))~=0
                        for j=1:n
                            if Node.ListtentCH(i,j) ~=0
                                if Node.tent_CH_Cost(i) >
Node.ListtentCH_Cost(i,j)
                                    Node.tent_CH(i)= Node.ListtentCH(i,j);
Node.tent_CH_Cost(i)=
Node.ListtentCH_Cost(i,j);
                                        end
                                    end
                                end
                                Node.my_finalCH(i)= Node.tent_CH(i);
Node.my_final_CH_Cost(i)=Node.tent_CH_Cost(i);
Node.CHprev(i)=1;

                            elseif Node.CHprob(i)==1
                                Node.IsClusterHeads(i)= FINAL_CH;
Node.my_finalCH(i)=i;
Node.my_final_CH_Cost(i)= Node.NumNbr(i);
cluster=cluster+1;
Node.type(i)= 'C';
ronde(r).N(i).type= 'C';
C(cluster).xd=Node.x(i);
ronde(r).C(cluster).xd=Node.x(i);
C(cluster).yd=Node.y(i);
ronde(r).C(cluster).yd=Node.y(i);
plot(Node.x(i),Node.y(i), 'k*');

```

```

Node.y(n+1))^2));
d2bs=sqrt(((Node.x(i)- Node.x(n+1))^2)+((Node.y(i)-
Node.y(n+1))^2));
C(cluster).d2bs=d2bs;
ronde(r).C(cluster).d2bs=d2bs;
C(cluster).id=i;
ronde(r).C(cluster).id=i;

for j=1:Node.NumNbr(i)
    curentnbr = Node.Nbr(i,j);
    Node.ListfinalCH(curentnbr,i)=i;

    Node.ListfinalCH_Cost(curentnbr,i)= Node.NumNbr(i);
    distance =sqrt((Node.x(i)-
Node.x(curentnbr)).^2)+((Node.y(i)- Node.y(curentnbr)).^2);

    if (distance>do)
        EntranPCH=ETX*bbytes + Emp*bbytes*(distance*
distance * distance * distance);
    end
    if (distance<=do)
        EntranPCH=ETX*bbytes + Efs*bbytes*(distance *
distance);
    end
    Node.EnNode(i)=Node.EnNode(i)-EntranPCH;
    if (Node.EnNode(i)<=0)
        Node.Isstop(i)=1;
    end

    EnRecP =( ERX *bbytes );

    Node.EnNode(curentnbr)=Node.EnNode(curentnbr)-
EnRecP;

    if (Node.EnNode(i)<=0)
        Node.Isstop(i)=1;
    end
end
Node.CHprev(i)=1;
elseif rand(1,1)<=Node.CHprob(i)
    temp=rand(1,1);
    Node.IsClusterHeads(i)=TENTATIVE_CH;
    Node.tent_CH(i)=i;
    Node.tent_CH_Cost(i)=Node.NumNbr(i);
    Node.my_finalCH(i)=i;
    Node.my_final_CH_Cost(i)= Node.NumNbr(i);
    cluster=cluster+1;
    Node.type(i)= 'C';
    ronde(r).N(i).type= 'C';
    C(cluster).xd=Node.x(i);
    ronde(r).C(cluster).xd=Node.x(i);
    C(cluster).yd=Node.y(i);
    ronde(r).C(cluster).yd=Node.y(i);
    plot(Node.x(i),Node.y(i), 'bv');

d2bs=sqrt(((Node.x(i)- Node.x(n+1))^2)+((Node.y(i)-
Node.y(n+1))^2));
C(cluster).d2bs=d2bs;
ronde(r).C(cluster).d2bs=d2bs;
C(cluster).id=i;

```



```

ronde(r).C(cluster).id=i;

for j=1:Node.NumNbr(i)
    curentnbr = Node.Nbr(i,j);
    Node.ListtentCH(curentnbr,i)=i;

    Node.ListtentCH_Cost(curentnbr,i)= Node.NumNbr(i);
    distance =sqrt((Node.x(i)-
Node.x(curentnbr)).^2)+((Node.y(i)- Node.y(curentnbr)).^2);

    if (distance>do)
        EntranPCH=ETX*bbytes + Emp*bbytes*( distance*
distance * distance * distance);
    end

    if (distance<=do)
        EntranPCH=ETX*bbytes + Efs*bbytes*( distance *
distance);
    end
    Node.EnNode(i)=Node.EnNode(i)-EntranPCH;
    if (Node.EnNode(i)<=0)
        Node.Isstop(i)=1;
    end

    EnRecP =( ERX *bbytes );

    Node.EnNode(curentnbr)=Node.EnNode(curentnbr)-
EnRecP;

    if (Node.EnNode(i)<=0)

        Node.Isstop(i)=1;
    end
end
Node.CHprev(i)=1;
else
Node.CHprev(i)=Node.CHprob(i);
Node.CHprob(i)=min(Node.CHprob(i).*2,1);
end
end
end
end
TOTAL(r).CH=cluster;

%penghitungan konsumsi energi
for i =1:n
    if Node.Isstop(i)==0;
        if Node.my_finalCH(i) ~= NON_CH
            if Node.IsClusterHeads(i)== NON_CH;
                Node.c(i)=Node.my_finalCH(i);
                Node.c_cost(i)=Node.my_final_CH_Cost(i);
                distance =sqrt((Node.x(i)-
Node.x(Node.c(i))).^2)+((Node.y(i)- Node.y(Node.c(i))).^2);
                Node.d2ch(i)=distance;
                if (distance>do)
                    EntranPCH=ETX*kbytes + Emp*kbytes*( distance*
distance * distance * distance);
                end
                if (distance<=do)
                    EntranPCH=ETX*kbytes + Efs*kbytes*( distance *
distance);

```

```

end

Node.EnNode(i)=Node.EnNode(i)-EntranPCH;
if (Node.EnNode(i)<=0)
    Node.Isstop(i)=1;
end

EnRecP=( (ERX + EDA)*kbytes );
Node.EnNode(Node.c(i))=Node.EnNode(Node.c(i))- EnRecP;

if (Node.EnNode(Node.c(i))<=0)

    Node.Isstop(Node.c(i))=1;

end

Node.csize(Node.c(i))=Node.csize(Node.c(i))+1;
dist2bs=sqrt(((Node.x(Node.c(i))-
Node.x(n+1))^2)+((Node.y(Node.c(i))- Node.y(n+1))^2));
if (dist2bs>do)
    ch2bs=( (ETX+EDA)*kbytes + Emp*kbytes*( dist2bs
)^4);
end

if (dist2bs<=do)
    ch2bs=( (ETX+EDA)*kbytes + Efs*kbytes*(distance
)^2);
end

Node.EnNode(Node.c(i))=Node.EnNode(Node.c(i))- ch2bs;

if (Node.EnNode(Node.c(i))<=0)

    Node.Isstop(Node.c(i))=1;
end
else

dist2bs=sqrt(((Node.x(i)- Node.x(n+1))^2)+((Node.y(i)-
Node.y(n+1))^2));
if (dist2bs>do)
    ch2bs=( (ETX)*kbytes + Emp*kbytes*( dist2bs )^4);
end
if (dist2bs<=do)
    ch2bs=( (ETX)*kbytes + Efs*kbytes*( distance )^2);
end

Node.EnNode(i)=Node.EnNode(i)-ch2bs;
if (Node.EnNode(i)<=0)

    Node.Isstop(i)=1;
    hold on;
end
end
end

if (Node.EnNode(i)>=0)

```

```
        Energy (r+1, i)=Node.EnNode (i);  
        Etotal (r)=Etotal (r)+Node.EnNode (i);  
    end  
end  
  
hold on;  
cluster;  
Dtotal (r)=dead;  
  
rcountCHs=rcountCHs+cluster;  
  
end
```



3. PEGASIS

```

clear;
clc;
clf;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NodeNums=10;
AreaR=100;
Bx=50;
By=50;
Tr=100;
ETX=50*0.000000001;
ERX=50*0.000000001;
Efs=100*0.000000000001;
Emp=0.0013*0.000000000001;
EDA=5*0.000000001;
do=sqrt(Efs/Emp);
rmax=300; %round
En=0.25; %energi awal

num=1;
num_plot=1;
send_to_sink=0;
ctl_pkt_leng=10;
data_pkt_length=100;
die_node_num=0;
die_node_num_pri=0;
transmitted_packet=0;
En_Cost=0;
En_Cost_pre=0;
inter_cost=0;
begin_to_send=0;
En_cost_per_round=0;

alive=1;
dead=0;

Node.x=AreaR*rand(1,NodeNums); %random posisi node
Node.y=AreaR*rand(1,NodeNums);
Node.pri=linspace(0,0,NodeNums);
Node.already=linspace(0,0,NodeNums);
Node.to_nbr_dis=zeros(NodeNums);
Node.to_pri_dis=linspace(0,0,NodeNums);
Node.send_dis=linspace(0,0,NodeNums);
Node.E=linspace(En,En,NodeNums);
Node.status=linspace(alive,alive,NodeNums);
Node.E_dis=linspace(0,0,NodeNums);

%penghitungan jarak ke BS
for i=1:1:NodeNums
    To_sink_dist(i)=sqrt((Node.x(i)-Bx)^2+(Node.y(i)-By)^2);
end

[Max_Dis, Max_num]=max(To_sink_dist);
now_node=Max_num;
Node.already(now_node)=1;
Max_num

```

```

now_node
node_head=Max_num;
send_to_sink=now_node;
node_to_send=now_node;
start_node=now_node;

Node.pri=linspace(0,0,NodeNums);
Node.already=linspace(0,0,NodeNums);
Node.to_nbr_dis=zeros(NodeNums);
Node.to_pri_dis=linspace(0,0,NodeNums);
Node.send_dis=linspace(0,0,NodeNums);
Node.E=linspace(En,En,NodeNums);
Node.status=linspace(alive,alive,NodeNums);
Node.E_dis=linspace(0,0,NodeNums);
send_to_sink=0;
die_node_num=0;
die_node_num_pri=0;
begin_to_send=0;

%pembentukan rantai
for run_round=1:1:300
    run_round

    num_plot=1;
        die_node_num_pri=die_node_num;
    En_Cost_pre=En_Cost;
    run_round
    die_node_num
    En_Cost

    for i=1:1:NodeNums
        if Node.status(i)==alive
            Node.E_dis(i)=Node.E(i)/(ETX+Efs*(To_sink_dist(i)^2));
            Node.E_dis(i)=Node.E(i);
        else
            Node.E_dis(i)=0;
        end

    Node.pri=linspace(0,0,NodeNums);
    num_plot=1;
    Node.already=linspace(0,0,NodeNums);
    run_round
    for i=1:1:NodeNums
        if Node.status(i)==alive
            to_sink_dist(i)=sqrt((Node.x(i)-Bx)^2+(Node.y(i)-By)^2);
        else
            to_sink_dist(i)=0;
        end
    end

    %pemilihan chain leader
    [Max_Dis, Max_num]=max(to_sink_dist);
    now_node=Max_num;
    Node.already(now_node)=1;
    node_head=Max_num;
    send_to_sink=now_node;
    node_to_send=now_node;
    start_node=now_node;
    now_node

```

```

%pengecekan node mati
while num_plot~=NodeNums-die_node_num
    if num_plot==NodeNums-die_node_num
        Node.pri(now_node)=0;
        node_tail=now_node;
    end

    for j=1:1:NodeNums
        if Node.already(j)==0&&Node.status(j)==alive
            Node.to_nbr_dis(now_node,j)=sqrt((Node.x(now_node)-
            Node.x(j))^2+(Node.y(now_node)-Node.y(j))^2);
            else
                Node.to_nbr_dis(now_node,j)=0;
            end
        end

        j=1;
        while Node.to_nbr_dis(now_node,j)==0
            j=j+1;
        end

        min_dis=Node.to_nbr_dis(now_node,j);
        min_num=j;
        for j=1:1:NodeNums
            if
                Node.to_nbr_dis(now_node,j)~=0&&Node.to_nbr_dis(now_node,j)<min_dis
                    min_dis=Node.to_nbr_dis(now_node,j);
                    min_num=j;
                end
            end
            Node.to_pri_dis(now_node)=min_dis;
            Node.pri(now_node)=min_num;

            Node.E(now_node)=Node.E(now_node)-
            (ETX*ctl_pkt_leng+Efs*2*(min_dis* min_dis)*ctl_pkt_leng);
            Node.E(Node.pri(now_node))=Node.E(Node.pri(now_node))-
            ETX*ctl_pkt_leng;

            if now_node==Max_num
                Node.to_pri_dis(now_node);
            end
            now_node=Node.pri(now_node);

            min_num
            Node.already(now_node)=1;
            Node.to_pri_dis(now_node)=min_dis;
            min_dis
            num_plot=num_plot+1;
        end

        node_tail=now_node;

        I_want=1;
        if I_want==1
            figure(1);
        end
    end
end

```

```

if run_round~=1
    clf;
end
plot(Bx,By,'+');
hold on;

for i=1:1:NodeNums
    if Node.status(i)==alive
        plot(Node.x(i), Node.y(i), 'o','markersize',5);
        hold on;
    end
end

plot(Node.x(node_tail), Node.y(node_tail), 'o-r','markersize',5);
plot(Node.x(node_head), Node.y(node_head), 'o-r','markersize',5);

for i=1:1:NodeNums
    if Node.status(i)==alive
        if Node.pri(i)~=0
            plot([Node.x(i);Node.x(Node.pri(i))],[Node.y(i);Node.y(Node.pri(i))]);
            hold on;
        end
    else
        plot(Node.x(i), Node.y(i), 'o-r');
    end
end
end
xlabel('X-coordinate');
ylabel('Y-coordinate');

plot([Node.x(send_to_sink);Bx],[Node.y(send_to_sink);By]);
hold on;
node_to_send=Max_num;
start_node=Max_num;

end

send_to_sink=node_head;
next_next=mod(run_round,NodeNums-die_node_num);

%pengumpulan dan pengiriman data
while next_next~=0
    send_to_sink=Node.pri(send_to_sink);
    next_next=next_next-1;
end

[max_node_En,send_to_sink]=max(Node.E_dis);
send_to_sink=mod(begin_to_send,NodeNums)+1;
run_round=run_round+1;
send=send_to_sink;
while Node.status(send_to_sink)~=alive
    send_to_sink=mod(send_to_sink,NodeNums)+1;
end
begin_to_send=send_to_sink+1;
transmitted_packet=transmitted_packet+NodeNums-die_node_num;

die_node_num_pri=die_node_num;
for i=1:1:NodeNums
    if Node.E(i)<0.01*En

```

```

        die_node_num_pri=die_node_num_pri+1;
    end
end

num=1;
right_send_num=1;
node_to_send=node_head;
start_node=node_head;

    while node_to_send~=send_to_sink
        right_send_num=right_send_num+1;
        node_to_send=Node.pri(node_to_send);
    end
    right_send_num
    Node.send_dis= linspace(0,0,NodeNums);
    while num~=NodeNums+1-die_node_num
        if num<right_send_num
Node.send_dis(start_node)=Node.to_pri_dis(start_node);
        start_node=Node.pri(start_node);
        end
        if num==right_send_num
Node.send_dis(start_node)=to_sink_dist(start_node);
        start_node_1=start_node;
        start_node=Node.pri(start_node);
        end
        if num>right_send_num
Node.send_dis(start_node)=Node.to_pri_dis(start_node_1);
        start_node_1=start_node;
        start_node=Node.pri(start_node);
        end
        num=num+1;
    end

    inter_cost=0;
    for i=1:1:NodeNums
        if i~=send_to_sink && Node.status(i)==alive
            inter_cost=inter_cost+ETX+ERX+Efs*Node.send_dis(i).^2;
        end
    end
    inter_cost/(NodeNums-die_node_num)

    if send_to_sink==node_head
        for i=1:1:NodeNums
            if Node.status(i)==alive
                if i==send_to_sink
                    if Node.send_dis(i)>do
                        Node.E(i)=Node.E(i)-
                        (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                        Node.E(i)=Node.E(i)-
                        (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                    else
                        Node.E(i)=Node.E(i)-
                        (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                    end
                end
            end
        end
    end
end

```



```

        end
        if i==node_tail
            if Node.send_dis(i)>do
                Node.E(i)=Node.E(i)-
                (ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                Node.E(i)=Node.E(i)-
                (ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
            else
                Node.E(i)=Node.E(i)-
                (ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
            end
        end
        if i~=send_to_sink&&i~=node_head&&i~=node_tail
            if Node.send_dis(i)>do
                Node.E(i)=Node.E(i)-
                (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                Node.E(i)=Node.E(i)-
                (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
            else
                Node.E(i)=Node.E(i)-
                (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
            end
        end
    end
end

if send_to_sink==node_tail
    for i=1:1:NodeNums
        if Node.status(i)==alive
            if i==send_to_sink
                if Node.send_dis(i)>do
                    Node.E(i)=Node.E(i)-
                    (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                    Node.E(i)=Node.E(i)-
                    (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                else
                    Node.E(i)=Node.E(i)-
                    (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                end
            end
            if i==node_head
                if Node.send_dis(i)>do
                    Node.E(i)=Node.E(i)-
                    (ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                    Node.E(i)=Node.E(i)-
                    (ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                else
                    Node.E(i)=Node.E(i)-
                    (ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                end
            end
            if i~=send_to_sink&&i~=node_head&&i~=node_tail
                if Node.send_dis(i)>do

```

```

        Node.E(i)=Node.E(i)-
        (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
        Node.E(i)=Node.E(i)-
        (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
        else
        Node.E(i)=Node.E(i)-
        (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
        end
    end
end
end
end

if send_to_sink~=node_tail&&send_to_sink~=node_head
    for i=1:1:NodeNums
        if Node.status(i)==alive
            if i==send_to_sink
                if Node.send_dis(i)>do
                    Node.E(i)=Node.E(i)-
                    (2*ERX*data_pkt_length+EDA*3*data_pkt_length+ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                    Node.E(i)=Node.E(i)-
                    (2*ERX*data_pkt_length+EDA*3*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                else
                    Node.E(i)=Node.E(i)-
                    (2*ERX*data_pkt_length+EDA*3*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                end
            end
            if i==node_head||i==node_tail
                if Node.send_dis(i)>do
                    Node.E(i)=Node.E(i)-
                    (ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                    Node.E(i)=Node.E(i)-
                    (ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                else
                    Node.E(i)=Node.E(i)-
                    (ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                end
            end
            if i~=send_to_sink&&i~=node_head&&i~=node_tail
                if Node.send_dis(i)>do
                    Node.E(i)=Node.E(i)-
                    (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Emp*(Node.send_dis(i).^4)*data_pkt_length);
                    Node.E(i)=Node.E(i)-
                    (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                else
                    Node.E(i)=Node.E(i)-
                    (ERX*data_pkt_length+EDA*2*data_pkt_length+ETX*data_pkt_length+Efs*(Node.send_dis(i).^2)*data_pkt_length);
                end
            end
        end
    end
end
end
end
end

```

```
for k=1:1:NodeNums
    if Node.pri(k)~=0

En_cost_per_round=En_cost_per_round+(ERX+ETX)*data_pkt_length+Efs*data_pkt_
length*(Node.send_dis(k).^2);
        else

En_cost_per_round=En_cost_per_round+(ERX+ETX)*data_pkt_length+Efs*data_pkt_
length*(Node.send_dis(k).^2);
        end
    end
    En_cost_per_round=0.25*100-sum(Node.E);
    En_Cost=En*NodeNums-sum(Node.E);
    round_cost=En_Cost-En_Cost_pre;
    round_cost
    send_to_sink
    Node.E
    if Node.pri(send_to_sink)==0
        send_to_sink=node_head;
    else
        send_to_sink=Node.pri(send_to_sink);
    end
end
die_node_num=0;
for i=1:1:NodeNums
    if Node.E(i)<0.001*En
        die_node_num=die_node_num+1;
        Node.status(i)=dead;
    end
end
end
end
```