PROYEK AKHIR

ANALISA KINERJA ROUTING PROTOKOL PADA JARINGAN SENSOR NIRKABEL DENGAN METODE GRADIENT BASED APPROACH

KUNPRAGA MAULANA ARROSSY
NRP 7207 040 036

Dosen Pembimbing
Tri Budi Santoso, ST. MT.
NIP. 197001051995021001

Ir. Prima Kristalina, MT.
NIP. 196505251990032001

JURUSAN TEKNIK TELEKOMUNIKASI
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2011

PROYEK AKHIR

ANALISA KINERJA ROUTING PROTOKOL PADA JARINGAN SENSOR NIRKAEBL DENGAN METODE GRADIENT BASED APPROACH

Kunpraga Maulana A
NRP. 7207 040 036

Dosen Pembimbing :
Tri Budi Santoso, ST. MT.
NIP. 197001051995021001

Ir. Prima Kristalina, MT.
NIP. 196505251990032001

JURUSAN TEKNIK TELEKOMUNIKASI
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2011

ANALISA KINERJA ROUTING PROTOKOL PADA JARINGAN SENSOR NIRKAEBL DENGAN METODE GRADIENT BASED APPROACH

Oleh:

KUNPRAGA MAULANA ARROSSY
7207 040 036

Proyek Akhir ini Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Sains Terapan (S.ST)
di
Politeknik Elektronika Negeri Surabaya
Institut Teknologi Sepuluh Nopember Surabaya

Disetujui oleh

Tim Penguji Proyek Akhir :

1. Ir. Nur Adi Siswandari, MT
   NIP. 196004301994032001
2. Okkie Puspitorini, ST. MT
   NIP. 197010111995122001
3. Hani'ah Mahmudah, ST. MT
   NIP. 197709162001122001

Dosen Pembimbing:

1. Tri Budi Santoso, ST. MT
   NIP. 197001051995021001
2. Ir. Prima Kristalina, MT
   NIP. 196505251990032001

Mengetahui
Ketua Jurusan Telekomunikasi

Arifin, ST, MT
NIP. 196005031988031004

ABSTRAK

Jaringan Sensor Nirkabel terdiri atas sejumlah besar sensor node yang bebas. Setiap node memiliki kemampuan untuk mengirim, menerima dan mendeteksi. Sensor node juga di lengkapi dengan peralatan pemrosesan data, penyimpanan data sementara atau memory, peralatan komunikasi dan power supply atau baterai. Dalam aplikasi WSN, sensor node harus mempunyai dimensi yang sangat kecil, sehingga sensor node mempunyai keterbatasan baik dalam processor, memory atau wireless.

Pada proyek akhir ini telah dibangun sebuah simulasi pengiriman data (routing) menggunakan Gradient Based Approach sebagai algoritma rute pengiriman data. Pendekatan secara gradient tidak menggunakan jarak yang absolut sebagai parameternya, tetapi dengan menggunakan jarak relatif. Beberapa parameter yang digunakan adalah perhitungan Hop, konsumsi energi, dan sisa energi pada sensor.

Hasil yang diproses pada proyek akhir ini adalah membuat sebuah jalur routing yang cepat dan efisien serta perencanaan konfigurasi sistem agar transmisi informasi dapat berlangsung dengan baik dan optimal dengan menggunakan metode Gradient-based routing. Pada proyek akhir ini diketahui pengiriman 100bit data dalam satu rute (dari sumber ke sink) yang membututuhkan energy rata-rata sebesar 287420.5 nJ.

Kata kunci : Routing protocol, Gradient routing protocol, Wireless Sensor Network

ABSTRACT

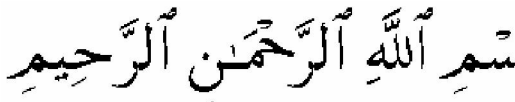
Wireless Sensor Network consist a large number of free sensor node. Each node has the ability to send, receive and sense. In addition, sensor nodes are also equipped with data processing equipment, data storage or memory, communication equipment, and power supply or battery. In WSN applications, sensor nodes must have a very small dimension, so that the sensor nodes have limitation in the processor, memory and wireless. Therefore many researchers are interested in WSN to optimize their energy needs in data processing or communication to the sink.

In this final project defined as the process of selecting the best route with the lowest energy celection until it reaches the destination point (gradient based). On the data source and other nodes, will determine the amount of energy required to sink, from the data source, data will be on flow by decreasing energy up to the sink. The problem will be resolved is how to design configuration, simulation implementation of gradient based routing in WSN and how these methods influence the performance of the system.

Expected results at the end of this project is to create the fastes path and efficient routing and planning system configuration for information transmission can take place properly and optimally using Gradient-based routing. At this project tells average energy that needed to transmit 100bits data to the sink in a single routing is 287420.5 nJ.

Keywords : Routing protocol, Gradient routing protocol, Wireless Sensor Network

KATA PENGANTAR



Dengan mengucap puji syukur kepada Allah, atas limpahan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan proyek akhir yang berjudul :

ANALISA KINERJA ROUTING PROTOKOL PADA JARINGAN SENSOR NIRKABEL DENGAN METODE GRADIENT BASED APPROACH

Proyek Akhir ini dibuat dengan maksud dan tujuan untuk memenuhi salah satu persyaratan menyelesaikan studi di Jurusan Telekomunikasi Politeknik Elektronika Negeri Surabaya, Institut Teknologi Sepuluh Nopember yang telah dijalani selama genap empat tahun. Disamping itu juga sebagai pembelajaran untuk diri pribadi dalam banyak hal, baik itu pengetahuan, mempraktekkan kuliah yang telah didapat selama ini dan sebuah persiapan untuk menghadapi dunia kerja.

Dengan selesainya buku laporan proyek akhir ini, penulis berharap semoga buku ini dapat membawa manfaat bagi pembaca umumnya dan juga bagi penulis pada khususnya serta semua pihak yang berkepentingan. Penulis juga berharap agar proyek akhir ini dapat dikembangkan lebih lanjut sehingga dapat benar-benar digunakan sebaik-baiknya untuk mendukung perkembangan ilmu pengetahuan.

Kami menyadari bahwa kami adalah manusia biasa yang tidak luput dari kesalahan dan kekurangan. Untuk itu, kritikan dan saran yang bersifat membangun kami harapkan untuk perbaikan selanjutnya.

Surabaya, Juli 2009

Penulis

UCAPAN TERIMA KASIH

Dengan segala kerendahan hati, pada kesempatan ini saya ingin menyampaikan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu saya dalam menyelesaikan proyek akhir ini, sehingga saya dapat menyelesaikan studi saya di Politeknik Elektronika Negeri Surabaya. Ucapan terima kasih saya tujukan kepada :

1. Keluarga tercinta yang telah membesarkan dan membimbing saya dengan penuh kasih sayang, pengorbanan, serta do'a yang tak henti-hentinya dipanjatkan demi kesuksesan saya.
2. Bapak Dr. Dadet Pramadihanto, M.Eng selaku Direktur Politeknik Elektronika Negeri Surabaya, Institut Teknologi Sepuluh Nopember.
3. Bapak Arifin, ST, MT selaku Kepala Jurusan Telekomunikasi
4. Bapak Tri Budi Santoso, ST, MT selaku dosen pembimbing yang selalu memberikan bimbingan, ilmu dan waktunya.
5. Ibu Ir. Prima Kristalina, MT selaku dosen pembimbing yang selalu memberikan bimbingan, ilmu dan waktunya.
6. Ibu Ir. Nur Adi Siswandari, MT dosen penguji untuk ilmu dan waktunya.
7. Ibu Okkie Puspitorini, ST. MT dosen penguji untuk ilmu dan waktunya.
8. Ibu Hani'ah Mahmudah, ST. MT dosen penguji atas ilmu dan waktunya.
9. Sahabat senasib seperjuangan Lab DSP atas dukungan, semangat, canda, dan tawanya.
10. Keluarga D4 TB untuk dukungan, semangat, persahabatan, bantuan, dan masih banyak lagi. Semoga kita bisa tetap dekat seperti keluarga walaupun telah berpisah nanti.
11. Semua teman – teman yang tidak bisa saya sebutkan satu presatu untuk do'a dan dukukannya.

Penulis menyadari "tak ada gading yang tak retak". Demikian juga dalam penyusunan buku Proyek Akhir ini yang jauh dari sempurna. Saran dan kritik yang membangun diharapkan penulis. Semoga buku ini dapat memberikan manfaat bagi kita semua.

Semoga Allah SWT senantiasa memberikan balasan yang lebih baik di kemudian hari.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Jaringan Sensor Nirkabel (Wireless Sensor Network) terdiri atas sejumlah besar sensor node yang bebas. Setiap node memiliki kemampuan untuk mengirim, menerima dan mendeteksi. Sensor node memiliki kemampuan terbatas didalam kapasitas memori, bandwidth, dan power[2].

Suatu sistem terkadang memerlukan data-data tertentu sebagai input atau acuan dalam merespon, atau dalam menentukan langkah yang akan dilakukan selanjutnya untuk mencapai tujuan dari sistem tersebut. Namun terkadang data-data tersebut merupakan data fisis yang yang harus didapatkan secara berkelanjutan hingga periode waktu yang sangat lama. Selain itu tidak jarang data tersebut hannya berada pada daerah-daerah yang bersifat ekstrim, dan tidak mudah untuk didapatkan.

Sensor node yang cenderung memiliki urkuran kecil, dan memiliki kemampuan untuk mengindra. Sensor sangat cocok untuk memonitor lingkungan. Sensor menjawab semua permasalahan diatas. Sensor sering memiliki error yang cukup besar, harga yang sangat mahal, dan kebanyakan memiliki DSN (Distributed Sensor Network) yang sering tidak terstruktur. Menyebabkan pembuatan routing pada DSN merupakan hal yang vital[3].

Routing protokol tradisional di DSN kebanyakan didasari pada flooding atau random-walk, menyebabkan biaya (cost) komunikasi yang besar dikarekanakn bnyaknya rute yang di temukan dan pengiriman yang tidak terarah.

Untuk menyeselsaikan masalah itu muncul berbagai macam algoritma, yang menggunakan bergai jenis parameter untuk mendapatkan tujuan yang berbeda-beda pula.

1.2 TUJUAN

Tujuan dari proyek akhir ini adalah membuat sebuah simulasi routing protokol pada WSN(Wireless Sensor Network) yang menggunakan metode gradient based approach pada, akan menunjukkan jalur mana yang tercepat dan terefisien pada sebuah node dalam mengirimkan informasi ke node yang lain untuk mencapai suatu node destination tertentu.

1.3 PERUMUSAN MASALAH

Metode yang digunakan pada tugas akhir ini didefinisikan sebagai proses pemilihan route terbaik dengan pemilihan energi terendah hingga mencapai destination point. Pada sumber data dan node-node yang lain, akan menentukan besar energi yang dibutuhkan untuk menuju sink, dari sumber data, data akan di alirkan berdasarkan penurunan energi hingga sampai pada sink.

1.4 BATASAN MASALAH

Berdasarkan perumusan masalah di atas, diberikan batasan – batasan sebagai berikut :

- Pemilihan route terbaik untuk pengiriman data
- Routing protokol yang digunakan pada pross simulasi adalah Indicator Based Approach Gradient Based routing
- Pembuatan simulasi gradient based routing protokol dengan menggunakan Java.

1.5 METODOLOGI

Metodologi pembahasan pada proyek akhir ini direncanakan seperti yang tercantum berikut ini :

- Rancangan sistem :

  Sistem ini dirancang dengan menggunakan java yang mana nantinya route yang dipilih akan di simulasikan kedalam program yang sudah di disain pada Java. Rute yang akan di pilih di tentukan berdasarkan energi terendah yang di butuhkan dalam melewatkan data.

- Pembuatan :
  Dari hasil perancangan, akan dilakukan pembuatan simulasi Routing Protokol dengan menggunakan Gradient Based pada Wireless Sensor Network yang akan menentukan routing mana yang paling efisien untuk proses pengiriman paket data dengan menggunakan bahasa pemrograman JAVA.

- Analisa hasil penelitian :
  Sesuai dengan rancangan, simulasi menggunakan metode/algoritma Gradient based routing yang mana penentuan arah routingnya didasarkan pada biaya yang terkecil. Pada simulasi dibuat node-node yang sudah ditentukan biaya. Kemudian arah routingnya ditentukan berdasarkan penurunan biaya hingga sampai ke node destination. Setiap node yang akan mengirimkan datanya ke node berikutnya, akan memastikan terlebih dahulu bahwa node tersebut merupakan jalur yang optimal.

- Kesimpulan :
  Pada simulasi yang telah dibuat dapat ditentukan bagaimana arah routing yang sebenarnya jika didasarkan pada gradient based routing. Pada simulasi ini juga dapat dilihat jalur mana yang tercepat dan efisien pada sebuah node dalam mengirimkan informasi ke node yang lainnya untuk mencapai ke sink tertentu.

1.6 SISTEMATIKA PENULISAN

Sistematika pembahasan yang akan diuraikan dalam buku laporan proyek akhir ini terbagi dalam bab-bab yang akan dibahas sebagai berikut:

BAB I PENDAHULUAN
Bab ini membahas tentang latar belakang, tujuan, batasan masalah, permasalahan, dan sistematika pembahasan yang digunakan dalam pembuatan proyek akhir ini.

BAB II TEORI PENUNJANG

Sistematika pembahasan yang akan diuraikan dalam buku laporan proyek akhir ini terbagi dalam bab-bab yang akan dibahas sebagai berikut:

- Wireless Sensor Network (WSN)
- Routing Protokol
- Algoritma Gradient Based Routing
- Java

BAB III PERENCANAAN DAN PEMBUATAN

Berisi tentang perencanaan dan pembuatan sistem. Terdiri dari proses perancangan software dan pembuatan simulasi routing protokol.

BAB IV PENGUJIAN DAN ANALISA

Berisi tentang hasil pengujian sistem yang telah dibangun baik secara bertahap, serta analisis terhadap hasil pengujian sistem.

BAB V. KESIMPULAN

Bab ini berisi kesimpulan dari pembahasan pada perancangan awal serta analisa yang diperoleh. Untuk lebih meningkatkan mutu dari sistem yang telah dibuat maka kami memberikan saran-saran untuk perbaikan dan penyempurnaan sistem.

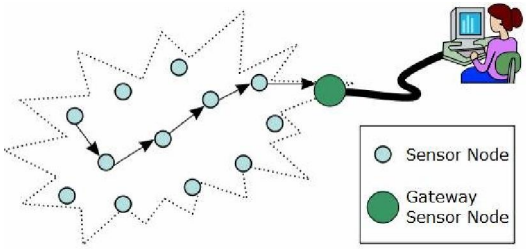
BAB VI. DAFTAR PUSTAKA

Pada bagian ini berisi tentang referensi-referensi yang telah dipakai oleh penulis sebagai acuan dan penunjang serta parameter yang mendukung penyelesaian proyek akhir ini baik secara praktik maupun sebagai teoritis.

BAB II
DASAR TEORI

2.1 WSN (Wireless Sensor Network)

Wireless sensor network merupakan sekumpulan sensor otomatis yang letaknya terdistribusi di berbagai tempat, dimana setiap titik sensor di dalam jaringan sensor dilengkapi dengan radio transceiver atau semacam alat komunikasi wireless. Sensor tersebut bekerja bersama-sama dan biasanya digunakan untuk memonitor kondisi lingkungan fisik, antara lain: suhu, gerakan, suara, getaran, perubahan warna, dan lain-lain, ilustrasi jaringan WSN dapat di lihat pada gambar 2.1. Setiap titik/node sensor biasanya dilengkapi juga dengan mikrokontroler dan sumber energi (biasanya battery atau mungkin solar cell). Sebuah Wireless sensor network biasanya merupakan jaringan wireless ad-hoc, yang berarti bahwa setiap sensor mendukung algoritma routing multi-hop dimana node-node juga berfungsi sebagai forwarder yang me-relay paket data ke stasiun pusat[4]. Penggunaan arsitektur ad-hoc dalam wireless sensor network dikarenakan arsitektur ini yang paling tepat dan paling murah untuk diterapkan dalam wireless, mengurangi biaya key factor pada banyak jaringan, seperti instalasi, maintenance dan ongoing operational needs. Karakteristiknya antara lain : self transformation function, self repair feature, dan multi hop function.



Gambar 2.1 Arsitektur Sederhana Wireless Sensor Network

Dari segi ukuran, node di dalam sensor network memiliki ukuran fisik bervariasi. Harganya juga bervariasi bergantung pada

ukuran sensor network serta kompleksitas dari sensor. Wireless sensor network memiliki karakteristik yang unik, antara lain:

- Daya / power yang dapat disimpan atau dipanen sangat terbatas, oleh karena itu sangat penting untuk menggunakan device yang hemat energi.

- Kemampuan menahan kondisi lingkungan yang keras. Device yang digunakan kemungkinan diletakkan di daerah yang mungkin saja bersuhu ekstrim atau di daerah daerah berbahaya, sehingga kemampuan ini sangat penting menjaga sensor tetap bisa digunakan meskipun kondisi lingkungan sangat ekstrim.

- Mobilitas dari node dan topologi jaringan yang dinamis. Device yang digunakan bisa saja lokasinya berpindah – pindah, misalkan sensor yang diletakkan pada armada truck pengiriman barang yang mana digunakan untuk men-tracking posisi dari armada pengiriman tersebut.

- Adanya kemungkinan kegagalan komunikasi ataupun kesalahan operasi

- Heterogenitas dari node, baik dari segi hardware (ukuran sensor, device yang digunakan, dan lain-lain) maupun software. Selain itu kemampuan yang dimiliki oleh device pun juga bisa beraneka ragam.

- Jumlah node dalam wireless sensor network bisa diperbanyak, yang membatasi jumlahnya adalah bandwidth dari gateway.

Wireless sensor network bisa diterapkan diberbagai bidang, umumnya digunakan untuk melakukan aktivitas monitoring dan tracking. Dalam bidang antisipasi dan pencegahan bencana, sensor dapat digunakan untuk mendeteksi kemungkinan bencana. Sensor diletakkan di berbagai daerah, ketika kemungkinan adanya bencana terdeteksi maka sensor akan mengirimkan data ke stasiun pusat. Selanjutnya di stasiun pusat terjadi pengolahan data, memberikan early warning system akan adanya bencana kepada para penduduk. Pemberitahuan dapat melalui berbagai media, melalui internet, ataupun sms. Selain itu, informasi dari sensor sensor dalam wireless sensor

network digabungkan dengan Geographic Information System dimungkinkan untuk mengetahui dimana titik aman yang terlindung dari bencana. Para penduduk selanjutnya bisa mengambil informasi tersebut dan mengeceknya pada GPS untuk melihat peta lokasi dari daerah yang aman bencana. Pada bidang pertanian, wireless sensor network dapat diterapkan untuk memonitor tanaman atau areal pertanian, misalkan saja tanaman teh. Tanaman teh dimana daun siap dipetik dengan daun belum siap dipetik memiliki warna yang berbeda. Sensor dapat memonitor citra daun teh, mengirimkannya ke stasiun pusat melalui wireless, kemudian dengan teknik image processing dapat diketahui daerah mana dari areal perkebunan yang sudah siap untuk dipetik.

Dari desain yang ada, yang jelas, sensor dalam Wireless sensor network bisa digunakan untuk mendapatkan data secara real time. Dan dengan adanya wireless, memberikan keuntungan sehingga alat bisa diletakkan dimana saja dan dapat dipantau meskipun dari jarak jauh. Contoh *divice* yang di gunakan dalam WSN dapat dilihat pada gambar 2.2.

Gambar 2.2 Contoh device yang digunakan sebagai sensor dalam Wireless Sensor Network

Tiap-tiap sensor memiliki transmission range yang berbeda-beda tergantung jenisnya, terlihat pada tabel 2.1.

Tabel 2.1 Tabel transmission range sensor [6]

Jenis Sensor	Radius coverage (m)	Harga ($)
1	1	50
2	5	150
3	8	160
4	10	250
5	15	300
6	20	600
7	25	700
8	30	800

9	35	825
10	40	850
11	45	900
12	50	1000

2.2 Routing protocol

Routing adalah suatu protokol yang digunakan untuk mendapatkan rute dari satu jaringan ke jaringan yang lain. Rute ini, disebut dengan route dan informasi route secara dinamis dapat diberikan ke router yang lain ataupun dapat diberikan secara statis ke router lain. Routing protocol adalah komunikasi antara router-router. Routing protocol mengijinkan router-router untuk sharing informasi tentang jaringan dan koneksi antar router.

Semua routing protokol bertujuan mencari rute tersingkat untuk mencapai tujuan. Dan masing-masing protokol mempunyai cara dan metodenya sendiri-sendiri. Secara garis besar, routing protokol dibagi menjadi Interior Routing Protocol dan Exterior Routing Protocol. Keduanya akan diterangkan sebagai berikut :

2.2.1 Interior Routing Protocol

Sesuai namanya, interior berarti bagian dalam. Dan interior routing protocol digunakan dalam sebuah network yang dinamakan autonomus systems (AS) . AS dapat diartikan sebagai sebuah network (bisa besar atau pun kecil) yang berada dalam satu kendali teknik. AS bisa terdiri dari beberapa sub network yang masing-masingnya mempunyai gateway untuk saling berhubungan. Interior routing protocol mempunyai beberapa macam implemantasi protokol, yaitu :

- RIP (Routing Information Protocol)

Merupakan protokol routing yang paling umum dijumpai karena biasanya sudah included dalam sebuah sistem operasi, biasanya unix atau novell. RIP memakai metode distance-vector algoritma. Algoritma ini bekerja dengan menambahkan satu angka metrik kepada ruting apabila melewati satu gateway. Satu kali data melewati satu gateway maka angka metriknya bertambah satu (atau dengan kata lain naik satu

hop). RIP hanya bisa menangani 15 hop, jika lebih maka host tujuan dianggap tidak dapat dijangkau. Oleh karena alasan tadi maka RIP tidak mungkin untuk diterapkan di sebuah AS yang besar. Selain itu RIP juga mempunyai kekurangan dalam hal network masking.

- OSPF (Open Shortest Path First)

Merupakan protokol routing yang kompleks dan memakan resource komputer. Dengan protokol ini, route dapat dapat dibagi menjadi beberapa jalan. Maksudnya untuk mencapai host tujuan dimungkinkan untuk mecapainya melalui dua atau lebih rute secara paralel. Lebih jauh tentang RIP dan OSPF akan diterangkan lebih lanjut.

2.2.2 Exterior Protocol

AS merupakan sebuah network dengan sistem policy yang pegang dalam satu pusat kendali. Internet terdiri dari ribuan AS yang saling terhubung. Untuk bisa saling berhubungan antara AS, maka tiap-tiap AS menggunakan exterior protocol untuk pertukaran informasi routingnya. Informasi routing yang dipertukarkan bernama reachability information (informasi keterjangkauan). Tidak banyak router yang menjalankan routing protokol ini. Hanya router utama dari sebuah AS yang menjalankannya. Protokol yang mengimplementasikan exterior :

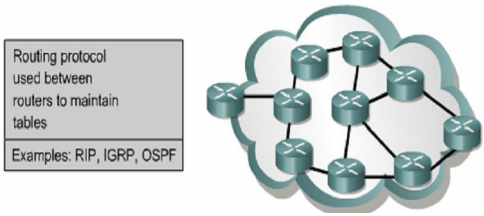
- EGP (Exterior Gateway Protocol)

Protokol ini mengumumkan ke AS lainnya tentang network yang berada di bawahnya. Pengumumannya kira-kira berbunyi : ” Kalau hendak pergi ke AS nomor sekian dengan nomor network sekian, maka silahkan melewati saya”. Router utama menerima routing dari router-router AS yang lain tanpa mengevaluasinya. Maksudnya, rute untuk ke sebuah AS bisa jadi lebih dari satu rute dan EGP menerima semuanya tanpa mempertimbangkan rute terbaik.

- BGP (Border Gateway Protocol)

BGP sudah mempertimbangkan rute terbaik untuk dipilih. Seperti EGP, BGP juga mepertukarkan reachability information.

Router menggunakan informasi ini untuk membangun dan memperbaiki table routingnya. Seperti terlihat pada gambar 2.3.



Gambar 2.3. Routing Protokol

2.3 Routing protokol pada WSN

Secara umum, Routing protocol pada WSN dibagi menjadi dua katagori :

§ Indicator-based
§ Indicator-free

Pada indicator-based, selalu terdapat fase inisialisasi dimana sebuah indicator algoritma tersebut digunakan. Berdasarkan algoritmanya, setiap node menyusun sebuah indicator untung membantu proses routing. Pada algoritma indicator-free, proses routing dibuat di udara.

2.4 Algoritma Indicator-Based

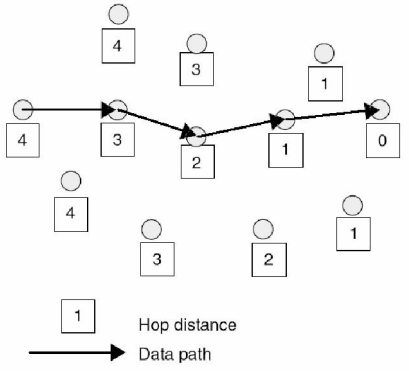
Pada katagori ini, indikatornya tersusun oleh sensor-sensor pada tahap setup. Sensor-sensor tersebut lalu mengikuti indicator ini untuk membuat suatu jalur routing yang tepat dan efisien. Berdasarkan pada perbedaan indicator, algoritma ini dibagi menjadi tiga subclass yaitu :

§ Geography-Based
§ Gradient-Based
§ Cluster-Based

2.5 Gradient Based

Daripada menggunakan jarak geometri yang absolut, solusi lain menggunakan jarak relatif antara node. Jarak relatif tersebut adalah "Gradien" yang menunjukkan arah aliran data dari pengirim tertentu. Beberapa pendekatan memilih sink sebagai sumber, seperti MCFN, GRAB, ARRIVE, dan GRAd. Yang lain memanfaatkan sumber data sebagai asal, seperti Direct Diffusion.

MCFN mempelajari masalah pengiriman paket dari sensor ke sink. Dalam MCFN, masing-masing node memiliki cost field sebagai biaya minimum dari node ke sink. Setelah cost field ditetapkan, paket mengalir ke sink sejalan dengan penurunan cost field. Paket hanya membawa biaya minimum dari sumber ke sink dan biaya yang dikeluarkan sampai ke node berikutnya. Forwarding node intermediate hanya menyebarkan paket ke semua tetangga tanpa menetapkan hop selanjutnya. Lingkungan penerima memutuskan untuk meneruskan jika jumlah biaya yang digunakan (yang terdapat dalam paket) dan biaya node (disimpan di node) cocok dengan biaya node sumber (yang terkandung dalam paket). Artinya, $Cost_{source} = Cost_{consumed} + Cost_{current\_node}$, berarti bahwa node sekarang pada jalur yang optimal. Dalam contoh ditunjukkan pada Gambar 2.4, jumlah hop digunakan sebagai cost field. Hal ini akan memastikan bahwa setiap node hanya mengiklankan cost fieldnya sekali dengan nilai yang benar. GRAB adalah versi mesh MFCN untuk meningkatkan kehandalan transmisi. Dalam Grab, daripada menggunakan 1 jalan optimal, sebuah mesh antara sumber dan sink terbentuk, dan semua node dalam mesh terlibat. Lebar mesh sebanding dengan jarak ke sink. Untuk membangun mesh, konsep kredit tambahan diperkenalkan sebagai anggaran yang dapat digunakan paket. GRAB sangat meningkatkan kinerja MCFN tanpa terlalu banyak memperkenalkan overhead-only header paket sedikit lebih lama. Dalam prakteknya, baik jumlah hop dan energi dapat digunakan sebagai cost field. Hal yang harus diperhatikan, bahwa MCFN dan GRAB keduanya perlu link simetris sehingga cost field yang dibentuk adalah valid ketika transmisi.



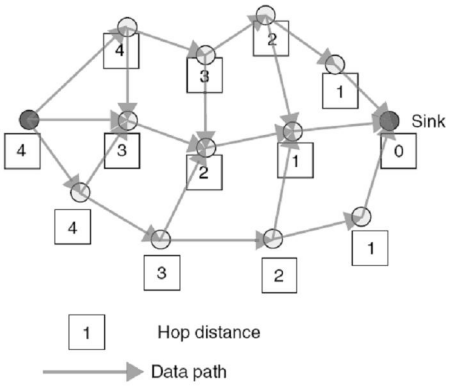
Gambar 2.4 Contoh Rute Transmisi pada MCFN

Kontribusi utama MCFN adalah bahwa ia menyajikan suatu skema untuk menghindari paket iklan berlebihan yang mungkin membingungkan jaringan saat membuat cost field. Algoritma MCFN adalah sebagai berikut :

1. Pertama kita inputkan jumlah node yang dibutuhkan
2. Tentukan biaya yang di butuhkan dari node ke sink
3. Tentukan Node Tetangga secara Acak
4. Pada Node tetangga cek apakah
   $Cost_{source}=Cost_{consumed}+Cost_{current\_node}$
   Jika tidak, kembali ke langkah 3
   Jika ya, ke langkah 5
5. Kirimkan ke node tersebut
6. Apakah sudah tiba pada tujuan?
   Jika tidak, kembali ke langkah 3
   Jika ya, program di hentikan

Dimana :

$Cost_{source}$ = Biaya Node Sumber
$Cost_{consumed}$ = Biaya Yang Digunakan
$Cost_{current\_node}$ = Biaya Pada Node



Gambar 2.5 Contoh rute transmisi mesh GARB dengan 1 extra kredit

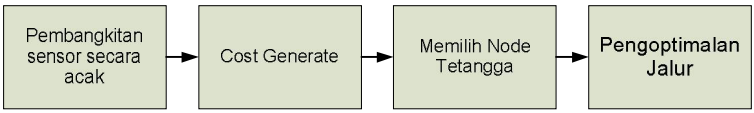
Pendekatan berbasis gradien tidak memerlukan informasi lokasi, tetapi perlu link simetris. Mereka diukur dengan jalur yang optimal. Namun, mereka menyediakan hanya sebagian kecil dari layanan pengiriman data, hanya untuk transmisi dari node lain ke sink. Algoritma GRAB adalah sebagai berikut :

1. Inputkan jumlah node yang dibutuhkan
2. Tentukan biaya yang di butuhkan dari node ke sink
3. Tentukan Node Tetangga secara Acak
4. Pada Node tetangga cek apakah
   $Cost_{source}=Cost_{consumed}+Cost_{current\_node}$
   atau
   $Cost_{source}=Cost_{current\_node}$
   Jika tidak, kembali ke langkah 3
   Jika ya, ke langkah 5
5. Kirimkan ke node tersebut
6. Apakah sudah tiba pada tujuan?
   Jika tidak, kembali ke langkah 3
   Jika ya, program di hentikan

= = = = Halaman ini sengaja dikosongkan = = = =

BAB III
PERANCANGAN SISTEM

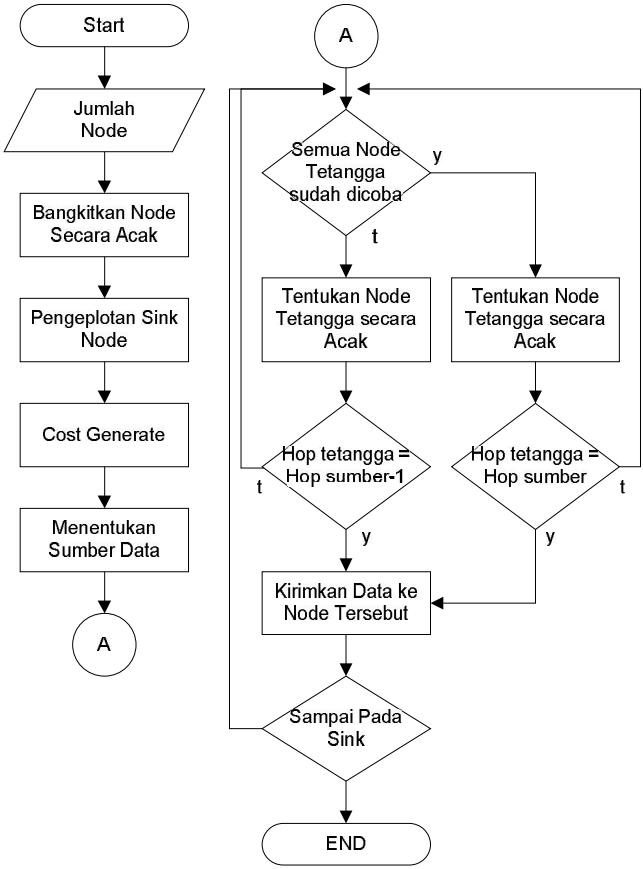
Blok diagram derancangan sistem dapat di lihat pada gambar 3.1. Perecnanaan sistem pada Sistem akan membangkitkan sensor secara acak setelah user memilih untuk membangkitkan sejumlah sensor dengan jumlah maksimal adalah 30 buah, setelah seluruh sensor telah di bangkitkan user dapat memilih sink node hingga cost field sebagai biaya minimum dari masing-masing node dapat terbentuk berdasarkan jumlah hop dari node tersebut hinnga sink node. User akan menetapkan sumber node(sumber data berasal) dari node sumber akan dipilih node tetangga sebagai penerus data menuju sink node menggunakan state transition. Data mengalir ke sink sesuai penurunan biaya, masing-masing node tidak akan mengirimkan data jika node yang dipilih bukan jalur yang optimal[1].



Gambar 3.1 Blok Diagram Perancangan Sistem

3.1 CARA KERJA

Perancangan sistem didasarkan pada perencanaan area simulasi, jumlah node sensor, dan system komunikasi antara node dengan sink. Area simulasi ditetapkan sebagai luasan yang didefinisikan sebagai sumbu x dan sumbu y. Jumlah node yang telah ditentukan kemudian diletakkan atau diposisikan dengan aturan yang telah ditentukan, dimana dalam perancangan ini jumlah sensor dibatasi sebanyak 30 buah dan sensor diposisikan secara random. Parameter – parameter simulasi ini diinisialisasikan pada program yang dibuat, pemrograman yang dibuat merupakan bahasa pemrograman berbasis JAVA. Flowchart perencanaan system ditunjukkan pada gambar 3.2.



Gambar 3.2 Flowchart Algoritma Gradient Based Approach

Pada gambar flowchart diatas dapat dijelaskan kembali langkah– langkah atau proses peembuatan yaitu pada proses awal adalah untuk menentukan jumlah node yang akan digunakan pada simulasi ini dalam hal ini jumlah node telah dibatasi oleh pemrogram maksimal sebanyak 30 node, kemudian langkah selanjutnya adalah mendapatkan posisi node – node yang telah disebar tersebut, meskipun pada simulasi ini node disebar secara random posisi node perlu diketahui untuk memudahkan dalam proses – proses selanjutnya. Kemudian proses selanjutnya adalah menentukan Sink Node, pada setiap node akan mentukan jarak Hop dari node tersebut hingga menuju Sink-nya.

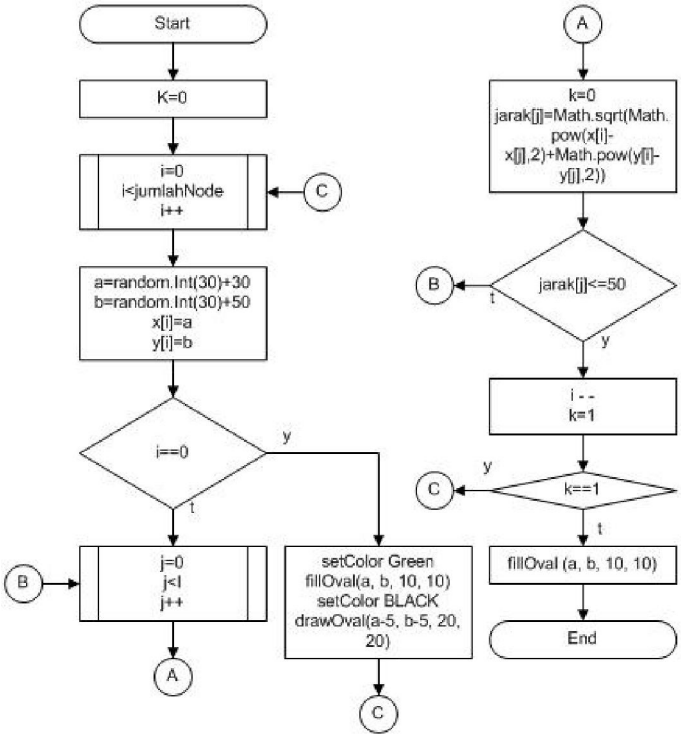
Kemudian user akan menetapkan sumber node (sumber data berasal), dari node sumber akan dipilih node tetangga sebagai penerus data menuju sink kemudian dicek apakah node yang di pilih merupakan jalur yang optimal dengan membandingkan apakah Hop tetangga = Hop sumber-1, jika kondisi tersebut terpenuhi maka data akan di teruskan pada node tersebut, jika kondisi tersebut tidak terpenuhi maka data tidak akan di kirimkan kepada node tersebut dan akan memilih node tetangga yang lain yang dapat memenuhi kondisi tersebut. Ketika semua node tetangga sudah dicoba dan tidak ada node tetangga yang dapat memenuhi persyaratan tersebut, maka akan di terapkan sistem 1 extra credit dengan membandingkan apakah Hop tetangga = Hop Sumber, jika kondisi tersebut terpenuhi maka data akan diteruskan pada node tersebut, jika tidak terpenuhi maka akan mengecek node-node yang lain. Proses tersebut terus berulang hingga data sampai pada Sink node-nya.

3.2 ALGORITMA GRADIENT BASED APPROACH

1. Penentukan jumlah Node
2. Penempatan Node secara random
3. Penentuan Sink Node
4. Penentuan Cost
5. Penentuan sumber data
6. Semua Node Tetangga sudah dicoba?
    -ya, ke langkah 9
    -tidak, ke langkah 7
7. Penentuan Node Tetanngga secara acak
8. Hop tetangga = Hop sumber-1 ?
    -ya, ke langkah 11
    -tidak, ke langkah 6
9. Penentuan Node Tetanngga secara acak
10. Hop tetangga = Hop sumber ?
    -ya, ke langkah 11
    -tidak, ke langkah 6
11. Kirimkan data ke Node tersebut
12. Sudah sampai pada Sink?
    -ya, ke langkah 13
    -tidak, ke langkah 6
13. selesai

3.3 PEMBANGKITAN SENSOR

Dalam pembangkitan sensor, dibuat deploy secara acak dengan syarat penyebaran sensor harus merata, agar coverage area dengan jumlah node tertentu dapat sedikit lebih optimal. Flowchart pembangkitan node dapat dilihat pada gambar 3.3.



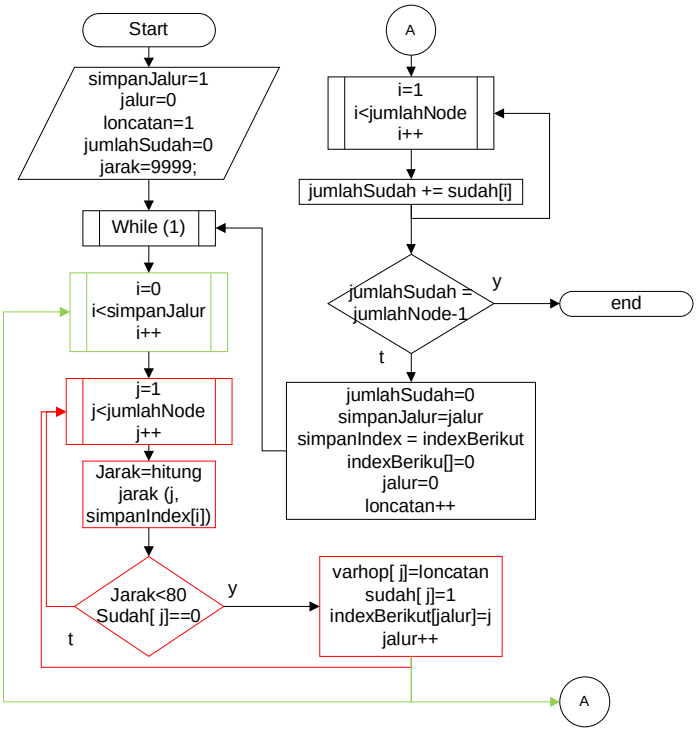
Gambar 3.3 Flowchart Pembangkitan Sensor Acak

Pada gambar di atas dapat dijelaskan secara global, langkah awal diinisialisasikan variabel k dengan nilai ’0’, setelah itu terdapat proseses looping sebanyak inputan jumlahNode. Untuk pembangkitan pertama akan langsung di set warna menjadi hijau dan di gambarkan lingkaran dengan diameter dua kali lebih besar yang akan membadakan node ini dengan node-node yang lain, menjadikan node ini sebagai sink. Untuk pembangkitan node-node berikutnya akan dicek terlebih dahulu

dengan koordinat sekian apakah terdapat node-node yang lain (node yang sebelumnya telah dibangkitkan) dalam jarak 50, jika iya maka proses looping pada saat itu akan di abaikan (i--) dan k akan diinisialisasikan dengan nilai ’1’, ketika nilai k samadengan ’1’ proses penggambaran node tidak akan di lakukan, lalu akan membangkitkan koordinat yang baru. Ketika pada koordinat tertentu tidak terdapat node dalam jarak 50, maka node akan digambar dan proses looping akan berjalan kembali.

3.4 COST GENERATE

Dalam pembangkitan nilai Cost Field digunakan perhitungan hop sebagai fungsinya. Flowchart cost generate dapat dilihat pada gambar 3.4.



Gambar 3.4 Flwochart Cost Generate

Pada gambar diatas dapat di jelaskan secara global, adalah pogram looping yang tidak akan berhenti, dan hanya akan berhenti ketika dilakukan break ketika jumlahSudah (jumlah sudah adalah jumlah node yang sudah diberi variable hop) sama dengan jumlah node yang dikurangi dengan 1 (jumlah node yang tidak termasuk sink node).

3.5 PERENCANAAN PROGRAM

Dari hasil perancangan, akan dilakukan pembuatan simulasi Routing Protokol dengan menggunakan Gradient Based pada Wireless Sensor Network yang akan menentukan routing mana yang paling efisien untuk proses pengiriman paket data dengan menggunakan JAVA.

Pertama dibuat class-class yang mendukung pembuatan simulasi Gradient Based Routing seperti gambar berikut.

Gambar 3.5 Class-class pendukung

Dari tampilan gambar 3.5 diketahui bahwa GradientBased.java merupakan main class nya. Main class tersebut akan digunakan sebagi tempat pembuatan simulasi, atau lebih dikenal sebagai frame.

Sesuai dengan rancangan, simulasi menggunakan metode/algoritma Gradient based routing yang mana penentuan arah routingnya didasarkan pada biaya yang terkecil. Pada simulasi dibuat node-node yang sudah ditentukan biaya. Kemudian arah routingnya ditentukan berdasarkan penurunan biaya hingga sampai ke node destination. Setiap node yang akan mengirimkan datanya ke node berikutnya akan memastikan terlebih dahulu bahwa

$$Cost_{source} = Cost_{consumed} + Cost_{current\_node} \quad (1)$$

Dimana :

$Cost_{source}$ = Biaya Node Sumber
$Cost_{consumed}$ = Biaya Yang Digunakan
$Cost_{current\_node}$ = Biaya Pada Node

Jika terpenuhi maka data akan dikirimkan ke node berikutnya.

= = = Halaman ini sengaja di kosongkan = = =

BAB IV
IMPLEMENTASI SISTEM DAN ANALISA

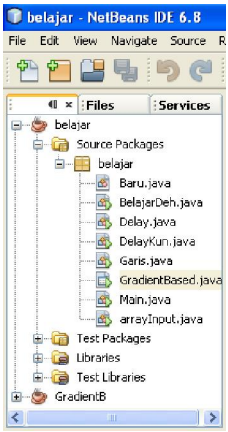
4.1 PEMBUATAN SISTEM

Pada bab ini dimaksudkan untuk mengetahui keseluruhan pengujian dari perencanaan hasil sistem yang telah dibuat. Dengan demikian akan diketahui tingkat keberhasilan dari sistem yang telah dibuat.

Class-class yang dibuat terdiri dari :

1. Baru.java
2. GradientBased.java

Contoh tampilan class dapat dilihat pada gambar 4.1.



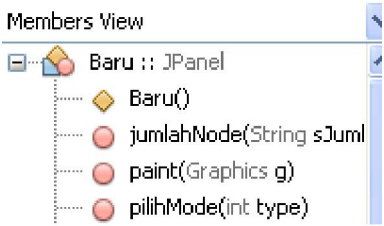
Gambar 4.1 Tampilan class

Pada gambar 4.1 terlihat Baru.java merupakan class dengan extends JPanel, yang dengan secara otomatis di dalam class ini dapet menggunakan method-method yang terdapat pada class JPanel. Pada class ini akan di buat program-program utama untuk menajalankan program simulasi, Routing protokol pada Wireless Sensor Network menggunakan algoritma Gradient Based Approach.

Didalam class ini terdapat, Constructor, beberapa method, dan beberapa parameter. Method paint(Graphics g) berisi program utama

untuk melakukan program dari awal hingga selesai, pada method ini terdapat 3 tugas utama, yang semuanya itu adalah menggambar, dikarenakan hal itu method ini menggunakan parameter yaitu Graphics g. Tiga tugas utama disini yang pertama adalah melakukan penggambaran Node secara random, yang nantinya setiap Node memiliki nilai koordinat x dan y, lalu menyimpan koordinat tersebut, karena untuk proses – proses selanjutnya kita akan sellau membutuhkan nilai – nilai tersebut. Tugas ke-2 dari method paint(Graphics g) ini adalah, untuk menghitung nilai hop yang terkandung pada node, dan menyimpannya pada variabel HOP(HOP adalah atribut pada class ini), dan tugas ke-3 adalah untuk menentukan rute-rute pada setiap node, agar dapat sampai pada sinknya. Hanya saja ketiga program utama disini baru dapat di jalankan ketika suatu kondisi terpenuhi, yaitu mode yang di pilih adalah sesuai dengan penugasan utama tersebut.

```
if(mode == RANDOM){
    Random generator = new Random();
    double[] jarak = new double[jumlahNode];
    int k = 0;
    for (int i=0; i<jumlahNode; i++){
       int a = generator.nextInt(300)+30;
       int b = generator.nextInt(300)+50;
       x[i]=a;
       y[i]=b;
if(mode == HOP){
    double[] jarak = new double[jumlahNode];
    int ici;
    char ci;
    for (int i=0; i<jumlahNode;i++){
       if(i==0){
          varhop[i]=0;
       }
       if(i!=0){
          jarak[i]=Math.sqrt(Math.pow(x[i]-
x[0],2)+Math.pow(y[i]-y[0],2));
       }
if(mode == ROUTE){
    int []sudah = new int[jumlahNode];
    double[] jarak = new double[jumlahNode];
    for (int i=0;i<jumlahNode;i++){
       if(i!=0){
```



Gambar 4.2 Members View Method and Constructor

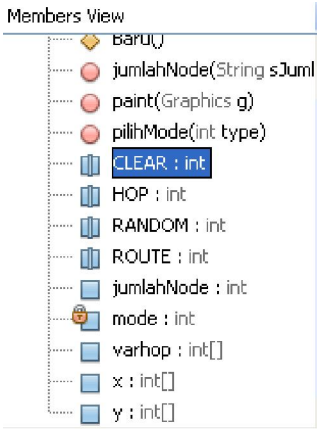
Method pilihMode() disini, bertugas untuk mengisi variable mode, lalu melakukan penggambaran ulang repaint(), method ini memiliki parameter yaitu int type, jadi ketika method ini dipanggil harus disertakan dengan argumen berupa nilai integer, yang kemudian masuk method ke variabel type, lalu method memasukkan nilai ini ke variabel (atribut) mode.

```
public void pilihMode(int type){
    mode = type;
    repaint();
}
```

Pada class ini memiliki atribut jumlahNode, mode, varhop, x[], dan y[], dan beberapa atribute yang bersifat final yaitu CLEAR, HOP, RANDOM, dan ROUTE. Atribut jumlahNode disini digunakan untuk menentukan jumlah node yang akan di bangkitkan secara random, dan kebanyakan dijadikan batas pada proses looping.

Method jumlahNode() disini, bertugas untuk mengisi variable jumlahNode, method ini memiliki parameter yaitu String sJumlahNode, jadi ketika method ini di panggil harus di sertakan dengan argumen berupa String. Argumen ini akan dirubah / dikonversikan menjadi nilai integer, hasil konversi ini akan di masukkan kedalam variable (atribut) jumlahNode.

```
public void jumlahNode(String sJumlahNode){
    jumlahNode = Integer.parseInt(sJumlahNode);
}
```

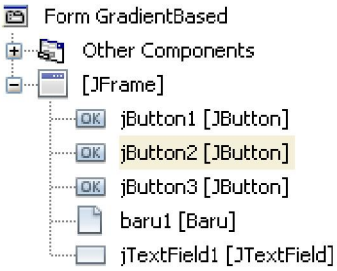


Gambar 4.3 Members View Atributs

Class Baru.java dibuat sedemikian rupa sehingga dapat membuat tampilan-tampilan yang diharapkan. Namun demikian class ini tidak memiliki main method, sehingga class ini tidak dapat di running, dan hanya merupakan sebuah container yaitu content pane.

Class GradientBased.java merupakan main class yang di dalamnya terdapat main method. Pada class ini kita membuat Frame sebagai Container utama, dan memanggil class Baru.java sebagai panel pada Frame ini, menggunakan Beans yang sudah tersedia saat kita membuat class ini menggunakan Swing GUI Forms.

Pada class GradientBased.java ini terdapat satu TextField dan tiga Button, dengan masing – masing component tersebut meiliki actionPerformed yang berbeda – beda.



Gambar 4.4 GradientBased Components

Untuk jButton1 adalah Random Generate, ketika di click akan melakukan actionPerformed memanggil method pilihMode() dengan argument Baru.RANDOM pada class Baru. Program ini bertujuan untuk mengisi variabel mode pada class Baru dengan isi yang sama dengan variable final RANDOM pada class Baru. Lalu menggambar ulang.

```
        private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
                // TODO add your handling code here:
                baru1.pilihMode(Baru.RANDOM);
            }
```

Untuk jButton2 adalah Hop Generate, ketika di click akan melakukan actionPerformed memanggil method pilihMode() dengan argument Baru.HOP pada class Baru. Program ini bertujuan untuk mengisi variabel mode pada class Baru dengan isi yang sama dengan variable final HOP pada class Baru. Lalu menggambar ulang.

```
        private void
jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
                // TODO add your handling code here:
                baru1.pilihMode(Baru.HOP);
            }
```

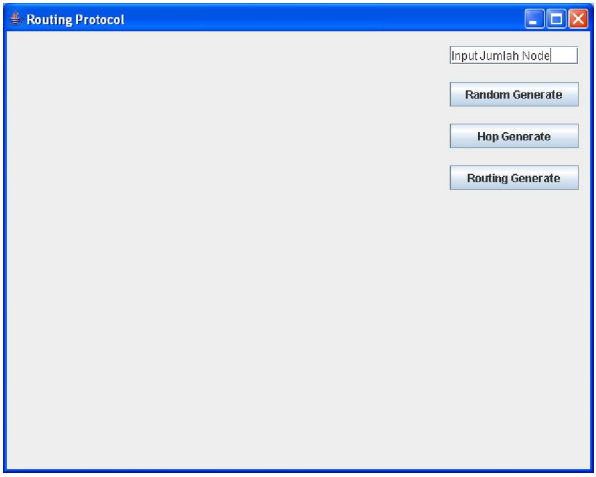
Untuk jButton3 adalah Routing Generate, ketika di click akan melakukan actionPerformed memanggil method pilihMode() dengan argument Baru.ROUTE pada class Baru. Program ini bertujuan untuk mengisi variabel mode pada class Baru dengan isi yang sama dengan variable final ROUTE pada class Baru. Lalu menggambar ulang.

```
        private void
jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
                // TODO add your handling code here:
                baru1.pilihMode(Baru.ROUTE);
            }
```

Untuk jTextField1 adalah Routing Generate, ketika di Enter akan melakukan actionPerformed memanggil method jumlahNode() dengan argument evt.getActionCommand(). Program ini bertujuan untuk mengisi variabel jumlahNode pada class Baru dengan isi sesuai dengan text yang di masukkan pada jTextField tersebut.

```
        private void
jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
                // TODO add your handling code here:
                baru1.jumlahNode(evt.getActionCommand());
            }
```

Tampilan awal setelah running GradientBased.java akan terlihat seperti pada gambar 4.5.



Gambar 4.5 Hasil running GradientBased.java

4.1.1 Pembangkitan Sensor Secara Acak

Ketika mode yang diaktifkan adalah RANDOM, maka program dibawah yang dijalankan dari method paint(). Pada awal program di buat objek Random yang di beri nama generator, membuat array jarak dengan tipe data double, dengan panjang array sejumlah jumlahNode, dan menginisialisasi variable k bertipe data int dengan data ‘0’.

Dilakukan perulangan sebanyak jumlah node yang di-inputkan, dan menggunakan variabel "i" sebagai variabel penandanya, betambah satu setiap perulangan terjadi. Dalam perulangan tersebut, dibangkitkan bilangan bertipe data integer random dengan range 30 sampai dengan 330, bilangan tersebut akan disimpan dalam variabel array "x" pada index ke-"i" sebagai koordinat sumbu x, pada koordinat kartesian. Sedangkan untuk variabel array "y", koordinat y dibangkitkan bilangan dengan range

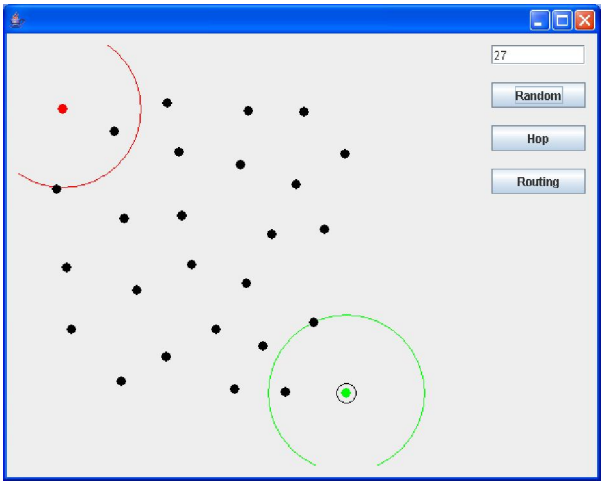
50 sampai dengan 350. Untuk koordinat node pertama adalah Sink dan node yang kedua adalah Source Node, diatur dengan koordinat tertentu yaitu (40, 60) dan (330, 350). Diletakkan pada posisi terjauh diluasan area yang digunakan.

Koordinat - koordinat tersebut akan digunakan untuk membangkitkan sensor node. Untuk pengeplotan sensor ke-3 dan seterusnya, akan dilakukan perhitungan jarak antara node yang akan diplot dan seluruh node yang sudah diplot sebelumnya, untuk node ke-3 akan dilakukan perhitungan terhadap node ke-2 (source node) dan node ke-1 (Sink). Jika ada salah satu jarak dengan node yang sudah terplot ada yang kurang dari 50, maka akan dilakukan pembangkitan nilai koordinat yang lain. Program yang dimaksut adalah sebagai berikut :

```
if(mode == RANDOM){
    Random generator = new Random();
    double[] jarak = new double[jumlahNode];
    int k = 0;
    for (int i=0; i<jumlahNode; i++){
        int a = generator.nextInt(300)+30;
        int b = generator.nextInt(300)+50;
        x[i]=a;
        y[i]=b;
        energi[i]=2000;// Dalam satuan Mili joule
        if (i==0){//kond 1, go
            x[i]=330;
            y[i]=350;
            g.setColor(Color.GREEN);
            g.fillOval(x[i], y[i], 10, 10);//penggambaran node
            g.drawOval(x[i]-75, y[i]-75, 160, 160);
            g.setColor(Color.BLACK);
            g.drawOval(x[i]-5, y[i]-5, 20, 20);
        }else{//kond1, stop; kond2, go
            if(i==1){
                x[i]=40;
                y[i]=60;
                g.setColor(Color.red);
                g.fillOval(x[i], y[i], 10, 10);
                g.drawOval(x[i]-75, y[i]-75, 160, 160);
                g.setColor(Color.BLACK);
            }else{
            for (int j=0; j<i; j++){
```

```
                k=0;
                jarak[j]=Math.sqrt(Math.pow(x[i]-
x[j],2)+Math.pow(y[i]-y[j],2));

                if(jarak[j]<=50){
                   i--;
                   k=1;
                   break;
                }
             }
             if(k==1){
                continue;
             }
             g.fillOval(a, b, 10, 10);//penggambaran node
             //g.drawOval(x[i]-75, y[i]-75, 160, 160);
          }
          }//kond2, stop
       }
       System.out.println("Penyebaran Node Selesai");
       System.out.println();
```



Gambar 4.6 Tampilan pembangkitan node

Inputkan jumlah node, lalu tekan tombol Random. Setelah itu ketika tombol Random Generate ditekan akan terlihat tampilan seperti pada gambar 4.6. Pada gambar 4.6, adalah contoh tampilan

pembangkitan node secara random dengan jumlah node yang telah dibatasi oleh pemrogram yaitu sebanyak 27 node. Pada simulasi ini posisi sink dan source node telah ditentukan. Untuk source node pada koordinat (40, 60) untuk sink pada koordinat (330, 350). Titik merah merupakan Source Node, merupakan Node yang merupakan sumber data. Titik hijau merupakan Sink, tempat tujuan terahir semua data dikumpulkan.

4.1.2 Pembangkitan Nilai Hop

Ketika variable mode berisi sama dengan variable HOP, maka program yang dijalankan adalah pogram looping yang tidak akan berhenti, dan hanya akan berhenti ketika dilakukan break ketika jumlahSudah (jumlah sudah adalah jumlah node yang sudah diberi variable hop) sama dengan jumlah node yang dikurangi dengan 1 (jumlah node yang tidak termasuk sink node). Setiap while ini melakukan akan menambah variabel loncatan, yang nantinya akan digunakan sebagai nilai dari varhop dari setiap node.

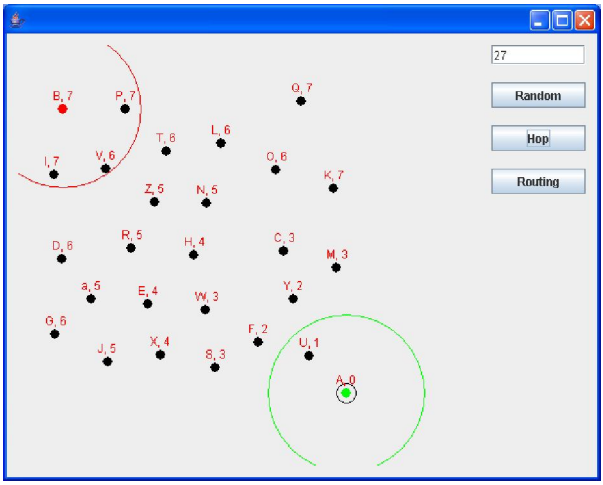
Didalam program while dilakukan perulangan ”a” sebanyak simpanJalur. Didalam perulangan ”a” terdapat perulangan ”b” yang dilakukan sejumlah node yang dikurangi 1, untuk menghintung jarak semua node (kecuali sink, yang merupakan pengurangan 1) terhadap node dengan index yang tersimpan pada simpanIndex[a]. Ketika jarak masuk ke coverage area yang di tentukan dan pada node yang ke-b belum menyimpan nilai hop, maka nilah hop disikan dengan nilai loncatan.

Untuk pemberian identitas pada setiap node, pada pembangkitan pertama akan di berikan abjad ’A’ dan seterusnya hingga ’Z’, jika jumlah node lebih dari 26, maka akan di lanjutkan dengan abjad ’a’, dst. Program yang dimaksut adalah sebagai berikut :

```
if(mode == HOP){
    int simpanJalur=1;
    int jalur=0;
    int loncatan=1;
    int jumlahSudah=0;
    int [] indexBerikut = new int[60];
```

```
        int [] simpanIndex = new int [60];
        int [] sudah = new int [jumlahNode];
        double jarak=9999;
        int ici;
        char ci;
        while (Boolean.TRUE){
           for(int i=0;i<simpanJalur;i++){
              for(int j=1;j<jumlahNode;j++){
                 jarak=Math.sqrt(Math.pow(x[j]-
x[simpanIndex[i]],2)+Math.pow(y[j]-y[simpanIndex[i]],2));
                 if(jarak<80 && sudah[j]==0){
                    varhop[j]=loncatan;
                    sudah[j]=1;
                    indexBerikut[jalur]=j;
                    jalur++;
                 }
              }
           }
           for(int i=1;i<jumlahNode;i++)
              jumlahSudah+=sudah[i];
           if(jumlahSudah==jumlahNode-1)break;
           jumlahSudah=0;
           simpanJalur=jalur;
           for(int i=0;i<simpanJalur;i++){
              simpanIndex[i]=indexBerikut[i];
              indexBerikut[i]=0;
           }
           jalur=0;
           loncatan++;
        }
        ici=0;
        for(int i=0;i<jumlahNode;i++){
           if(i<=25){
              ici=65+i;
           }
           ci=(char)ici;
```

```
            g.setColor(Color.red);
            g.drawString(""+ci+", "+varhop[i], x[i]-5, y[i]-4);
            if(i>25){
               ici=71+i;
            }
        }
    }
```



Gambar 4.7 Tampilan pembangkitan nilai Hop

Jika ditekan tombol Hop Generate akan telihat tampilan seperti gambar 4.7. Pada gambar 4.7, adalah tampilan program pembangkitan nilai hop yang terlihat dan terletak pada sebelah kanan dari posisi penamaan node. Sedangkan abjad yang tertera pada penamaan node adalah sesuai dengan urutan pembangkitan node tesebut.

4.1.3 Pembentukan Jalur Ruting

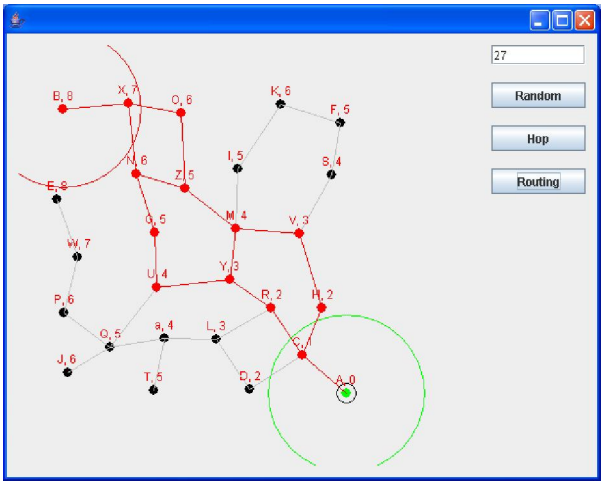
Ketika mode yang dipilih adalah ROUTE, maka akan dilalukan proses looping sejumlah inputan node, untuk mencari node tetangga dari suatu node pada coverage area tertentu. Ketika menemukan suatu node tetangga, maka akan dilakukan pengecekan apakah node tetangga tersebut merupakan jalur yang optimal, data tidak akan

dikirimkan ketika node tersebut bukanlah jaluk yang optimal. Ketika pada coverage area tersebut tidak satupun ditemukan node tetangga yang merupakan jalur optimal, untuk menghindari terputusnya jalur ruting maka diterapkan sistem kredit untuk menemukan jalur alternatifnya. Proses tersebut terus berulang hingga semua node yang di bangkitkan telah menemukan jalur routingnya hingga sampai pada sink node.

Pada program terlihat terdapat dua pengkondisian dalam satu loop pencarian node tetangga, kondisi pertama ketika hop tetangga sama dengan hop sumber yang dikurangi 1. Ini adalah kondisi yang harus dipenuhi agar terbentuknya jalur optimal. Kondisi kedua ketika hop tetangga sama dengan hop sumbernya, merupakan sistem kredit yang diterapkan untuk mendapatkan jalur alternantif. Program yang dimaksut adalah sebagai berikut :

```
if(mode == ROUTE){
      int []sudah = new int[jumlahNode];
      int []sudah1 = new int[jumlahNode];
      double[] jarak = new double[jumlahNode];
      int bit=100;
      double ET=bit*50;//Dalam Nano Joule
      double EK;
      int[] HopBerikut = new int [20];
      int[] simpanHopBerikut = new int [20];
      int jalur=0;
      int simpanJalur=0;
      int selesai=0;
      for (int i=0;i<jumlahNode;i++){
         if(i!=0){
            for(int j=0;j<jumlahNode;j++){
               jarak[j]=Math.sqrt(Math.pow(x[i]-
x[j],2)+Math.pow(y[i]-y[j],2));
               if(jarak[j] < 80 && varhop[j]==varhop[i]-1){
                  try { //progran dellay
                     Thread.sleep(200);
                  }catch (InterruptedException ie) {
                     ie.printStackTrace();
```

```
                }//end of dellay program
                //energi[i]-=150;
                //energi[j]-=150;
                g.setColor(Color.LIGHT_GRAY);
                g.drawLine(x[i]+5, y[i]+5, x[j]+5, y[j]+5);
                sudah[i]=1;
            }
        }
        if(sudah[i]!=1){
            for(int j=0;j<jumlahNode;j++){
                jarak[j]=Math.sqrt(Math.pow(x[i]-
x[j],2)+Math.pow(y[i]-y[j],2));
                if(jarak[j] < 80 && varhop[j]==varhop[i]){
                    try { //progran dellay
                        Thread.sleep(10);
                    }catch (InterruptedException ie) {
                        ie.printStackTrace();
                    }//end of dellay program
                    g.setColor(Color.LIGHT_GRAY);
            //        energi[i]-=150;
            //        energi[j]-=150;
                    g.drawLine(x[i]+5, y[i]+5, x[j]+5, y[j]+5);
                    sudah[i]=1;
                }
            }
        }
    }
    for (int i=0; i<jumlahNode; i++){
        System.out.println("energi pada node ke-"+i+" adalah =
"+energi[i]);
    }
}
```



Gambar 4.8 Tampilan pembangkitan jalur routing

Jika ditekan tombol Routing Generate akan terlihat tampilan seperti gambar 4.8. Pada gambar 4.8, merupakan tampilan perutean data yang terbentuk pada simulasi. Untuk jalur dengan warna abu-abu adalah semua kemungkinan koneksi yang dapat dibentuk. Sedangkan jalur dengan warna merah adalah rute yang terbentuk dari Source Node hingga ke Sink Node. Pada Source Node B yang memiliki nilai Hop 8, walaupun terlihat dekat dengan node E, node B tidak dapat mengirimkan data ke node tersebut, terlihat dari gambar node E diluar dari coverage area node B, melainkan mengirimkan data ke node X, dalam coverage area node X terdapat 2 node yang memungkinkan untuk menerima data yaitu O dan N. Data ditransmisikan ke kedua node tersebut, hal ini terjadi juga pada node N, dan node M. Terbentuk 5 jalur yang berbeda yaitu BXOZMVHCA; BXOZMYRCA; BXNZMVHCA; BXNZMYRCA; dan BNGUYRCA. Hal ini membuat data masih tetap dapat dirutekan melalui jalur yang lain ketika salah satu jalur tidak memungkinkan untuk dilalui.

4.2 PENGUJIAN DAN ANALISA

Pada setiap WSN(Wireless Sensor Network), energi selalu menjadi salah satu parameter yang digunakan. Hal ini dikarenkan WSN merupakan perangkat yang membutuhkan baterai dan

kebanyakan digunakan untuk memonitor daerah-daerah yang sulit dijangkau manusia, life-time benar-benar harus dipertimbangkan.

Pada Proyek Akhir ini diasumsikan transmiter dan receifer membutuhkan energi yang sama E_{elec} untuk mentransmisikan dan umtuk menerima satu bit data. Sedangkan untuk pengirim juga membutuhkan energy $\varepsilon_{fs}\ d^2$ atau $\varepsilon_{mp}\ d^4$, tergantung dari jarak transmisinya[5]. Persamaan Energi transmit E_{Tx} sejumlah l-bit data pada suatu jarak d adalah sbb :

$$(,) = \begin{matrix} + \ \varepsilon_f \ d , & d < d \\ + \ \varepsilon \ \ d , & d \geq d \end{matrix} \qquad (2)$$

Untuk Energi Receifer E_{Rx} adalah sbb :

$$( ) = \qquad (3)$$

dimana :

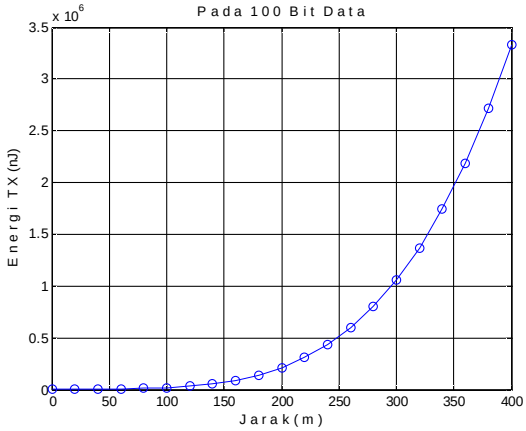
E_{Tx} : Energi Transmiter
E_{Rx} : Energi Receifer
l : Jumlah bit data
d : Jarak antar node
d_0 : Jarak Treshold 87,7 m
E_{elec} : Energi yang di 50nJ / bit
ε_{fs} : 10pJ / (bit . m^2)
ε_{mp} : 0,0013pJ / (bit . m^4)

Dari rumusan diatas didapatkan grafik yang dapat dilihat pada gambar 4.5. Yang menunjukan hubungan antara jarak (d), jumblah bit data (l), dan Energi. Dari rumusan di atas, dapat kita ketahui nilai Energi terhadap jarak, dengan 100 bit data, pada tabel 4.1.

Tabel 4.1 Energi terhadap jarak

Jarak (m)	Energi Tx (nJ)
0	5000
20	5400
40	6600
60	8600
80	11400
100	18000
120	31956.8
140	54940.8
160	90196.8
180	141468.8
200	213000
220	309532.8
240	436308.8
260	599068.8
280	804052.8
300	1058000
320	1368148.8
340	1742236.8
360	2188500.8
380	2715676.8
400	3333000

Dari tabel diatas didapatkan grafiknya seperti yang ditunjukkan pada Gambar 4.9.



Gambar 4.9 Grafik energi Tx terhadap jarak

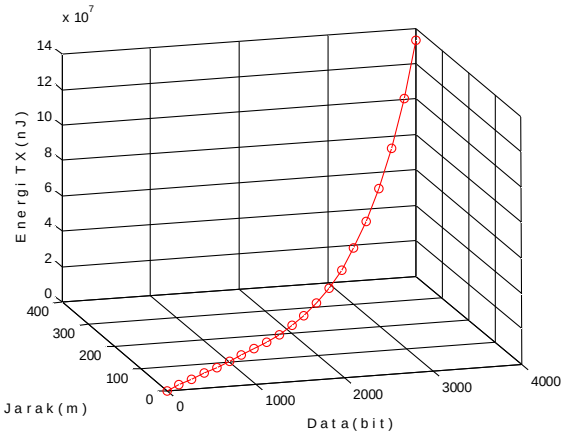
Terlihat pada grafik diatas merupakan energi yang dibutuhkan untuk mengirimkan data sebesar 100 bit pada jarak-jarak tertentu. Pada grafik di atas, kita dapat mengetahui jarak minimal agar dapat menggunakan energy se-efisien mungkin, yaitu pada $d < d_0$. Untuk nilai energi transmit terhadap kedua parameter, jarak dan jumlah bit data, dapat dilihat pada tabel 4.2.

Tabel 4.2 Energi terhadap jarak dan bit data

Bit (bit)	Jarak (m)	Energi Tx (nJ)
0	0	0
200	20	10800
400	40	26400
600	60	51600
800	80	91200
1000	100	180000
1200	120	383481,6
1400	140	769171,2
1600	160	1443148,8
1800	180	2546438,4
2000	200	4260000
2200	220	6809721,6
2400	240	10471411,2
2600	260	15575788,8

2800	280	22513478,4
3000	300	31740000
3200	320	43780761,6
3400	340	59236051,2
3600	360	78786028,8
3800	380	103195718
4000	400	133320000

Dari tabel diatas didapatkan grafiknya seperti yang ditunjukkan pada Gambar 4.10.



Gambar 4.10 Energi terhadap jarak dan bit data

Terlihat pada gambar 4.10, semakin tinggi data dan jaraknya maka semakin tinggi pula energi yang dibutuhkan untuk transmit. Dapat dilihat pada grafik meningkat derastis setelah jarak 100 m dan 1000 bit data.

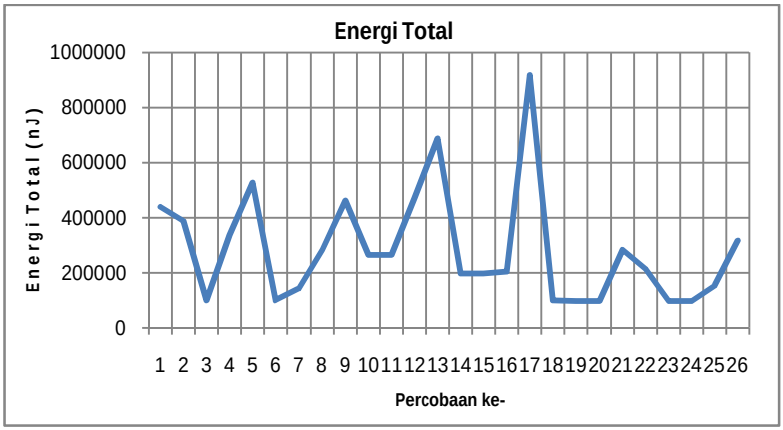
Percobaan selanjutnya adalah untuk menghitung energi dalam satu rute (dari sumber ke sink). Dengan koordinat Source Node [330, 350] (Pixel) dan Destination Point [40, 60] (Pixel), jarak antara Source dan Destination adalah 410,1219 (Pixel), jumlah node adalah 27, dengan area panjang dan lebar 469 x 429, Besar data yang ditransmisikan sebersar 100 bit. Dilakukan 26 kali percobaan untuk mendapatkan nilai rata-rata energi yang dibutuhkan untuk

algoritma Gradient Based Approach. Berikut tabel energi rata-rata yang dibutuhkan menggunakan algoritma Gradient Based Approach.

Tabel 4.3 Total energi yang dibutuhkan

Percobaan ke-	Energi Total (nJ)
1	440434
2	389326
3	100356
4	338737
5	529116
6	100006
7	142406
8	285300
9	464170
10	266969
11	265074
12	473117
13	689759
14	198546
15	198546
16	204426
17	919023
18	100324
19	99476
20	99626
21	284061
22	214283
23	98510
24	99358
25	153990
26	317994
Rata - rata	287420.5

Dari data pada table 4.3, akan dibuat grafik seperti pada gambar 4.11.



Gambar 4.11 Grafik energi total

Pada gambar 4.11, merupakan gambar grafik Energi yang dibutuhkan dalam satu rute (dari sumber ke sink). Total energi yang dimaksud adalah total energi yang dibutuhkan dari sensor sumber mengirim data ke sensor tetangga, energi yang dibutuhkan sensor tetangga untuk menerima data dari sensor sumber, seterusnya hingga data berhasil dikirim pada sink. Dengan energi kirim E_{Tx} dan energi terima E_{Rx} sesuai dengan persamaan (2) dan (3). Dilakukan sebanyak 26 kali percobaan.

Pada setiap percobaan didapatkan nilai energi dibutuhkan dalam satu rute adalah tidak sama. Hal ini disebabkan peletakan sensor node yang acak, menyebabkan jumlah hop yang dibutuhkan hingga sampai ke sink berbeda-beda. Selain menyebabkan jumlah hop yang dibutuhkan singga sampai ke sink yang berbeda, jarak antar hop yang selalu berbeda-beda pula, sesuai dengan rumus (2) dimana E_{Tx} merupakan fungsi dari jumlah bit data & jarak. Dalam 26 kali percobaan dapat kita ketauhui bahwa pengiriman 100 bit data dalam satu perutean (dari sumber ke sink) membututuhkan energi rata-rata sebesar 287420,5 nJ.

BAB V
KESIMPULAN DAN SARAN

5.1 KESIMPULAN

Dari analisa data terhadap simulasi Gradient Based Routing dapat disimpulkan bahwa :

- Ketika salah satu jalur tidak memungkinkan untuk dilalui, maka data masih tetap dapat dirutekan melalui jalur yang lain.
- Terkadang nilai kebutuhan energi terbesar untuk mengirimkan data ke Sink justru tidak pada Source Node, Node X membutukan cost energi 359598 nJ hingga sampai ke sink (Lampiran Percobaan ke-2).
- Pengiriman 100bit data dalam satu rute (dari sumber ke sink) membututuhkan energi rata-rata sebesar 287420,5 nJ.

5.2 SARAN

Mengingat masih banyaknya hal-hal yang belum dapat diimplementasikan pada proyek akhir ini, maka penulis memberikan beberapa saran kepada pembaca yang ingin mengembangkan proyek akhir ini dengan kualitas yang lebih baik.

1. Dapat mencari routing yang lebih efisien lagi dengan algoritma gradient routing.
2. Dapat meng-implementasikan lebih baik lagi parameter-parameter yang ada pada simulasi.

= = = Halaman ini sengaja di kosongkan = = =

DAFTAR PUSTAKA

[1] Jabed Faruque, Konstantinos Psounis, Ahmed Helmy, Analysis of Gradient-Based Routing Protocols in Sensor Networks* : In International Conference on Distributed Computing in Sensor System (DCOSS), Marina del Rey, CA, USA: IEEE/ACM, June 2005.

[2] Jabed Faruque, Ahmed Helmy, Gradient-Based Routing Protocols in Sensor Networks : Departement of Electrical Engineering, University of Shouthern California, Los Ageles, 2005.

[3] Joern Ploennigs, Volodymyr Vasyutynskyy, Mario Neugebauer, and Klaus Kabitzsch, Poster Abstract: Gradient-based Integral Sampling for WSNs in Building Automation, 2009

[4] Thomas Watteyne, Kris Pister, Dominique Barthel, Mischa Dohler, Isabelle Auge-Blum, Implementation of Gradient Routing in Wireless Sensor Networks, In the direction of IEEE Communications Society subject matter experts for publication, 2009

[5] Qi Yang, Yuxiang Zhuang, Hui Li, An Multi-hop Cluster Based Routing Protocol for Wireless Sensor Networks. Journal of Convergence Information Technology, Volume 6, Number 3. March 2011.

[6] Sami J. Habib, Modeling and simulating coverage in sensor networks, Kuwait University, Engineering Department, 2006.

= = = = Halaman ini sengaja dikosongkan = = = =

LAMPIRAN

Spesifikasi Zigbee

ZigBee adalah spesifikasi untuk protokol komunikasi tingkat tinggi yang menggunakan radio digital yang kecil dan berdaya kecil. ZigBee mengacu pada stadard IEEE 802.15.4 (2003) yang berhubungan dengan wireless personal area networks (WPANs). Contoh WPANs antara lain wireless headphones yang terhubung dengan telepon genggam melalui radio jarak dekat.

Teknologi yang memenuhi spesifikasi ZigBee dimaksudkan untuk membuat lebih simpel dan tidak lebih mahal dari WPANs lain, seperti Bluetooth. ZigBee difokuskan pada penggunaan radio frequency yang membutuhkan kecepatan transfer rendah, hemat daya, dan jaringan yang aman.

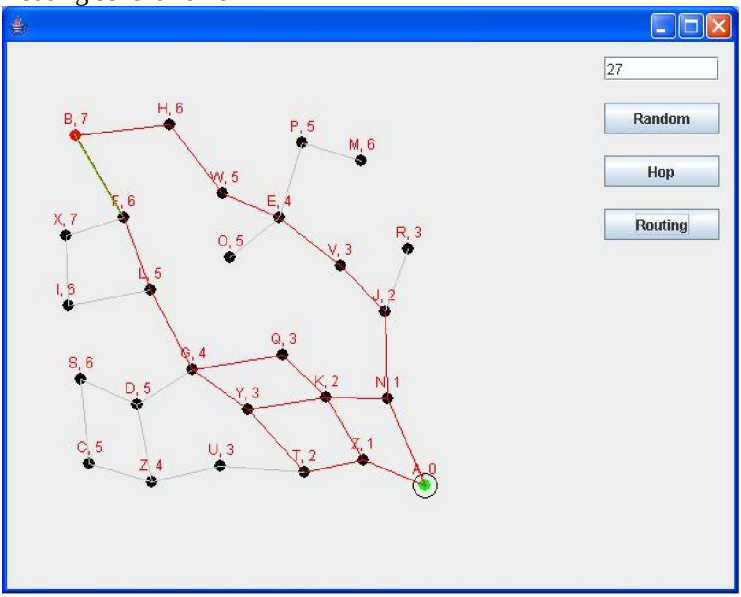
Market Name Standard	ZigBee™ 802.15.4	GPRS/GSM 1xRTT/CDMA	Wi-Fi™ 802.11b	Bluetooth™ 802.15.1
Application Focus	Monitoring & Control	Wide Area Voice & Data	Web, Email, Video	Cable Replacement
System Resources	4KB - 32KB	16MB+	1MB+	250KB+
Battery Life (days)	100 - 1,000+	1-7	.5 - 5	1 - 7
Network Size	Unlimited (2^{64})	1	32	7
Bandwidth (KB/s)	20 - 250	64 - 128+	11,000+	720
Transmission Range (meters)	1 - 100+	1,000+	1 - 100	1 - 10+
Success Metrics	Reliability, Power, Cost	Reach, Quality	Speed, Flexibility	Cost, Convenience

Sumber : http://nico89s.wordpress.com/2010/02/26/zigbee/

Percobaan ke-1

Inisial	Node Ke-	x	y	Hop	E ke Sink (nJ)
A	1	330	350	0	0
B	2	40	60	7	341771
C	3	243	152	4	93880
D	4	228	97	5	149780
E	5	62	187	6	288692
F	6	111	325	4	136530
G	7	121	225	5	193025
H	8	146	116	5	180105
I	9	275	261	2	25874
J	10	34	292	6	301369
K	11	315	59	7	327271
L	12	225	202	3	54815
M	13	321	162	5	165964
N	14	222	263	3	67628
O	15	58	119	7	354376
P	16	183	324	3	81056
Q	17	181	170	4	106840
R	18	293	316	1	12525
S	19	73	258	5	208958
T	20	317	111	6	248139
U	21	100	90	6	260931
V	22	155	267	4	121345
W	23	260	57	6	235522
X	24	30	226	6	314242
Y	25	241	332	2	38834
Z	26	102	146	6	273767
a	27	53	349	5	222898

Routing sekenario ke-2

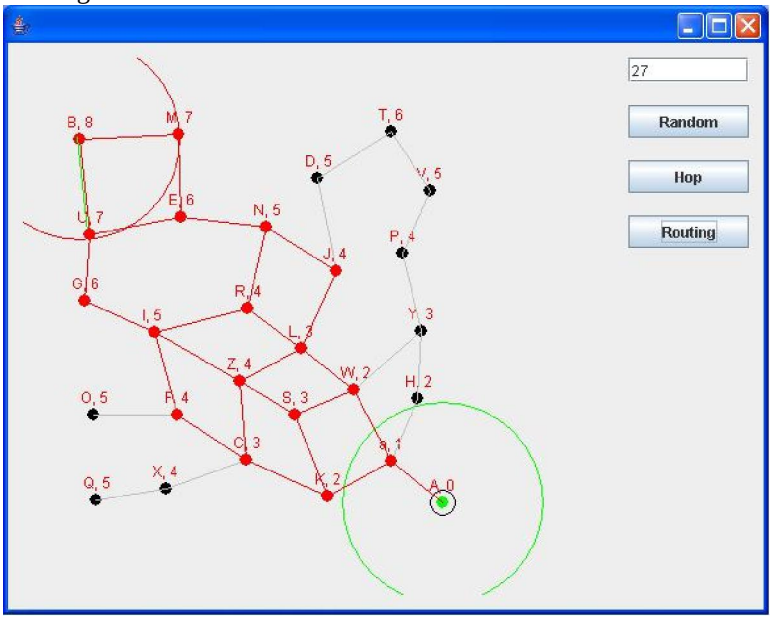


Percobaan ke-2

Inisial	Node Ke-	x	y	Hop	E ke Sink (nJ)
A	1	330	350	0	0
B	2	40	60	7	347069
C	3	51	332	5	261566
D	4	91	283	5	233056
E	5	209	128	4	151328
F	6	80	128	6	316111
G	7	137	254	4	167097
H	8	118	51	6	289377
I	9	34	201	6	330904
J	10	297	206	2	44375
K	11	248	277	2	56977
L	12	102	188	5	248637
M	13	277	81	6	274192

N	14	299	278	1	16145
O	15	168	161	5	193285
P	16	228	66	5	207490
Q	17	212	242	3	107877
R	18	316	154	3	82543
S	19	44	262	6	302027
T	20	230	339	2	69478
U	21	160	334	3	137127
V	22	260	168	3	95356
W	23	162	108	5	220099
X	24	32	143	7	359598
Y	25	183	287	3	122202
Z	26	279	329	1	29187
a	27	103	347	4	180515

Routing Sekenario ke-3



Percobaan ke-3

Inisial	Node Ke-	x	y	Hop	E ke Sink (nJ)
A	1	330	350	0	0
B	2	40	60	8	363576
C	3	173	316	3	81236
D	4	230	91	5	264571
E	5	121	122	6	320551
F	6	118	280	4	137380
G	7	44	189	6	305863
H	8	310	267	2	25711
I	9	100	214	5	221164
J	10	245	165	4	180627
K	11	238	345	2	39096
L	12	217	227	3	108990
M	13	119	56	7	347736
N	14	189	130	5	278932
O	15	51	280	5	235653
P	16	298	151	4	123059
Q	17	53	348	5	248870
R	18	174	195	4	193500
S	19	212	280	3	96137
T	20	289	54	6	292102
U	21	48	136	7	333376
V	22	320	101	5	206484
W	23	259	260	2	53245
X	24	109	339	4	152005
Y	25	313	213	3	66170
Z	26	168	253	4	165999
a	27	289	317	1	12770

Percobaan ke-4

Inisial	Node Ke-	x	y	Hop	E ke Sink (nJ)
A	1	330	350	0	0
B	2	40	60	7	366456
C	3	190	138	5	224101
D	4	262	104	6	268547
E	5	115	91	6	283597
F	6	132	160	5	195539
G	7	185	320	3	39489
H	8	79	253	5	141256
I	9	82	196	5	211192
J	10	155	214	4	111170
K	11	94	311	5	154780
L	12	80	128	6	297325
M	13	302	173	5	169065
N	14	243	304	2	25869
O	15	298	264	3	54114
P	16	47	292	6	236646
Q	17	175	81	6	323929
R	18	200	191	4	126355
S	19	248	210	4	95617
T	20	144	279	4	80201
U	21	212	262	3	66839
V	22	31	173	6	310455
W	23	321	95	7	337491
X	24	250	155	5	182094
Y	25	237	58	7	350232
Z	26	278	348	1	12708
a	27	30	349	6	252186

Percobaan ke-5

Inisial	Node Ke-	x	y	Hop	E ke Sink (nJ)
A	1	330	350	0	0
B	2	40	60	8	360885
C	3	112	56	7	319412
D	4	271	90	5	166009
E	5	319	273	1	16050
F	6	43	302	6	276442
G	7	318	215	2	45312
H	8	187	253	3	100231
I	9	256	153	4	113291
J	10	202	116	5	180294
K	11	310	165	3	70806
L	12	137	250	4	151815
M	13	240	209	3	86926
N	14	111	120	6	245617
O	15	83	186	6	261397
P	16	319	112	4	126181
Q	17	254	339	1	31947
R	18	114	304	5	220289
S	19	231	290	2	58242
T	20	182	68	6	232993
U	21	40	218	7	345685
V	22	185	199	4	139306
W	23	53	114	7	332812
X	24	167	338	6	290407
Y	25	69	348	6	304368
Z	26	151	152	5	206844
a	27	312	56	5	193479

Percobaan ke-6

Inisial	Node Ke-	x	y	Hop	E ke Sink (nJ)
A	1	330	350	0	0
B	2	40	60	8	364154
C	3	111	137	6	231764
D	4	312	234	2	27294
E	5	253	52	5	178362
F	6	123	285	5	203823
G	7	120	347	6	260667
H	8	229	150	4	109238
I	9	310	301	1	12801
J	10	173	90	6	246814
K	11	255	317	2	40575
L	12	258	197	3	68021
M	13	39	259	7	348614
N	14	207	237	3	96188
O	15	176	161	5	149885
P	16	266	266	2	53736
Q	17	107	211	5	216963
R	18	253	106	5	162397
S	19	173	327	6	274931
T	20	114	68	7	319672
U	21	313	172	3	81866
V	22	63	310	6	289156
W	23	37	120	7	335437
X	24	42	172	6	304902
Y	25	311	103	4	124003
Z	26	153	243	4	136955
a	27	325	52	5	191159

Listing Program
Gradient.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package hop;

import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;
import javax.swing.JPanel;

/**
 *
 * @author kunpraga Maulana
 */
public class Gradient extends JPanel{

    public int jumlahNode;
    public int bit=100;
    public int[] x = new int[60];
    public int[] y = new int[60];
    public double[] energi = new double [60];
    public double energitot=0;
    public double[] cost=new double [60];
    public double costtot=0;
    public int varhop [] = new int[50];
    private int mode;
    public static final int CLEAR = 0;
    public static final int RANDOM = 1;
    public static final int HOP = 2;
    public static final int ROUTE = 3;

    public Gradient(){
        pilihMode(CLEAR);
    }
```

```
public void jumlahNode(String sJumlahNode){
   jumlahNode = Integer.parseInt(sJumlahNode);
}

public void pilihMode(int type){
   mode = type;
   paint(getGraphics());
   //repaint();
}

/*public void reset (){
   while(){}
   jumlahNode().remove(0);
   repaint();
}*/


//---------------------------------------------------------------------------
@Override
public void paint (Graphics g){
if(mode == RANDOM){
   Random generator = new Random();
   double[] jarak = new double[jumlahNode];
   int k = 0;
   for (int i=0; i<jumlahNode; i++){
      int a = generator.nextInt(300)+30;
      int b = generator.nextInt(300)+50;
      x[i]=a;
      y[i]=b;
      energi[i]=2000;// Dalam satuan Mili joule
      if (i==0){//kond 1, go
         x[i]=330;
         y[i]=350;
         g.setColor(Color.GREEN);
         g.fillOval(x[i], y[i], 10, 10);//penggambaran node
         g.drawOval(x[i]-75, y[i]-75, 160, 160);
         g.setColor(Color.BLACK);
```

```
            g.drawOval(x[i]-5, y[i]-5, 20, 20);
         }else{//kond1, stop; kond2, go
            if(i==1){
               x[i]=40;
               y[i]=60;
               g.setColor(Color.red);
               g.fillOval(x[i], y[i], 10, 10);
               g.drawOval(x[i]-75, y[i]-75, 160, 160);
               g.setColor(Color.BLACK);
            }else{
            for (int j=0; j<i; j++){
               k=0;
               jarak[j]=Math.sqrt(Math.pow(x[i]-x[j],2)+Math.pow(y[i]-
y[j],2));

               if(jarak[j]<=50){
                  i--;
                  k=1;
                  break;
               }
            }
            if(k==1){
               continue;
            }
            /*try { //progran dellay
               Thread.sleep(0);
            }catch (InterruptedException ie) {
               ie.printStackTrace();
            }//end of dellay program*/
            g.fillOval(a, b, 10, 10);//penggambaran node
            //g.drawOval(x[i]-75, y[i]-75, 160, 160);
         }
         }//kond2, stop
      }
      System.out.println("Penyebaran Node Selesai");
      System.out.println();
      System.out.println("x");
      for(int i=0;i<jumlahNode;i++){
```

```
            System.out.println(x[i]);
        }
        System.out.println("y");
        for(int i=0;i<jumlahNode;i++){
            System.out.println(y[i]);
        }
    }
    //---------------------------------------------------------------------------

    //---------------------------------------------------------------------------
    if(mode == HOP){
        double ET=bit*50;//Dalam Nano Joule
        double EK;
        int simpanJalur=1;
        int jalur=0;
        int loncatan=1;
        int jumlahSudah=0;
        int [] indexBerikut = new int[60];
        int [] simpanIndex = new int [60];
        int [] sudah = new int [jumlahNode];
        double jarak=9999;
        int ici;
        char ci;
        while (Boolean.TRUE){
            //System.out.println("Masuk While");
            for(int i=0;i<simpanJalur;i++){
                for(int j=1;j<jumlahNode;j++){
                    //System.out.println("antara "+j+" dan "+simpanIndex[i]);
                    jarak=Math.sqrt(Math.pow(x[j]-
x[simpanIndex[i]],2)+Math.pow(y[j]-y[simpanIndex[i]],2));
                    //System.out.println("Jaraknya adalah "+jarak);
                    if(jarak<80 && sudah[j]==0){
                        //System.out.println("Masuk pada IF dengan jarak
"+jarak);
                        //g.drawLine(x[j], y[j], x[simpanIndex[i]],
y[simpanIndex[i]]);
                        EK=bit*50+bit*0.01*Math.pow(jarak, 2);
                        costtot+=(ET+EK);
```

```
                cost[j]=costtot;
                varhop[j]=loncatan;
                sudah[j]=1;
                indexBerikut[jalur]=j;
                jalur++;
            }
        }
    }
    for(int i=1;i<jumlahNode;i++){
        jumlahSudah+=sudah[i];
    }
    if(jumlahSudah==jumlahNode-1)break;
    jumlahSudah=0;
    simpanJalur=jalur;
    for(int i=0;i<simpanJalur;i++){
        simpanIndex[i]=indexBerikut[i];
        indexBerikut[i]=0;
    }
    jalur=0;
    loncatan++;
    //System.out.println("Loncatan = "+loncatan);
}
//for(int i=0;i<jumlahNode;i++){
    //System.out.println("Hop sensor ke-"+i+" adalah "+varhop[i]);
//}
ici=0;
for(int i=0;i<jumlahNode;i++){
    if(i<=25){
        ici=65+i;
    }
    else{
        ici=71+i;
    }
    ci=(char)ici;
    g.setColor(Color.red);
    g.drawString(""+ci+", "+varhop[i], x[i]-5, y[i]-4);
}
System.out.println("Hop");
```

```
        for (int i=0;i<jumlahNode;i++){
            System.out.println(varhop[i]);
        }
        System.out.println("Energi yang dibutuhkan hingga ke Sink");
        for (int i=0;i<jumlahNode;i++){
            System.out.println(cost[i]);
        }
    }
    //---------------------------------------------------------------------------

    //---------------------------------------------------------------------------
    if(mode == ROUTE){
        int []sudah = new int[jumlahNode];
        int []sudah1 = new int[jumlahNode];
        double[] jarak = new double[jumlahNode];
        double[] jarakSink = new double[jumlahNode];
        double ET=bit*50;//Dalam Nano Joule
        double EK;
        double waktuUdara=0;
        double waktuProses=4*Math.pow(10, -6)*bit;//4 micro secon/bit X
Jmlah bit = mircro secon
        double waktuTotal=0;
        int[] HopBerikut = new int [20];
        int[] simpanHopBerikut = new int [20];
        int jalur=0;
        int simpanJalur=0;
        int selesai=0;
        int indexSementara=0;
        for (int i=0;i<jumlahNode;i++){
            if(i!=0){
                for(int j=0;j<jumlahNode;j++){
                    jarak[j]=Math.sqrt(Math.pow(x[i]-x[j],2)+Math.pow(y[i]-
y[j],2));
                    if(jarak[j] < 80 && varhop[j]==varhop[i]-1){
                        /*
                        try { //progran dellay
                            Thread.sleep(200);
                        }catch (InterruptedException ie) {
```

```
                    ie.printStackTrace();
                  }//end of dellay program
                 */
                  //energi[i]-=150;
                  //energi[j]-=150;
                  g.setColor(Color.LIGHT_GRAY);
                  g.drawLine(x[i]+5, y[i]+5, x[j]+5, y[j]+5);
                  sudah[i]=1;
               }
            }
            if(sudah[i]!=1){
               for(int j=0;j<jumlahNode;j++){
                  jarak[j]=Math.sqrt(Math.pow(x[i]-
x[j],2)+Math.pow(y[i]-y[j],2));
                  if(jarak[j] < 80 && varhop[j]==varhop[i]){
                    /*
                     try { //progran dellay
                        Thread.sleep(10);
                     }catch (InterruptedException ie) {
                        ie.printStackTrace();
                     }//end of dellay program
                    */
                     g.setColor(Color.LIGHT_GRAY);
               //      energi[i]-=150;
               //      energi[j]-=150;
                     g.drawLine(x[i]+5, y[i]+5, x[j]+5, y[j]+5);
                     sudah[i]=1;
                  }
               }
            }
         }
      }
      //for (int i=0; i<jumlahNode; i++){
         //System.out.println("energi pada node ke-"+i+" adalah =
"+energi[i]);
      //}
//Mulai Program Perutean Datanya
      for(int i=1; i<jumlahNode;i++){
```

```
        jarakSink[i]=Math.sqrt(Math.pow(x[i]-x[0],2)+Math.pow(y[i]-
y[0],2));
      }// for ini buat ngitung semua jarak node terhadap sink
      for (int i=2; i<jumlahNode;i++){
        jarak[i]=Math.sqrt(Math.pow(x[i]-x[1],2)+Math.pow(y[i]-
y[1],2));
        if(jarak[i] < 80 && varhop[i] == varhop[1]-1){
          jalur++;
          HopBerikut[jalur-1]=i;
          g.setColor(Color.red);
          /*
          try { //progran dellay
            Thread.sleep(200);
          }catch (InterruptedException ie) {
            ie.printStackTrace();
          }
          */
          EK=bit*50+bit*0.01*Math.pow(jarak[i], 2);
          // Dalam satuan Nano Joule
          energi[1]-=EK;energitot+=EK;
          energi[i]-=ET;energitot+=ET;
          g.drawLine(x[1]+5, y[1]+5, x[i]+5, y[i]+5);
          g.fillOval(x[i], y[i], 10, 10);
          g.setColor(Color.BLACK);
          //System.out.println("Sudah jalan Hop-1");
          sudah1[1]=1;
        }
      }
      //Program tambahan coba2 mulai (Berhasil!!)
      if (sudah1[1]==1){
        double jarakSementara=9999;
        for(int i=0;i<jalur;i++){
          if(jarakSink[HopBerikut[i]]<jarakSementara){
            jarakSementara=jarakSink[HopBerikut[i]];
            indexSementara=HopBerikut[i];
          }
        }
        //g.setColor(Color.GREEN);
```

```
        //g.drawLine(x[1]+3,          y[1]+3,          x[indexSementara]+3,
y[indexSementara]+3);
        g.fillOval(x[indexSementara], y[indexSementara], 10, 10);
      }
      //System.out.println("sudah [1] = "+sudah1[1]);
      if(sudah1[1]!=1){
        for (int i=2; i<jumlahNode;i++){
          jarak[i]=Math.sqrt(Math.pow(x[i]-x[1],2)+Math.pow(y[i]-
y[1],2));
          if(jarak[i] < 80 && varhop[i] == varhop[1]){
            jalur++;
            HopBerikut[jalur-1]=i;
            g.setColor(Color.red);
           /*
            try { //progran dellay
              Thread.sleep(200);
            }catch (InterruptedException ie) {
              ie.printStackTrace();
            }
           */
            EK=bit*50+bit*0.01*Math.pow(jarak[i], 2);
            // Dalam satuan Nano Joule
            energi[1]-=EK;energitot+=EK;
            energi[i]-=ET;energitot+=ET;
            g.drawLine(x[1]+5, y[1]+5, x[i]+5, y[i]+5);
            g.fillOval(x[i], y[i], 10, 10);
            g.setColor(Color.BLACK);
            sudah1[1]=1;
          }
        }

        double jarakSementara=9999;
        //pemilihan 1 jalur terjauh yang dapat di jangkau menjuju sink
        for(int i=0;i<jalur;i++){
          if(jarakSink[HopBerikut[i]]<jarakSementara){
            jarakSementara=jarakSink[HopBerikut[i]];
            indexSementara=HopBerikut[i];
          }
```

```
        }
        //g.setColor(Color.GREEN);
        //g.drawLine(x[1]+3, y[1]+3, x[indexSementara]+3,
y[indexSementara]+3);
        g.fillOval(x[indexSementara], y[indexSementara], 10, 10);

      }
      simpanJalur=jalur;
      //System.out.println("simpanJalur = "+simpanJalur);
      for(int i=0;i<jalur;i++){
        simpanHopBerikut[i]=HopBerikut[i];
        //System.out.println("simpanHopBerikut, pada index ke-"+i+" =
"+simpanHopBerikut[i]+", HopBerikut, pada index ke-"+i+" =
"+HopBerikut[i]);
        HopBerikut[i]=0;
      }
      jalur=0;//set jalur menjadi 0 kembali sebelum digunakan
selanjutnya
      while(selesai==0){
        for(int i=0;i<simpanJalur;i++){
          //System.out.println("hop kirim = Hop
"+simpanHopBerikut[i]);
          for (int j=0; j<jumlahNode;j++){
            if(j==1)continue;
            if(sudah1[j]==1)continue;
            jarak[j]=Math.sqrt(Math.pow(x[j]-
x[simpanHopBerikut[i]],2)+Math.pow(y[j]-y[simpanHopBerikut[i]],2));
            if(jarak[j] < 80 && varhop[j] ==
varhop[simpanHopBerikut[i]]-1){
              jalur++;
              HopBerikut[jalur-1]=j;
              g.setColor(Color.red);
              /*
              try { //progran dellay
                Thread.sleep(200);
              }catch (InterruptedException ie) {
                ie.printStackTrace();
              }
```

```
                */
                EK=bit*50+bit*0.01*Math.pow(jarak[j], 2);
                // Dalam satuan Nano Joule
                energi[i]-=EK;energitot+=EK;
                energi[j]-=ET;energitot+=ET;
                g.drawLine(x[simpanHopBerikut[i]]+5,
y[simpanHopBerikut[i]]+5, x[j]+5, y[j]+5);
                g.fillOval(x[simpanHopBerikut[i]],
y[simpanHopBerikut[i]], 10, 10);
                if(j==0) selesai=1;
                g.setColor(Color.BLACK);
                if(jarak[0] < 80 && varhop[0] ==
varhop[simpanHopBerikut[i]]-1)selesai=1;
                sudah1[simpanHopBerikut[i]]=1;
                //sudah1[j]=1;
              }
            }
            if(sudah1[simpanHopBerikut[i]]!=1){
              for (int j=2; j<jumlahNode;j++){
                if(sudah1[j]==1)continue;
                jarak[j]=Math.sqrt(Math.pow(x[j]-
x[simpanHopBerikut[i]],2)+Math.pow(y[j]-y[simpanHopBerikut[i]],2));
                if(jarak[j] < 80 && varhop[j] ==
varhop[simpanHopBerikut[i]] &&
jarakSink[j]<jarakSink[simpanHopBerikut[i]]){
                  jalur++;
                  HopBerikut[jalur-1]=j;
                  g.setColor(Color.red);
                  /*
                  try { //progran dellay
                    Thread.sleep(200);
                  }catch (InterruptedException ie) {
                    ie.printStackTrace();
                  }
                  */
                  EK=bit*50+bit*0.01*Math.pow(jarak[j], 2);
                  // Dalam satuan Nano Joule
                  energi[simpanHopBerikut[i]]-=EK;energitot+=EK;
```

```
                    energi[j]-=ET;energitot+=ET;
                    g.drawLine(x[simpanHopBerikut[i]]+5,
y[simpanHopBerikut[i]]+5, x[j]+5, y[j]+5);
                    g.fillOval(x[simpanHopBerikut[i]],
y[simpanHopBerikut[i]], 10, 10);
                    g.setColor(Color.BLACK);
                    sudah1[simpanHopBerikut[i]]=1;
                    //sudah1[j]=1;
                  }
                }
              }
              //for (int j=0;j<jalur;j++)
                 //System.out.println("hop yang tersimpan = Hop
"+HopBerikut[j]+"; Terhubung pada Hop "+simpanHopBerikut[i]);
          }
          //System.out.println();
          //System.out.println("Jumlah jalur adalah "+jalur);
          simpanJalur=jalur;
          for(int i=0;i<jalur;i++){
             simpanHopBerikut[i]=HopBerikut[i];
             HopBerikut[i]=0;
          }
          jalur=0;
       }
       System.out.println("Energi Total yang di butuhkan adalah
"+energitot+" nJ");
       System.out.println("selesai = "+selesai);
//Ahir Program perutean datanya

//Awal Program perutean jalur tunggal
       /*
       int kondisi=1;
       int sudah2=0;
       int simpanIndex;
       while(kondisi==1){
          simpanIndex=indexSementara;
          sudah2=0;
          for(int i=0;i<jumlahNode;i++){
```

```
            if(i==1) continue;
            if(i==simpanIndex)continue;
            jarak[i]=Math.sqrt(Math.pow(x[i]-
x[simpanIndex],2)+Math.pow(y[i]-y[simpanIndex],2));
            if(jarak[i] < 80 && varhop[i] == varhop[simpanIndex]-1){
               jalur++;
               HopBerikut[jalur-1]=i;
               sudah2=1;
            }
         }
         if(sudah2!=1){
            for(int i=0;i<jumlahNode;i++){
               if(i==1) continue;
               if(i==simpanIndex)continue;
               jarak[i]=Math.sqrt(Math.pow(x[i]-
x[simpanIndex],2)+Math.pow(y[i]-y[simpanIndex],2));
               if(jarak[i] < 80 && varhop[i] == varhop[simpanIndex] &&
jarakSink[i]<jarakSink[simpanIndex]){
                  jalur++;
                  HopBerikut[jalur-1]=i;
                  sudah2=1;
               }
            }
         }
         double jarakSementara=9999;
         for(int i=0;i<jalur;i++){
            if(jarakSink[HopBerikut[i]]<jarakSementara){
               jarakSementara=jarakSink[HopBerikut[i]];
               indexSementara=HopBerikut[i];
            }
         }
         waktuUdara=(jarakSementara/8)/(3*Math.pow(10, 8));
         //System.out.println("waktu proses = "+waktuProses+", waktu
udara = "+waktuUdara);
         waktuTotal+=(waktuProses+waktuUdara);//dalam satuan detik
         //System.out.println("Waktu Proses adalah "+waktuProses);
         //System.out.println("Waktu Diudara adalah "+waktuUdara);
```

```
        //System.out.println("dari index-"+simpanIndex+" ke index-
"+indexSementara);
        g.setColor(Color.GREEN);
        g.drawLine(x[indexSementara]+3, y[indexSementara]+3,
x[simpanIndex]+3, y[simpanIndex]+3);
        if(indexSementara==0){
           kondisi=0;
           System.out.println("Kondisi = "+kondisi);
        }
      }
      System.out.println("Waktu Total yang dibutuhkan =
"+waktuTotal*Math.pow(10, 3)+"ms");//dalam satuan ms
      System.out.println("Program berhasil keluar dari Wile");
      */
      //Akhir perutean dari jalur tunggal
      //System.out.println("Jarak S ke D adalah "+
Math.sqrt(Math.pow(x[1]-x[0],2)+Math.pow(y[1]-y[0],2))
           //+"dalam Pixel. Sedangkan dalam meter adalah "+
Math.sqrt(Math.pow(x[1]-x[0],2)+Math.pow(y[1]-y[0],2))/8);
   }
   //---------------------------------------------------------------------------

   }
}
```

DAFTAR RIWAYAT HIDUP

Nama : Kunpraga Maulana Arrossy
TTL : Jakarta, 20 Agustus 1990
Alamat : Jl. Kedung Tarukan Baru I No. 34 Surabaya 60285
Telp : 083830073430
Hobi : Bulutangkis dan main game
E-mail : kunpraga@gmail.com
Motto : Jangan lari, tapi hadapi, dan tanggung jawab.

Riwayat pendidikan formal yang pernah ditempuh:
1995 – 2001 : SD Ahmad Yani IV Purwakarta
2001 – 2004 : SMP Negeri 4 Surabaya
2004 – 2007 : SMA Negeri 7 Surabaya
2007 – 2011 : Politeknik Elektronika Negeri Surabaya (PENS)
Jurusan Teknik Telekomunikasi

Penulis telah mengikuti Seminar Proyek Akhir pada tanggal 20 Juli 2011, sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Sains Terapan (SST).