# **OPTIMASI DEPLOYMENT WSN MENGGUNAKAN ALGORITMA PSO UNTUK PENDETEKSI KEBAKARAN HUTAN**

**SKRIPSI**

****

**I MADE SUASTIKA**

**NIM. 1708561001**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**JIMBARAN**

**2021**

# **LEMBAR PENGESAHAN TUGAS AKHIR**

|  |  |
| --- | --- |
| Judul : | Optimasi Dployment WSN Menggunakan Algoritma PSO untuk Pendeteksi Kebakaran Hutan. |
| Nama : | I Made Suastika |
| NIM : | 1708561001 |
| Tanggal Seminar : | 8 September 2021 |

Disetujui oleh :

Pembimbing I

I Gusti Agung Gede Arya Kadyanan, S.Kom, M.Kom.

NIP. 198501302015041003

Penguji III

Agus Muliantara,S.Kom.,M.Kom.

NIP. 198006162005011001

Pembimbing II

Dr. Ngurah Agus Sanjaya ER, S.Kom., M.Kom.

NIP. 197803212005011001

Penguji II

I Komang Ari Mogi, S.Kom.,M.Kom.

NIP. 198409242008011007

Penguji I

Made Agung Raharja, S.Si.,M.Cs.

NIP. 1985091920181113001

Mengetahui,

Program Studi Informatika

FMIPA

KPS

Dr. Ir. I Ketut Gede Suhartana, S.Kom., M.Kom

NIP. 197201102008121001

# 

|  |  |
| --- | --- |
| Judul : | Optimasi Dployment WSN Menggunakan Algoritma PSO untuk Pendeteksi Kebakaran Hutan. |
| Nama : | I Made Suastika |
| NIM : | 1708561001 |

Pembimbing : 1. I GA Gede Arya Kadyanan, S.Kom, M.Kom

2. Dr. Ngurah Agus Sanjaya ER, S.Kom., M.Kom

# **ABSTRAK**

Kebakaran hutan merupakan bencana yang kerap terjadi beberpa tahun terakhir. Hal ini memberikan dampak yang sangat besar baik itu bagi lingkungan maupun masyarakat itu sendiri. Penanganan kebakaran yang terlambat menjadi salah satu faktor pemicu besarnya kerugian yang ditimbulkan dari bencana tersebut. Penggunaan Wireless Sensor Network merupakan salah satu solusi agar informasi terkait kebakaran tersampaikan kepada pihak yang berwajib dengan cepat sehingga penanganannya dapat dilakukan dengan lebih cepat. Pada penelitian ini dibuat suatu simulasi untuk mengetahui posisi optimal suatu node agar dapat mendeteksi kebakaran secara optimal. Simulasi ini dijalankan di *Software* NS3 pada Sistem Operasi Linux Ubuntu 18.04. Pada proses optimasi dilakukan dengan algoritma PSO yang dijalankan dengan Google Colab. Hasil setiap Iterasi pada PSO akan disimulasikan di NS3 dan dilihat komunikasi antar node nya. Terdapat 10 kali Iterasi dan terdapat 10 kali Simulasi sesuai dengan jumlah iterasinya. Dari 10 kali Simulasi yang telah dilakukan, diketahui bahwa pada iterasi terakhir dari 10 node yang dipasang semua node berkomunikasi. Komunikasi antar node dapat dilihat melalui file .pcap dan grafik pada NetAnim, komunikasi tersebut dicirikan dengan tersampaikannya pesan kebakaran kemasing-masing node yang di pasang. Pada iterasi terakhir telihat 10 node menerima pesan kebakaran.

Kata Kunci: *Wireless Sensor Network, Kebakaran Hutan, Particle swarm optimization*

|  |  |
| --- | --- |
| Title : | Optimasi Dployment WSN Menggunakan Algoritma PSO untuk Pendeteksi Kebakaran Hutan. |
| Name : | I Made Suastika |
| NIM : | 1708561001 |

Pembimbing : 1. I GA Gede Arya Kadyanan, S.Kom, M.Kom

2. Dr. Ngurah Agus Sanjaya ER, S.Kom., M.Kom

# **ABSTRACT**

*Forest fires are disasters that often occur in recent years. This has a huge impact on both the environment and society itself. Delayed handling of fires is one of the triggering factors for the large losses caused by the disaster. The use of a Wireless Sensor Network is one solution so that information related to fires is conveyed to the authorities quickly so that the handling can be done more quickly. In this study, a simulation was made to determine the optimal position of a node in order to detect fires optimally. This simulation is run on NS3 Software on Ubuntu 18.04 Linux Operating System. In the optimization process, the PSO algorithm is run with Google Colab. The results of each iteration on the PSO will be simulated in NS3 and the communication between nodes will be seen. There are 10 iterations and there are 10 simulations according to the number of iterations. From 10 simulations that have been carried out, it is known that in the last iteration of the 10 nodes installed, all nodes communicate. Communication between nodes can be seen through .pcap files and graphs on NetAnim, the communication is characterized by sending fire messages to each installed node. In the last iteration, 10 nodes received fire messages.*

**Keywords***: Wireless Sensor Network, Forest Fire, Particle swarm optimization*

# **KATA PENGANTAR**

Puji Syukur dipanjatkan ke hadirat Tuhan Yang Maha Esa karena atas berkat karunia-Nya penulis dapat menyelesaikan laporan tugas akhir dengan judul, “Optimasi *Dployment* WSN Menggunakan Algoritma PSO untuk Pendeteksi Kebakaran Hutan.” tepat pada waktunya.

Sehubungan dengan telah terselesaikan tugas akhir ini maka, diucapkan terima kasih dan penghargaan kepada berbagai pihak yang telah membantu pengusul, antara lain :

1. Bapak I Gusti Agung Gede Arya Kadyanan, S.Kom, M.Kom. selaku Pembimbing I yang telah banyak meluangkan tenaga, waktu, dan pikiran untuk memberikan bimbingan, pengetahuan, petunjuk, serta pengarahan kepada penulis dalam menyempurnakan tugas akhir ini;
2. Bapak Dr. Ngurah Agus Sanjaya ER, S.Kom., M.Kom. selaku Pembimbing II yang telah bersedia meluangkan waktunya untuk mengkoreksi, dan membantu menyempurnakan tugas akhir ini;
3. Bapak I Komang Ari Mogi, S.Kom., M.Kom. sebagai pembimbing di penjaluran Jaringan Sensor Nirkabel yang telah banyak membantu, memberikan petunjuk, bimbingan, serta pengarahan untuk menyempurnakan proposal ini.
4. Bapak Dr. I Ketut Suhartana, S.Kom., M.Kom, selaku Koordinator Program Studi Informatika Fakultas MIPA Universitas Udayana;
5. Bapak-bapak dan ibu-ibu dosen di Program Studi Informatika, yang telah meluangkan waktu turut memberikan saran dan masukan dalam penyempurnaan tugas akhir ini;
6. Bapak dan Ibu Staff Tata Usaha di Program Studi Informatika Fakultas MIPA Universitas Udayana yang telah membantu penulis dalam perkuliahan dan penyelesaian skripsi ini;
7. Kawan-kawan di Program Studi Informatika yang telah memberikan dukungan moral dalam penyelesaian proposal ini.

Disadari pula bahwa sudah tentu tugas akhir ini masih mengandung kelemahan dan kekurangan. Memperhatikan hal ini, maka adanya masukan dan saran-saran penyempurnaan sangat diharapkan.

Bukit Jimbaran, September 2021

Penyusun

I Made Suastika

# **DAFTAR ISI**

[**OPTIMASI DEPLOYMENT WSN MENGGUNAKAN ALGORITMA PSO UNTUK PENDETEKSI KEBAKARAN HUTAN** i](#_Toc81815648)

[**LEMBAR PENGESAHAN TUGAS AKHIR** ii](#_Toc81815649)

[**ABSTRAK** iii](#_Toc81815650)

[**ABSTRACT** iv](#_Toc81815651)

[**KATA PENGANTAR** v](#_Toc81815652)

[**DAFTAR ISI** vii](#_Toc81815653)

[**DAFTAR GAMBAR** ix](#_Toc81815654)

[**DAFTAR TABEL** x](#_Toc81815655)

[**DAFTAR LAMPIRAN** xii](#_Toc81815656)

[**BAB I** 1](#_Toc81815657)

[1.1 Latar Belakang 1](#_Toc81815658)

[1.2 Rumusan Masalah 2](#_Toc81815659)

[1.3 Tujuan Penelitian 2](#_Toc81815660)

[1.4 Batasan Masalah 2](#_Toc81815661)

[1.5 Manfaat Penelitian 3](#_Toc81815662)

[1.6 Sistematika Penulisan 3](#_Toc81815663)

[**BAB II** 5](#_Toc81815664)

[2.1 Tinjauan Empiris 5](#_Toc81815665)

[2.1.1 Tinjauan Studi 5](#_Toc81815666)

[2.2 Tinjauan Teoritis 6](#_Toc81815667)

[2.2.1 Wireless Sensor Network 6](#_Toc81815668)

[2.2.2 Node Sensor 8](#_Toc81815669)

[2.2.3 Particle Swarm Optimization 9](#_Toc81815670)

[**BAB III** 13](#_Toc81815671)

[3.1 Data 13](#_Toc81815672)

[3.2 Desain Sistem Secara Keseluruhan 13](#_Toc81815673)

[3.2.1 Instalasi NS3 14](#_Toc81815674)

[3.2.2 Istalasi Net Anim 14](#_Toc81815675)

[3.3 Tahap Prosesing 15](#_Toc81815676)

[3.3.1 Optimasi 15](#_Toc81815677)

[3.3.2 Simulasi 16](#_Toc81815678)

[3.3.3 Spesifikasi Program 16](#_Toc81815679)

[**BAB IV** 18](#_Toc81815680)

[4.1 Implementasikan 18](#_Toc81815681)

[4.1.1 Script dan Penjelasan Program 18](#_Toc81815682)

[4.1.2 Tahap Menjalankan Simulasi 31](#_Toc81815683)

[4.2 Hasil dan Pengujian 31](#_Toc81815684)

[4.2.1 Perhitungan dengan PSO dan Simulasi Pada NS3 31](#_Toc81815685)

[4.2.2 Hasil Pcap 58](#_Toc81815686)

[4.2.3 Pengaruh Jumlah Iterasi terhadap Node yeng menerima pesan 59](#_Toc81815687)

[**BAB V** 61](#_Toc81815688)

[5.1 Kesimpulan 61](#_Toc81815689)

[5.2 Saran 61](#_Toc81815690)

[**DAFTAR PUSTAKA** 62](#_Toc81815691)

[**LAMPIRAN** 63](#_Toc81815692)

# **DAFTAR GAMBAR**

[Gambar 2.1 Wireless Sensor Networke 7](#_Toc81815146)

[Gambar 2.2 Node Sensor 8](#_Toc81815147)

[Gambar 3.1 Alur Penelitian 13](#_Toc81815148)

[Gambar 3.2 Alur Simulasi 14](#_Toc81815149)

[Gambar 3.3 Algoritma PSO 15](#_Toc81815150)

[Gambar 4.1 Hasil Runing NS3 31](#_Toc81815152)

[Gambar 4.2 Posisi Node Iterasi 0 33](#_Toc81815155)

[Gambar 4.3 Komunikasi Node Pada Iterasi 0 33](#_Toc81815156)

[Gambar 4.4 Posisi Node Iterasi 1 36](#_Toc81815160)

[Gambar 4.5 Komunikasi Node Iterasi 1 36](#_Toc81815161)

[Gambar 4.6 Posisi Node Iterasi 2 38](#_Toc81815165)

[Gambar 4.7 Komunikasi Node Iterasi2 38](#_Toc81815166)

[Gambar 4.8 Posisi Node Iterasi 3 40](#_Toc81815170)

[Gambar 4.9 Komunikasi Node Iterasi 3 41](#_Toc81815171)

[Gambar 4.10 Posisi Node Iterasi 4 43](#_Toc81815175)

[Gambar 4.11 Komunikasi Node Iterasi 4 43](#_Toc81815176)

[Gambar 4.12 Posisi Node Iterasi 5 45](#_Toc81815180)

[Gambar 4.13 Komunikasi Node Iterasi 5 46](#_Toc81815181)

[Table 4.19 Hasil Iterasi 6 46](#_Toc81815182)

[Gambar 4.14 Posisi Node Iterasi 6 48](#_Toc81815185)

[Gambar 4.15 Komunikasi Node Iterasi 6 48](#_Toc81815186)

[Gambar 4.16 Posisi Node Iterasi 7 50](#_Toc81815190)

[Gambar 4.17 Komunikasi Node Iterasi 7 50](#_Toc81815191)

[Gambar 4.18 Posisi Node Iterasi 8 52](#_Toc81815195)

[Gambar 4.19 Komunikasi Node Iterasi 8 53](#_Toc81815196)

[Gambar 4.20 Posisi Node Iterasi 9 55](#_Toc81815200)

[Gambar 4.21 Komunikasi Node Iterasi 9 55](#_Toc81815201)

[Gambar 4.22 Posisi Node Iterasi 10 57](#_Toc81815205)

[Gambar 4.23 Komunikasi Node Iterasi 10 57](#_Toc81815206)

[Gambar 4.24 Tampilan File .pcap Node yang tidak menerima Pesan Kebakaran 58](#_Toc81815207)

[Gambar 4.25 Tampilan File .pcap Node yang menerima Pesan Kebakaran 59](#_Toc81815208)

[Gambar 4.26 Grafik Node Yang Menerima Pesan Kebakaran 60](#_Toc81815210)

# **DAFTAR TABEL**

[Table 4.1 Hasil Iterasi PSO 21](#_Toc81815440)

[Table 4.2 Hasil Iterasi 0 32](#_Toc81815442)

[Table 4.3 Hasil Iterasi 1 Penyesuaian NS3 32](#_Toc81815443)

[Table 4.4 Hasil Iterasi 1 34](#_Toc81815446)

[Table 4.5 Hasil Iterasi 1 Yang di Normalkan 34](#_Toc81815447)

[Table 4.6 Hasil Iterasi 1 Penyesuaian NS3 35](#_Toc81815448)

[Table 4.7 Hasil Iterasi 2 36](#_Toc81815451)

[Table 4.8 Hasil Iterasi 2 yang Dinormalkan 37](#_Toc81815452)

[Table 4.9 Hasil Iterasi 2 Penyesuaian NS3 37](#_Toc81815453)

[Table 4.10 Hasil Iterasi 3 39](#_Toc81815456)

[Table 4.11 Hasil Iterasi 3 yang di Normalkan 39](#_Toc81815457)

[Table 4.12 Hasil Iterasi 3 Penyesuaian NS3 40](#_Toc81815458)

[Table 4.13 Hasil Iterasi 4 41](#_Toc81815461)

[Table 4.14 Hasil Iterasi 4 yang di Normalkan 42](#_Toc81815462)

[Table 4.15 Hasil Iterasi 4 Penyesuaian NS3 42](#_Toc81815463)

[Table 4.16 hasil Iterasi 5 43](#_Toc81815466)

[Table 4.17 Hasil Iterasi 5 yang di Normalkan 44](#_Toc81815467)

[Table 4.18 Hasil Iterasi 5 Penyesuaian NS3 44](#_Toc81815468)

[Table 4.19 Hasil Iterasi 6 46](#_Toc81815471)

[Table 4.20 Hasil Iterasi 6 yang di Normalkan 47](#_Toc81815472)

[Table 4.21 Hasil Iterasi 6 Penyesuaian NS3 47](#_Toc81815473)

[Table 4.22 Hasil Iterasi 7 48](#_Toc81815476)

[Table 4.23 Hasil Iterasi 7 yang di Normalkan 49](#_Toc81815477)

[Table 4.24 Hasil Iterasi 7 Penyesuaian NS3 49](#_Toc81815478)

[Table 4.25 Hasil Iterasi 8 51](#_Toc81815481)

[Table 4.26 Hasil Iterasi 8 yang di normalkan 51](#_Toc81815482)

[Table 4.27 Hasil Iterasi 8 Penyesuaian NS3 52](#_Toc81815483)

[Table 4.28 Hasil Iterasi 9 53](#_Toc81815486)

[Table 4.29 Hasil Iterasi 9 yang di Normalkan 54](#_Toc81815487)

[Table 4.30 Hasil Iterasi 9 Penyesuaian NS3 54](#_Toc81815488)

[Table 4.31 Hasil Iterasi 10 55](#_Toc81815491)

[Table 4.32 Hasil Iterasi 10 yang di Normalkan 56](#_Toc81815492)

[Table 4.33 Hasil Iterasi 10 Penyesuaian NS3 56](#_Toc81815493)

[Table 4.34 Node Yang Menerima Pesan Kebakaran 59](#_Toc81815498)

# **DAFTAR LAMPIRAN**

[A. Lampiran Code PSO 63](#_Toc81777697)

[B. Lampiran Code Simulasi 66](#_Toc81777698)

# **BAB I**

**PENDAHULUAN**

## **Latar Belakang**

Kebakaran hutan merupakan salah satu bencana alam yang belakangan ini sering terjadi di Indonesia. Menurut Kementerian Lingkungan Hidup dan Kehutanan (KLHK), berdasarkan citra satelit landsat sampai pada bulan September 2019 kebakaran hutan dan lahan mencapai 857.755 hektare. Untuk lahan mineral 630.451 hektare, dan lahan gambut 227.304 hektare*.* ([*https://www.mongabay.co.id/2019/11/16/kebakaran-hutan-dan-lahan-terus-*](https://www.mongabay.co.id/2019/11/16/kebakaran-hutan-dan-lahan-terus-terjadi-bagaimana-solusinya/)[*terjadi-bagaimana-solusinya/*](https://www.mongabay.co.id/2019/11/16/kebakaran-hutan-dan-lahan-terus-terjadi-bagaimana-solusinya/)*).* Selain menyebabkan kerusakan lingkungan, dampak dari kebakaran hutan adalah biaya yang tinggi dalam menanganinya serta proses pemulihan pasca kebakaran yang relatif panjang. Dampak kebakaran yang sangat dirasakan manusia ialah berupa kerugian ekonomis yaitu hilangnya manfaat dari potensi hutan seperti pohon hutan yang biasa digunakan untuk memenuhi kebutuhan akan bahan bangunan, bahan makanan, dan obat-obatan serta satwa untuk memenuhi kebutuhan akan protein hewani dan sebagai sarana rekreasi. Kerugian lainnya berupa kerugian ekologis yaitu berkurangnya luas wilayah hutan, tidak tersedianya udara bersih yang dihasilkan dari vegetasi hutan serta hilangnya fungsi hutan sebagai pengatur tata air dan pencegah terjadinya erosi. Besarnya kerugian akibat kebakaran hutan biasanya timbul akibat terlambatnya penanganan terhadap kasus tersebut, oleh karena itu sangat penting untuk membangun suatu sistem untuk mendeteksi kebakaran secara *real time*.

*Wireless Sensor Network* (WSN) atau Jaringan sensor nirkabel merupakan perangkat otonom yang terdistribusi khusus menggunakan sensor dalam memantau kondisi fisik atau lingkungan seperti suhu, suara, getaran, tekanan dan gerakan di lokasi yang berbeda. Belakangan ini WSN banyak digunakan untuk melakukan pemantauan terhadap suatu lokasi atau tempat yang sekiranya rawan terjadi suatu bencana. Dengan menggunakan perangkat WSN untuk memantau kondisi lingkungan hutan, maka ketika terdapat potensi kebakaran diharapkan perangkat mampu mendeteksinya dan memberikan pesan kepada pengawas hutan terdekat.

Namun dalam proses penempatan sensornya tentu harus memperhatikan jarak atar sensor agar komunikasi antar sensor dapat berjalan lebih efisien. Untuk menyiasati hal tersebut maka digunakanlah suatu algoritma optimasi yang disebut *particle swarm optimization* (PSO). PSO merupakan salah satu algoritma optimasi yang cukup populer yang terinspirasi dari prilaku kelompok ikan atau burung dalam mencari makan. Dalam penerapannya setelah sensor diletakkan sesuai dengan koordinat optimasi dari algoritma PSO diharapkan sensor akan mengumpulkan data secara *real time* sehingga apabila terjadi potensi kebakaran sistem akan memberi tanggapan berupa pesan sehingga pihak terkait dapat menangani kebakaran dengan cepat. Berdasarkan latar belakang tersebut maka dibuatlah rancangan sistem “Optimasi *Dployment* WSN Menggunakan Algoritma PSO untuk Pendeteksi Kebakaran Hutan”. Sistem tersebut akan dibuat dengan cara simulasi, hal tersebut dilakukan untuk meminimalisir hal-hal yang tidak diinginkan seperti kesalahan pada penempatan sensor ataupun kesalahan-kesalahan lainnya.

## **Rumusan Masalah**

Berdasarkan latar belakang di atas, maka dapat dirumuskan masalah sebagai berikut:

1. Bagaimana melakukan *deployment* node sensor yang efisien agar dapat mendeteksi pesan kebakaran secara optimal.
2. Begaimana mengujii ke optimalan *deployment* node sensor dalam mendeteksi pesan kebakaran.

## **Tujuan Penelitian**

Berdasarkan rumusan masalah tersebut, maka tujuan peneilitian ini adalah sebagai berikut:

* 1. Untuk mengetahui proses melakukan *deployment* node sensor yang efisien agar didapat hasil yang optimal.
  2. Untuk mengetahui tingkat ke optimalan *deployment* node sensor dalam mendeteksi pesan kebakaran.

## **Batasan Masalah**

Beberapa batasan masalah yang dijadikan acuan dalam penelitian ini adalah sebagai berikut:

1. Penelitian hanya fokus pada pembuatan simulasi *deployment* node sensor.
2. Penelitian dilakukan dengan cara simulasi menggunakan NS3.

## **Manfaat Penelitian**

Beberapa manfaat yang dapat diambil dari penelitian ini adalah sebagai berikut:

1. Diharapkan dengan adanya penelitian ini, hasil yang diperoleh dapat menjadi pedoman dalam mengembangkan penelitian selanjutnya mengenai *deployment* jaringan *sensor network*.
2. Hasil penelitian ini diharapkan menjadi pedoman *dployment node sensor* untuk EWS pada kebakaran hutan secara nyata.

## **Sistematika Penulisan**

Untuk mempermudah melihat dan mengetahui pembahasan yang ada pada penelitian ini secara menyeluruh, maka perlu dikemukakan sistematika yang merupakan kerangka dan pedoman dalam penulisan penelitian. Adapun sistematika penulisan pada penelitian ini adalah sebagai berikut:

Penyajian penelitian ini menggunakan sistematika penulisan sebagai berikut:

1. Bagian Awal Skripsi

Bagian awal memuat halaman sampul depan, halaman judul, lembar pengesahan tugas akhir, abstrak, kata pengantar, daftar isi, daftar gambar, daftar tabel, serta daftar lampiran.

1. Bagian Utama Skripsi

Bagian utama terbagi atas bab dan sub bab diantaranya yaitu:

BAB I PENDAHULUAN

Dalam bab ini menjelaskan tentang latar belakang dari topik penelitian yang diangkat, rumusan masalah, batasan-batasan masalah yang digunakan, tujuan dilakukannya penelitian, dan manfaat penelitian ini.

BAB II TINJAUAN PUSTAKA

Dalam bab ini menjelaskan tentang teori yang digunakan dalam penelitian ini seperti penjelasan *wireless sensor network*, pengertian *early warning system*, serta algoritma yang digunakan yaitu *particle swarm optimization* (PSO).

BAB III ANALISIS DAN PERANCANGAN SISTEM

Dalam bab ini menjelaskan tentang data yang digunakan, analisis kebutuhan sistem baik itu kebutuhan software maupun hardware, dan skenario simulasi yang akan dilakukan. Agar lebih sistematis bab analisis dan perancangan sistem terdiri dari:

1. Pengumpulan Data
2. Desain Penelitian
3. Desain Algoritma
4. Pengujian

BAB IV HASIL DAN PEMBAHASAN

Bab ini menjelaskan tentang gambaran umum dari sistem, implementasi dan hasil yang didapat dari simulasi, serta analisis dari hasil penelitian. Agar sistematis maka diklasifikasikan sebagai berikut:

1. Impelentasi
2. Hasil Pengujian

BAB V PENUTUP

Bab ini menjelaskan kesimpulan dan saran yang didapat setelah dilakukannya pengujian dan analisis terhadap sistem yang dibangun.

1. Bagian Akhir Skripsi

Bagian akhir dari penelitian ini berisikan daftar pustaka dan daftar lampiran.

# **BAB II**

**TINJAUAN PUSTAKA**

## **Tinjauan Empiris**

### **Tinjauan Studi**

1. **Deployment Jaringan Sensor Nirkabel Berdasarkan Cakupan Area Sensor Node Menggunakan Algoritma *Particle Swarm Optimization*** (*Hani Rubiani, 2016* )

Penelitian ini membahas mengenai perancangan suatu aplikasi untuk menempatkan sensor node pada lingkungan secara nyata sehingga kondisi optimal dari penempatan sensor node di suatu gedung tercapai. Adapun metode deployment yang digunakan dalam penelitian ini yaitu dengan menggunakan algoritma Particle Swarm Optimization (PSO) dengan mempertimbangkan konsumsi energi sensor node. Masing-masing partikel atau node dibangkitkan secara acak dan selanjutnya dioptimasi untuk menghasilkan solusi yang baik. Pada penelitian ini fokus optimasinya adalah mengatur posisi node sensor ketika disebar agar dapat menghemat konsumsi energi sehingga dapat digunakan dalam waktu yang lebih lama. Berdasarkan hasil pengujian yang dilakukan maka diketahui bahwa radius pengindraan suatu sensor node berpengaruh terhadap konsumsi energi. Semakin besar radius pengindraan suatu sensor node maka semakin besar pula konsumsi energinya sehingga masa hidup jaringan akan menjadi pendek dan begitu pula sebaliknya.

1. **Rancang Bangun Aplikasi *Early Warning* Dengan Pemanfaatan Pengukuran Suhu Ruangan Berbasis Arduino Mega 2560** *(Gerry Christofer, Herry Sujaini, M.Azhar Irwansyah, 2015 )*

Pada penelitian ini membahas mengenai pembuatan aplikasi pemantauan suhu sebagai deteksi dini kebakaran rumah. Aplikasi akan memantau kenaikan atau penurunan suhu dari batas yang telah ditentukan, jika terjadi kenaikan atau penurunan suhu maka aplikasi akan mengirimkan peringatan dini berupa SMS. Perngkat sensor dibangun dengan Arduino Mega. Untuk antar muka aplikasi mengatur koneksi, SMS, Kontak, serta memulai pemantauan. Dalam penerpannya pastikan antara perangkat Arduino dan perangkat modem GSM untuk SMS telah terhubung dan Driver telah terpasang.

1. **Perbandingan Algoritma Genetika dan *Particle Swarm Optimization* dalam Optimasi Penjadwalan Matakuliah** ( Yuniar Marbun, Nerfita, ST, M.CS, Martaleli Bettiza, S.Si, M.Sc, 2015 )

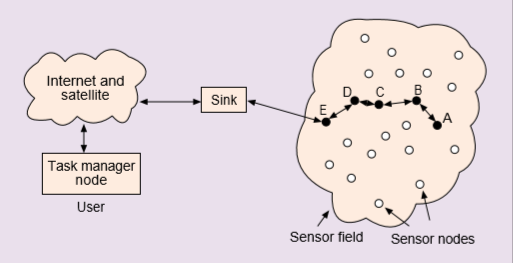
Pada penelitian ini membahas mengenai pembuatan sebuah aplikasi untuk menyelesaikan masalah penjadwalan dengan membandingkan 2 algoritma optimasi yaitu algoritma Genetika (GA) dengan algoritma Particle Swarm Optimization ( PSO ). Secara umum kedua algoritma memiliki hasil yang variatif tergantung parameter input yang dimasukkan. Berdasarkan pengujian yang dilakukan dalam penelitian tersebut dikerahui bahwa algoritma PSO memiliki standar deviasi yang cenderung lebuh rendah dibandingkan GA dengan artian hasil fitness yang dihasilkan oleh PSO lebih stabil jika dibandingkan GA.

## **Tinjauan Teoritis**

Pada penelitian ini digunakan beberapa teori yang dijadikan sebagai acuan dalam proses penelitian antara lain:

### **Wireless Sensor Network**

Wireless Sensor Network atau Jaringan sensor nirkabel merupakan perangkat otonom yang terdistribusi khusus menggunakan sensor dalam memantau kondisi fisik atau lingkungan, seperti suhu, suara, getaran, tekanan, gerakan di lokasi yang berbeda. Wireless Sensor Network terdiri dari dua komponen utama yaitu node sensor yang memiliki fungsi penyimpanan (storage), pemrosesan dan komunikasi, dan sensor yang dapat mengukur atau sensing fenomena atau besaran fisis dari lingkungan dimana sensor tersebut ditempatkan.



**Gambar 2.1 Wireless Sensor Networke**

(Sumber : Akyildiz, dkk, 2002)

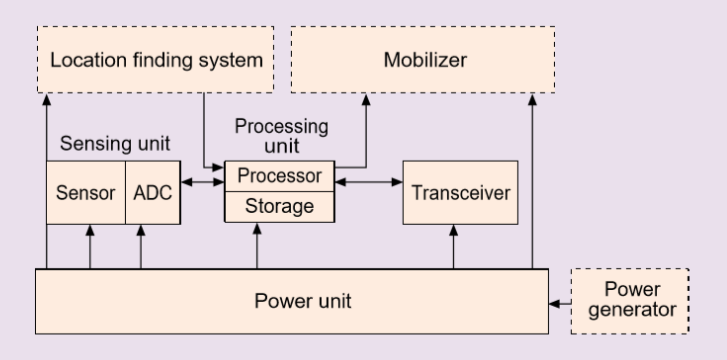
*Node sensor* biasanya tersebar pada *sensor field* seperti yang ditunjukkan pada gambar diatas. Masing-masing node sensor tersebar memiliki kemampuan untuk mengumpulkan data dan merutekan data kembali ke sink. Data dapat dialihkan kembali pada sink oleh arsitektur infrastruktur *multihop* melalui *sink*. Lalu *sink* dapat berkomunikasi dengan task manager node melalui internet dan satellite(Akyildiz, dkk, 2002).

Pada dasarnya WSN merupakan jaringan dari node-node yang berdiri sendiri secara individual, yang memiliki kemampuan sensing atau pengukuran besaran fisis, menyimpan, memproses serta memiliki kemampuan komunikasi, dibangun dengan sedikit atau tanpa infrastruktur yang tetap, node-node tersebut bekerja sama untuk memantau area, situasi atau lingkungan tertentu. Sekarang ini WSN telah dikembangkan pada beberapa lokasi, area atau lingkungan. Berdasarkan itu, WSN dapat diklasifikasikan dalam lima tipe yaitu:

* + - 1. Terestrial WSN, terdiri dari seratus atau ribuan node sensor yang disebar di tanah pada area tertentu, bersifat adhoc dan berkomunikasi dengan base station.
      2. Underground WSN, sejumlah sensor yang disebar dalam gua, atau di pertambangan atau di bawah tanah untuk memantau kondisi di bawah tanah. Untuk berkomunikasi dengan base station biasanya ditempatkan sebuah sink node di atas permukaan tanah.
      3. Underwater WSN, dimana sejumlah sensor ditempatkan di bawah air, seperti di lingkungan lautan. Sensor yang disebar hanya beberapa dan dilengkapi dengan kendaraan yang autonomous yang digunakan untuk mengeksplorasi atau mengambil data dari sensor-sensor tersebut.
      4. Multimedia WSN, terdiri dari node sensor yang dilengkapi dengan peralatan kamera dan mikrofon. Sensor dapat menyimpan, memproses dan menerima data multimedia seperti video, audio dan image.
      5. Mobile WSN, terdiri dari node sensor bergerak yang dapat berpindah dan berinteraksi dengan lingkungan fisik. Pergerakan tersebut dengan cara mengatur dan memposisikan ulang dirinya di jaringan dan untuk dapat melakukan pengukuran, pemrosesan dan berkomunikasi.

### **Node Sensor**

*Node Sensor* adalah komponen penyusun *Wireless Sensor Network* (WSN) yang memiliki bagian khusus untuk mengirim dan menerima data. Salah satu permasalahan yang dihadapi oleh node sensor yaitu hanya memiliki pasokan sumber daya yang terbatas seperti baterai, akan tetapi tetap dituntut untuk menjalankan berbagai proses secara terus menerus.



**Gambar 2.2 Node Sensor**

(Sumber: Akyildiz et al., 2002).

Node sensor terdiri dari empat komponen dasar, yaitu *Sensing Unit,* Processing Unit, Transceiver Unit, dan Power Unit. Keempat komponen tersebut mungkin juga memiliki komponen tambahan yang bergantung pada aplikasi seperti location finding system, power generator, and mobilizer. Sensing unit biasanya terdiri dari dua subunit, yaitu sensors dan *analog-to-digital converters* (ADCs). Sinyal analog dihasilkan oleh sensor berdasarkan kejadian yang diamati, dan dikonversi menjadi sinyal digital oleh ADC, lalu dimasukkan ke dalam Processing Unit. Pada Processing Unit, node sensor berkolaborasi dengan node lain untuk melakukan penginderaan yang ditugaskan. Lalu ada juga transceiver yang digunakan untuk menghubungkan node ke jaringan. Dan salah satu komponen terpenting dari suatu node sensor adalah power unit (unit daya). Unit daya mungkin didukung oleh power scavenging units seperti sel surya, ataupun subunit lain tergantung pada aplikasinya(Akyildiz et al., 2002).

### **Particle Swarm Optimization**

Particle Swarm Optimization (PSO) pertama kali diperkenalkan pada tahun 1995. Gagasan tentang PSO berasal dari perilaku alami burung mencari makanan. Ketika sekelompok burung mencari makanan bersama-sama, masing-masing burung akan melihat-lihat di daerah yang dekat dengan dirinya. Setiap burung akan berkomunikasi dengan burung-burung lain dimana ia menemukan jumlah makanan yang paling dekat dengan wilayahnya. Dengan demikian, semua burung dapat mengetahui daerah mana yang memiliki jumlah besar makanan di seluruh area makan kolektif mereka. Burung akan terus mencari makanan di tempat-tempat terdekat, terutama jumlah sebagian besar makanan yang ditemukan di seluruh wilayah. Meskipun setiap burung mempunyai keterbatasan dalam hal kecerdasan, biasanya ia akan mengikuti kebiasaan (rule) seperti berikut:

1. Seekor burung tidak berada terlalu dekat dengan burung yang lain
2. Burung tersebut akan mengarahkan terbangnya ke arah rata- rata keseluruhan burung
3. Akan memposisikan diri dengan rata-rata posisi burung yang lain dengan menjaga sehingga jarak antar burung dalam kawanan itu tidak terlalu jauh.

Dengan demikian perilaku kawanan burung akan didasarkan pada kombinasi dari 3 faktor simpel berikut:

1. Kohesi - terbang bersama
2. Separasi - jangan terlalu dekat
3. Penyesuaian (alignment) - mengikuti arah bersama

Pada algoritma PSO ini, pencarian solusi dilakukan oleh suatu populasi yang terdiri dari beberapa partikel. Populasi dibangkitkan secara random dengan batasan nilai terkecil dan terbesar. Setiap partikel merepresentasikan posisi atau solusi dari permasalahan yang dihadapi. Setiap partikel melakukan pencarian solusi yang optimal dengan melintasi ruang pencarian (search space). Hal ini dilakukan dengan cara setiap partikel melakukan penyesuaian terhadap posisi terbaik dari partikel tersebut (local best) dan penyesuaian terhadap posisi partikel terbaik dari seluruh kawanan (global best) selama melintasi ruang pencarian. Jadi, penyebaran pengalaman atau informasi terjadi di dalam partikel itu sendiri dan antara suatu partikel dengan partikel terbaik dari seluruh kawanan selama proses pencarian solusi. Setelah itu, dilakukan proses pencarian untuk mencari posisi terbaik setiap partikel dalam sejumlah iterasi tertentu sampai didapatkan posisi yang relatif *steady* atau mencapai batas iterasi yang telah ditetapkan. Pada setiap iterasi, setiap solusi yang direpresentasikan oleh posisi partikel, dievaluasi performansinya dengan cara memasukkan solusi tersebut kedalam *fitness function*. Setiap partikel diperlakukan seperti sebuah titik pada suatu dimensi ruang tertentu. Kemudian terdapat dua faktor yang memberikan karakter terhadap status partikel pada ruang pencarian yaitu posisi partikel dan kecepatan partikel [Kennedy and Eberhart, 1995]. Berikut ini merupakan formulasi matematika yang menggambarkan posisi dan kecepatan partikel pada suatu dimensi ruang tertentu:

Xi (t) = xi1 (t), xi2(t),....,xiN(t) …………..(1)

Vi (t) = vi1 (t), vi2(t),....,viN(t) …………..(2)

Dimana:

X = Posisi partikel

V = Kecepatan partikel i = Indeks partikel

t = Iterasi ke –t

N = Ukuran dimensi ruang

Berikut ini merupakan model matematika yang menggambarkan mekanisme updating status partikel kennedy and Eberhart [1995]:

Vi(t) = vi(t – 1) + ………(3)

………………..(4)

Dimana merepresentasikan local best dari partikel ke-i. Sedangkan merepresentasikan global best dari seluruh kawanan. Sedangkan c1 dan c2 adalah suatu konstanta yang bernilai positif yang biasanya disebut sebagai learning factor. Kemudian r1 dan r2 adalah suatu bilangan random yang bernilai antara 0 sampai 1. Persamaan (3) digunakan untuk menghitung kecepatan partikel yang baru berdasarkan kecepatan sebelumnya, jarak antara posisi saat ini dengan posisi terbaik partikel (local best), dan jarak antara posisi saat ini dengan posisi terbaik kawanan (global best). Kemudian partikel terbang menuju posisi yang baru berdasarkan persamaan (4). Setelah algoritma PSO ini dijalankan dengan sejumlah iterasi tertentu hingga mencapai kriteria pemberhentian, maka akan didapatkan solusi yang terletak pada global best.

Ada beberapa parameter utama untuk model PSO yaitu n1 , n2 , r1, r2, Vmax atau velocity dan ukuran Swarm S. Semua parameter ini dapat menentukan bagaimana mengoptimalkan ruang pencarian untuk fungsi dan mendapatkan hasil yang optimal dalam banyak masalah. Namun, beberapa pengaturan parameter tidak menjamin dalam mencari solusi untuk masalah yang berbeda. Dengan demikian, pengaturan yang sesuai dari masalah untuk masalah harus diletakkan.

1. Swarm: Populasi dari suatu Algoritma. Ukuran swarm atau populasi yang dipilih adalah tergantung pada persoalan yang dihadapi. Ukuran swarm yang umum digunakan berkisar antara 20 sampai 50, meski tidak menutup kemungkinan untuk menggunakan ukuran swarm yang lebih besar lagi. Carlisle A dan Dozier G (2001) menyatakan bahwa ukuran swarm akan mempengaruhi kecepatan proses. Ukuran swarm yang terlalu kecil bisa terjebak pada optimum lokal meskipun waktu prosesnya sangat cepat. Sebaliknya, ukuran swarm yang besar jarang terjebak pada optimum lokal tetapi waktu prosesnya lebih lama.
2. Partikel: Anggota (individu) misalnya burung atau ikan pada suatu swarm. Setiap partikel merepresentasikan suatu solusi yang potensial pada permasalahan yang diselesaikan. Posisi dari suatu partikel ditentukan oleh representasi solusi saat itu.
3. Pbest (personal best): posisi Pbest suatu partikel yang menunjukkan posisi partikel yang dipersiapkan untuk mendapatkan suatu solusi yang terbaik.
4. Gbest (global best): posisi terbaik partikel pada swarm atau posisi terbaik partikel dalam kelompok.
5. Velocity (kecepatan) menentukan arah dimana suatu partikel diperlukan untuk berpindah untuk memperbaiki posisinya semula.

# **BAB III**

**ANALISIS DAN PERANCANGAN**

Pada bagian metodelogi penelitian ini menjelaskan gambaran langkah-langkah yang akan dilakukan dalam menjalankan penelitian ini, langkah-langkah tersebut meliputi pengumpulan data, desain sistem secara keseluruhan, alur simulasi.

## **Data**

Data yang digunakan dalam penelitian ini berupa data acak yang nilainya diperoleh dari pembangkitan dengan *Linear Congruential Generator* (LCG). Nilai acak tersebut akan digunakan sebagai nilai awal koordinat (x,y) dari masing-masing node sensor (partikel).

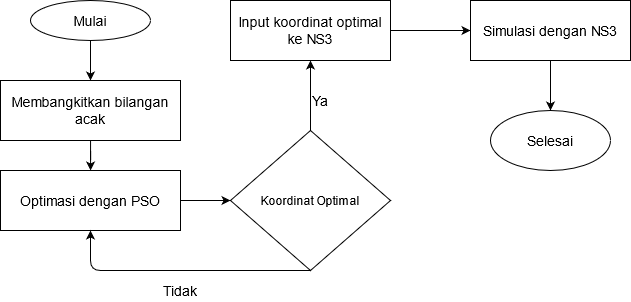
## **Desain Sistem Secara Keseluruhan**

Pada bagian ini akan digambarkan alur secara umum dari penelitian yang akan dilakukan penulis, yaitu dimulai dengan menentukan masalah kemudian mengumpulkan data dan informasi. Setelah data terkumpul kemudian mulai membangun rancangan hingga implementasi sistem, selanjutnya kemudian dilakukan pengujian dan analisi apakah terdapat sensor yang tidak aktif atau tidak terhubung antara satu dengan yang lain menggunakan NS3.



**Gambar 3.1 Alur Penelitian**

Sistem yang akan dibangun merupakan implementasi dalam simulasi. Simulasi akan dijalankan dengan *software* NS3 pada Sister Operasi Linux Ubuntu 18.04. Alur simulasi secara keseluruhan diilustrasikan oleh diagram dibawah ini:



**Gambar 3.2 Alur Simulasi**

Dalam penelitian ini simulasi akan dilakukan dengan NS3 sedangkan untuk optimasi PSO dilakukan dengan menggunakan google colab. Berikut merupakan proses instalasi aplikasi yang dibutuhkan dalam simulasi.

### **Instalasi NS3**

Untuk melakukan tahap simulasi jaringan WSN, disini akan menggunakan program NS3. Proses instalasi dan cara menjalankan program NS-3 dilakukan sebagai berikut.

1. Hal pertama yang dilakukan yaitu menginstall NS-3 dengan prasyarat sebagai berikut.
2. Unduh program NS-3 pada tautan <https://www.nsnam.org/ns-3-dev/download/>.
3. Masuk ke folder utama NS-3 (ns-3.xx.x/ns-allinone-3.xx.x/ns-3.xx.x/)
4. Build program NS-3 di terminal dengan perintah

./waf

1. Untuk proses *compile* dan *running* program dapat dilakukan dengan perintah

./waf --run [nama program]

### **Istalasi Net Anim**

NetAnim merupakan salah satu program animasi yang dapat mem-visualisasikan simulasi jaringan menggunakan file pelacakan XML yang dikumpulkan selama simulasi pada NS-3. Berikut merupakan langkah-langkah instalasi dan cara penggunaan NetAnim.

1. Install program prasyarat yang diperlukan NetAnim dengan perintah sebagai berikut.

sudo apt-get install mercurial

sudo apt-get install qt4-dev-tools

1. Unduh program NetAnim dengan perintah berikut.

hg clone http://code.nsnam.org/netanim

1. Build program NetAnim dengan perintah berikut.

make clean

qmake NetAnim.pro

make

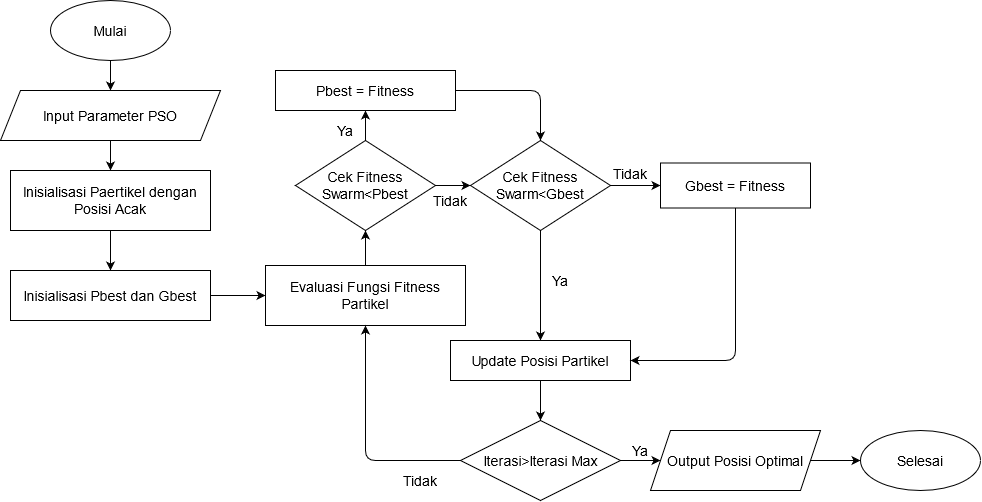
1. Masuk ke dalam folder utama NetAnim
2. Untuk menjalankan NetAnim dengan perintah berikut.

./NetAnim

## **Tahap Prosesing**

### **Optimasi**

Tahap optimasi akan dilakukan dengan algoritma *Particle Swarm Optimization* (PSO).



**Gambar 3.3 Algoritma PSO**

Pada gambar terlihat bahwa proses optimasi diawali dengan menginputkan beberapa parameter seperti jumlah partikel, jumlah iterasi maksimum dan beberapa parameter lainnya. Setelah menentukan jumlah partikelnya kemudian dilakukan inisialisasi posisi (X, Y) dari masing-masing partikel dengan bilangan acak yang dibangkitkan dengan *linear congruential generator* (LCG). Dari posisi masing-masing partikel yang telah ditentukan tersebut, kemudian di inisialisasikan nilai Pbest dan Gbest nya. Setelah semua nilai didapat selanjutnya akan dievaluasi nilai fitness masing-masing partikel yang kemudian hasilnya digunakan untuk menentukan nilai Pbest dan Gbest pada iterasi selanjutnya. Setelah itu proses selanjutnya ialah melakukan update posisi terbaru dari masing-masing partikel berdasarkan nilai Pbest dan Gbest yang terbaru. Pada tahap akhir akan di cek apakah iterasi sudah mencapai iterasi maksimum, jika belum makan akan dilakukan evaluasi fungsi fitness partikel dan proses akan dilakukan seperti sebelumnya. Jika iterasi mencapai maksimum maka hasil dari masing-masing iterasi akan di cetak dan proses optimasi dinyatakan selesai. Hasil dari proses optimasi tiap iterasi kemudian akan diujikan pada simulasi dengan NS3.

### **Simulasi**

Tahap simulasi dilakukan untuk menguji koneksi antar node sensor, pengujian dilakukan dengan membuat sebuah node titik api yang mengirimkan pesan kebakaran ke node terdekatnya. Pesan kebakaran tersebut kemudian akan dikirimkan ke masing-masing node sensor dan akan dilihat node mana yang terhubung dan yang tidak terhubung. Simulasi dilakukan selama 1500 detik dengan radius jangkauan sensor di tetapkan sebesar 100 meter. Adapun jumlah node yang digunakan pada simulasi ini yaitu sebanyak 10 buah node sensor dan 1 buah node api. Simulasi akan dijalankan per iterasi, jadi akan ada 10 kali simulasi karena akan ada 10 iterasi pada tahap operasi PSO nya.

### **Spesifikasi Program**

Tahap simulasi dibagi menjadi dua dan akan dijalankan dengan menggunakan dua sistem operasi yang berbeda, hal ini penulis lakukan karena keterbatasan perangkat yang penulis gunakan.

Tahap pertama yaitu tahap optimasi akan dijalankan pada perangkat dengan spesifikasi *hardware* dan *software* sebagai berikut:

Sistem Operasi : Windows 10 (64-bit)

Memori : 4000MB

Tools : Google Colab

Bahasa pemrograman: Python

Tahap kedua yaitu tahap Simulasi akan dijalankan pada perangkat dengan spesifikasi hardware dan software sebagai berikut:

Sistem Operasi : Ubuntu Desktop (64-bit)

Memori : 2048MB

Tools : NS3 Simulator dan Wireshark

Bahasa pemrograman: C++

# **BAB IV**

**HASIL DAN PEMBAHASAN**

* 1. **Implementasikan**

### **Script dan Penjelasan Program**

Dalam tahap implementasi, dibahas mengenai tahapan-tahapan yang dilakukan dalam proses optimasi dengan particle swarm optimization dan simulasi yang dilakukan dengan NS3. Setelah sebelumnya dilakukan proses perancangan, maka kemudian akan dilakukan pembuatan script program, melakukan optimasi, menjalankan simulasi, dan mengumpulkan data yang dihasilkan pada proses optimasi dan simulasi.

Dalam pembuatan program optimasi dan simulasi ini, dibangun berdasarkan rancangan tahap simulasi dan flowchart yang telah dibuat sebelumnya. Pada NS3 sudah tersedia beberapa modul dan contoh script yang dapat digunakan untuk membantu proses pembuatan program ini. Sedangkan untuk tahap optimasi sendiri mengambil referensi dari beberapa jurnal terkait dan internet. Setelah program optimasi dan simulasi dibuat sesuai dengan rancangan, maka selanjutnya program dijalankan. Pada tahap optimasi akan didapat hasil berupa koordinat dari masing-masing node, sedangkan pada tahap simulasi hasil yang didapat berupa file .xml dan .pcap, kemudian dilakukan tahap analisis.

* + - 1. **Tahap Optimasi**

1. Membangkitkan bilangan random

Berikut merupakan script pembangkitan bilangan random, yang hasilnya kemudian akan digunakan untuk dioptimalkan dengan PSO. Hasil dari script ini nantinya berupa nilai (x,y).

|  |
| --- |
| def lcg(nilai):      a = 21      c = 3      m = 16      global lcg\_result      lcg\_result = (a\*lcg\_result + c) % m      return lcg\_result  global lcg\_c  global lcg\_x  global lcg\_y  lcg\_x = lcg(10)  lcg\_y = lcg(10)  lcg\_c = lcg(100) |

1. Inisialisasi PSO properti

Pada penelitian ini menggunakan 10 node partikel dan jumlah iterasi sebanyak 10 kali iterasi.

|  |
| --- |
| n\_iterations = 10  n\_particles = 10  target\_error =1e-6 |

1. Inisialisasi pbest dan gbest

|  |
| --- |
| def \_\_init\_\_(self):          x = lcg(10)          y = lcg(10)            self.position = np.array([x, y])          self.pBest\_position = self.position          self.pBest\_value = float('inf')          self.velocity = np.array([0,0]) |
| def \_\_init\_\_(self, target, target\_error, n\_particles):          self.target = target          self.target\_error = target\_error          self.n\_particles = n\_particles          self.particles = []          self.gBest\_value = float('inf')          self.gBest\_position = np.array([lcg(10), lcg(10)]) |

1. Pengecekan pbest dengan fitness

|  |
| --- |
| def set\_pBest(self):          for particle in self.particles:              fitness\_candidate = self.fitness(particle)              if(particle.pBest\_value > fitness\_candidate):                  particle.pBest\_value = fitness\_candidate                  particle.pBest\_position = particle.position |

1. Pengecekan gbest dengan fitness

|  |
| --- |
| def set\_gBest(self):     for particle in self.particles:       best\_fitness\_candidate = self.fitness(particle)       if(self.gBest\_value > best\_fitness\_candidate):       self.gBest\_value = best\_fitness\_candidate           self.gBest\_position = particle.position |

1. Update posisi

|  |
| --- |
| def update\_particles(self):          confidence = lcg(100)          global i          i = 0          for particle in self.particles:              global W              i += 1              inertial = W \* particle.velocity              self\_confidence = c1 \* rd.random() \* (particle.pBest\_position - particle.position)              swarm\_confidence = c2 \* rd.random() \* (self.gBest\_position - particle.position)              new\_velocity = inertial + self\_confidence + swarm\_confidence              particle.velocity = new\_velocity              particle.update() |
| def update(self):      self.position = self.position + self.velocity      print (f'position {i} :',self.position) |

1. Iterasi

|  |
| --- |
| iteration = 0  while(iteration < n\_iterations):      # set particle best & global best      search\_space.set\_pBest()      search\_space.set\_gBest()      # visualization      search\_space.show\_particles(iteration)        # check conditional      if(abs(search\_space.gBest\_value - search\_space.target) <= search\_space.target\_error):          break      search\_space.update\_particles()      iteration += 1 |

1. Hasil

Berikut merupakan hasil dari salah satu iterasi, tepatnya hasil dari iterasi ke sepuluh.

**Table 4.1 Hasil Iterasi PSO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iterasi** | **posisi node sensor** | | **X** | **Y** |
| **Iterasi 0** | posisi | 1 | 4.899297 | 7.899297 |
| posisi | 2 | 12.20324 | 8.229958 |
| posisi | 3 | 0 | 3 |
| posisi | 4 | 0.541374 | 5.706868 |
| posisi | 5 | 1.834755 | 4.834755 |
| posisi | 6 | 2.428866 | 2.190378 |
| posisi | 7 | 4.831606 | 7.831606 |
| posisi | 8 | 1.846243 | 3.369249 |
| posisi | 9 | 3.156743 | 6.156743 |
| posisi | 10 | 5.344487 | 5.290494 |
|  |  |  |  |  |
| **Iterasi 1** | posisi | 1 | -0.43515 | 2.564852 |
| posisi | 2 | 11.10052 | 7.757367 |
| posisi | 3 | 0 | 3 |
| posisi | 4 | -0.23227 | 1.838662 |
| posisi | 5 | -0.81116 | 2.188844 |
| posisi | 6 | -0.28182 | 3.093941 |
| posisi | 7 | 1.888342 | 4.888342 |
| posisi | 8 | -2.50235 | 2.49953 |
| posisi | 9 | -1.41078 | 1.589216 |
| posisi | 10 | -2.90681 | 1.754226 |
|  |  |  |  |  |
| **Iterasi 2** | posisi | 1 | -4.03408 | -0.40794 |
| posisi | 2 | 4.538743 | 4.945175 |
| posisi | 3 | -0.07978 | 2.96581 |
| posisi | 4 | -2.94784 | -0.16896 |
| posisi | 5 | -3.38571 | 0.606319 |
| posisi | 6 | -2.49505 | 3.107882 |
| posisi | 7 | -2.49105 | 1.516196 |
| posisi | 8 | -4.76218 | 1.907057 |
| posisi | 9 | -4.38304 | -0.61861 |
| posisi | 10 | -7.03245 | -0.01391 |
|  |  |  |  |  |
| **Iterasi 3** | posisi | 1 | -8.05327 | -1.60263 |
| posisi | 2 | -4.14703 | 1.222703 |
| posisi | 3 | -2.7931 | 1.802957 |
| posisi | 4 | -7.14332 | -1.06505 |
| posisi | 5 | -6.47803 | -0.49194 |
| posisi | 6 | -6.44347 | 1.159657 |
| posisi | 7 | -5.81106 | -0.55071 |
| posisi | 8 | -7.1935 | 0.509653 |
| posisi | 9 | -6.96427 | -1.47258 |
| posisi | 10 | -9.09528 | -0.89798 |
|  |  |  |  |  |
| **Iterasi 4** | posisi | 1 | -10.962 | -1.59195 |
| posisi | 2 | -12.7469 | -2.46296 |
| posisi | 3 | -9.39207 | -1.02517 |
| posisi | 4 | -9.57349 | -1.48464 |
| posisi | 5 | -10.1725 | -1.37436 |
| posisi | 6 | -9.0579 | -0.31123 |
| posisi | 7 | -8.48133 | -1.69098 |
| posisi | 8 | -8.42393 | -0.19998 |
| posisi | 9 | -9.44221 | -1.57941 |
| posisi | 10 | -10.1267 | -1.34001 |
|  |  |  |  |  |
| **Iterasi 5** | posisi | 1 | -12.4548 | -1.60535 |
| posisi | 2 | -17.0468 | -4.30579 |
| posisi | 3 | -14.6222 | -3.26664 |
| posisi | 4 | -12.2944 | -2.15866 |
| posisi | 5 | -13.4725 | -2.42987 |
| posisi | 6 | -13.5082 | -2.87997 |
| posisi | 7 | -10.439 | -2.37378 |
| posisi | 8 | -12.4364 | -2.33317 |
| posisi | 9 | -13.4886 | -2.38341 |
| posisi | 10 | -12.2836 | -2.26441 |
|  |  |  |  |  |
| **Iterasi 6** | posisi | 1 | -14.0859 | -2.13232 |
| posisi | 2 | -19.1968 | -5.22721 |
| posisi | 3 | -18.7519 | -5.03652 |
| posisi | 4 | -17.4463 | -4.20864 |
| posisi | 5 | -15.8985 | -3.3649 |
| posisi | 6 | -18.8528 | -5.42124 |
| posisi | 7 | -15.1331 | -3.80148 |
| posisi | 8 | -18.4343 | -5.10763 |
| posisi | 9 | -16.8493 | -3.50802 |
| posisi | 10 | -17.2244 | -4.3819 |
|  |  |  |  |  |
| **Iterasi 7** | posisi | 1 | -17.5701 | -4.01179 |
| posisi | 2 | -20.2718 | -5.68791 |
| posisi | 3 | -21.0423 | -6.01814 |
| posisi | 4 | -21.1411 | -5.88461 |
| posisi | 5 | -19.4796 | -5.16947 |
| posisi | 6 | -21.807 | -6.53287 |
| posisi | 7 | -20.3663 | -5.52789 |
| posisi | 8 | -21.5007 | -6.50545 |
| posisi | 9 | -19.189 | -4.55324 |
| posisi | 10 | -20.9181 | -5.9649 |
|  |  |  |  |  |
| **Iterasi 8** | posisi | 1 | -21.6844 | -6.36308 |
| posisi | 2 | -21.5475 | -6.32455 |
| posisi | 3 | -22.4588 | -6.69155 |
| posisi | 4 | -23.2323 | -6.95999 |
| posisi | 5 | -22.7055 | -6.91264 |
| posisi | 6 | -23.2841 | -7.08869 |
| posisi | 7 | -23.885 | -7.02041 |
| posisi | 8 | -23.2157 | -7.22063 |
| posisi | 9 | -21.2204 | -5.72724 |
| posisi | 10 | -23.1552 | -7.00577 |
|  |  |  |  |  |
| **Iterasi 9** | posisi | 1 | -24.0968 | -7.64483 |
| posisi | 2 | -22.9032 | -6.85659 |
| posisi | 3 | -23.5873 | -7.12517 |
| posisi | 4 | -24.2824 | -7.49809 |
| posisi | 5 | -24.6154 | -7.81135 |
| posisi | 6 | -24.2565 | -7.34002 |
| posisi | 7 | -25.6444 | -7.76667 |
| posisi | 8 | -24.4937 | -7.45242 |
| posisi | 9 | -23.3503 | -6.855 |
| posisi | 10 | -24.8274 | -7.53731 |
|  |  |  |  |  |
| **Iterasi 10** | posisi | 1 | -25.9432 | -8.33611 |
| posisi | 2 | -25.6329 | -7.80381 |
| posisi | 3 | -25.1476 | -7.65259 |
| posisi | 4 | -25.292 | -7.8627 |
| posisi | 5 | -26.1846 | -8.23403 |
| posisi | 6 | -25.4346 | -7.67837 |
| posisi | 7 | -26.524 | -8.1398 |
| posisi | 8 | -25.4664 | -7.65947 |
| posisi | 9 | -25.1603 | -7.71495 |
| posisi | 10 | -25.7634 | -7.83113 |

* + - 1. **Tahap Simulasi**

1. Fungsi Utama

Fungsi utama berisikan inisialisasi kelas untuk simulasi, set time resolusi, inisialisasi protokol UDP, dan beberpa parameter yang digunakan, salah satunya distanceToRx yang merepresentasikan jangkauan node sensor dalam mengirimkan data.

|  |
| --- |
| int main (int argc, char \*argv[])  {  std::string phyMode ("DsssRate2Mbps");  double Prss = -80; // dBm  uint32\_t PpacketSize = 200; // bytes  bool verbose = false;  double distance = 100;  uint32\_t NodeFire = 1;  // simulation parameters  uint32\_t numPackets = 100; // number of packets to send  double interval = 1; // seconds  double startTime = 0.0; // seconds  double distanceToRx = 100.0; // meters  double offset = 81;  CommandLine cmd;  cmd.AddValue ("phyMode", "Wifi Phy mode", phyMode);  cmd.AddValue ("Prss", "Intended primary RSS (dBm)", Prss);  cmd.AddValue ("PpacketSize", "size of application packet sent", PpacketSize);  cmd.AddValue ("numPackets", "Total number of packets to send", numPackets);  cmd.AddValue ("startTime", "Simulation start time", startTime);  cmd.AddValue ("distanceToRx", "X-Axis distance between nodes", distanceToRx);  cmd.AddValue ("verbose", "Turn on all device log components", verbose);  cmd.Parse (argc, argv);  // Convert to time object  Time interPacketInterval = Seconds (interval);  // disable fragmentation for frames below 2200 bytes  Config::SetDefault ("ns3::WifiRemoteStationManager::FragmentationThreshold",  StringValue ("2200"));  // turn off RTS/CTS for frames below 2200 bytes  Config::SetDefault ("ns3::WifiRemoteStationManager::RtsCtsThreshold",  StringValue ("2200"));  // Fix non-unicast data rate to be the same as that of unicast  Config::SetDefault ("ns3::WifiRemoteStationManager::NonUnicastMode",  StringValue (phyMode)); |

1. Fungsi *Command Setup*

Fungsi *Command Setup* berfungsi untuk menyimpan nilai parameter saat menjalankan perintah ./waf menggunakan terminal. Nilai parameter itu seperti packet size, waktu simulasi dimulasi, jumlah node, dan sebagainya.

|  |
| --- |
| CommandLine cmd;  cmd.AddValue ("phyMode", "Wifi Phy mode", phyMode);  cmd.AddValue ("Prss", "Intended primary RSS (dBm)", Prss);  cmd.AddValue ("PpacketSize", "size of application packet sent", PpacketSize);  cmd.AddValue ("numPackets", "Total number of packets to send", numPackets);  cmd.AddValue ("startTime", "Simulation start time", startTime);  cmd.AddValue ("distanceToRx", "X-Axis distance between nodes", distanceToRx);  cmd.AddValue ("verbose", "Turn on all device log components", verbose);  cmd.Parse (argc, argv); |

1. *Print Received Packet*

*Print Received Packet* berfungsi untuk mencetak paket data yang dikirim dan diterima oleh sebuah node pada saat simulasi berjalan.

|  |
| --- |
| static inline std::string  PrintReceivedPacket (Address& from)  {  InetSocketAddress iaddr = InetSocketAddress::ConvertFrom (from);  std::ostringstream oss;  oss << "--\nReceived one packet! Socket: " << iaddr.GetIpv4 ()  << " port: " << iaddr.GetPort ()  << " at time = " << Simulator::Now ().GetSeconds ()  << "\n--";  return oss.str ();  } |

1. Pembuatan *Node*

Pada tahap ini ditunjukkan simulasi akan dibuat dengan 10 node. Selain itu, akan dibuat juga node api.

|  |
| --- |
| NodeContainer c;  c.Create (10);  NodeContainer networkNodes;  networkNodes.Add (c.Get (0));  networkNodes.Add (c.Get (1));  networkNodes.Add (c.Get (2));  networkNodes.Add (c.Get (3));  networkNodes.Add (c.Get (4));  networkNodes.Add (c.Get (5));  networkNodes.Add (c.Get (6));  networkNodes.Add (c.Get (7));  networkNodes.Add (c.Get (8));  networkNodes.Add (c.Get (9));  NodeContainer f;  f.Create(NodeFire); |

1. *Set Node sender and receiver*

Program akan menentukan node yang akan mengirimkan paket (dalam hal ini paket berupa pesan kebakaran) dan node yang akan menjadi tujuan atau penerimanya. Kemudian juga ditentukan kapan node akan mulai mengirim paket dan kapan berhenti.

|  |
| --- |
| TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory");  Ptr<Socket> recvSink = Socket::CreateSocket (networkNodes.Get (9), tid);  InetSocketAddress local = InetSocketAddress (Ipv4Address::GetAny (), 80);  recvSink->Bind (local);  recvSink->SetRecvCallback (MakeCallback (&ReceivePacket));  Ptr<Socket> source = Socket::CreateSocket (networkNodes.Get (0), tid);  InetSocketAddress remote = InetSocketAddress (i.GetAddress(9,0), 80);  source->SetAllowBroadcast (true);  source->Connect (remote); |

1. Inisialisasi *Wifi Setting* dan *Wifi Phy Channel*

*Wifi Setting* adalah jenis wifi yang digunakan pada simulasi ini. Wifi berfungsi sebagai sarana pengirim dan penerima data pada program. Kemudian program membuat *channel* yang berfungsi sebagai penghubung antar node.

|  |
| --- |
| /\*\* Wifi PHY \*\*/  YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default ();  wifiPhy.Set ("RxGain", DoubleValue (-10));  wifiPhy.Set ("TxGain", DoubleValue (offset + Prss-15.0));  wifiPhy.Set ("CcaMode1Threshold", DoubleValue (0.0));    /\*\* wifi channel \*\*/  YansWifiChannelHelper wifiChannel;  wifiChannel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");  wifiChannel.AddPropagationLoss ("ns3::FriisPropagationLossModel");  Ptr<YansWifiChannel> wifiChannelPtr = wifiChannel.Create ();  wifiPhy.SetChannel (wifiChannelPtr);  /\*\* MAC layer \*\*/  NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default ();  wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager", "DataMode",  StringValue (phyMode), "ControlMode",  StringValue (phyMode)); |

1. Membuat *Device*

Pembuatan device yang berfungsi penghubung antara node dengan channel.

|  |
| --- |
| // Set it to ad-hoc mode  wifiMac.SetType ("ns3::AdhocWifiMac");  /\*\* install PHY + MAC \*\*/  NetDeviceContainer devices = wifi.Install (wifiPhy, wifiMac, networkNodes);  NetDeviceContainer fakeDevices = wifi.Install(wifiPhy, wifiMac, f); |

1. Set Ip Address

Program akan membuat range IP Address, kemudian akan memberikan IP pada setiap node.

|  |
| --- |
| Ipv4AddressHelper ipv4;  NS\_LOG\_INFO ("Assign IP Addresses.");  ipv4.SetBase ("10.1.1.0", "255.255.255.0");  Ipv4InterfaceContainer i = ipv4.Assign (devices);  Ipv4InterfaceContainer j = ipv4.Assign(fakeDevices);  UdpEchoServerHelper echoServer(91); |

1. Membuat *Animation interface* dan pcap

Setelah melakukan perhitungan paket data dan mencatat pergerakan node, kemudian program akan mencetak trace node dalam bentuk animasi yang disimpan dalam file yang berekstensi .xml dan .pcap

|  |
| --- |
| wifiPhy.EnablePcap("Pcap\_Kebakaran222", devices);  AnimationInterface anm("SimulasiKebakaran.xml"); |

1. Simulasi dimulai dan berhenti

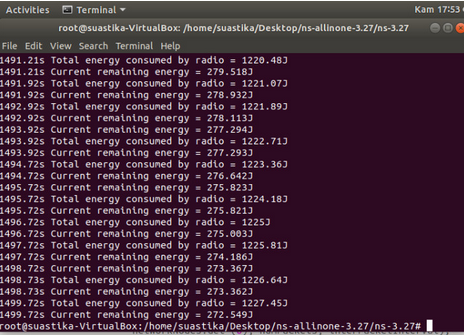
|  |
| --- |
| Simulator::Stop (Seconds (1500.0));  Simulator::Run ();  Simulator::Destroy (); |

### **Tahap Menjalankan Simulasi**

Setelah implementasi pembuatan program selesai, kemudian program akan dijalankan dengan perintah-perintah tertentu yang sudah di konfigurasi sebelumnya. Perintah-perintah ini akan menyesuaikan dengan skenario simulasi yang dijalankan. Sebelum menjalankan program, user harus masuk ke dalam direktori aplikasi NS3 yang terdapat file .waf. Kemudian user membuka terminal, lalu untuk menjalankan proses simulasi perlu ketik perintah:

|  |
| --- |
| ./waf - -run scratch “Nama File” |

Setelah itu, perintah akan mengeksekusi program dan proses simulasi dengan mem-build dan meng-compile program dan simulasi berjalan. Kemudian simulasi akan berakhir dan NS3 akan berhenti.



**Gambar 4.1 Hasil Runing NS3**

* 1. **Hasil dan Pengujian**

### **Perhitungan dengan PSO dan Simulasi Pada NS3**

Pada tahap optimasi dengan menggunakan metode perhitungan PSO akan menghasilkan output berupa nilai koordinat masing-masing node atau partikel. Berikut merupakan hasil koordinat yang didapat pada setiap iterasi:

1. Posisi Awal

Posisi awal node didapat dari pembangkitan bilangan random dengan metode perhitungan LCG. Berikut merupakan tabel hasil pembangkitan bilangan random dengan LCG tersebut:

**Table 4.2 Hasil Iterasi 0**

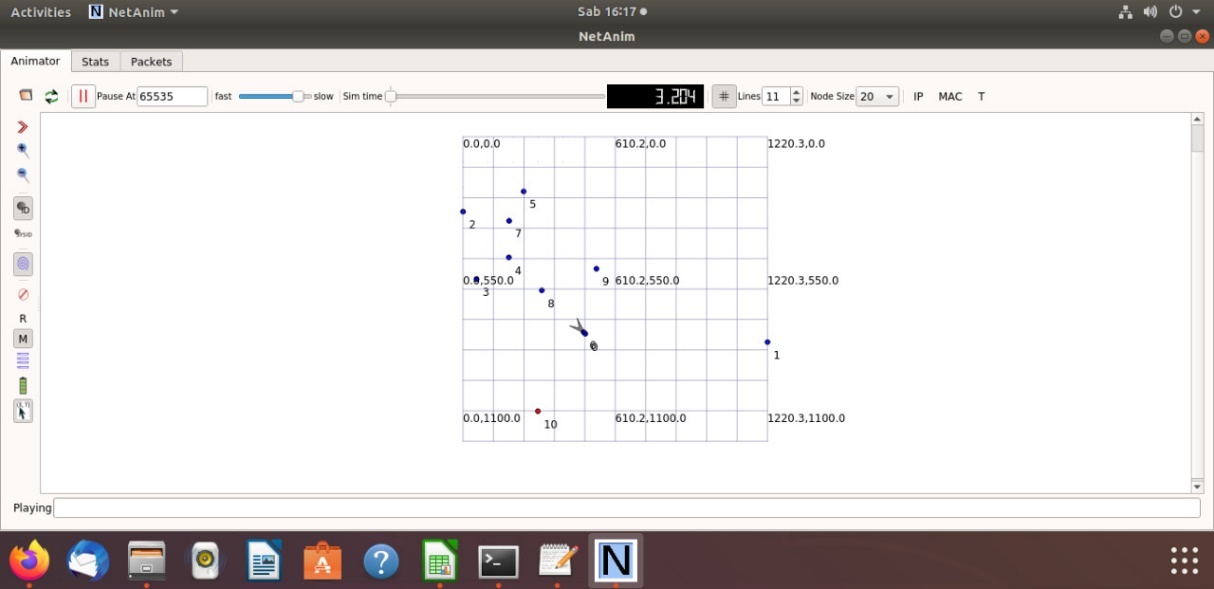
|  |  |  |  |
| --- | --- | --- | --- |
|  |  | X | Y |
| posisi | 1 | 4.89929697 | 7.89929697 |
| posisi | 2 | 12.20323555 | 8.22995809 |
| posisi | 3 | 0 | 3 |
| posisi | 4 | 0.54137351 | 5.70686753 |
| posisi | 5 | 1.83475466 | 4.83475466 |
| posisi | 6 | 2.42886614 | 2.19037795 |
| posisi | 7 | 4.8316055 | 7.8316055 |
| posisi | 8 | 1.84624317 | 3.36924863 |
| posisi | 9 | 3.15674262 | 6.15674262 |
| posisi | 10 | 5.34448665 | 5.29049428 |

Posisi tersebut kemudian dijadikan sebagai patokan dalam iterasi selanjutnya. Karena pada NS3 skala yang akan digunakan yaitu kisaran nilainya ratusan maka nilai dari posisi yang telah didapat akan disesuaikan dengan mengalikan 100 pada setiap nilai posisi nodenya. Hal ini dilakukan agar posisi saat simulasi tidak terlalu dekat. Berikut merupakan tabel hasil penyesuaiannya:

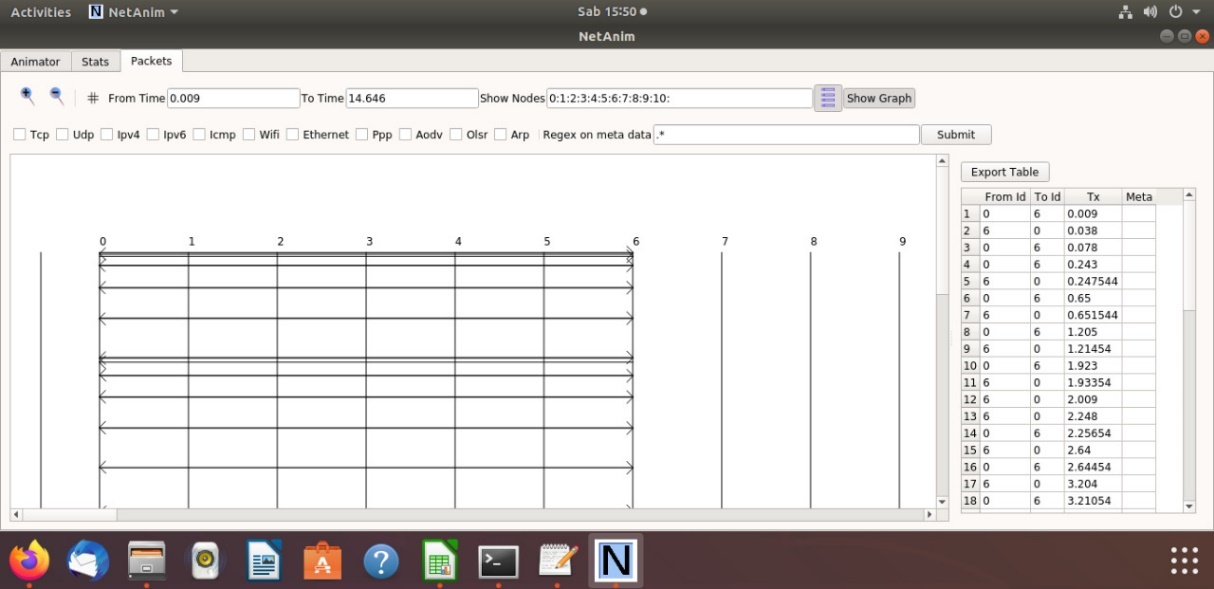
**Table 4.3 Hasil Iterasi 1 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 489.929697 | 789.929697 |
| Posisi 2 | 1220.323555 | 822.995809 |
| Posisi 3 | 0 | 300 |
| Posisi 4 | 54.137351 | 570.686753 |
| Posisi 5 | 183.475466 | 483.475466 |
| Posisi 6 | 242.886614 | 219.037795 |
| Posisi 7 | 483.16055 | 783.16055 |
| Posisi 8 | 184.624317 | 336.924863 |
| Posisi 9 | 315.674262 | 615.674262 |
| Posisi 10 | 534.448665 | 529.049428 |

Untuk deployment dilakukan dengan NS3 dan akan ditampilkan dengan NetAnim. Berikut merupakan gambaran posisi nodenya.



**Gambar 4.2 Posisi Node Iterasi 0**



**Gambar 4.3 Komunikasi Node Pada Iterasi 0**

Dari gambar tersebut dapat diketahui bahwa ada 2 node yang mendapat pesan kebakaran, 2 node tersebut yaitu node 0 dan node 6.

1. Posisi pada Iterasi 1

Posisi pada iterasi pertama diapat dilihat pada tabel berikut:

**Table 4.4 Hasil Iterasi 1**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | X | Y |
| posisi | 1 | -0.43514752 | 2.56485248 |
| posisi | 2 | 11.10052394 | 7.7573674 |
| posisi | 3 | 0 | 3 |
| posisi | 4 | -0.23226751 | 1.83866247 |
| posisi | 5 | -0.81115552 | 2.18884448 |
| posisi | 6 | -0.28182178 | 3.09394059 |
| posisi | 7 | 1.88834224 | 4.88834224 |
| posisi | 8 | -2.50235002 | 2.49953 |
| posisi | 9 | -1.41078436 | 1.58921564 |
| posisi | 10 | -2.9068067 | 1.7542257 |

Dari tabel tersebut dapat diketahui bahwa ada bebenrapa nilai posisi yang negatif, sehingga perlu di normalkan agar menjadi positif. Hal ini dikarenakan pada NetAnim di Ns3 saat di tampilkan nilai posisinya dimulai dari 0 sehingga nilai negatif perlu di normalkan. Agar jarak antar node tidak berubah maka dalam upaya penormalannya posisi akan di Transformasikan dengan cara Translasi, Translasi merupakan pergeseran atau pemindahan semua titik pada bidang geometri sejauh dan arah yang sama. Pada kasus ini posisi node akan digeser dengan cara memindahkan posisi node paling negatif mendekati titik nol. Dari tabel diatas dapat diketahui posisi node 10 merupakan posisi paling negatif yaitu sebesar (-2.9068067), sehingga posisi tersebut akan digeser ke posisi nol. Berikut merupakan tabel hasil translasi posisi node pada iterasi pertama:

**Table 4.5 Hasil Iterasi 1 Yang di Normalkan**

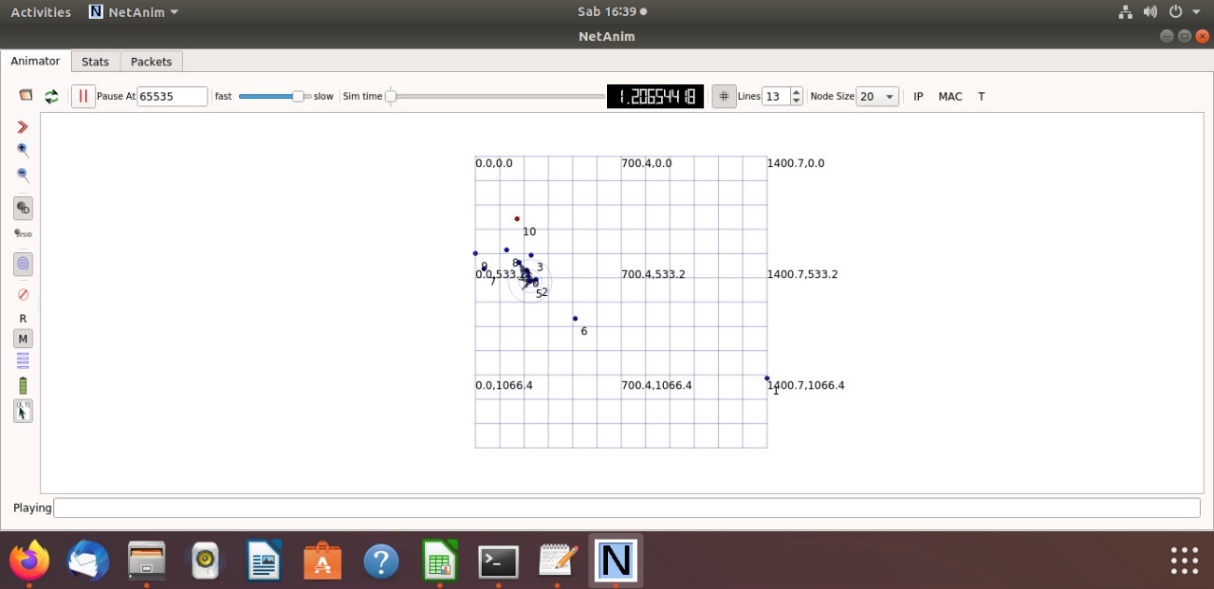
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 2.47165918 | 5.47165918 |
| Posisi 2 | 14.0073306 | 10.6641741 |
| Posisi 3 | 2.9068067 | 5.9068067 |
| Posisi 4 | 2.67453919 | 4.74546917 |
| Posisi 5 | 2.09565118 | 5.09565118 |
| Posisi 6 | 2.62498492 | 6.00074729 |
| Posisi 7 | 4.79514894 | 7.79514894 |
| Posisi 8 | 0.40445668 | 5.4063367 |
| Posisi 9 | 1.49602234 | 4.49602234 |
| Posisi 10 | 0 | 4.6610324 |

Selanjutnya setelah semua posisi node bernilai positif, kemudia posisi tersebut akan disesuaikan dengan skala NS3 yaitu dengan mengalikan nilai posisinya 100. Berikut merupakan posisi node yang telah disesuaikan dengan skala NS3:

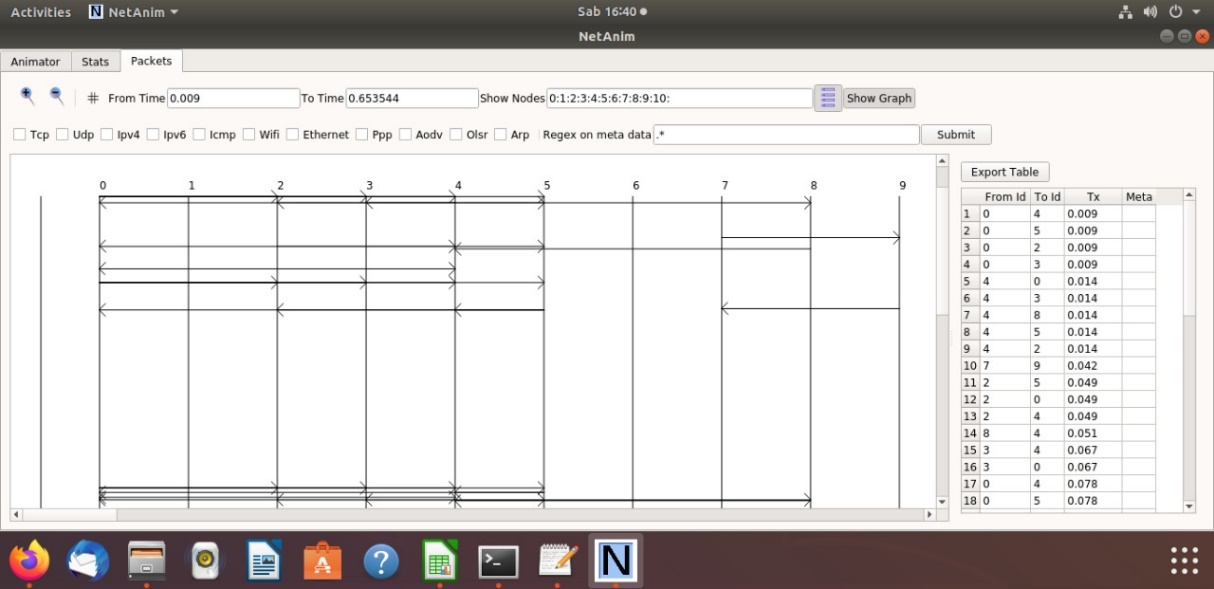
**Table 4.6 Hasil Iterasi 1 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 247.165918 | 547.165918 |
| Posisi 2 | 1400.733064 | 1066.41741 |
| Posisi 3 | 290.68067 | 590.68067 |
| Posisi 4 | 267.453919 | 474.546917 |
| Posisi 5 | 209.565118 | 509.565118 |
| Posisi 6 | 262.498492 | 600.074729 |
| Posisi 7 | 479.514894 | 779.514894 |
| Posisi 8 | 40.445668 | 540.63367 |
| Posisi 9 | 149.602234 | 449.602234 |
| Posisi 10 | 0 | 466.10324 |

Setelah nilai posisi node disesuaikan dengan skala NS3, simulasi dijalankan dan berikut merupakan gambaran posisi node nya ketika ditampilkan dengan NetAnim:



**Gambar 4.4 Posisi Node Iterasi 1**



**Gambar 4.5 Komunikasi Node Iterasi 1**

Berdasarkan gambar tersebut dapat diketahui bahwa ada 8 node yang menerima pesan kebakaran. 8 node tersebut ialah node 0, node 2, node 3, node 4, node 5, node 7, node 8 dan node 9. Pada iterasi berikutnya akan dilakukan hal yang sama secara berulang sampai pada iterasi berakhir.

1. Posisi pada Iterasi 2

Berikut merupakan tabel posisi node pada iterasi ke dua:

**Table 4.7 Hasil Iterasi 2**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -4.034077 | -0.40794055 |
| Posisi 2 | 4.53874273 | 4.94517546 |
| Posisi 3 | -0.07977715 | 2.96580979 |
| Posisi 4 | -2.94784404 | -0.16896025 |
| Posisi 5 | -3.38570986 | 0.60631922 |
| Posisi 6 | -2.49505203 | 3.10788205 |
| Posisi 7 | -2.49104631 | 1.51619633 |
| Posisi 8 | -4.76217917 | 1.90705731 |
| Posisi 9 | -4.38304044 | -0.61860767 |
| Posisi 10 | -7.03245337 | -0.01390859 |

Nilai posisi node pada iterasi ke dua ini kemudian akan di normalkan dengan cara di Translasikan, berikut merupakan tabel posisi node yang sudah di normalkan:

**Table 4.8 Hasil Iterasi 2 yang Dinormalkan**

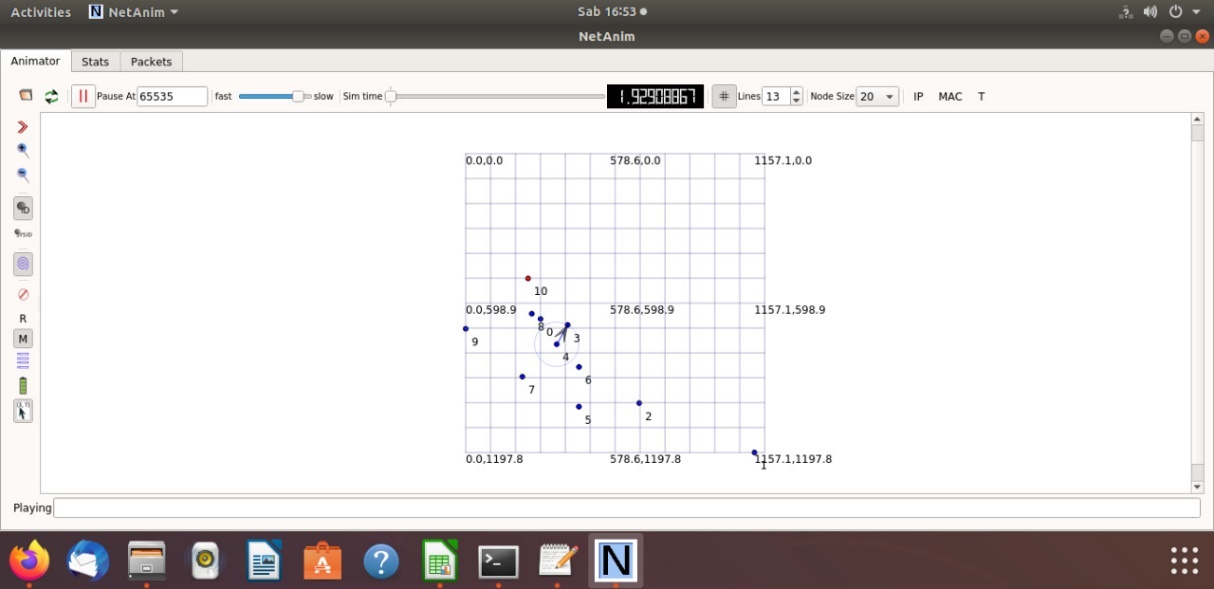
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 2.99837637 | 6.62451282 |
| Posisi 2 | 11.5711961 | 11.97762883 |
| Posisi 3 | 6.95267622 | 9.99826316 |
| Posisi 4 | 4.08460933 | 6.86349312 |
| Posisi 5 | 3.64674351 | 7.63877259 |
| Posisi 6 | 4.53740134 | 10.14033542 |
| Posisi 7 | 4.54140706 | 8.5486497 |
| Posisi 8 | 2.2702742 | 8.93951068 |
| Posisi 9 | 2.64941293 | 6.4138457 |
| Posisi 10 | 0 | 7.01854478 |

Dari posisi node yang sudah dinormalkan tersebut selanjutnya akan di sesuaikan dengan skala NS3, berikut merupakan tabel nilai posisi node yang telah sesuaikan skala nya:

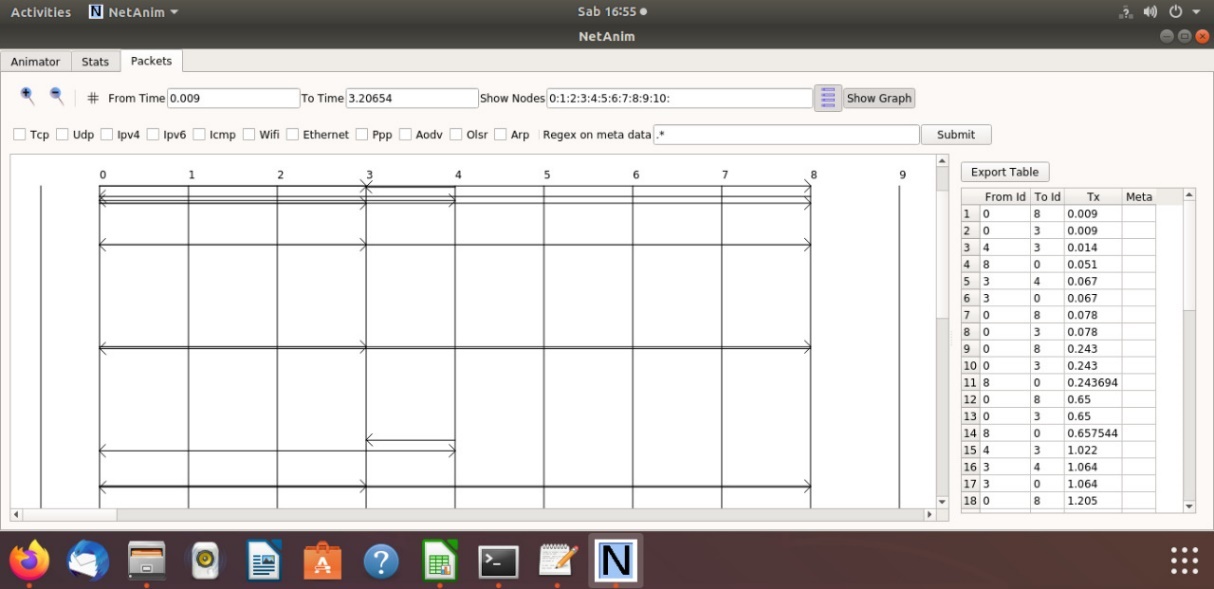
**Table 4.9 Hasil Iterasi 2 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 299.837637 | 662.451282 |
| Posisi 2 | 1157.11961 | 1197.76288 |
| Posisi 3 | 695.267622 | 999.826316 |
| Posisi 4 | 408.460933 | 686.349312 |
| Posisi 5 | 364.674351 | 763.877259 |
| Posisi 6 | 453.740134 | 1014.03354 |
| Posisi 7 | 454.140706 | 854.86497 |
| Posisi 8 | 227.02742 | 893.951068 |
| Posisi 9 | 264.941293 | 641.38457 |
| Posisi 10 | 0 | 701.854478 |

Berdasarkan nilai posisi node yang sudah di sesuaikan dengan skala NS3 tersebut, kemudian simulasi akan dijalankan. Berikut merupakan gambaran posisi node setelah dijalankan di NetAnim.



**Gambar 4.6 Posisi Node Iterasi 2**



**Gambar 4.7 Komunikasi Node Iterasi2**

Pada gambar tersebut dapat dilihat bahwa ada 4 node yang mendapat pesan kebakaran. 4 node tersebut diantaranya yaitu node 0, node 3, node 4, dan node 8.

1. Posisi pada Iterasi 3

Berikut merupakan tabel nilai posisi node hasil iterasi ke tiga:

**Table 4.10 Hasil Iterasi 3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -8.05327232 | -1.60263093 |
| Posisi 2 | -4.1470255 | 1.22270336 |
| Posisi 3 | -2.79310008 | 1.80295711 |
| Posisi 4 | -7.14331955 | -1.06505308 |
| Posisi 5 | -6.47803109 | -0.49194019 |
| Posisi 6 | -6.44346877 | 1.15965662 |
| Posisi 7 | -5.81106441 | -0.55070883 |
| Posisi 8 | -7.19349817 | 0.50965287 |
| Posisi 9 | -6.96426579 | -1.47257552 |
| Posisi 10 | -9.09527671 | -0.89797573 |

Nilai hasil iterasi ketiga tersebut kemudian akan di normalkan dengan cara di translasikan, berikut merupakan tabel nilai posisi node yang sudah di normalkan:

**Table 4.11 Hasil Iterasi 3 yang di Normalkan**

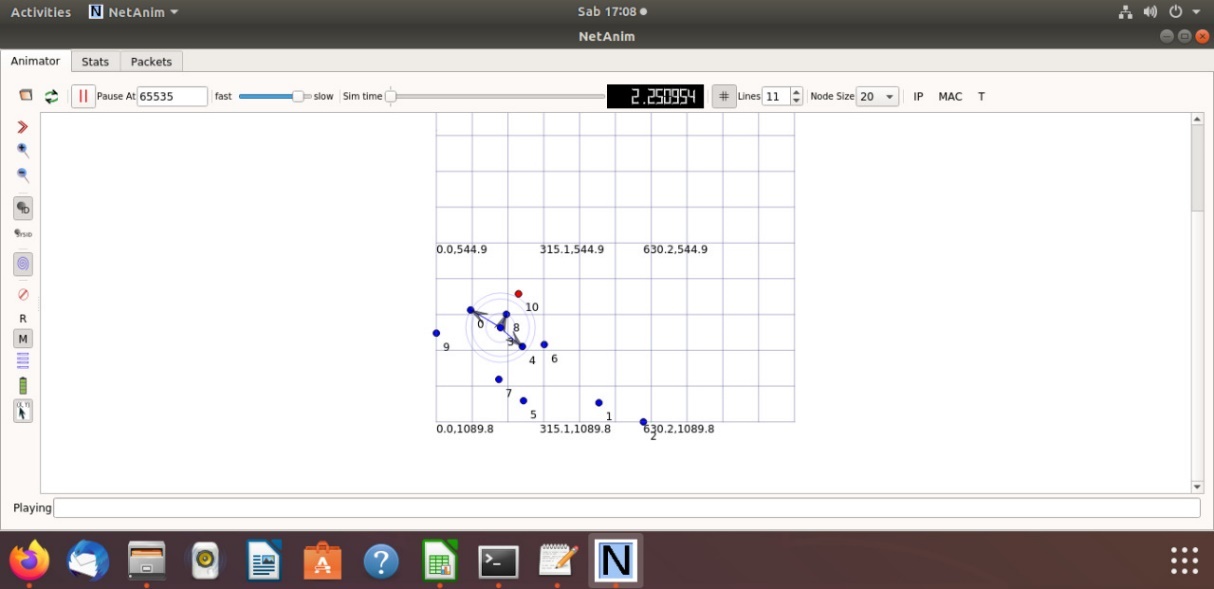
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 1.04200439 | 7.49264578 |
| Posisi 2 | 4.94825121 | 10.31798007 |
| Posisi 3 | 6.30217663 | 10.89823382 |
| Posisi 4 | 1.95195716 | 8.03022363 |
| Posisi 5 | 2.61724562 | 8.60333652 |
| Posisi 6 | 2.65180794 | 10.25493333 |
| Posisi 7 | 3.2842123 | 8.54456788 |
| Posisi 8 | 1.90177854 | 9.60492958 |
| Posisi 9 | 2.13101092 | 7.62270119 |
| Posisi 10 | 0 | 8.19730098 |

Kemudian nilai posisi node yang telah dinormalkan tersebut akan disesuaikan dengan skala NS3, berikut merupakan tabel nilai posisi node yang sudah di sesuaikan dengan skala NS3:

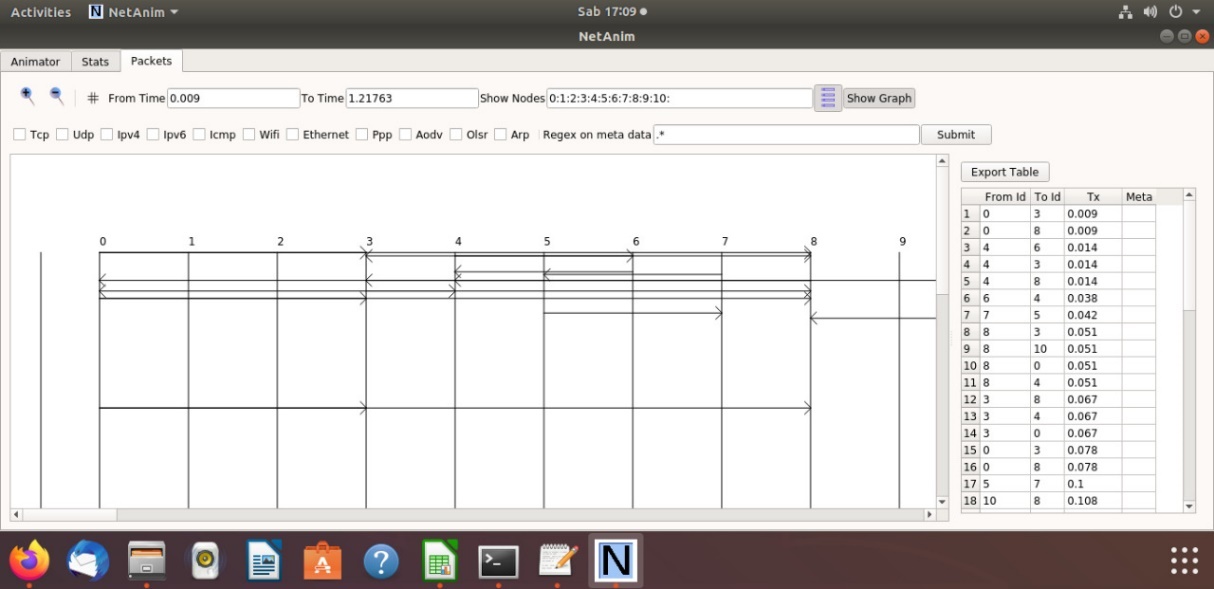
**Table 4.12 Hasil Iterasi 3 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 104.200439 | 749.264578 |
| Posisi 2 | 494.825121 | 1031.79801 |
| Posisi 3 | 630.217663 | 1089.82338 |
| Posisi 4 | 195.195716 | 803.022363 |
| Posisi 5 | 261.724562 | 860.333652 |
| Posisi 6 | 265.180794 | 1025.49333 |
| Posisi 7 | 328.42123 | 854.456788 |
| Posisi 8 | 190.177854 | 960.492958 |
| Posisi 9 | 213.101092 | 762.270119 |
| Posisi 10 | 0 | 819.730098 |

Berdasarkan nilai posisi node yang telah sesuai itu, kemudian simulasi akan dijalankan. Berikut merupakan tampilan posisi node yang digambarkan pada NS3:



**Gambar 4.8 Posisi Node Iterasi 3**



**Gambar 4.9 Komunikasi Node Iterasi 3**

Dari gambar tersebut dapat dilihat bahwa ada 7 node yang menerima pesan. Adapun 7 node tersebut yaitu node 0, node 3, node 4, node 5, node 6, node 7, dan node 8.

1. Posisi pada Iterasi 4

Berikut merupakan tabel nilai posisi node hasil iterasi ke empat:

**Table 4.13 Hasil Iterasi 4**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -10.9619877 | -1.59194797 |
| Posisi 2 | -12.7469034 | -2.46295862 |
| Posisi 3 | -9.39206816 | -1.02517207 |
| Posisi 4 | -9.57349328 | -1.48464471 |
| Posisi 5 | -10.1725256 | -1.37435918 |
| Posisi 6 | -9.05789954 | -0.31122749 |
| Posisi 7 | -8.48133252 | -1.69098445 |
| Posisi 8 | -8.42392854 | -0.19998222 |
| Posisi 9 | -9.44221251 | -1.57941004 |
| Posisi 10 | -10.1266884 | -1.3400093 |

Nilai hasil iterasi ke empat tersebut kemudian akan di normalkan dengan cara di translasikan, berikut merupakan tabel nilai posisi node yang sudah di normalkan:

**Table 4.14 Hasil Iterasi 4 yang di Normalkan**

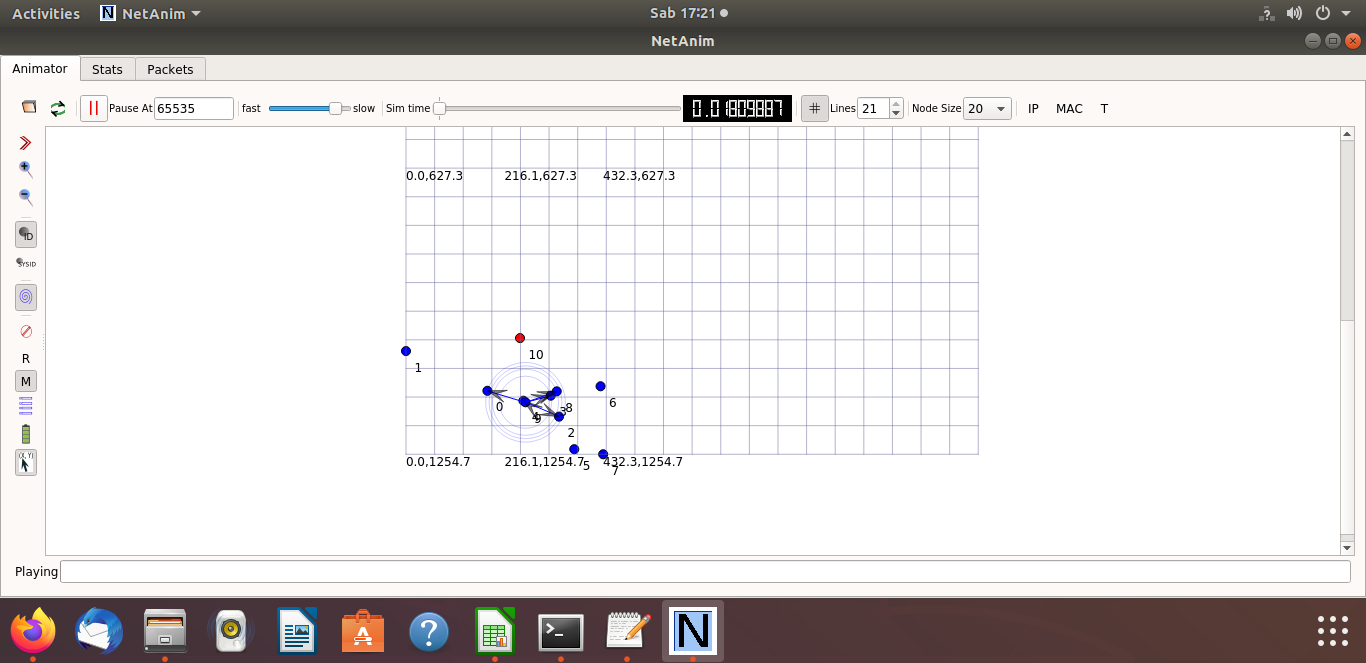
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 1.7849157 | 11.15495547 |
| Posisi 2 | 0 | 10.28394482 |
| Posisi 3 | 3.35483528 | 11.72173137 |
| Posisi 4 | 3.17341016 | 11.26225873 |
| Posisi 5 | 2.57437785 | 11.37254426 |
| Posisi 6 | 3.6890039 | 12.43567595 |
| Posisi 7 | 4.26557092 | 11.05591899 |
| Posisi 8 | 4.3229749 | 12.54692122 |
| Posisi 9 | 3.30469093 | 11.1674934 |
| Posisi 10 | 2.62021506 | 11.40689414 |

Kemudian nilai posisi node yang telah dinormalkan tersebut akan disesuaikan dengan skala NS3, berikut merupakan tabel nilai posisi node yang sudah di sesuaikan dengan skala NS3:

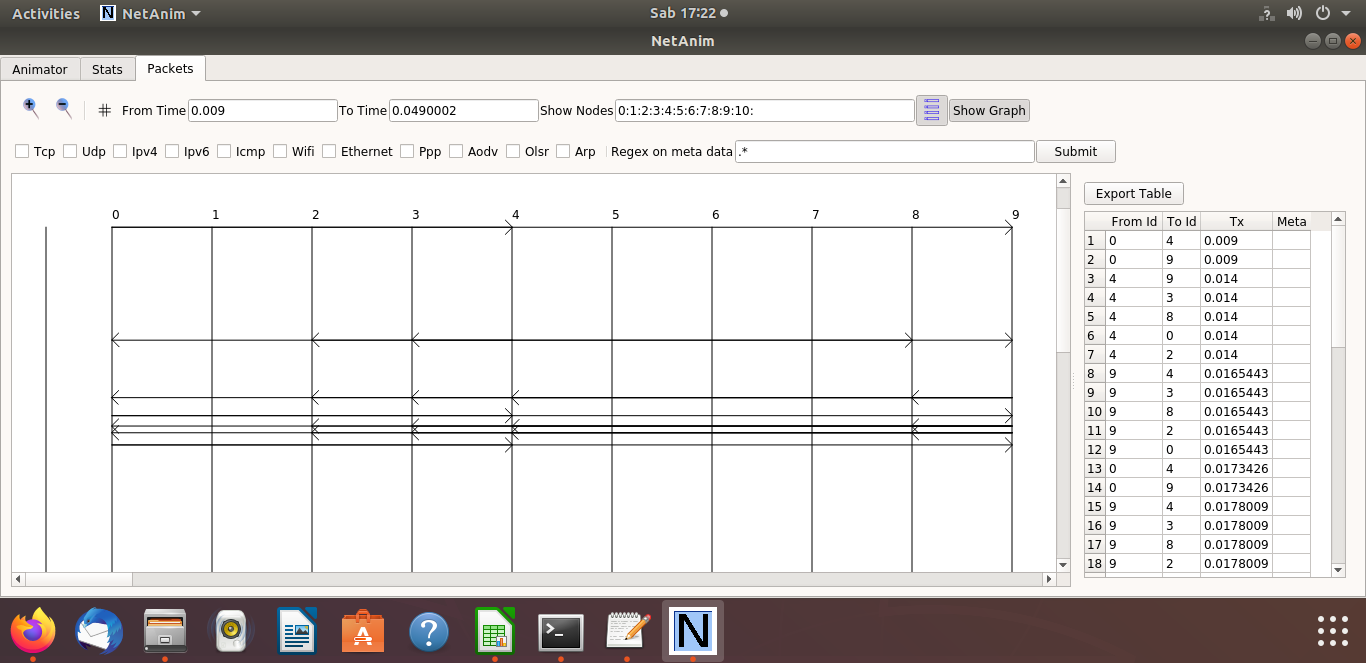
**Table 4.15 Hasil Iterasi 4 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 178.49157 | 1115.49555 |
| Posisi 2 | 0 | 1028.39448 |
| Posisi 3 | 335.483528 | 1172.17314 |
| Posisi 4 | 317.341016 | 1126.22587 |
| Posisi 5 | 257.437785 | 1137.25443 |
| Posisi 6 | 368.90039 | 1243.5676 |
| Posisi 7 | 426.557092 | 1105.5919 |
| Posisi 8 | 432.29749 | 1254.69212 |
| Posisi 9 | 330.469093 | 1116.74934 |
| Posisi 10 | 262.021506 | 1140.68941 |

Berdasarkan nilai posisi node yang telah sesuai itu, kemudian simulasi akan dijalankan. Berikut merupakan tampilan posisi node yang digambarkan pada NS3:



**Gambar 4.10 Posisi Node Iterasi 4**



**Gambar 4.11 Komunikasi Node Iterasi 4**

Dari gambar diatas dapat dilihat bahwa ada 6 node yang menerima pesan. Adapun 6 node tersebut diantaranya node 0, node 2, node 3, node 4, node 8, dan node 9.

1. Posisi pada Iterasi 5

Berikut merupakan tabel nilai hasil iterasi ke lima:

**Table 4.16 hasil Iterasi 5**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -12.4547544 | -1.60534943 |
| Posisi 2 | -17.0468424 | -4.3057896 |
| Posisi 3 | -14.6221591 | -3.26663962 |
| Posisi 4 | -12.2944057 | -2.15866355 |
| Posisi 5 | -13.4725043 | -2.4298696 |
| Posisi 6 | -13.5081819 | -2.87996527 |
| Posisi 7 | -10.4389536 | -2.37377867 |
| Posisi 8 | -12.436365 | -2.33316574 |
| Posisi 9 | -13.4885648 | -2.38341356 |
| Posisi 10 | -12.2836295 | -2.26441263 |

Nilai hasil iterasi terebut kemudian akan di normalkan terlebih dahulu, berikut merupakan nilai posisi node yang telah di normalkan:

**Table 4.17 Hasil Iterasi 5 yang di Normalkan**

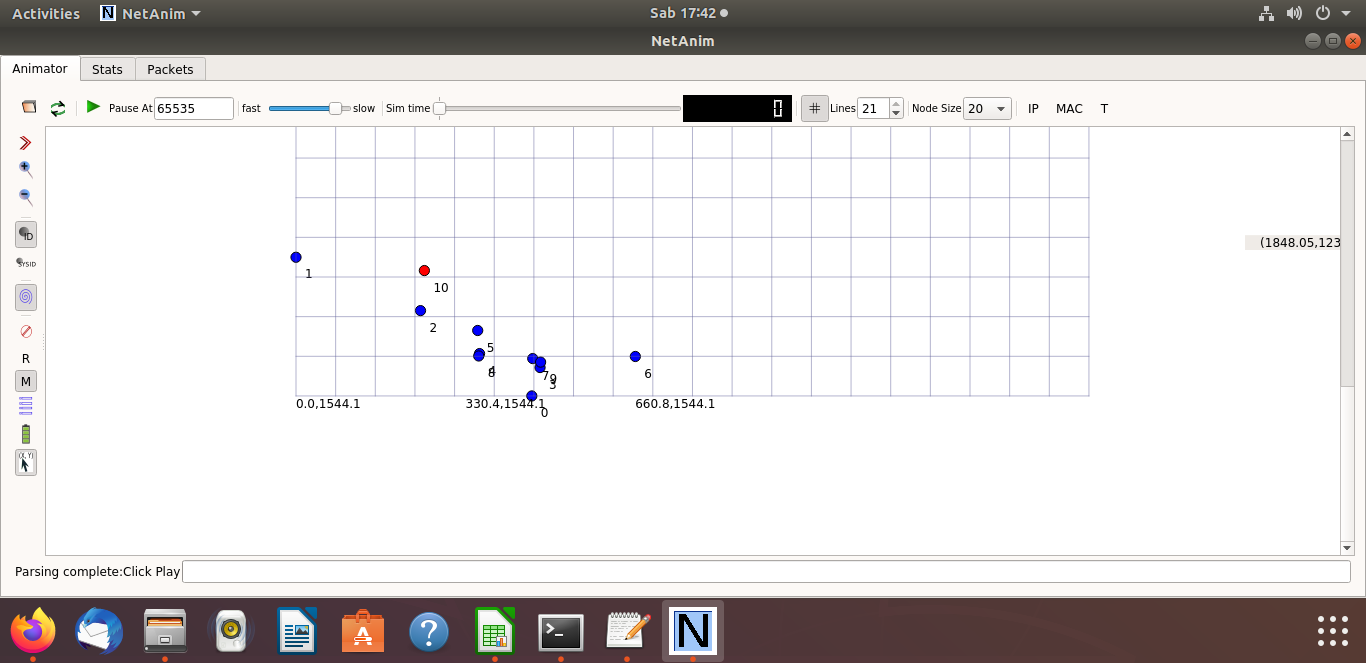
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 4.59208806 | 15.44149298 |
| Posisi 2 | 0 | 12.74105281 |
| Posisi 3 | 2.4246833 | 13.78020279 |
| Posisi 4 | 4.75243672 | 14.88817886 |
| Posisi 5 | 3.57433807 | 14.61697281 |
| Posisi 6 | 3.53866049 | 14.16687714 |
| Posisi 7 | 6.60788879 | 14.67306374 |
| Posisi 8 | 4.61047744 | 14.71367667 |
| Posisi 9 | 3.55827766 | 14.66342885 |
| Posisi 10 | 4.76321294 | 14.78242978 |

Setelah nilai posisi node normal, maka selanjutnya akan disesuaikan terlebih dahulu dengan skala pada NS3 sebelum kemudian di simulasikan nantinya. Berikut merupakan tabel nilai posisi node yang telah di sesuaikan dengan skala NS3:

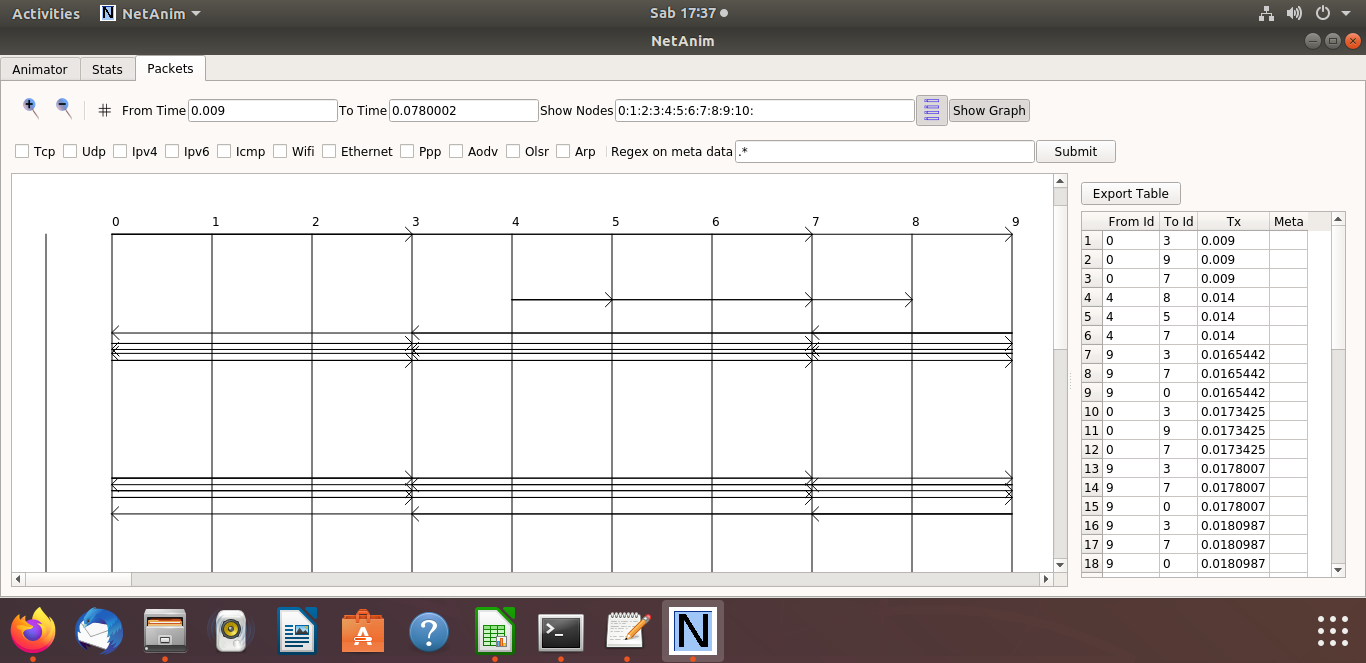
**Table 4.18 Hasil Iterasi 5 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 459.208806 | 1544.1493 |
| Posisi 2 | 0 | 1274.10528 |
| Posisi 3 | 242.46833 | 1378.02028 |
| Posisi 4 | 475.243672 | 1488.81789 |
| Posisi 5 | 357.433807 | 1461.69728 |
| Posisi 6 | 353.866049 | 1416.68771 |
| Posisi 7 | 660.788879 | 1467.30637 |
| Posisi 8 | 461.047744 | 1471.36767 |
| Posisi 9 | 355.827766 | 1466.34289 |
| Posisi 10 | 476.321294 | 1478.24298 |

Nilai yang telah disesuaikan itu selanjutnya akan dimasukkan ke program NS3 dan simulasi kemudian dijalankan. Berikut merupakan gambaran posisi node setalah di jalankan pada NetAnim:



**Gambar 4.12 Posisi Node Iterasi 5**



**Gambar 4.13 Komunikasi Node Iterasi 5**

Dari gambar tersebut dapat diketahui bahwa ada 7 node yang menerima pesan. Adapun 7 node tersebut adalah node 0, node 3, node 4, node 5, node 7, node 8, dan node 9.

1. Posisi pada Iterasi 6

Berikut merupakan tabel nilai posisi node pada iterasi ke enam:

**Table 4.19 Hasil Iterasi 6**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -14.0858611 | -2.13232386 |
| Posisi 2 | -19.1968119 | -5.2272051 |
| Posisi 3 | -18.7518903 | -5.03652442 |
| Posisi 4 | -17.4463242 | -4.2086358 |
| Posisi 5 | -15.898511 | -3.36490201 |
| Posisi 6 | -18.8527578 | -5.42124079 |
| Posisi 7 | -15.1331475 | -3.80147753 |
| Posisi 8 | -18.4342835 | -5.1076334 |
| Posisi 9 | -16.8492627 | -3.50801769 |
| Posisi 10 | -17.2244247 | -4.38189628 |

Nilai tersebut kemudian akan di normalkan terlebih dahulu, sebelum digunakan di NS3. Berikut merupakan tabel nilai posisi node iterasi ke enam yang telah di normalkan:

**Table 4.20 Hasil Iterasi 6 yang di Normalkan**

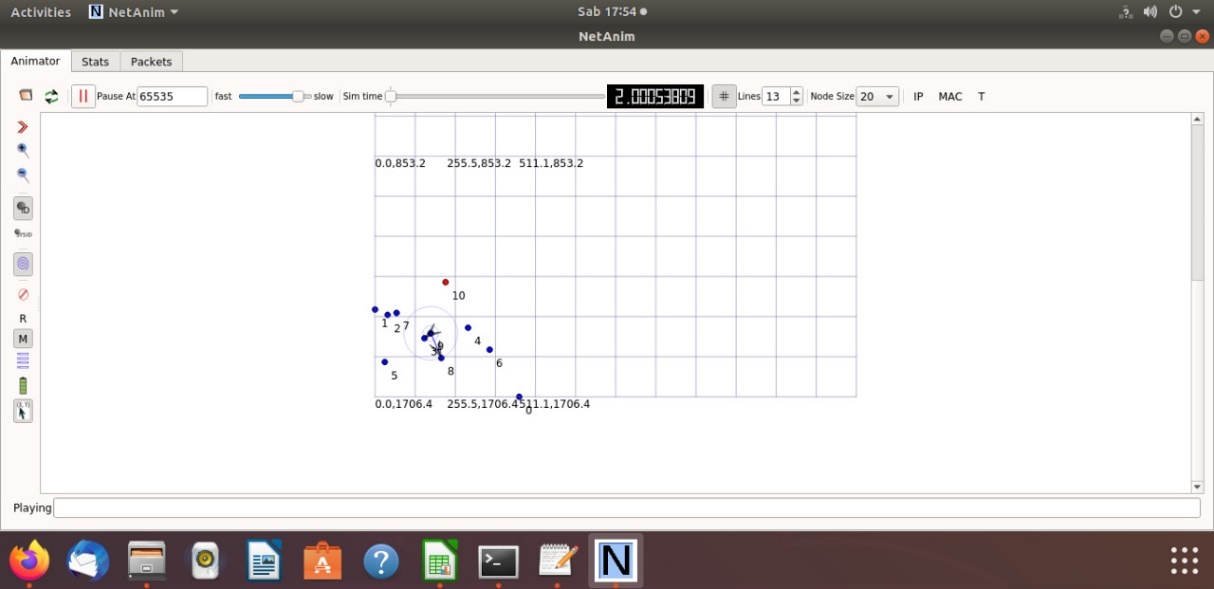
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 5.11095081 | 17.06448803 |
| Posisi 2 | 0 | 13.96960679 |
| Posisi 3 | 0.44492159 | 14.16028747 |
| Posisi 4 | 1.75048772 | 14.98817609 |
| Posisi 5 | 3.29830085 | 15.83190988 |
| Posisi 6 | 0.34405413 | 13.7755711 |
| Posisi 7 | 4.06366444 | 15.39533436 |
| Posisi 8 | 0.76252839 | 14.08917849 |
| Posisi 9 | 2.34754923 | 15.6887942 |
| Posisi 10 | 1.97238723 | 14.81491561 |

Setelah nilai posisi node nya normal maka kemudian akan disesuaikan dengan skala NS3 agar kemudian dapat di jalankan pada proses simulasi. Berikut merupakan tabel nilai posisi node yang sudah disesuaikan skalanya dengan NS3:

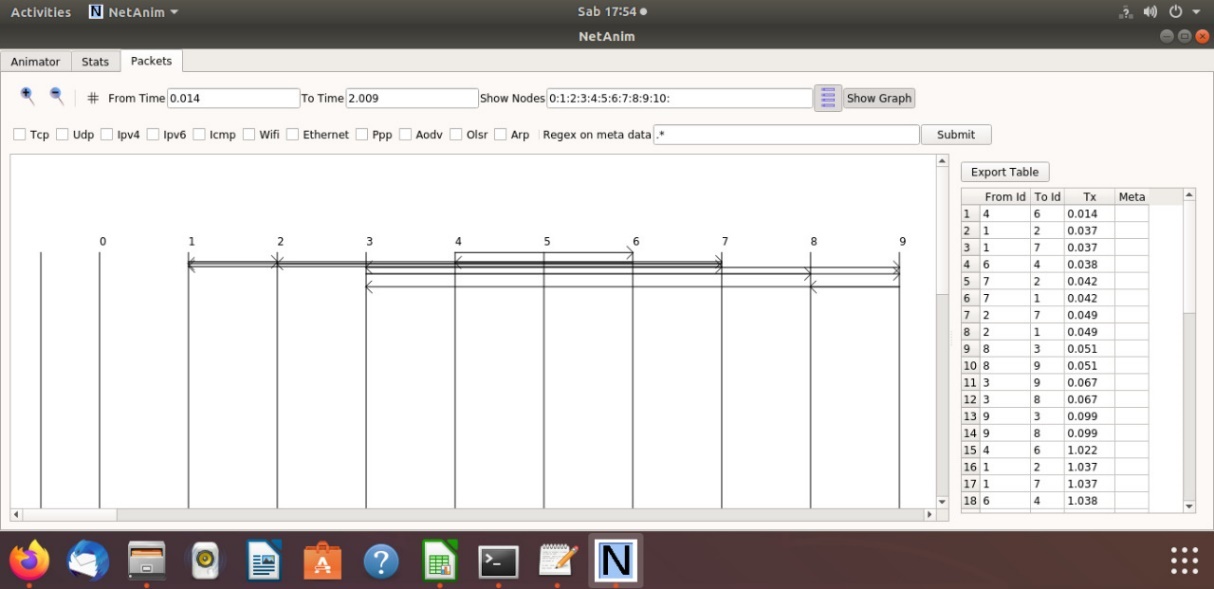
**Table 4.21 Hasil Iterasi 6 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 511.095081 | 1706.4488 |
| Posisi 2 | 0 | 1396.96068 |
| Posisi 3 | 44.492159 | 1416.02875 |
| Posisi 4 | 175.048772 | 1498.81761 |
| Posisi 5 | 329.830085 | 1583.19099 |
| Posisi 6 | 34.405413 | 1377.55711 |
| Posisi 7 | 406.366444 | 1539.53344 |
| Posisi 8 | 76.252839 | 1408.91785 |
| Posisi 9 | 234.754923 | 1568.87942 |
| Posisi 10 | 197.238723 | 1481.49156 |

Nilai yang sudah sesuai dengan skala NS3 ini kemudian akan digunakan pada tahap simulasi. Berikut merupakan gambaran posisi node yang ditampilkan dengan NetAnim:



**Gambar 4.14 Posisi Node Iterasi 6**



**Gambar 4.15 Komunikasi Node Iterasi 6**

Berdasarkan gambar tersebut dapat dilihat bahwa ada 8 node yang menerima pesa. Adapun 8 pesan tersebut yaitu node 1, node 2, node 3, node 4, node 6, node 7, node 8, dan node 9.

1. Posisi pada Iterasi 7

Berikut merupakan tabel nilai posisi node pada iterasi ke tujuh:

**Table 4.22 Hasil Iterasi 7**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -17.5700831 | -4.01179468 |
| Posisi 2 | -20.2717966 | -5.68791284 |
| Posisi 3 | -21.0423287 | -6.01814087 |
| Posisi 4 | -21.141052 | -5.88460808 |
| Posisi 5 | -19.4795564 | -5.1694739 |
| Posisi 6 | -21.806984 | -6.53287421 |
| Posisi 7 | -20.3662773 | -5.52788516 |
| Posisi 8 | -21.5006994 | -6.50544507 |
| Posisi 9 | -19.1890427 | -4.55324276 |
| Posisi 10 | -20.9180993 | -5.96489972 |

Nilai posisi node tersebut selanjutnya akan di normalkan terlebih dahulu. Berikut merupakan tabel nilai posisi node yang telah dinormalkan:

**Table 4.23 Hasil Iterasi 7 yang di Normalkan**

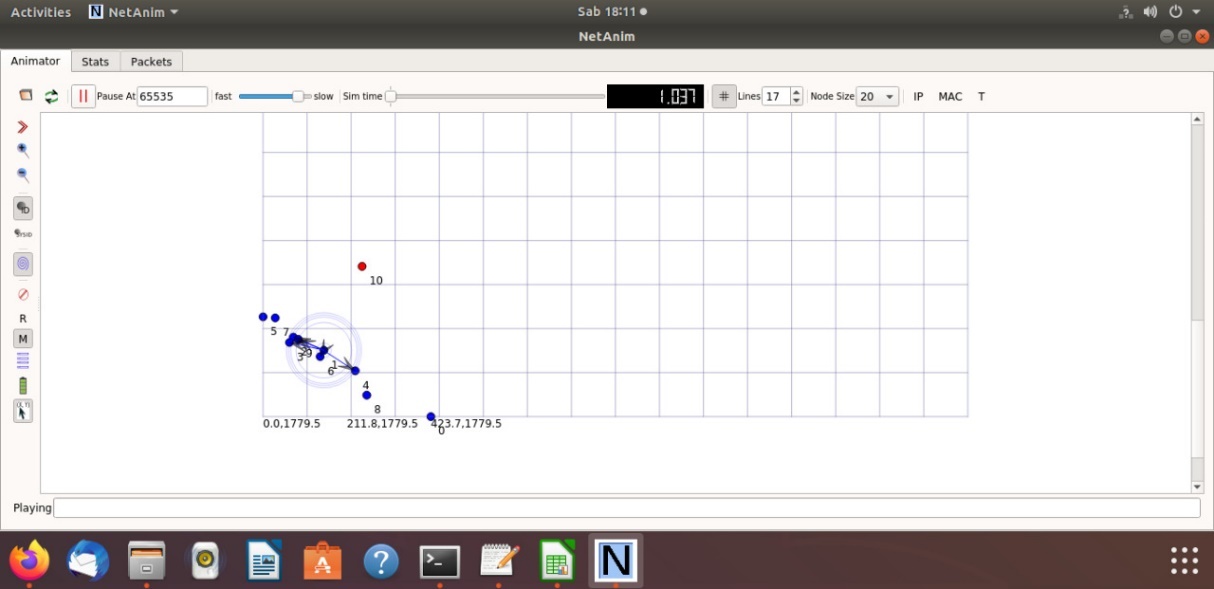
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 4.23690087 | 17.79518933 |
| Posisi 2 | 1.53518737 | 16.11907117 |
| Posisi 3 | 0.7646553 | 15.78884314 |
| Posisi 4 | 0.66593206 | 15.92237593 |
| Posisi 5 | 2.32742763 | 16.63751011 |
| Posisi 6 | 0 | 15.2741098 |
| Posisi 7 | 1.44070667 | 16.27909885 |
| Posisi 8 | 0.3062846 | 15.30153894 |
| Posisi 9 | 2.61794136 | 17.25374125 |
| Posisi 10 | 0.88888467 | 15.84208429 |

Setelah didapat nilai posisi node yang normal, maka selanjutnya nilai tersebut akan disesuaikan ke skala NS3 terlebih dahulu. Berikut merupakan nilai posisi node yang telah disesuaikan dengan skala NS3:

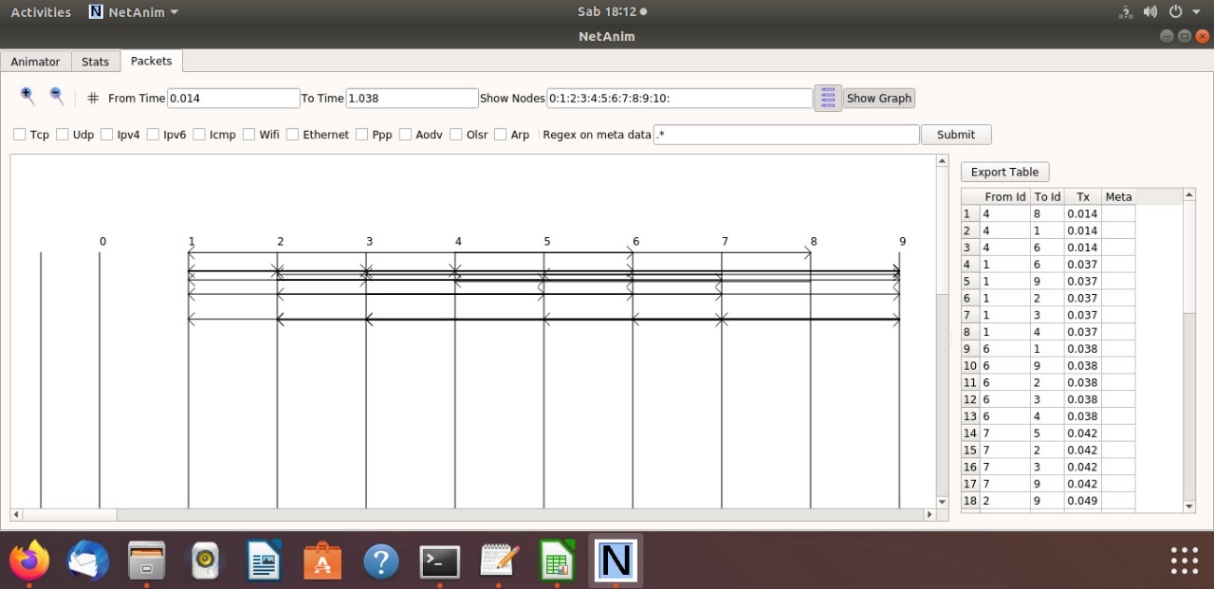
**Table 4.24 Hasil Iterasi 7 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 423.690087 | 1779.51893 |
| Posisi 2 | 153.518737 | 1611.90712 |
| Posisi 3 | 76.46553 | 1578.88431 |
| Posisi 4 | 66.593206 | 1592.23759 |
| Posisi 5 | 232.742763 | 1663.75101 |
| Posisi 6 | 0 | 1527.41098 |
| Posisi 7 | 144.070667 | 1627.90989 |
| Posisi 8 | 30.62846 | 1530.15389 |
| Posisi 9 | 261.794136 | 1725.37413 |
| Posisi 10 | 88.888467 | 1584.20843 |

Dari nilai tersebut kemudian akan digunakan dalam simulasi di NS3. Berikut merupakan gambaran posisi node yang di tampilkan dengan NetAnim:



**Gambar 4.16 Posisi Node Iterasi 7**



**Gambar 4.17 Komunikasi Node Iterasi 7**

Dari gambar tersebut terdapat 9 node yang menerima pesan, adapun node yang menerima pesan tersebut adalah dari node 1 sampai node 9.

1. Posisi pada Iterasi 8

Berikut merupakan tabel nilai posisi node pada iterasi ke delapan:

**Table 4.25 Hasil Iterasi 8**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -21.6844247 | -6.36307642 |
| Posisi 2 | -21.5474523 | -6.324549 |
| Posisi 3 | -22.4588027 | -6.69154628 |
| Posisi 4 | -23.2322762 | -6.95998545 |
| Posisi 5 | -22.7055264 | -6.91264072 |
| Posisi 6 | -23.2840971 | -7.08869092 |
| Posisi 7 | -23.8850046 | -7.02040737 |
| Posisi 8 | -23.2156731 | -7.22062884 |
| Posisi 9 | -21.2203507 | -5.72724123 |
| Posisi 10 | -23.1551931 | -7.00576526 |

Nilai tersebut kemudian akan di normalkan terlebih dahulu, sebelum digunakan di NS3. Berikut merupakan tabel nilai posisi node iterasi ke delapan yang telah di normalkan:

**Table 4.26 Hasil Iterasi 8 yang di normalkan**

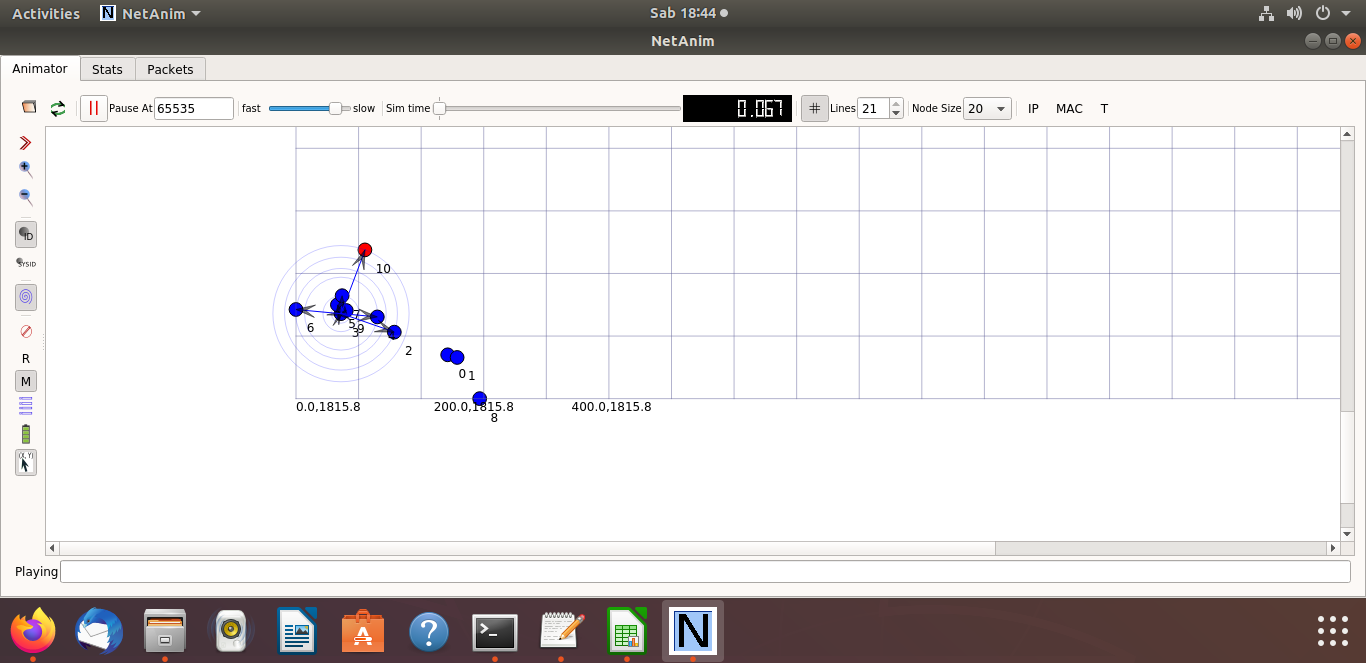
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 2.20057982 | 17.52192814 |
| Posisi 2 | 2.33755229 | 17.56045556 |
| Posisi 3 | 1.42620182 | 17.19345828 |
| Posisi 4 | 0.65272834 | 16.92501911 |
| Posisi 5 | 1.17947815 | 16.97236384 |
| Posisi 6 | 0.60090742 | 16.79631364 |
| Posisi 7 | 0 | 16.86459719 |
| Posisi 8 | 0.66933143 | 16.66437572 |
| Posisi 9 | 2.6646539 | 18.15776333 |
| Posisi 10 | 0.72981148 | 16.8792393 |

Setelah nilai posisi node nya normal maka kemudian akan disesuaikan dengan skala NS3 agar kemudian dapat di jalankan pada proses simulasi. Berikut merupakan tabel nilai posisi node yang sudah disesuaikan skalanya dengan NS3:

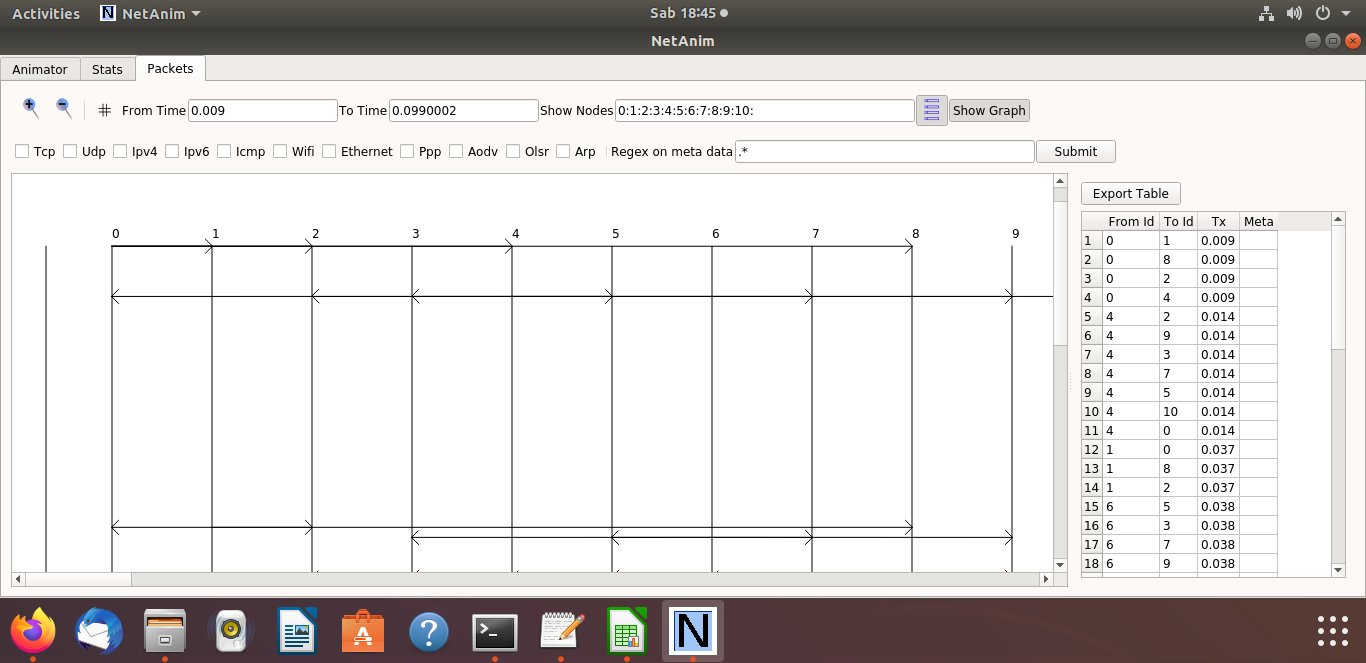
**Table 4.27 Hasil Iterasi 8 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 23.88500456 | 220.057982 |
| Posisi 2 | 23.88500456 | 233.755229 |
| Posisi 3 | 23.88500456 | 142.620182 |
| Posisi 4 | 23.88500456 | 65.272834 |
| Posisi 5 | 23.88500456 | 117.947815 |
| Posisi 6 | 23.88500456 | 60.090742 |
| Posisi 7 | 23.88500456 | 0 |
| Posisi 8 | 23.88500456 | 66.933143 |
| Posisi 9 | 23.88500456 | 266.46539 |
| Posisi 10 | 23.88500456 | 72.981148 |

Nilai yang sudah sesuai dengan skala NS3 ini kemudian akan digunakan pada tahap simulasi. Berikut merupakan gambaran posisi node yang ditampilkan dengan NetAnim:



**Gambar 4.18 Posisi Node Iterasi 8**



**Gambar 4.19 Komunikasi Node Iterasi 8**

Dari gambar tersebut dapat diketahui bahwa ada 9 node yang menerima pesan. Adapun 9 node tersebut yaitu node 0, node 1, node 2, node 3, node 4, node 5, node 7, node 8, dan node 9.

1. Posisi pada Iterasi 9

Berikut merupakan tabel posisi node pada iterasi ke sembilan:

**Table 4.28 Hasil Iterasi 9**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -24.0968406 | -7.64483187 |
| Posisi 2 | -22.9032325 | -6.85659201 |
| Posisi 3 | -23.5873478 | -7.12516582 |
| Posisi 4 | -24.2824261 | -7.49809418 |
| Posisi 5 | -24.615414 | -7.81135154 |
| Posisi 6 | -24.2565323 | -7.3400227 |
| Posisi 7 | -25.6443682 | -7.76666848 |
| Posisi 8 | -24.4936907 | -7.45242463 |
| Posisi 9 | -23.3502753 | -6.85500004 |
| Posisi 10 | -24.8273673 | -7.53730538 |

Nilai posisi node pada iterasi ke sembilan ini kemudian akan di normalkan dengan cara di Translasikan, berikut merupakan tabel posisi node yang sudah di normalkan:

**Table 4.29 Hasil Iterasi 9 yang di Normalkan**

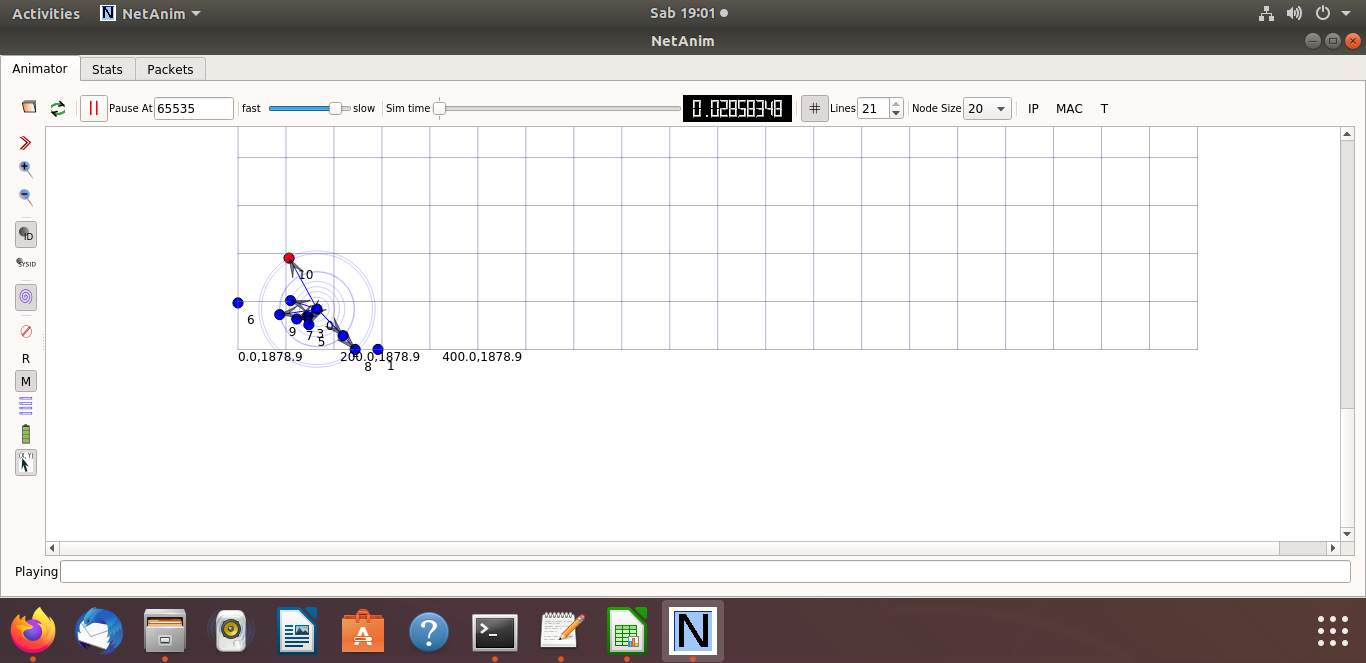
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 1.54752757 | 17.9995363 |
| Posisi 2 | 2.74113567 | 18.78777616 |
| Posisi 3 | 2.05702039 | 18.51920235 |
| Posisi 4 | 1.36194206 | 18.14627399 |
| Posisi 5 | 1.02895419 | 17.83301663 |
| Posisi 6 | 1.38783586 | 18.30434547 |
| Posisi 7 | 0 | 17.87769969 |
| Posisi 8 | 1.15067746 | 18.19194354 |
| Posisi 9 | 2.29409284 | 18.78936813 |
| Posisi 10 | 0.81700083 | 18.10706279 |

Dari posisi node yang sudah dinormalkan tersebut selanjutnya akan di sesuaikan dengan skala NS3, berikut merupakan tabel nilai posisi node yang telah sesuaikan skala nya:

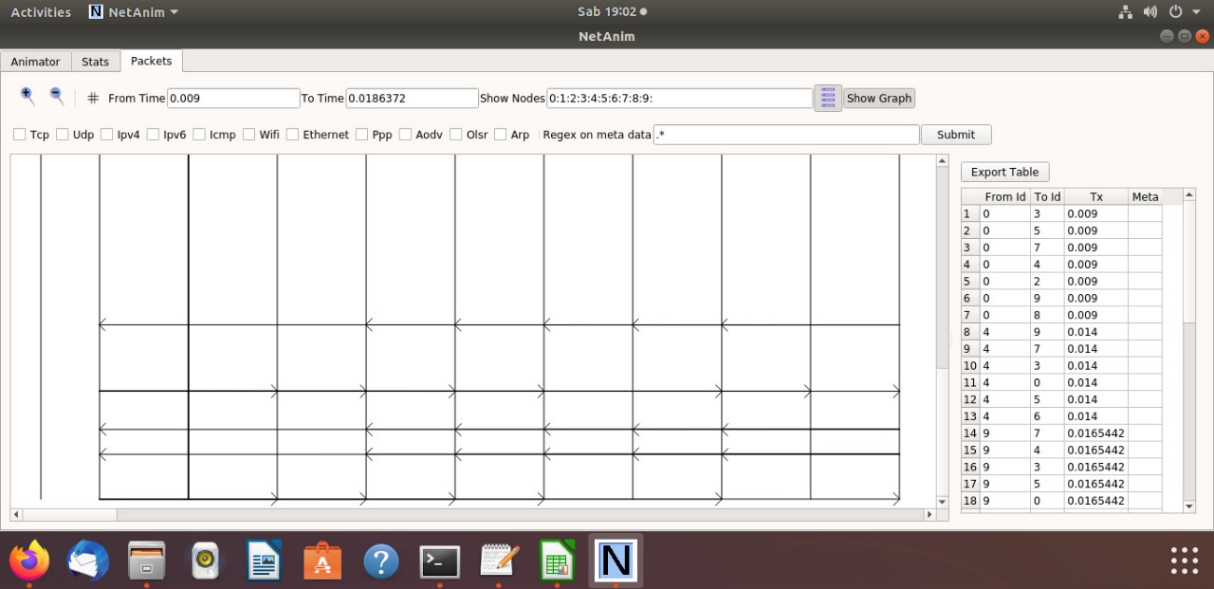
**Table 4.30 Hasil Iterasi 9 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 154.752757 | 1799.95363 |
| Posisi 2 | 274.113567 | 1878.77762 |
| Posisi 3 | 205.702039 | 1851.92024 |
| Posisi 4 | 136.194206 | 1814.6274 |
| Posisi 5 | 102.895419 | 1783.30166 |
| Posisi 6 | 138.783586 | 1830.43455 |
| Posisi 7 | 0 | 1787.76997 |
| Posisi 8 | 115.067746 | 1819.19435 |
| Posisi 9 | 229.409284 | 1878.93681 |
| Posisi 10 | 81.700083 | 1810.70628 |

Berdasarkan nilai posisi node yang sudah di sesuaikan dengan skala NS3 tersebut, kemudian simulasi akan dijalankan. Berikut merupakan gambaran posisi node setelah dijalankan di NetAnim.



**Gambar 4.20 Posisi Node Iterasi 9**



**Gambar 4.21 Komunikasi Node Iterasi 9**

Pada gambar tersebut dapat diketahui bahwa terdapat 9 node yang menerima pesan. Adapun 9 node tersebut yaitu node 0, node 2, node 3, node 4, node 5, node 6, node 7, node 8, dan node 9.

1. Posisi pada Iterasi 10

Berikut merupakan tabel nilai hasil iterasi terakhir:

**Table 4.31 Hasil Iterasi 10**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | -25.9432124 | -8.33610959 |
| Posisi 2 | -25.6328709 | -7.80380849 |
| Posisi 3 | -25.1476118 | -7.65258562 |
| Posisi 4 | -25.2920272 | -7.86269685 |
| Posisi 5 | -26.1845879 | -8.23403357 |
| Posisi 6 | -25.4345771 | -7.67836874 |
| Posisi 7 | -26.52405 | -8.13979903 |
| Posisi 8 | -25.4664399 | -7.65946522 |
| Posisi 9 | -25.1602583 | -7.71494931 |
| Posisi 10 | -25.7633906 | -7.83113131 |

Nilai hasil iterasi tersebut kemudian akan di normalkan terlebih dahulu, berikut merupakan nilai posisi node yang telah di normalkan:

**Table 4.32 Hasil Iterasi 10 yang di Normalkan**

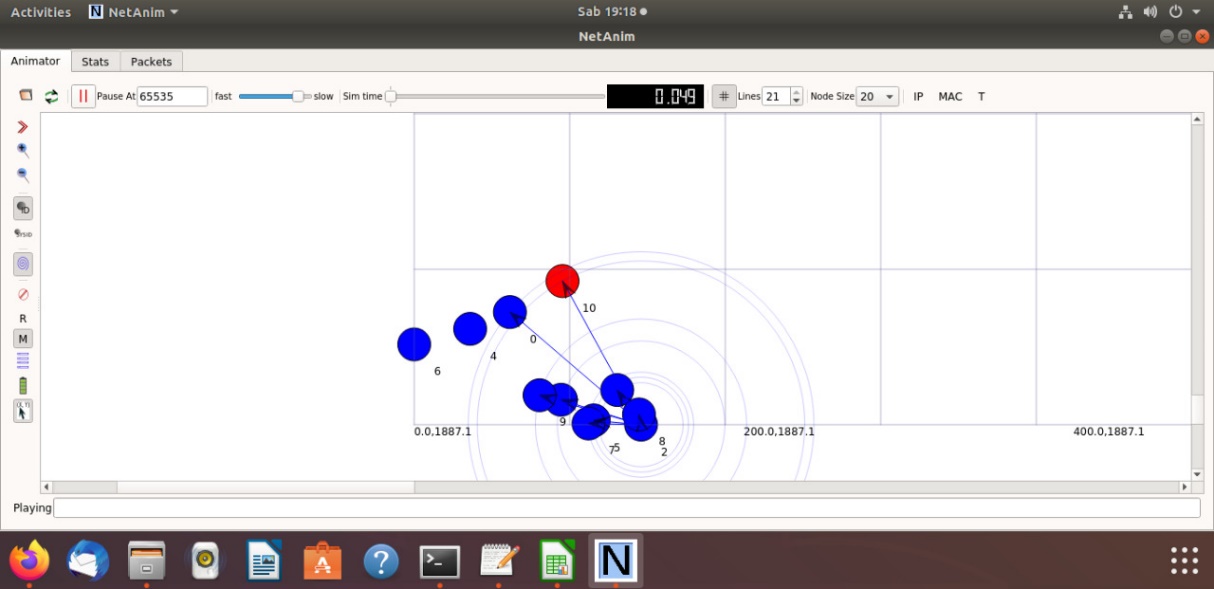
|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 0.58083761 | 18.18794038 |
| Posisi 2 | 0.89117906 | 18.72024148 |
| Posisi 3 | 1.37643818 | 18.87146435 |
| Posisi 4 | 1.23202281 | 18.66135312 |
| Posisi 5 | 0.33946203 | 18.2900164 |
| Posisi 6 | 1.08947291 | 18.84568123 |
| Posisi 7 | 0 | 18.38425094 |
| Posisi 8 | 1.05761012 | 18.86458475 |
| Posisi 9 | 1.36379171 | 18.80910066 |
| Posisi 10 | 0.76065933 | 18.69291866 |

Setelah nilai posisi node normal, maka selanjutnya akan disesuaikan terlebih dahulu dengan skala pada NS3 sebelum kemudian di simulasikan nantinya. Berikut merupakan tabel nilai posisi node yang telah di sesuaikan dengan skala NS3:

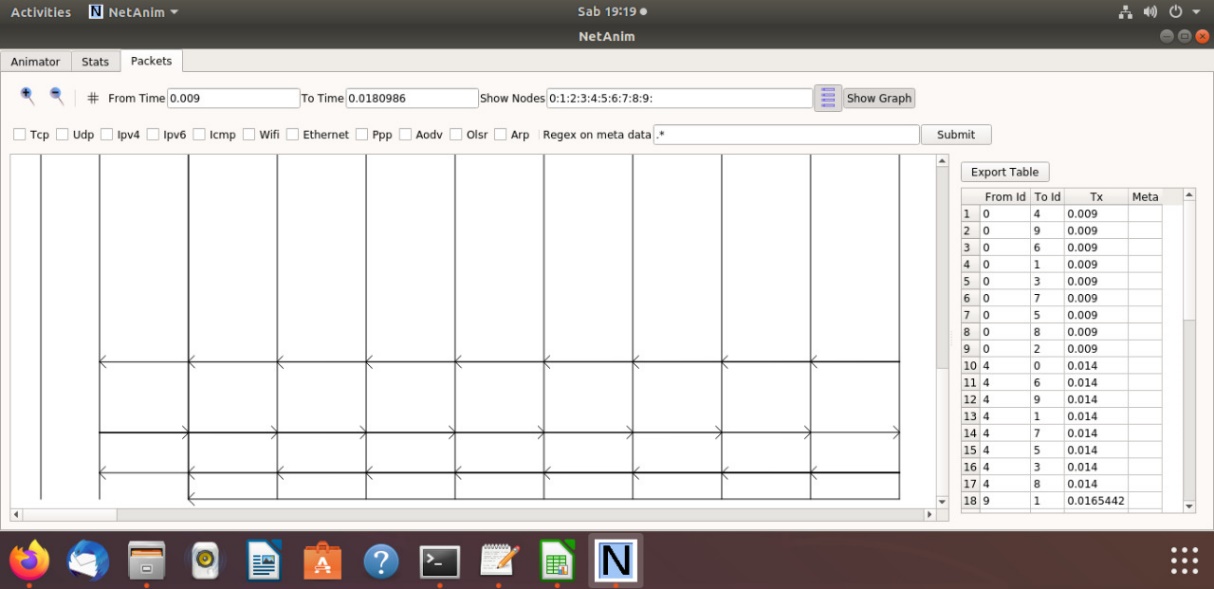
**Table 4.33 Hasil Iterasi 10 Penyesuaian NS3**

|  |  |  |
| --- | --- | --- |
|  | X | Y |
| Posisi 1 | 58.083761 | 1818.79404 |
| Posisi 2 | 89.117906 | 1872.02415 |
| Posisi 3 | 137.643818 | 1887.14644 |
| Posisi 4 | 123.202281 | 1866.13531 |
| Posisi 5 | 33.946203 | 1829.00164 |
| Posisi 6 | 108.947291 | 1884.56812 |
| Posisi 7 | 0 | 1838.42509 |
| Posisi 8 | 105.761012 | 1886.45848 |
| Posisi 9 | 136.379171 | 1880.91007 |
| Posisi 10 | 76.065933 | 1869.29187 |

Nilai yang telah disesuaikan itu selanjutnya akan dimasukkan ke program NS3 dan simulasi kemudian dijalankan. Berikut merupakan gambaran posisi node setalah di jalankan pada NetAnim:



**Gambar 4.22 Posisi Node Iterasi 10**

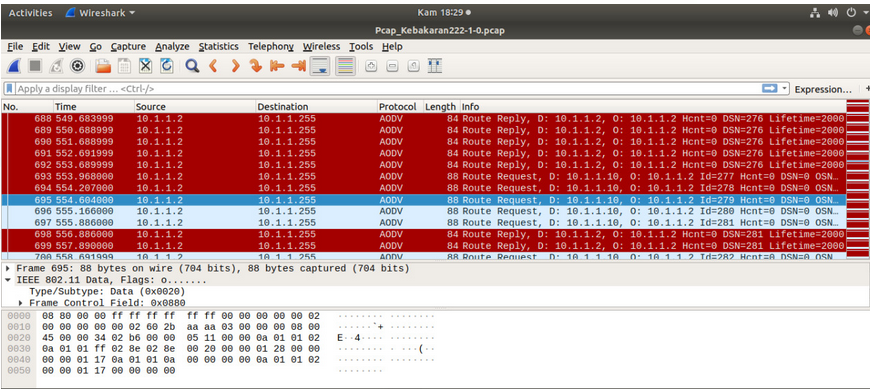


**Gambar 4.23 Komunikasi Node Iterasi 10**

Dari gambar tersebut dapat dilihat bahwa semua dari 10 node yang disiapkan menerima pesan. Berdasrkan hal tersebut maka dari sepuluh Iterasi yang dilakukan, maka dapat disimpulkan hasil pada iterasi ini dapat dianggap yang paling optimal.

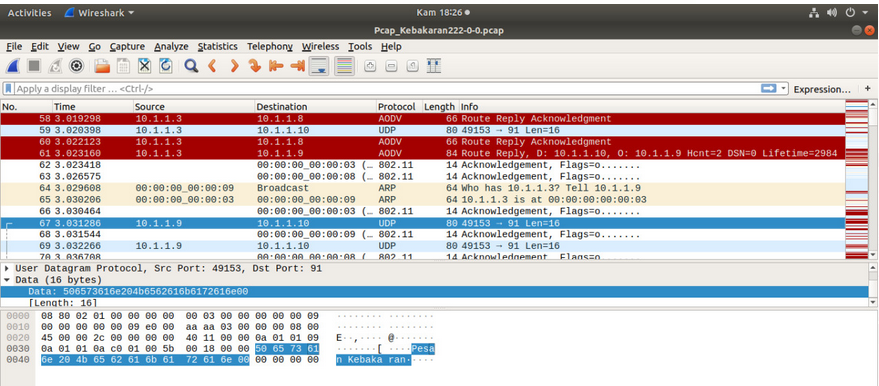
### **Hasil Pcap**

Setelah program selesai dieksekusi maka akan menghasilkan output berupa file .pcap, file tersebut merupakan file yang dibuat untuk analisis jaringan dengan *Wireshark.* Berikut merupakan tampilan file .pcap ketika di jalankan dengan *Wireshark.*



**Gambar 4.24 Tampilan File .pcap Node yang tidak menerima Pesan Kebakaran**

Gambar tesebut merupakan file pcap untuk node sensor yang diluar jangkauan node lainnya, sehingga node tersebut tidak menerima pesan apapun. Sedangkan untuk file .pcap node sensor yang berhasil berkomunikas dengan node lainnya ditunjukkan sebagai berikut.



**Gambar 4.25 Tampilan File .pcap Node yang menerima Pesan Kebakaran**

Dari kedua gambar tersebut maka dapat diketahui untuk node yang terhubung dengan node lainnya akan berkomunikasi dengan mengirimkan pesan teks berupa “Pesan Kebakaran”. Dalam simulasi ini akan beberapa file .pcap, sesuai dengan jumlah node yang dibuat.

### **Pengaruh Jumlah Iterasi terhadap Node yeng menerima pesan**

**Table 4.34 Node Yang Menerima Pesan Kebakaran**

|  |  |
| --- | --- |
| Iterasi ke- | Node yang Menerima Pesan |
| 0 | 2 |
| 1 | 8 |
| 2 | 4 |
| 3 | 7 |
| 4 | 6 |
| 5 | 7 |
| 6 | 8 |
| 7 | 9 |
| 8 | 9 |
| 9 | 9 |
| 10 | 10 |

**Gambar 4.26 Grafik Node Yang Menerima Pesan Kebakaran**

Dari grafik tersebut dapat diketahui bahwa jumlah iterasi memiliki pengaruh terhadap node yang menerima pesan kebakaran. Dalam penerapan pada simulasinya di NS3 jarak node yang terlalu jauh menjadi faktor utama yang mempengaruhi komunikasi node, semakin jauh jarak antar node maka semakin susah atau bahkan tidak dapat untuk berkomunikasi dengan node lainnya. Dari diagram tersebut dapat dilihat bahwa pada iterasi awal yaitu dari iterasi pertama sampai iterasi ke empat jumlah node yang menerima pesan mengalami kenaikan dan penurunan dengan cukup tacam. Sedangkan dari iterasi ke empat sampai iterasi terakhir, grafik terlihat mengalami peningkatan secara perlahan hingga pada iterasi ke sepuluh grafik sampai pada titik puncaknya.

# **BAB V**

**KESIMPULAN**

## **Kesimpulan**

Berdasarkan Penelitian dan Simulasi yang telah dilakukan, maka dapat ditarik kesimpulan sebagai berikut:

1. Proses Simulasi deployment node sensor dapat berjalan dengan baik di NS3 pada Sistem Operasi Linux Ubuntu 18.04. Simulasi dilakukan untuk mengetahui jumlah node sensor yang terhubung sehingga pengiriman pesan kebakaran dapat berjalan dengan baik. Visualisasi simulasi ditunjukkan dengan menggunakan aplikasi NetAnim, sedangkan detail pengiriman pesan kebakaran ditunjukkan melalui file .pcap.
2. Dalam tahap pengujiannya, masing-masing data hasil iterasi pada PSO akan disimulasikan sehingga dapat diketahui jumlah node yang terhubung dan yang tidak. Tingkat optimalnya akan dilihat dari seberapa banyak node yang terhubung dalam iterasi tersebut, semakin banyak node yang terhubung maka semakin optimal pula hasi pada iterasi tersebut.

## **Saran**

Adapun saran penulis untuk penelitian dan pengembangan lebih lanjut adalah sebagai berikut:

1. Gunakan aplikasi simulator yang lebih *advance* yang sudah mendukung fitur-fitur jaringan sensorik nirkabel beserta proses pengukuran performansinya yang lebih mendetail dan hasil yang didapat menjadi lebih optimal.
2. Tambahkan beberapa parameter pengujian lainnya selain jumlah node yang terhubung, agar hasil deployment yang didapat menjadi lebih optimal lagi.

# **DAFTAR PUSTAKA**

Christofer, G., Sujaini, H., & Irwansyah, M. (2015). Rancang Bangun Aplikasi Early Warning dengan pemanfaatan Pengukuran Suhu Ruangan Berbasis Arduino Mega 2560.

Marbun, Y., Nikentari, N., & Bettiza, M. (2015). Perbandingan Algoritma Genetika dan Particle Swarm Optimization dalam Optimasi Penjadwalan Matakuliah. *Jurnal Jurusan Teknik Informatika*.

Mubarok, F. (2019, November 16). Retrieved from [www.mongabay.co.id:https://www.mongabay.co.id/2019/11/16/kebakaran-hutan-dan-lahan-terus-terjadi-bagaimana-solusinya/](http://www.mongabay.co.id:https://www.mongabay.co.id/2019/11/16/kebakaran-hutan-dan-lahan-terus-terjadi-bagaimana-solusinya/)

Murdiansyah, D. T. (2016). Optimasi Jaringan Sensor Nirkabel Menggunakan Algoritma Two Sub-Swarm PSO Diskrit. *Journal On Computing*.

Rubiani, H., & Mulyatum, S. (2016). Deployment Jaringan Sensor Nirkabel Berdasarkan Cakupan Area Sensor Node Menggunakan Algoritma Particle Swarm Optimization. *Semnar Nasional Aplikasi Teknologi Informasi (SNATi).*

Santoso, B., & willy, P. (2011). *Metode Metaheuristik*, *Konsep dan Implementasi*. Surabaya: Graha Ilmu.

Suendri. (2017). Implementasi Algoritma Linear Congruentials Generator Untuk Menentukan Posisi Jabatan Kepanitiaan. *Jurnal Sistem Informasi*.

# **LAMPIRAN**

1. **Lampiran Code PSO**

|  |
| --- |
| import random as rd  import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  rd.seed(12)  #### NILAI AWAL  W = 0.5  c1 = 0.8  c2 = 0.9  lcg\_result = 0  lcg\_x = 0  lcg\_y = 0  #### PSO Properties  n\_iterations = 11  n\_particles = 10  target\_error =1e-6  def lcg(nilai):      a = 21      c = 3      m = 16      global lcg\_result      lcg\_result = (a\*lcg\_result + c) % m      return lcg\_result  global lcg\_c  global lcg\_x  global lcg\_y  lcg\_x = lcg(10)  lcg\_y = lcg(10)  lcg\_c = lcg(100)  class Particle():      def \_\_init\_\_(self):          x = lcg(10)          y = lcg(10)            self.position = np.array([x, y])          self.pBest\_position = self.position          self.pBest\_value = float('inf')          self.velocity = np.array([0,0])      def update(self):          self.position = self.position + self.velocity          print (f'position {i} :',self.position)          def get\_self\_position(self):        return self.position   class Space():        def \_\_init\_\_(self, target, target\_error, n\_particles):          self.target = target          self.target\_error = target\_error          self.n\_particles = n\_particles          self.particles = []          self.gBest\_value = float('inf')          self.gBest\_position = np.array([lcg(10), lcg(10)])        def fitness(self, particle):          x = particle.position[0]          y = particle.position[1]          f = x\*2 + y\*2 + 1          return f        def set\_pBest(self):            for particle in self.particles:              fitness\_candidate = self.fitness(particle)              if(particle.pBest\_value > fitness\_candidate):                  particle.pBest\_value = fitness\_candidate                  particle.pBest\_position = particle.position          def set\_gBest(self):          for particle in self.particles:              best\_fitness\_candidate = self.fitness(particle)              if(self.gBest\_value > best\_fitness\_candidate):                  self.gBest\_value = best\_fitness\_candidate                  self.gBest\_position = particle.position        def update\_particles(self):          confidence = lcg(100)          global i          i = 0          for particle in self.particles:              global W              i += 1              inertial = W \* particle.velocity              self\_confidence = c1 \* rd.random() \* (particle.pBest\_position - particle.position)              swarm\_confidence = c2 \* rd.random() \* (self.gBest\_position - particle.position)              new\_velocity = inertial + self\_confidence + swarm\_confidence              particle.velocity = new\_velocity              particle.update()        def show\_particles(self, iteration):          print(iteration, 'iterations')          print('BestPosition in this time:', self.gBest\_position)          print('BestValue in this time:', self.gBest\_value)          for particle in self.particles:              plt.plot(particle.position[0], particle.position[1], 'bo')          plt.plot(self.gBest\_position[0], self.gBest\_position[1], 'ro')          plt.show()    ##### MAIN PROGRAM  search\_space = Space(1, target\_error, n\_particles)  particle\_vector = [Particle() for \_ in range(search\_space.n\_particles)]  search\_space.particles = particle\_vector  iteration = 0  while(iteration < n\_iterations):      # set particle best & global best      search\_space.set\_pBest()      search\_space.set\_gBest()      # visualization      search\_space.show\_particles(iteration)        # check conditional      if(abs(search\_space.gBest\_value - search\_space.target) <= search\_space.target\_error):          break      search\_space.update\_particles()      iteration += 1    print("The best solution is: ", search\_space.gBest\_position) |

1. **Lampiran Code Simulasi**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | #include "ns3/core-module.h" | |  |  |  | | --- | | #include "ns3/network-module.h" | |  |  |  | | --- | | #include "ns3/mobility-module.h" | |  |  |  | | --- | | #include "ns3/config-store-module.h" | |  |  |  | | --- | | #include "ns3/wifi-module.h" | |  |  |  | | --- | | #include "ns3/energy-module.h" | |  |  |  | | --- | | #include "ns3/internet-module.h" | |  |  |  | | --- | | #include "ns3/olsr-helper.h" | |  |  |  | | --- | | #include "ns3/ipv4-static-routing-helper.h" | |  |  |  | | --- | | #include "ns3/ipv4-list-routing-helper.h" | |  |  |  | | --- | | //#include "ns3/config-store-module.h" | |  |  |  | | --- | | #include "ns3/netanim-module.h" | |  |  |  | | --- | | #include "ns3/animation-interface.h" | |  |  |  | | --- | | #include "ns3/applications-module.h" | |  |  |  | | --- | |  | |  |  |  | | --- | | #include <iostream> | |  |  |  | | --- | | #include <string> | |  |  |  | | --- | | #include <fstream> | |  |  |  | | --- | | #include <iomanip> | |  |  |  | | --- | |  | |  |  |  | | --- | | #include "ns3/core-module.h" | |  |  |  | | --- | | #include "ns3/network-module.h" | |  |  |  | | --- | | #include "ns3/internet-module.h" | |  |  |  | | --- | | #include "ns3/point-to-point-module.h" | |  |  |  | | --- | | #include "ns3/applications-module.h" | |  |  |  | | --- | | #include "ns3/netanim-module.h" | |  |  |  | | --- | | #include "ns3/animation-interface.h" | |  |  |  | | --- | | #include "ns3/packet-sink-helper.h" | |  |  |  | | --- | | #include "ns3/uinteger.h" | |  |  |  | | --- | | #include "ns3/mobility-module.h" | |  |  |  | | --- | | #include "ns3/olsr-helper.h" | |  |  |  | | --- | | #include "ns3/ipv4-static-routing-helper.h" | |  |  |  | | --- | | #include "ns3/ipv4-list-routing-helper.h" | |  |  |  | | --- | | #include "ns3/wifi-module.h" | |  |  |  | | --- | | #include "ns3/gnuplot.h" | |  |  |  | | --- | | #include "ns3/flow-monitor-module.h" | |  |  |  | | --- | | #include "ns3/aodv-module.h" | |  |  |  | | --- | | #include "ns3/config-store-module.h" | |  |  |  | | --- | | #include "ns3/wifi-mac.h" | |  |  |  | | --- | | #include "ns3/pointer.h" | |  |  |  | | --- | | #include "ns3/boolean.h" | |  |  |  | | --- | | #include "ns3/core-module.h" | |  |  |  | | --- | | #include "ns3/network-module.h" | |  |  |  | | --- | | #include "ns3/internet-module.h" | |  |  |  | | --- | | #include "ns3/point-to-point-module.h" | |  |  |  | | --- | | #include "ns3/applications-module.h" | |  |  |  | | --- | | #include "ns3/netanim-module.h" | |  |  |  | | --- | | #include "ns3/animation-interface.h" | |  |  |  | | --- | | #include "ns3/packet-sink-helper.h" | |  |  |  | | --- | | #include "ns3/uinteger.h" | |  |  |  | | --- | | #include "ns3/mobility-module.h" | |  |  |  | | --- | | #include "ns3/olsr-helper.h" | |  |  |  | | --- | | #include "ns3/ipv4-static-routing-helper.h" | |  |  |  | | --- | | #include "ns3/ipv4-list-routing-helper.h" | |  |  |  | | --- | | #include "ns3/wifi-module.h" | |  |  |  | | --- | | #include "ns3/gnuplot.h" | |  |  |  | | --- | | #include "ns3/flow-monitor-module.h" | |  |  |  | | --- | | #include "ns3/aodv-module.h" | |  |  |  | | --- | | #include "ns3/config-store-module.h" | |  |  |  | | --- | | #include "ns3/wifi-mac.h" | |  |  |  | | --- | | #include "ns3/pointer.h" | |  |  |  | | --- | | #include "ns3/boolean.h" | |  |  |  | | --- | |  | |  |  |  | | --- | | #include "assert.h" | |  |  |  | | --- | |  | |  |  |  | | --- | | #include <iostream> | |  |  |  | | --- | | using std::cout; | |  |  |  | | --- | | using std::cerr; | |  |  |  | | --- | | using std::endl; | |  |  |  | | --- | |  | |  |  |  | | --- | | #include <fstream> | |  |  |  | | --- | | #include <vector> | |  |  |  | | --- | | #include <string> | |  |  |  | | --- | | using std::string; | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | NS\_LOG\_COMPONENT\_DEFINE ("Suastika"); | |  |  |  | | --- | |  | |  |  |  | | --- | | using namespace ns3; | |  |  |  | | --- | |  | |  |  |  | | --- | | static inline std::string | |  |  |  | | --- | | PrintReceivedPacket (Address& from) | |  |  |  | | --- | | { | |  |  |  | | --- | | InetSocketAddress iaddr = InetSocketAddress::ConvertFrom (from); | |  |  |  | | --- | |  | |  |  |  | | --- | | std::ostringstream oss; | |  |  |  | | --- | | oss << "--\nReceived one packet! Socket: " << iaddr.GetIpv4 () | |  |  |  | | --- | | << " port: " << iaddr.GetPort () | |  |  |  | | --- | | << " at time = " << Simulator::Now ().GetSeconds () | |  |  |  | | --- | | << "\n--"; | |  |  |  | | --- | |  | |  |  |  | | --- | | return oss.str (); | |  |  |  | | --- | | } | |  |  |  | | --- | |  | |  |  |  | | --- | | double energyConstant = 0; | |  |  |  | | --- | | /\*\* | |  |  |  | | --- | | \* \param socket Pointer to socket. | |  |  |  | | --- | | \* | |  |  |  | | --- | | \* Packet receiving sink. | |  |  |  | | --- | | \*/ | |  |  |  | | --- | | void | |  |  |  | | --- | | ReceivePacket (Ptr<Socket> socket) | |  |  |  | | --- | | { | |  |  |  | | --- | | Ptr<Packet> packet; | |  |  |  | | --- | | Address from; | |  |  |  | | --- | | while ((packet = socket->RecvFrom (from))) | |  |  |  | | --- | | { | |  |  |  | | --- | | if (packet->GetSize () > 0) | |  |  |  | | --- | | { | |  |  |  | | --- | | NS\_LOG\_UNCOND (PrintReceivedPacket (from)); | |  |  |  | | --- | | } | |  |  |  | | --- | | } | |  |  |  | | --- | | } | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* | |  |  |  | | --- | |  | |  |  |  | | --- | | \* Traffic generator. | |  |  |  | | --- | | \*/ | |  |  |  | | --- | | static void | |  |  |  | | --- | | GenerateTraffic (Ptr<Socket> socket, uint32\_t pktSize, Ptr<Node> n, | |  |  |  | | --- | | uint32\_t pktCount, Time pktInterval) | |  |  |  | | --- | | { | |  |  |  | | --- | | if (pktCount > 0) | |  |  |  | | --- | | { | |  |  |  | | --- | | socket->Send (Create<Packet> (pktSize)); | |  |  |  | | --- | | Simulator::Schedule (pktInterval, &GenerateTraffic, socket, pktSize, n, | |  |  |  | | --- | | pktCount - 1, pktInterval); | |  |  |  | | --- | | } | |  |  |  | | --- | | else | |  |  |  | | --- | | { | |  |  |  | | --- | | socket->Close (); | |  |  |  | | --- | | } | |  |  |  | | --- | | } | |  |  |  | | --- | |  | |  |  |  | | --- | | /// Trace function for remaining energy at node. | |  |  |  | | --- | | void | |  |  |  | | --- | | RemainingEnergy (double oldValue, double remainingEnergy) | |  |  |  | | --- | | { | |  |  |  | | --- | | NS\_LOG\_UNCOND (Simulator::Now ().GetSeconds () | |  |  |  | | --- | | << "s Current remaining energy = " << remainingEnergy-(energyConstant\*Simulator::Now ().GetSeconds ()) << "J"); | |  |  |  | | --- | | } | |  |  |  | | --- | |  | |  |  |  | | --- | | /// Trace function for total energy consumption at node. | |  |  |  | | --- | | void | |  |  |  | | --- | | TotalEnergy (double oldValue, double totalEnergy) | |  |  |  | | --- | | { | |  |  |  | | --- | | NS\_LOG\_UNCOND (Simulator::Now ().GetSeconds () | |  |  |  | | --- | | << "s Total energy consumed by radio = " << totalEnergy+(energyConstant\*Simulator::Now ().GetSeconds ()) << "J"); | |  |  |  | | --- | | } | |  |  |  | | --- | |  | |  |  |  | | --- | | int | |  |  |  | | --- | | main (int argc, char \*argv[]) | |  |  |  | | --- | | { | |  |  |  | | --- | |  | |  |  |  | | --- | | std::string phyMode ("DsssRate2Mbps"); | |  |  |  | | --- | | double Prss = -80; // dBm | |  |  |  | | --- | | uint32\_t PpacketSize = 200; // bytes | |  |  |  | | --- | | bool verbose = false; | |  |  |  | | --- | |  | |  |  |  | | --- | | double distance = 100; | |  |  |  | | --- | | uint32\_t NodeFire = 1; | |  |  |  | | --- | |  | |  |  |  | | --- | | // bool tracing = false; | |  |  |  | | --- | |  | |  |  |  | | --- | | // simulation parameters | |  |  |  | | --- | | uint32\_t numPackets = 100; // number of packets to send | |  |  |  | | --- | | double interval = 1; // seconds | |  |  |  | | --- | | double startTime = 0.0; // seconds | |  |  |  | | --- | | double distanceToRx = 100.0; // meters | |  |  |  | | --- | | /\* | |  |  |  | | --- | | \* This is a magic number used to set the transmit power, based on other | |  |  |  | | --- | | \* configuration. | |  |  |  | | --- | | \*/ | |  |  |  | | --- | | double offset = 81; | |  |  |  | | --- | |  | |  |  |  | | --- | | CommandLine cmd; | |  |  |  | | --- | | cmd.AddValue ("phyMode", "Wifi Phy mode", phyMode); | |  |  |  | | --- | | cmd.AddValue ("Prss", "Intended primary RSS (dBm)", Prss); | |  |  |  | | --- | | cmd.AddValue ("PpacketSize", "size of application packet sent", PpacketSize); | |  |  |  | | --- | | cmd.AddValue ("numPackets", "Total number of packets to send", numPackets); | |  |  |  | | --- | | cmd.AddValue ("startTime", "Simulation start time", startTime); | |  |  |  | | --- | | cmd.AddValue ("distanceToRx", "X-Axis distance between nodes", distanceToRx); | |  |  |  | | --- | | cmd.AddValue ("verbose", "Turn on all device log components", verbose); | |  |  |  | | --- | | cmd.Parse (argc, argv); | |  |  |  | | --- | |  | |  |  |  | | --- | | // Convert to time object | |  |  |  | | --- | | Time interPacketInterval = Seconds (interval); | |  |  |  | | --- | |  | |  |  |  | | --- | | // disable fragmentation for frames below 2200 bytes | |  |  |  | | --- | | Config::SetDefault ("ns3::WifiRemoteStationManager::FragmentationThreshold", | |  |  |  | | --- | | StringValue ("2200")); | |  |  |  | | --- | | // turn off RTS/CTS for frames below 2200 bytes | |  |  |  | | --- | | Config::SetDefault ("ns3::WifiRemoteStationManager::RtsCtsThreshold", | |  |  |  | | --- | | StringValue ("2200")); | |  |  |  | | --- | | // Fix non-unicast data rate to be the same as that of unicast | |  |  |  | | --- | | Config::SetDefault ("ns3::WifiRemoteStationManager::NonUnicastMode", | |  |  |  | | --- | | StringValue (phyMode)); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | string plain = "Pesan Kebakaran"; | |  |  |  | | --- | | cout << "pesan: " << plain << endl; | |  |  |  | | --- | |  | |  |  |  | | --- | | NodeContainer c; | |  |  |  | | --- | | c.Create (10); // create 2 nodes | |  |  |  | | --- | | NodeContainer networkNodes; | |  |  |  | | --- | | networkNodes.Add (c.Get (0)); | |  |  |  | | --- | | networkNodes.Add (c.Get (1)); | |  |  |  | | --- | | networkNodes.Add (c.Get (2)); | |  |  |  | | --- | | networkNodes.Add (c.Get (3)); | |  |  |  | | --- | | networkNodes.Add (c.Get (4)); | |  |  |  | | --- | | networkNodes.Add (c.Get (5)); | |  |  |  | | --- | | networkNodes.Add (c.Get (6)); | |  |  |  | | --- | | networkNodes.Add (c.Get (7)); | |  |  |  | | --- | | networkNodes.Add (c.Get (8)); | |  |  |  | | --- | | networkNodes.Add (c.Get (9)); | |  |  |  | | --- | |  | |  |  |  | | --- | | NodeContainer f; | |  |  |  | | --- | | f.Create(NodeFire); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | // The below set of helpers will help us to put together the wifi NICs we want | |  |  |  | | --- | | WifiHelper wifi; | |  |  |  | | --- | | if (verbose) | |  |  |  | | --- | | { | |  |  |  | | --- | | wifi.EnableLogComponents (); | |  |  |  | | --- | | } | |  |  |  | | --- | | wifi.SetStandard (WIFI\_PHY\_STANDARD\_80211b); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* Wifi PHY \*\*/ | |  |  |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |  |  |  | | --- | | YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default (); | |  |  |  | | --- | | wifiPhy.Set ("RxGain", DoubleValue (-10)); | |  |  |  | | --- | | // wifiPhy.Set ("TxGain", DoubleValue (offset + Prss)); | |  |  |  | | --- | | wifiPhy.Set ("TxGain", DoubleValue (offset + Prss-15.0)); | |  |  |  | | --- | | wifiPhy.Set ("CcaMode1Threshold", DoubleValue (0.0)); | |  |  |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* wifi channel \*\*/ | |  |  |  | | --- | | YansWifiChannelHelper wifiChannel; | |  |  |  | | --- | | wifiChannel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel"); | |  |  |  | | --- | | wifiChannel.AddPropagationLoss ("ns3::FriisPropagationLossModel"); | |  |  |  | | --- | | // create wifi channel | |  |  |  | | --- | | Ptr<YansWifiChannel> wifiChannelPtr = wifiChannel.Create (); | |  |  |  | | --- | | wifiPhy.SetChannel (wifiChannelPtr); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* MAC layer \*\*/ | |  |  |  | | --- | | // Add a non-QoS upper MAC, and disable rate control | |  |  |  | | --- | | NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default (); | |  |  |  | | --- | | wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager", "DataMode", | |  |  |  | | --- | | StringValue (phyMode), "ControlMode", | |  |  |  | | --- | | StringValue (phyMode)); | |  |  |  | | --- | | // Set it to ad-hoc mode | |  |  |  | | --- | | wifiMac.SetType ("ns3::AdhocWifiMac"); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* install PHY + MAC \*\*/ | |  |  |  | | --- | | NetDeviceContainer devices = wifi.Install (wifiPhy, wifiMac, networkNodes); | |  |  |  | | --- | | NetDeviceContainer fakeDevices = wifi.Install(wifiPhy, wifiMac, f); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | MobilityHelper mobility; | |  |  |  | | --- | | mobility.SetPositionAllocator ("ns3::GridPositionAllocator", | |  |  |  | | --- | | "MinX", DoubleValue (0.0), | |  |  |  | | --- | | "MinY", DoubleValue (0.0), | |  |  |  | | --- | | "DeltaX", DoubleValue (distance), | |  |  |  | | --- | | "DeltaY", DoubleValue (distance), | |  |  |  | | --- | | "GridWidth", UintegerValue (5), | |  |  |  | | --- | | "LayoutType", StringValue ("RowFirst")); | |  |  |  | | --- | | mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel"); | |  |  |  | | --- | | mobility.Install (c); | |  |  |  | | --- | | mobility.Install(f); | |  |  |  | | --- | |  | |  |  |  | | --- | | // Enable OLSR | |  |  |  | | --- | | AodvHelper aodv; | |  |  |  | | --- | | Ipv4StaticRoutingHelper staticRouting; | |  |  |  | | --- | |  | |  |  |  | | --- | | Ipv4ListRoutingHelper list; | |  |  |  | | --- | | list.Add (staticRouting, 0); | |  |  |  | | --- | | list.Add (aodv, 10); | |  |  |  | | --- | | InternetStackHelper internet; | |  |  |  | | --- | | internet.SetRoutingHelper (list); | |  |  |  | | --- | | internet.Install (networkNodes); | |  |  |  | | --- | |  | |  |  |  | | --- | | InternetStackHelper aodvInternet; | |  |  |  | | --- | | aodvInternet.SetRoutingHelper(aodv); | |  |  |  | | --- | | aodvInternet.Install(f); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* Energy Model \*\*/ | |  |  |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |  |  |  | | --- | | /\* energy source \*/ | |  |  |  | | --- | | BasicEnergySourceHelper basicSourceHelper; | |  |  |  | | --- | | // configure energy source | |  |  |  | | --- | | basicSourceHelper.Set ("BasicEnergySourceInitialEnergyJ", DoubleValue (1500.0)); | |  |  |  | | --- | | // install source | |  |  |  | | --- | | EnergySourceContainer sources = basicSourceHelper.Install (c); | |  |  |  | | --- | | /\* device energy model \*/ | |  |  |  | | --- | | WifiRadioEnergyModelHelper radioEnergyHelper; | |  |  |  | | --- | | // configure radio energy model | |  |  |  | | --- | | radioEnergyHelper.Set ("TxCurrentA", DoubleValue (0.00000)); | |  |  |  | | --- | | // install device model | |  |  |  | | --- | | DeviceEnergyModelContainer deviceModels = radioEnergyHelper.Install (devices, sources); | |  |  |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* Internet stack \*\*/ | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | Ipv4AddressHelper ipv4; | |  |  |  | | --- | | NS\_LOG\_INFO ("Assign IP Addresses."); | |  |  |  | | --- | | ipv4.SetBase ("10.1.1.0", "255.255.255.0"); | |  |  |  | | --- | | Ipv4InterfaceContainer i = ipv4.Assign (devices); | |  |  |  | | --- | | Ipv4InterfaceContainer j = ipv4.Assign(fakeDevices); | |  |  |  | | --- | | UdpEchoServerHelper echoServer(91); | |  |  |  | | --- | |  | |  |  |  | | --- | | TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory"); | |  |  |  | | --- | | Ptr<Socket> recvSink = Socket::CreateSocket (networkNodes.Get (9), tid); // node 14, receiver | |  |  |  | | --- | | InetSocketAddress local = InetSocketAddress (Ipv4Address::GetAny (), 80); | |  |  |  | | --- | | recvSink->Bind (local); | |  |  |  | | --- | | recvSink->SetRecvCallback (MakeCallback (&ReceivePacket)); | |  |  |  | | --- | |  | |  |  |  | | --- | | Ptr<Socket> source = Socket::CreateSocket (networkNodes.Get (0), tid); // node 0, sender | |  |  |  | | --- | | InetSocketAddress remote = InetSocketAddress (i.GetAddress(9,0), 80); | |  |  |  | | --- | | source->SetAllowBroadcast (true); | |  |  |  | | --- | | source->Connect (remote); | |  |  |  | | --- | |  | |  |  |  | | --- | | //------------------------------------------------------------------------- | |  |  |  | | --- | | ApplicationContainer serverApps = echoServer.Install(c.Get(9)); | |  |  |  | | --- | | serverApps.Start(Seconds(1.0)); | |  |  |  | | --- | | serverApps.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | UdpEchoClientHelper echoClient(i.GetAddress(9), 91); | |  |  |  | | --- | | echoClient.SetAttribute("MaxPackets", UintegerValue(1000)); | |  |  |  | | --- | | echoClient.SetAttribute("Interval", TimeValue(Seconds(2.0))); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_0 = echoClient.Install(c.Get(0)); | |  |  |  | | --- | | client\_app\_0.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_0.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_1 = echoClient.Install(c.Get(1)); | |  |  |  | | --- | | client\_app\_1.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_1.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_2 = echoClient.Install(c.Get(2)); | |  |  |  | | --- | | client\_app\_2.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_2.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_3 = echoClient.Install(c.Get(3)); | |  |  |  | | --- | | client\_app\_3.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_3.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_4 = echoClient.Install(c.Get(4)); | |  |  |  | | --- | | client\_app\_4.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_4.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_5 = echoClient.Install(c.Get(5)); | |  |  |  | | --- | | client\_app\_5.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_5.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_6 = echoClient.Install(c.Get(6)); | |  |  |  | | --- | | client\_app\_6.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_6.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_7 = echoClient.Install(c.Get(7)); | |  |  |  | | --- | | client\_app\_7.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_7.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_8 = echoClient.Install(c.Get(8)); | |  |  |  | | --- | | client\_app\_8.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_8.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer client\_app\_9 = echoClient.Install(c.Get(9)); | |  |  |  | | --- | | client\_app\_9.Start(Seconds(2.0)); | |  |  |  | | --- | | client\_app\_9.Stop(Seconds(1500.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | | //----------------connecting NodeFire to network------------------- | |  |  |  | | --- | | UdpEchoClientHelper fakeClient(i.GetAddress(4), 91); | |  |  |  | | --- | | fakeClient.SetAttribute("MaxPackets", UintegerValue(1000)); | |  |  |  | | --- | | fakeClient.SetAttribute("Interval", TimeValue(Seconds(2.0))); | |  |  |  | | --- | |  | |  |  |  | | --- | | ApplicationContainer fake\_app\_0 = fakeClient.Install(f.Get(0)); | |  |  |  | | --- | | fake\_app\_0.Start(Seconds(2.0)); | |  |  |  | | --- | | fake\_app\_0.Stop(Seconds(200.0)); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | Ipv4GlobalRoutingHelper::PopulateRoutingTables(); | |  |  |  | | --- | |  | |  |  |  | | --- | | //-implement encryption- | |  |  |  | | --- | | echoClient.SetFill(client\_app\_0.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_1.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_2.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_3.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_4.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_5.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_6.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_7.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_8.Get(0), plain); | |  |  |  | | --- | | echoClient.SetFill(client\_app\_9.Get(0), plain); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*-----------------------------------------------\*/ | |  |  |  | | --- | | fakeClient.SetFill(fake\_app\_0.Get(0), "Pesan kebakaran"); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* connect trace sources \*\*/ | |  |  |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |  |  |  | | --- | | // all sources are connected to node (1) | |  |  |  | | --- | | // energy source | |  |  |  | | --- | | Ptr<BasicEnergySource> basicSourcePtr = DynamicCast<BasicEnergySource> (sources.Get (1)); | |  |  |  | | --- | | basicSourcePtr->TraceConnectWithoutContext ("RemainingEnergy", MakeCallback (&RemainingEnergy)); | |  |  |  | | --- | |  | |  |  |  | | --- | | // device energy model | |  |  |  | | --- | | Ptr<DeviceEnergyModel> basicRadioModelPtr = | |  |  |  | | --- | | basicSourcePtr->FindDeviceEnergyModels ("ns3::WifiRadioEnergyModel").Get (0); | |  |  |  | | --- | | NS\_ASSERT (basicRadioModelPtr != NULL); | |  |  |  | | --- | | basicRadioModelPtr->TraceConnectWithoutContext ("TotalEnergyConsumption", MakeCallback (&TotalEnergy)); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |  |  |  | | --- | |  | |  |  |  | | --- | | //Output | |  |  |  | | --- | | AsciiTraceHelper ascii; | |  |  |  | | --- | | wifiPhy.EnableAsciiAll(ascii.CreateFileStream("wireless\_no\_encrypt10.tr")); | |  |  |  | | --- | | wifiPhy.EnablePcap("Pcap\_Kebakaran222", devices); | |  |  |  | | --- | |  | |  |  |  | | --- | | Ptr<OutputStreamWrapper> routingStream = Create<OutputStreamWrapper>("rute", std::ios::out); | |  |  |  | | --- | | //olsr.PrintRoutingTableAllEvery(Seconds(2.0),routingStream); | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*Animation setup\*/ | |  |  |  | | --- | | AnimationInterface anm("SimulasiKebakaran.xml"); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*Setting titik di NetAnim\*/ | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(0),244.964849, 218.34505); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(1),697.327746, 997.327746); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(2),662.720957, 432.544191); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(3),324.824104, 624.824104); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(4),642.164131, 575.213199); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(5),000.0, 300.0); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(6),120.790137, 903.950687); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(7),73.849727, 373.849727); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(8),157.837131, 247.387623); | |  |  |  | | --- | | anm.SetConstantPosition(c.Get(9),305.399237, 605.399237); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | //------------------------------------------------- | |  |  |  | | --- | |  | |  |  |  | | --- | | anm.SetConstantPosition(f.Get(0), 300.0, 300.0); | |  |  |  | | --- | | //anm.SetConstantPosition(f.Get(0), 300.0, 300.0); | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | | /\*\* simulation setup \*\*/ | |  |  |  | | --- | | // start traffic | |  |  |  | | --- | | Simulator::Schedule (Seconds (startTime), &GenerateTraffic, source, PpacketSize, | |  |  |  | | --- | | networkNodes.Get (0), numPackets, interPacketInterval); | |  |  |  | | --- | |  | |  |  |  | | --- | | Simulator::Stop (Seconds (1500.0)); | |  |  |  | | --- | | Simulator::Run (); | |  |  |  | | --- | | Simulator::Destroy (); | |  |  |  | | --- | |  | |  |  |  | | --- | | return 0; | |  |   } |