UNIVERSIDADE ESTADUAL DE CAMPINAS
Instituto de Computação

Avaliação do Consumo de Energia em Rede WSN em Ambiente Simulado

MO809A Tópicos de Computação Distribuída

Autor:
Eduardo Seiti Ito

Professor:
Leandro Villas

Campinas - SP
Junho de 2019

Sumário

1 Introdução

Em Redes WSN (Wireless Sensor Network) se requer uma duração prolongada das baterias de sensores de forma que há um interesse em entender o comportamento do consumo de energia dos mesmos.

De acordo com o estudo realizado por Transviña-Moreno et al [2], o consumo de energia em dispositivos WiFi se deve à máquina de estado finito (FSM Finite State Machine) que descreve os três estados de um nó sensor.

- Estado de Inicialização: Inicialização do hardware (aquecimento do oscilador e periféricos, inicialização de variáveis, etc.)
- Estado ativo. Tarefa de sensoreamento como por exemplo transmissão de dados. Enquanto em operação, o nó sensor habilita o conversor AD periférico e o desabilita na conclusão da mesma. No caso da difusão da mensagem, o módulo de rádio necessitará se juntar à rede, reportar os dados e fechar a conexão de energia para evitar o consumo desnecessário de energia.
- Estado de Baixa Potência. O módulo de rádio será desligado, o microcontrolador será mantido em modo de dormência e todos os periféricos serão desabilitados ou ligados à um estado de menor nível de consumo.

A figura abaixo ilustra o estado FSM:

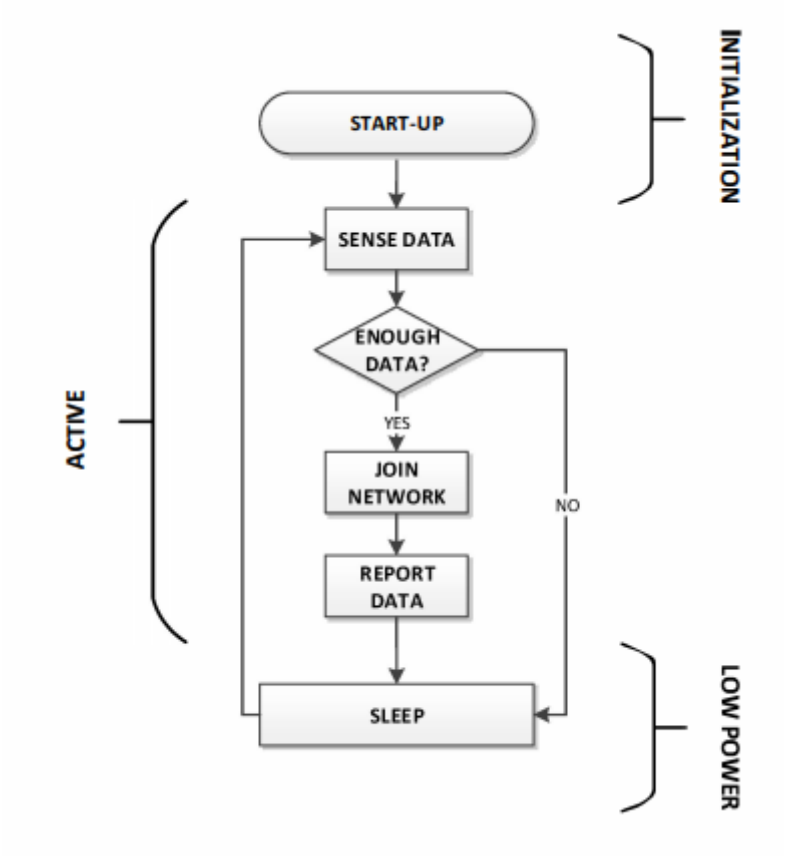

Figura 1: Máquina de Estado Finito (FSM)

O objetivo deste trabalho é verificar por meio de ambiente simulado o comportamento da energia em um nó sensor além dos estados FSM. Se há outros fatores que podem também influenciar o consumo de energia tais como o distância entre os sensores, versões de protocolo, vazão (throughput), etc. Perda de pacotes seria outro item a ser investigado. Finalmente, concluir com alguma percepção sobre a saída dos simuladores e alguns resultados de análise dos logs. /newline /newline É importante chegar àlguma conclusão sobre o consumo de energia, pois sempre será requerido dimensionamento da duração da bateria em projetos WSN. Sensores normalmente operam de forma autônoma, contando apenas com a bateria, eventualmente com uma suplementação de carga com um coletor externo, e.g. placas fotovoltaicas.

Este documento está dividido na seção da metodologia utilizada, análise dos resultados e uma conclusão final.

2 Metodologia

Nesta seção apresentamos os principais conceitos e técnicas que serão abordadas para a realização deste Trabalho Prático.

Todo o material produzido, shell script, programa python e .cc estão armazenados no GitHub[1]

Os simuladores mais conhecidos são os seguintes: NS3[2], OMNET++[3], WSNET[4], SENSE[5]

Será priorizado neste trabalho prático, os simuladores NS3 e OMNET++, pelo fato de já terem sido apreciado em classe, como mostrados nos seguintes subcapítulos.

[1]https://github.com/edbkei/mo809a-trabalhopratico/
[2]https://www.nsnam.org/
[3]https://omnetpp.org/
[4]http://wsnet.gforge.inria.fr/
[5]https://www.ita.cs.rpi.edu/

2.1 NS3

O módulo de energia do NS3 é baseado no Energy Model[6], que é o resultado do modelo de energia proposto por Taparello et all [1], a figura abaixo ilustra o diagrama de energia.

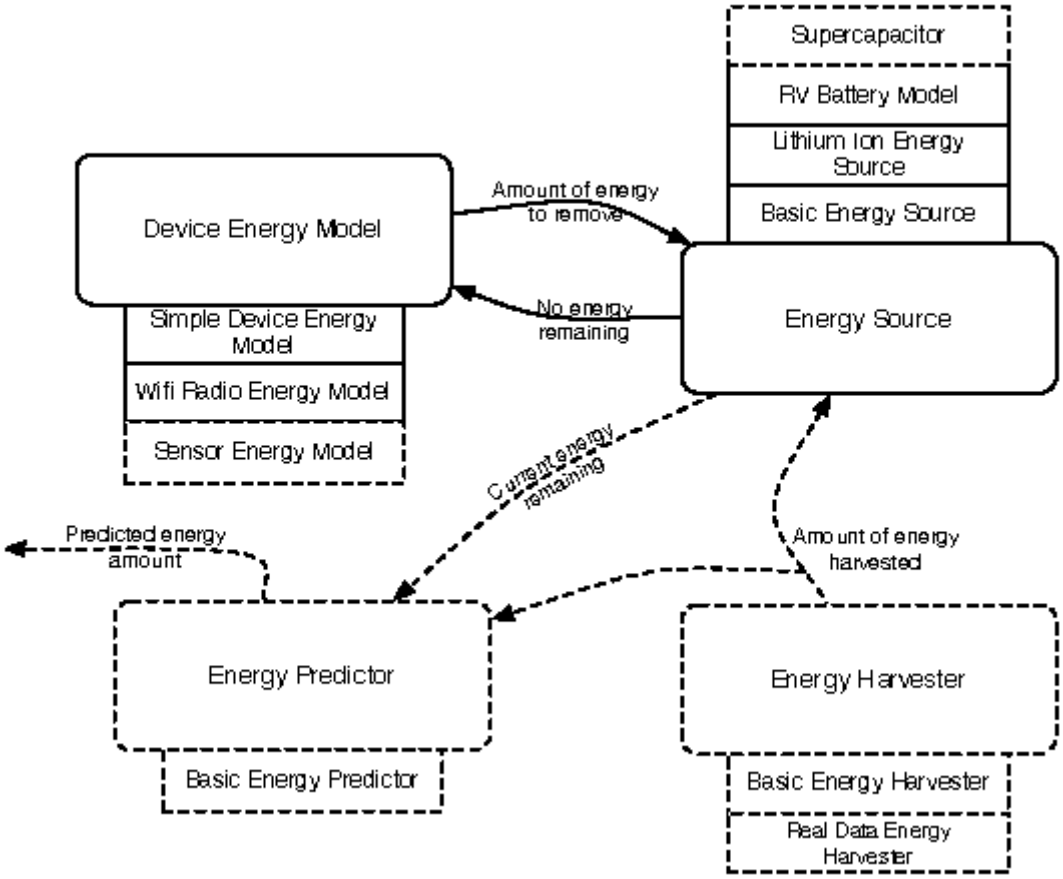

Figura 2: Modelo de Energia no NS3.

À cada dispositivo WSN uma fonte de suprimento de energia (Energy Source) é alocado com uma certa carga inicial, em Joules, para durar um certo período de operação. O Energy Source será um objeto da classe Basic Energy Source. O modelo de energia Energy Model será um objeto da classe Wifi Radio Energy Model. Uma alimentação adicional (Energy Harvester) pode ser acoplada ao Energy Source. Um objeto energy harvester será baseado na classeBasic Energy Harvester. O harvester dá mais autonomia para a operação de um dispositivo WSN, mas a principal fonte de energia é proveniente da carga inicial. Um exemplo prático de harvester seria uma placa de energia foto-voltaica. Por fim, vamos utilizar um exemplo dado na instalação do NS3, o energy-model-with-harvesting-example.cc quando o harvester é utilizado e o outro exemplo é o wireless-animation.cc que utiliza o utilitário NetAnim para animação da rede.

A instalação do NS3 foi realizada em uma máquina virtual Ubuntu 18.04.2 LTS e os programas de energia estão dispostas da seguinte forma:

[6]https://www.nsnam.org -¿ Documents -¿ development-tree -¿ Doxygen -¿ Modules -¿ Energy Models

- Diretório /repos/ns-3-allinone/ns-3-dev/scratch
  - runtrab.sh, shell script para rodar de forma interativa o trabprat.cc e o programa stat-trab.py. Este script permite a escolha do número de interações da distância, que é dado em metros. Um parâmetro de entrada é o tempo de simulação em segundos. Este script contém o comando waf para rodar e compilar o programa trabprat.cc.
  - trabprat.cc, programa em C++, baseado no exemplo dado do NS3 energy-model-with-harvesting-example.cc. Programa principal para geração do log de energia (em Joules), baseado em distâncias (metros). Outras características do sensor podem ser definidas, como por exemplo: o protocolo Wifi IEEE 802.11a ou b (no entanto, há outras versões como o h, n), número de nós, adição do coletor adicional de energia (harvester).
  - stattrab.py, script em Python para obtenção da estatística baseados nos logs gerados pelo trabprat.cc. Basicamente a entrada do log com pacotes enviados, recebidos, energia consumida pelo nó são plotados em um gráfico. Exemplo de um log do NS3 é mostrado no GitHub.

O programa trabprat.cc vai simular 2 nós Wifi, onde o nó de origem (Tx) envia pacotes UDP para o nó de destino. Como cada nó se conecta à um módulo de energia e este à um coletor suplementar de energia, de forma que será suficiente a coleta de log da energia consumida de apenas um nó, e definiremos o nó Rx, como o coletor do log de energia. Isto é realizado através da sintaxe DynamicCast¡BasicEnergySource¿(sources.Get(1)) e TraceconnectWithoutContext. Há outros comandos para conectar o coletor suplementar (harvester).

Depois de uma investigação preliminar, foram observadas que o consumo do dispositivo WiFi, estão situadas por volta de 0.82 J/s. Portanto uma observação de duração de 100 s, necessitaria de pelo menos uma carga inicial de 82 J, mas vamos considerar 100 J definido no parâmetro BasicEnergySourceInitialEnergyJ. Se não for feito este dimensionamento, haverá perda de pacotes por falta de energia durante à observação. No entanto, há uma suplementação de energia importante proveniente do harvester, que dá mais autonomia para um sensor. Isto pode ser observado executando o script test1.sh.
newline
newline Quaisquer mudanças no comportamento do sensor, tais como mudança no padrão IEEE802.11x, distância até o Rx, tempo de simulação, etc devem ser realizados por parâmetros externos ou por mudança no próprio código .cc. Nem tudo está parametrizado, caso haja necessidade, pode ser efetuado pelo comando cmd.AddValue. Os items abaixo descrevem o que são os parâmetros variáveis e fixos.

- Parâmetros variáveis utilizados:
  - simTime, tempo de simulação em segundos.
  - distanceToRx, distância até o Rx em metros.
  - outros parâmetros existentes eventualmente podem ser utilizados. phyMode (Wifi Physical Mode), Prss (Intended primary RSS (dBm)), PacketSize (tamanho do pacote), numPackets (número de pacotes para serem enviados), startTime (momento para o início da simulação)

- Parâmetros fixos utilizados (modificação apenas no código fonte .cc)
  - Número de nós. Comando create().
  - modificação do padrão Wifi. SetStandard().
  - posicionamento do dispositivo Wifi. positionAlloc.
  - Corrente em Ampere do Tx e Rx. radioEnergyhelper.Set().
  - Mudança do número IP. ipv4.SetBase().

Para avaliar o comportamento de energia no nó sensor Rx, serão propostos os seguintes casos de teste.

- TC1. Verificar o nível de consumo (em Joules) variando a distância até o Rx (em metros). Pré-condição: Valor padrão para o IEEE802.11b tais como phyMode DsssRate1Mbs, Prss -80 dBm, PacketSize 200, RxGain -10 dBm, TxGain -80 (Prss)+81(offset). Pode ser utilizado diretamente o script trabprat.sh com parâmetros de distância até o Rx, número de interações e tempo de simulação. Ou pode ser utilizado o test4.sh com edição desses parâmetros para o uso do próprio trabprat.sh. A distância até o Rx vai variar desde 100m, 200m, até 1400m, onde deve ocorrer a maior perda de pacotes.

- TC2. Verificar o nível de consumo (em Joules) variando a distância até o Rx (em metros). O mesmo que TC1, mas aumentando o taxa de transmissão, com DsssRate11Mbs. Seria o mesmo que TC1 com phyMode=DsssRate11Mbs, i.e. tx de 11 Mbps.

- TC3. Verificar o nível de consumo (em Joules) variando a distância até o Rx (em metros). Utilizando o padrão IEEE802.11a phyMode=OfdmRate12Mbps. Versões mais antigas do padrão WiFi, pode ter uma diferença considerável no consumo de energia. Configuração pode ser encontrada na referência do ns3 [7]
  Foram utilizados os scripts test5.sh, trabpratb.cc, runtrabb.sh e stattrab.py, que estão armazenados no GitHub.

- TC4. Verificar o nível de consumo (em Joules) baseado no tempo de simulação. A intenção é verificar se o coletor suplementar de energia (harvester) contribui com uma carga razoavelmente boa em relação ao consumo do nó Rx. Será utilizado basicamente o script test1.sh com uma duração de 120 segundos e distância de 10 metros fixo. Foi utilizado o padrão Wifi IEEE802.11b.

- TC5. Verificar o nível de consumo (em Joules) baseado no tempo na distância até Tx, NS3/NetAnim. A intenção é verificar o comportamento de energia conforme a variação posiçionamento do nó Rx, desta vez utilizando o aplicação NetAnim, o animador de rede, para isso será utilizado como base de código fonte, o programa wireless-animation.cc, disponibilizado como exemplo na instalação do NS3. O programa runtrabanim.cc foi baseado nele. Com algumas modificações, como a introdução do parâmetro de distância (distanceToRx) e tempo de simulação (simTime) e o número de nós Wifi configurado para 1 nó STA (Station), 1 AP (Access Point), e 1 nó CSMA(Ethernet LAN). O nó Wifi será o nó Rx, receptor das mensagens e atrelado à um módulo de suprimento de energia.

[7] https://www.nsnam.org/bugzilla/attachment.cgi?id=2230

2.2 OMNETPP

A utilização do OMNETPP será proposto para verificar a existência de algum contraditório ou alguma convergência em relação aos resultados obtidos pelo NS3.

O módulo de energia do OMNETPP será baseado no inet-framework[8].

Antes de se utilizar o OMNETPP, é importante reparar um problema de configuração ou de software encontrado. A versão do OMNETPP v5.4.1 não consegue ser inicializado na versão do OS Ubuntu 18.04.2 LTS por conta de um bug encontrado que o impede de selecionar o workspace. Um workaround foi encontrado e registrado no arquivo config.ini que está armazenado no GitHub.

A instalação do NS3 foi realizada em uma máquina virtual Ubuntu 18.04.2 LTS e os programas de energia estão dispostas da seguinte forma:

- Diretório inet/examples/wireless/power
  - omnetpp.ini, arquivo de configuração onde os parâmetros mais importante são o apontamento do arquivo .ned (a rede WSN em questão), o tempo de simulação (aqui definiremos como 100 segundos como foi feito no NS3), número de hosts (será definido como sendo 2 nós, como definido no NS3), parâmetros de energia: modelo de armazenamento de energia nos dispositivos energyStorage SimpleEpEnergyStorage, e consumo de energia energyConsumer StateBasedEpEnergyConsumer, energia no modo repouso sleepPowerConsumption 0.1mW, consumo de energia no modo livre na recepção receiverIdlePowerConsumption 2mW, no modo ocupado receiverBusyPowerConsumption 5mW, consumo na recepção receiverReceivingPowerConsumption 10mW, consumo de potência na transmissão no modo livre transmitterIdlePowerConsumption 2mW, consumo de energia na transmissão transmitterTransmitterPowerConsumption 10mW, modelo de gerenciamento energyManagement SimpleEpEnergyManagement, capacidade de armazenamento nominal energyStorage.nominalCapacity 0.05J, capacidade no desligamento do nó energyManagement.nodeShutdownCapacity 0J, capacidade do nó na inicialização energyManagement.nodeStartCapacity 0.025J, capacidade inicial de cada dispositivo energyStorage.initialCapacity um valor aleatório entre 0J e sua capacidade nominal de 0.05J, modelo de reposição de energia energyGenerator AlternatingEpEnergyGenerator, geração de energia powerGeneration 1mW. Área de posiçionamento dos nós restrita a uma área com dimensão 2000m x 2000 para ficarem sob uma mesma área de cobertura. O tipo de mobilidade será a mobilidade estacionária StationaryMobility, a distância entre os nós será realizada entre um nó receptor Rx sempre situado na posição X=0 e Y=0, um nó transmissor Tx situado numa posição variável (X, Y), cuja distância será variada entre 100m, 200m, ..., 1400m, como realizadas no NS3.
  - PowerNetwork.ned, arquivo de elementos da rede WSN, não modificado, usado como dado no exemplo.
  - General.anf, arquivo em XML que devido a saída do OMNETPP. O valor escalar no arquivo General-*,sca e o valor vetorial definido em General-*.vec,

[8]https://github.com/inet-framework/inet-showcases/tree/master/wireless/power

eventualmente se necessitar um arquivo log da simulação em General-*.elog (mas não será utilizado). O arquivo General.anf pode ser aberto no módulo gráfico para extração de gráfico e dados da simulação. Por meio deste, após a simulação se extrai o consumo total de energia, a média e o desvio padrão. Pelo console do OMNETPP se obtém os pacotes enviados, recebidos e os perdidos.

A visualização da rede WSN é realizada por meio da execução do arquivo de configuração omnetpp.ini, conforme mostrado abaixo.

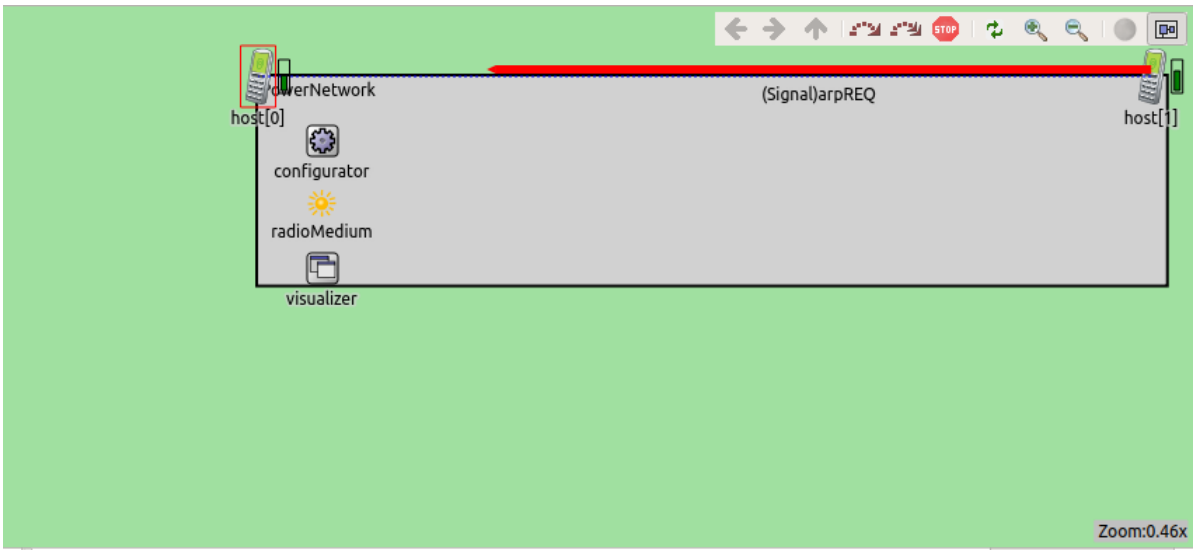

Figura 3: Visualização da Rede no OMNETPP

A visualização mostra que o nó Tx (host[1]) envia inicialmente uma mensagem de reconhecimento (beacon) para o nó Rx (host[0]) e espera uma resposta. Isto é representado pelo sinal ARP req e uma resposta ARP reply é retornada quando há um reconhecimento por parte de Rx e quando o nó Tx recebê-lo, o mesmo confirma com WLAN ack. Quando uma distância muito grande é definida, o Tx envia várias mensagens ARQ req, tentando reconhecimento, mas as mesmas podem não atingir o nó Rx, ou possivelmente a potência de chegada pode ser inferior à potência de sensibilidade (Rx sensibility).

Depois de estabelecida a conexão entre Tx e Rx, o nó Tx envia sinais ping, como forma para testar o envio e recepção de pacotes, para Rx. E espera reconhecimento. Isto é realizado quando Tx envia sinal ping para Rx, este por sua vez reconhece o sinal e o mesmo retorna os sinais WLAN Ack e ping reply em sequência de volta para Tx. Este após receber estes sinais, devolve WLAN Ack para Rx.

Portanto, aproveitaremos dessa configuração para variar a distância entre os nós e faremos observações em cima do nó Rx. Contaremos o número de pacotes recebidos e enviados por Rx, e também observaremos os dados de energia como consumo total, média e desvio padrão, de modo a possibilitar uma melhor visualização do comportamento da energia ao longo da distância entre os nós.

Para avaliar o comportamento de energia no nó sensor Rx, serão propostos os seguintes casos de teste.

- TC6. Verificar o nível de consumo (em Joules) variando a distância até o Rx (em metros). Similar a TC1. A distância até o Rx vai variar desde 100m, 200m, até 1400m nas mesmas condição de TC1. A variação deverá ocorrer apenas no arquivo omnetpp.ini no nó Tx nos parâmetros *.host[1].mobility.initialX e *.host[1].mobility.initialY. Nada mais será modificado no nó Rx, ou seja na configuração de host[0]. A coleta dos dados será realizada manualmente para um arquivo .csv à cada simulação, como não há opção para iteração por meio da variação da distância.

- TC7. Verificar o comportamento de consumo de energia entre os nós Rx e Tx durante o tempo de simulação. Será apenas realizado uma visualização dos dados gerado no modo gráfico do arquivo General.anf.

3 Análise dos Resultados

3.1 TC1, Variação da distância até o Rx com Wifi padrão IEEE802.11b

Os dados de energia na simulação com variação de distância até o Rx podem ser observados na tabela abaixo.

Tabela de Energia WiFi IEEE802.11b				
simTime	distanceToRx	pktReceived	pktSent	totalEnergy
100	100	100	100	81.7252
100	200	100	100	81.7252
100	300	100	100	81.7252
100	400	100	100	81.7252
100	500	100	100	81.7252
100	600	100	100	81.7252
100	700	99	100	81.7252
100	800	75	100	81.7252
100	900	16	100	81.7252
100	100	0	100	81.7252
100	1100	0	100	81.7252
100	1200	0	100	81.9
100	1300	0	100	81.9
100	1400	0	100	81.9

Observe que há pouca variação do consumo de energia do nó Rx nos 100 segundos de simulação com variação da distância. Apenas os valores 81.7252J e 81.9J.

O teste hipótese

$$Ho : \mu = 81.7252$$

utilizando o teste de t-student seria aceito para nível de significância $\alpha < 5\%$ pois o pvalue = 0.0823 (ou 8.23%) e com estimador t-student = 1.883, de acordo com o pacote scipy.stats do Python. O valor 81.9 seria rejeitado pois o pvalue=0.

Foi observado que a versão ns-3.29 instalado com bake, só retornava valores 81.9J para os mesmos casos da tabela acima. Com a instalação manual da mesma versão ns-3.29 utilizando o Mercurial, vem outros valores como o 81.7252J e também com 81.9J.

O teste foi realizado como mostrado abaixo:

```
In [42]: x = np.array([81.7252, 81.7252,  81.7252, 81.7252,  81.7252, 81.7252, 81.7252, 81.7252, 81.7252,
                       81.7252, 81.7252, 81.9, 81.9, 81.9])
         print ('t-student = %6.3f pvalue = %6.4f' %  stats.ttest_1samp(x, 81.7252))
         print ('t-student = %6.3f pvalue = %6.4f' %  stats.ttest_1samp(x, 81.9))

         t-student =  1.883 pvalue = 0.0823
         t-student = -6.904 pvalue = 0.0000
```

Figura 4: t-Student Test

Foi observado que não há variação de valores na tabela de energia, considerando 10 simulações nas mesmas condições. De forma que não há como plotar os gráfico com errorbar do Python. Os valores estão num arquivo chamado dataStorageJune19.zip no GitHub.

O gráfico de energia fica da seguinte forma:

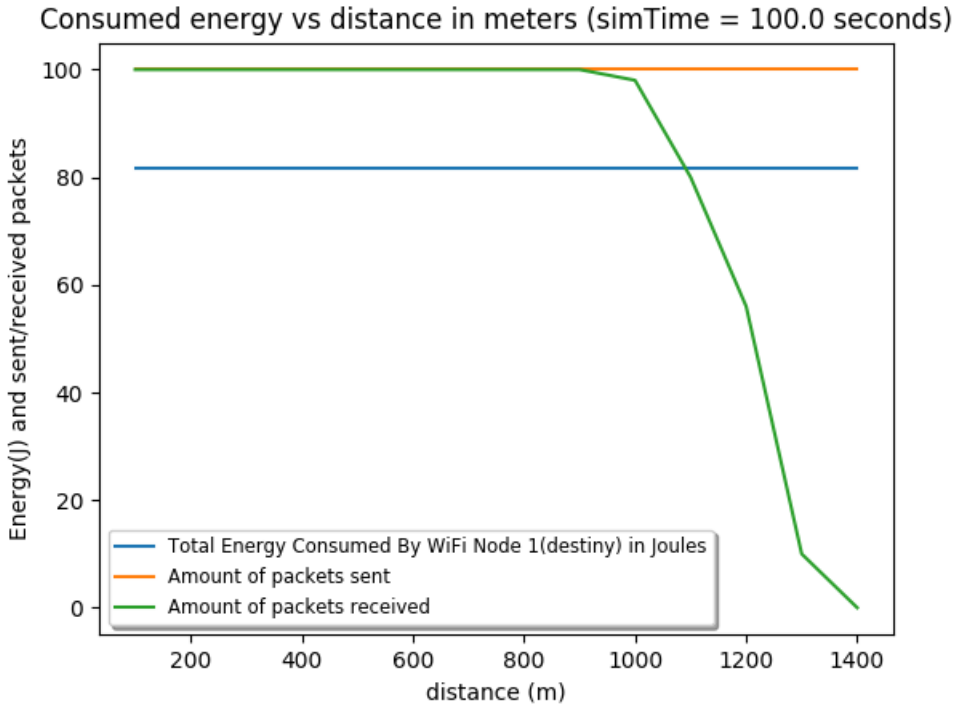

Figura 5: Gráfico de energia vs distância até o Rx, IEEE802.11b (DsssRate1Mbs)

A cada passo de 100m com tempo de simulação de 100s, não há diferença de consumo de energia do nó Rx, sempre a 81.7252J. O que se percebe é a perda de pacotes com a distância a partir de 800m, possivelmente devido à degradação do sinal Wifi e a potência na recepção seja menor do que é estabelecido pelo Rx Sensibility, no nó de destino.

3.2 TC2, Variação da distância até o Rx com Wifi padrão IEEE802.11b com phyMode=DsssRate11Mbs

Os dados de energia na simulação com variação de distância até o Rx podem ser observados na tabela abaixo.

Tabela de Energia WiFi IEEE802.11b, phyMode=DsssRate11Mbs				
simTime	distanceToRx	pktReceived	pktSent	totalEnergy
100	100	100	100	81.8711
100	200	100	100	81.8711
100	300	99	100	81.8711
100	400	13	100	81.8711
100	500	0	100	81.8711
100	600	0	100	81.8711
100	700	0	100	81.8711
100	800	0	100	81.8711
100	900	0	100	81.8711
100	100	0	100	81.8711
100	1100	0	100	81.8711
100	1200	0	100	81.9
100	1300	0	100	81.9
100	1400	0	100	81.9

O teste hipótese

$$Ho : \mu = 81.8711$$

utilizando o teste de t-student seria aceito para alfa menor que 5 porcento porque t-student = 1.883 pvalue = 0.0823, de acordo com o pacote scipy.stats do Python. O valor 81.9 seria rejeitado com pvalue=0.

O teste foi realizado como mostrado abaixo:

```
In [43]: x = np.array([81.8711, 81.8711,  81.8711, 81.8711,  81.8711, 81.8711, 81.8711, 81.8711, 81.8711,
                       81.8711, 81.8711, 81.9, 81.9, 81.9])
         print ('t-student = %6.3f pvalue = %6.4f' %  stats.ttest_1samp(x, 81.8711))
         print ('t-student = %6.3f pvalue = %6.4f' %  stats.ttest_1samp(x, 81.9))

         t-student =  1.883 pvalue = 0.0823
         t-student = -6.904 pvalue = 0.0000
```

Figura 6: t-Student Test

Mas se for comparar as médias de TC1 e deste TC2, a hipótese

$$Ho : \mu 1 = \mu 2$$

seria rejeitada com 1 porcento de chance porque pvalue=5.562873858838586e-06. O teste foi realizado como mostrado abaixo:

```
IEEE80122bDsss1 = np.array([81.7252, 81.7252,  81.7252, 81.7252,  81.7252, 81.7252, 81.7252, 81.7252, 81.7252,
             81.7252, 81.7252, 81.9, 81.9, 81.9])
IEEE80122bDsss11 = np.array([81.8711, 81.8711,  81.8711, 81.8711,  81.8711, 81.8711, 81.8711, 81.8711, 81.8711,
             81.8711, 81.8711, 81.9, 81.9, 81.9])
#print ('t-student = %6.3f pvalue = %6.4f' %  stats.ttest_1samp(IEEE80122bDsss1, 81.8711))
#print ('t-student = %6.3f pvalue = %6.4f' %  stats.ttest_1samp(IEEE80122bDsss1, 81.9))
stats.ttest_ind(IEEE80122bDsss1, IEEE80122bDsss11)
```

```
Ttest_indResult(statistic=-5.68545591014042, pvalue=5.562873858838586e-06)
```

Figura 7: t-Student Test Two Samples

O gráfico de energia ficaria da seguinte forma:

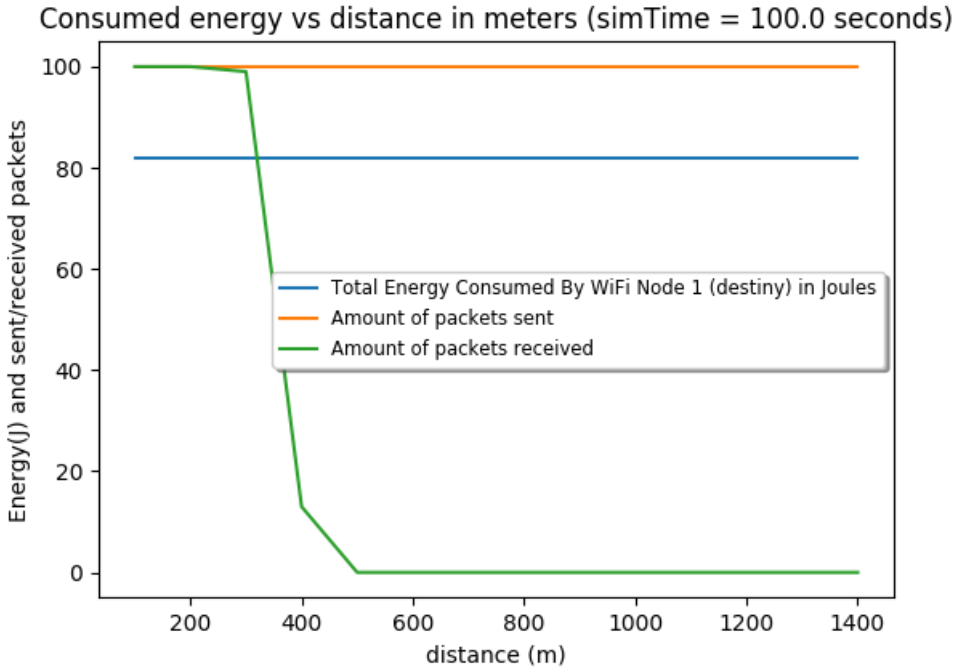

Figura 8: Gráfico de energia vs distância até o Rx, IEEE802.11b (DsssRate11Mbs)

A cada passo de 100m com tempo de simulação de 100s, não há diferença de consumo de energia do nó Rx, sempre a 81.8711J. O que se percebe é a perda de pacotes com a distância a partir de 200m, possivelmente devido à degradação do sinal Wifi e a potência na recepção seja menor do que é estabelecido pelo Rx Sensibility, no nó de destino.

3.3 TC3, Variação da distância até o Rx com Wifi padrão IEEE802.11a com phyMode=OfdmRate12Mbps

Os dados de energia na simulação com variação de distância até o Rx podem ser observados na tabela abaixo.

Tabela de Energia WiFi IEEE802.11a, phyMode=OfdmRate12Mbps				
simTime	distanceToRx	pktReceived	pktSent	totalEnergy
100	100	0	100	81.8851
100	200	0	100	81.9
100	300	0	100	81.9
100	400	0	100	81.9
100	500	0	100	81.9
100	600	0	100	81.9
100	700	0	100	81.9
100	800	0	100	81.9
100	900	0	100	81.9
100	100	0	100	81.9
100	1100	0	100	81.9
100	1200	0	100	81.9
100	1300	0	100	81.9
100	1400	0	100	81.9

A hipótese nula de que a média é 81.8851 é rejeitada com pValue=0, de acordo com a aplicação stats.ttest_1samp do Python/scipy. E aceita com a média 81.9J com pValue=0.3356.

Mas se for comparar as médias de TC1 e deste TC3, a hipótese

$$Ho : \mu 1 = \mu 3$$

seria rejeitada com 1%, ou menos, de chance porque pvalue=4.1062537540007385e-07.

O gráfico de energia ficaria da seguinte forma:

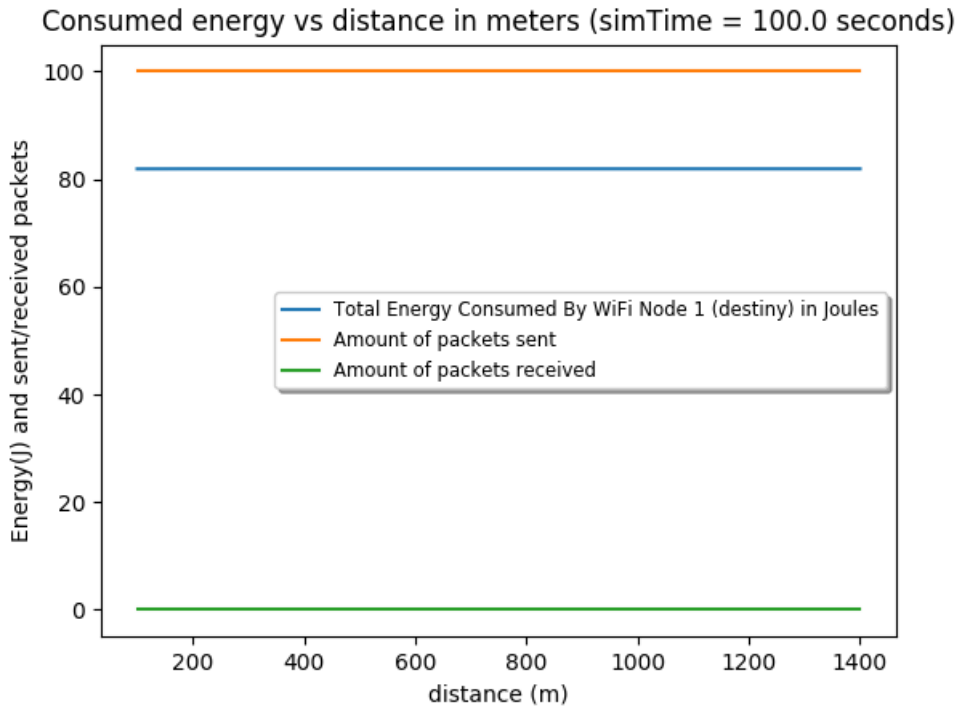

Figura 9: Gráfico de energia vs distância até o Rx, IEEE802.11a

Observe que além de o consumo manter constante à 81.9 J em 100 segundos. O nó Rx não recebe nenhum pacote do nó Tx. Mesmo dobrando o valor da amperagem em Rx (RxCurrentRxA), nenhum pacote é recebido. Isso merece uma investigação separada.

3.4 TC4: Verificar o nível de consumo (em Joules) baseado no tempo de simulação (em segundos)

Os dados de energia na simulação com tempo de simulação de 120s e com distância até o Rx fixa de 10 metros podem ser observados na tabela abaixo.

Tabela de Energia consumida e colhida	
totalEnergyConsumedByRadio(J)	totalEnergyHarvestedByHarvester(J)
95.6185	5.66049

Observe que a energia colhida representa apenas 6 porcentos da energia consumida, no final da simulação.

O gráfico de energia consumida e colhida ficaria da seguinte forma:

O gráfico mostra que o nó Rx tem pouca autonomia de energia com apenas a carga fornecida pelo coletor suplementar (harvester). A energia consumida é muito maior e a consequência é a perda de pacote. Portanto, a maior autonomia seria obtida por conta, em grande parte, da carga inicial.

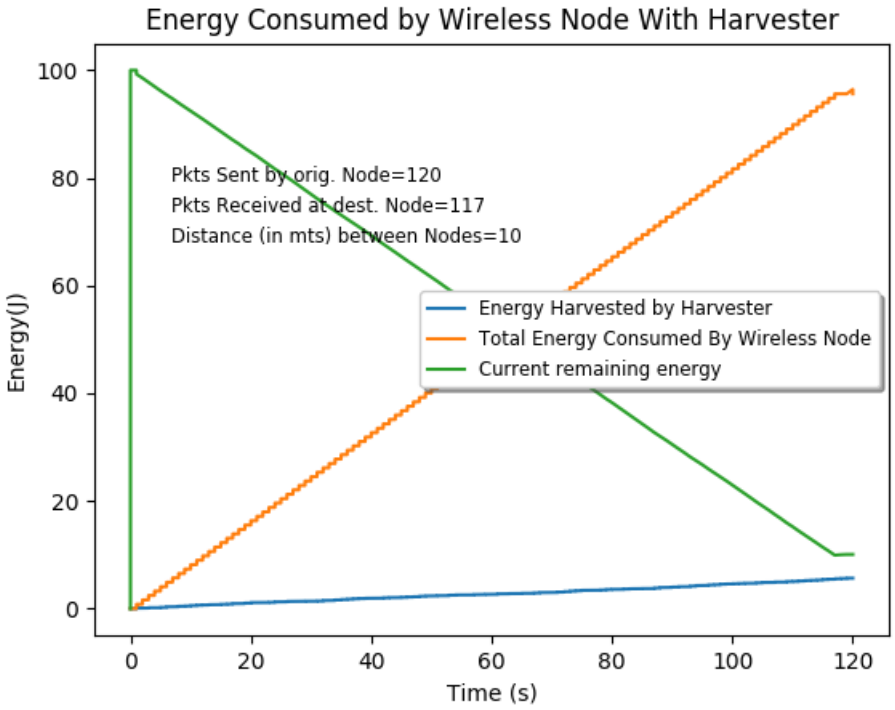

Figura 10: Gráfico de energia vs tempo de simulação de 120s, IEEE802.11b

3.5 TC5, Verificar o nível de consumo (em Joules) variando a distância até o Rx (em metros), usando NS3/NetAnim

Os dados de energia na simulação com variação de distância até o Rx podem ser observados na tabela abaixo.

Tabela de Energia WiFi, NetAnim				
simTime	distanceToRx	pktReceived	pktSent	totalEnergy
100	25	90	100	81.9
100	50	90	100	81.9
100	75	90	100	81.9
100	100	63	100	81.9
100	125	0	100	81.9
100	150	0	100	81.9
100	175	0	100	81.9
100	200	0	100	81.9
100	225	0	100	81.9
100	250	0	100	81.9
100	275	0	100	81.9
100	300	0	100	81.9
100	325	0	100	81.9
100	350	0	100	81.9

Observe que não há variação do consumo de energia do nó Rx. Sempre 81.9J a cada 100s.

O teste hipótese

$$Ho : \mu = 81.9$$

utilizando o teste de t-student seria aceito porque o desvio padrão seria s=0, onde

$$s = \sqrt{\frac{\sum(x_i - 81.9)^2}{n-1}}$$

Apesar de o TC1 e o TC5 utilizarem-se do padrão Wifi IEEE802.11b, um teste de igualdade de média utilizando t-student mostra que suas médias de energia são distintas pois o $pValue = 5.56 \times 10^{-6}$, de acordo com o pacote scipy.stats do Python, indica que é muito inferior ao nível de significância de alta precisão $\alpha = 0.1\%$. O que pode indicar que as bases dos código-fonte utilizados no TC1 (energy-model-with-harvesting-example.cc) e o TC5 (wireless-animation.cc) tem diferenças na implementação.

O gráfico de energia ficaria da seguinte forma:

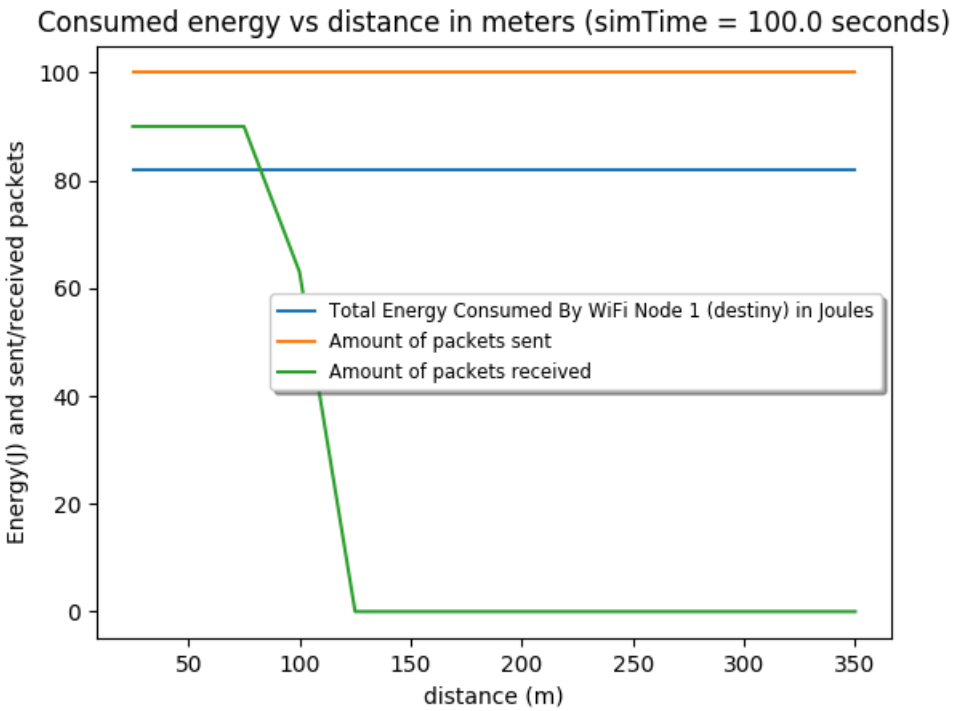

Figura 11: Gráfico de energia vs distância até o Rx, NetAnim

A cada passo de 25m com tempo de simulação de 100s, não há diferença de consumo de energia do nó Rx, sempre a 81.9J. O que se percebe é a perda de pacotes com a distância a partir de 75m, possivelmente devido à degradação do sinal Wifi e a potência na recepção seja menor do que é estabelecido pelo Rx Sensibility no nó de destino Rx. Essa simulação não foi feita de 100, ..., 1400 como em TC1 pois a quantidade de pacotes já caem bastante antes de 100m.

O script além de fornecer dados para o gráfico, gerou arquivos .xml para ser executado pela aplicação NetAnim para a visualização da rede WSN, como pode ser visto abaixa.

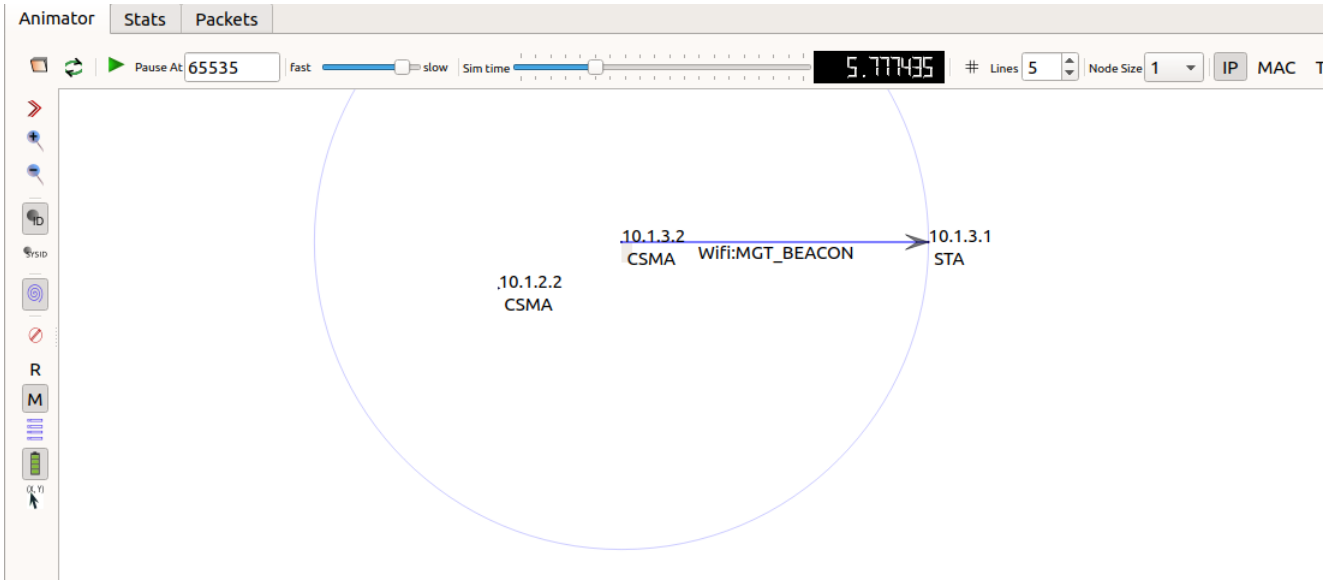

Figura 12: Visualização da Rede WSN com o NetAnim

A única saída do NetAnim relativo à energia é o gráfico de energia remanescente. Pode pode ser vista no gráfico abaixo.

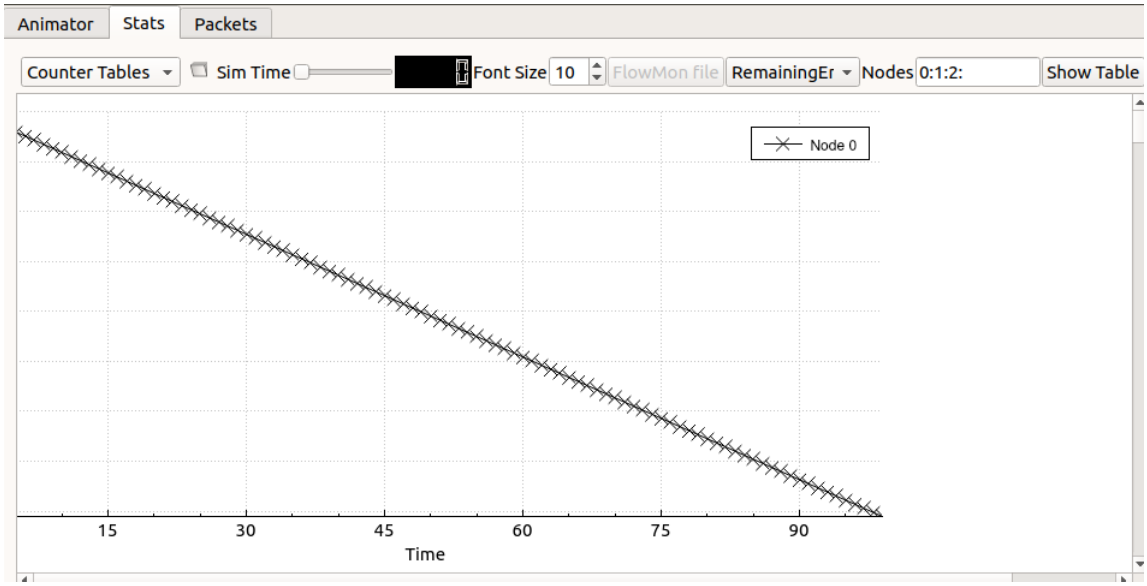

Figura 13: Visualização da Rede WSN com o NetAnim

Este gráfico apenas mostra uma tendência decrescente de energia remanescente decorrente do uso do sensor ao longo do tempo de simulação. O total de energia consumida e a quantidade pacotes enviadas e recebidos por Rx teve que ser codificada à parte.

3.6 TC6, Verificar o nível de consumo (em Joules) variando a distância até o Rx (em metros), usando OMNETPP

Os dados de energia na simulação com variação de distância até o Rx podem ser observados na tabela abaixo.

Tabela de Energia usando OMNETPP						
simTime	distance	Consumption	Mean	StdDev	pkt sent	pkt rec
100	100	0.00862546	0.00862546	0.02507486	19	18
100	200	0.00862546	0.00862546	0.02507486	19	18
100	300	0.00862546	0.00871772	0.02518177	19	18
100	400	0.00862546	0.00871772	0.02518177	19	18
100	500	0.00862546	0.00862546	0.02507586	19	18
100	600	0.00862546	0.00862546	0.02507486	19	18
100	700	0.00862546	0.00862546	0.02507486	19	18
100	800	0.00033333	0.00033333	0.00077849	1	0
100	900	0.00033333	0.00033333	0.00084327	1	0
100	1000	0.00033333	0.00033333	0.00077849	1	0
100	1100	0.00033333	0.00033333	0.00077849	1	0
100	1200	0.00033333	0.00033333	0.00077849	1	0
100	1300	0.00033333	0.00033333	0.00077849	1	0
100	1400	0.00033333	0.00033333	0.00077849	1	0

Observa-se que a distância até Rx a partir de 800 metros (ou até um pouco antes disso), já não há pacotes recebidos (pkt rec). Isto pode indicar que o sinal proveniente de Tx pode estar chegando atenuado devido à distância.

O gráfico error bar abaixo dá uma ideia do comportamento da energia ao longo da distância, utilizando-se das médias e desvios padrões de energia da tabela acima.

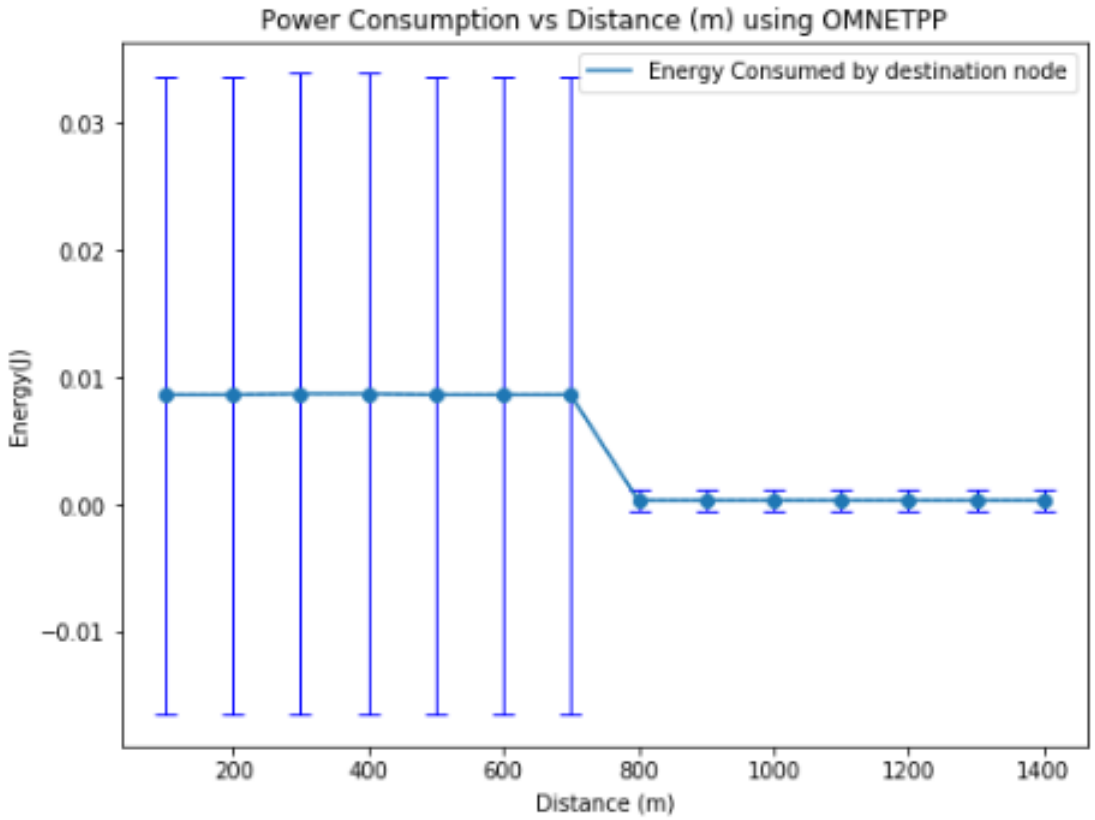

Figura 14: Gráfico de energia vs distância até o Rx

O gráfico mostra que há uma queda no consumo de energia entre a distância 700 e 800 metros, talvez porque o nó Rx entrou em estado de repouso (ou estado de baixa potência de acordo com a máquina de estado FSM), decorrente de um possível fato de que não haver sinais de Tx recebidos com potência suficientemente alta para ultrapassar o limiar da sensibilidade de Rx. Porém não dá para afirmar que há uma diferença de valor de energia ao longo das distâncias escolhidas pois os valores dos desvios padrões do "possível"estado ativo do FSM estão entre 100 a 700 m é bastante grande em comparação aos desvios padrões de 800 até 1400 m de forma que não se rejeita a hipótese nula de igualdade nos valores de energia.

3.7 TC7, Verificar o comportamento do consumo de energia entre os nós Rx e Tx durante o tempo de simulação, usando OMNETPP

O gráfico abaixo dá uma ideia do comportamento da energia durante o tempo de simulação nos nós Tx e Rx.

O gráfico mostra a evolução da capacidade residual dos nós Rx (host[0]) e Tx (host[1]). Inicializam-se de acordo com a configuração do arquivo omnetpp.ini, onde a energia residual inicial é um valor aleatório entre 0 e sua capacidade nominal de 0.05J. A capacidade de energia residual não excede sua capacidade nominal, mas pode ser reduzida até o apagamento por completo do dispositivo i.e. nó Tx ou Rx, onde não se trata nenhuma

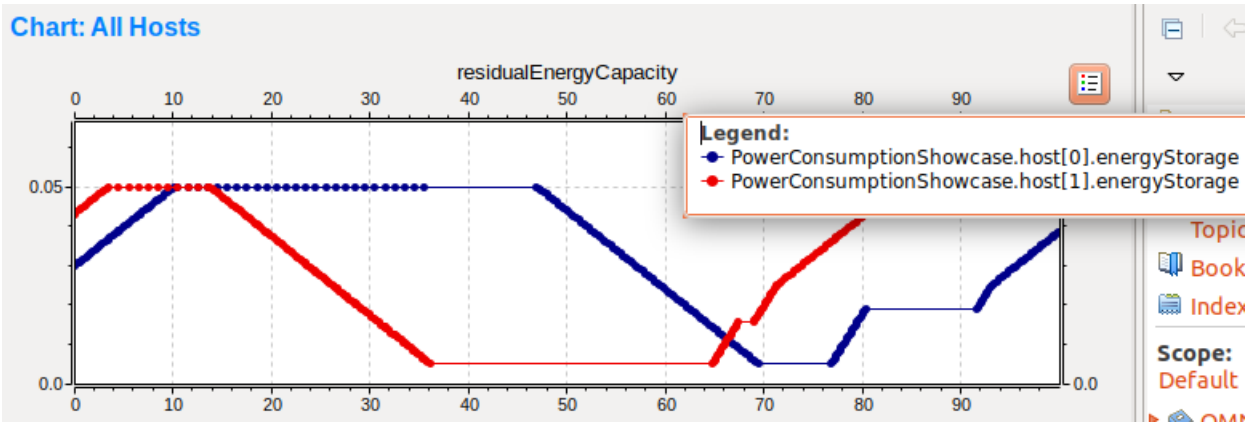

Figura 15: Gráfico de Residual de Energia nos nós Tx e Rx

mensagem. O dispositivo Tx ou Rx pode ser recarregada pelo gerador até chegar à sua capacidade nominal, quando a capacidade de energia estiver completamente exaurida.

4 Conclusão

De acordo com os resultados obtidos pelo simulador NS3, não há diferença do consumo de energia se considerar a variação de distância até o Rx. Poderá ter alguma diferença de consumo se variar alguma característica do padrão Wifi, como o physical mode (e.g. se utilizar DsssRate1Mbs ou DsssRate11Mbs) ou variar a versão do protocolo Wifi como por exemplo alguma versão IEEE802.11 (a, b, d, e, f, g, h, ...).

Um coletor de energia suplementar (harvester) contribui com muito pouco na alimentação do nó Rx. Uma carga inicial maior é requerida para uma maior autonomia.

A saída do NetAnim oferece uma boa visualização dos eventos que ocorrem durante o tempo de simulação, porém a lógica de consumo de energia teve que ser programada à parte. A única informação de energia foi a energia remanescente do dispositivo, porém pouco útil para a conclusão deste trabalho.

Diferença de valores de energia foram observadas com o NS3, versão 3.29, implementada manualmente com Mercurial e com Bake.

Os resultados verificados com a ferramenta OMNETPP não mostraram nenhuma diferença de consumo de energia variando a distância até o Rx possivelmente devido a uma alta flutuação dos valores de energia, ou variância.

O OMNETPP tem uma vantagem sobre o NS3 no que se refere ao gerenciamento da capacidade de energia residual. Quando a mesma termina, ocorre o apagamento do dispositivo WSN por completo, nenhuma operação é possível neste estado, tão logo isso ocorra, uma recarga é realizada até a sua capacidade nominal, de modo a recobrar ao seu estado ativo.

Uma desvantagem do OMNETPP em relação ao NS3, é que a primeira não tem limitações para configurar padrões WiFi IEEE.

Um trabalho futuro é recomendado para investigar o consumo de energia com todas as características de um sensor, não apenas aquele baseado em IEEE802.11x, mas outros protocolos também seriam de grande valia, como o LoRaWan.

Referências

[1] C. Tapparello, H. Ayatollahi, and W. Heinzelman. Energy harvesting framework for network simulator 3 (ns-3). 11 2014.

[2] C. Trasviña-Moreno, A., n. Asensio, R. Casas, R. Blasco, and l. Marco. Wifi sensor networks: a estudy of energy consumption. *ResearchGate*, Feb. 2014.