Relatório - EP3 - MAC0463

Renê Cardozo - 9797315

December 13, 2021

1 Introdução

Para realizar as simulações foi utilizado o simulador de redes ns-3 junto ao módulo de simulação 5G "5G-LENA". Esta combinação permite a simulação de redes 5G conforme o documento *Release 15 - New Radio* da 3GPP, organização que define os padrões mundiais para comunicação móvel.

Como a integração do simulador com determinadas distribuições podem apresentar problemas de dependências com relação à versão das bibliotecas do gerenciador de pacotes, as simulações foram realizadas utilizando um container *Docker*, com imagem disponível na plataforma *Docker Hub*, no repositório reneepc/ns3-5g.

2 Elementos Gerais do Simulador

Equipamentos de rede no ns-3 são representados por objetos denominados nós. Ao invés de utilizar herança para representar diferentes tipos de equipamentos, o simulador utiliza um sistema de agregação no qual é possível associar outros objetos a um nó. Para cada nó é possível associar múltiplos dispositivos de rede da classe NetDevice, bem como associar aplicações com a classe Application.

Para possibilitar a comunicação entre diferentes nós é necessário que seja especificado um canal de comunicação, o qual será associado a cada dispositivo de rede (NetDevice) que deseja comunicar-se entre si. Portanto, é possível especificar diferentes canais associados a diferentes dispositivos de rede, como um computador com uma placa de rede Ethernet, uma Bluetooth e outra Wi-Fi.

Para que seja possível produzir pacotes IP é necessário instalar a a pilha de protocolos TCP/IP, a qual é definida como uma aplicação que pode ser agregada a um nó. Uma vez instalada a pilha de protocolos é possível ainda utilizar uma aplicação que coordena o envio dos pacotes, como UdpClient e UdpServer ou TcpClient e TcpServer, entre outras aplicações.

Algoritmos de perda, delay e beamforming podem ser especificados para definir simulações mais realistas, bem como o estabelecimento de cenários já prontos como o meio de prédios ou campo aberto.

Todas as simulações envolvendo redes sem fio devem utilizar um modelo de mobilidade no qual todos os nós possuem uma posição em um determinado

período de tempo, mesmo que esta seja estática durante toda a simulação. Para isso podem ser utilizados os diferentes modelos de mobilidade do ns-3.

As posições dos nós são essenciais para o devido cálculo do delay, taxa de transmissão, perda de pacotes e outras características de redes sem fio.

3 Módulo 5G-LENA

O módulo 5G-LENA é capaz de simular redes 5G, sendo baseado no antigo módulo para simulação de LTE do ns-3. Com este módulo é possível definir estações base (gNB) e estações de usuário (UE), para as quais é possível realizar uma série de modificações nos dispositivos de rede (NetDevices) associados. Estes podem definir diferentes potências, número de antenas, frequência de uso, modo do canal, etc.

No novo modelo de redes 5G há a possibilidade de definir diferentes modos para a utilização do canal, como Ultra-Low Latency e Voice. Estes modos definem a prioridade dos pacotes na fila de processamento, a taxa de transmissão e outras características do canal de transmissão.

4 Simulações

Um requisito para produzir as simulações é possuir *Docker* instalado. Uma vez feito, pode-se criar um container com a imagem do simulador com o comando:

```
docker run -it reneepc/ns3-5g
```

Este comando fará o download da imagem do container, caso não exista, e iniciará em modo iterativo, abrindo um shell para o usuário.

O programa waf é utilizado para gerenciar a compilação e execução de scripts. Para este exemplo, foi utilizado o script cttc-nr-demo.cc do módulo 5G-LENA. Este se encontra no caminho ./contrib/nr/examples/cttc-nr-demo.cc, relativo ao diretório padrão de entrada do container.

Este script utiliza inicialmente somente uma estação base (gNB) e duas estações de usuário (UE). Mais estações base ou de usuário podem ser definidas por meio de argumentos passados ao script. As estações de usuário pares serão definidas como tendo modo de transmissão ultra-low latency e as ímpares como sendo modo de transmissão voice.

Os nós serão distribuídos em uma estrutura de grid, estando todos na mesma altura. Esta estrutura de grid é definida por um método auxiliar do próprio módulo do 5G-LENA.

Uma vez determinado o modelo de mobilidade e modo de transmissão, o canal de transmissão é definido e associado às interfaces de rede, as quais serão agregadas posteriormente aos nós. Neste passo é possível definir diversas características do canal, como o número de sub canais, o número de antenas, o número de sub-bandas, condição de atualização do canal, frequência de transmissão, algoritmo de beamforming, largura de banda total, etc.

Após isso, uma aplicação UdpServer é instalada na gNB e UdpClients são instalados nas UEs. Por padrão, as UEs irão transmitir 10000 pacotes no intervalo de 1 segundo de duração para a gNB.

A utilização do módulo flow-monitor permite o cálculo de diversos parâmetros de rede do cenário de rede estudado.

Para executar este script com o waf o seguinte comando deve ser utilizado:

```
./waf --run contrib/nr/examples/cttc-nr-demo.cc
```

5 Variação do Número de UEs

Com o objetivo de explorar os efeitos do aumento do número de estações de usuário e um possível cenário de congestionamento de rede foram realizadas simulações com 2 (padrão), 5 e 10 UEs. O número diferente de UEs pode ser especificado da seguinte forma:

```
./waf --run "contrib/nr/examples/cttc-nr-demo.cc
--ueNumPergNb=<numero-de-UEs>"
```

Abaixo estão os outputs para as simulações com 2, 5 e 10 UEs para apenas 1 estação base (gNB).

Output para 2 UEs

```
Flow 1 (1.0.0.2:49153 -> 7.0.0.2:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.271518 ms
  Mean jitter:  0.030006 ms
  Rx Packets: 5998
Flow 2 (1.0.0.2:49154 -> 7.0.0.3:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   7671040
  Throughput: 102.280533 Mbps
  Mean delay:  0.835065 ms
  Mean jitter:  0.119991 ms
  Rx Packets: 5993


  Mean flow throughput: 56.258560
  Mean flow delay: 0.553292
```

Output para 5 UEs

```
Flow 1 (1.0.0.2:49153 -> 7.0.0.2:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.271605 ms
  Mean jitter:  0.030040 ms
  Rx Packets: 5998
Flow 2 (1.0.0.2:49154 -> 7.0.0.3:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.276077 ms
  Mean jitter:  0.030038 ms
  Rx Packets: 5998
Flow 3 (1.0.0.2:49155 -> 7.0.0.4:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.280548 ms
  Mean jitter:  0.030037 ms
  Rx Packets: 5998
Flow 4 (1.0.0.2:49156 -> 7.0.0.5:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   7671040
  Throughput: 102.280533 Mbps
  Mean delay:  0.836465 ms
  Mean jitter:  0.119917 ms
  Rx Packets: 5993
Flow 5 (1.0.0.2:49157 -> 7.0.0.6:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   7667200
  Throughput: 102.229333 Mbps
  Mean delay:  0.900982 ms
  Mean jitter:  0.119907 ms
```

```
  Rx Packets: 5990
```

Output para 10 UEs

```
Flow 1 (1.0.0.2:49153 -> 7.0.0.2:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.271605 ms
  Mean jitter:  0.030040 ms
  Rx Packets: 5998
Flow 2 (1.0.0.2:49154 -> 7.0.0.3:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.276077 ms
  Mean jitter:  0.030038 ms
  Rx Packets: 5998
Flow 3 (1.0.0.2:49155 -> 7.0.0.4:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.280550 ms
  Mean jitter:  0.030037 ms
  Rx Packets: 5998
Flow 4 (1.0.0.2:49156 -> 7.0.0.5:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767744
  Throughput: 10.236587 Mbps
  Mean delay:  0.285017 ms
  Mean jitter:  0.030037 ms
  Rx Packets: 5998
Flow 5 (1.0.0.2:49157 -> 7.0.0.6:1234) proto UDP
  Tx Packets: 6000
  Tx Bytes:   768000
  TxOffered:  10.240000 Mbps
  Rx Bytes:   767616
```

```
  Throughput: 10.234880 Mbps
  Mean delay:  0.289482 ms
  Mean jitter:  0.030039 ms
  Rx Packets: 5997
Flow 6 (1.0.0.2:49158 -> 7.0.0.7:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   7185920
  Throughput: 95.812267 Mbps
  Mean delay:  20.514970 ms
  Mean jitter:  0.121268 ms
  Rx Packets: 5614
Flow 7 (1.0.0.2:49159 -> 7.0.0.8:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   6886400
  Throughput: 91.818667 Mbps
  Mean delay:  32.283522 ms
  Mean jitter:  0.122190 ms
  Rx Packets: 5380
Flow 8 (1.0.0.2:49160 -> 7.0.0.9:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   4784640
  Throughput: 63.795200 Mbps
  Mean delay:  114.419989 ms
  Mean jitter:  0.131937 ms
  Rx Packets: 3738
Flow 9 (1.0.0.2:49161 -> 7.0.0.10:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   4783360
  Throughput: 63.778133 Mbps
  Mean delay:  114.425767 ms
  Mean jitter:  0.131937 ms
  Rx Packets: 3737
Flow 10 (1.0.0.2:49162 -> 7.0.0.11:1235) proto UDP
  Tx Packets: 6000
  Tx Bytes:   7680000
  TxOffered:  102.400000 Mbps
  Rx Bytes:   4783360
  Throughput: 63.778133 Mbps
```

```
Mean delay:  114.461481 ms
Mean jitter:  0.131937 ms
Rx Packets: 3737


Mean flow throughput: 43.016363
Mean flow delay: 39.750846
```

É possível identificar, por exemplo, que os nós que estão em modo ultra-low latency apresentam uma taxa de transmissão de 10Mbps, enquanto os nós que estão em voice apresentam uma taxa de transmissão de 100 Mbps. Sacrificando um pouco a largura de banda é possível obter um delay menor. Para a simulação com 2 UEs, o nó em ultra-low latency apresenta um delay quase 4 vezes menor do que o nó em voice, bem como um jitter 3 vezes menor.

Uma vez aumentados os nós de estações de usuário, temos que os dispositivos em ultra-low latency mantém o seu delay e taxa de transmissão relativamente estáveis, independente de quantos nós são adicionados. A taxa de perda de pacotes (Tx Packets - Rx Packets) permanece inalterada. Contudo, para os nós em modo voice, o delay é aumentado substancialmente, quase 1000x o valor inicial para alguns nós. Além disso, cerca $\frac{1}{3}$ dos pacotes é perdido.

OBS: Os arquivos de simulação, Dockerfile e este relatório estão disponíveis no repositório: EP3-MAC0463