# Experiment III: TCP and LTE Handover

EE 595 B

March 1, 2019

## I. Overview

The goal of this assignment is to experiment with some configurations and settings of a simple LTE network, and to explore the performance evaluation of a long-running TCP flow from an Internet server to the UE.

Your repository should be at changeset 0373e855 (Friday, March 1), or later, of the 'course' branch.

## II. TCP over wireless

TCP is the most widely used transport protocol in use on the Internet. TCP operates above the IP layer and provides a reliable byte stream channel between applications through the network 'socket' abstraction. Bytes that are streamed into a network socket on one end can be read out in the same order from the network socket on the other end. TCP provides reliable, in-order guarantee through the use of a selective repeat ARQ protocol, detecting losses via sequence number inspection, and recovering from losses through retransmission.

TCP also is responsible for performing end-to-end congestion control. In the late 1980s, a fundamental algorithm called "additive increase, multiplicative decrease" (developed by Raj Jain, then at Digital Equipment Corp.) was added to allow TCP to slowly increase its sending rate (by approximately one TCP segment size per round trip time (RTT)) if no signs of congestion are present, but to decrease its sending rate by half upon detection of congestion. At the time, the introduction of the algorithm alleviated the pending "congestion collapse" of the Internet.

There are actually two responses to detected congestion, one less severe than the other. The first response is called "fast retransmit and recovery," and allows the TCP to perform the operation described above; although the sending rate is halved, the TCP flow still continues through this process. Fast retransmit and recovery is performed upon the first detection of a loss in the network. The second response is more

severe, and is called a “retransmission timeout.” If the first retransmission does not succeed, the TCP will stall and a timer (usually around 2 seconds duration) will expire, and the lost segment will be retransmitted again in an exponential backoff pattern until successful. The sending rate will be reduced to one segment per RTT (to grow again from there), and there will be a pause in data transmission.

TCP interprets packet losses as signs of congestion (a queue overflow from within the network). However, a wireless link may lose packets due to transmission errors, and not congestion. While such errors should be recovered, in principle they should not cause the sending rate to slow down (if not due to congestion). Therefore, there has been much research over the past twenty years to explore how to improve TCP performance over wireless links that incur packet losses.

In this experiment, we will experiment with a long-running file download from a notional Internet server to a mobile handset, while varying the configuration to induce different delays and losses. We will not experiment with TCP variants to overcome the performance hits, but will instead look at the base TCP performance in this environment.

## III. NS-3 Experiment: Performance of TCP over LTE and X2 handover

Unless you want to use debug logging, it is recommended that after you perform ’git pull’ on the course branch, that you build the program in optimized mode:

```
$ ./waf configure --enable-examples -d optimized
$ ./waf build
```

Fig. 1 illustrates a simple LTE-based network. A notional UE is configured to move along the x-dimension, at a distance ‘yDistanceForUe’ meters from the x-axis. By default, ‘yDistanceForUe’ is 1000 meters, so the initial position of the UE is (0, 1000).

Two eNBs are configured, separated by a distance ‘x2Distance’ from each other, and at a distance of ‘x2Distance’ from the x-axis. The first eNB is also a distance of ‘x2Distance’ from the y-axis. The default value of ‘x2Distance’ is 500 meters, so the first eNB position is at (500, 500), and the second at (1000, 500).

The UE moves horizontally at a speed, default 20 m/s. The simulation time is configured to correspond to the time it takes for the UE to move from initial position for a distance of $3 * x2Distance$. This allows the UE to move to the halfway point between the eNBs around the halfway point of the simulation.

The program options are as follows:

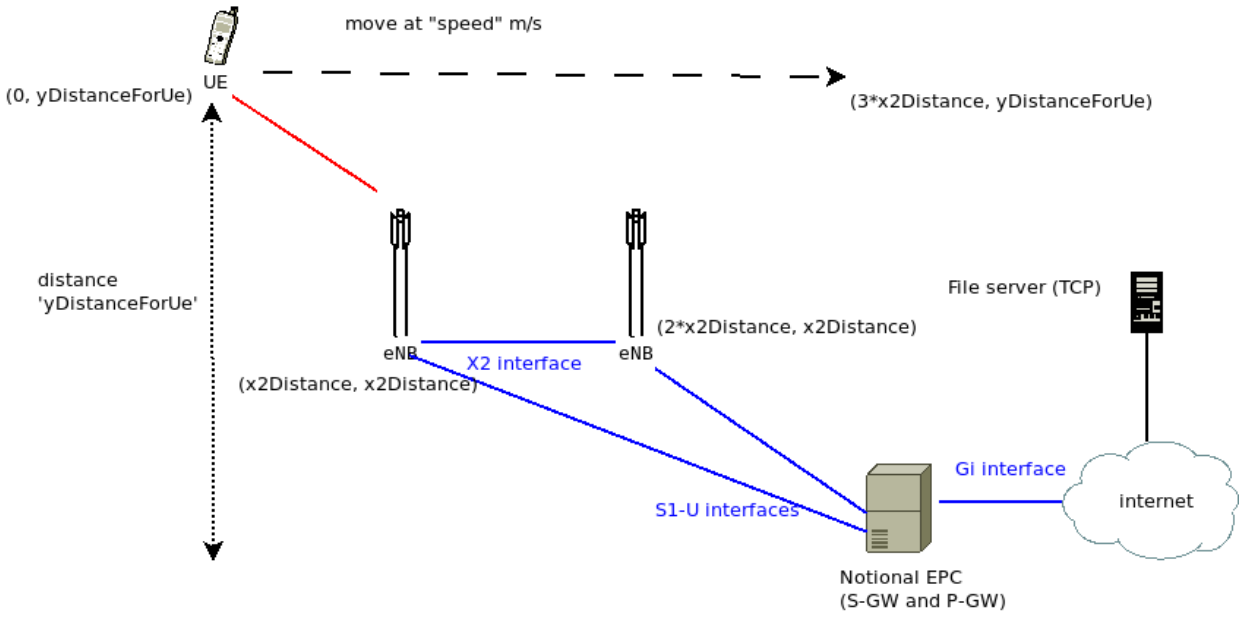


Fig. 1. Experiment layout

```
$ ./waf --run lte-tcp-x2-handover --PrintHelp
Program Options:
    --speed:           Speed of the UE (m/s) [20]
    --x2Distance:      Distance between eNB at X2 (meters) [500]
    --yDistanceForUe:  y value (meters) for UE [1000]
    --enbTxPowerDbm:   TX power (dBm) used by eNBs [46]
    --useRlcUm:        Use LTE RLC UM mode [false]
    --handoverType:    Handover type (A2A4 or A3Rsrp) [A2A4]
    --pcap:            Enable pcap tracing [false]
    --verbose:         Enable verbose logging [false]
```

Three additional options shown above can further tune the simulation parameters. The ‘enbTxPowerDbm’ can be used to change the downlink eNB power. The ‘useRlcUm’ can be used to force the use of the RLC Unacknowledged Mode (by default, RLC Acknowledged Mode is used). Finally, the ‘handoverType’ argument allows the user to change between the A2A4 or A3Rsrp algorithms.

The last two command line arguments can enable PCAP traces (for use in Wireshark) and some ns-3 logging (if the simulator has been built in debug mode).

Upon running the program, a number of data files are generated, as described below. The output to terminal is as follows:

```
$ ./waf --run lte-tcp-x2-handover
Initial positions:  UE: (0,1000), eNB1: (500,500), eNB2: (1000,500)
Simulation time: 75 sec
```

```
0.0202143 node 4 UE IMSI 1: connected to CellId 1 with RNTI 1
0.0202143 node 2 eNB CellId 1: successful connection of UE with IMSI 1 RNTI 1
*** Simulation time: 10.0s
*** Simulation time: 20.0s
*** Simulation time: 30.0s
38.2 node 2 eNB CellId 1: start handover of UE with IMSI 1 RNTI 1 to CellId 2
38.2 node 4 UE IMSI 1: previously connected to CellId 1 with RNTI 1, doing handover to CellId 2
38.2 node 4 UE IMSI 1: successful handover to CellId 2 with RNTI 1
38.2 node 3 eNB CellId 2: completed handover of UE with IMSI 1 RNTI 1
*** Simulation time: 40.0s
*** Simulation time: 50.0s
*** Simulation time: 60.0s
*** Simulation time: 70.0s
```

The first two statements print out the initial positions and the simulation time, respectively. Simulation progress is also noted by statements preceded by asterisks. The first events near the start of the simulation indicate initial cell acquisition of cell 1 by the UE. At time 38.2, the handover algorithm initiates handover, which is quickly completed. The simulation ends at time 75 seconds.

The example uses the Round Robin scheduler (`RrFfMacScheduler`) that has been slightly modified to output a wideband CQI trace, and in general, other ns-3 LTE defaults.

### *A. LTE stats*

There are various built-in LTE statistics that are written, with names such as 'DlMacStats.txt', 'DlPdcpStats.txt', etc. The 'Dl' corresponds to downlink; there are uplink ones labelled 'Ul'. The ns-3 documentation provides descriptions of these files. In this experiment, we will use the 'DlMacStats.txt' file to obtain MCS information.

The following excerpt from 'DlMacStats.txt' shows the key fields used in this experiment:

```
% time cellId IMSI frame sframe RNTI mcsTb1 sizeTb1 mcsTb2 sizeTb2ccId
1.011 1 1 102 2 1 28 2196 0 0 0
1.026 1 1 103 7 1 28 2196 0 0 0
1.046 1 1 105 7 1 28 2196 0 0 0
```

The scheduler allocates, at MCS value 28, a transport block of size 2196. If one searches the trace, one will find that at most 2196 bytes are allocated each subframe (1 ms), leading to a peak throughput of $2196 * 8/0.001 = 17.6Mb/s$. In actuality, the long-running average TCP throughput that can be achieved will be somewhat less than this due to overhead.

### B. TCP stats

TCP statistics are gathered in two files. The first is ‘lte-tcp-x2-handover.tcp-receive.dat’, which lists the number of bytes and timestamp for each socket read operation at the receiver (UE). This file can be used to compute end-to-end throughput (a plotting file described below does this). The second is ‘lte-tcp-x2-handover.tcp-state.dat’ which lists changes to the TCP congestion control state machine. The key event to look for in this file is any example of a ‘RECOVER’ or ‘LOSS’ event.

For instance, the following example shows a TCP fast recovery event:

```
# time   congState
 38.186 CA_DISORDER
 38.188 CA_RECOVERY
 38.413 CA_OPEN
```

The first ‘DISORDER’ event represents TCP detecting a sequence number gap. This leads to a ‘RECOVERY’ event, then an ‘OPEN’ event signifying the end of the loss recovery.

A TCP timeout looks similar, but is identified by a ‘LOSS’ event:

```
# time   congState
 50.887 CA_DISORDER
 51.852 CA_LOSS
 52.155 CA_OPEN
```

### C. CQI data

The wideband CQI data is written to the file ‘lte-tcp-x2-handover.cqi.dat‘.

### D. Position data

The position of the UE is written to the file ‘lte-tcp-x2-handover.position.dat’ every 5 seconds.

### E. UeMeasurements

The RSRP and RSRQ reported values are written to the file ‘lte-tcp-x2-handover.ue-measurements.dat’. The file has the following format:

```
# time   cellId   isServingCell?  RSRP(dBm)  RSRQ(dB)
  0.200   1   1 -107.143   -6.072
  0.200   2   0 -107.147   -6.076
  0.400   1   1 -107.143   -6.072
  0.400   2   0 -107.147   -6.076
```

The UE reports the values of all cells that it hears. The third column of the trace file indicates which cell the UE belongs to (the 'isServingCell?' field). For instance, at time 0.200, cell 1 is serving and cell 2 is not.

The following table is provided as a general guide as to how to interpret the values of RSRP and RSRQ, in practice:

TABLE I

LTE RSRP AND RSRQ

| **channel state** | **RSRP (dBm)** | **RSRQ (dB)** |
|---|---|---|
| excellent | $>= -80$ | $>= -10$ |
| good | -80 to -90 | -10 to -15 |
| mid cell | -90 to -100 | -15 to -20 |
| cell edge | $<= -100$ | $< -20$ |

(table is drawn from https://www.cablefree.net/wirelesstechnology/4glte/lte-rsrq-sinr/)

### *F. Experiment directory*

The program run can be orchestrated with a shell script in the directory `contrib/simple-wireless/experiments/lte-tcp-x2-handover/`.

This automates the generation of a number of plots:

- TCP throughput over time
- CQI over time
- MCS over time
- RSRP over time
- RSRQ over time

The following figures provide the plots generated from the default program configuration. When values are plotted in red, this is due to the UE associating with the cell 1, and values plotted in blue are for times for which the UE is associated with cell 2.

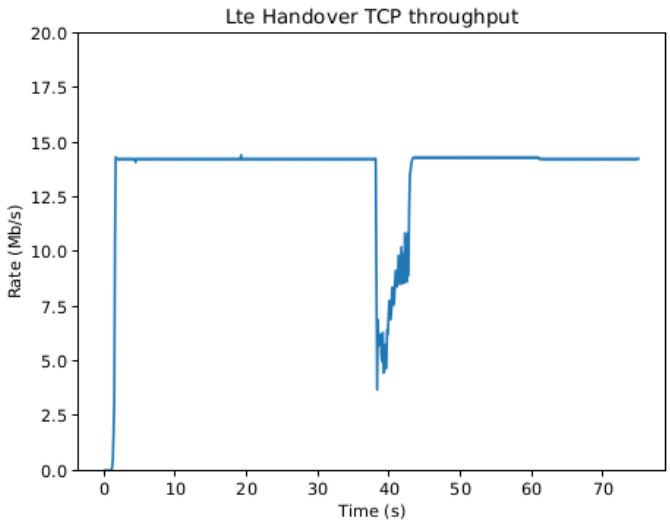


Fig. 2. TCP throughput vs. time

The TCP throughput in Figure 2 shows a disruption to throughput performance at the handover time; the rate drops by half (or more) and then must rebound after the loss event. The handoff is not 'hitless' as there must be some packet drop that the RLC AM is not able to recover.

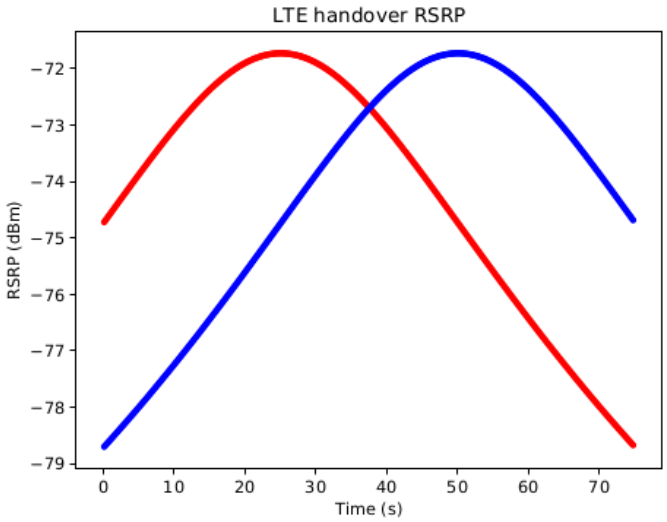


Fig. 3. RSRP vs. time

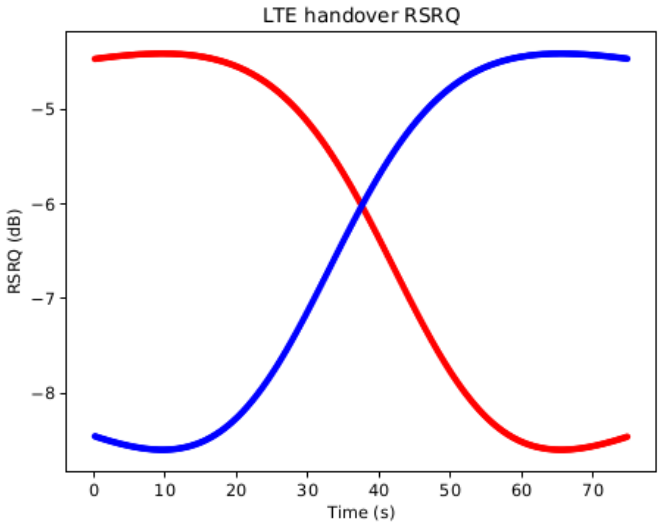


Fig. 4. RSRQ vs. time

The mobility-induced RSRP and RSRQ changes are clearly seen in Figures 3 and 4.

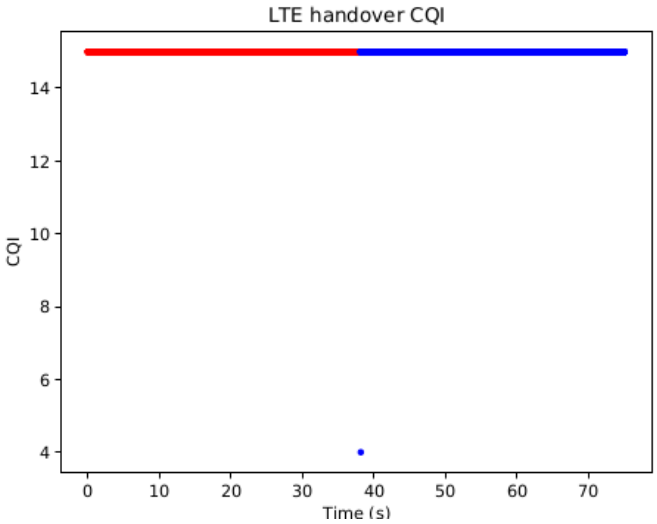


Fig. 5. CQI vs. time

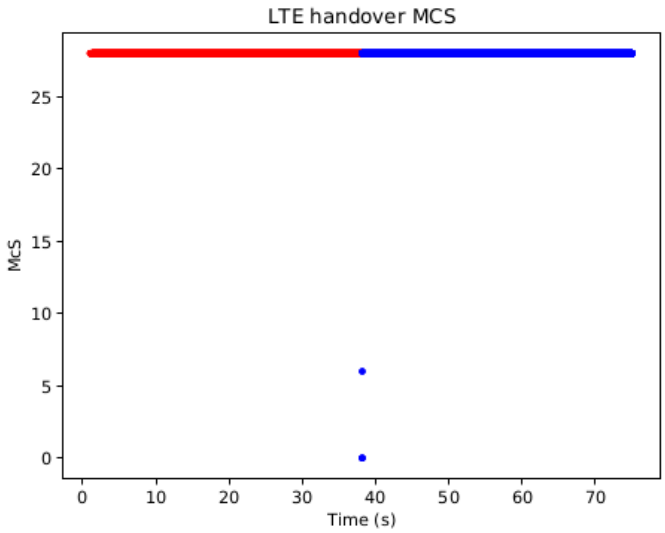


Fig. 6. MCS vs. time

## IV. Questions

Please use the simulator and any necessary background reading to answer the following.

### *A. TCP performance vs distance, stationary UE*

1) By default, the channel quality is excellent. For a stationary UE, what values of 'yDistanceForUe' correspond to channel quality values of good, mid-cell, and cell edge? Use the 'speed=0' argument and vary the 'yDistanceForUe', and use the RSRP measurement for cell 1, comparing the value to the ranges in Table I. It is sufficient to round 'yDistanceForUe' to the nearest hundred as you step through values; for instance, a distance of 1700m brings the RSRP below -80 dBm (into the 'good' range).

2) What are roughly the maximum TCP throughputs observed for each of the cell channel qualities from the previous question? Use the provided plotting capability to answer this, or write your own throughput calculator script.

3) For this configuration, the best CQI reading will show 15 (maximum value) and the best MCS 28 (maximum value). What are the CQI and MCS values for each of these other channel qualities (good, mid-cell, cell edge)?

4) For a 'yDistanceForUe' of 30000 m (RSRP approximately -107 dB), what is the MCS selected, the CQI value reported, and what is the corresponding maximum TCP throughput?

### *B. TCP performance vs distance, LTE RLC UM*

5) Repeat the question 2) from above but with the RLC UM enabled (instead of acknowledged mode). The argument `--useRlcUm=1` can be added to disable acknowledged mode at the RLC layer. Explain your findings qualitatively (why is TCP throughput much worse)?

### *C. TCP handover performance*

6) Using program defaults unless otherwise noted, set the 'yDistanceForUe' to 5000 m and run a mobile simulation (default speed of 20 m/s). Obtain a plot of the TCP throughput. How many LOSS or RECOVER events are in the lte-tcp-x2-handover.tcp-state.dat file? Does changing the speed (to values of 10, 30, 50 m/s) affect the TCP performance?

### *D. Handover algorithm differences*

7) Using program defaults unless otherwise noted, experiment with changing the handover algorithm between A2-A4 and A3-RSRP. The argument `--handoverType=A2A4` or `--handoverType=A3Rsrp` can change the behavior. Explain or justify why the simulation time that handover occurs differs between the two algorithms? Consult the lte-tcp-x2-handover.ue-measurements.dat file for information, the lecture 7 slides, and the following references as needed:

- https://www.nsnam.org/docs/release/3.29/doxygen/classns3_1_1_a2_a4_rsrq_handover_algorithm.html#details
- https://www.nsnam.org/docs/release/3.29/doxygen/classns3_1_1_a3_rsrp_handover_algorithm.html#details

8) TCP performance can be sensitive to small changes in loss patterns, which can be induced by simulation run number changes. Try the following command, varying RngRun between 1 and 2.

```
$ ./waf --run 'lte-tcp-x2-handover --handoverType=A3Rsrp --RngRun=2
```

Does TCP take a timeout (a LOSS event) in either case?