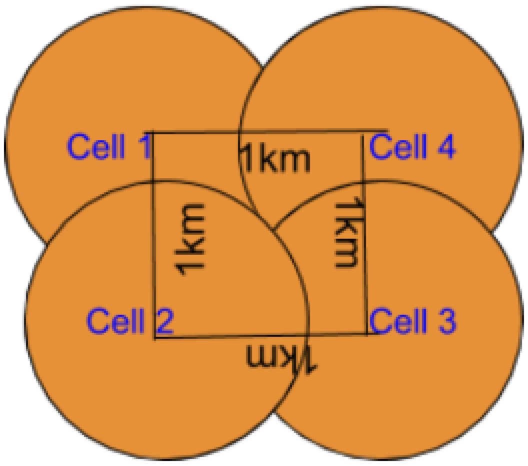
NWS Assignment-2

Vikas Ravi Patil(cs22mtech12006)
Arjit Gupta(cs23mtech12001)

The objective of this assignment is to understand and change code of LTE Schedulers algorithms in NS-3 for necessary stats collection. Further, we evaluated and compared the performance of different Scheduler algorithms.

Topology used-

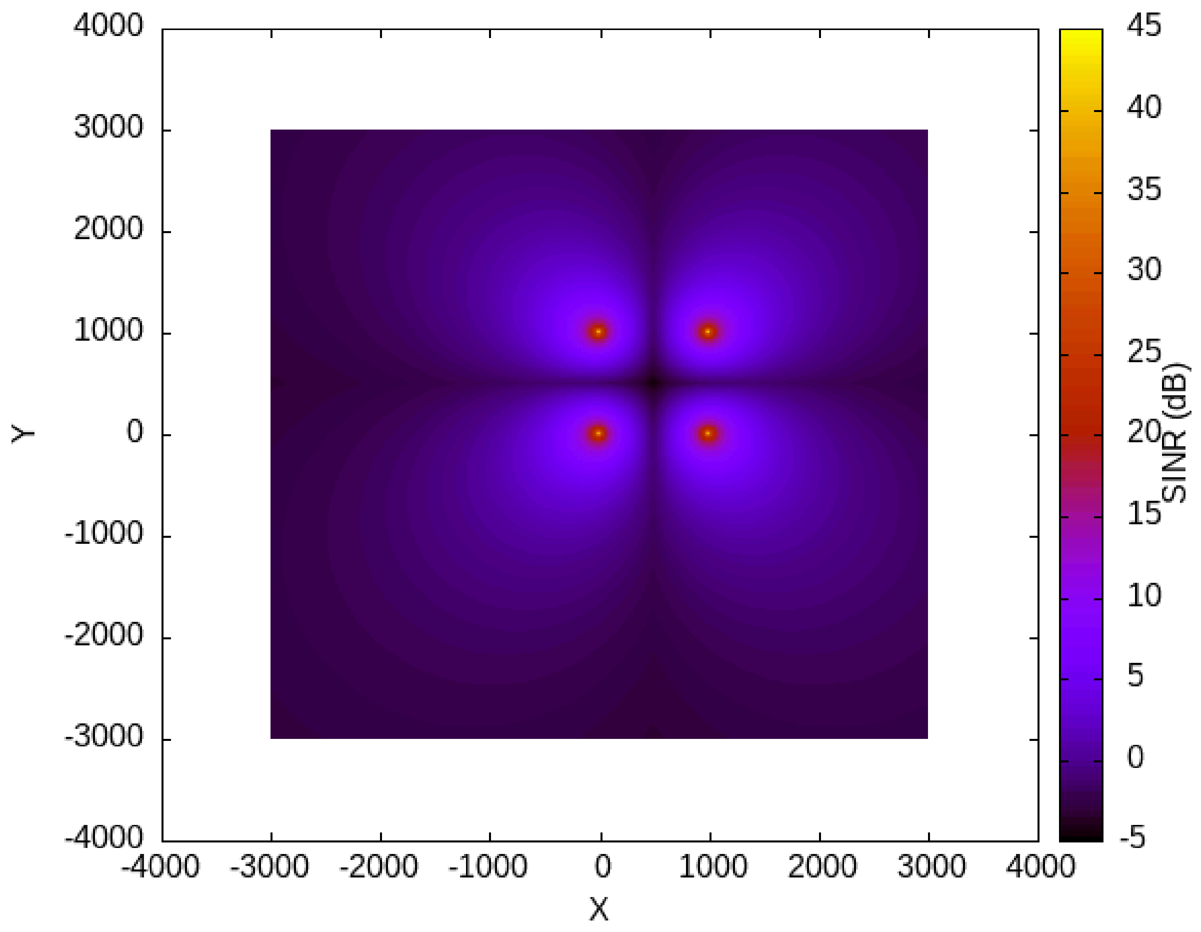


Configuration used

Simulation Parameter	**Value**
Number of UEs	5 per eNB; 1 Downlink UDP Flow per UE from the Remote Host
Number of eNBs	4
Inter distance between eNBs	1 KM
eNB Tx Power	30 dBm (1W)
Application Type	UDP
Full buffer case (UDP Traffic)	1500 bytes per every **1ms** by UDP; Each UE is configured with 1 DL UDP flow of 12 Mbps
Non Full buffer case (UDP Traffic)	1500 bytes per every **10ms** by UDP; Each UE is configured with 1 DL UDP flow of 1.2 Mbps
UE mobility speeds	0, 5 m/s; where in a given expt all UEs are configured with one of these two speeds
UE mobility model	RandomWalk2d Mobility
UEs placement in a Cell	Random disc placement within 500m radius of eNB
# of RBs	50 in DL and 50 in UL (LTE FDD)
UE attachment to eNB	Automatic to one of eNBs based on received signal strength, so handovers may take place during mobility
Total simulation time	10 seconds

Number of seeds per experiment	5;
	RngRun1 = "Last TWO DIGITS of one of your ROLL NUMBERS"
	RngRun5 = RngRun1+4

Graph 1: SINR Radio Environment Map (REM) of 4-cell topology given above.



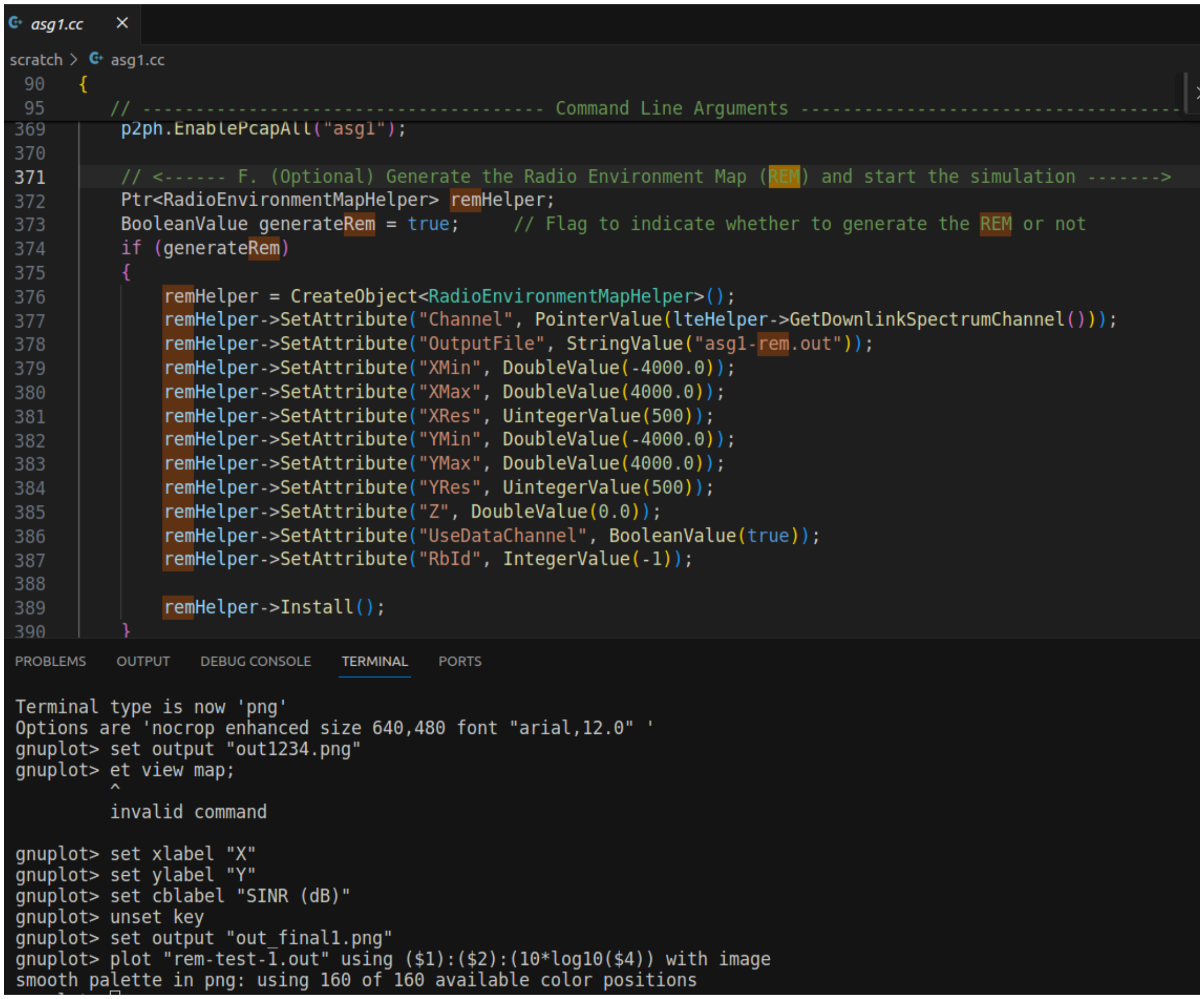
Observations:

Four High SINR Zones: There are four specific zones where the SINR values are notably high, aligning with the positions of the eNodeBs. These areas boast top-notch signal quality, as indicated by the brighter spots that signify proximity to the eNodeBs.

Symmetrical Distribution: The SINR distribution showcases a symmetrical pattern surrounding each eNodeB, hinting at a consistent propagation environment with minimal obstacles or terrain fluctuations.

SINR Gradient: As one moves away from the eNodeBs, the SINR values gradually decline. This decline is a natural outcome of signal strength weakening over distance.

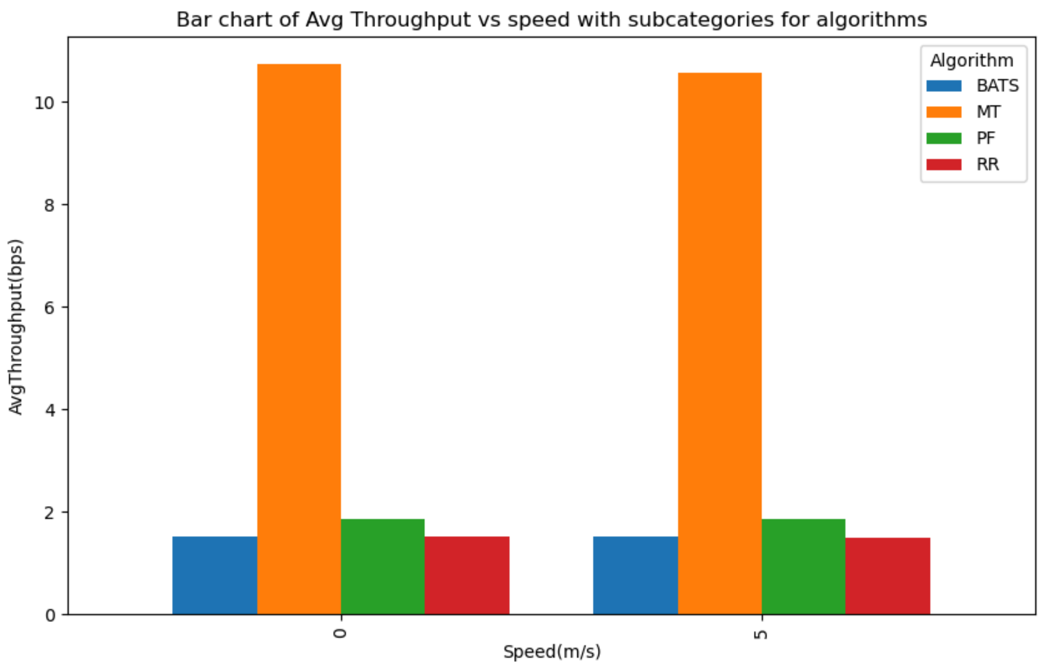
How we made SINR graph(using GNU plot)



```
{
    // ------------------------------------- Command Line Arguments ------------------------------------
    p2ph.EnablePcapAll("asg1");

    // <------ F. (Optional) Generate the Radio Environment Map (REM) and start the simulation ------->
    Ptr<RadioEnvironmentMapHelper> remHelper;
    BooleanValue generateRem = true;     // Flag to indicate whether to generate the REM or not
    if (generateRem)
    {
        remHelper = CreateObject<RadioEnvironmentMapHelper>();
        remHelper->SetAttribute("Channel", PointerValue(lteHelper->GetDownlinkSpectrumChannel()));
        remHelper->SetAttribute("OutputFile", StringValue("asg1-rem.out"));
        remHelper->SetAttribute("XMin", DoubleValue(-4000.0));
        remHelper->SetAttribute("XMax", DoubleValue(4000.0));
        remHelper->SetAttribute("XRes", UintegerValue(500));
        remHelper->SetAttribute("YMin", DoubleValue(-4000.0));
        remHelper->SetAttribute("YMax", DoubleValue(4000.0));
        remHelper->SetAttribute("YRes", UintegerValue(500));
        remHelper->SetAttribute("Z", DoubleValue(0.0));
        remHelper->SetAttribute("UseDataChannel", BooleanValue(true));
        remHelper->SetAttribute("RbId", IntegerValue(-1));

        remHelper->Install();
    }
```

```
Terminal type is now 'png'
Options are 'nocrop enhanced size 640,480 font "arial,12.0" '
gnuplot> set output "out1234.png"
gnuplot> et view map;
         ^
         invalid command

gnuplot> set xlabel "X"
gnuplot> set ylabel "Y"
gnuplot> set cblabel "SINR (dB)"
gnuplot> unset key
gnuplot> set output "out_final1.png"
gnuplot> plot "rem-test-1.out" using ($1):($2):(10*log10($4)) with image
smooth palette in png: using 160 of 160 available color positions
```

Graph 2: X-axis: Speed (0, 5) m/s; Y-axis: (Average Aggregate System throughput) with bars for four scheduler algorithms for full buffer scenario. Get sum of throughputs of all 4 cells (i.e., all 20 UEs flows) in different runs by varying seed values and then get the average of that for plotting.



Observations:

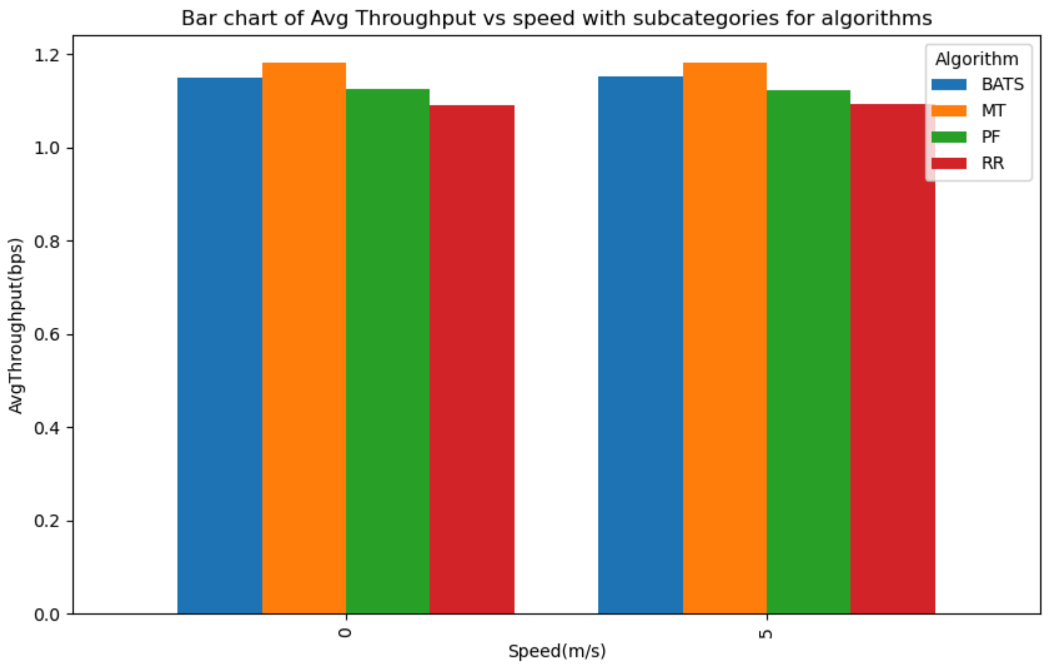
- BATS: At 0 m/s, the throughput is approximately 1 unit (Mbps), and there seems to be no change in throughput as the speed increases to 10 m/s.
- MT: The throughput at 0 m/s is just slightly above 1 unit, similar to BATS. However, unlike BATS, the MT algorithm experiences a dramatic increase to around 10 units at 10 m/s, indicating a significant improvement in performance with increased speed.
- PF: This algorithm has a throughput of around 2.5 units at 0 m/s and shows a minor increase to approximately 3 units at 10 m/s.
- RR: There's a throughput of about 1 unit at 0 m/s, which remains virtually unchanged at 10 m/s.

  Note: 1 unit is 1Mb/sec

Trends and Improvement in System Throughput:

- BATS and RR maintain a stable throughput as speed changes, suggesting that these algorithms are not sensitive to the changes in speed between 0 m/s and 10 m/s.
- MT shows a substantial improvement as speed increases, suggesting that this algorithm might be optimizing resources better at higher speeds.
- PF has a slight improvement with increased speed, indicating some level of optimization but not as pronounced as MT.

Non Full Buffer



Observations:

Analysis of the trends:

- BATS: The throughput remains consistent between 0 m/s and 10 m/s, with the value slightly above 1 unit. This suggests that the BATS algorithm's performance is not affected by the change in speed in this scenario.
- MT: The throughput is nearly identical to BATS at both speeds, also slightly above 1 unit. Like BATS, MT's performance seems unaffected by speed changes.

- PF: For this algorithm, the throughput is consistently around 1 unit at both 0 m/s and 10 m/s. There is no noticeable change, indicating stability in performance across these speeds.
- RR: The throughput for RR is very similar to both BATS and MT, remaining just above 1 unit and showing no significant change between the two speeds.

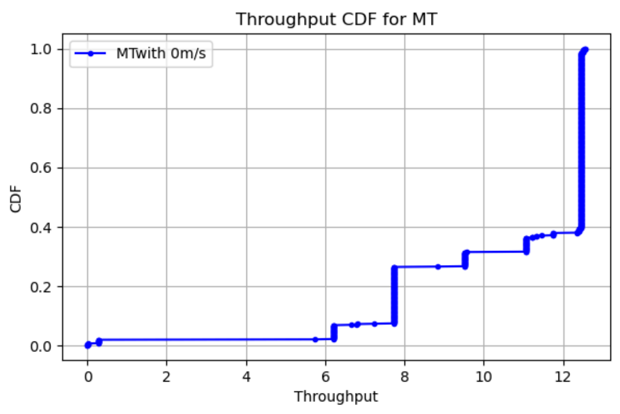
Observations and Trends:

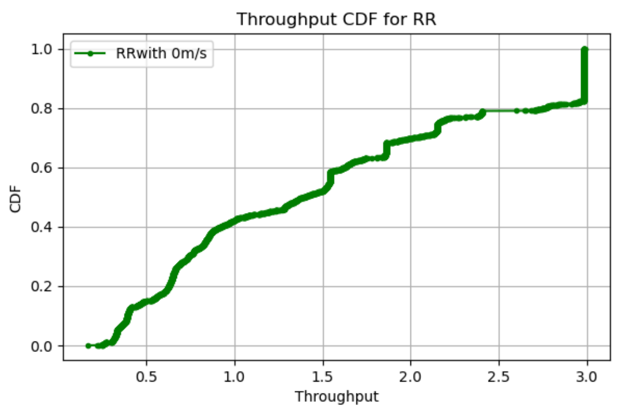
- All algorithms demonstrate stable throughput between the two speeds. Unlike the previous data, where MT showed a significant increase, in this data set, there is no change.
- The throughput values for all algorithms in this scenario are tightly grouped, with no algorithm outperforming or underperforming relative to the others between the two speeds.
- The consistent performance across different speeds may suggest that the algorithms are designed to cope with speed changes in this particular range.

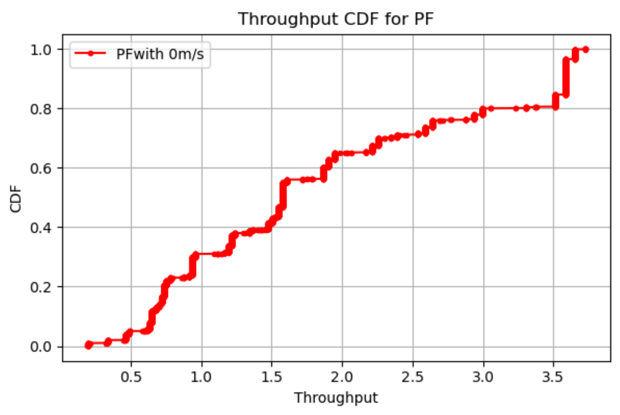
Graph 3: Throughput CDF plot for different schedulers at Speed (0,5) m/s for full buffer scenario; One curve each for 0 m/s and 5 m/s. But here you need not to do any averaging. Have list of per UE throughputs across all cells in all different runs by varying seed value and use that for plotting CDF.

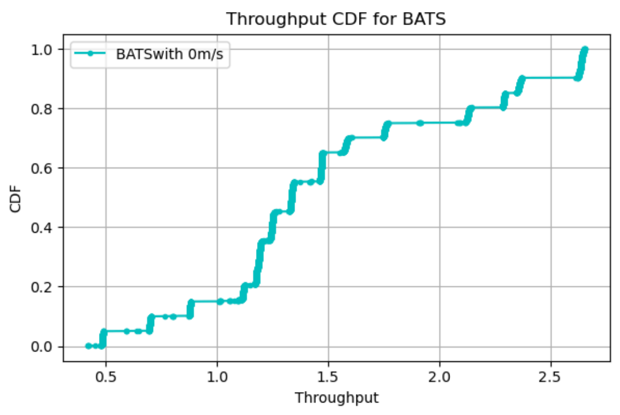
-Full Buffer case

For 0 m/s









Throughput CDF for MT (MTwith 0m/s):

- The CDF shows a step-like progression with a few significant jumps, suggesting a variation in throughput at different quantiles.
- A large proportion of the CDF is concentrated at higher throughput values, close to 10, with a sudden vertical rise indicating that a high throughput is achieved in many instances.

Throughput CDF for PF (PFwith 0m/s):

- The CDF rises steadily, suggesting a more even distribution of throughput values.
- Most of the data points are distributed between 1 and 3 throughput units.
- The CDF reaches 100% below 4 units of throughput, indicating that all users achieve less than this throughput value.

Throughput CDF for RR (RRwith 0m/s):

- The CDF curve is smooth and gradual, indicating a spread-out distribution of throughput values.
- The curve extends to just over 2.5 throughput units, suggesting that all users experience a throughput of 2.5 or less.

    - The gradual rise implies that the throughput increases slowly across users.

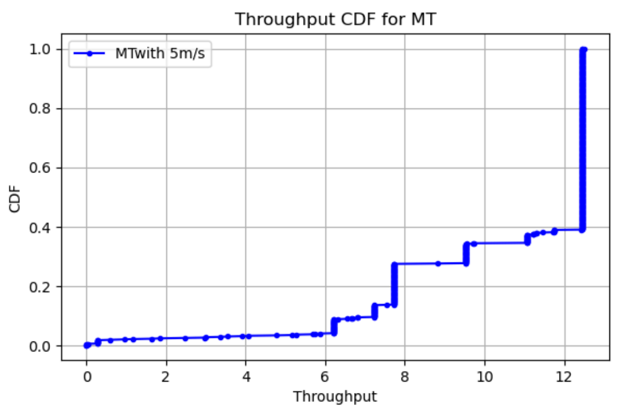
Throughput CDF for BATS (BATSwith 0m/s):

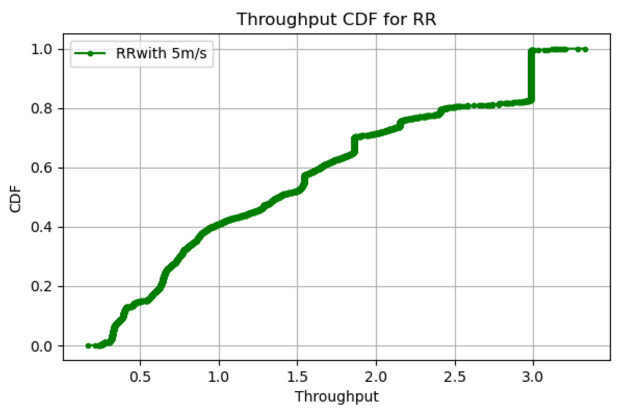
- The CDF shows a stepped progression with several jumps, similar to MT but with lower throughput values.
- The throughput values are concentrated below 2.5 units, and the curve reaches 100% before 3 units, indicating a lower throughput performance compared to MT.

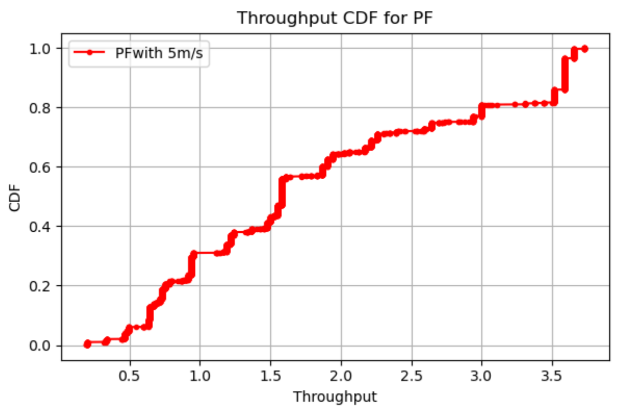
To quantify the improvement or change in system throughput for the scheduling algorithms at 0 m/s:

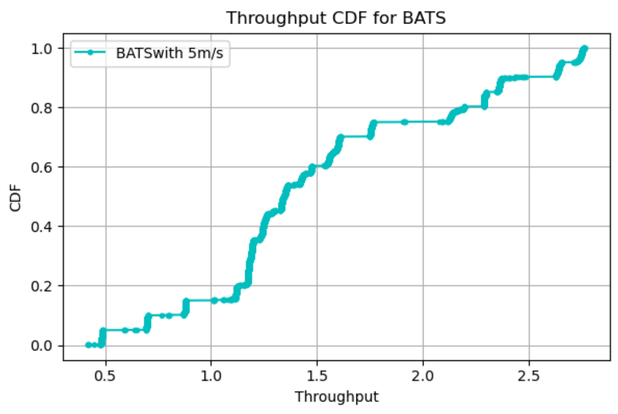
- MT: Almost all users achieve high throughput, close to the maximum of the scale, which is around 12 units.
- PF: The throughput is moderate, with all users achieving less than 4 units.
- RR: Throughput is distributed evenly, but all users are below 2.5 units, indicating lower performance.
- BATS: Similar to RR, all users achieve below 3 units of throughput.

For 5 m/s









MT (MTwith 5m/s vs. MTwith 0m/s):

- The CDF at 5 m/s shows a similar step-like progression to the one at 0 m/s.
- There's a slight leftward shift in the CDF at 5 m/s, indicating a decrease in throughput at higher quantiles compared to 0 m/s.
- The sharp rise to 100% around a throughput of 10 units is still present, suggesting that while there may be some impact due to speed, a large proportion of the throughput values remain high.

PF (PFwith 5m/s vs. PFwith 0m/s):

- The CDF at 5 m/s is similar to the CDF at 0 m/s, indicating stable throughput distribution.
- There does not appear to be a significant shift in the CDF, meaning the change in speed to 5 m/s has little impact on the throughput for PF.

RR (RRwith 5m/s vs. RRwith 0m/s):

- The CDF at 5 m/s shows a gradual progression similar to the one at 0 m/s, but with a slightly higher curve.

- This indicates an improvement in throughput at lower quantiles, as more users are reaching higher throughput values sooner than at 0 m/s.

BATS (BATSwith 5m/s vs. BATSwith 0m/s):

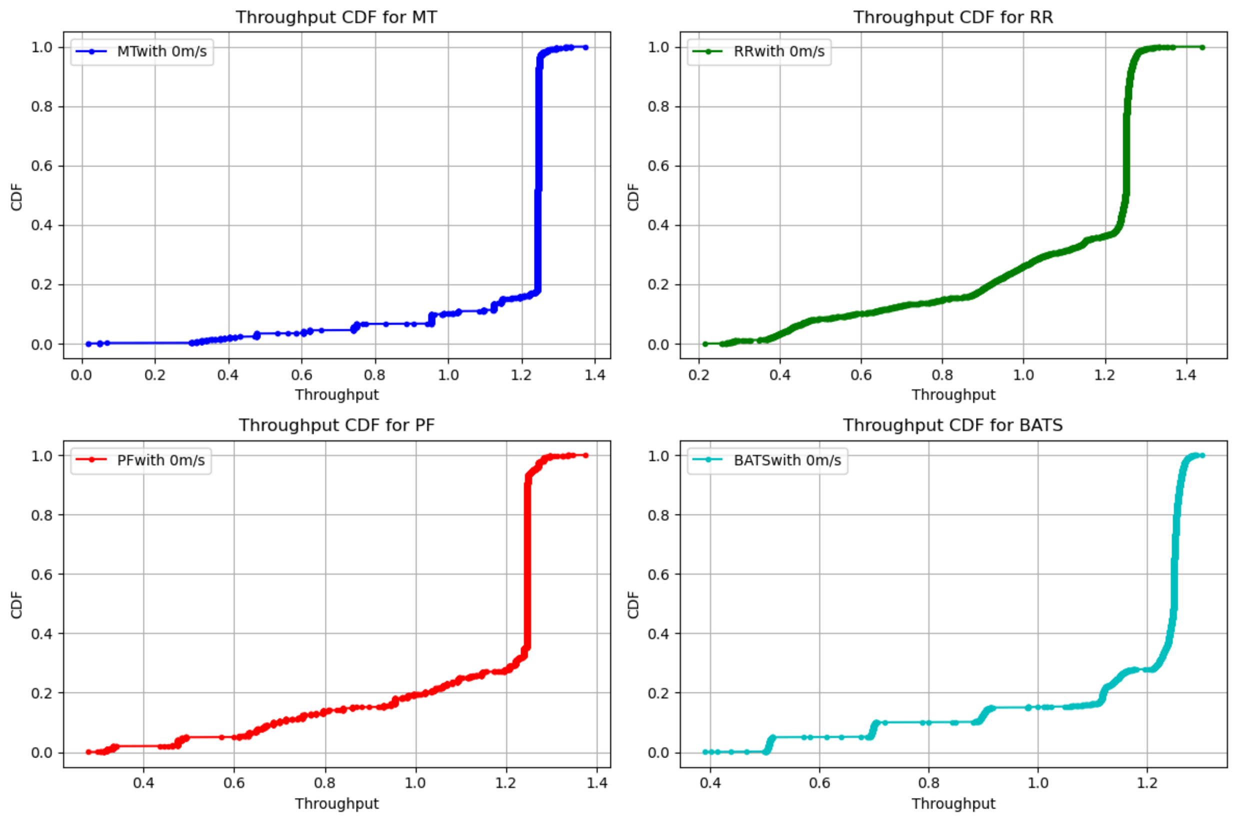
- The CDF at 5 m/s is again similar to the one at 0 m/s, with a slight rightward shift.
- This indicates an improvement in throughput across the board, as more users are reaching higher throughput values at 5 m/s compared to 0 m/s.

From these comparisons, we can conclude:

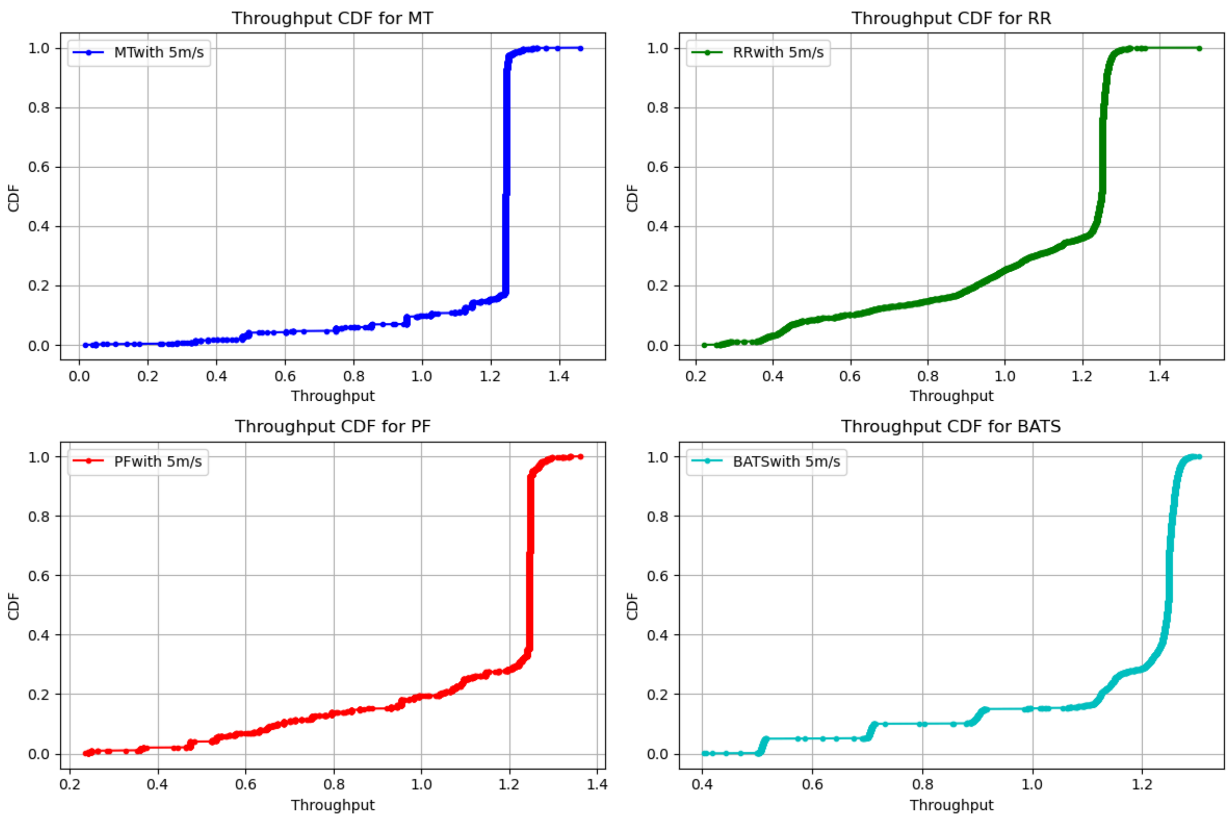
- MT: Slight decrease in throughput at 5 m/s, particularly for higher quantiles.
- PF: Throughput distribution remains stable with the increase in speed to 5 m/s.
- RR: Improvement in throughput at 5 m/s, especially for lower quantiles.
- BATS: General improvement in throughput at 5 m/s across all quantiles.

-Non-Full Buffer case.

For 0 m/s



For 5 m/s



MT Algorithm:

- At 0 m/s, almost all the values are concentrated at a throughput of 1.2.
- At 5 m/s, we see the same behavior, suggesting that the speed change does not affect the throughput for the MT algorithm.

PF Algorithm:

- At 0 m/s, there's an immediate jump in the CDF at 1.2, indicating that all throughput values are exactly or very close to 1.2.
- At 5 m/s, the behavior is identical to the CDF at 0 m/s. Therefore, the throughput for PF is also unaffected by the speed change.

RR Algorithm:

- At 0 m/s, the throughput gradually increases and gets steeper past 0.8, eventually reaching 1.2.
- At 5 m/s, the curve is slightly higher at lower throughputs and reaches 1.2 as well, indicating a slight improvement at 5 m/s, particularly for those previously at lower throughput values.

BATS Algorithm:

- At 0 m/s, the CDF steps up at various points and eventually reaches a throughput of 1.2.
- At 5 m/s, there's a rightward shift in the curve, with the CDF reaching 1.2, suggesting that at 5 m/s, users generally achieve slightly higher throughput than at 0 m/s.

From these observations, we can deduce the following:

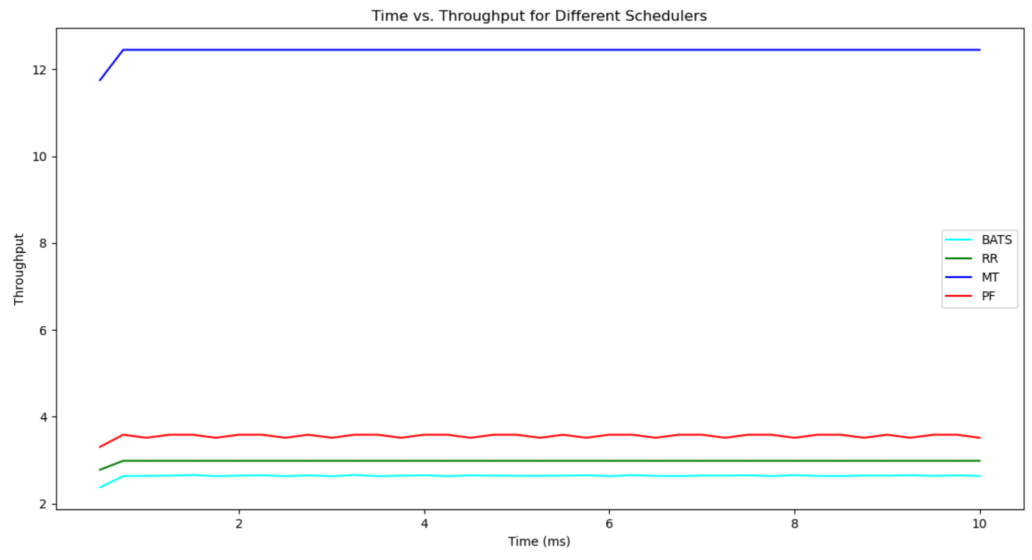
- MT and PF algorithms show an invariant throughput distribution relative to the speed change from 0 m/s to 5 m/s. This could indicate robustness to speed changes within this range, or it could suggest that the algorithms have reached a throughput cap that does not vary with speed.
- RR algorithm demonstrates a modest improvement with speed, particularly beneficial for users who had lower throughput at 0 m/s.
- BATS also shows a slight overall improvement at 5 m/s compared to 0 m/s, indicating that this algorithm adapts positively to an increase in speed.

Graph 4: SINR/Instantaneous throughput values for UE 0 in the simulation for one seed (RngRun1).X-axis: Time in msec, Y-axis: SINR and Instantaneous throughputs of UE0 for Speed of 0 m/s for all four schedulers for full buffer scenario. Refer Help section at the end of this document to know how to measure Instantaneous throughputs.

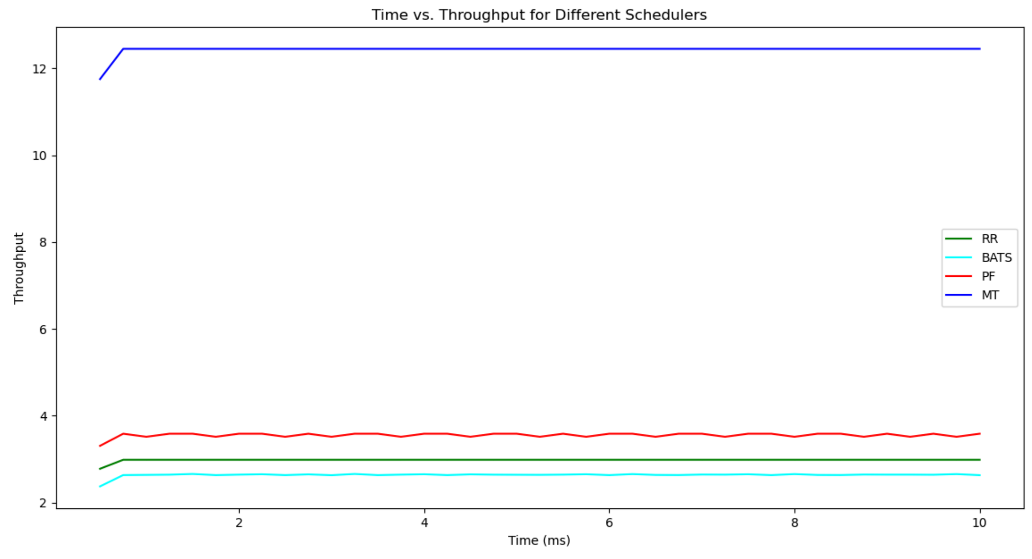
Graph 5: SINR/Instantaneous throughput values for UE 0 in the simulation for one seed (RngRun1). X-axis: Time in msec, Y-axis: SINR and Instantaneous throughputs of UE0 for Speed of 5 m/s for all four schedulers for full buffer scenario

1-Full Buffer Case

For 0m/s



For 5 m/s



Observations at 0 m/s (First Image):

- MT: It quickly reaches a throughput of just above 12 and maintains that level consistently.
- PF: It remains steady at a level just above 4.
- BATS: Throughput stays at about 3.

- RR: This algorithm maintains a throughput just above the 2 mark.

Observations at 5 m/s (Second Image):

- MT: The performance is consistent with the 0 m/s case, maintaining a throughput just above 12.
- PF: Throughput is just above 4, consistent with the 0 m/s case.
- BATS: Similarly, BATS maintains a throughput at about 3, as in the 0 m/s case.
- RR: The throughput for RR also remains steady at just above 2, as in the 0 m/s scenario.

Analysis of Trends:

- Across both speeds, the MT algorithm exhibits the highest throughput, which significantly outperforms the other three algorithms.
- The PF algorithm shows a consistently moderate level of throughput.
- BATS and RR have the lowest throughput among the four algorithms.
- The throughput for each algorithm does not seem to be affected by the change in speed from 0 m/s to 5 m/s, indicating that their performance is stable under these conditions.

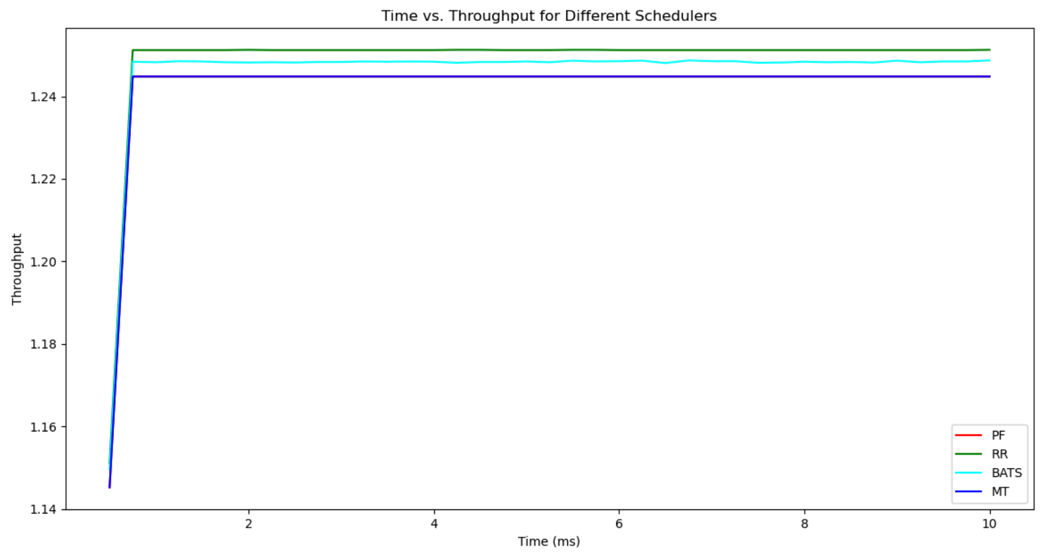
Quantification of Throughput Improvement:

Since the throughput for each algorithm at 0 m/s and 5 m/s remains consistent, there is no observed improvement or degradation to quantify. The throughput values are as follows:

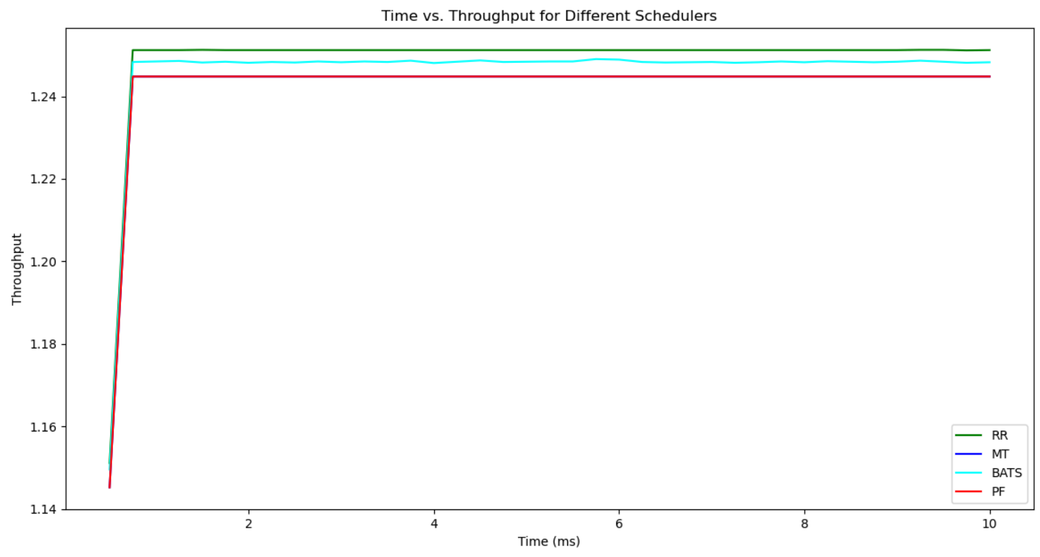
- MT: Approximately 12 units for both 0 m/s and 5 m/s.
- PF: Approximately 4 units for both 0 m/s and 5 m/s.
- BATS: Approximately 3 units for both 0 m/s and 5 m/s.
- RR: Approximately 2 units for both 0 m/s and 5 m/s

2-Non Full Buffer

For 0m/s



For 5 m/s



NOTE: PF and MT are overlapping because of the akin values in case of non full buffer

Observations at 0 m/s

The throughput for all scheduling algorithms seems to stabilize quickly within the first few milliseconds.

- All algorithms converge to a throughput value of approximately 1.24 and maintain that level consistently over time.
- There is no variation or crossing of lines after the initial period, which indicates that the throughput for each scheduler remains constant after the initial spike.

Observations at 5 m/s :

- Similar to the 0 m/s scenario, all scheduling algorithms quickly reach a stable throughput, but the convergence point appears slightly lower, at around 1.22.
- After the initial stabilization, all lines remain parallel, indicating no change in relative throughput performance over time.

Analysis of Trends and System Throughput Improvement:

- Stabilization Speed: In both cases, all algorithms reach their stable throughput almost instantaneously. This suggests that the algorithms are very responsive to changes and stabilize quickly.
- Speed Impact: Comparing the throughput values at 0 m/s and 5 m/s, there is a slight decrease in stable throughput values when the speed is increased to 5 m/s (from 1.24 to 1.22). This suggests that the increase in mobility slightly impacts the system throughput.

References:
1. https://www.nsnam.org/docs/models/html/lte-design.html#mac
2. https://www.nsnam.org/docs/models/html/lte-user.html#radio-environment-maps
3. https://www.nsnam.org/docs/models/html/lte-user.html
4. https://www.nsnam.org/docs/models/html/lte-design.html#round-robin-rr-scheduler
5. http://code.nsnam.org/ns-3-dev/file/028452e3b558/src/lte/examples/lena-rem.cc
6.http://code.nsnam.org/ns-3-dev/file/028452e3b558/src/lte/examples/lena-intercell-interference.cc

Note: All the plotting scripts, helping scripts, main program, automation script, graphs, plotting data, readme.md and extracted data is given.