EE550 - Data Networks Design and Analysis
Spring 2023
Project Report

Submitted by
Rahul Somayaji and **Aditya Dhabekar**
rsomayaj@usc.edu , dhabekar@usc.edu

Optimization of data rate of n-channel wireless network system using Drift Plus Penalty Algorithm

Motivation for this project : -

Primarily to formulate a convex program to maximize network utility of multiple channels of a wireless access point subject to Power and delay queue constraints using Drift Plus Penalty Algorithm for solving convex programs.

Problem Definition: -

Maximize the sum of output data rates received at the wireless devices accessing the Wifi Access Point: -

$$\textbf{Max : } \sum_{i=1}^{n} ri$$

$$\textbf{Subject to: } 1) \sum_{i=1}^{n} pi <= a$$

2) 20<pi<200mW, $\forall\, i \in \{1,\ldots\ldots, n\}$

3) $\lambda i <= log(1 + \gamma ipi)$, where $ri = log(1 + \gamma ipi)$, $\forall\, i \in \{1,\ldots\ldots, n\}$

4) $\lambda i <= c(ri - \lambda i)$, where $(\lambda i/(ri - \lambda i)) = c$, $\forall\, i \in \{1,\ldots\ldots, n\}$

5) ri∈[(1+c)λi/c,b]

Where the following parameters ,consisting of a Wireless Network deployed in a given area and '**n**' Wireless devices connected to the network, are considered in our network model.

1. Received output rate (ri)
2. Power (pi)
3. Number of users accessing the network resource at a time (n).
4. For downlink case
5. Input data rate λi
6. Channel attenuation γi

7. Delay c

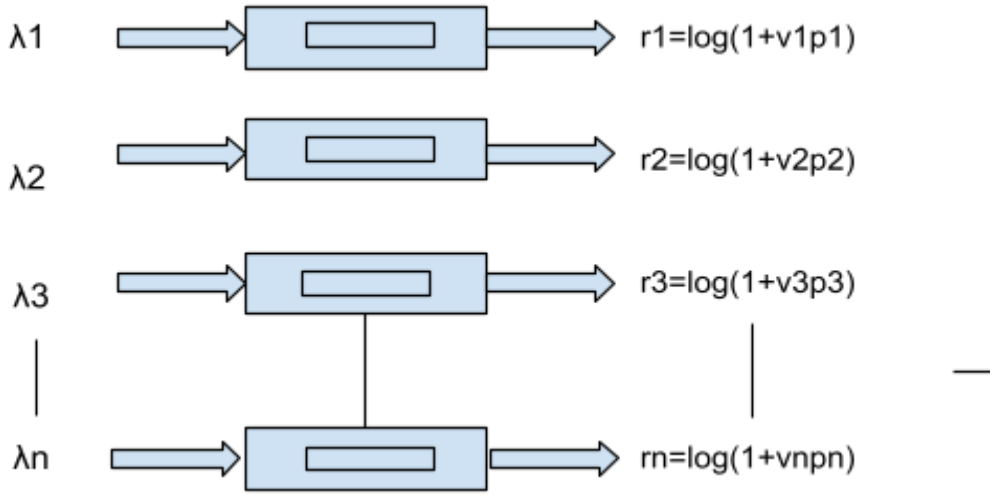

Figure 1: Flow allocation model

Results: -

Given : $(\lambda 1, \lambda 2, \lambda 3, \ldots\ldots, \lambda n)$, *a,b,c* , $(\gamma 1, \gamma 2, \gamma 3, \ldots\ldots\ldots\ldots, \gamma n)$
Decisions : *(P1,P2,P3,.....,Pn), (ri,r2,r3,.....,rn)*

Virtual Queues:

$Q(t+1) = max[Q(t) + \sum_{i=1}^{n} Pi(t) - a, 0]$

$H_i(t+1) = max[H_i(t) + r_i(t) - log(1 + \gamma ipi(t)), 0], \forall\, i \in \{1, \ldots\ldots, n\}$

Decisions (V>0):

$-V \sum_{i=1}^{n} ri(t) + Q(t)[\sum_{i=1}^{n} Pi(t) - a] + \sum_{i=1}^{n} H_i(t)[r_i(t) - log(1 + \gamma ipi(t))] + \sum_{i=1}^{n} Z_i(t)[\lambda i - (c(ri(t) - \lambda i))]$

$The\ above\ equation\ is\ separable\ \forall\, i \in \{1, \ldots\ldots, n\}$

$-Vri(t) + Q(t)[Pi(t) - a] + H_i(t)[r_i(t) - log(1 + \gamma ipi(t))]$

Differentiating w.r.to pi ***and equating to 0 we get: -***

$Q_i(t) + H_i(t) \times [-\gamma i/1 + \gamma ipi(t)] = 0$

$Q_i(t) = H_i(t) \times [\gamma i/1 + \gamma ipi(t)]$

$1 + \gamma ipi(t)$ = *[γi x $H_i(t)$ / $Q_i(t)$]*

$\gamma ipi(t)$ = *[γi x $H_i(t)$ / $Q_i(t)$] - 1*

$pi * (t) =$ *[($H_i(t)$ / $Q_i(t)$) - 1 / γi] $pi^* \in [0,a]$*

Differentiating w.r.to r_i

And equating to 0 we get: -

-V + $H_i(t) = 0$

$V = H_i(t)$; ri* ∈ [(1+c)λ*i*/c,b]

Therefore:

Min: ri(t)(Hi(t)-V)
S.t.: ri ∈ [(1+c)λ*i*/c,b]

Observations: -

Simulation of the above problem using Python: -

I. **Test Case 1:**

The Convex program used to simulate the scenario is as follows: -

Max : $\sum_{i=1}^{30} ri$

Subject to: 1) $\sum_{i=1}^{30} pi <= 1.5W$

2) 20mW<pi<200 mW

, $\forall\, i \in \{1,\ldots..,30\}$

3) $\lambda i <= log(1 + \gamma ipi)$, where $ri = log(1 + \gamma ipi)$,

$\gamma i = [1000, 10000]\ in\ increments\ of\ 10,\ \forall\, i \in \{1,\ldots..,30\}$

4) $\lambda i <= c(ri - \lambda i)$, where $(\lambda i/(ri - \lambda i)) =$10 msec , $\forall\, i \in \{1,\ldots..,30\}$

5) ri∈[(1+c)λ*i*/c,b] ; c = 0.1,λ*i=1*10^6*, b=0.5*10^6

Plots obtained from the Test Case 1: -

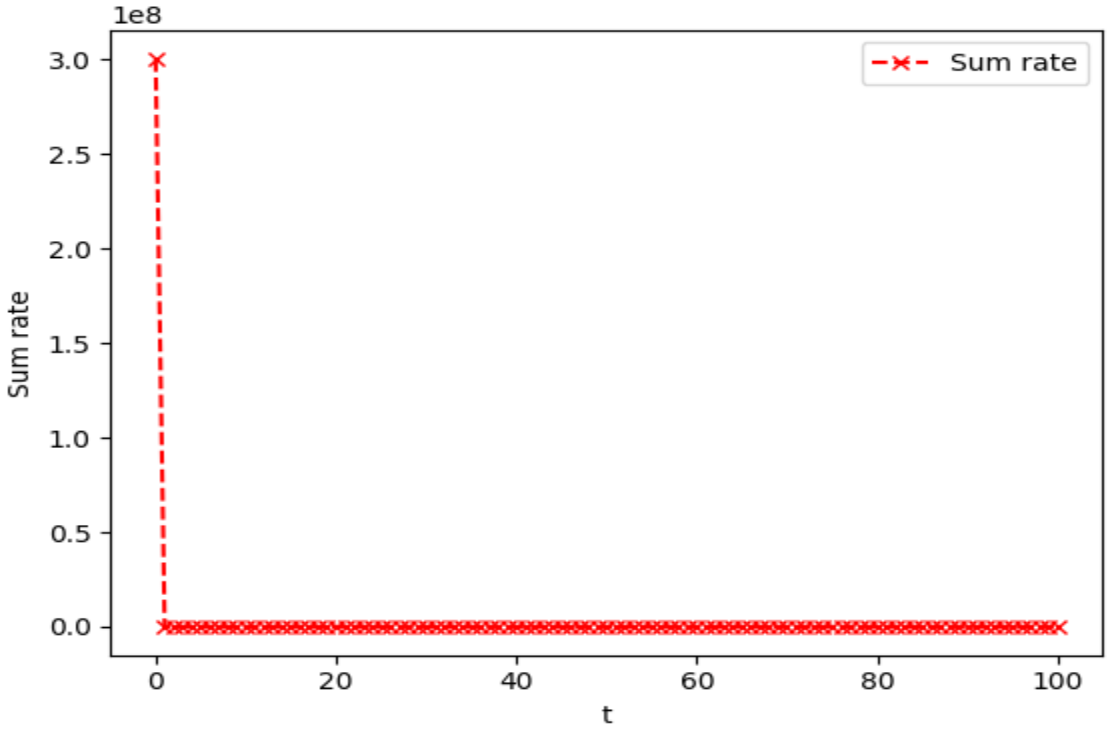

Figure 2: Plot of Sum data rate (in bps) vs time ‘t’ for test case 1

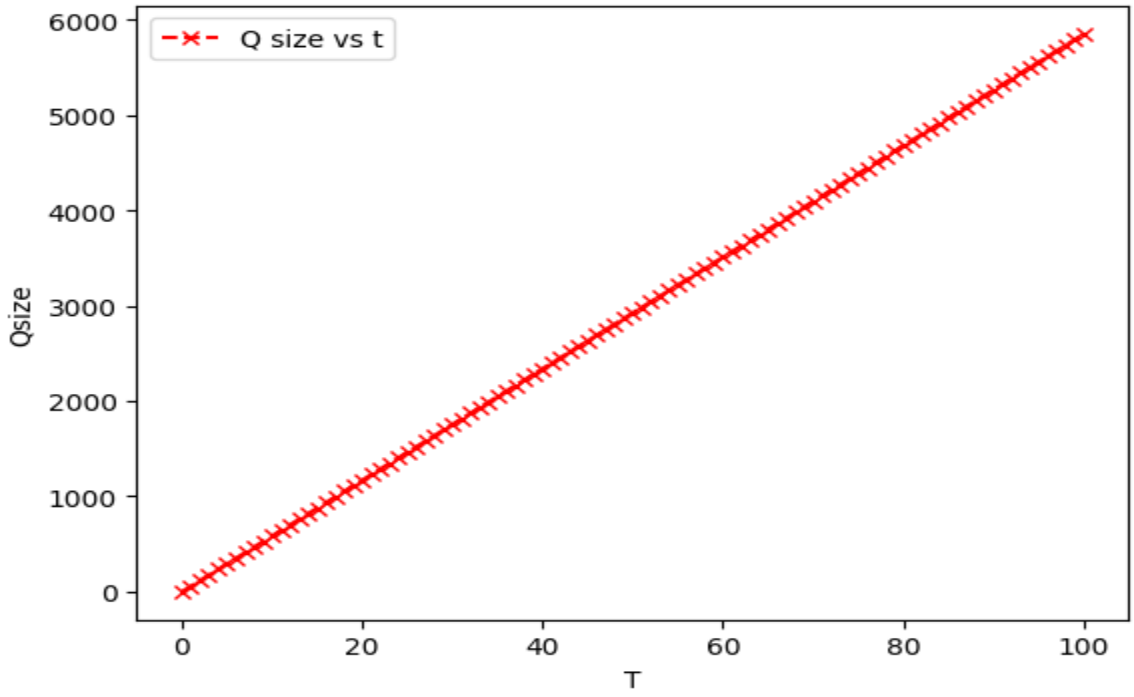

Figure 3: Plot of Queue size of queue ‘Q’ vs time ‘t’ for test case 1

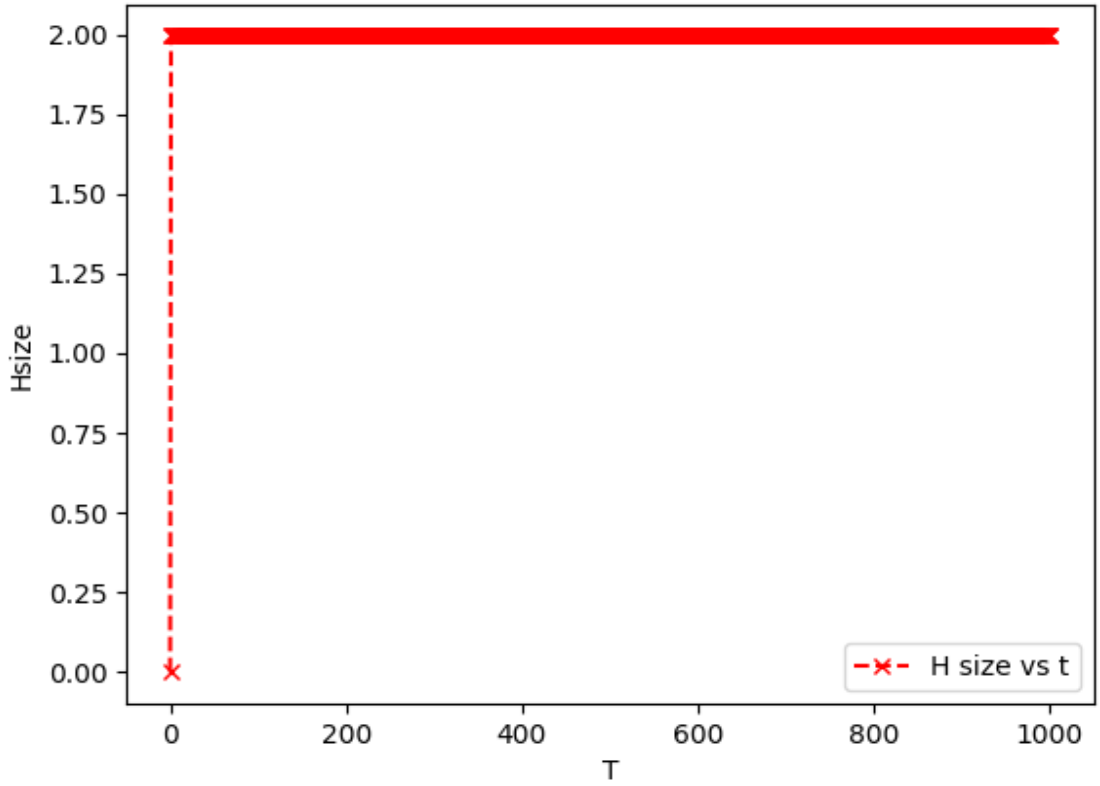

Figure 4: Plot of Queue size of queue ‘H’ vs time ‘t’ test case 1

Observations for Test Case 1:

After simulating the above scenario for the given set of constraints used in the convex program we noticed that the queue size was increasing linearly and was not stabilizing to a constant value for any number of timeslots run. Hence these constraints used to maximize the sum data rates using Drift Plus penalty algorithm are not feasible.

II. Test Case 2:

Since, the constraints in Test Case 1 were not feasible to optimize the framed Convex program using the Drift Plus Penalty algorithm we changed the constraints in the current test run. The modified convex optimization problem with new constraints is as follows:

$$\textbf{Max : } \sum_{i=1}^{30} ri$$

$$\textbf{Subject to: } 1) \sum_{i=1}^{30} pi <= 2.5W$$

2) 20mW<pi<200 mW

$$, \forall\, i \in \{1,\ldots\ldots, 30\}$$

$$3)\ \lambda i <= log(1 + \gamma ipi)\ \text{, where } ri = log(1 + \gamma ipi)\ ,$$

$$\gamma i = [1000, 10000]\ in\ increments\ of\ 10,\ \forall\, i \in \{1,\ldots\ldots, 30\}$$

$$4)\ \lambda i <= c(ri - \lambda i),\ \text{ where } (\lambda i/(ri - \lambda i)) = 80 \text{ msec}\ ,\ \forall\, i \in \{1,\ldots\ldots, 30\}$$

5) ri∈[(1+c)λi/c,b] ; c = 0.1, λi=*0.8*, b=4

Code (using Python): -

```
import numpy as np
import matplotlib.pyplot as plt

n = 30
t = 2000
Pi_min =0.02
Pi_max = 0.1
g = np.arange(1000, 10000, 10)
c = 0.8
ri_max=4
ri_min=(c+1)/c
r = np.zeros((n, t+1))
p = np.zeros((n, t+1))


p[:,0]= np.random.uniform(low=Pi_min, high=Pi_max, size=(n,))

Q = np.zeros(t + 1)
H = np.zeros((n, t + 1))
sump=np.zeros((1, t+1))
```

```
for i in range(0, t):
   for j in range(0, n):
       if Q[i] > 0:
           p[j][i] = ((H[j][i] / Q[i]) - (1 / g[j]))
           if (p[j][i]<=0):
               p[j][i]=Pi_min

           elif(p[j][i]>Pi_max):
               p[j][i]= Pi_max

       elif (Q[i] == 0):
           p[j][i] = Pi_max
       if (H[j][i] <= 0):
           r[j][i] = ri_max
       elif (H[j][i] > 0):
           r[j][i] = ri_min

       sump =np.sum(p,axis=0)
       pc= np.log(1 + g[j] * p[j][i])
       H[j][i + 1] = max(H[j][i] + r[j][i] - np.log(1 + g[j] * p[j][i]), 0)
   Q[i + 1] = max(Q[i] + sump[i] - 2.5, 0)

R = np.sum(r, axis=0)

print('Power:', p)
print('Data Rate:', r)
print('Sum Data rate:', R)
print('Queue value of Q:', Q)
print('Queue value of H:', H)

# Plot theoretical values

plt.plot(R, 'rx', label='Sum rate')

# Add axis labels and legend
plt.xlabel('t')
plt.ylabel('Sum rate (in bps)')
plt.legend()

plt.show()
plt.plot(Q, 'rx', label='Q size vs t')

# Add axis labels and legend
plt.xlabel('T')
plt.ylabel('Size of Queue Q')
plt.legend()
```

```
plt.show()
plt.plot(H[1], 'rx', label='H size vs t')
plt.xlabel('T')
plt.ylabel('Size of Queue H')
plt.legend()

plt.show()
```

Analysis: -

Plots obtained from the Test run 2: -

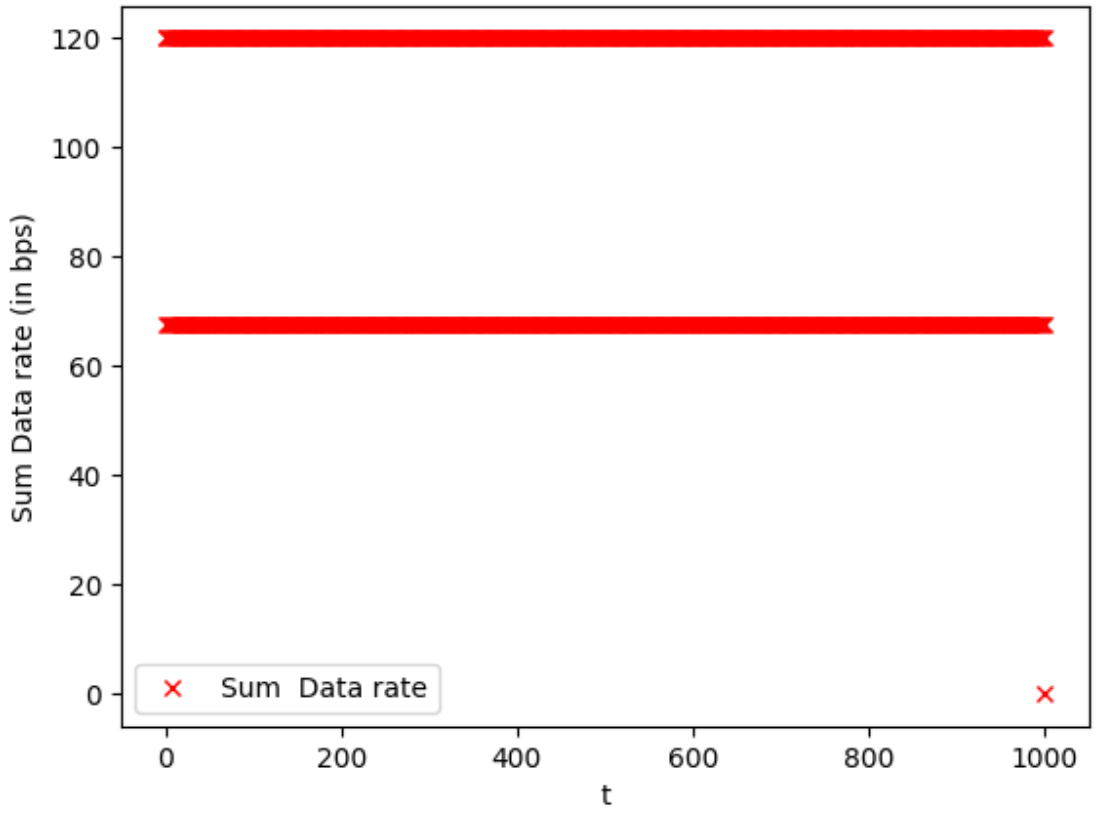

Figure 5: Plot of Sum of Data Rates (in bps) vs time ‘t’ for test case 2.

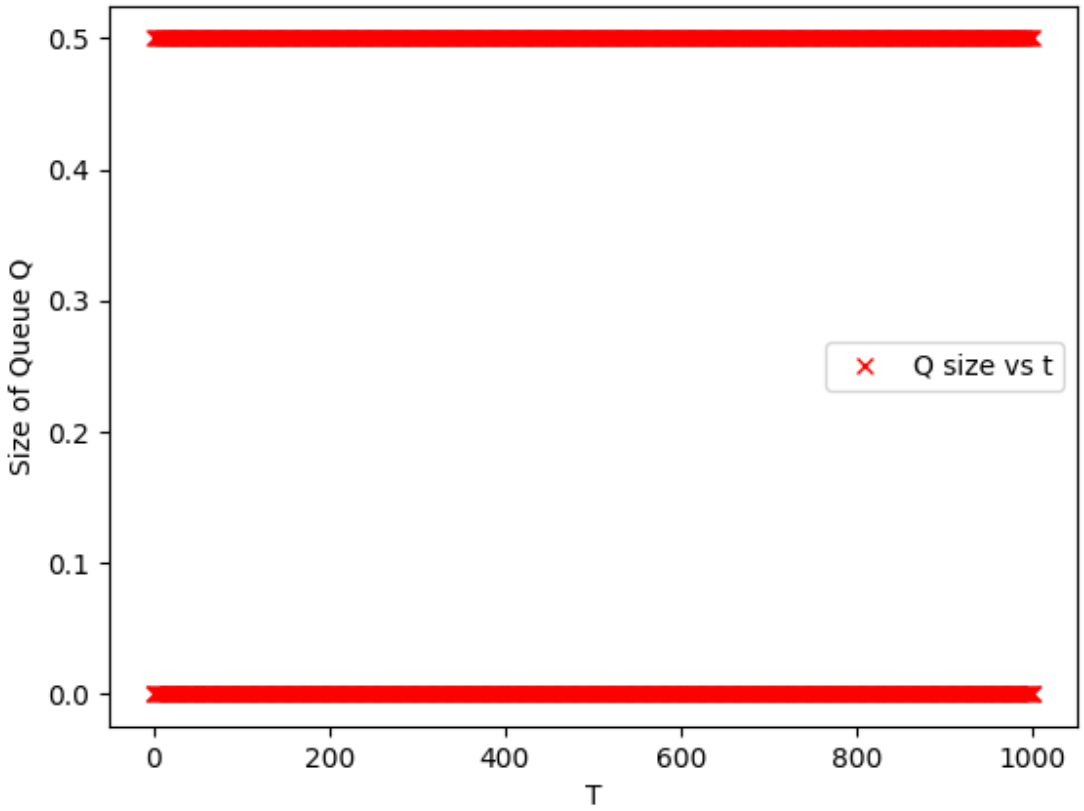

Figure 6: Plot of Queue size for queue ‘Q’ vs time ‘t’ for test case 2.

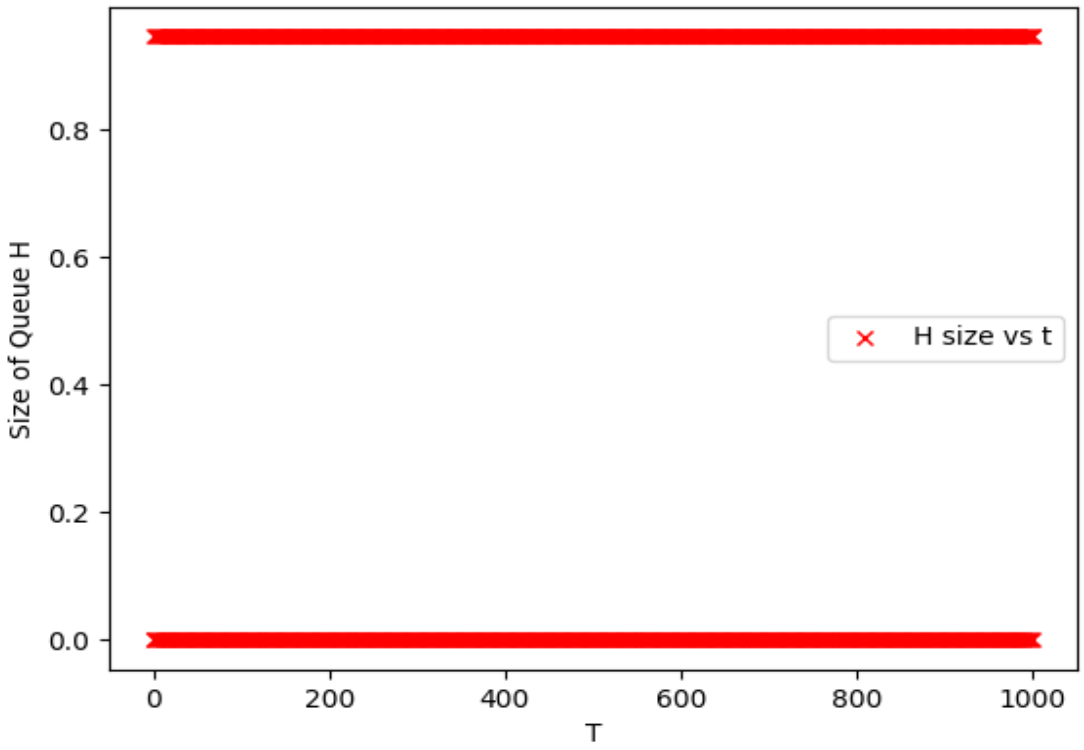

Figure 7: Plot of Queue size for queue 'H' vs time 't' for test case 2.

For the current test case with the modified constraints we can observe the changes in the plots for Queue sizes and the sum data rate with respect to time.

Description of the simulated algorithm is provided below:

We created two virtual queues Q[t] and H_i[t] for optimizing the power and sum of data rates across each channel present in the wireless environment.
So, the queue Q[t] is the virtual queue corresponding to the total power utilized at time 't' by all the wireless channels.
And the queue H_i[t] is the virtual queue corresponding to the data rate for each channel at time 't'.

The way the algorithm works for Queue size of queue 'Q' is when the value of Q = 0 at time t = 0, we put maximum data rate on each channel.
After that at each time slot 't' based on the updates in queue Q[t] and H_i[t] the optimal power put into each channel is decided using the relation: -

$$pi * (t) = [(H_i(t) / Q_i(t)) - 1 / \gamma i]$$

Corresponding to the optimal power values obtained from the above relation we put modified data rates on each channel which are calculated using the relation:

Min: *ri(t)(Hi(t)-V)*
S.t.: *ri* ∈ *[(1+c)λi/c,b]*

When (Hi(t)-V) = 0,
The data rate 'ri(t)' on the ith channel at the current time slot 't' takes on the maximum value 'b' as defined in the above constraints.

When (Hi(t)-V) > 0,
The data rate 'ri(t)' on the ith channel at the present time slot 't' takes on minimum value '(1+c)λ *i/*c' as defined in the above constraints.

Thus, we can observe from Figure 5 that the Sum data rate switches between **67.5 bps** and **120 bps** respectively.

The Queue size of queue 'H' varies based on the gamma values which are different for each channel. These gamma values indicate the attenuation constants used in the optimal rate calculation which are unique for each channel.
The relationship between power and data rate could be seen below:

$$ri(t) = log(1 + \gamma ipi(t))$$

We can see that on average Queue size of queue 'H' is **0.8** as observed in **Figure 7**.

Queue size of queue Q varies between **0** and **0.5** at each time slot 't' for the given constraints which is observable in **Figure 6**.

References: -

1. M.J.Neely. *Stability and Probability 1 Convergence for Queueing Networks via Lyapunov Optimization.* Hindawi Journal of Applied Mathematics. doi:10.1155/2012/831909, 2012

2. M.J. Neely. *A Simple Convergence Time Analysis of Drift-Plus-Penalty for Stochastic Optimization and Convex Programs*. ArXiv technical report, arXiv:1412.0791v1, Dec 2014

3. IEEE802.11ac article. Wikipedia. https://en.wikipedia.org/wiki/IEEE_802.11ac-2013