Network Systems Capstone @CS.NYCU

Lab5: Traffic generation and Random Backoff

Example Code

1. Generate initial packets for each station
2. For each time-slot
    a. Generate new arrival packets based on the <u>constant</u> inter-arrival time
    b. Assign each UE a <u>constant backoff counter</u>
    c. When the medium is idle, check who count to zero and have a buffered packet. Then, use a <u>constant probability to select a winner</u>
    d. Reset the backoff bounter to the <u>pre-defined constant</u> when count down to 0
    e. If no collision occurs, remove the successfully delivered packet
3. Calculate the number of Tx attempts and number of collisions
4. Calculate the average throughput

Parameters of Example Code

```
NumSta=[10:10:50];

SimuTime=1000;                % Simulation duration
SlotTime=1e-6;                % Duration of each timeslot
PktLen=180;                   % Number of bits per packet
DataRate=[6 9 12 18].*1e6;    % Available data rate (Mbps)
CWmin=32;                                         % Minimum contention window
CWmax=1024;                                       % Maximum contention window

for i=1:length(NumSta)
    nn = NumSta(i);
StaRate=DataRate(ceil(rand(1,nn)*4));   % Random rate of each station

lambda=ones(1,nn)*10000;                  % Mean arrival of stations
InterPktTime=(ones(1,nn)./lambda)/SlotTime; % Arrival time of the first packets
StaPktCnt=zeros(1,nn);                    % Accumulated packet count of each station
StaPktQ=[];                                   % Packet queue of each station
SentBitCnt=zeros(1,nn);                   % Number of sent bits per station

CW=ones(1,nn)*CWmin/8;                    % CW of each station
backoff=ones(1,nn).*CW;                   % Initial backoff counter

isBusy=0;           % flag denoting whether the medium is busy
NumTx=0;            % Accumulated number of transmissions
NumCollision=0;     % Accumulated number of collisions
```

Snapshot of Example Code

```
% TODO: modify the following block to enable CSMA/CA
% identify the station ID of senders
ix = find(backoff(uid)==0);
uid = uid(ix)
winner_ix = find(rand(1,length(uid)) < 0.2/nn*10) % TODO: now pick by probability. Should be removed
winner = uid(winner_ix)
% TODO END
```

Change random selection to random backoff

```
% TODO: update the backoff counter of each sender
% TODO: modify this based on exponential random backoff
backoff(uid) = CWmin/8
```

Update backoff bounder based on the Tx result

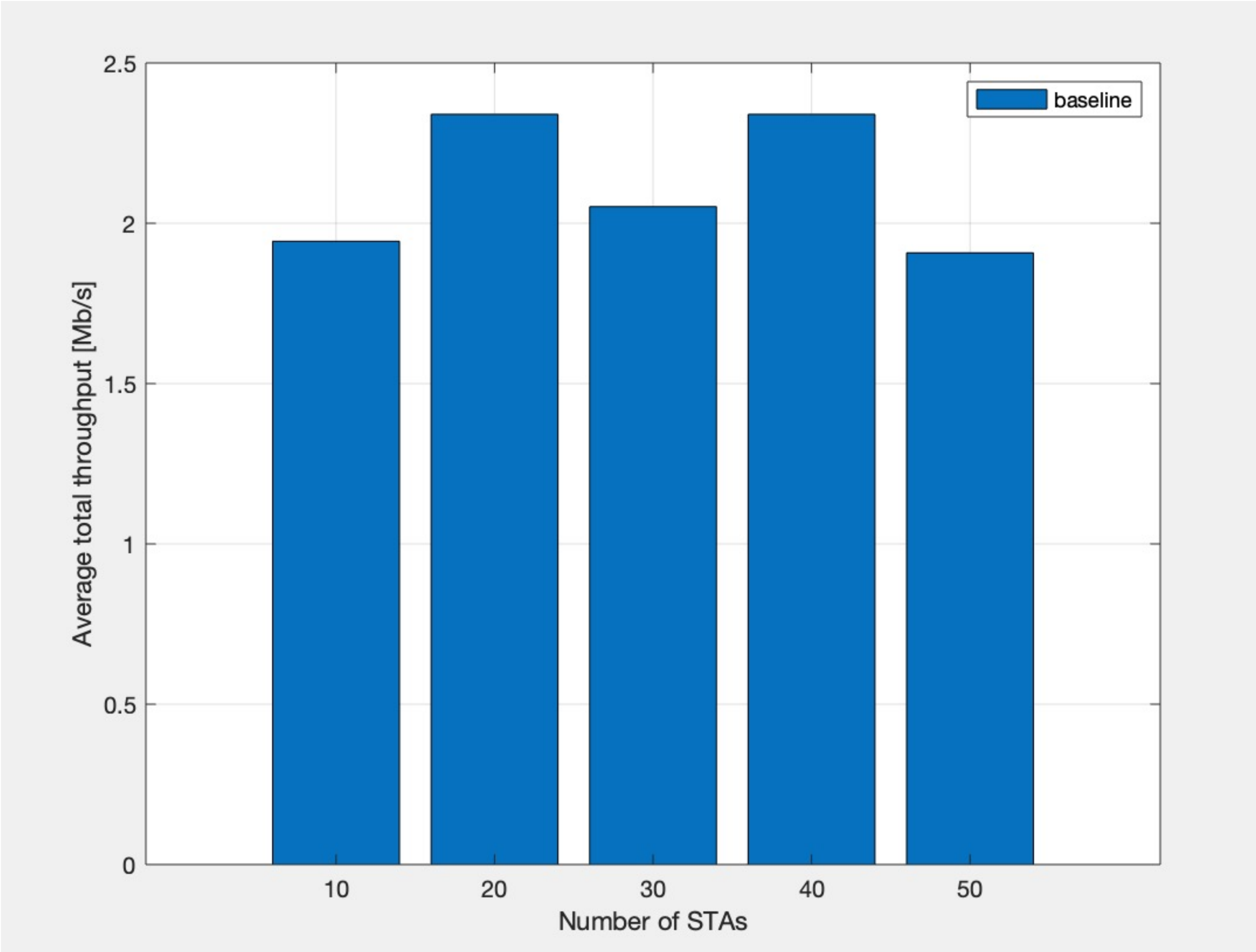
Snapshot of Example Code

```
function [InterPktTime, StaPktQ, StaPktCnt] = generate_pkt(InterPktTime, nn, PktLen, SlotTime, StaPk
InterPktTime = InterPktTime - 1;
for u=1:length(InterPktTime)
    if(floor(InterPktTime(u)) == 0)
        StaPktCnt(u) = StaPktCnt(u) + 1;
        StaPktQ(u,StaPktCnt(u)) = PktLen;
        InterPktTime(u)=1/lambda(u)/SlotTime;   % TODO: update the interarrival time here
        if (u == nn)          % for debugging
            StaPktCnt
        end
    end
end
return;
```

Update the inter-arrival time based on the Poisson dist. with mean arrival rate "lambda"

Output

- Average results of 5 iterations



Debugging

```
NumSta=[10:10:50];
NumIter=5;

SimuTime=1000;                          % Simulation duration
```

If you want to debug, change the configuration as follows

```
NumSta=[10:10:10];
NumIter=1;

SimuTime=500;                           % Simulation duration
```

TODO

Tasks

1. Generate initial packets for each station
2. For each time-slot
    a. Generate new arrival packets based on the **inter-arrival time of Poisson distribution (20%)**
    b. Assign each UE an **inirial random backoff counter (10%)**
    c. When the medium is idle, check who count to zero and have a buffered packet. **Everyon counts down to 0 will transmit (10%)**
    d. **Update the backoff counter using the exponential backoff algorithm based on the Tx result (20%)**
    e. If no collision occurs, remove the successfully delivered packet
3. Calculate the number of Tx attempts and number of collisions
4. Calculate the average throughput

Poisson Arrival

- Traffic arrival pattern follows the Poisson distribution with mean lambda
  - Check the ***CDF*** f(t) of the exponential dist. from WiKi
  - Uniformly pick a random number *p* from [0.1]
  - $t = f^{-1}(p)$

Random backoff in CSMA/CA

- Initialize the backoff counter to CW_{min}
- If the transmission is successful, $CW = CW_{min}$
- Otherwise, $CW = \min(CW * 2, CW_{max})$

- $CW_{min} = 32$
- $CW_{max} = 1024$

Code Submission

- Deadline: May. 27 (Thu.) 23:59
- Submit to new E3
  - Source code: lab5_<student_id>.m
  - Figure: rate_<student_id>.jpg
  - Report: report_<student_id>.jpg, **including a short discussion summarizing your observations**

Output

- Output
  - Average rate of the baseline and CSMA/CA
  - Figure: Rate bar graph
    - x-axis: number of stations
    - y-axis: average rate
    - Compare the baseline scheme with CSMA/CA

Grading

- Code (60%)
  - Traffic generation (20%)
  - Initial backoff counter (10%)
  - Winner selection (10%)
  - Backoff counter update (20%)
- Report (40%)
  - Figure (20%)
  - Discussion (20%)
- Late penalty
  - 20% off within 1 week of the deadline
  - After a week, 20% off per day