Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria en Ingenierías y Tecnologías Avanzadas

“UPIITA”

Ingeniería Telemática

Unidad de aprendizaje

Redes inteligentes

“Proyecto final”

Simulación de protocolo MAC

Profesora:

Villordo Jiménez Iclia

Integrantes:

- **López López Arturo**
- **Rodríguez Venegas Thomas Francisco**

Grupo: 1TM5

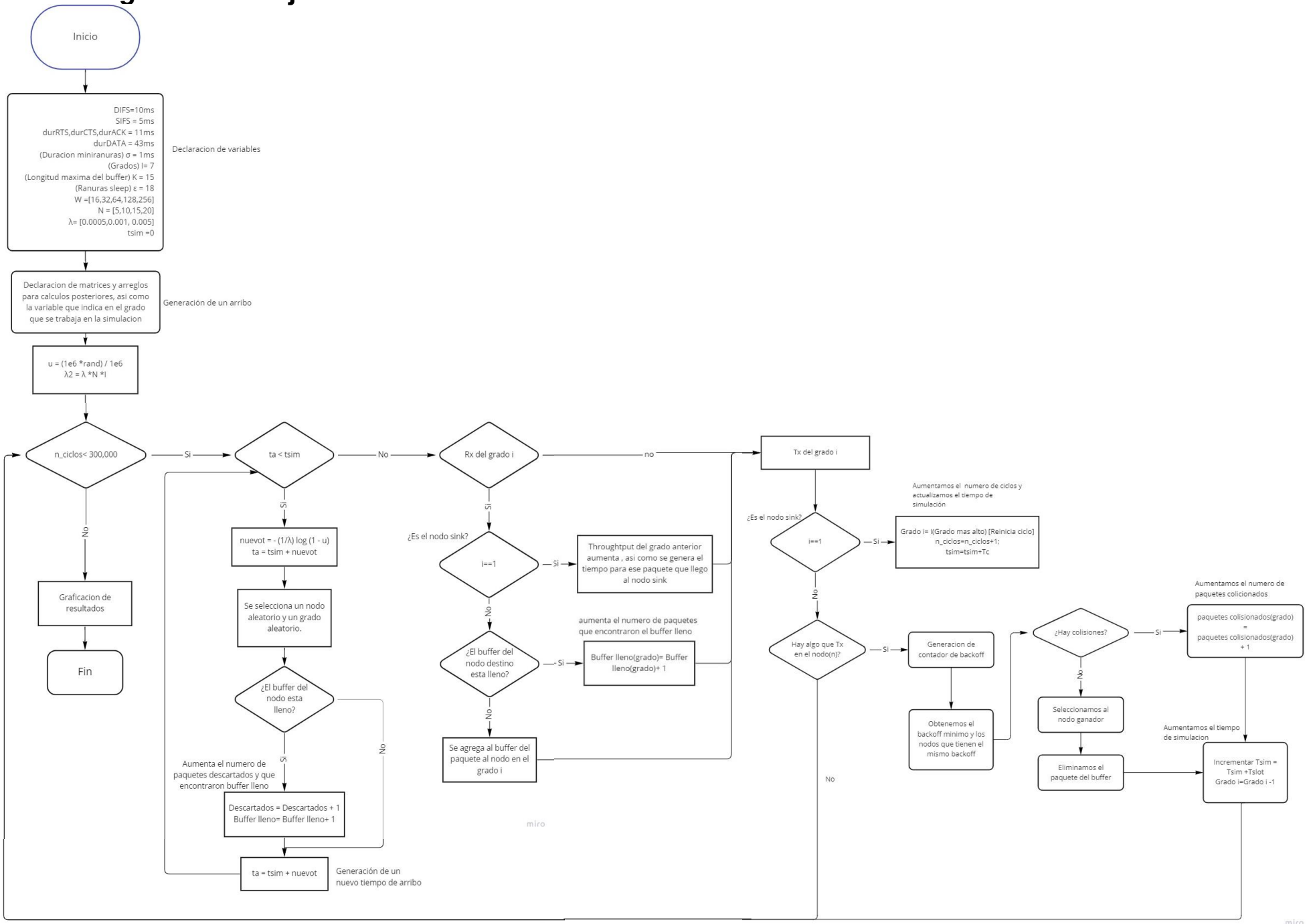
Introducción

En el documento que a continuación se muestra, presentamos un protocolo que opera dentro de ciclos de trabajo sincronizados con ayuda del software de programación Matlab, mostrando un proceso de trabajo donde los nodos despiertan selectivamente según la densidad de nodos y las condiciones en las que se encuentre su carga de tráfico.

La realización del presente proyecto conlleva a su análisis que pretende evaluar el desempeño general de la red en cada grado. Este análisis permite identificar aquellos grados específicos que pueden comprometer la vida útil y la confiabilidad de toda la red. El diseño de una red lineal de sensores LSN es otra forma de ver el funcionamiento de los paquetes cuando intentan ser transmitidos, recibidos o para su simple análisis de información.

Una vez mencionado que el protocolo propuesto está diseñado específicamente para trabajar en un LSN con un solo grado con nodos sink en un extremo de la red, y los nodos restantes son los encargados de generar nuevos paquetes y actuar como retransmisores para otros nodos sin realizar ningún procesamiento, agregación de datos o retransmisión de paquetes, procedemos a mostrar el diagrama de flujo correspondiente para observar con mejor detalle el funcionamiento del codigo.

Diagrama de flujo



Código

```
clear;
close all;
clc;

tic

DIFS = 10e-3;
SIFS = 5e-3;
durRTS = 11e-3;
durCTS = 11e-3;
durACK = 11e-3;
durDATA = 43e-3 ;
sigma = 1e-3 ;

I = 7;
K = 15;
Epsilon = 18;

W = [ 16, 32, 64, 128, 256];
N = [ 5, 10, 15, 20];
lambda = [ 0.0005, 0.005, 0.03 ] ;

W_index=3;
N_index=4;
lambda_index=1;

T=durDATA+durRTS+durCTS+DIFS+durACK+(sigma.*W(W_index))+(3*SIFS);
Tc = (2+Epsilon-I).*T;

tsim = 0;
ta=0;

N_zeros=3000000;
Buffer = zeros(K,N(N_index),I);
Pkt=zeros(N_zeros,6);
backoff=zeros(N(N_index));

a_colisiones = zeros(1,7);
aux_buffer_ = 0;


a_buffer_lleno=zeros(1,7);
descartado_grados=zeros(1,7);
```

```
RetardoTotal = zeros(1, 7);
aux_retardo=0;
g_ExitososVsPerdidos = zeros(1, 2);

n_ciclos=0;
n_paquetes=0;
n_colisiones=0;
n_paquetes_sink=0;
Throughput=zeros(1,I);
ranuras=1;
grado_iterable=I;
Rx_flag=false;
Tx_flag=true;

lambda_2=lambda(lambda_index)*N(N_index)*I;

while n_ciclos <300000

    while ta<=tsim

        nodo_random=randi(N(N_index),1);
        grado_random= randi([2 I],1) ;


        if Buffer(15,nodo_random,grado_random)==0
            n_paquetes=n_paquetes+1;
            Pkt(n_paquetes,1) = n_paquetes;
            Pkt(n_paquetes,2) = nodo_random;
            Pkt(n_paquetes,3) = grado_random;
            Pkt(n_paquetes,4) = ta;
            Buffer(15,nodo_random,grado_random)=n_paquetes;
            Buffer(:,nodo_random,grado_random)=FIFO_buffer(Buffer(:,nodo_ran
dom,grado_random));

        else
            for e=1:I
                if e==grado_random
                    a_buffer_lleno(e)=a_buffer_lleno(e)+1;
                    descartado_grados(e)=descartado_grados(e)+1;
                end
            end

        end

        U = (1e6*rand())/1e6;
```

```
        nuevot = -(1/lambda_2)*log(1-U);
        ta=tsim+nuevot;

    end


        if Rx_flag==true
            if grado_iterable~=1
                if Buffer(15,Colision(1),grado_iterable)==0
                    Buffer(15,Colision(1),grado_iterable)=Aux_n_pkt;
                    Buffer(:,Colision(1),grado_iterable)=FIFO_buffer(Buffer(
:,Colision(1),grado_iterable));
                    Throughput(grado_iterable+1)=Throughput(grado_iterable+1
)+1;

                else

                    Pkt(Aux_n_pkt,5)=3;

                    for e=1:I
                        if e==(grado_iterable+1)
                            a_buffer_lleno(e)=a_buffer_lleno(e)+1;
                        end
                    end
                end

            else
                    n_paquetes_sink=n_paquetes_sink+1;
                    Throughput(grado_iterable+1)=Throughput(grado_iterable+1
)+1;
                    Pkt(Aux_n_pkt,5)=1;
                    Pkt(Aux_n_pkt,6)=-Pkt(Aux_n_pkt,4)+tsim;

            end
        else
                Rx_flag=false;
        end

        if grado_iterable~=1
            backoff=zeros(1,N(N_index));
             for nodo=1:N(N_index)
                if Buffer(1,nodo,grado_iterable)~=0
                    backoff(nodo)=randi(W(W_index),1);
                else
                    backoff(nodo)=W(W_index)+1;
```

```
                end
            end

            backoff_min=min(backoff(:));

            if backoff_min~=(W(W_index)+1)

                Colision=find(backoff(:)==backoff_min);
                len_colision=length(Colision);

                if len_colision==1
                    Aux_n_pkt=Buffer(1,Colision(1),grado_iterable);
                    Buffer(1,Colision(1),grado_iterable)=0;
                    Buffer(:,Colision(1),grado_iterable)=FIFO_buffer(Buffer(
:,Colision(1),grado_iterable));
                    Rx_flag=true;

                    grado_iterable=grado_iterable-1;
                    tsim=tsim+T;

                else
                    for col=1:len_colision
                        n_colisiones=n_colisiones+1;

                        for e=1:I
                            if e==grado_iterable
                                a_colisiones(e)=a_colisiones(e)+1;
                            end
                        end

                        aux_index=Colision(col);
                        aux_colision=Buffer(1,aux_index,grado_iterable);
                        Pkt(aux_colision,5)=2;
                        Buffer(1,aux_index,grado_iterable)=0;
                        Buffer(:,aux_index,grado_iterable)=FIFO_buffer(Buffe
r(:,aux_index,grado_iterable));
                    end
                    Rx_flag=false;
                    grado_iterable=grado_iterable-1;
                    tsim=tsim+T;

                end
            else
                grado_iterable=grado_iterable-1;
                tsim=tsim+T;
```

```
                Rx_flag=false;

            end

        else
            grado_iterable=I;
            n_ciclos=n_ciclos+1;
            tsim=tsim+13*T;
            Rx_flag=false;

        end

end
toc



Pkt2=Pkt(1:n_paquetes,:);
retardo_por_grado_promedio=zeros(I,5);
for a=1:n_paquetes
    for e=2:I
        if  Pkt2(a,3)==e && Pkt2(a,5)==1
            retardo_por_grado_promedio(e,1)=retardo_por_grado_promedio(e,1)+
Pkt2(a,6);
            retardo_por_grado_promedio(e,2)=retardo_por_grado_promedio(e,2)+
1;
            break
        end
    end
end

for a=2:length(retardo_por_grado_promedio)
    retardo_por_grado_promedio(a,3)=retardo_por_grado_promedio(a,1)/retardo_
por_grado_promedio(a,2);
end


figure()
stem( Throughput, 'LineWidth',2)
xlim([0 8])
title('Throughput')
ylabel('Paquetes transmitidos exitosamente')
xlabel('Grado')
grid on
```

```
figure()
labels = { 'Grado 2', 'Grado 3', 'Grado 4', 'Grado 5', 'Grado 6','Grado 7'};
pie(Throughput(2:I), '%.3f%%')
title('Throughput')
lgd = legend(labels);

figure()
stem(descartado_grados, 'LineWidth',2)
xlim([0 8])
title('Paquetes descartados por grado')
ylabel('# paquetes descartados')
xlabel('Grado')
grid on

figure()
labels = {'Grado 1', 'Grado 2', 'Grado 3', 'Grado 4', 'Grado 5', 'Grado
6','Grado 7'};
pie(descartado_grados, '%.3f%%')
title('Paquetes descartados por grado')
lgd = legend(labels);

figure()
stem(a_colisiones, 'LineWidth',2)
xlim([0 8])
title('Paquetes colisionados')
ylabel('# paquetes colisionados')
xlabel('Grado')
grid on

figure()
labels = {'Grado 2', 'Grado 3', 'Grado 4', 'Grado 5', 'Grado 6','Grado 7'};
pie(a_colisiones(2:7), '%.3f%%')
title('Paquetes colisionados')
lgd = legend(labels);

figure()
stem(a_buffer_lleno, 'LineWidth',2)
xlim([0 8])
title('Paquetes que encontraron el buffer lleno')
ylabel('# paquetes sin Tx')
xlabel('Grado')
grid on


figure()
```

```
labels = {'Grado 1', 'Grado 2', 'Grado 3', 'Grado 4', 'Grado 5', 'Grado
6','Grado 7'};
pie(a_buffer_lleno, '%.3f%%')
title('Paquetes que encontraron el buffer lleno')
lgd = legend(labels);

figure()
stem(a_colisiones + a_buffer_lleno, 'LineWidth',2)
xlim([0 8])
title('Paquetes Tx perdidos por grado')
ylabel('# paquetes perdidos')
xlabel('Grado')
grid on

figure()
labels = {'Grado 2', 'Grado 3', 'Grado 4', 'Grado 5', 'Grado 6','Grado 7'};
pie(a_colisiones(2:I) + a_buffer_lleno(2:I), '%.3f%%')
title('Paquetes Tx perdidos')
lgd = legend(labels);

g_ExitososVsPerdidos(1, 1) = sum(a_buffer_lleno)+ sum(a_colisiones);
g_ExitososVsPerdidos(1, 2) = sum(Throughput);

figure()
labels = {'Perdidos', 'Exitosamente'};
pie(g_ExitososVsPerdidos, '%.3f%%')
title('Paquetes Tx')
lgd = legend(labels);

figure()
stem(retardo_por_grado_promedio(:,3), 'LineWidth',2)
xlim([0 8])
title('Retardo por grado')
ylabel('tiempo [ms]')
xlabel('Grado')
grid on

figure()
labels = { 'Grado 2', 'Grado 3', 'Grado 4', 'Grado 5', 'Grado 6','Grado 7'};
pie(retardo_por_grado_promedio(2:7,3), '%.3f%%')
title('Retardo por grado')
lgd = legend(labels);


function [Buffer]=FIFO_buffer(Buffer)
```

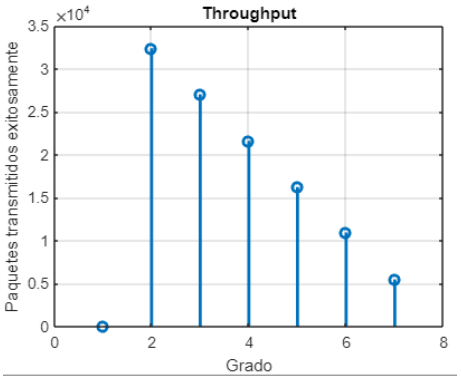
```
Buffer=Buffer.';
len_aux=length(Buffer);

Buffer=Buffer(Buffer~=0);
len_aux2=length(Buffer);

len_aux_faltante=len_aux-len_aux2;

Buffer=[Buffer zeros(1,len_aux_faltante)];
Buffer=Buffer.';

end
```

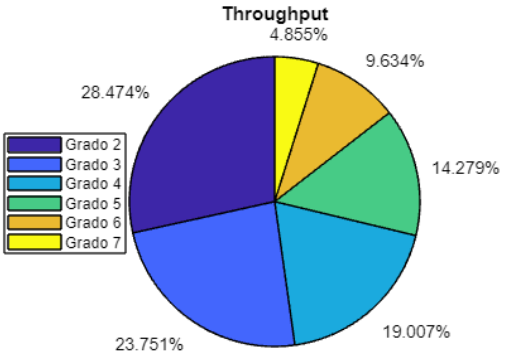
Resultados

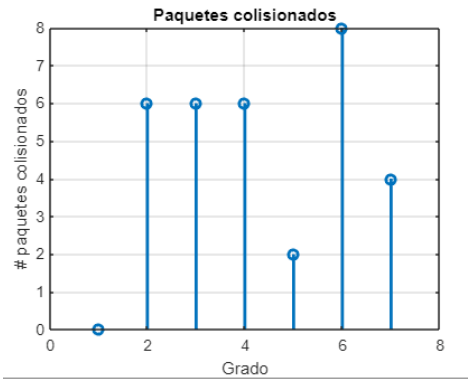
Una vez establecidas las variables con las que se inicializa nuestro programa de ejecución, procedemos a variar algunos aspectos, para poder analizar con mayor a profundidad, en que escenarios la simulación del protocolo de acceso al medio obtiene los mejores resultados.

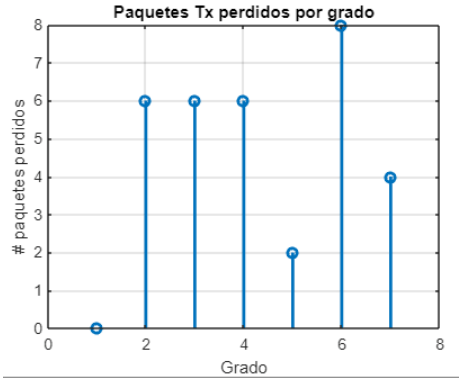
Una vez establecido la tasa de generación paquetes (λ), el número de nodos por grado (N) y un total de 7 grados (I), mostramos a continuación los resultados obtenidos con una generación de paquetes con los siguientes valores.

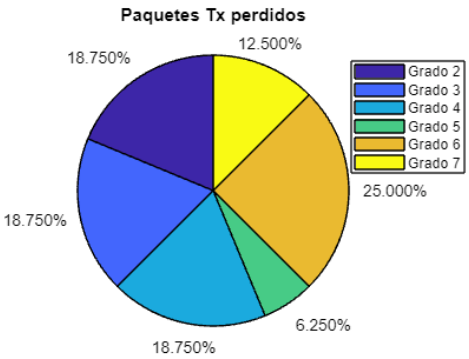
- $w = 64$
  - $N=10\ \lambda=0.0005$

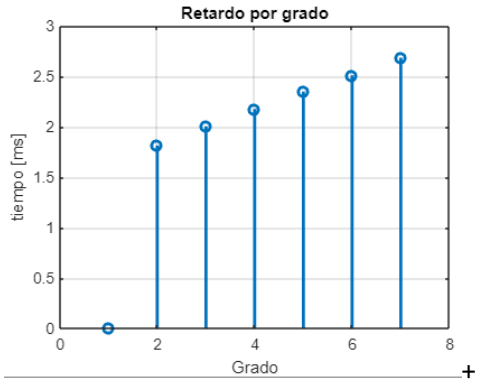


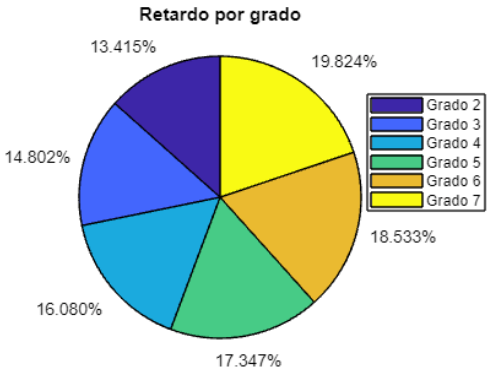


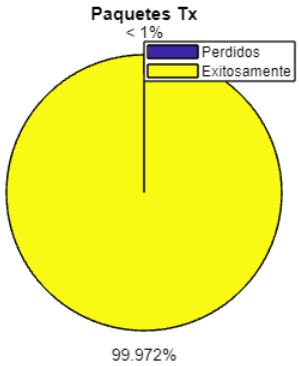




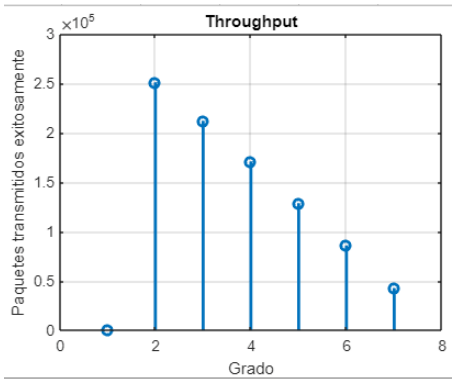


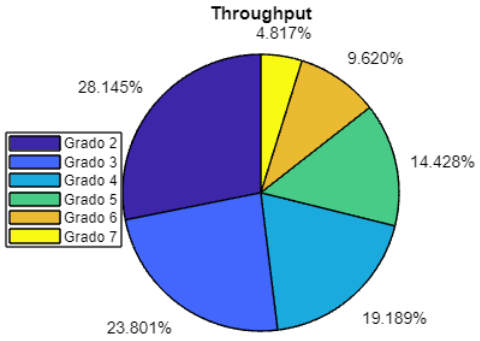

Retardo por grado
tiempo [ms]
Grado
0
0.5
1
1.5
2
2.5
3
0
2
4
6
8

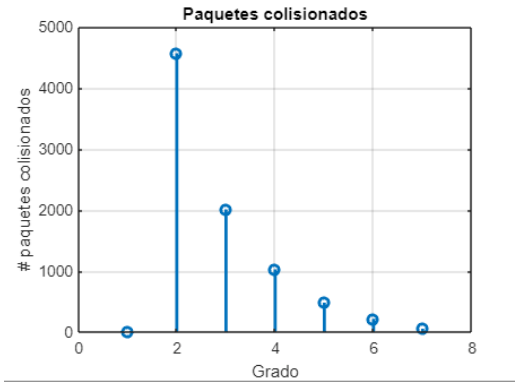

Retardo por grado
13.415%
19.824%
14.802%
18.533%
16.080%
17.347%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

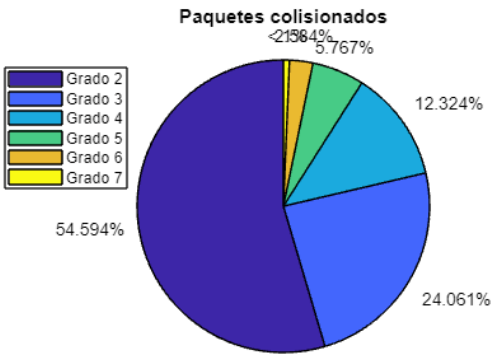

Paquetes Tx
< 1%
99.972%
Perdidos
Exitosamente

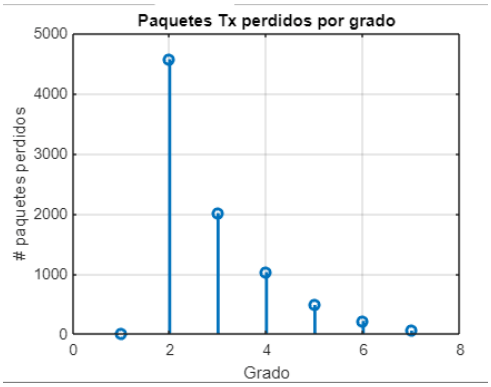
- N=10 λ=0.005

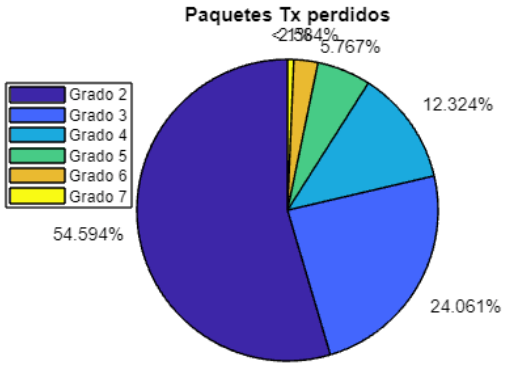

Throughput
×10^5
Paquetes transmitidos exitosamente
Grado
0
0.5
1
1.5
2
2.5
3
0
2
4
6
8

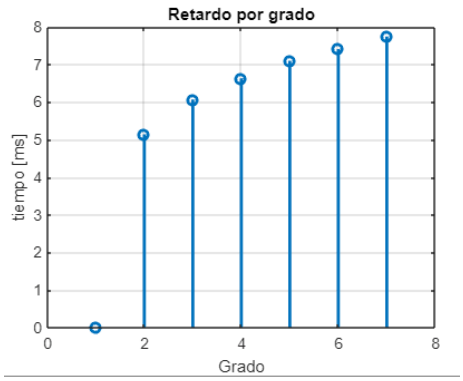

Throughput
4.817%
9.620%
28.145%
14.428%
23.801%
19.189%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

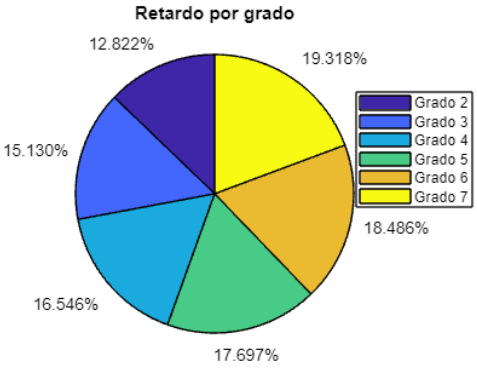

Paquetes colisionados
paquetes colisionados
Grado
0
1000
2000
3000
4000
5000
0
2
4
6
8

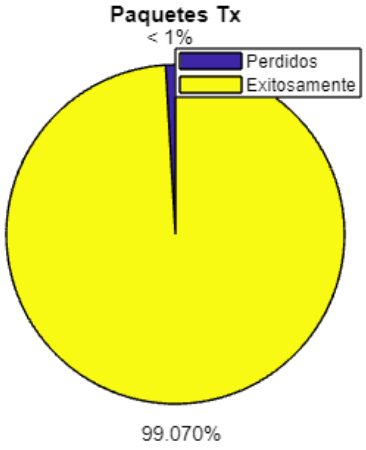

Paquetes colisionados
5.767%
12.324%
54.594%
24.061%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Paquetes Tx perdidos por grado
paquetes perdidos
Grado
0
1000
2000
3000
4000
5000
0
2
4
6
8

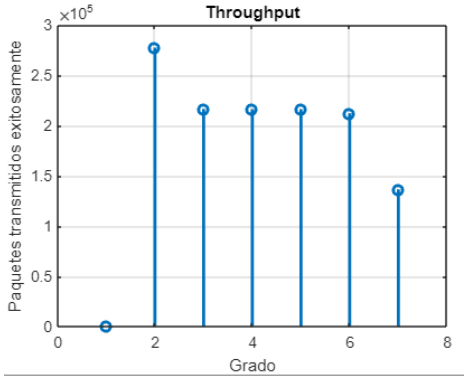

Paquetes Tx perdidos
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7
54.594%
24.061%
12.324%
5.767%

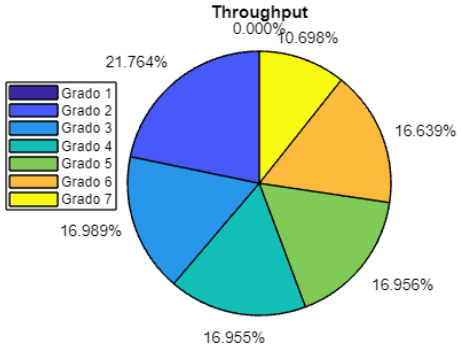

Retardo por grado
tiempo [ms]
Grado
0
1
2
3
4
5
6
7
8
0
2
4
6
8

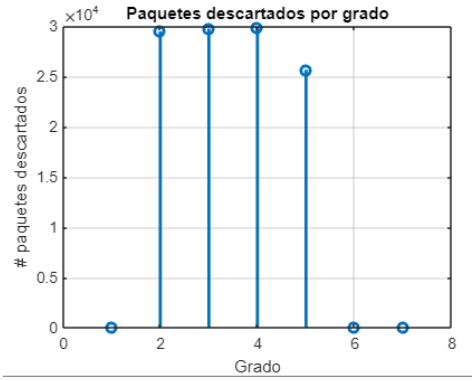

Retardo por grado
12.822%
19.318%
15.130%
18.486%
16.546%
17.697%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

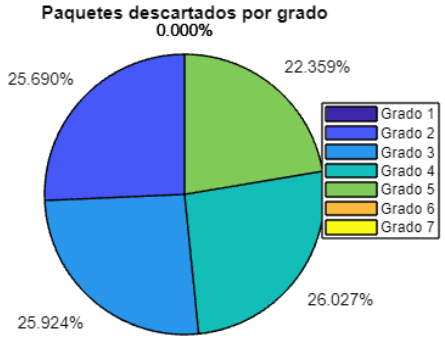

Paquetes Tx
< 1%
Perdidos
Exitosamente
99.070%

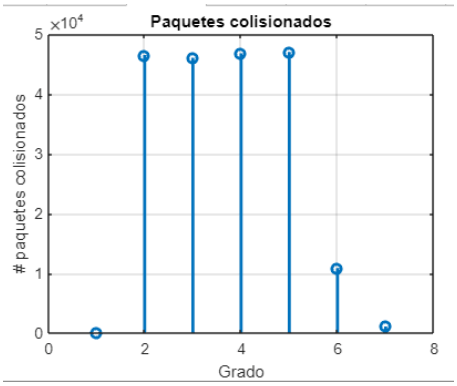
- N=10 λ=0.03

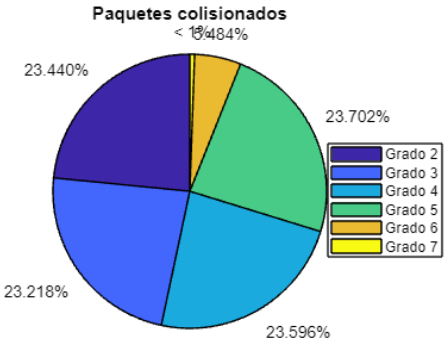

Throughput
×10^5
Paquetes transmitidos exitosamente
Grado

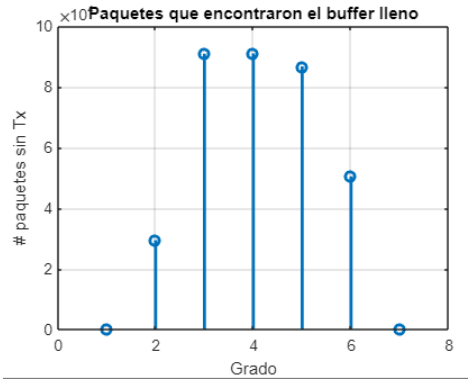

Throughput
0.000%
10.698%
21.764%
16.639%
16.989%
16.956%
16.955%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

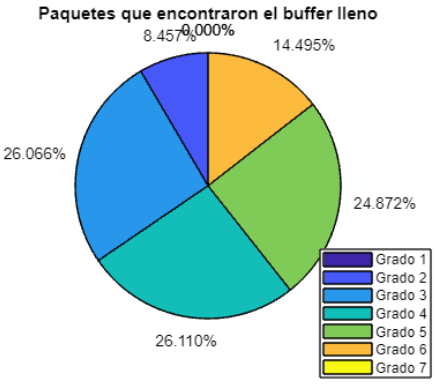

Paquetes descartados por grado
×10^4
paquetes descartados
Grado

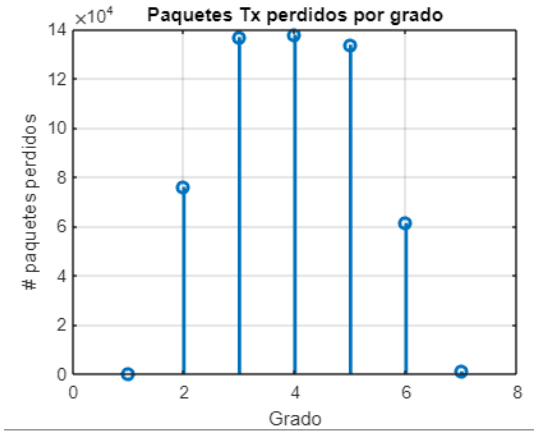

Paquetes descartados por grado
0.000%
25.690%
22.359%
26.027%
25.924%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

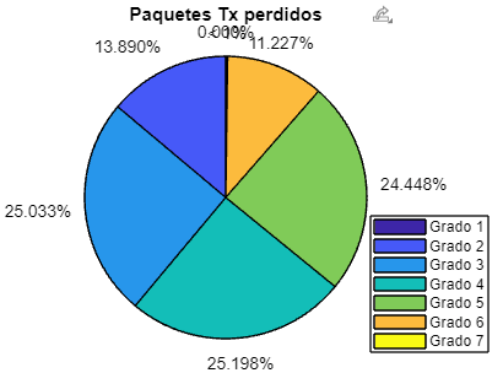

Paquetes colisionados
×10^4
paquetes colisionados
Grado

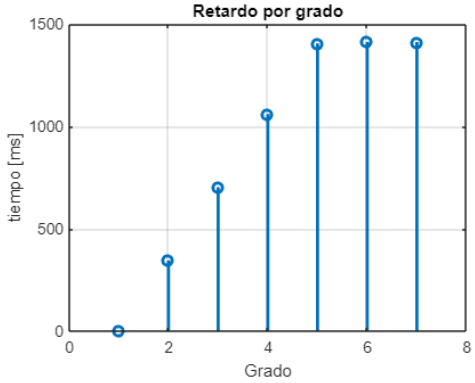

Paquetes colisionados
23.440%
23.702%
23.218%
23.596%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

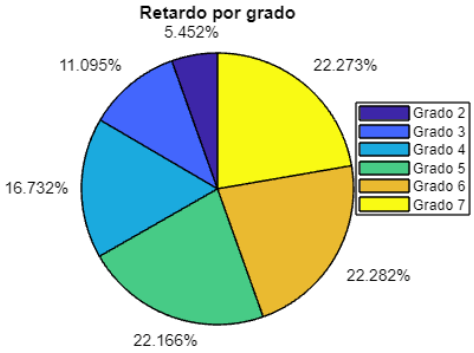

Paquetes que encontraron el buffer lleno
×10
paquetes sin Tx
Grado

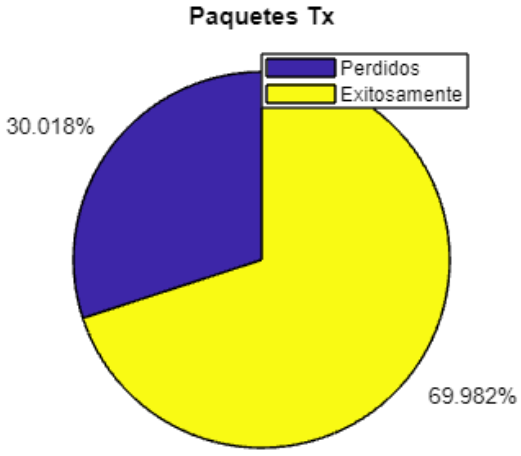

Paquetes que encontraron el buffer lleno
8.457%
0.000%
14.495%
26.066%
24.872%
26.110%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Paquetes Tx perdidos por grado
×10^4
paquetes perdidos
Grado
0
2
4
6
8
10
12
14

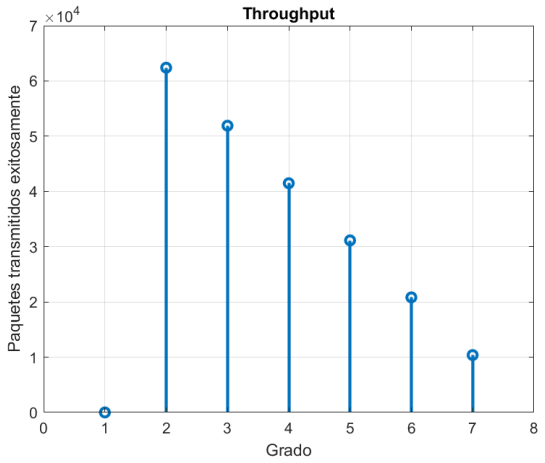

Paquetes Tx perdidos
13.890%
11.227%
24.448%
25.033%
25.198%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

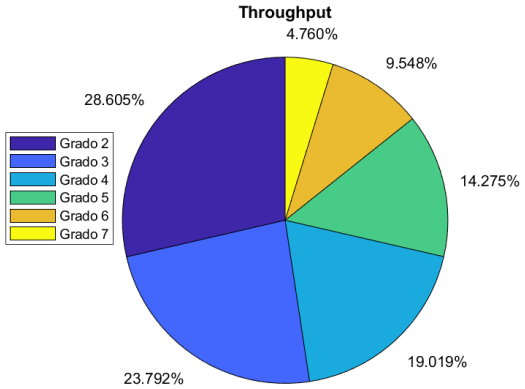

Retardo por grado
tiempo [ms]
Grado
0
500
1000
1500

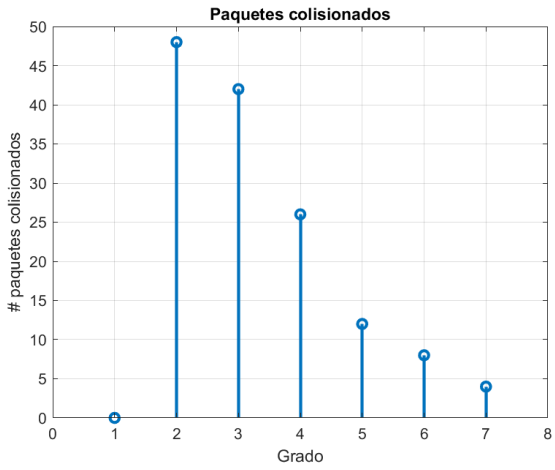

Retardo por grado
5.452%
11.095%
22.273%
16.732%
22.282%
22.166%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

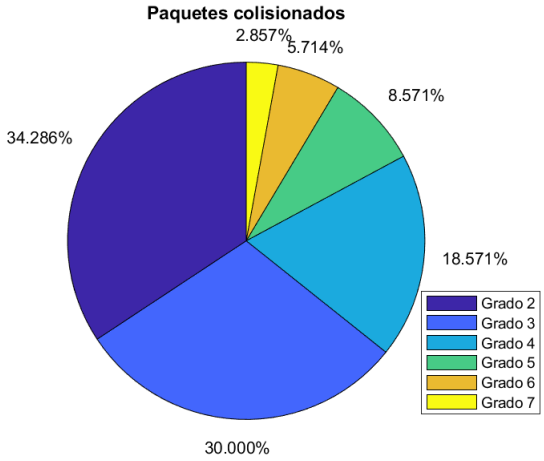

Paquetes Tx
Perdidos
Exitosamente
30.018%
69.982%

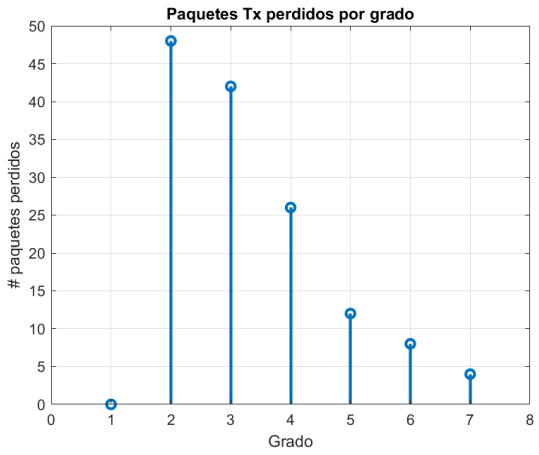
- N=20 λ=0.0005

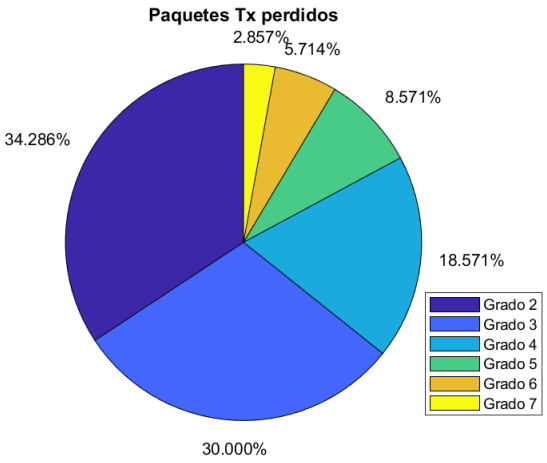

Throughput
×10^4
Paquetes transmitidos exitosamente
Grado


Throughput
4.760%
9.548%
28.605%
14.275%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7
19.019%
23.792%

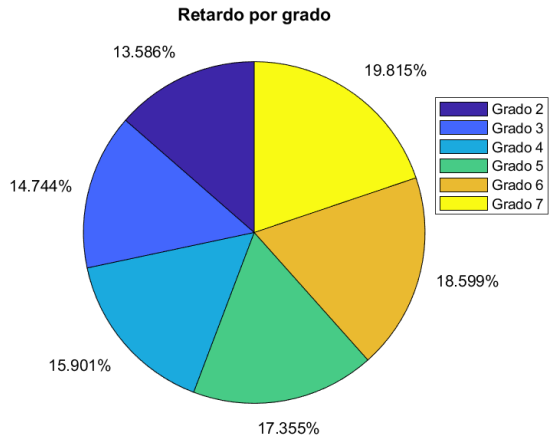

Paquetes colisionados
paquetes colisionados
Grado

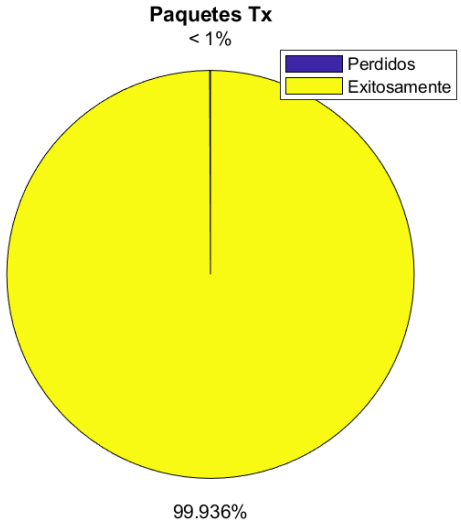

Paquetes colisionados
2.857%
5.714%
8.571%
34.286%
18.571%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7
30.000%


Paquetes Tx perdidos por grado
paquetes perdidos
Grado

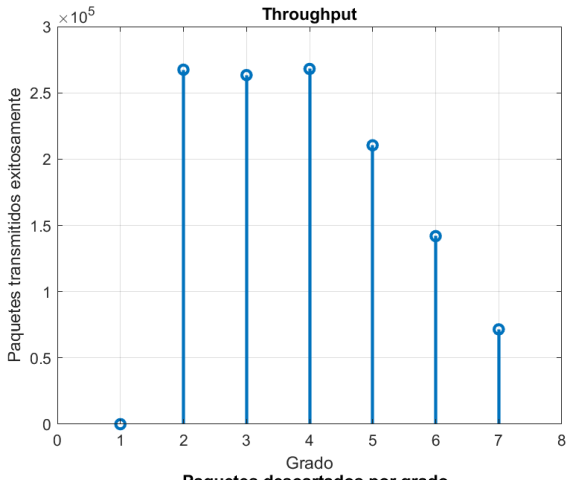

Paquetes Tx perdidos
2.857%
5.714%
8.571%
34.286%
18.571%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7
30.000%

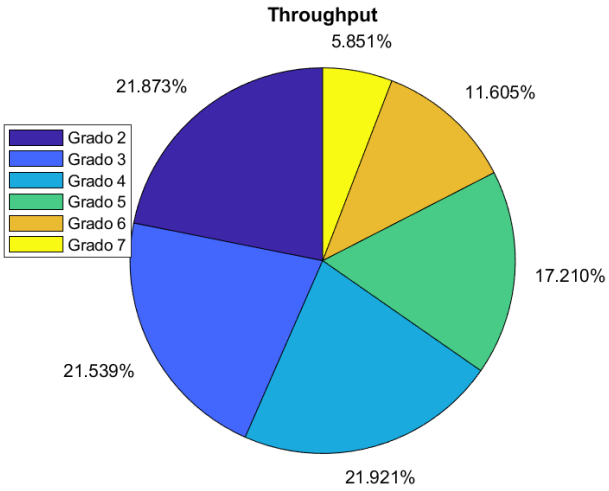

Retardo por grado
tiempo [ms]
Grado
0
0.5
1
1.5
2
2.5
3
0
1
2
3
4
5
6
7
8

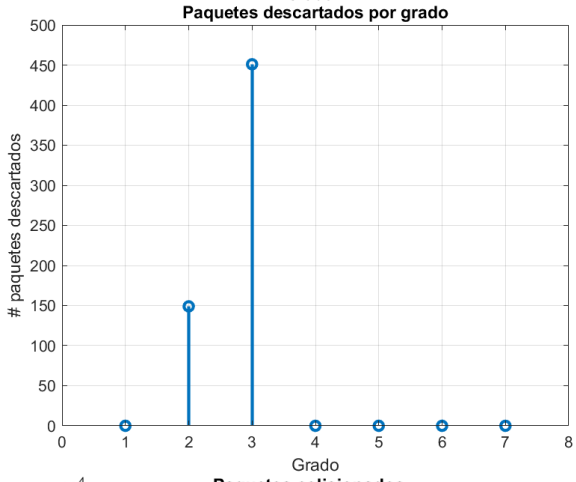

Retardo por grado
13.586%
19.815%
14.744%
18.599%
15.901%
17.355%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Paquetes Tx
< 1%
99.936%
Perdidos
Exitosamente

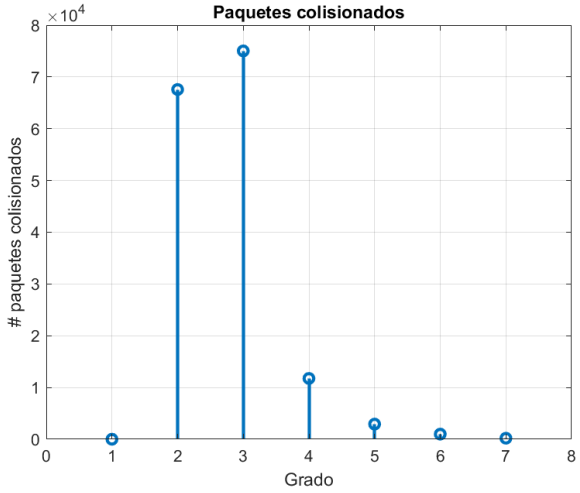
- N=20 λ=0.005

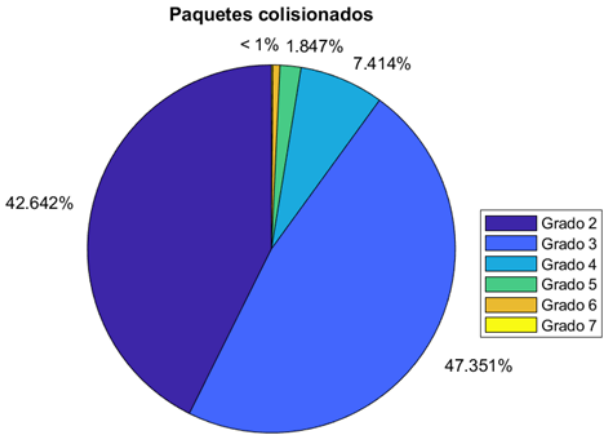

Throughput
×10^5
Paquetes transmitidos exitosamente
Grado

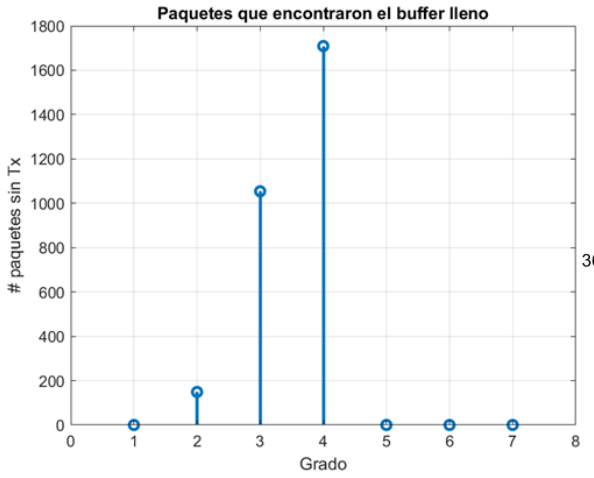

Throughput
5.851%
11.605%
17.210%
21.921%
21.539%
21.873%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

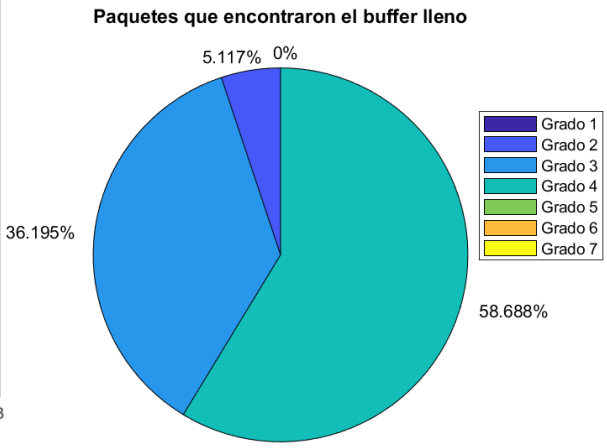

Paquetes descartados por grado
paquetes descartados
Grado

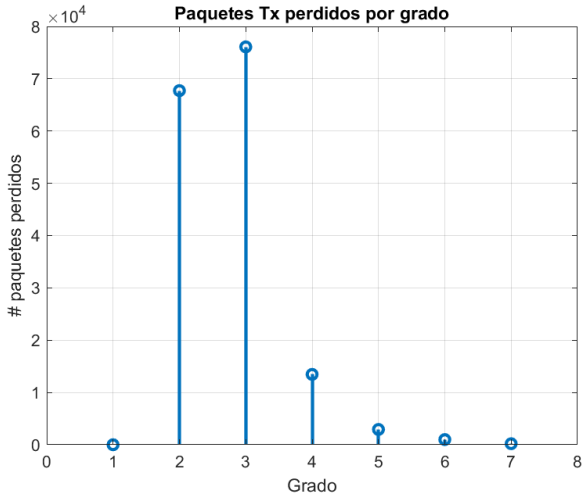

Paquetes descartados por grado
0.000%
24.833%
75.167%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

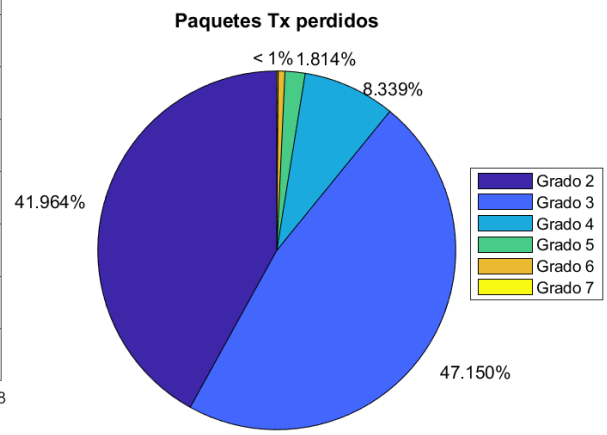

Paquetes colisionados
×10^4
paquetes colisionados
Grado

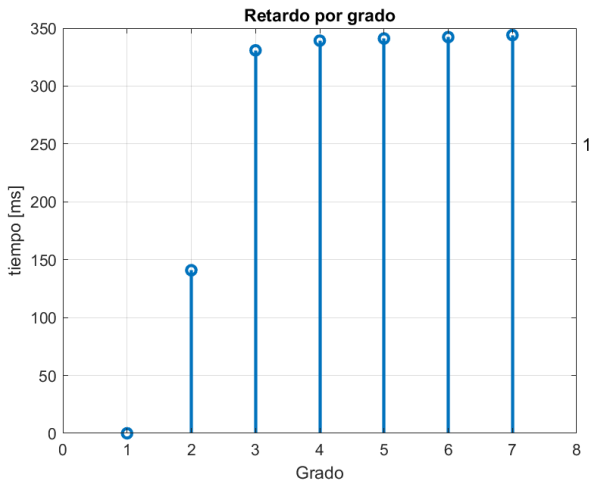

Paquetes colisionados
< 1%
1.847%
7.414%
42.642%
47.351%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

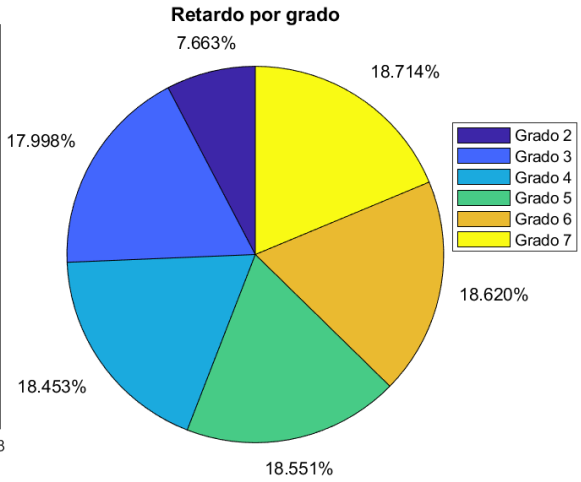

Paquetes que encontraron el buffer lleno
paquetes sin Tx
Grado
0
200
400
600
800
1000
1200
1400
1600
1800
0
1
2
3
4
5
6
7
8

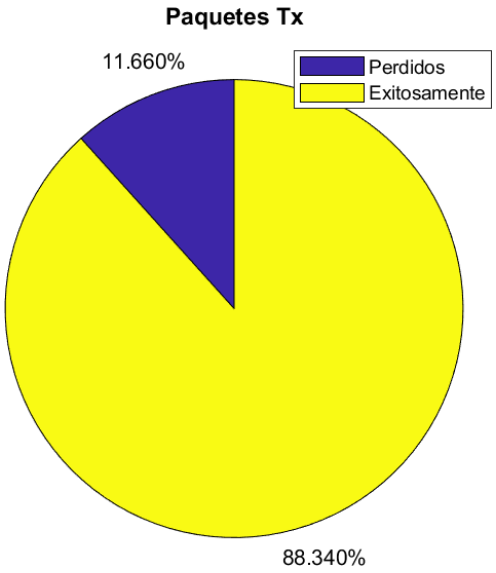

Paquetes que encontraron el buffer lleno
5.117%
0%
36.195%
58.688%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Paquetes Tx perdidos por grado
×10^4
paquetes perdidos
Grado
0
1
2
3
4
5
6
7
8

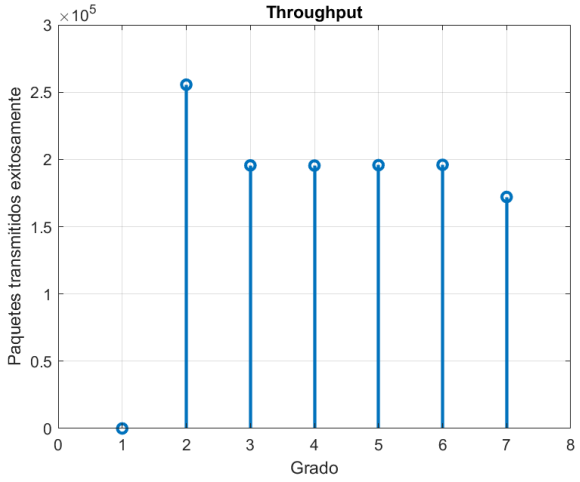

Paquetes Tx perdidos
< 1%
1.814%
8.339%
41.964%
47.150%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

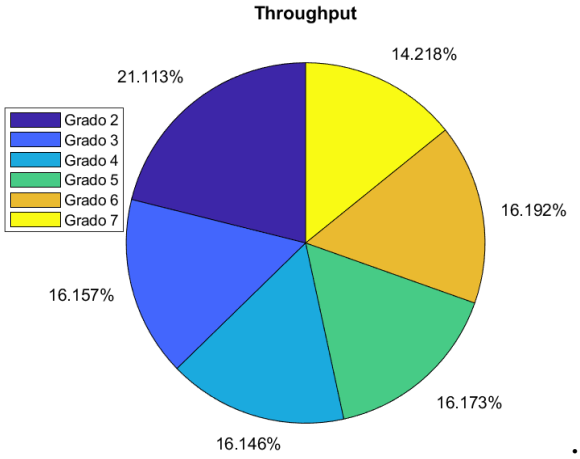

Retardo por grado
tiempo [ms]
Grado
0
50
100
150
200
250
300
350
0
1
2
3
4
5
6
7
8

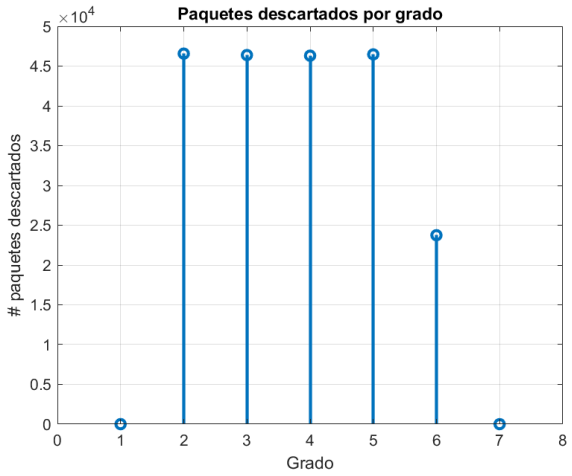

Retardo por grado
7.663%
18.714%
17.998%
18.620%
18.453%
18.551%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

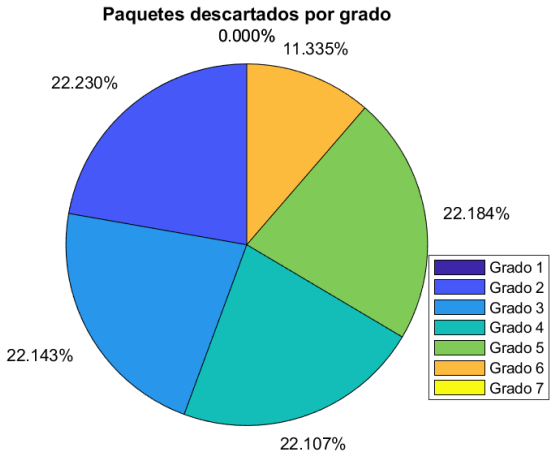

Paquetes Tx
11.660%
Perdidos
Exitosamente
88.340%

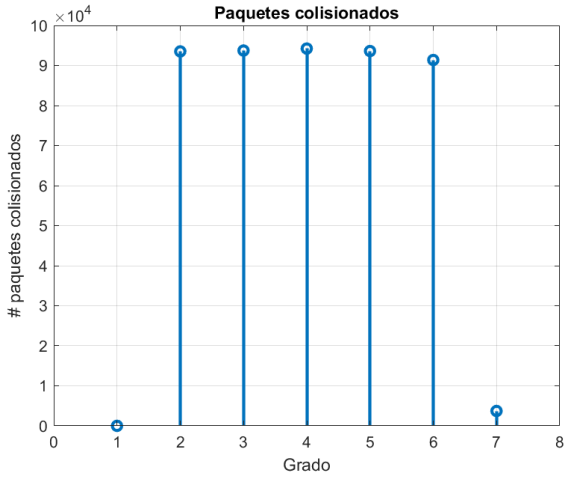
- N=20 λ=0.03

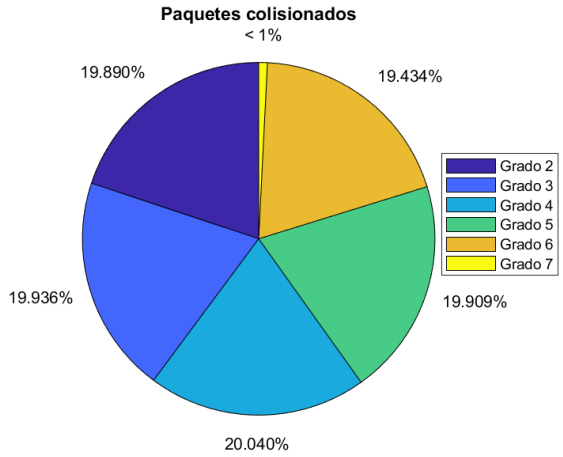

Throughput
×10^5
Paquetes transmitidos exitosamente
Grado

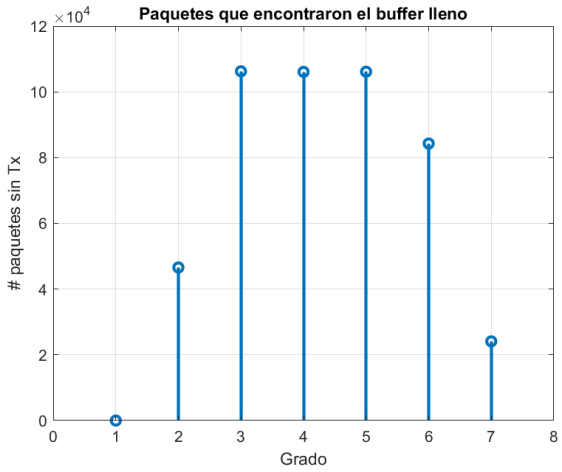

Throughput
21.113%
14.218%
16.192%
16.157%
16.173%
16.146%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

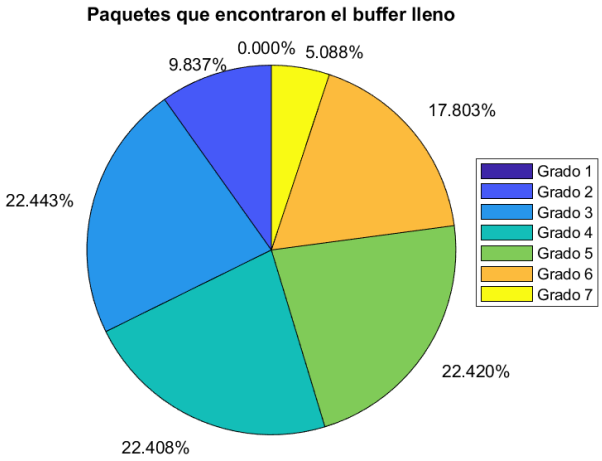

Paquetes descartados por grado
×10^4
paquetes descartados
Grado

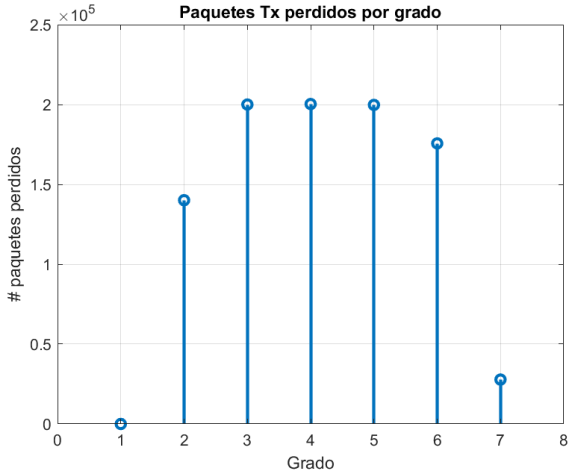

Paquetes descartados por grado
0.000%
11.335%
22.230%
22.184%
22.143%
22.107%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

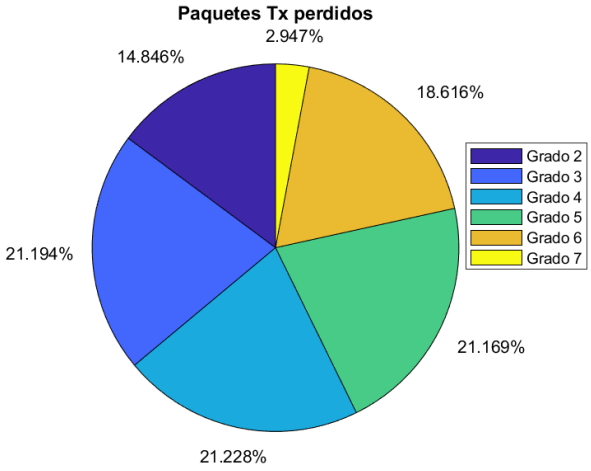

Paquetes colisionados
×10^4
paquetes colisionados
Grado

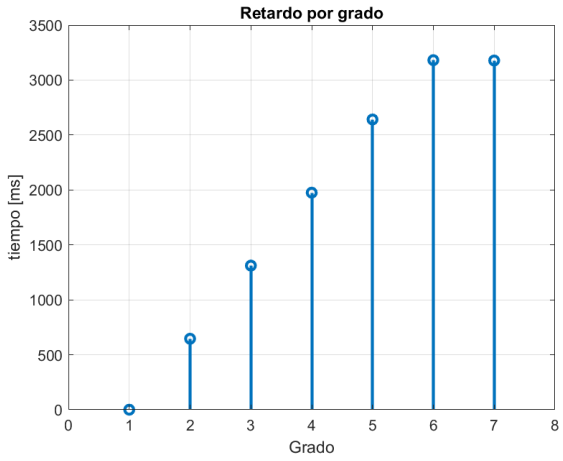

Paquetes colisionados
< 1%
19.890%
19.434%
19.936%
19.909%
20.040%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7

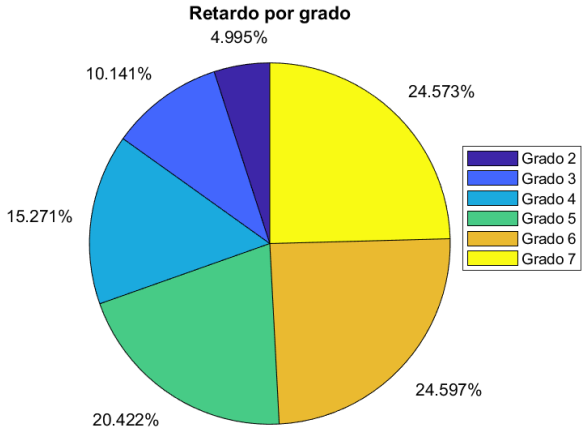

Paquetes que encontraron el buffer lleno
×10^4
paquetes sin Tx
Grado

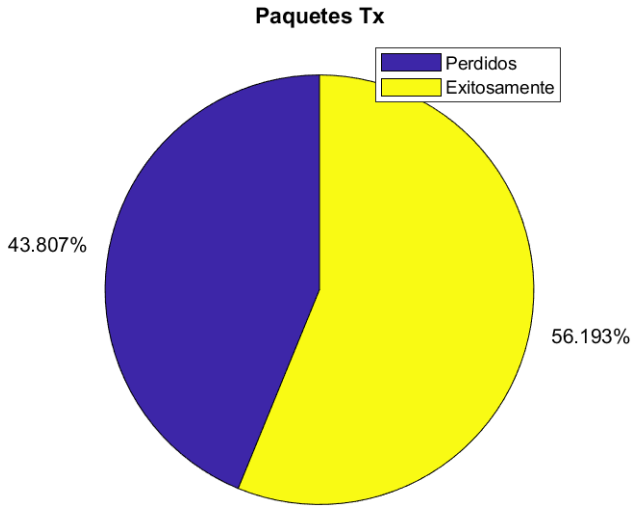

Paquetes que encontraron el buffer lleno
0.000%
5.088%
9.837%
17.803%
22.443%
22.420%
22.408%
Grado 1
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Paquetes Tx perdidos por grado
×10^5
paquetes perdidos
Grado


Paquetes Tx perdidos
2.947%
14.846%
18.616%
21.194%
21.169%
21.228%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Retardo por grado
tiempo [ms]
Grado
0
500
1000
1500
2000
2500
3000
3500
0
1
2
3
4
5
6
7
8


Retardo por grado
4.995%
10.141%
24.573%
15.271%
24.597%
20.422%
Grado 2
Grado 3
Grado 4
Grado 5
Grado 6
Grado 7


Paquetes Tx
Perdidos
Exitosamente
43.807%
56.193%

Conclusión

A lo largo de la simulación se tenían variables para la cantidad de nodos dentro de los grados de la red, así como también el que indicaba el número máximo de mini ranuras y la tasa de generación de paquetes (ʎ). Cada una de las variables anteriores afecta significativamente los resultados que arroje la simulación, ya sea que este aumente o disminuya los paquetes generados y dependiendo de lo anterior, que se presenten mayores casos de colisión o buffer lleno.

Como se logra observar en las diversas gráficas, se mantuvo el valor de las mini ranuras en 64 para las diversas simulaciones logrado así observar el cómo las demás variables de lambda y la cantidad de nodos afectaban el desempeño de la red, el mas evidente de todos fue la taza de generación de paquetes independientemente de los nodos, ya que cada nodo contaba con un buffer y podía lograr almacenar y enviar K cantidad de paquetes en ciclos de trabajo, pero si la tasa de generación era alta, el trafico aumentaba y se iban descartando los paquetes conforme se generaban, claramente la cantidad de nodos ayudaba a dispersar los paquetes pero aun así se generaba una cantidad de paquetes considerable ocasionando que la red se sature.

Los cambios mas evidentes en la simulación ocurren en los nodos que se encuentran más cerca y más lejos del nodo sink, puesto que al ser una jerarquía asimétrica, los nodos que se encuentran más cercanos, transmiten más paquetes y tienen mayor índice de paquetes transmitidos exitosamente que los que se encuentran más alejados que este, pero estos nodos, al tener mayor tráfico, también obtienen como resultado mayor probabilidad de encontrar el buffer lleno, de colisionar, o de perder el paquete al momento de generarlo.

Nuestras evaluaciones nos mostraron que es necesario considerar factores, como lo son la tasa de generación de paquetes, y el numero de nodos por grado para poder obtener el escenario indicado para que nuestra simulación sea la mas optima posible, pero como se mencionó anteriormente si despreciamos estos factores en las simulaciones, el comportamiento es similar en cada una de ellas, con una comportamiento exponencial descendente en la transmisión de paquetes transmitidos exitosamente y con una ascendente en escenarios como el retardo por grado.

La simulación del protocolo MAC en Matlab fue algo nuevo para ambos integrantes del equipo, puesto a que no habíamos desarrollado algo en cuanto a una capa del acceso al medio, en específico como lo fue en esta simulación, esta consistía en el envío y recepción de paquetes por medio del protocolo S-MAC a través de una red LSN, sobre los diversos grados y nodos que esta la componía y consideramos que es necesario abordar otros aspectos como lo son la probabilidad de caída de paquetes, el consumo de energía y la retransmisión de paquetes, que nos da como inicio una nueva problemática para desarrollar posteriormente.