

گزارش پروژه

هلیا قربانی - ۹۸۲۴۳۵۳

کلاس ها و فایل های استفاده شده

• MySender

به عنوان سرور (فرستنده) عمل می کند. در این کلاس ابتدا مقادیر تایمر، احتمال گم شدن ack و احتمال خطای بیتی تنظیم می شود. سپس با توجه به فایل ورودی نوع انتقال که Selective Repeat یا Go Back N است مشخص می شود. علاوه بر آن مقادیر رشته بیتی، ساینز پنجره، timeout و ساینز سگمنت از فایل خوانده می شود و شماره port و تعداد بسته ها به عنوان آرگومان ورودی مشخص می شود.

هر بار یک کاراکتر به صورت رندم به عنوان دیتای ارسالی مشخص می شود. در این کلاس روش های انتقال به صورت نوع ۰ که Go Back N را نشان می دهد و نوع ۱ که Selective Repeat را نشان می دهد در نظر گرفته شده است. همچنین آدرس IP در نظر گرفته شده، localhost است. یک تایمر برای بسته های ارسال شده در نظر گرفته شده که اگر زمان دریافت ack بسته ارسالی از مقدار مشخص شده بیشتر شود، timeout اتفاق می افتد و بسته مجددا ارسال می شود. همچنین در کلاس مشخص می شود در هر گام کدام بسته از فرستنده به گیرنده ارسال می شود. همچنین اگر بسته به درستی و بدون خطا به گیرنده رسیده باشد، فرستنده ack آن را دریافت خواهد کرد. در صورت بروز هر گونه که گیرنده بسته را دریافت نکند و یا بسته دریافتی حاوی دیتای درستی نباشد بسته مجددا ارسال خواهد شد.

با توجه به تعداد ارسال ها و تعداد ارسال های مجدد، در صد گم شدن بسته ها نیز محاسبه می شود.

• MyReceiver

به عنوان کلاینت (گیرنده) عمل می کند. در این کلاس ابتدا احتمال گم شدن بسته ها مشخص می شود. شماره port به عنوان آرگومان ورودی داده می شود. در این کلاس اگر بسته ای به درستی و بدون خطا دریافت شود فرستنده برای دریافت بسته بعدی ack ارسال می کند. در غیر اینصورت به این معناست که بسته گم شده است و یا حاوی دیتای درست نبوده است. دو نوع انتقال مشابه فرستنده در نظر گرفته شده اند. اگر انتقال از نوع Go Back N باشد تمام بسته های دریافتی بعد از بسته ای که به گیرنده نرسیده، نادیده گرفته می شوند و زمانی که گیرنده بسته گم شده را دریافت کند، بسته های بعد از آن مجددا توسط فرستنده ارسال می شوند. اگر انتقال از نوع Selective Repeat باشد، اگر گیرنده بسته ای را دریافت نکند، تمام بسته های دریافتی بعد از آن را در بافر خود ذخیره می کند و بعد از اینکه بسته گم شده مجددا توسط فرستنده ارسال شد و گیرنده آن را به درستی دریافت کرد، بسته های دریافتی پس از آن که در بافر گیرنده ذخیره شده اند در لیست داده های گیرنده به ترتیب قرار می گیرند. علاوه بر این برای افزایش دقت checksum نیز در نظر گرفته شده که بسته ای که به گیرنده رسیده حاوی داده درست باشد. در غیر اینصورت به این معناست که دریافت بسته با خطا مواجه شده که آن بسته نادیده گرفته می شود یا بسته از دست رفته است که در هر دو حالت بسته ها باید مجددا

توسط فرستنده ارسال شوند. با توجه به زمان شروع شبیه سازی و زمان پایان شبیه سازی می توان مدت زمانی که تمامی بسته ها به گیرنده رسیده اند را محاسبه کرد.

- **AckData**

برای ارسال جزئیات مربوط به acknowledgement استفاده می شود.

- **InitiateTransfer**

برای مشخصات دیتای ارسالی استفاده می شود که شامل packet size ، window size و تعداد packet هایی است که به گیرنده ارسال می شود.

- **SegmentData**

برای ارسال دیتا استفاده می شود. در اینجا character به عنوان دیتا ارسال می کنیم. به صورت تصادفی دیتای character تولید می کنیم.

- **InitialConfig**

فایل InitialConfig.txt ، فایل ورودی شامل protocol type ، Packet size و window size . بنابراین برای تست Go Back N ابتدای فایل InitialConfig.txt ، GNB و برای تست Selective Repeat در ابتدای آن SR می گذاریم.

- **کلاس های مربوط به پیاده سازی گرافیکی**

تمامی این کلاس ها صرفا جنبه گرافیکی دارند و برای گرافیک کار طراحی شده اند.

ACKNO ○

Packet ○

SDbt ○

DataEntity ○

DataEntityController ○

FXMLDocument ○

GraphicalSimulation ○

Scroll.css ○

- **تصاویر**

First.jpg ○

Second.jpg ○

Third.jpg ○

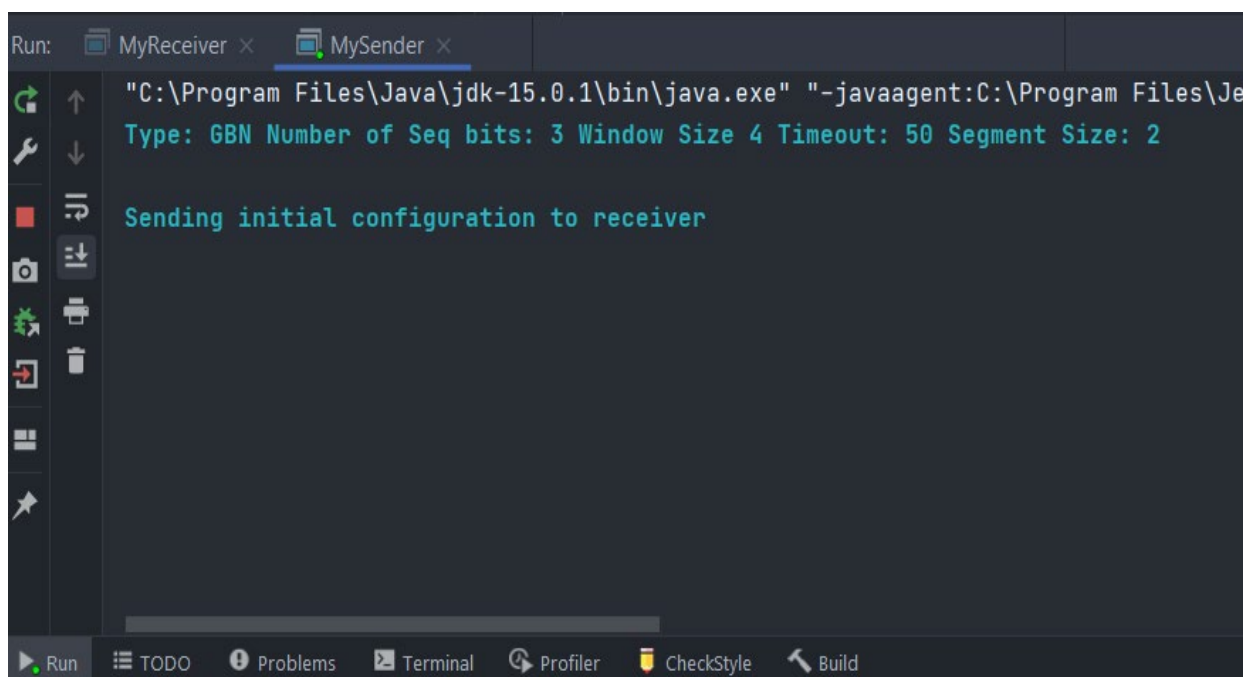
Icon.jpg ○

- کلاس MyReceiver با تعیین کردن آرگومان port number در ide یا در command line با کد زیر اجرا شود.
 - Javac MyReceiver.java
 - Java MyReceiver 8080
 - 8080 – port number
- کلاس MySender با تعیین کردن آرگومان های InitialConfig.txt ، port number و تعداد packet های ارسالی در ide یا در command line با کد زیر اجرا شود.
 - Javac MySender.java
 - Java MySender InitialConfig.txt 8080 20
 - 8080 – port number
 - 20 – number of packets to be sent

Here we are setting the window size as 4, the total number of packets send is 20.

اینجا ما window size را 4 ، مجموع تعداد packet های ارسالی را 20 تنظیم می کنیم.

(۱) در ابتدا فرستنده بسته همگام سازی حاوی اطلاعات اولیه مانند packet size ، window size ، تعداد packet ها، Timeout را به گیرنده می فرستد.



```
Run: MyReceiver x MySender x
"C:\Program Files\Java\jdk-15.0.1\bin\java.exe" "-javaagent:C:\Program Files\Jet...
Type: GBN Number of Seq bits: 3 Window Size 4 Timeout: 50 Segment Size: 2

Sending initial configuration to receiver
```

Figure 1 : Initial configuration

(۲) تصویر زیر برای فرستنده و گیرنده است، در ابتدا ما ۴ بسته با شماره ۰ تا ۳ ارسال می کنیم. اما در سمت گیرنده بسته ۱ به آن نرسیده

است، بنابراین گیرنده پیام "Packet Lost" را نیز چاپ می کند. طبق پروتکل GBN ، و بسته های زیر (بسته های ۲، ۳ و ۴) کنار

گذاشته می شوند. برای این منظور گیرنده پیام "Packet Discarded" را در خط فرمان چاپ می کند.

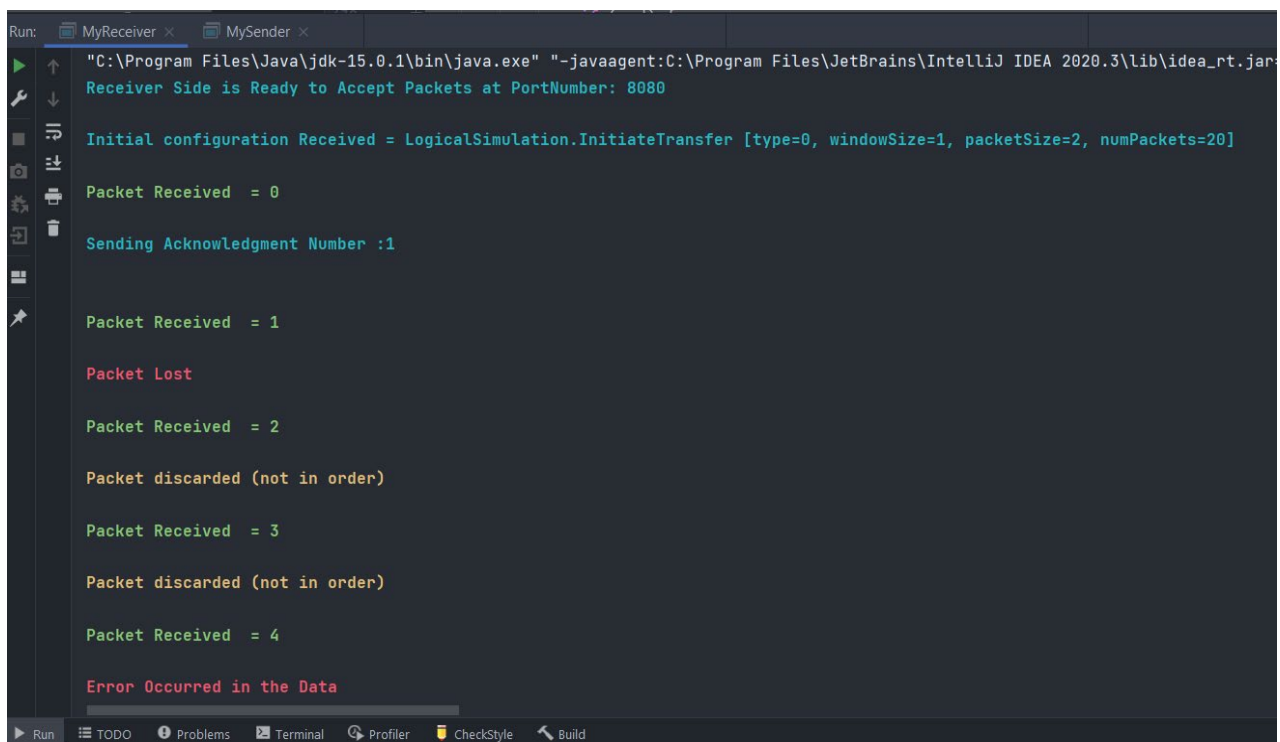
(۳) هنگامی که گیرنده یک بسته را به درستی دریافت کرد، گیرنده ACK را برای آن بسته ارسال می کند، این پیام در پنجره فرستنده به

عنوان پیام "Received ACK" نمایش داده می شود.

(۴) هنگامی که فرستنده بسته های ACK ارسال شده توسط گیرنده را دریافت کرد، پنجره (window) برای قرار دادن مجموعه بسته های

بعدی حرکت می کند. این در تصاویر زیر نشان داده شده است، فرستنده ACK را برای بسته ۱ دریافت می کند، پس از آن پنجره جابجا

می شود و بسته های ۱، ۲، ۳ و ۴ دوباره ارسال می شوند. به دلیل اینکه فرستنده ACK بسته ۱ را ارسال نکرده و timeout اتفاق افتاده است.



```
Run: MyReceiver x MySender x
"C:\Program Files\Java\jdk-15.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.3\lib\idea_rt.jar"
Receiver Side is Ready to Accept Packets at PortNumber: 8080

Initial configuration Received = LogicalSimulation.InitiateTransfer [type=0, windowSize=1, packetSize=2, numPackets=20]

Packet Received = 0

Sending Acknowledgment Number :1

Packet Received = 1

Packet Lost

Packet Received = 2

Packet discarded (not in order)

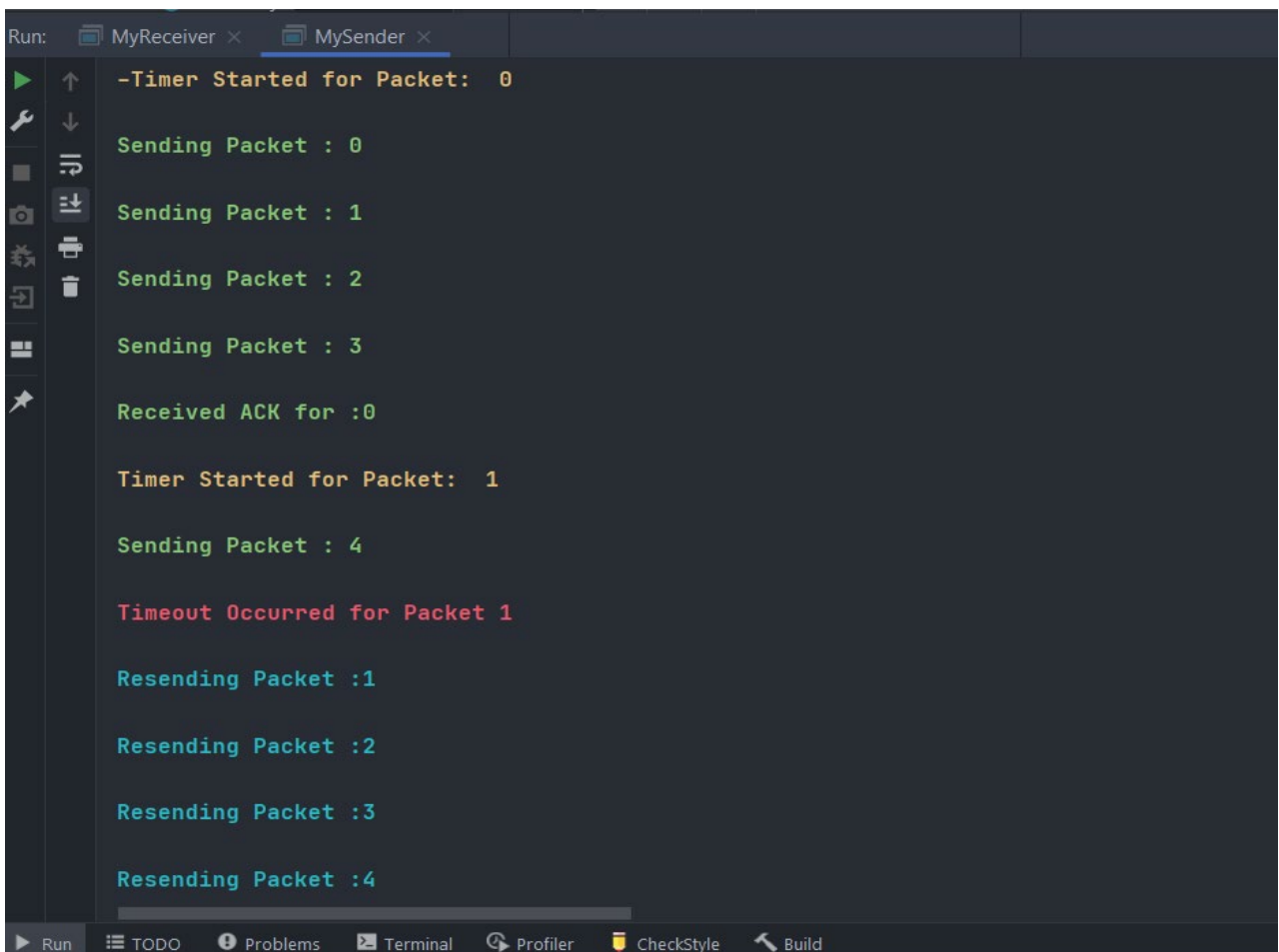
Packet Received = 3

Packet discarded (not in order)

Packet Received = 4

Error Occurred in the Data
```

Figure 2: Packet lost receiver



```
Run: MyReceiver x MySender x
-Timer Started for Packet: 0

Sending Packet : 0

Sending Packet : 1

Sending Packet : 2

Sending Packet : 3

Received ACK for :0

Timer Started for Packet: 1

Sending Packet : 4

Timeout Occurred for Packet 1

Resending Packet :1

Resending Packet :2

Resending Packet :3

Resending Packet :4
```

Figure 3: Packet lost sender

- (۱) در ابتدا فرستنده بسته همگام سازی حاوی اطلاعات اولیه مانند اندازه بسته، اندازه پنجره، تعداد بسته ها را به گیرنده ارسال می کند.
- (۲) اندازه پنجره را ۴ و تعداد بسته ها را ۲۰ تنظیم می کنیم. فرستنده بسته ۰ - ۳ را در پنجره اول ارسال می کند و گیرنده تأییدیه را پس می فرستد.
- (۳) در شکل (۴)، بسته ۰ در انتقال گم می شود، بنابراین بسته هایی که بعد از آن می آیند، یعنی بسته ۱ تا ۳ در بافر برنامه ذخیره می شود، و تایمر در فرستنده بسته ۰ شروع می شود، زیرا ACK آن دریافت نشده است. در بازه زمانی (۳۰۰۰ میلی ثانیه) دریافت نمی شود، timeout رخ می دهد، بنابراین فرستنده دوباره بسته را ارسال می کند. هنگامی که گیرنده بسته ۰ را دریافت کرد، ACK را ارسال می کند و بسته ۱-۳ ذخیره شده در بافر به برنامه تحویل داده می شود، همانطور که در شکل (۴) نشان داده شده است.

```

Run: MyReceiver x MySender x
"C:\Program Files\Java\jdk-15.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.3\lib\idea_rt.jar"
Receiver Side is Ready to Accept Packets at PortNumber: 8080

Initial configuration Received = LogicalSimulation.InitiateTransfer [type=1, windowSize=4, packetSize=2, numPackets=20]

Packet 0 Lost in the Transmission

Packet 1 received
Sending Acknowledgment for Packet :1

Packet 1 Stored in Buffer

Packet 2 received
Sending Acknowledgment for Packet :2

Packet 2 Stored in Buffer

Packet 3 received
Sending Acknowledgment for Packet :3

Packet 3 Stored in Buffer

Packet 0 received
Sending Acknowledgment for Packet :0

Packet 1 delivered to Application From Buffer

Packet 2 delivered to Application From Buffer

Packet 3 delivered to Application From Buffer
  
```

Figure 4: Packet lost receiver

```
Run: MyReceiver x MySender x
"C:\Program Files\Java\jdk-15.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.3\lib\
Type: SR Number of Seq bits: 3 Window Size 4 Timeout: 3000 Segment Size: 2

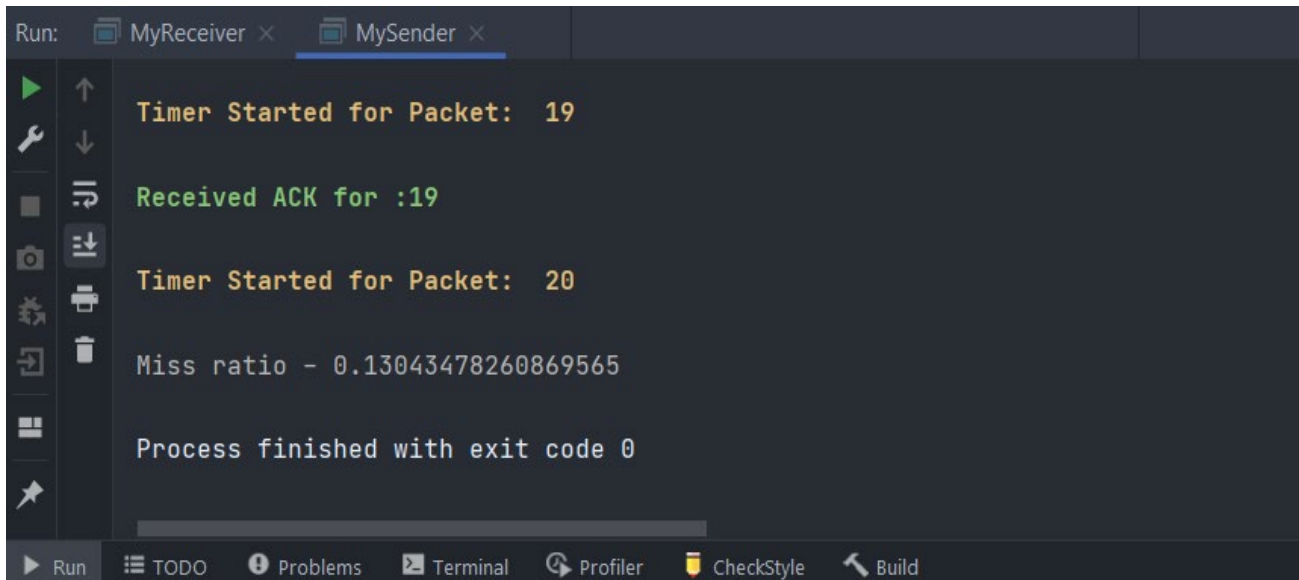
Sending Initial Data
Timer Started for Packet: 0
Sending Packet : 0
Timer Already Running
Sending Packet : 1
Timer Already Running
Sending Packet : 2
Timer Already Running
Sending Packet : 3
Received ACK for :1
Timer already Running for 0
Received ACK for :2
Timer already Running for 0
Received ACK for :3
Timer already Running for 0
Timeout Occurred
Resending Packet :0
Received ACK for :0
```

Figure 5: Packet lost sender

مقایسه عملکرد

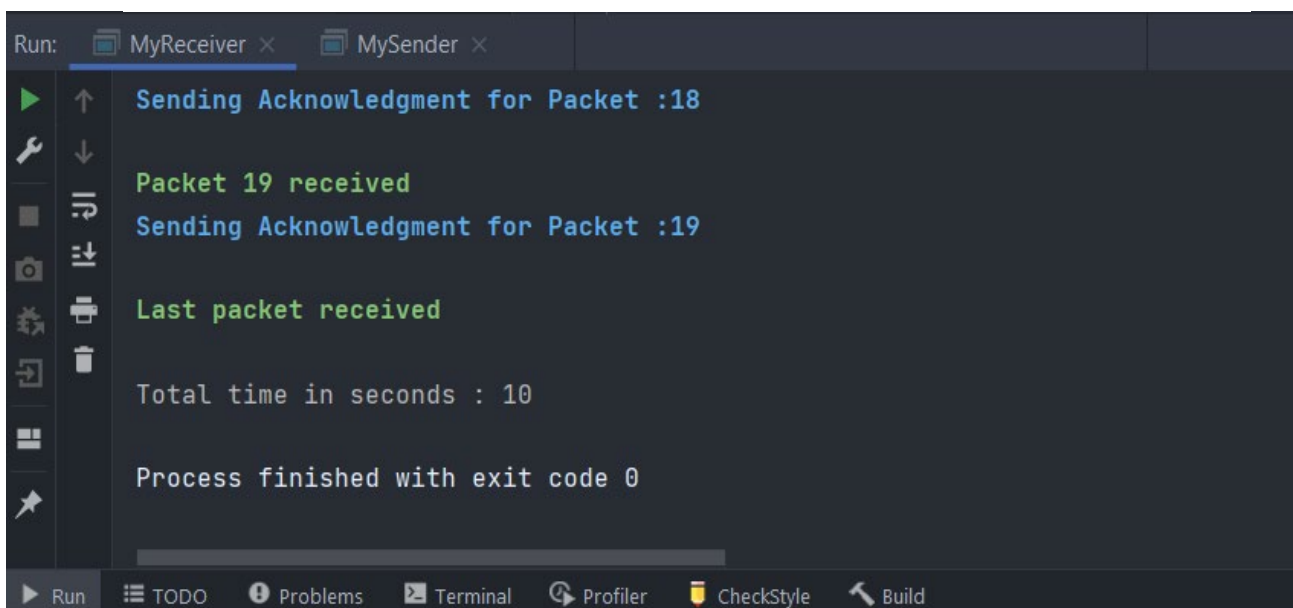
برای مقایسه عملکرد Selective Repeat و Go Back N، اندازه‌گیری‌های زیر را انجام شده است.

- زمان صرف شده برای تکمیل انتقال
- نسبت از دست دادن بسته



```
Run: MyReceiver x MySender x
Timer Started for Packet: 19
Received ACK for :19
Timer Started for Packet: 20
Miss ratio - 0.13043478260869565
Process finished with exit code 0
```

Figure 6: Package miss ratio



```
Run: MyReceiver x MySender x
Sending Acknowledgment for Packet :18
Packet 19 received
Sending Acknowledgment for Packet :19
Last packet received
Total time in seconds : 10
Process finished with exit code 0
```

Figure 7: Total time taken

زمان صرف شده برای تکمیل انتقال

در اینجا زمان صرف شده برای تکمیل انتقال را با تعداد بسته‌های مختلف اندازه‌گیری شده است.

Go Back N

Number of Packets	Time Taken
150 – window size = 10	16
200 – window size = 15	24
400 - window size = 20	47
500 - window size = 25	59
1000 - window size = 30	114

Selective Repeat

Number of Packets	Time Taken
150 – window size = 10	10
200 – window size = 15	15
400 - window size = 20	29
500 - window size = 25	36
1000 - window size = 30	73

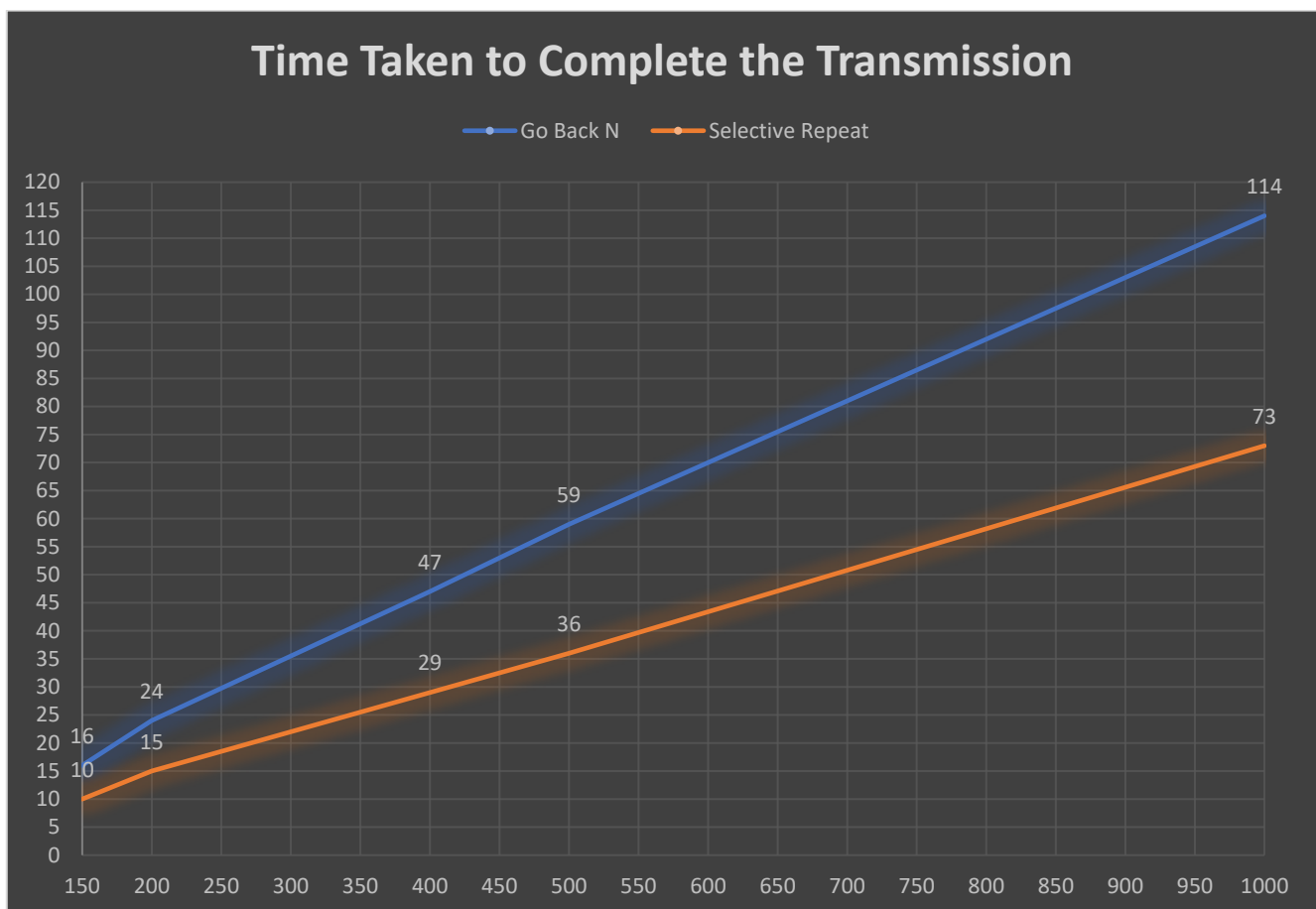


Figure 8: Time taken to deliver packets

Package Miss Ratio

$$\text{Packet miss ratio} = \frac{\text{No. of packets Retransmitted}}{\text{Total no. of packets}}$$

Go Back N

Number of Packets	Package Miss Ratio
150 – window size = 10	0.52
200 – window size = 15	0.5
400 - window size = 20	0.49
500 - window size = 25	0.48
1000 - window size = 30	0.48

Selective Repeat

Number of Packets	Package Miss Ratio
150 – window size = 10	0.17
200 – window size = 15	0.18
400 - window size = 20	0.21
500 - window size = 25	0.15
1000 - window size = 30	0.14

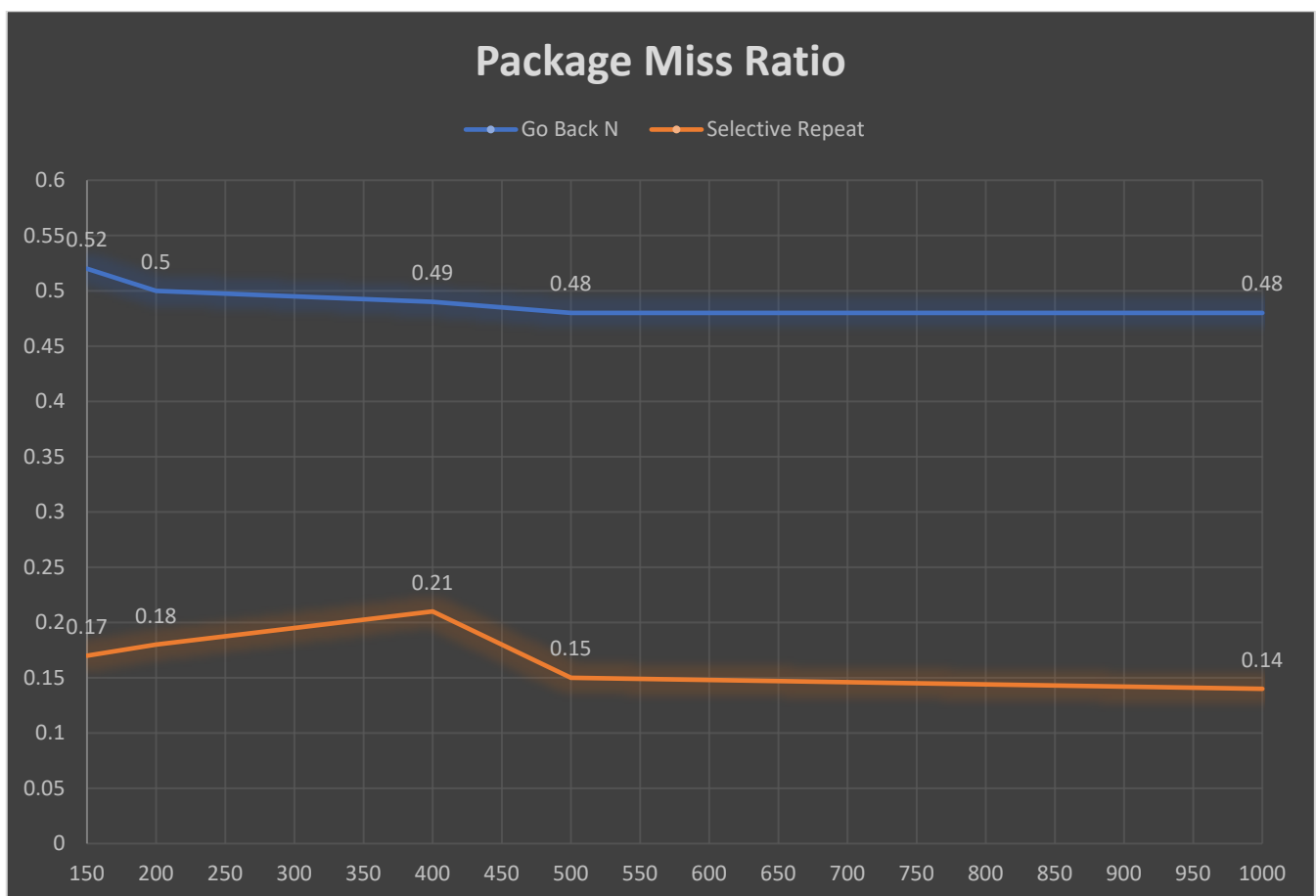


Figure 9: Package miss ratio

- Time Taken to Complete the Transmission

- Selective Repeat

- Go Back N

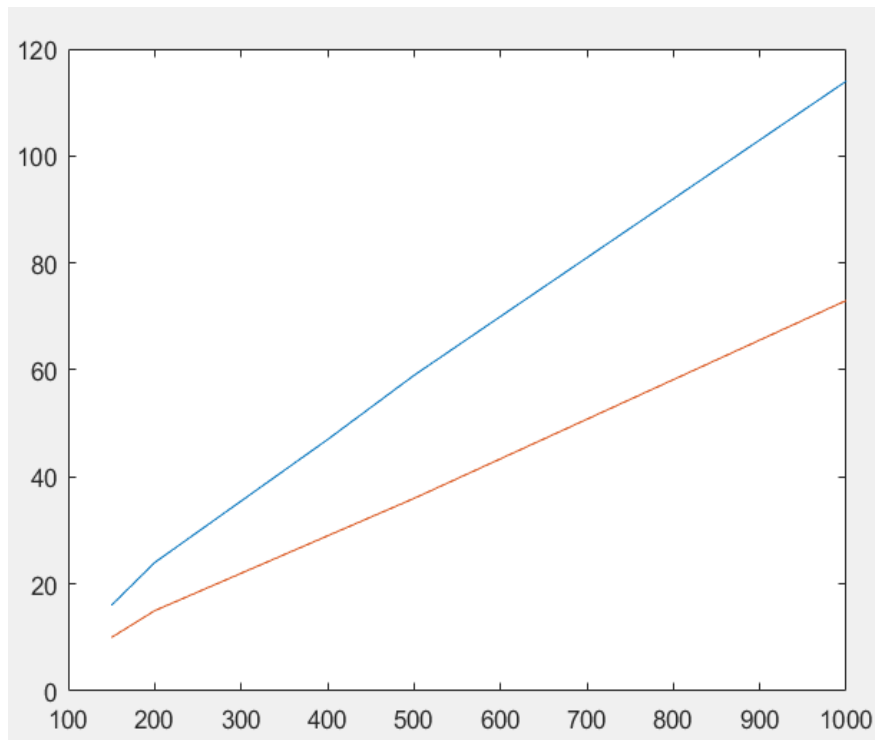


Figure 10: Time taken to deliver packets

- Package Miss Ratio

- Selective Repeat

- Go Back N

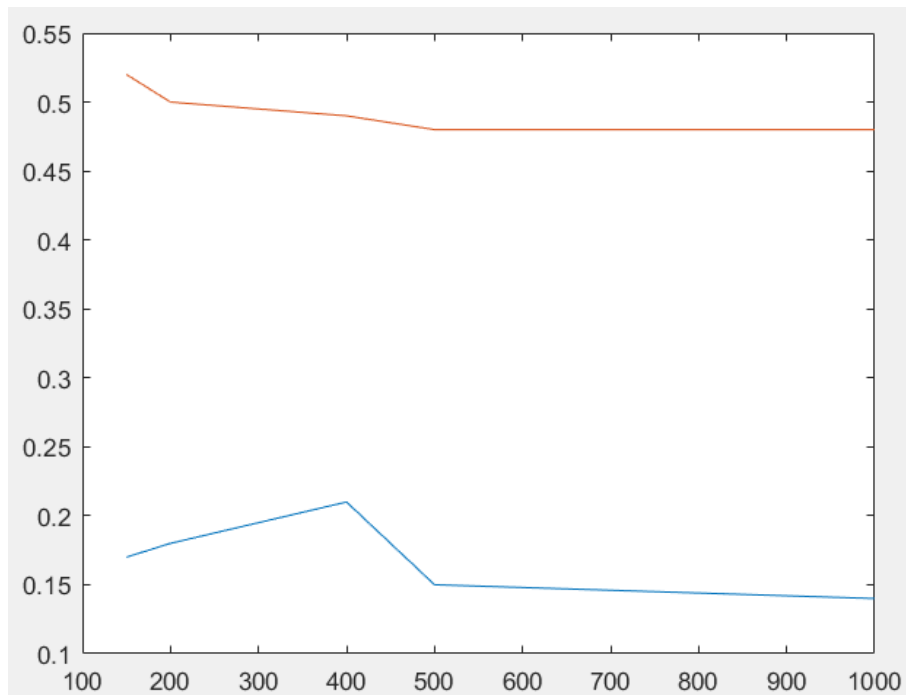


Figure 11: Package miss ratio

(۱) ابتدا تعداد بیت های انتقالی و سایز پنجره را وارد می کنیم. برای سادگی تعداد بیت ها به طور پیش فرض ۱۵ تنظیم شده است.

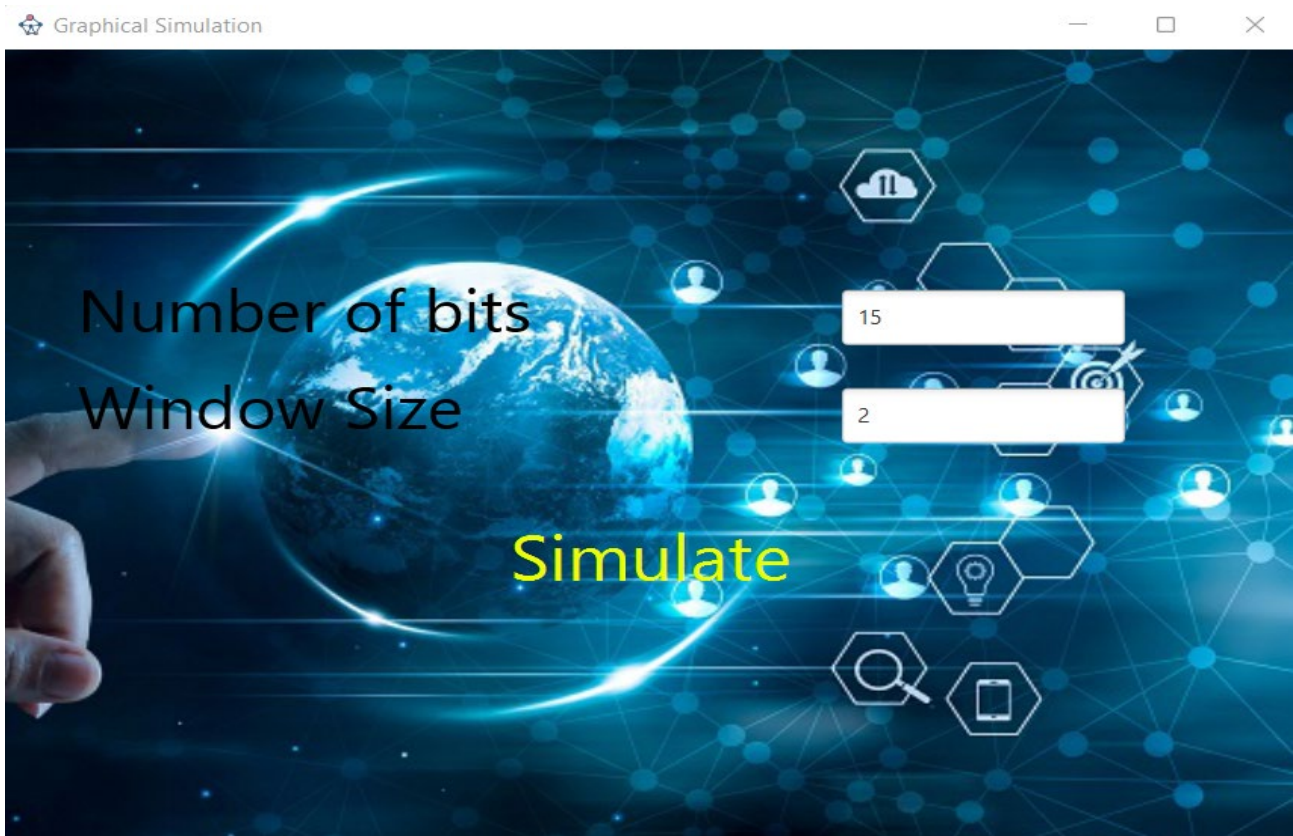


Figure 12

(۲) به تعداد بیت ها و به اندازه پنجره یک مستطیل قرمز رنگ دور داده ها داریم.

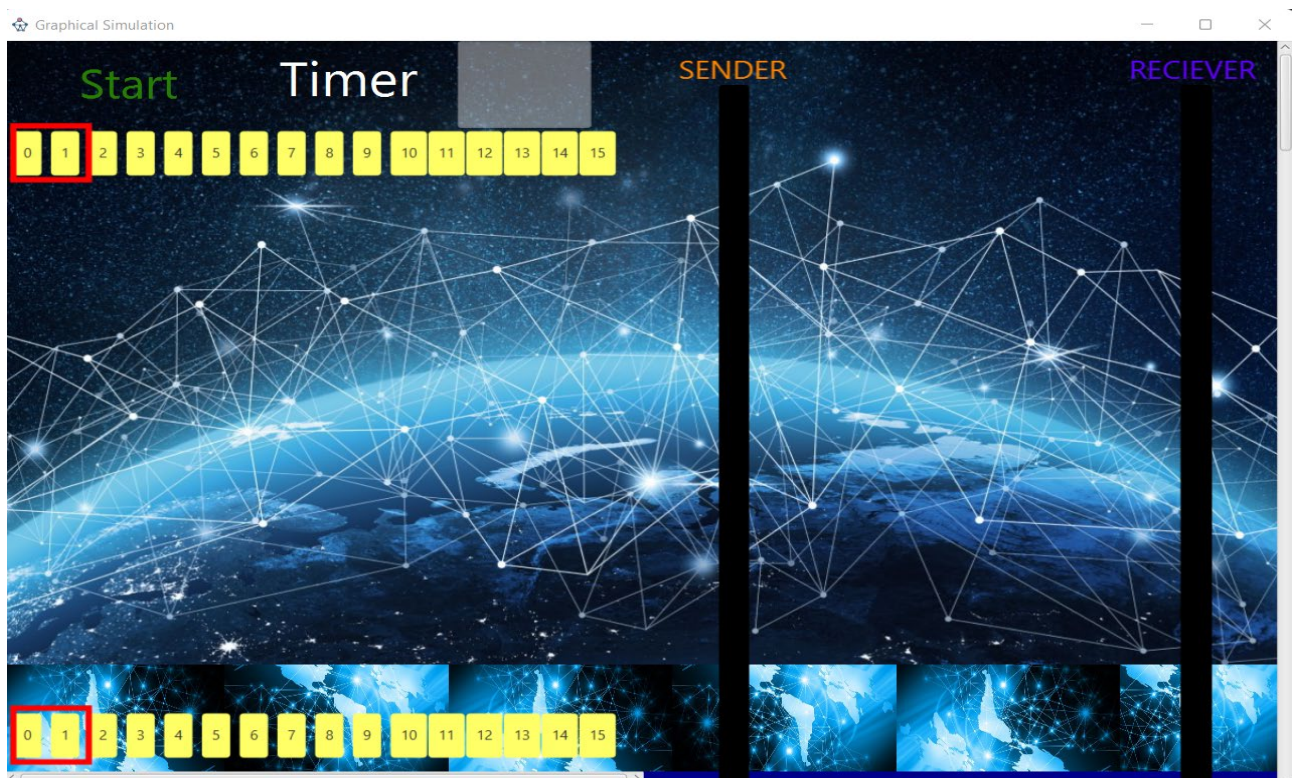


Figure 13

۳) تایمر در داخل برنامه قابل تنظیم است که به طور پیش فرض در نظر گرفته شده که بعد از گذشته آن داده های یک پنجره ارسال می شوند.

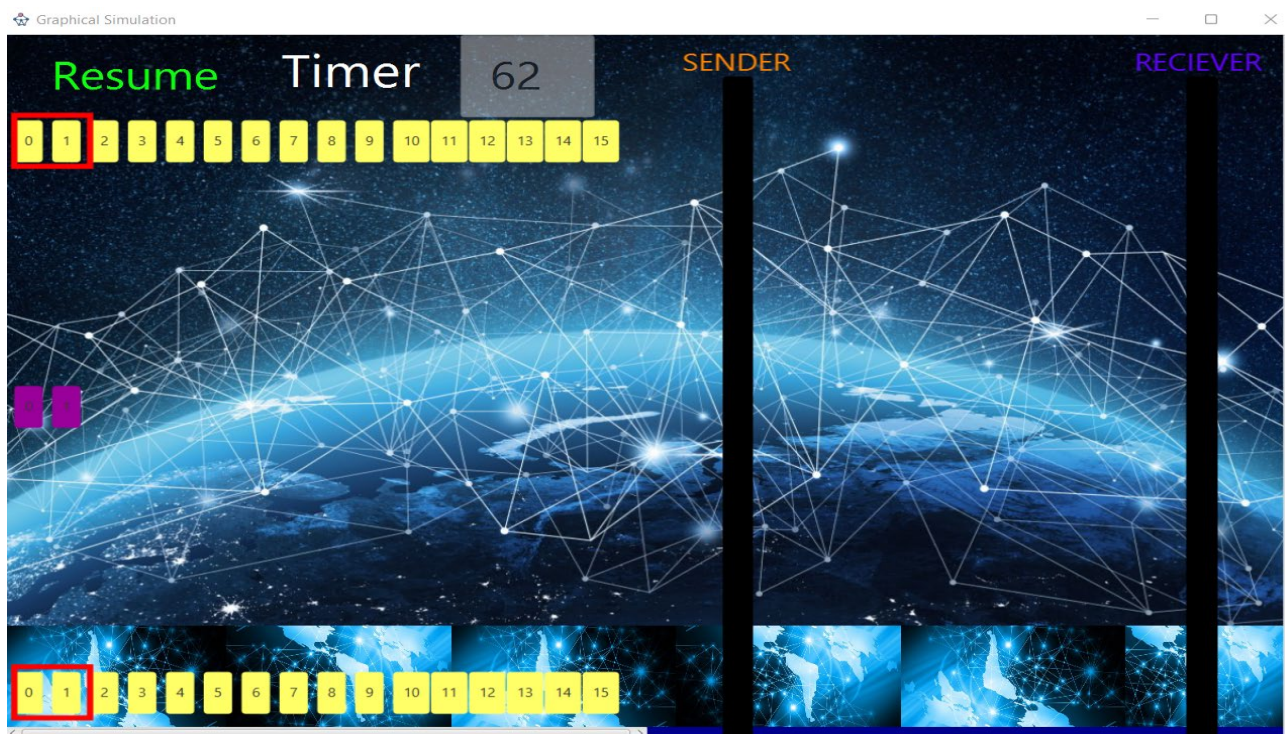


Figure 14

۴) شکل زیر نشان می دهد داده های ۰ و ۱ از فرستنده به گیرنده ارسال شده اند.

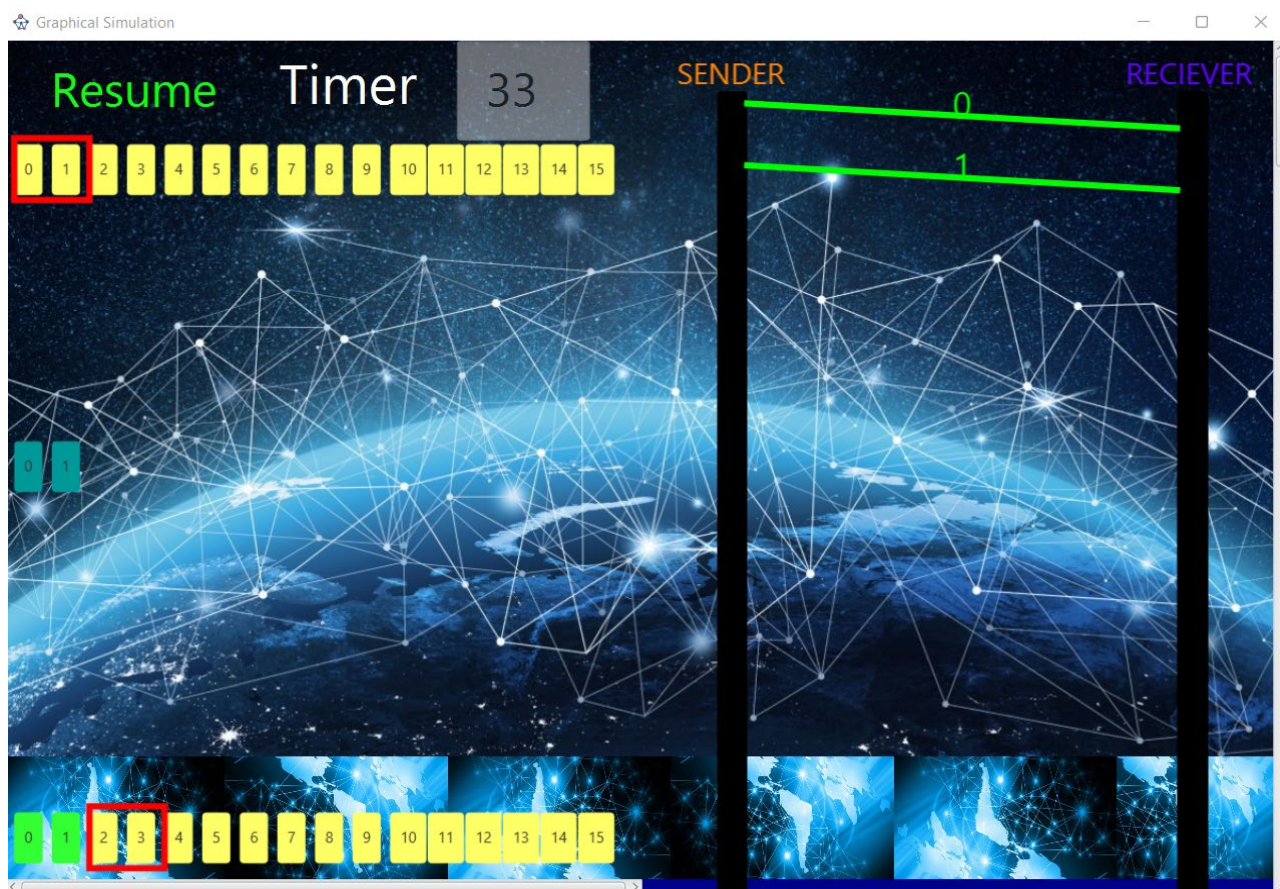


Figure 15

۵) به کمک قابلیت Stop و Resume برنامه می توان بسته ها و ack های ارسالی را مدیریت کرد و کلیک بر روی آنها باعث می شود که بسته ها یا ack های ارسالی حین انتقال گم شوند. به طور مثال در نمونه زیر ack بسته ۱ توسط فرستنده به گیرنده نرسیده است همچنین بسته ۲ در حین ارسال از فرستنده به گیرنده گم شده است.

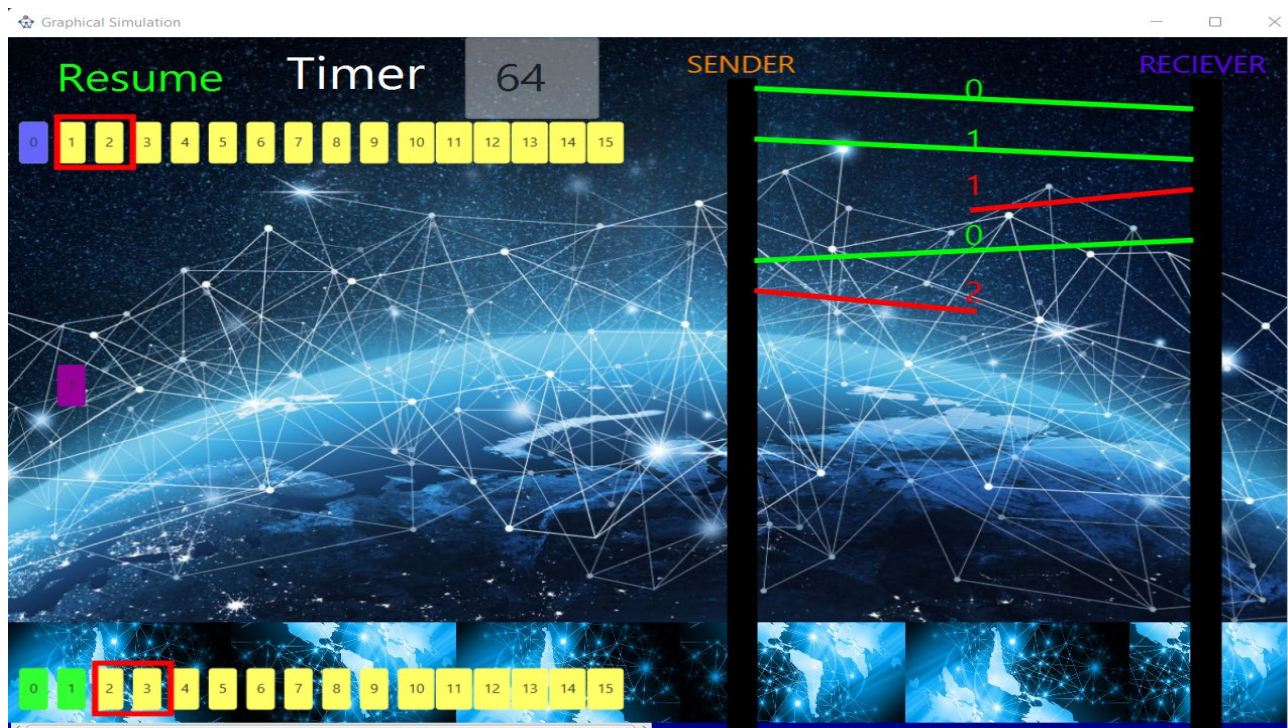


Figure 16

۶) چون بعد از گذشت زمان مشخص شده timeout اتفاق افتاده و ack بسته ۱ به فرستنده نرسیده دوباره بسته ۱ را ارسال می کند و ack آن را دریافت می کند. همچنین بسته ۲ را که حین ارسال گم شده بود مجدد ارسال می کند.

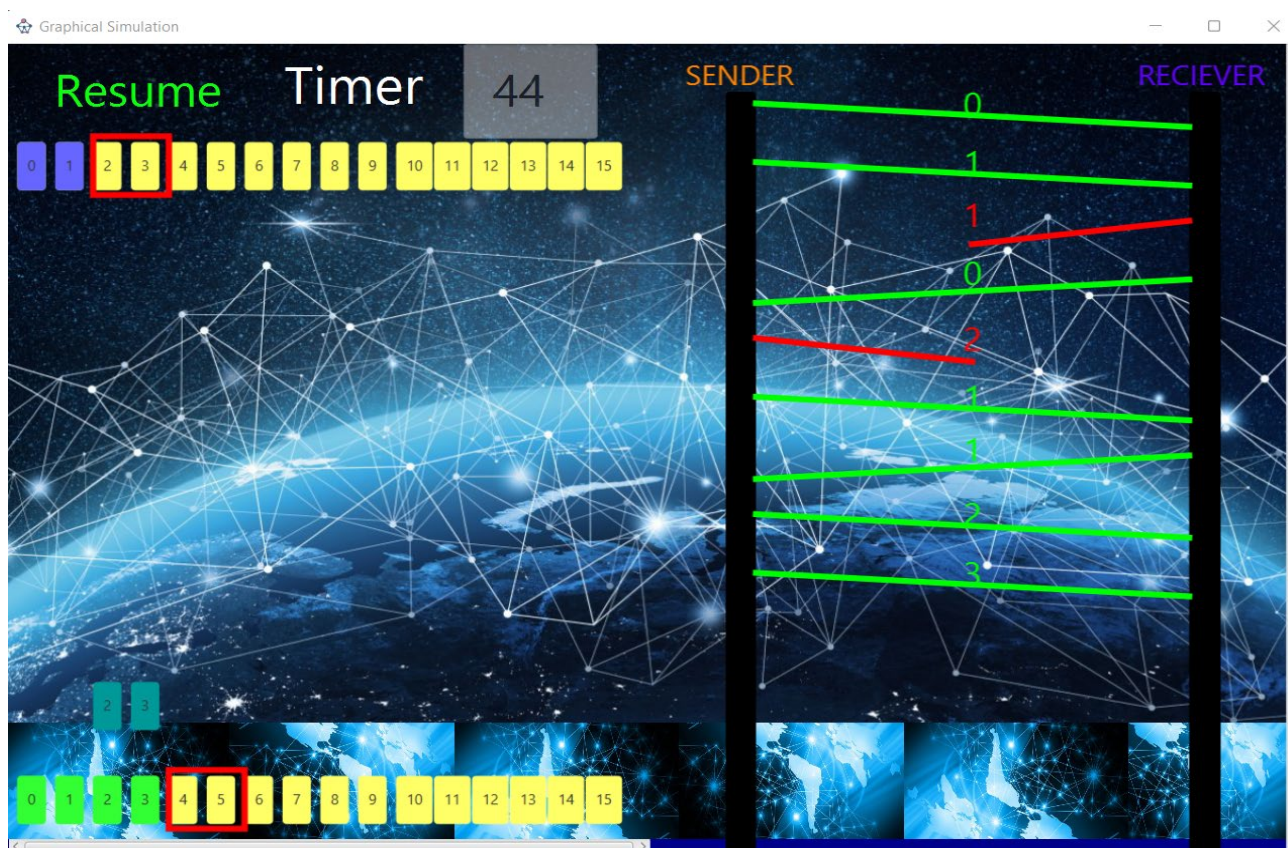


Figure 17

(۷) در نمونه زیر نیز بسته های ۴ و ۵ در حین ارسال گم شده اند و سپس مجدد توسط فرستنده ارسال شده اند.

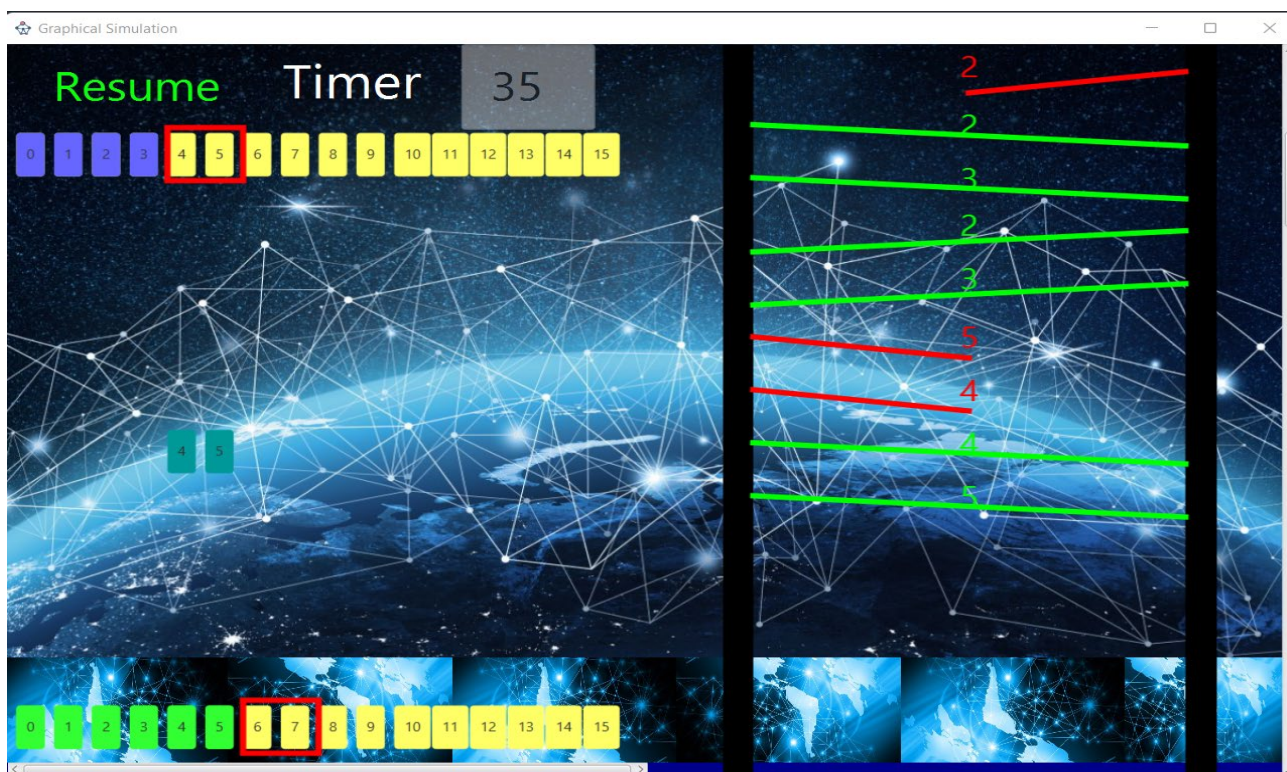


Figure 18

(۸) شکل زیر انتقال بدون مشکل را نشان می دهد.

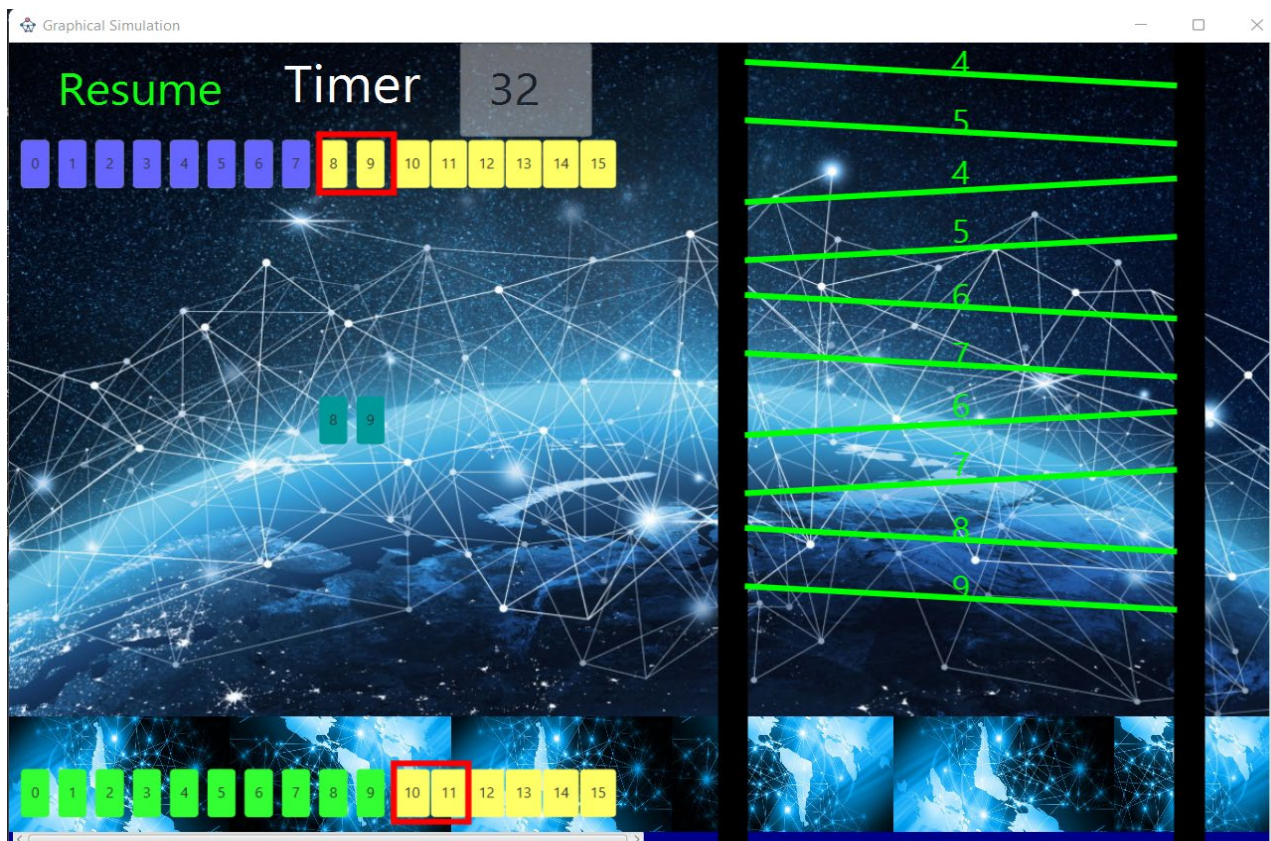


Figure 19

۹) پس از اینکه گیرنده تمامی بسته ها را بدون مشکل دریافت کند، شبیه سازی به پایان می رسد.

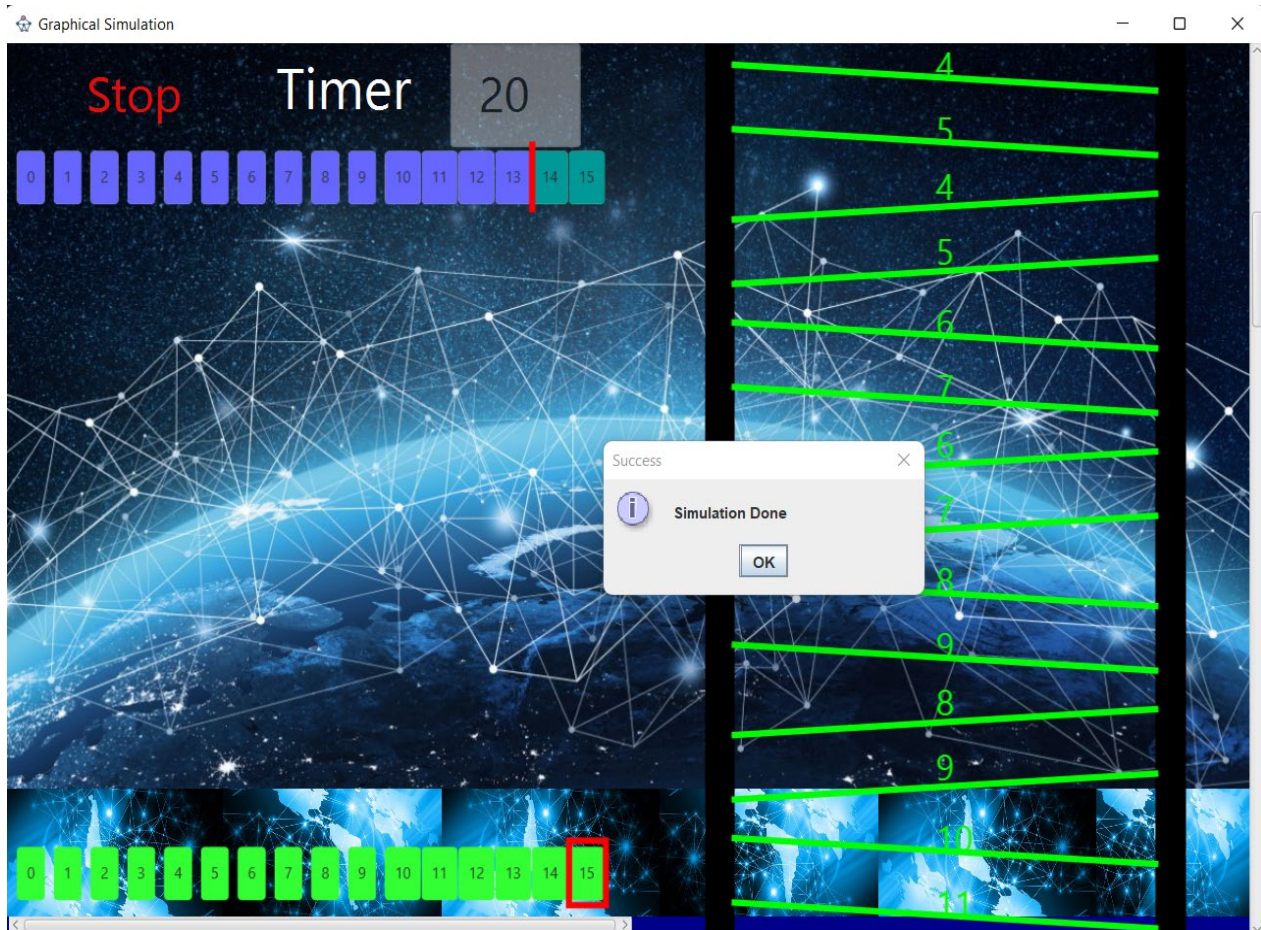


Figure 20

نتیجه

- انتقال Selective Repeat زمان کمتری برای تحویل بسته ها در مقایسه با انتقال Go Back N طول می کشد.
- انتقال Selective Repeat نسبت از دست دادن بسته کمتری را در مقایسه با انتقال Go Back N دارد.