

NETWORK

ASSIGNMENT #FIVE

DEVELOPMENT ENVIRONMENT

OS : Ubuntu 16.04
Language : Python3
Version : Python3.5.2

CONTACT

Name : 남재윤
Student ID : 2014310198
Department : Software

ALGORITHM AND DATA STRUCTURE

1. Sender.py : send, receive, log 를 담당하는 3개의 쓰레드로 동작한다.

1) send thread : send 쓰레드는 이전 과제와 같이 sendbase에서 window size 만큼 패킷을 전송한다.

2) receive thread : receive 쓰레드는 정상적인 Ack을 받으면 window사이즈를 하나 증가시키고 duplicated Ack 혹은 time out이 발생 시에 window size를 1로 줄인다.

3) log thread : 2초마다 sender의 여러정보들을 수집하여 로그 파일을 작성한다.

2. Receiver.py : RM, NEM, NEMtoRM, log를 담당하는 4개의 쓰레드로 동작한다.

1) RM thread : 이 전 과제의 receive and ack을 담당하는 쓰레드와 같이 동작한다. 다만, 소켓통신을 통해 직접 읽어오는 것이 아니라, NEM이 읽어서 queue에 넣어놓은 정보를 읽는다. 포트번호로 그에 맞는 cumulative Ack을 얻어와서 처리한다.

2) NEM thread : 소켓통신을 통해 sender들에게서 패킷을 얻어 queue에 저장한다. 1/BLR 초마다 RM에게 전송한다.

3) log thread : 2초마다 sender의 여러정보들을 수집하여 로그 파일을 작성한다.

3. Receiver에서 여러 Sender들의 요청을 처리해 주기 위한 자료 구조.

port = [] | port 1 | --> | port 2 | --> | port 3 | --> --> | port n |

cum = [] | cum of port 1 | --> | cum of port 2 | --> --> | cum of port 3 |

other = [] | | --> | | --> --> | |

port 번호로 port 리스트에서 index를 찾아서 cumulative Ack을 구해와서 각각 sender에게 맞는 처리를 할 수 있다.

NETWORK

ASSIGNMENT #FIVE

HOW TO RUN

1. Left side shows how to run sender.py and right side shows how to run receiver.py

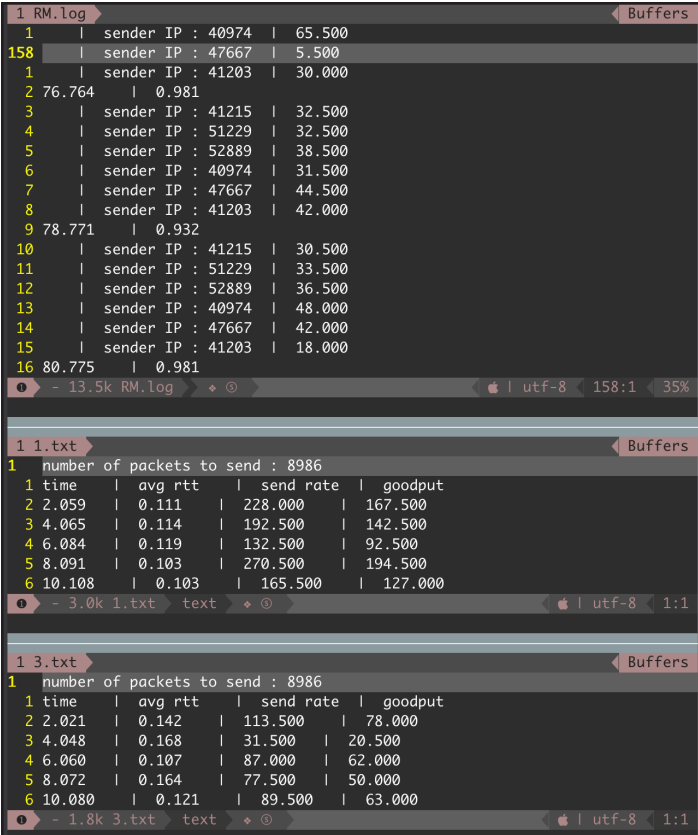
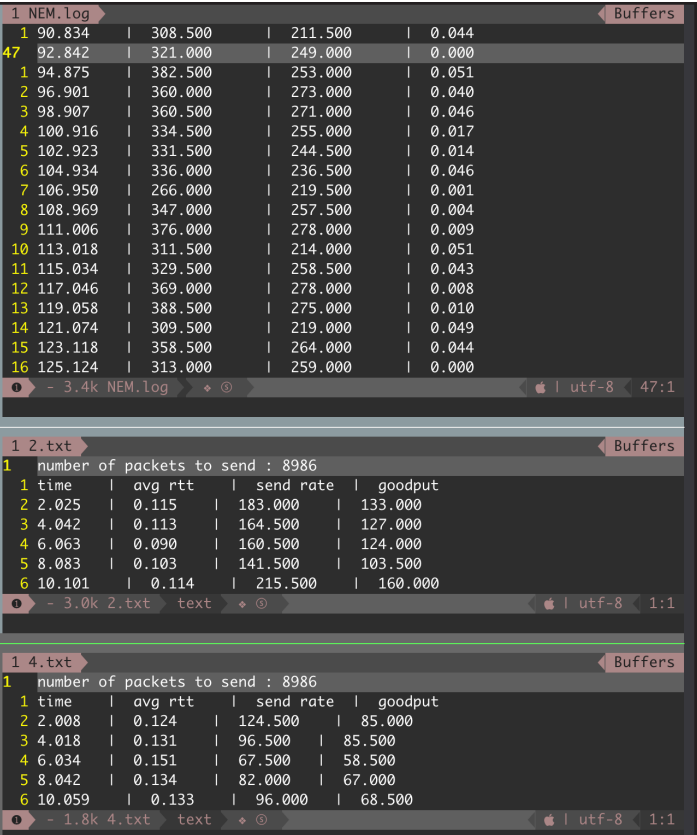
```
siisee11@siisee11ui-MacBook-Pro:~/Google Drive File Stream/My Drive/Study/Network/assignment5/2014310198_Namjaeyoun% python3 sender.py
Receiver IP address (127.0.0.1):
file name (hotel.jpg):
command>>start 5
command>>stop
Sender finished
siisee11@siisee11ui-MacBook-Pro:~/Google Drive File Stream/My Drive/Study/Network/assignment5/2014310198_Namjaeyoun%

siisee11@siisee11ui-MacBook-Pro:~/Google Drive File Stream/My Drive/Study/Network/assignment5/2014310198_Namjaeyoun% python3 receiver.py
configure>>100 10
socket recv buffer size: 786896
socket recv buffer size updated: 1000000

receiver program starts...
receiver wait...
time interval : 0.0100
```

HOW TO RUN

1. Below screen shot shows result.

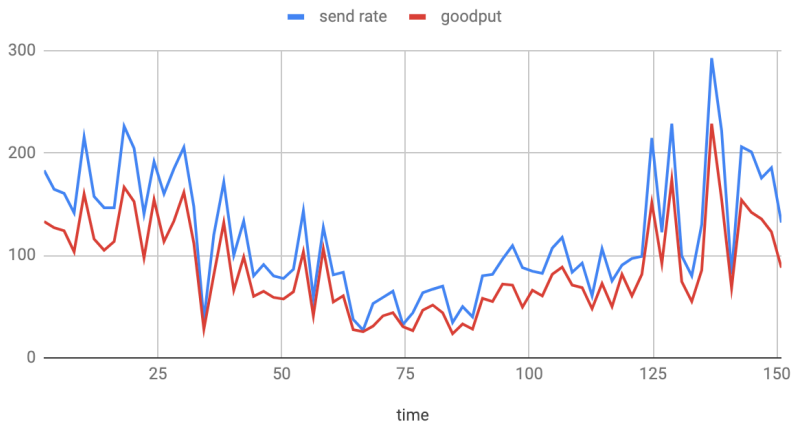
NEM.log sender1 sender3				RM.log sender2 sender4			
							

NETWORK

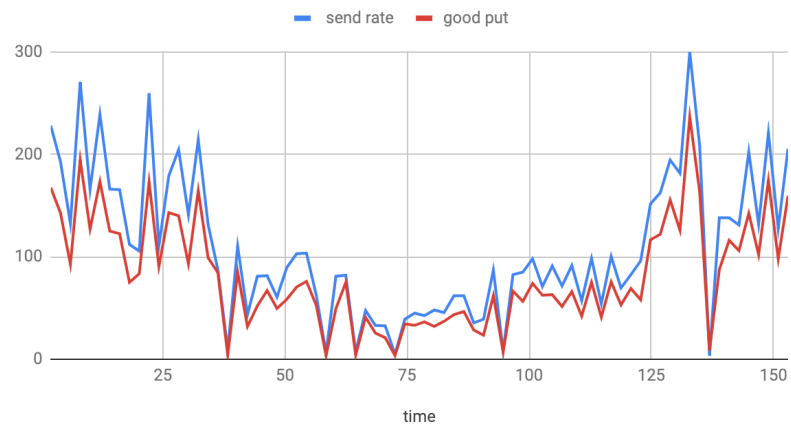
ASSIGNMENT #FIVE

GRAPH

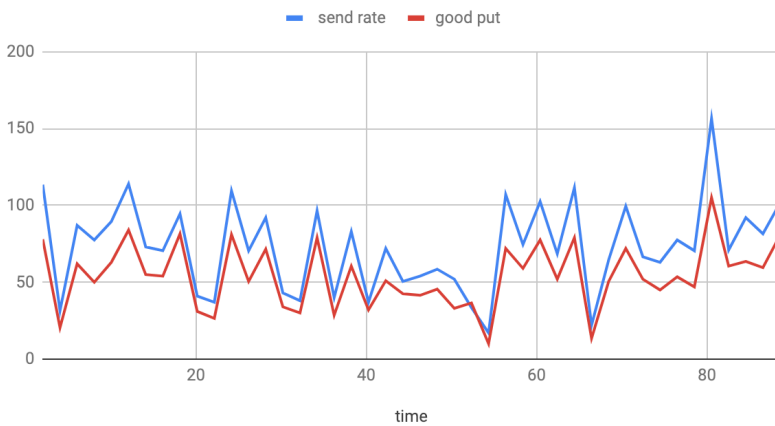
sender 1의 send rate 및 goodput



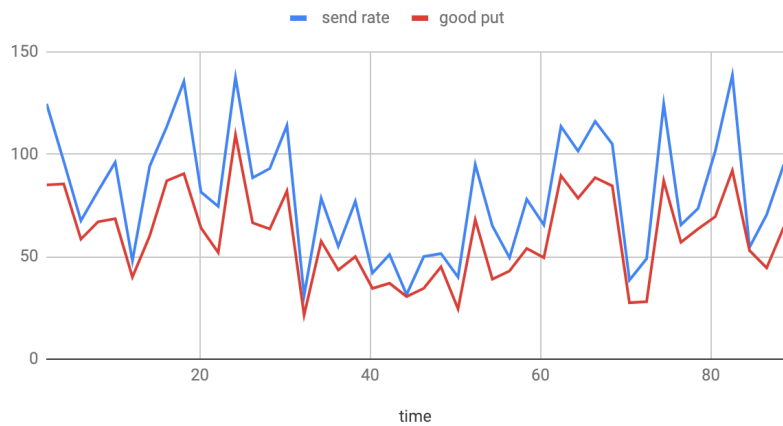
sender 2의 send rate, good put



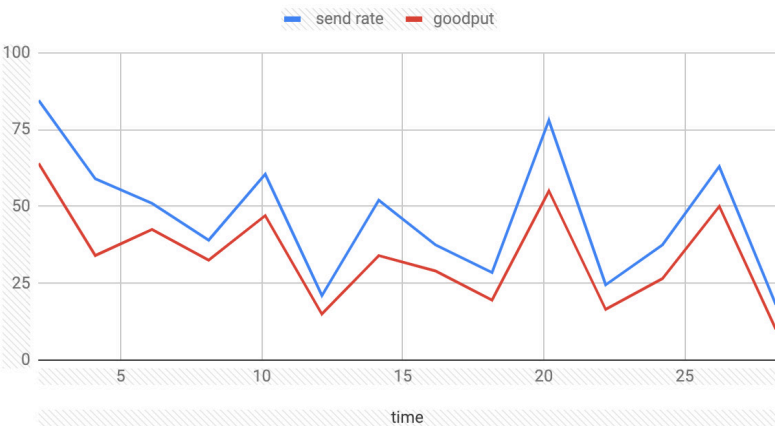
sender 3의 send rate, good put



sender 4의 send rate 및 good put

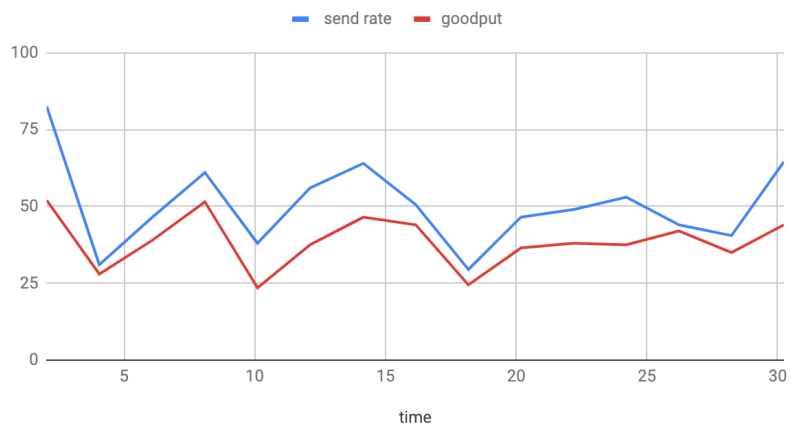


sender 5의 send rate 및 goodput



⋮

sender 6의 send rate 및 goodput

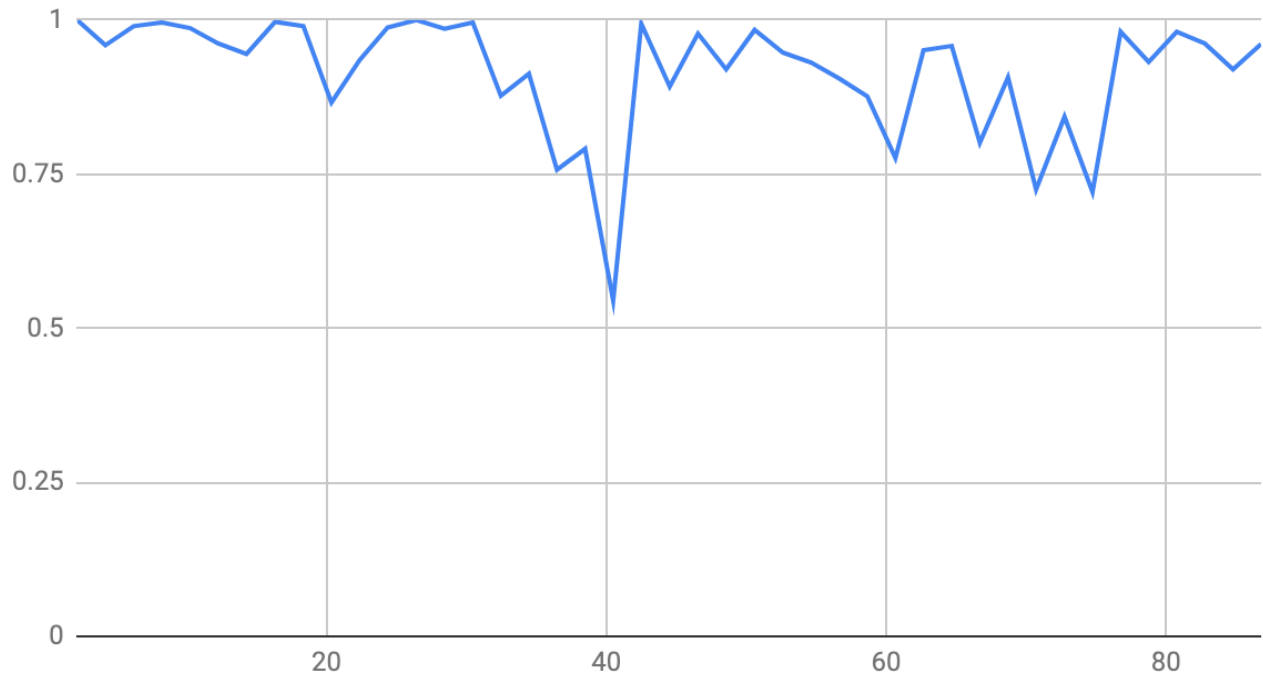


NETWORK

ASSIGNMENT #FIVE

GRAPH

Fairness graph



time에 대한 queue utilization의 값

