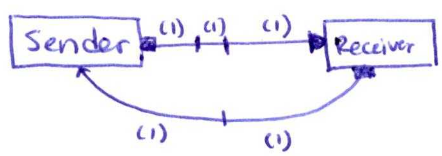
Carlos Lazo
ECE506 - Introduction to LAN/WAN
Homework # 7

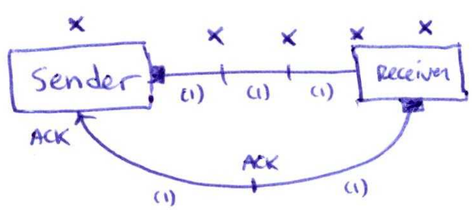
Problem # 1

* Normalized time to transmit fixed length segment = 1
* End-to-End propagation delay = 3
* Time to deliver data from received segment to transport user = 2
* Sender grants credit of 7 segments

→ Given this information, a pictoral representation can be drawn:



~ with a packet allocation / transmission of 1 segment / time interval, and given a conservative flow control policy with credit allocation at every step... ⇒ this implies that there are 5 unacknowledged packets at all times.



this means that at steady-state, the channel is in full at ALL times. (5 unacknowledged, 2 ACK)

∴ One system steady state is reached (segment flow is linear), S (Throughput) = 1. [normalized MAX]

```
// Carlos Lazo
// ECE 506 - Introduction to LAN/WAN
// Homework #7 - Problem #2

#include <iostream>
#include <vector>

using namespace std;

int main()
{

  // Define variables common to both parts of the problem.

  double alpha = .85;

  /* PART A */

  // Define vector to store all SRTT values accumulated.

  vector <double> SRTT_A(20);

  // Define initial vector conditions & variables.

  SRTT_A[0] = 3;  // SRTT(0) = 3 seconds
  int RTT_A = 1;  //  RTT(K) = 1 second

  for (int i = 1; i < SRTT_A.size(); i++)
  {
    SRTT_A[i] = (alpha * SRTT_A[i-1]) + ((1-alpha) * RTT_A);
    cout << "Value of SRTT(" << i << ") for Part A is: " << SRTT_A[i] << endl;
  }

  cout << endl;
  cout << "In Part A, calculated SRTT(19) = " << SRTT_A[SRTT_A.size()-1] << " seconds. " ↙
  << endl << endl;

  /* PART B */

  // Define vector to store all SRTT values accumulated.

  vector <double> SRTT_B(20);

  // Define initial vector conditions & variables.

  SRTT_B[0] = 1;  // SRTT(0) = 3 seconds
  int RTT_B = 3;  //  RTT(K) = 1 second

  for (int i = 1; i < SRTT_B.size(); i++)
  {
    SRTT_B[i] = (alpha * SRTT_B[i-1]) + ((1-alpha) * RTT_B);
    cout << "Value of SRTT(" << i << ") for Part B is: " << SRTT_B[i] << endl;
  }

  cout << endl;
  cout << "In Part B, calculated SRTT(19) = " << SRTT_B[SRTT_B.size()-1] << " seconds. " ↙
  << endl << endl;

  return 0;
}
```

3

```
/* PROGRAM OUTPUT */

/*

Value of SRTT(1) for Part A is: 2.7
Value of SRTT(2) for Part A is: 2.445
Value of SRTT(3) for Part A is: 2.22825
Value of SRTT(4) for Part A is: 2.04401
Value of SRTT(5) for Part A is: 1.88741
Value of SRTT(6) for Part A is: 1.7543
Value of SRTT(7) for Part A is: 1.64115
Value of SRTT(8) for Part A is: 1.54498
Value of SRTT(9) for Part A is: 1.46323
Value of SRTT(10) for Part A is: 1.39375
Value of SRTT(11) for Part A is: 1.33469
Value of SRTT(12) for Part A is: 1.28448
Value of SRTT(13) for Part A is: 1.24181
Value of SRTT(14) for Part A is: 1.20554
Value of SRTT(15) for Part A is: 1.17471
Value of SRTT(16) for Part A is: 1.1485
Value of SRTT(17) for Part A is: 1.12623
Value of SRTT(18) for Part A is: 1.10729
Value of SRTT(19) for Part A is: 1.0912

In Part A, calculated SRTT(19) = 1.0912 seconds.

Value of SRTT(1) for Part B is: 1.3
Value of SRTT(2) for Part B is: 1.555
Value of SRTT(3) for Part B is: 1.77175
Value of SRTT(4) for Part B is: 1.95599
Value of SRTT(5) for Part B is: 2.11259
Value of SRTT(6) for Part B is: 2.2457
Value of SRTT(7) for Part B is: 2.35885
Value of SRTT(8) for Part B is: 2.45502
Value of SRTT(9) for Part B is: 2.53677
Value of SRTT(10) for Part B is: 2.60625
Value of SRTT(11) for Part B is: 2.66531
Value of SRTT(12) for Part B is: 2.71552
Value of SRTT(13) for Part B is: 2.75819
Value of SRTT(14) for Part B is: 2.79446
Value of SRTT(15) for Part B is: 2.82529
Value of SRTT(16) for Part B is: 2.8515
Value of SRTT(17) for Part B is: 2.87377
Value of SRTT(18) for Part B is: 2.89271
Value of SRTT(19) for Part B is: 2.9088

In Part B, calculated SRTT(19) = 2.9088 seconds.

Press any key to continue . . .

*/
```

Problem #3

PART A

$R = 1 \times 10^9$ bits/sec

1 segment = 576 octets

$D = 60$ ms $= .060$ sec

$$W = \frac{RD}{data}$$

$$W = \frac{(1\times 10^9 \text{ bits/sec}) \cdot (.060 \text{ sec})}{576 \frac{\text{octets}}{1 \text{ segment}} \cdot 8 \frac{\text{bits}}{\text{octet}}}$$

$W = 13020.83$ segments
$W \cong 13021$ segments

~ Assuming a linear growth in window size from 1 (Jacobson), 13021 round trips need to be made.

$$\therefore \text{Total Time} = 13{,}021 \text{ round trips} \times \frac{.06 \text{ sec}}{1 \text{ round-trip}}$$

$$= 781.26 \text{ sec}$$

$$\boxed{\text{Total Time} = 13.021 \text{ minutes}}$$

PART B

* Change: 1 segment = 16,000 bytes = 16,000 octets

$$W = \frac{RD}{data}$$

$$W = \frac{(1\times 10^9 \text{ bits/sec}) \cdot (.060 \text{ sec})}{16000 \frac{\text{octets}}{\text{segment}} \cdot 8 \frac{\text{bits}}{\text{octet}}}$$

$W \cong 469$ segments

~ Assuming same conditions as Part A:

$$\therefore \text{Total Time} = 469 \text{ round trips} \times \frac{.06 \text{ sec}}{\text{round trip}}$$

$$\boxed{\text{Total Time} = 28.14 \text{ seconds}}$$