

## LAB 3: Principles of Reliable Data Transfer

- Ujjwal (1904139)  
Om Patil (1904125)

**Q1)** We have implemented Protocol\_rdt1 which works only when the channel is assumed to be ideal i.e. when  $P_c = 0$  and  $P_l = 0$ .

Lets first run the protocol with  $P_c = 0$  and  $P_l = 0$ . The screenshot of the output is shown below:

```
[Running] python -u "c:\Users\Om\Downloads\Testbench.py"
TIME: 4 SENDING APP: sending data 0
TIME: 4 RDT_SENDER: sending packet on data channel Packet(seq_num=0, payload=0, corrupted=False)
TIME: 4 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 5 SENDING APP: sending data 1
TIME: 5 RDT_SENDER: sending packet on data channel Packet(seq_num=1, payload=1, corrupted=False)
TIME: 5 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 6 RDT_RECEIVER: received packet from ack channel Packet(seq_num=0, payload=0, corrupted=False)
TIME: 6 RECEIVING APP: received data 0
TIME: 7 SENDING APP: sending data 2
TIME: 7 RDT_SENDER: sending packet on data channel Packet(seq_num=2, payload=2, corrupted=False)
TIME: 7 DATA_CHANNEL : udt_send called for Packet(seq_num=2, payload=2, corrupted=False)
TIME: 7 RDT_RECEIVER: received packet from ack channel Packet(seq_num=1, payload=1, corrupted=False)
TIME: 7 RECEIVING APP: received data 1
TIME: 9 RDT_RECEIVER: received packet from ack channel Packet(seq_num=2, payload=2, corrupted=False)
TIME: 9 RECEIVING APP: received data 2
TIME: 11 SENDING APP: sending data 3
TIME: 11 RDT_SENDER: sending packet on data channel Packet(seq_num=3, payload=3, corrupted=False)
TIME: 11 DATA_CHANNEL : udt_send called for Packet(seq_num=3, payload=3, corrupted=False)
TIME: 13 SENDING APP: sending data 4
TIME: 13 RDT_SENDER: sending packet on data channel Packet(seq_num=4, payload=4, corrupted=False)
TIME: 13 DATA_CHANNEL : udt_send called for Packet(seq_num=4, payload=4, corrupted=False)
TIME: 13 RDT_RECEIVER: received packet from ack channel Packet(seq_num=3, payload=3, corrupted=False)
TIME: 13 RECEIVING APP: received data 3
TIME: 15 RDT_RECEIVER: received packet from ack channel Packet(seq_num=4, payload=4, corrupted=False)
TIME: 15 RECEIVING APP: received data 4
TIME: 16 SENDING APP: sending data 5
TIME: 16 RDT_SENDER: sending packet on data channel Packet(seq_num=5, payload=5, corrupted=False)
TIME: 16 DATA_CHANNEL : udt_send called for Packet(seq_num=5, payload=5, corrupted=False)
TIME: 18 RDT_RECEIVER: received packet from ack channel Packet(seq_num=5, payload=5, corrupted=False)
```

Here, we can clearly verify that the Protocol runs perfectly under the above mentioned conditions.

In the next case let us consider  $P_c = 0.5$  (instead of 0). Following is the screenshot of the output:

```
[Running] python -u "c:\Users\Om\Downloads\Testbench.py"
TIME: 3 SENDING APP: sending data 0
TIME: 3 RDT_SENDER: sending packet on data channel Packet(seq_num=0, payload=0, corrupted=False)
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 3 DATA_CHANNEL : Packet(seq_num=0, payload=$h!T, corrupted=True) was corrupted!
TIME: 5 RDT_RECEIVER: received packet from ack channel Packet(seq_num=0, payload=$h!T, corrupted=True)
TIME: 6 SENDING APP: sending data 1
TIME: 6 RDT_SENDER: sending packet on data channel Packet(seq_num=1, payload=1, corrupted=False)
TIME: 6 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 8 RDT_RECEIVER: received packet from ack channel Packet(seq_num=1, payload=1, corrupted=False)
TIME: 8 RECEIVING APP: received data 1
ERROR!! RECEIVING APP: received wrong data: 1 ,expected: 0
Halting simulation...

[Done] exited with code=0 in 0.274 seconds
```

In the output we can very well see that the Protocol fails as in the end it outputs that there is some error due to receiving the wrong data packet and thus it is halting the simulation.

This error occurred because the first packet which was sent (packet '0') got corrupted (corrupted = true, payload = \$h!T) while flowing through the unreliable channel. Thus, the receiver received the corrupted packet but was waiting for the uncorrupted packet '0'. In the next iteration when the sender sends packet '1' it does not get corrupted, now the receiver receives packet '1' but it was expecting packet '0' so it halts the simulation as it realizes something went wrong and it can't rectify it.

**Q2)** Implementing Protocol\_rdt2 which is simple ACK/NAK based protocol which can work even if the data packets get corrupted.

We can verify its functionality by setting  $P_c > 0$  (for eg.  $P_c = 0.5$ ) for data packets. Following is the screenshot of the output :

```
[Running] python -u "c:\Users\Om\Downloads\Testbench.py"
TIME: 3 SENDING APP: trying to send data 0
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 5 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 5 RECEIVING APP: received data 0
TIME: 6 SENDING APP: trying to send data 1
TIME: 7 RDT_RECEIVER: sending ACK Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 11 SENDING APP: trying to send data 1
TIME: 11 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 11 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 13 SENDING APP: trying to send data 2
TIME: 13 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 15 RDT_RECEIVER: sending NAK Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 15 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 15 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 17 SENDING APP: trying to send data 2
TIME: 17 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 19 RDT_RECEIVER: sending NAK Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 19 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 19 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 20 SENDING APP: trying to send data 2
TIME: 21 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 23 RDT_RECEIVER: sending NAK Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 23 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 23 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 24 SENDING APP: trying to send data 2
TIME: 25 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
```

```
TIME: 23 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 23 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 24 SENDING APP: trying to send data 2
TIME: 25 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 27 RDT_RECEIVER: sending NAK Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 27 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 28 SENDING APP: trying to send data 2
TIME: 29 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 29 RECEIVING APP: received data 1
TIME: 30 SENDING APP: trying to send data 2
TIME: 31 RDT_RECEIVER: sending ACK Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 31 SENDING APP: trying to send data 2
TIME: 31 DATA_CHANNEL : udt_send called for Packet(seq_num=2, payload=2, corrupted=False)
TIME: 33 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 33 RECEIVING APP: received data 2
TIME: 35 RDT_RECEIVER: sending ACK Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 36 SENDING APP: trying to send data 3
TIME: 36 DATA_CHANNEL : udt_send called for Packet(seq_num=3, payload=3, corrupted=False)
TIME: 36 DATA_CHANNEL : Packet(seq_num=3, payload=$H!T, corrupted=True) was corrupted!
TIME: 38 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 40 SENDING APP: trying to send data 4
TIME: 40 RDT_RECEIVER: sending NAK Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 40 DATA_CHANNEL : udt_send called for Packet(seq_num=3, payload=3, corrupted=False)
TIME: 40 DATA_CHANNEL : Packet(seq_num=3, payload=$H!T, corrupted=True) was corrupted!
TIME: 41 SENDING APP: trying to send data 4
TIME: 42 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 44 RDT_RECEIVER: sending NAK Packet(seq_num=0, payload=NAK, corrupted=False)
```

From the above screenshot we can verify that the Protocol\_rdt2 works correctly even when there is some probability of data packets getting corrupted.

Q3) Here is a sample screenshot for  $P_c = 0.75$ .

```

17 rdt_sender = rdt_Sender(env)
18 rdt_receiver = rdt_Receiver(env)
19 channel_for_data = UnreliableChannel(env=env,Pc=0.75,P1=0,delay=2,name="DATA_CHANNEL")
20 channel_for_ack = UnreliableChannel(env=env,Pc=0,P1=0,delay=2,name="ACK_CHANNEL")
21
22 # connect the objects together

```

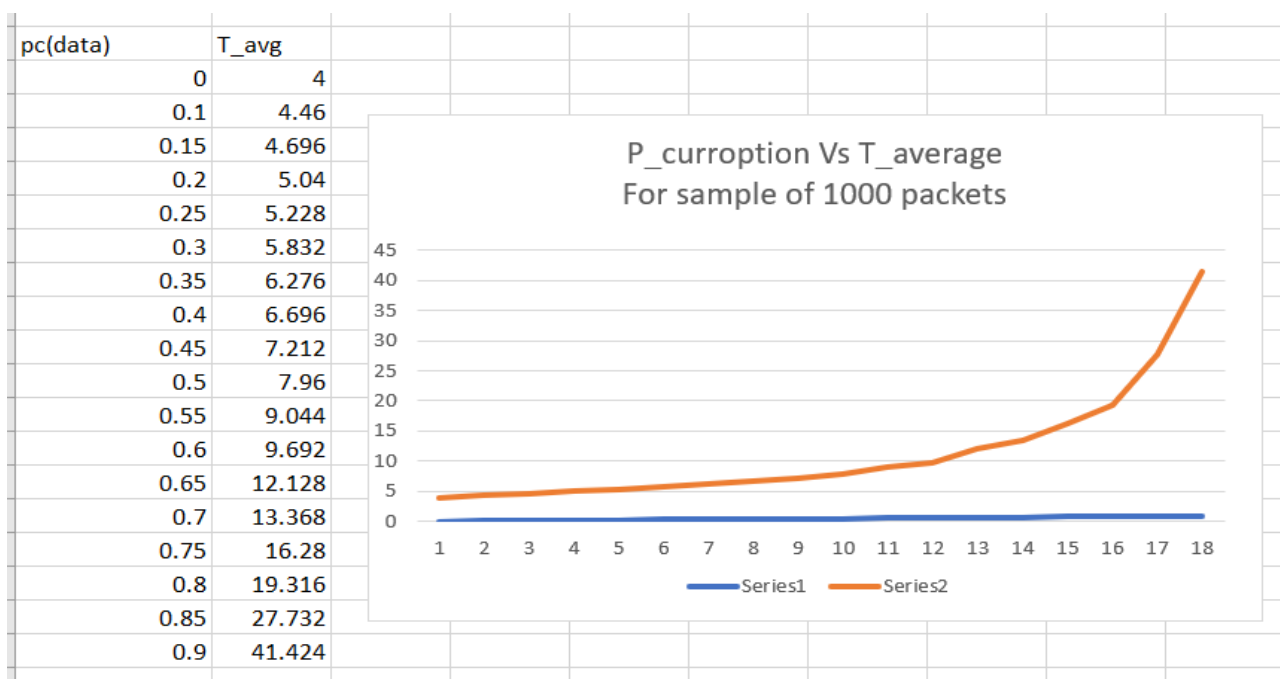
TERMINAL    DEBUG CONSOLE    PROBLEMS    OUTPUT

```

TIME: 18153 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 18155 DATA_CHANNEL : udt_send called for Packet(seq_num=998, payload=998, corrupted=False)
TIME: 18157 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 18157 RECEIVING APP: received data 998
TIME: 18162 DATA_CHANNEL : udt_send called for Packet(seq_num=999, payload=999, corrupted=False)
TIME: 18162 SENDING APP: sent data 999
TIME: 18162 DATA_CHANNEL : Packet(seq_num=999, payload=$H!T, corrupted=True) was corrupted!
TIME: 18164 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 18166 DATA_CHANNEL : udt_send called for Packet(seq_num=999, payload=999, corrupted=False)
TIME: 18168 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 18168 RECEIVING APP: received data 999
-----All Acks recived-----
16.28 Is the AVG time
PS D:\Desktop\Lab-3>

```

Similarly, doing for all we get the below graph which is exponential in nature.



The graph is exponential because as the Probability of Corruption increases more packets will get corrupted which implies we need to send even more number of packets for that one corrupted packet that means these packets may also get corrupted so more and more packets need to be sent which in turn increases the time.

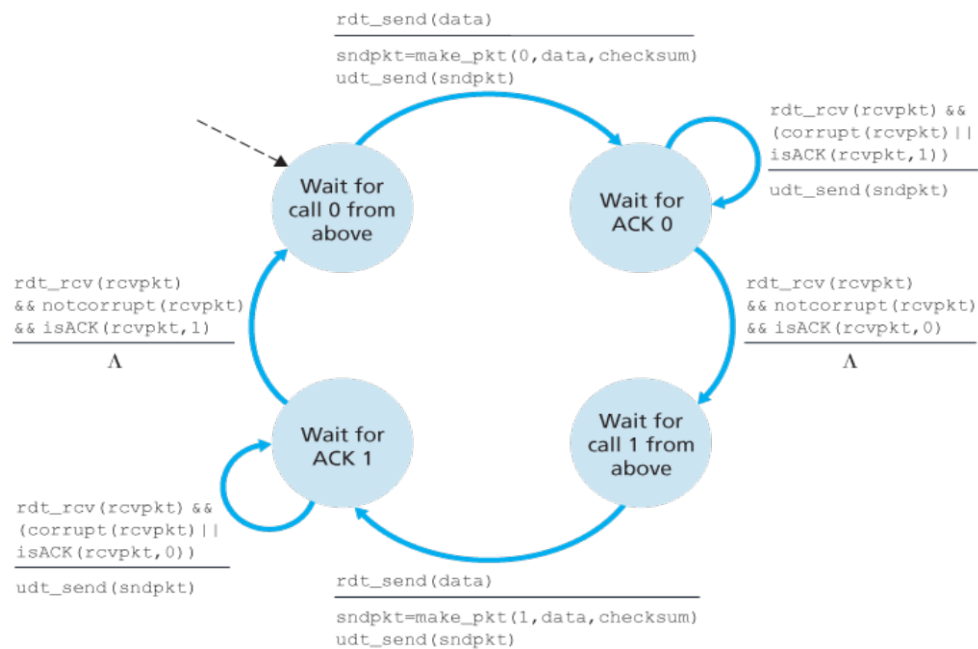
**Q4)** The Protocol\_rdt2 which we implemented in the 2nd question will not work when the acknowledgement packets start to get corrupted. Following is the screenshot of the output when the Pc for the ACK/NAK channel is greater than zero (Let, Pc = 0.5) :

```
[Running] python -u "c:\Users\Om\Downloads\Testbench.py"
TIME: 3 SENDING APP: trying to send data 0
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 5 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 5 RECEIVING APP: received data 0
TIME: 5 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 6 SENDING APP: trying to send data 1
ERROR! rdt_rcv() was expecting an ACK or a NAK. Received a corrupted packet.
Halting simulation...

[Done] exited with code=0 in 0.282 seconds
```

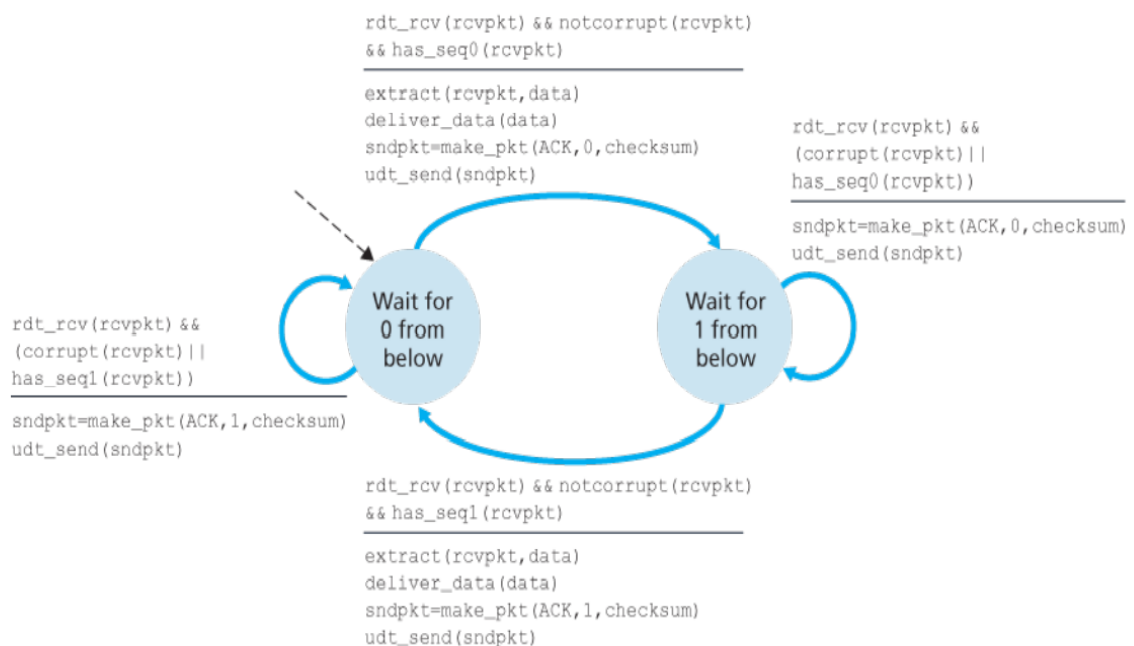
Here, we can observe that the output gives us an error specifying that the ACK/NAK which the sender received from the receiver was corrupted and therefore can't continue further with the simulation.

**Q5)** Following is the FSM model , we need to develop in our code :



Now there is no need for NAK, instead, we can give ACK of that packet which was not expected to serve as NAK.

Similarly, we develop at receiver :





The protocol Works well for both channels having probability of corruption 0.1 as shown below

```
9  from Protocol_rdt22 import *
10
11  # Create a simulation environment
12  env=simpy.Environment()
13
14  # Populate the simulation environment with objects:
15  sending_app = SendingApplication(env)
16  receiving_app = ReceivingApplication(env)
17  rdt_sender = rdt_Sender(env)
18  rdt_receiver = rdt_Receiver(env)
19  channel_for_data = UnreliableChannel(env=env,Pc=0.1,Pl=0,delay=2,name="DATA_CHANNEL")
20  channel_for_ack = UnreliableChannel(env=env,Pc=0.1,Pl=0,delay=2,name="ACK_CHANNEL")
21
22  # connect the objects together
23  # .....forward path...
24  sending_app.rdt_sender = rdt_sender
25  rdt_sender.channel = channel_for_data
```

TERMINAL    DEBUG CONSOLE    PROBLEMS    OUTPUT

```
TIME: 4973 RECEIVING APP: received data 710
TIME: 4973 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 4975 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=710, corrupted=False)
TIME: 4977 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4980 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=711, corrupted=False)
TIME: 4980 SENDING APP: sent data 711
TIME: 4982 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 4982 RECEIVING APP: received data 711
TIME: 4985 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=712, corrupted=False)
TIME: 4985 SENDING APP: sent data 712
TIME: 4987 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4987 RECEIVING APP: received data 712
TIME: 4990 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=713, corrupted=False)
TIME: 4990 SENDING APP: sent data 713
TIME: 4992 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 4992 RECEIVING APP: received data 713
TIME: 4996 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=714, corrupted=False)
TIME: 4996 SENDING APP: sent data 714
TIME: 4998 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4998 RECEIVING APP: received data 714
```

Q6)

- A) If the packet is lost, the sender is stuck in a listing state as it will not hear back any acknowledgment and thus, would be always truck in that state until simulation times out as we can observe below

```
13
14 # Populate the simulation environment with objects:
15 sending_app = SendingApplication(env)
16 receiving_app = ReceivingApplication(env)
17 rdt_sender = rdt_Sender(env)
18 rdt_receiver = rdt_Receiver(env)
19 channel_for_data = UnreliableChannel(env=env,Pc=0.1,Pl=0.4,delay=2,name="DATA_CHANNEL")
20 channel_for_ack = UnreliableChannel(env=env,Pc=0.1,Pl=0,delay=2,name="ACK_CHANNEL")
21
22 # connect the objects together
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
PS D:\Desktop\Lab-3> python .\Testbench.py
TIME: 5 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 5 SENDING APP: sent data 0
TIME: 7 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 7 RECEIVING APP: received data 0
TIME: 9 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 9 SENDING APP: sent data 1
TIME: 11 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 11 RECEIVING APP: received data 1
TIME: 11 ACK_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 13 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 13 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 15 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK1, corrupted=False)
TIME: 19 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=2, corrupted=False)
TIME: 19 SENDING APP: sent data 2
TIME: 19 DATA_CHANNEL : Packet(seq_num=0, payload=2, corrupted=False) was lost!
PS D:\Desktop\Lab-3>
```

B)

```
16 receiving_app = ReceivingApplication(env)
17 rdt_sender = rdt_Sender(env)
18 rdt_receiver = rdt_Receiver(env)
19 channel_for_data = UnreliableChannel(env=env,Pc=0.2,Pl=0.1,delay=2,name="DATA_CHANNEL")
20 channel_for_ack = UnreliableChannel(env=env,Pc=0.2,Pl=0.1,delay=2,name="ACK_CHANNEL")
21
22 # connect the objects together
23 # .....forward path...
24 sending_app.rdt_sender = rdt_sender
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
PS D:\Desktop\Lab-3> python .\Testbench.py
TIME: 4962 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4962 RECEIVING APP: received data 440
TIME: 4962 ACK_CHANNEL : Packet(seq_num=0, payload=ACK0, corrupted=False) was lost!
-----Retrying the lost packet-----
TIME: 4970 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=440, corrupted=False)
TIME: 4972 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4972 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 4974 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=440, corrupted=False)
TIME: 4974 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 4976 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4976 ACK_CHANNEL : Packet(seq_num=0, payload=ACK0, corrupted=False) was lost!
-----Retrying the lost packet-----
TIME: 4984 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=440, corrupted=False)
TIME: 4986 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 4990 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=441, corrupted=False)
TIME: 4990 SENDING APP: sent data 441
TIME: 4992 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 4992 RECEIVING APP: received data 441
TIME: 4995 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=442, corrupted=False)
TIME: 4995 SENDING APP: sent data 442
TIME: 4995 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 4997 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK1, corrupted=False)
TIME: 4997 ACK_CHANNEL : Packet(seq_num=0, payload=ACK1, corrupted=False) was lost!
PS D:\Desktop\Lab-3>
```



The developed protocol works well with the both loss and corruption probability in both Ack and Data channels

c)

Following is one example of the average time .

```

15  sending_app  = SendingApplication(env)
16  receiving_app = ReceivingApplication(env)
17  rdt_sender   = rdt_Sender(env)
18  rdt_receiver  = rdt_Receiver(env)
19  channel_for_data = UnreliableChannel(env=env,Pc=0.2,Pl=0.2,delay=2,name="DATA_CHANNEL")
20  channel_for_ack  = UnreliableChannel(env=env,Pc=0.2,Pl=0.2,delay=2,name="ACK_CHANNEL")
21
22  # connect the objects together
23  # .....forward path...

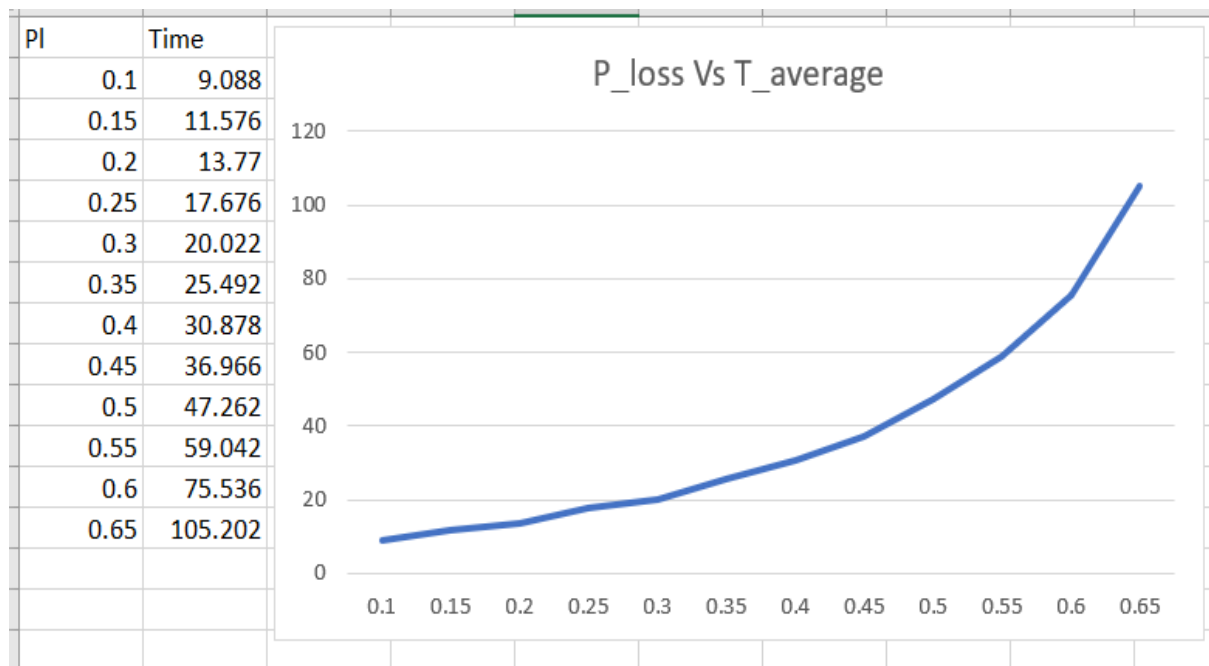
```

```

TERMINAL  DEBUG CONSOLE  PROBLEMS  OUTPUT
TIME: 14682 RECEIVING APP: received data 996
TIME: 14685 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=997, corrupted=False)
TIME: 14685 SENDING APP: sent data 997
TIME: 14685 DATA_CHANNEL : Packet(seq_num=1, payload=997, corrupted=False) was lost!
-----Retrying the lost packet-----
TIME: 14695 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=997, corrupted=False)
TIME: 14697 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 14697 RECEIVING APP: received data 997
TIME: 14700 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=998, corrupted=False)
TIME: 14700 SENDING APP: sent data 998
TIME: 14702 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 14702 RECEIVING APP: received data 998
TIME: 14705 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=999, corrupted=False)
TIME: 14705 SENDING APP: sent data 999
TIME: 14707 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 14707 RECEIVING APP: received data 999
-----All Acks received-----
13.264 Is the AVG time
PS D:\Desktop\Lab-3> 

```

We observe the following graph in this simulation



E)

```
44:         # Now wait for "delay" amount of time
45:
46:         delay_for_packet = random.randint(0,3*delay)
47:         yield self.env.timeout(delay_for_packet)
48:
49:
50:         # deliver the packet by calling the rdt_rcv()
51:         # function on the receiver side.
52:         self.receiver.rdt_rcv(packet)
```

TERMINAL    DEBUG CONSOLE    PROBLEMS    OUTPUT

```
TIME: 84 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 86 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 90 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=3, corrupted=False)
TIME: 90 SENDING APP: sent data 3
TIME: 94 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=3, corrupted=False)
TIME: 94 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 96 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 99 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 99 RECEIVING APP: received data 3
TIME: 105 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=4, corrupted=False)
TIME: 105 SENDING APP: sent data 4
TIME: 105 DATA_CHANNEL : Packet(seq_num=0, payload=4, corrupted=False) was lost!
TIME: 110 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=5, corrupted=False)
TIME: 110 SENDING APP: sent data 5
TIME: 110 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK1, corrupted=False)
TIME: 120 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=6, corrupted=False)
TIME: 120 SENDING APP: sent data 6
TIME: 130 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK0, corrupted=False)
TIME: 130 RECEIVING APP: received data 6
ERROR!! RECEIVING APP: received wrong data: 6 ,expected: 4
Halting simulation...
PS D:\Desktop\Lab-3>
```

The only modification we did in the code was, change the delay to , be any random number between 0 and  $3 \times \text{delay}$ . So  $3 \times \text{delay}$  servers as maximum delay faced by a packet.

And as we can see clearly, the code fails by reordering the packet.