Seyyed mohammad hosseini - 5g Laboratory – http://5glab.net – iasbs university
My Page : http://seyyedmohammad.ir

IN THE NAME OF GOD

How Calculate Queue's Size in Network Simulator 2 (NS2) ?

SeyyedMohammad Hosseini

IASBS UNIVERSITY

2016

To solve the problem, you need to Edit Your Queue's files That you used In Your Simulate.

For Example , I used Drop-tail Queue.

Find Your Queue's File in NS2's folder And Open theme (Example : Drop-tail.h And Drop-Tail.cc)

-.h file :

- **Add This Header :**
  - `#include <fcntl.h>`
- **Add A Function (Example : Void calculateQueue(Packet*P);) Under PROTECTED**
  - `void calculateQueue(Packet *p);`

- .cc file :

- **Find void DropTail::enque(Packet* p) And Call Your Function:**

```
1.  void DropTail::enque(Packet* p)
2.  {
3.      if (summarystats) {
4.                  Queue::updateStats(qib_?q_->byteLength():q_->length());
5.      }
6.
7.      int qlimBytes = qlim_ * mean_pktsize_;
8.      if ((!qib_ && (q_->length() + 1) >= qlim_) ||
9.      (qib_ && (q_->byteLength() + hdr_cmn::access(p)-
    >size()) >= qlimBytes)){
10.         // if the queue would overflow if we added this packet...
11.         if (drop_front_) { /* remove from head of queue */
12.             q_->enque(p);
13.             Packet *pp = q_->deque();
14.             drop(pp);
15.         } else {
16.             drop(p);
17.         }
18.     } else {
19.         calculateQueue(p); //call your function here
20.         q_->enque(p);
21.     }
22. }
```

- **Your Function Body**

```
1.  void DropTail:: calculateQueue(Packet* p){
2.      u_int32_t dst, src;
3.      struct hdr_mac802_11 *dh = HDR_MAC802_11(p);        // access MAC
4.      dst = ETHER_ADDR(dh->dh_ra);
5.      src = ETHER_ADDR(dh->dh_ta);
6.      fl = fopen("queue.tr","a");
7.      fprintf(fl,"%3.5f %7d %2d %2d\n ",Scheduler::instance().clock(),q_-
    >length(),src,dst);
8.      fclose(fl);
9.  }
```

- **At Finaly Compile Your NS2**

My Code

Drop-tail.h

```
1.  #ifndef ns_drop_tail_h
2.  #define ns_drop_tail_h
3.
4.  #include <string.h>
5.  #include "queue.h"
6.  #include "config.h"
7.  #include <fcntl.h>
8.
9.  /*
10.  * A bounded, drop-tail queue
11.  */
12. class DropTail : public Queue {
13.   public:
14.
15.     DropTail() {
16.
17.         q_ = new PacketQueue;
18.         pq_ = q_;
19.         bind_bool("drop_front_", &drop_front_);
20.         bind_bool("summarystats_", &summarystats);
21.         bind_bool("queue_in_bytes_", &qib_);  // boolean: q in bytes?
22.         bind("mean_pktsize_", &mean_pktsize_);
23.         //      _RENAMED("drop-front_", "drop_front_");
24.
25.     }
26.     ~DropTail() {
27.         delete q_;
28.     }
29.   protected:
30.     void reset();
31.     int command(int argc, const char*const* argv);
32.     void enque(Packet*);
33.     Packet* deque();
34.     void shrink_queue();    // To shrink queue and drop excessive packets.
35.     void calculateQueue(Packet *p);
36.     PacketQueue *q_;    /* underlying FIFO queue */
```

```
    int drop_front_;    /* drop-from-front (rather than from tail) */
    int summarystats;
    void print_summarystats();
    int qib_;           /* bool: queue measured in bytes? */
    int mean_pktsize_;  /* configured mean packet size in bytes */
    FILE *fl;
};

#endif
```

Drop-tail.cc

```
#ifndef lint
static const char rcsid[] =
    "@(#) $Header: /cvsroot/nsnam/ns-2/queue/drop-tail.cc,v 1.17 2004/10/28 23:35:37 haldar Exp $ (LBL)";
#endif

#include "drop-tail.h"
#include <god.h>
#include <mac-802_11.h>

static class DropTailClass : public TclClass {
 public:
    DropTailClass() : TclClass("Queue/DropTail") {}
    TclObject* create(int, const char*const*) {
        return (new DropTail);
    }
} class_drop_tail;

void DropTail::reset()
{
    Queue::reset();
}

int
DropTail::command(int argc, const char*const* argv)
{
    if (argc==2) {
        if (strcmp(argv[1], "printstats") == 0) {
            print_summarystats();
            return (TCL_OK);
        }
        if (strcmp(argv[1], "shrink-queue") == 0) {
            shrink_queue();
            return (TCL_OK);
        }
    }
    if (argc == 3) {
        if (!strcmp(argv[1], "packetqueue-attach")) {
            delete q_;
            if (!(q_ = (PacketQueue*) TclObject::lookup(argv[2])))
                return (TCL_ERROR);
            else {
                pq_ = q_;
                return (TCL_OK);
            }
        }
    }
    return Queue::command(argc, argv);
}
```

```
49.
50. /*
51.  * drop-tail
52.  */
53. void DropTail::enque(Packet* p)
54. {
55.     if (summarystats) {
56.                 Queue::updateStats(qib_?q_->byteLength():q_->length());
57.     }
58.
59.     int qlimBytes = qlim_ * mean_pktsize_;
60.     if ((!qib_ && (q_->length() + 1) >= qlim_) ||
61.     (qib_ && (q_->byteLength() + hdr_cmn::access(p)->size()) >= qlimBytes)){
62.         // if the queue would overflow if we added this packet...
63.         if (drop_front_) { /* remove from head of queue */
64.             q_->enque(p);
65.             Packet *pp = q_->deque();
66.             drop(pp);
67.         } else {
68.             drop(p);
69.         }
70.     } else {
71.         calculateQueue(p);
72.         q_->enque(p);
73.     }
74. }
75.
76. //AG if queue size changes, we drop excessive packets...
77. void DropTail::shrink_queue()
78. {
79.         int qlimBytes = qlim_ * mean_pktsize_;
80.     if (debug_)
81.         printf("shrink-
    queue: time %5.2f qlen %d, qlim %d\n",Scheduler::instance().clock(),q_-
    >length(), qlim_);
82.         while ((!qib_ && q_->length() > qlim_) ||
83.             (qib_ && q_->byteLength() > qlimBytes)) {
84.                 if (drop_front_) { /* remove from head of queue */
85.                         Packet *pp = q_->deque();
86.                         drop(pp);
87.                 } else {
88.                         Packet *pp = q_->tail();
89.                         q_->remove(pp);
90.                         drop(pp);
91.                 }
92.         }
93. }
94.
95. Packet* DropTail::deque()
96. {
97.         if (summarystats && &Scheduler::instance() != NULL) {
98.                 Queue::updateStats(qib_?q_->byteLength():q_->length());
99.         }
100.            return q_->deque();
101.        }
102.
103.        void DropTail::print_summarystats()
104.        {
105.            //double now = Scheduler::instance().clock();
106.                printf("True average queue: %5.3f", true_ave_);
107.                if (qib_)
108.                        printf(" (in bytes)");
109.                printf(" time: %5.3f\n", total_time_);
110.        }
```

```
111.        /*------ SeyyedMohammad Hosseini ------*/
112.        void DropTail::calculateQueue(Packet* p){
113.            u_int32_t dst, src;
114.            struct hdr_mac802_11 *dh = HDR_MAC802_11(p);        // access MAC
115.            dst = ETHER_ADDR(dh->dh_ra);
116.            src = ETHER_ADDR(dh->dh_ta);
117.            fl = fopen("queue.tr","a");
118.            fprintf(fl,"%3.5f %7d %2d %2d\n ",Scheduler::instance().clock(),q_-
    >length(),src,dst);
119.            fclose(fl);
120.        }
```

Result.txt

0.00278 0 2 -1

0.00364 0 1 2

0.00750 0 1 2

0.01027 0 1 2

0.01385 0 2 -1

0.02003 1 2 -1

0.02051 0 1 2

0.02231 0 0 2

0.02791 0 2 -1

0.02980 1 2 0

0.03075 0 1 2

0.03294 0 0 2

0.03646 0 0 2

0.03779 0 0 2

0.04099 1 0 2

0.04099 0 1 2

0.04334 0 2 1

0.04646 0 0 2

0.05114 1 2 0

0.05122 1 0 2

0.05122 1 1 2

0.05646 2 0 2

0.06146	3	0	2
0.06146	1	1	2
0.06307	1	2	1
0.06646	3	0	2
0.06887	2	2	1
0.07171	3	0	2
0.07171	2	1	2
0.08071	2	2	0
0.08195	4	0	2
0.08195	2	1	2
0.09219	5	0	2
0.09219	3	1	2
0.09262	2	2	1
0.09845	3	2	0
0.10243	5	0	2
0.10243	3	1	2
0.11267	6	0	2
0.11267	4	1	2
0.12226	1	2	1
0.12291	6	0	2
0.12291	5	1	2
0.12804	2	2	1
0.13315	6	0	2
0.13315	6	1	2
0.13982	2	2	0
0.14339	7	0	2
0.14339	6	1	2
0.14578	3	2	1
0.15163	4	2	0
0.15363	7	0	2

0.15363	6	1	2
0.15745	5	2	0
0.16387	8	0	2
0.16387	6	1	2
0.16939	5	2	1
0.17411	8	0	2
0.17411	7	1	2
0.17534	6	2	1
0.18435	8	0	2
0.18435	8	1	2
0.18700	6	2	0
0.19459	9	0	2
0.19459	8	1	2
0.19894	6	2	0
0.20478	7	2	1
0.20483	10	0	2
0.20483	8	1	2
0.21077	8	2	0
0.21507	10	0	2
0.21507	8	1	2
0.21661	9	2	0
0.22237	10	2	0
0.22531	11	0	2
0.22531	7	1	2
0.23402	10	2	1
0.23555	11	0	2
0.23555	8	1	2
0.24579	12	0	2
0.24579	9	1	2
0.24602	10	2	0

0.25185	11	2	1
0.25603	12	0	2
0.25603	9	1	2
0.26627	13	0	2
0.26627	10	1	2
0.26976	10	2	1
0.27651	13	0	2
0.27651	11	1	2
0.28140	10	2	0
0.28675	14	0	2
0.28675	11	1	2
0.28733	11	2	0
0.29315	12	2	0
0.29699	15	0	2
0.29699	10	1	2
0.29898	13	2	1
0.30495	14	2	0
0.30723	15	0	2
0.30723	10	1	2
0.31662	14	2	0
0.31747	16	0	2
0.31747	10	1	2
0.32771	17	0	2
0.32771	11	1	2
0.32852	14	2	1
0.33441	15	2	0
0.33795	17	0	2
0.33795	11	1	2
0.34639	15	2	1
0.34819	17	0	2

0.34819	12	1	2
0.35225	16	2	0
0.35843	18	0	2
0.35843	12	1	2
0.36867	19	0	2
0.36867	13	1	2
0.36999	15	2	1
0.37891	19	0	2
0.37891	14	1	2
0.38915	20	0	2
0.38915	15	1	2
0.39939	21	0	2
0.39939	16	1	2
0.40770	11	2	0
0.40963	22	0	2
0.40963	16	1	2
0.41352	12	2	0
0.41987	23	0	2
0.41987	16	1	2
0.42515	12	2	0
0.43011	24	0	2
0.43011	16	1	2
0.44035	25	0	2
0.44035	17	1	2
0.44335	11	2	1
0.44918	12	2	0
0.45059	25	0	2
0.45059	17	1	2
0.46079	12	2	1
0.46083	26	0	2

0.46083	18	1	2
0.46657	13	2	1
0.47107	25	0	2
0.47107	19	1	2
0.48131	26	0	2
0.48131	20	1	2
0.48448	12	2	1
0.49155	26	0	2
0.49155	21	1	2
0.49644	12	2	1
0.50179	26	0	2
0.50179	22	1	2
0.50227	13	2	0
0.51203	27	0	2
0.51203	22	1	2
0.51392	13	2	0
0.52227	28	0	2
0.52227	22	1	2
0.52560	13	2	1
0.53151	14	2	0
0.53251	28	0	2
0.53251	22	1	2
0.53729	15	2	0
0.54275	29	0	2
0.54275	22	1	2
0.54924	15	2	0
0.55299	30	0	2
0.55299	22	1	2
0.55500	16	2	0
0.56323	31	0	2

0.56323	22	1	2
0.57347	32	0	2
0.57347	23	1	2
0.57837	14	2	1
0.58371	32	0	2
0.58371	24	1	2
0.58417	15	2	1
0.59395	32	0	2
0.59395	25	1	2
0.59621	15	2	1
0.60419	32	0	2
0.60419	26	1	2
0.60818	15	2	1
0.61401	16	2	0
0.61443	32	0	2
0.61443	26	1	2
0.62102	17	2	1
0.62467	32	0	2
0.62467	27	1	2
0.62678	18	2	1
0.63306	19	2	1
0.63491	31	0	2
0.63491	28	1	2
0.63891	20	2	1
0.64515	31	0	2
0.64515	29	1	2
0.65539	32	0	2
0.65539	30	1	2
0.65692	19	2	0
0.66274	20	2	0

0.66563	33	0	2
0.66563	29	1	2
0.67456	20	2	0
0.67587	34	0	2
0.67587	29	1	2
0.68040	21	2	1
0.68611	34	0	2
0.68611	30	1	2
0.69228	21	2	0
0.69635	35	0	2
0.69635	30	1	2
0.70422	21	2	0
0.70659	36	0	2
0.70659	30	1	2
0.71601	21	2	0
0.71683	37	0	2
0.71683	30	1	2
0.72707	38	0	2
0.72707	31	1	2
0.72783	21	2	1
0.73361	22	2	1
0.73731	37	0	2
0.73731	32	1	2
0.73956	23	2	0
0.74533	24	2	0
0.74755	38	0	2
0.74755	31	1	2
0.75119	25	2	0
0.75779	39	0	2
0.75779	31	1	2

0.76803	40	0	2
0.76803	32	1	2
0.76902	24	2	0
0.77487	25	2	1
0.77827	40	0	2
0.77827	32	1	2
0.78665	25	2	1
0.78851	40	0	2
0.78851	33	1	2
0.79869	25	2	1
0.79875	41	0	2
0.79875	34	1	2
0.80452	26	2	1
0.80899	40	0	2
0.80899	35	1	2
0.81032	27	2	1
0.81923	40	0	2
0.81923	36	1	2
0.82226	27	2	1
0.82947	40	0	2
0.82947	37	1	2
0.83399	27	2	1
0.83971	40	0	2
0.83971	38	1	2
0.83985	28	2	1
0.84563	29	2	1
0.84995	39	0	2
0.84995	39	1	2
0.85149	30	2	1
0.85733	31	2	0

0.86019	39	0	2
0.86019	39	1	2
0.87043	40	0	2
0.87043	40	1	2
0.87513	30	2	0
0.88067	41	0	2
0.88067	40	1	2
0.88702	30	2	0
0.89091	42	0	2
0.89091	40	1	2
0.89289	31	2	1
0.89865	32	2	1
0.90115	41	0	2
0.90115	41	1	2
0.91035	32	2	0
0.91139	42	0	2
0.91139	41	1	2
0.92163	43	0	2
0.92163	42	1	2
0.92216	32	2	1
0.92809	33	2	0
0.93187	43	0	2
0.93187	42	1	2
0.94211	44	0	2
0.94211	43	1	2
0.95148	31	2	0
0.95235	45	0	2
0.95235	43	1	2
0.95735	32	2	1
0.96259	45	0	2

0.96259	44	1	2
0.96324	33	2	0
0.96911	34	2	0
0.97283	46	0	2
0.97283	43	1	2
0.98073	34	2	0
0.98307	47	0	2
0.98307	43	1	2
0.98659	35	2	0
0.99247	36	2	0
0.99331	48	0	2
0.99331	42	1	2
0.99839	37	2	1
1.00355	48	0	2
1.00355	43	1	2
1.00426	38	2	0
1.01379	49	0	2
1.01379	43	1	2
1.01614	38	2	1
1.02208	39	2	1
1.02403	48	0	2
1.02403	44	1	2
1.02792	40	2	0
1.03427	49	0	2
1.03427	44	1	2
1.04451	50	0	2
1.04451	45	1	2
1.05176	38	2	0
1.05475	51	0	2
1.05475	45	1	2

1.05760	39	2	1
1.06345	40	2	0
1.06499	51	0	2
1.06499	45	1	2
1.07507	40	2	1
1.07523	52	0	2
1.07523	46	1	2
1.08116	41	2	0
1.08547	52	0	2
1.08547	46	1	2
1.09571	53	0	2
1.09571	47	1	2
1.09877	40	2	1
1.10595	53	0	2