Performance Analysis of *ns3* Models of 802.11n Adhoc Grid Networks with Linear Traffic Flows

James M. Westall

June 25, 2020

Abstract

This document describes aspects the use of *ns3* to model the performance of adhoc 802.11n wireless networks. The models are build upon a middleware system that is designed to be easily configurable by a problem domain analyst and easily extensible by a software engineer with *ns3* expertise.

It is shown that a regular grid with linear flows from one nodes on one edge to corresponding nodes on the other edge obtain *very* poor performance. As the physical separation between the linear flows increases performance improves until it reaches approximately 2.5 times the maximum range for successful reception. From that point on the grid performs as n independent linear networks.

1 Introduction

1.1 The simulated network

For the results reported here the simulated network is an adhoc 802.11n with 20 Mhz bandwidth using the short (400ns) intersymbol gap. The network is configured as an $n_x \times n_y$ planar rectangular grid where the n values specify the number of nodes in each dimensions. The distance between nodes d_x and d_y is configurable in both dimensions. The wireless grid can be connected to a remote servers via a configurable number of disjoint two hop point to point back haul paths. For the results reported here, no backhaul is used.

2 Overview of 802.11n at 20Mhz with a single spatial stream

IEEE 802.11n networks employ orthogonal frequency division multiplexing *OFDM* in which a wideband channel is subdivided into a large number of subchannels operating in parallel. Each OFDM physical layer supports a discrete set of *modulation and coding schemes (MCS)*. These are defined in the associated standard. The modulation scheme defines the total number of data and ECC

bits encoded in each symbol as shown in table 1. The phase shift keying (PSK) schemes employ only phase modulation while the quadrature amplitude modulation (QAM) schemes modulate both phase and amplitude.

Table 1: Modulation schemes

Name	Symbol States	Bits per Symbol
BPSK	2	1
QPSK	4	2
QAM-16	16	4
QAM-64	64	6

The product, $symbol\_rate \times bits\_per\_symbol \times number\_of\_subchannels$ defines the *coded bit rate* of the channel. For each modulation scheme, an OFDM protocol can support multiple *coding rates*. The coding rate defines the ratio of data bits to data plus error correcting code *ECC* bits. Higher coding rates require higher signal to noise and interference ratios *SINRs*.

The *nominal bit rate* of a wireless channel is the product of the current coded bit rate and the current coding rate. It is maximum data bit rate that could be delivered to the UE in the absence of all management and arbitration overhead. We define the *effective bit rate* of a channel to be the bit rate achieved by a UE when the UE has elastic demand and is the only UE attempting to transmit. The effective bit rate includes the impact of management and arbitration overhead which can consume up to 50% of channel bandwidth. Though each standard precisely defines the supported MCS codings, the mapping from SINR to specific MCS is left to the equipment vendor.

2.1 802.11n performance model

For a 20 Mhz 802.11n channel, and for each modulation and coding scheme (MCS) the the nominal bit rate is the product of

- The OFDM symbol rate: The symbol time is $3.6\mu s$ for a $400ns$ inter-symbol gap time. So the rate is a constant $(1/3.6\mu s) = 0.2\bar{7}Msym/chan/sec$
- The number of data carrying channels: This is a constant equal to 52. Therefore, the aggregate symbol rate is $52 \times 0.2\bar{7} = 14.\bar{4}Msym/sec$.
- The number of coded bits per symbol: This includes ECC bits and depends on the modulation scheme as (BPSK - 1, QPSK - 2, QAM16 - 4, QAM64 - 6)
- The coding rate: This is the ratio of data bits to data and ECC bits and can be (1/2, 2/3, 3/4, 5/6) depending on modulation and coding scheme.

The MCS ID, name, nominal data bit rates, and effective rates are shown for 802.11n with a single spatial stream in table 2. The effective bit rate computation assumes a single sender with elastic demand sending standard 1500 byte NPDUs with a single NPDU carried by each PPDU and each PPDU requiring the standard CSMA-CA channel arbitration protocol. The effective bit rate is computed by dividing 12246 (the number of bits in a PPDU carrying a 1500 bytes by the total

channel time (including framing and arbitration overhead) required to send it. The efficiency is computed as the time to send the 12246 bits of the PPDU divided by the sum of the time to the PPDU, PPDU framing overhead, and CSMA-CA channel arbitration overhead. The effective bit rate of the PPDU is then the nominal bit rate multiplied by the efficiency. Details of the model underlying table 2 are in Appendix A.

Table 2: 802.11n MCS data

MCS	Name	Coding Rate	Coded Bits	Coded Bit Rate	Nom Bit Rate	Effective Bit Rate	Efficiency
0	BPSK 1/2	0.500	1	14.4	7.2	6.46	0.89
1	QPSK 1/2	0.500	2	28.9	14.4	11.97	0.83
2	QPSK 3/4	0.750	2	28.9	21.7	16.79	0.77
3	Q-16 1/2	0.500	4	57.8	28.9	20.95	0.73
4	Q-16 3/4	0.750	4	57.8	43.3	27.89	0.64
5	Q-64 2/3	0.667	6	86.7	57.8	33.50	0.58
6	Q-64 3/4	0.750	6	86.7	65.0	35.98	0.55
7	Q-64 5/6	0.833	6	86.7	72.2	38.20	0.53

Note that the effective bit rate does not include any operating system induced delays or network induced delays caused by routing protocols or other overhead. Consequently it should be viewed as upper bound that is not likely achievable in practice.

2.2 Improving efficiency

As shown in table 2, at the highest nominal bit rates, efficiency barely exceeds 50 %. Therefore, the IEEE 802.11n protocol defines three methods for increasing efficiency The 802.11n standard defines two types of frame aggregation: MAC service data unit (MSDU) aggregation and MAC protocol data unit (MPDU) aggregation. Both types of aggregation group several MAC or PHY data frames into one large frame that incurs the cost of framing and arbitration only once.

The third mechanism is the TXOP. The TXOP allows a sender to avoid arbitration overhead and continue to send a stream of PPDUs for a specified amount of time when a transmission succeeds. All frames sent in the TXOP incur the cost of arbitration only once.

All three of these methods can make significant increases to efficiency, especially for the higher MCS levels, but they trade potentially significant increases in latency for the increase in efficiency. For that reason, they are not considered in this work.

3 *ns3* models and performance

In this section we analyze three *ns3* simulation models of a grid type adhoc 802.11n Wifi network. These networks are: a two node linear network with a single simplex traffic stream; a seven node linear network with a single simplex traffic stream from node 0 to node 6; and a five by seven grid network with 5 linear traffic streams identical to the single stream in in the linear network. We

analyze latency, throughput, and loss characteristics while varying MCS, inter node distance and transmission rate.

The results reported here were obtained with the *ns3-quic* variant of the *ns3* system. The *ns3-quic* variant is built upon the *ns3-dev* version from *nsnam.org*. Default settings for power and antennas were used.

3.1 The baseline model

The purpose of this simple model was compare and evaluate the results obtained from an *ns3* simulation with the analytic results shown in table 2 and to determine the effective coverage range of each MCS in the nominal distance units of *ns3*.

The model is a linear grid with $n_x = 2$ and $n_y = 1$. Node 0 transmits and Node 1 receives. The model uses the default power and antenna gain and a single spatial stream. The values of d_x and MCS index are varied during the test.

3.1.1 Latency and distance coverage

In this test for each MCS index in (0-7), d_x was increased in steps of one distance unit until communication was lost. For each value of d_x, a simulated thirty second burst of UDP packets of APDU size 1472 bytes were sent at approximately one packet per second rate from node 0 to node 1 for 30 seconds of simulated time. The ns reported latency for every packet was logged. Table 3 summarizes the results.

Table 3: Latency and range

MCS	Lat(NS)	Lat(Mod)	No Delay	No Loss	Max Dist
0	1746	1794	91	98	100
1	892	948	72	78	80
2	608	663	55	63	65
3	464	523	43	47	49
4	324	383	32	37	39
5	252	311	24	26	27
6	226	289	21	24	24
7	208	271	20	22	22

3.1.2 One way latency

Two values of latency are reported. the column labeled *Lat(NS)* contains the minimum end to end latency captured at the application layer by the *ns3* model. The column labeled *Lat(Mod)* reports latency as computed by the analytic model presented in Appendix A. The analytic model predicts consistently higher latency with the difference between the predictions increasing from 49 μs to 63 μs as the MCS index increases.

3.1.3 Two way latency

To capture two way latency we used a TCP workload with a small d_x that enabled optimal performance at all MCS levels. NPDU size remained 1500 bytes.

Our *ns3* modeling middleware has hooks that capture each packet at each hop in which it is processed by either by the simulated application or ipV4l3. Each packet processed is logged with a action code, connection ID, processing *ns3* nodeId, processing timestamp, the packet's *ns3* unique identifier and the application layers assigned sequence number. Action codes are list here:

```
#define ACT_TRANSMIT 0      // app layer send
#define ACT_SOG      1      // ipv4l3 send outgoing
#define ACT_UCFWD    2      // ipv4l3 unicast forward
#define ACT_LDELIVER 3      // ipv4l3 local deliver
#define ACT_RECEIVE  4      // app layer receive
#define ACT_ECHOXMIT 5      // app layer echo of udp ping
#define ACT_ACKSEND  6      // ipv4l3 ack send outgoing
#define ACT_ACKFWD   7      // ipv4l3 unicast forward
#define ACT_ACKRECV  8      // ipv4l3 ack local deliver
```

Post processing of the raw log data creates an action log for each simulated packet which is sorted by connection ID and app sequence number with acks matched to the packet that triggered the ack. Traces of two packets are shown in table 4.

Table 4: Packet processing action log

Action	cxId	Node	Latency	ns3 Uid	Seq
0	0	0	0	157	3
1	0	0	0	157	3
3	0	1	324	157	3
4	0	1	0	157	3
6	0	1	200000	162	3
8	0	0	54	162	3
0	0	0	0	166	4
1	0	0	0	166	4
3	0	1	324	166	4
4	0	1	0	166	4
6	0	1	200000	171	4
8	0	0	54	171	4

The *Latency* column shows the time in microseconds relative to the last processing of the simulated packet. It can be seen that 0 simulated time elapses as packets move up and down the protocol stack within the simulated send and receiving nodes but that at MCS 4, the one way latency between send outgoing and local deliver is 324 μs as previously reported. The 200 ms delay between receipt and ack transmission is the standard TCP delayed ack triggered by the low transmission rate.

For MCS 4, the delay from ack transmission to ack receipt is is consistently 54 μs for each packet in the *ns3* simulation and it continues to decrease as MCS gets larger. For the analytic model, the ack

latency is a constant 56 μs for MCS 4 and larger because the ack requires 2 symbols in each case. We conclude from this that the *ns3* model approximates the actual OFDM encoding and for each packet computes the transmission time based upon a computed effective bit rate and the number of bits in the packet.

Table 5: Ack and 2-way latency

MCS	Ack (Ns)	Ack(mod)	Total (NS)	Total(Mod)	Ack Diff	Total diff
0	140	103	1886	1897	-37	11
1	90	77	982	1025	-13	43
2	72	67	680	730	-5	50
3	65	59	529	582	-6	53
4	54	56	378	439	2	61
5	51	56	303	367	5	64
6	50	56	276	345	6	69
7	47	56	255	327	9	72

3.1.4 Distance effects

As distance, d_x, increases the performance degrades through 4 phases:

- No loss and all packets experience minimum latency
- No loss but some packets experience a small multiple of minimum latency. This indicates that errors are occurring but that retransmission attempts are (eventually) successful.
- Some loss and even longer latency for some packets occurs. This indicates that some retransmission attempts are unsuccessful
- All packets are lost.

The last three columns of table 3 report for each MCS the largest distance between nodes associated with each of the first three states itemized above. For MCS 0, no loss and no latency increase occurs until the distance exceeds 91. For distance 92-98 there is no loss but gradual increase in mean latency. For distance 99 and 100 there is packet loss. Approximate 10% loss occurs at distance 99 and 50% at distance 100.

For all values of MCS, optimal performance is achieved over approximately 90% of the distance range where connectivity is possible and the distance range in which degraded connectivity exists is only 10%.

3.1.5 *ns-3.30* maximum distance

The experiment was conducted again on an unmodified *ns-3.30* installation. The results for MCS 0 are summarized in table 6.

The monitored UDP connection was run from time 20 to time 50. A short 1.5 second UDP connection was run from time 18 to time 19.5 in order to eliminate ARP resolution delay. first packet

Table 6: Latency and range

Dist	Tx Rate Kbps	Rx Rate Kbps	Mean Latency
51.0	12.6	12.6	1.746
51.1	12.6	12.6	1.746
51.2	12.6	12.6	1.746
51.3	12.6	12.6	1.746
51.4	12.6	12.6	1.746
51.5	12.6	0.0	-
51.6	12.6	0.0	-
51.7	12.6	0.0	-
51.8	12.6	0.0	-
51.9	12.6	0.0	-

of the short connection had a latency of 6.210 ms due to ARP resolution but all the rest had 1.746 ms latency.

It was found that the maximum range for MCS 0 was reduced by half to 51.4 units. Unlike the *ns3-quic* test there was no gradual increase in loss and latency as the distance approached the maximum for MCS 0 through MCS 2.

The change in maximum distance from *ns3-quic* to *ns3-3.0* was strongly non-linear. Table 3 is replicated for *ns3-3.0* in table 7. MCS 0 through MCS 2 all give identical maximum range with no degradation before maximum range is reached. However, minimum latency values are within $1\mu s$ in both tables for all MCS indices. For MCS 7, the maximum range values between *ns3-quic* to *ns3-3.0* are comparable.

Table 7: Latency and range

MCS	Lat(NS)	Lat(Mod)	No Delay	No Loss	Max Dist
0	1746	1794	51.0	51.0	51.0
1	893	948	51.0	51.0	51.0
2	608	663	51.0	51.0	51.0
3	464	523	44.5	48.0	50.0
4	324	383	36.0	38.0	39.5
5	252	311	24.5	26.0	27.5
6	227	289	22.0	24.0	25.0
7	209	271	20.0	22.0	22.5

3.2 Maximum achievable throughput

The baseline model was then used to compare the maximum achievable UDP throughput with the upper bound predicted by the analytic model. Grid spacing d_x was 4 in all tests. This provided no packet loss or throughput degradation due to MAC layer retransmissions. For the results reported here the *OLSR* routing protocol was enabled, and our modeling middleware showed a four packet exchange of 36 byte NPDUS every 2 seconds.

The results are shown in table 8. For each MCS index the offered load was incrementally increased by 0.1 Mbps until maximum received throughput was reached. APDU(NS) maximum throughput was captured from the *ns3* model and PPDU(ns) was computed by multiplying the by the ratio of the two sizes. The column PPDU(Mod) is the maximum effective throughput predicted by the analytic model.

Table 8: Maximum throughput (Mbps)

MCS	Nom Rate	APDU(NS)	PPDU(NS)	PPDU(Mod)	Diff
0	7.2	6.1	6.5	6.5	0.0
1	14.4	11.2	11.8	12.0	0.2
2	21.7	15.2	16.1	16.8	0.6
3	28.9	18.9	20.0	20.9	1.0
4	43.3	24.4	25.8	27.9	2.1
5	57.8	28.7	30.3	33.5	3.2
6	65.0	30.5	32.3	36.0	3.7
7	72.2	32.0	33.9	38.2	4.3

3.2.1 Latency at maximum throughput

The baseline model can be viewed as a (D/G/1) queuing system. The intervals between arrivals of packets for transmission is deterministic as implemented in the model. The service time is general because each packet can experience a uniform arbitration delay of 0-7 slots which corresponds to 0-63 μs. Consequently as the offered load approaches the the service rate queue length can expected to be bounded only by the modeled queue capacity.

The default queue discipline (*Qdisc*) in *ns3* is the *PFifoFast* discipline of *Linux* which performs no AQM. Table 9 shows that latency at high throughput exceeds minimum latency by two orders of magnitude.

For all MCS values the per packet latency is initially equal to the values shown in table 3, grows progressively. The queue length in the 30 seconds runs executed here never reached the maximum of approximately 750 packets in the queue that can be achieved with longer runs.

The *Qlen* value is computed as *MaxLat* as reported in table **??**) converted from *msecs* to *secs* and multiplied by simulated throughput in PPDUs per second.

The values shown in the table are certainly unacceptable for low latency applications and demonstrate the need for AQM or other measures to manage latency under high loads.

3.3 The linear model

Proceeding in the direction of the full grid, we next analyze a linear network in which $n_x = 7$ and $n_y = 1$. The sender is node 0 and the receiver is node 6. It is obvious that if d_x is sufficiently small then the end to end path can and will be covered in a single hop. By choosing d_x with consideration

Table 9: Latency at maximum throughput (msecs))

MCS	MeanLat	MaxLat	Pkts/Sec	Qlen @ Max Lat
0	176	354	521	184
1	177	350	947	332
2	183	363	1297	471
3	94	189	1604	304
4	132	247	2071	511
5	57	115	2434	279
6	83	165	2593	427
7	125	192	2720	523

of the MCS in use and *Max Dist* column of table 3 it is also obvious that the end to end path can be made to require 2, 3, or 6 hops or be unusable.

3.3.1 Routing

In the *ns3* system, if an adhoc network is to support store and forward routing at the network layer, it is necessary to enable a dynamic routing protocol such as *OLSR* or *AODV* or to handcraft static routing tables for each node.

We have evaluated the OLSR and AODV routing protocols on a simulated 5×5 a grid network with 4 back haul of back haul network gateways (one at each corner of the grid). OLSR was demonstrated to do a much better job of load balancing than did AODV. These tests were conducted using the dynamic MCS management capability of *ns3*. The baseline model previously presented used static MCS management.

In constructing the linear model we encountered behavior that indicated OLSR was using MCS 0 for all routing exchanges regardless of whether the constant rate manager was selected and an MCS other than 0 was used.

Routes were set up that minimized hops at MCS 0 but, depending on d_x, could render the route unusable by "user data" sent at a fixed MCS other than 0. For example, when d_x was 15 the distance between node 0 and 6 was 90 and OLSR would create a single hop route direct from node 0 to node 6. This route was functional at MCS 0 but no other.

There may be a way to force OLSR to operate at a fixed MCS other than 0, but we have not yet discovered it. It is the case that static routes can support the use of fixed MCS, for this simple network but we have not yet pursued the use of MCS index other than 0 with static routes.

The first results presented in this section were obtained with OLSR routing, a fixed MCS 0, and $d_x = 80$. This ensures a 6 hop route.

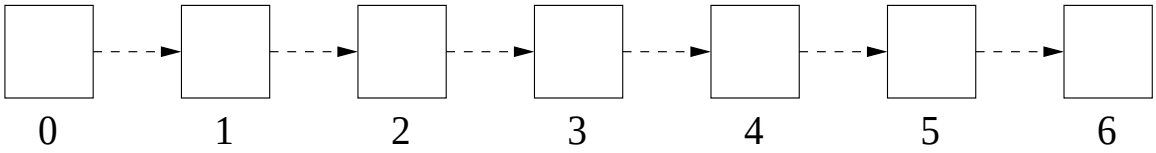

Figure 1: Seven node linear network with $d_x = 80$

3.3.2 Optimal throughput and latency

The linear network is depicted in figure 1. The distance between adjacent nodes is 80. Consequently, node 2 cannot decode communications between node 0 and node 1, but node 2 cannot transmit successfully to node 3 while node node 0 is transmitting to node 1 because node 1 can decode *both* node 0 and node 2. In particular a *maximum* of 2 successful transmissions can be simultaneously in progress on the seven node net.

Each packet must be transmitted 6 times while traversing the network. Therefore, the maximum sustainable throughput is 1/3 of the baseline throughput.

Nevertheless, the maximum distance at which a transmission can be successfully received is definitely *less than* the distance at which it can interfere with another. It will be shown that in practice, when d_x is 80 the only transmissions that can be overlapped are node 0 to node 1 with node 5 to node 6.

Moreover, for reasons that are not well understood, but associated with OLSR, the linear network experienced anomalous performance problems at various offered loads commencing at 400 Kbps. These included multisecond total loss bursts. It will be subsequently shown that these bursts are attributable to loss of routing connectivity.

At low loads when there is at most a single packet in the network at any instant it time, end to end latency should approximate 6 $\times$ the minimum one hop latency of 1746 μs = 10.476 ms. In practice store and forward latency was slightly higher than first hop latency. But the end-to-end latency was less than 10% higher than the absolute lower bound.

3.3.3 Observed latency at low loads

Table 10 shows hop by hop latency for packets 614 and 615. The values shown are representative of the entire trace. Both have the minimum latency of the first hop but marginally higher latencies thereafter. The mean end to end latency for the 620 packets transferred was 11.347ms which is within 1ms of the lower bound of 10.476ms.

3.3.4 Observed throughput and latency at high loads

Throughput and latency are shown for the 7 node linear network in table 11. The data was collected from a 40 second transmission burst beginning at time 20 and ending at time 60. Reported rates are at the APDU (1472 bytes per packet) level. Inter packet transmission time is computed by dividing 8 $\times$ 1472 by the nominal bit rate.

Table 10: Per packet per hop latency

Action	cxId	Node	Rel Time	Lapsed Time	ns3 Uid	Seq
0	0	0	0	0	13711	614
1	0	0	0	0	13711	614
2	0	1	1746	1746	13711	614
2	0	2	1930	3676	13711	614
2	0	3	1948	5624	13711	614
2	0	4	1940	7564	13711	614
2	0	5	1930	9494	13711	614
3	0	6	1903	11397	13711	614
4	0	6	0	11397	13711	614
0	0	0	0	0	13730	615
1	0	0	0	0	13730	615
2	0	1	1746	1746	13730	615
2	0	2	1895	3641	13730	615
2	0	3	1858	5499	13730	615
2	0	4	1948	7447	13730	615
2	0	5	1903	9350	13730	615
3	0	6	1958	11308	13730	615
4	0	6	0	11308	13730	615

Observed bit rates are computed by dividing the total bits transmitted or received by the last attempted transmit time minus the first attempted transmit time. Initial transmit times are randomized to prevent undesirable synchronous behavior when multiple simulated connections are present. Actual start time is computed here as 20 + U(0, 1) - 0.5. For the results reported here, the actual start time was 20.31 seconds. Transmission stops when the time of the next transmit exceeds the stop time (60). Packets received after the nominal stop time are included in the performance data.

The anomalous behavior can first be observed in table 11 when the nominal rate is 400 Kbps. Note that the measured Tx rate is 355.3 and the measured Rx rate is slightly lower at 350.9. These values reflect two sources of packet loss.

The reduction in measured transmit rate as compared to nominal reflects *Tx Rejects* in which the call to *SendTo* fails with a return code of -1. The further reduction from 355.3 to 350.9 reflects packets that are dropped in the network after being successfully sent.

An example of a *Tx Reject* is shown below from the log. Notice that the routing table no longer has a route to the destination which has IP address 192.168.1.7.

```
Tx FAILED at seq 2595 on CxId0

Node: 0, Time: +42.5s, Local time: +42.5s, Ipv4ListRouting table
  Priority: 100 Protocol: ns3::olsr::RoutingProtocol
Node: 0, Time: +42.5s, Local time: +42.5s, OLSR Routing table
Destination      NextHop  Interface Distance
192.168.0.2    192.168.0.2    1       1
```

Table 11: Throughput and latency at varying load ($d_x = 80$)

Nominal Rate(Kbps)	Tx Rate	Rx Rate	One way Latency (ms)
200	203.6	203.6	11.381
300	305.3	305.3	11.369
400	355.3	350.9	11.370
500	508.6	508.6	11.360
600	610.2	608.8	11.356
700	711.9	526.7	11.368
800	813.6	813.6	11.357
900	863.6	739.0	11.364
1000	784.8	759.3	12.549
1100	873.2	715.9	12.619
1200	1127.3	960.5	12.349
1300	1057.2	952.7	13.926
1400	1063.8	910.4	250.042
1500	1213.9	829.3	494.820

```
192.168.0.3     192.168.0.2     1       2
192.168.0.4     192.168.0.2     1       3
192.168.0.5     192.168.0.2     1       4
```

This shows that the cause of the *Tx Reject* at low loads is routing failure at the sending node. Consequently drops later along the path at low loads are due to routing failure failure along the path. Per hop latency in the action logs demonstrate that *none* was caused by queuing overflows until the nominal rate reached 1400 Kbps where the latency begins explosive growth.

Table 12: Packet loss characteristics at varying load ($d_x = 80$)

Nominal Rate(Kbps)	Tx Attempts	Tx Pkts	Rx Pkts	Tx Rejects	Drops	Drops per Event
200	687	687	687	0	0	0.0
300	1029	1029	1029	0	0	0.0
400	1372	1198	1183	174	15	15.0
500	1714	1714	1714	0	0	0.0
600	2057	2057	2052	0	5	5.0
700	2399	2399	1775	0	624	208.0
800	2742	2742	2742	0	0	0.0
900	3085	2911	2491	174	420	420.0
1000	3427	2645	2559	782	86	86.0
1100	3770	2943	2413	827	530	75.7
1200	4112	3799	3237	313	562	140.5
1300	4455	3563	3211	892	352	176.0
1400	4797	3585	3068	1212	517	4.4
1500	5140	4091	2795	1049	1296	8.1

Because of the anomalous behavior that was observed, two additional Tx strategies were attempted.

In the first one RTS/CTS was enabled. In the second RTS/CTS was disabled but Gaussian noise with a variety of standard deviations ranging from 10^-5 to 10^-3 was added to the base inter-packet time.

All three transmit strategies demonstrated similar anomalous behavior, but at differing transmit speeds and with different magnitudes. The following conclusions can be drawn from considering the tables together.

- The latency values show that no significant queuing occurs until the transmit rate reaches 1400 Kpbs.
- The losses that occur at rates 400 Kbps - 1300 Kbps are not caused by full queues.
- At a transmit rate of 1000 Kbps interpacket time is 0.011 seconds and so a 500 packet loss burst occupies 5.5 seconds of simulated time.
- The offered load becomes unsustainable at 1400 Kbps.

3.3.5 Burst loss example

The dynamics of a typical burst lost are shown in table 13. This is taken from the 500 Kbps workload with deterministic packet interarrival times. The workload has a single burst loss commencing at packet 425. Packet 424 shows perfectly normal hop latencies with zero queuing delays. Then packet 425 is successfully transmitted from node 0 to node 1 minimum latency and then it is dropped before reaching node 2. This process is repeated until packet 515 after which node 1 becomes unreachable. The entire lost burst consumes over 6 seconds of simulated time. At packet 615 the system resumes perfectly normal operation which is maintained until the simulation ends packet 1293.

3.3.6 Maximum system occupancy and overlap

Minimal system residency time of 11.4 ms is maintained up to 1200 Kbps. This corresponds to 101.9 packets per sec. The mean system population is thus $101.9 \times 0.0114 = 1.16$ packets.

For a multistage queuing system with homogeneous stages (such as this on) with X_{stage} equal to the maximum throughput of a single stage then the throughput X of an M stage system is $X_{stage} \times \bar{N}/M$, where $\bar{N}$ is mean population of the system. If $\bar{N} = 1$ then transactions are single threaded through the system and $X = X_{stage}/M$. If $\bar{N} = M$ then transactions are single threaded through the system and $X = X_{stage}$.

In this case, occupancy is 1 1/6 which indicates that in stable operation if node 0 is transmitting to node 1 then it is possible for node 5 to overlap its transmission to node 6. But otherwise only a single node can be transmitting. The distance from node 0 to node 3 is 240 and, since it was shown in the preceding section that the maximum distance at which a successful transmission at MCS 0 was 100 we thought it possible that node 0 to node 1 and node 3 to node 4 could be overlapped. Moreover node 0 to node 1 and node 4 to node 5 apparently can't be overlapped either even though they are a distance of 320 apart. So it appears that interference distance is at least 160.

Table 13: Burst loss example

Action	Node	Rel Time	Lapsed time	Abs Time	Seq
0	0	0	0	9800063	424
1	0	0	0	9800063	424
2	1	1746	1746	9801809	424
2	2	1922	3668	9803731	424
2	3	1885	5553	9805616	424
2	4	1975	7528	9807591	424
2	5	1894	9422	9809485	424
3	6	1877	11299	9811362	424
4	6	0	11299	9811362	424
0	0	0	0	9823231	425
1	0	0	0	9823231	425
2	1	1746	1746	9824977	425
0	0	0	0	9846399	426
1	0	0	0	9846399	426
2	1	1746	1746	9848145	426
0	0	0	0	11931519	516
1	0	0	0	11931519	516
0	0	0	0	16078591	695
1	0	0	0	16078591	695
0	0	0	0	16101759	696
1	0	0	0	16101759	696
2	1	1746	1746	16103505	696
2	2	1957	3703	16105462	696
2	3	1967	5670	16107429	696
2	4	1984	7654	16109413	696
2	5	1957	9611	16111370	696
3	6	1957	11568	16113327	696
4	6	0	11568	16113327	696

The 160 interference distance is certainly believable but the 10+ seconds loss of service bursts with only a single connection are not understood by us at this time.

3.4 Static routing

To confirm that the anomalous behavior was due to routing failure we repeated the linear network experiment with static routing. The results are shown in tables 14 and 15.

There are no *Tx Rejects* at any load. There is no packet loss at offered loads up to and including 1300 Kbps. Latency remains minimal until the load reaches 1400 Kbps just as it did with OLSR routing.

Table 14: Throughput and latency static routing ($d_x = 80$)

Nominal Rate(Kbps)	Tx Rate	Rx Rate	One way Latency (ms)
200	203.6	203.6	11.375
300	305.3	305.3	11.360
400	406.9	406.9	11.358
500	508.6	508.6	11.358
600	610.2	610.2	11.349
700	711.9	711.9	11.353
800	813.6	813.6	11.351
900	915.2	915.2	11.354
1000	1016.9	1016.9	11.360
1100	1118.5	1118.5	11.359
1200	1220.2	1220.2	11.373
1300	1321.8	1321.8	13.454
1400	1423.5	1209.8	569.658
1500	1525.2	1174.7	628.130

Table 15: Packet loss characteristics with static routing ($d_x = 80$)

Nominal Rate(Kbps)	Tx Attempts	Tx Pkts	Rx Pkts	Tx Rejects	Drops	Drops per Event
200	687	687	687	0	0	0.0
300	1029	1029	1029	0	0	0.0
400	1372	1372	1372	0	0	0.0
500	1714	1714	1714	0	0	0.0
600	2057	2057	2057	0	0	0.0
700	2399	2399	2399	0	0	0.0
800	2742	2742	2742	0	0	0.0
900	3085	3085	3085	0	0	0.0
1000	3427	3427	3427	0	0	0.0
1100	3770	3770	3770	0	0	0.0
1200	4112	4112	4112	0	0	0.0
1300	4455	4455	4455	0	0	0.0
1400	4797	4797	4077	0	720	2.3
1500	5140	5140	3959	0	1181	3.2

3.5 Static routing with $d_x = 60$

To further illustrate the effects of interference we reran static routing with d_x reduced to 60. As shown in table 16, for nominal rate 1100 Kbps there is no any longer overlapped transmission at all.

Packet 55 is sent and processed by the send outgoing component of Ipv4l3 at time 568669. But instead of the minimal first hop latency seen when $d_x = 80$ as node 0 waits for node 5 to complete its transmission of packet 54 to node 6. Packet 54 is received at time 571516 and then packet 55 is forwarded no node 1 at 572504.

Table 16: No packet overlap

Action	Node	Rel Time	Lapsed time	Abs Time	Seq
0	0	0	0	558138	54
1	0	0	0	558138	54
2	1	3691	3691	561829	54
2	2	1876	5567	563705	54
2	3	1948	7515	565653	54
2	4	1985	9500	567638	54
2	5	1948	11448	569586	54
3	6	1930	13378	571516	54
4	6	0	13378	571516	54
0	0	0	0	568669	55
1	0	0	0	568669	55
2	1	3835	3835	572504	55
2	2	1957	5792	574461	55
2	3	1948	7740	576409	55
2	4	1894	9634	578303	55
2	5	1850	11484	580153	55
3	6	1867	13351	582020	55
4	6	0	13351	582020	55

The throughput, latency, and packet loss results are shown in tables 17 and 18. Because of the loss of all transmission overlap loss free throughput is limited to a nominal rate of 1100 Kbps after which the loss rate increases linearly. As with the previous static routing example, there is none of the *Tx Rejects* commonly seen with OLSR routing.

Table 17: Throughput and latency static routing ($d_x = 60$)

Nominal Rate(Kbps)	Tx Rate	Rx Rate	One way Latency (ms)
200	203.6	203.6	11.374
300	305.3	305.3	11.360
400	406.9	406.9	11.361
500	508.6	508.6	11.357
600	610.2	610.2	11.349
700	711.9	711.9	11.354
800	813.6	813.6	11.359
900	915.2	915.2	11.366
1000	1016.9	1016.9	11.370
1100	1118.5	1118.5	13.386
1200	1220.2	1133.5	479.841
1300	1321.8	1104.7	532.581
1400	1423.5	1093.2	562.684
1500	1525.2	1081.3	567.961

Table 18: Packet loss characteristics with static routing ($d_x = 60$)

Nominal Rate(Kbps)	Tx Attempts	Tx Pkts	Rx Pkts	Tx Rejects	Drops	Drops per Event
200	687	687	687	0	0	0.0
300	1029	1029	1029	0	0	0.0
400	1372	1372	1372	0	0	0.0
500	1714	1714	1714	0	0	0.0
600	2057	2057	2057	0	0	0.0
700	2399	2399	2399	0	0	0.0
800	2742	2742	2742	0	0	0.0
900	3085	3085	3085	0	0	0.0
1000	3427	3427	3427	0	0	0.0
1100	3770	3770	3770	0	0	0.0
1200	4112	4112	3820	0	292	1.5
1300	4455	4455	3723	0	732	2.4
1400	4797	4797	3684	0	1113	3.0
1500	5140	5140	3644	0	1496	3.3

3.6 Optimizing for ARP with $d_x = 80$

One final optimization permits low latency loss free throughput at offered loads up to nominal loads up to 1400 Kbps. To eliminate delays associated with ARP resolution at the start of transmission we introduced a low throughput connection from node 0 to 6 that commenced 3 seconds before the monitored connection began and terminated 1 second before the monitored connection and reran the experiment.

The impact of the arp delay at an offered load of 1400 Kbps is shown in table 19 The top half of the table shows traces of the first two packets of the monitored connection without ARP pre-resolution. The bottom half shows the first two packets with ARP pre-resolution.

Table 19: No packet overlap

Action	Node	Rel Time	Lapsed time	Abs Time	Seq
0	0	0	0	0	1
1	0	0	0	0	1
2	1	8210	8210	8210	1
2	2	7365	15575	15575	1
2	3	9282	24857	24857	1
2	4	11828	36685	36685	1
2	5	8452	45137	45137	1
3	6	6228	51365	51365	1
4	6	0	51365	51365	1
0	0	0	0	8274	2
1	0	0	0	8274	2
2	1	3784	3784	12058	2
2	2	5475	9259	17533	2
2	3	11122	20381	28655	2
2	4	9933	30314	38588	2
2	5	8506	38820	47094	2
3	6	6147	44967	53241	2
4	6	0	44967	53241	2
0	0	0	0	0	1
1	0	0	0	0	1
2	1	1746	1746	1746	1
2	2	1894	3640	3640	1
2	3	1912	5552	5552	1
2	4	1859	7411	7411	1
2	5	1957	9368	9368	1
3	6	1957	11325	11325	1
4	6	0	11325	11325	1
0	0	0	0	8274	2
1	0	0	0	8274	2
2	1	1746	1746	10020	2
2	2	1876	3622	11896	2
2	3	1958	5580	13854	2
2	4	1903	7483	15757	2
2	5	1867	9350	17624	2
3	6	1894	11244	19518	2
4	6	0	11244	19518	2

At 1400 Kbps when ARP pre-resolution is not done, the system is driven into an unrecoverable unstable state and mean overall latency is 569.658ms as previously shown in table 14.

When ARP pre-resolution is done the system remains in the low latency, high throughput, no loss state for the duration of the run as shown in tables 20 and 21.

Table 20: Throughput and latency static routing ($d_x = 80$)

Nominal Rate(Kbps)	Tx Rate	Rx Rate	One way Latency (ms)
100	102.0	102.0	11.336
200	203.6	203.6	11.334
300	305.3	305.3	11.333
400	406.9	406.9	11.332
500	508.6	508.6	11.332
600	610.2	610.2	11.330
700	711.9	711.9	11.330
800	813.6	813.6	11.330
900	915.2	915.2	11.330
1000	1016.9	1016.9	11.331
1100	1118.5	1118.5	11.331
1200	1220.2	1220.2	11.332
1300	1321.8	1321.8	11.330
1400	1423.5	1423.5	11.330
1500	1525.2	1219.5	605.450

Table 21: Packet loss characteristics with static routing ($d_x = 80$)

Nominal Rate(Kbps)	Tx Attempts	Tx Pkts	Rx Pkts	Tx Rejects	Drops	Drops per Event
100	344	344	344	0	0	0.0
200	687	687	687	0	0	0.0
300	1029	1029	1029	0	0	0.0
400	1372	1372	1372	0	0	0.0
500	1714	1714	1714	0	0	0.0
600	2057	2057	2057	0	0	0.0
700	2399	2399	2399	0	0	0.0
800	2742	2742	2742	0	0	0.0
900	3085	3085	3085	0	0	0.0
1000	3427	3427	3427	0	0	0.0
1100	3770	3770	3770	0	0	0.0
1200	4112	4112	4112	0	0	0.0
1300	4455	4455	4455	0	0	0.0
1400	4797	4797	4797	0	0	0.0
1500	5140	5140	4110	0	1030	2.9

3.7 The grid model

In this section we describe the results of extending the linear model to a rectangular grid. The grid consists of five instances of the seven node linear model spaced at various distances d_y. As before all transmissions are six hop linear. Node 0 transmits to node 6, node 7 to node 13, etc.

It is obvious that if d_y is sufficiently large, then the system operates as 5 independent linear networks. It is also true that if the linear nets transmit on different channels crosstalk interference between them will be reduced.

In this set of experiments assumptions are:

- $d_x = 80$;
- All 35 nodes use the same channel;
- There is a single end to end flow on each of the five linear nets;
- The five monitored flows start at time 20 + (U(0.0, 1.0) - 1);
- The five monitored flows transmit at a specified constant rate for 60 seconds;
- Static routes are used;
- Pre-resolution of ARP bindings is employed;

Consequently, this configuration represents the most favorable conditions of all those evaluated with the linear net and thus the results represent a best case scenario.

The flows are numbered 0-4 in order of increasing row number in the grid. Flows 0 and 4 have neighbor flows on one side only and intuitively the most favorable environment with respect to RF interference. In contrast, flow 2 has two flows on each side and a more challenging RF environment.

3.7.1 Grid throughput and latency

Throughput and loss for flows 0 and 2 are presented graphically in figures 2 through 5. Each figure contains four plots with each representing a value of $d_y \in \{80, 120, 160, 200, 240\}$.

As would be expected flow 0 throughput initially increases linearly with offered load and then flattens at decreasing bit rates with decreasing d_y. At $d_y = 240$ throughput increases linearly to 1400 Kbps. This replicates the behavior of the linear network with ARP pre-resolution indicating no crosstalk degradation.

The middle flow 2 obviously suffers more significant degradation. With $d_y = d_x = 80$ maximum throughput is 300 Kbps and for offered loads greater than 600 Kbps it falls to 100 Kbps.

3.7.2 Latency

For $d_x = 80$, minimal latency is sustainable on flow 2 only up to an offered load of 300 Kbps per flow. For all values of d_y other than 240, latency exceeds 1/2 second at offered loads of 900 Kbps and higher.

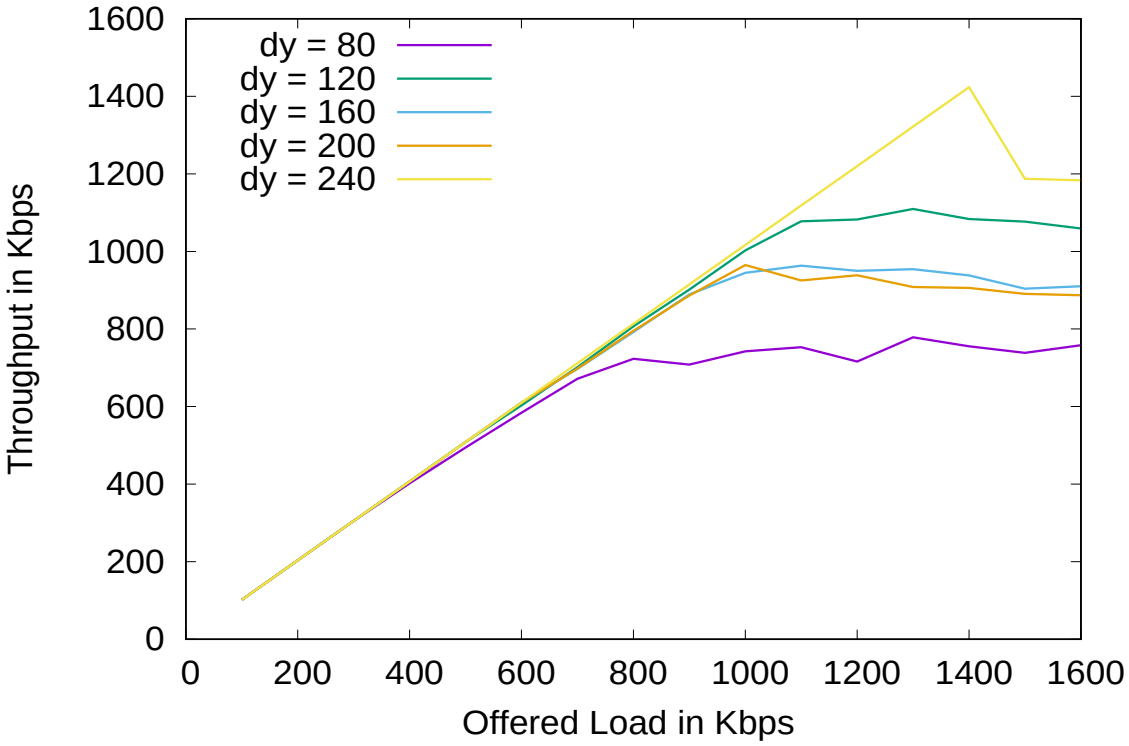

Figure 2: 7×5 grid flow 0 throughput with varying load and d_x.

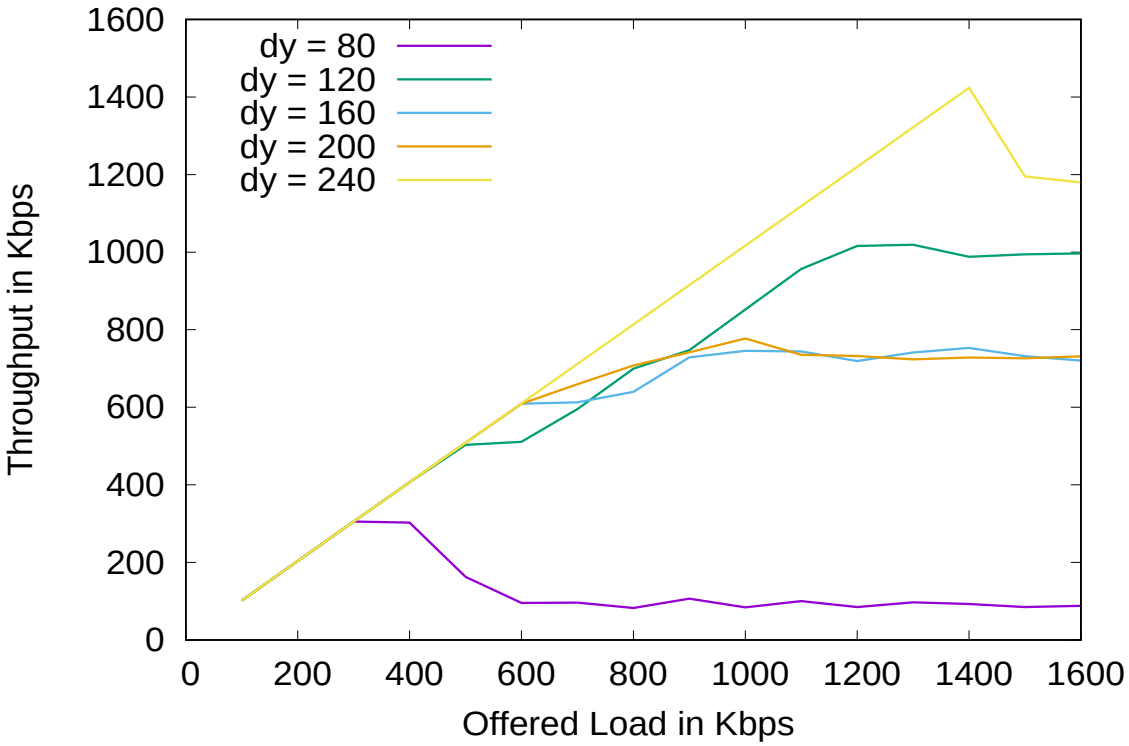

Figure 3: 7×5 grid flow 2 throughput with varying load d_x.

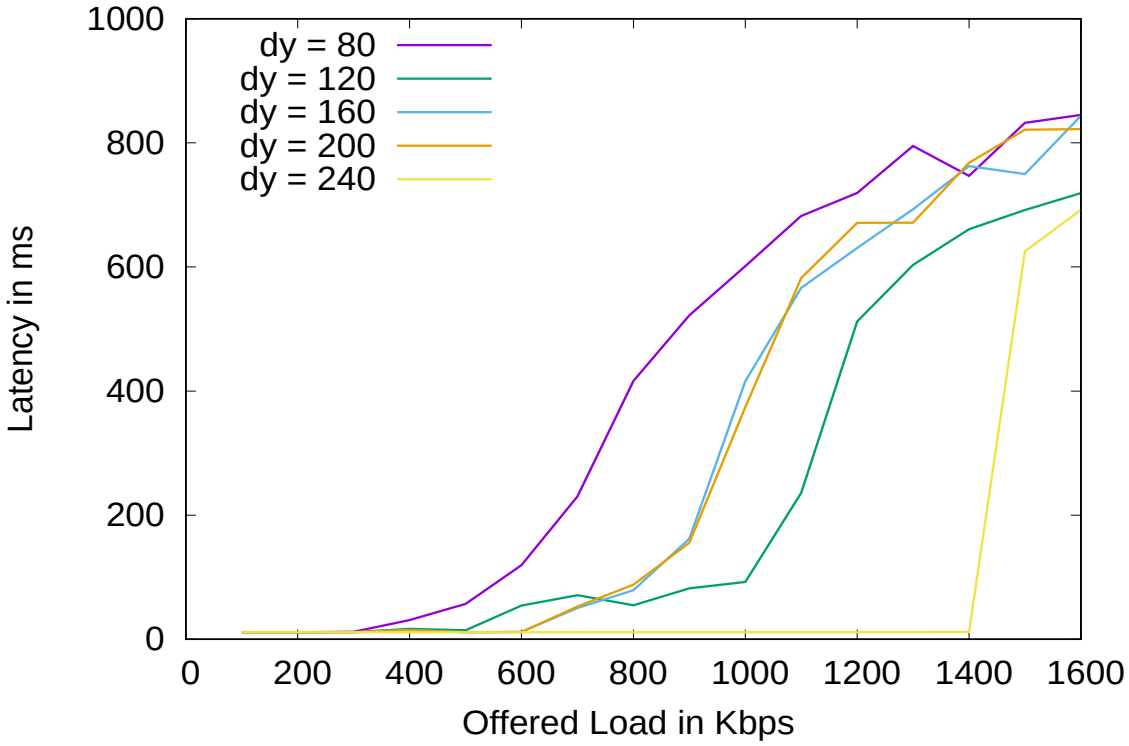

Figure 4: 7×5 grid flow 0 latency with varying load and d_x.

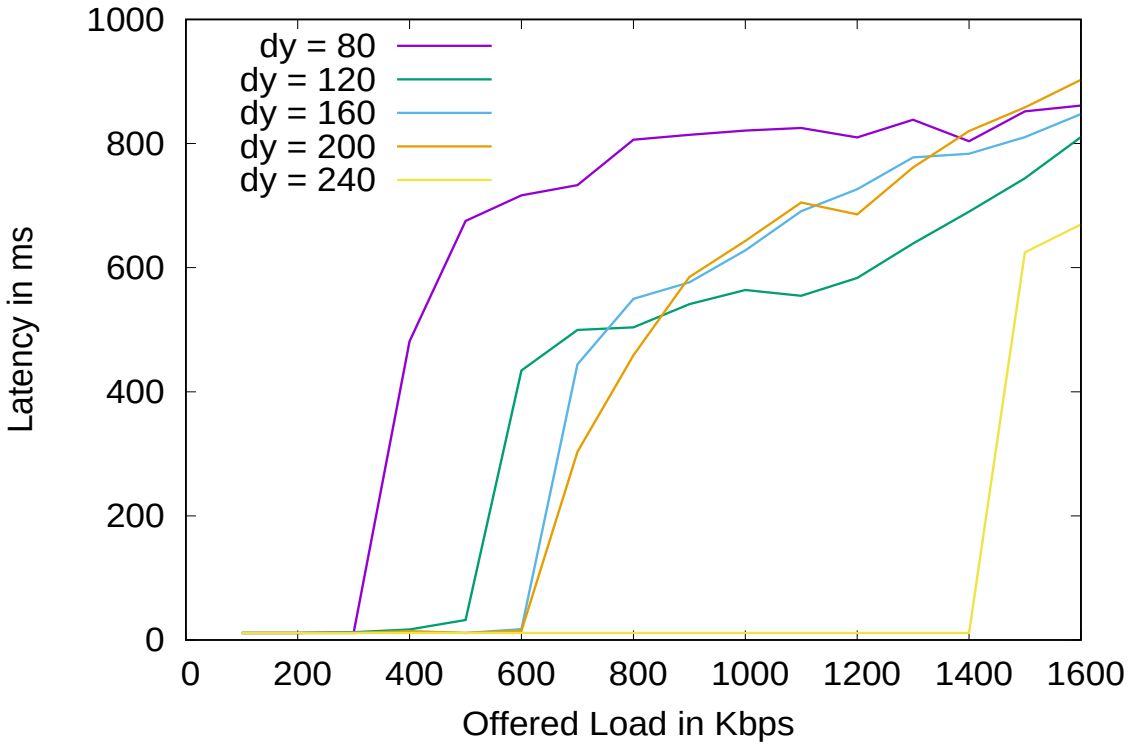

Figure 5: 7×5 grid flow 2 latency with varying load d_x.

For $d_y = 120$ both latency and throughput are anomalous in that in that at loads of 900 Kbps and higher higher throughput and lower latency are achieved with $d_y = 120$ than at 160 or 200.

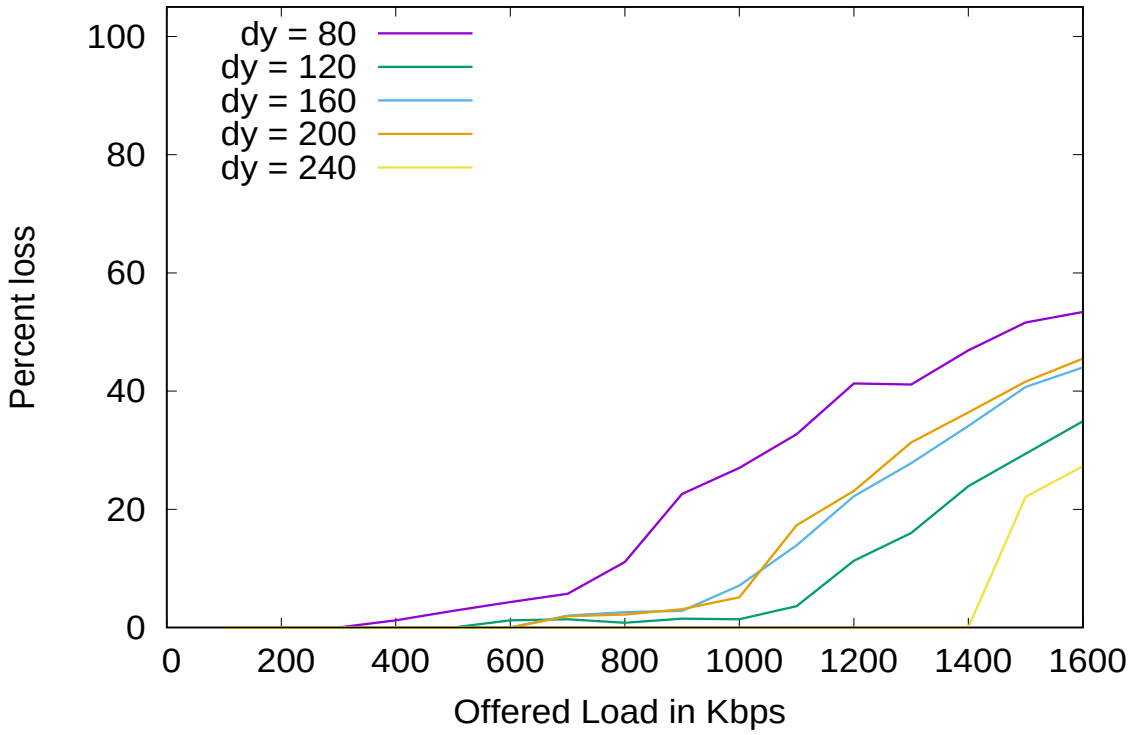

Figure 6: 7×5 grid flow 0 loss with varying load and d_x.

3.7.3 Grid loss behavior

Figures 6 and 7 the percentage of transmissions lost. As with static routes in the linear model there are no transmissions is which *SendTo* returns -1.

When $d_y = 240$ there are no losses in either flow 0 or 2 until the offered load reaches 1500 Kbps. For all other values of d_y losses commence at offered loads between 300 Kbps and 600 Kbps. For $d_y \in \{120, 160, 200\}$ loss rates remain below 10% for flow 0 and 20% for flow 2 while offered load is less than 1 Mbps. With $d_y = 80$ the network is unusable at speeds of 500 Kbps and higher.

The loss rates are again anomalous in the sense that loss at $d_y = 120$ is lower than at 160 and 200. Figure 8 demonstrates that $d_y = 120$ is a point of instability in the *ns3* propagtion model. Flows 0, 2, and 4 obtain good performance approximating or exceeding that of $dy = 160$ or 200 while flows 1 and 3 approximate the poor performance of $d_y = 80$.

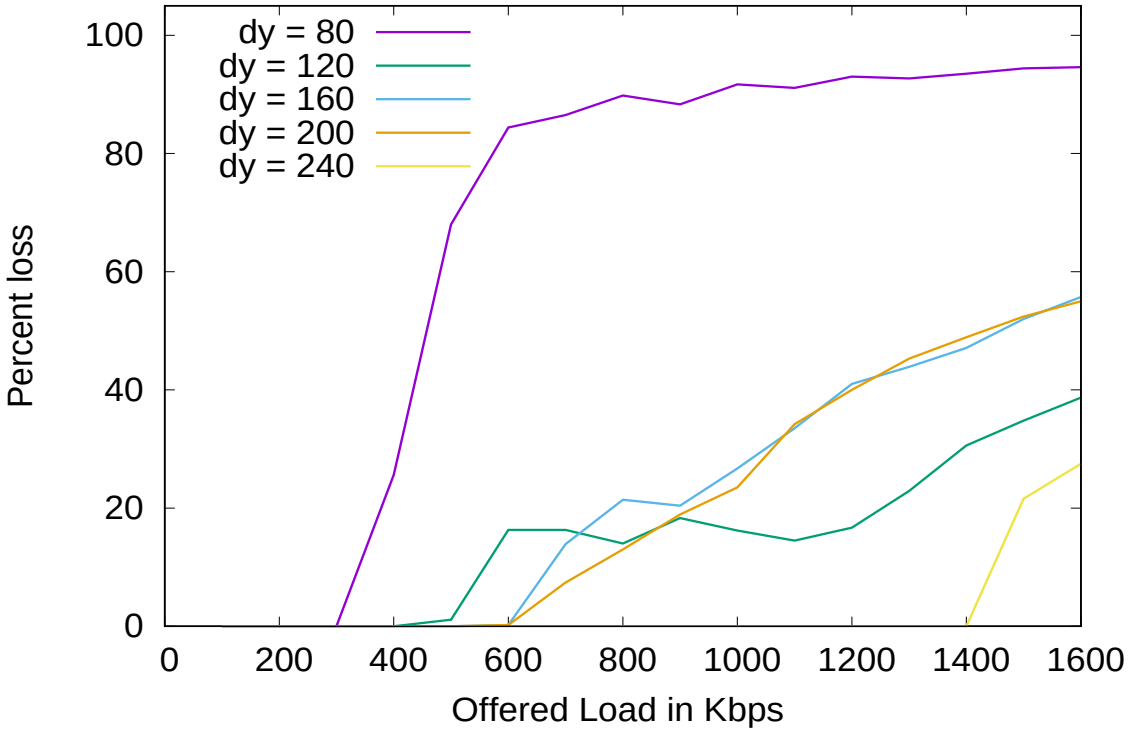

Figure 7: 7×5 grid flow 2 loss with varying load and d_x.

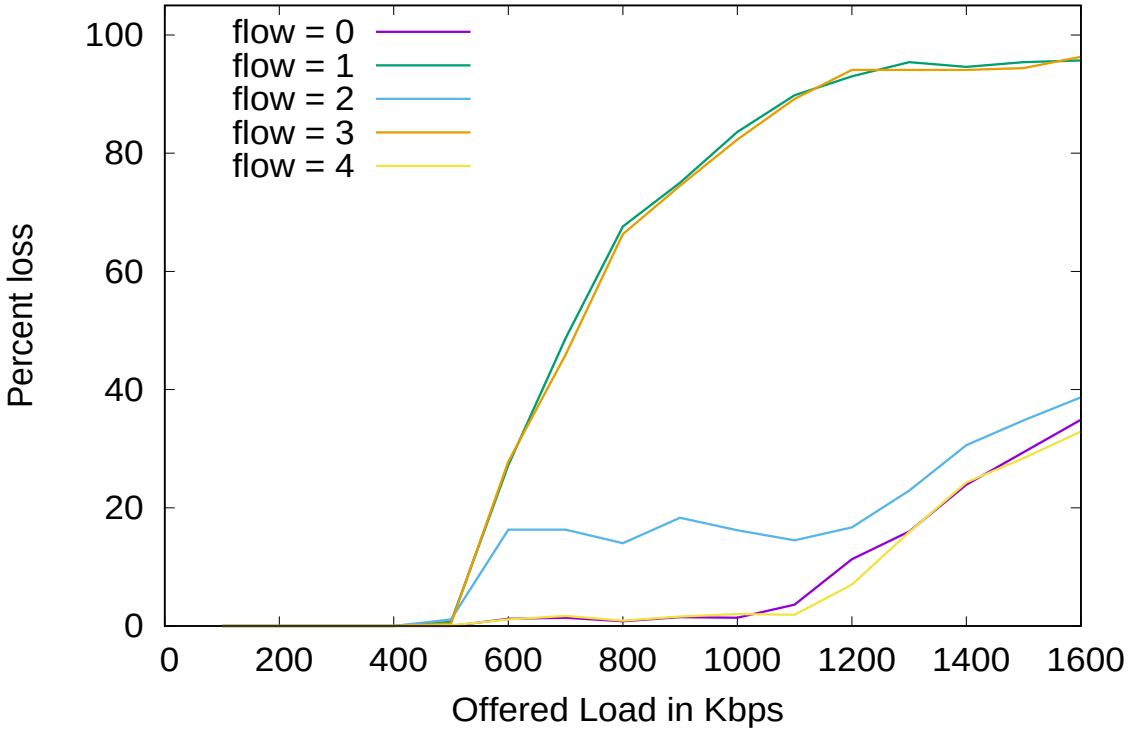

Figure 8: 7×5 grid loss for all flows with $d_x = 120$

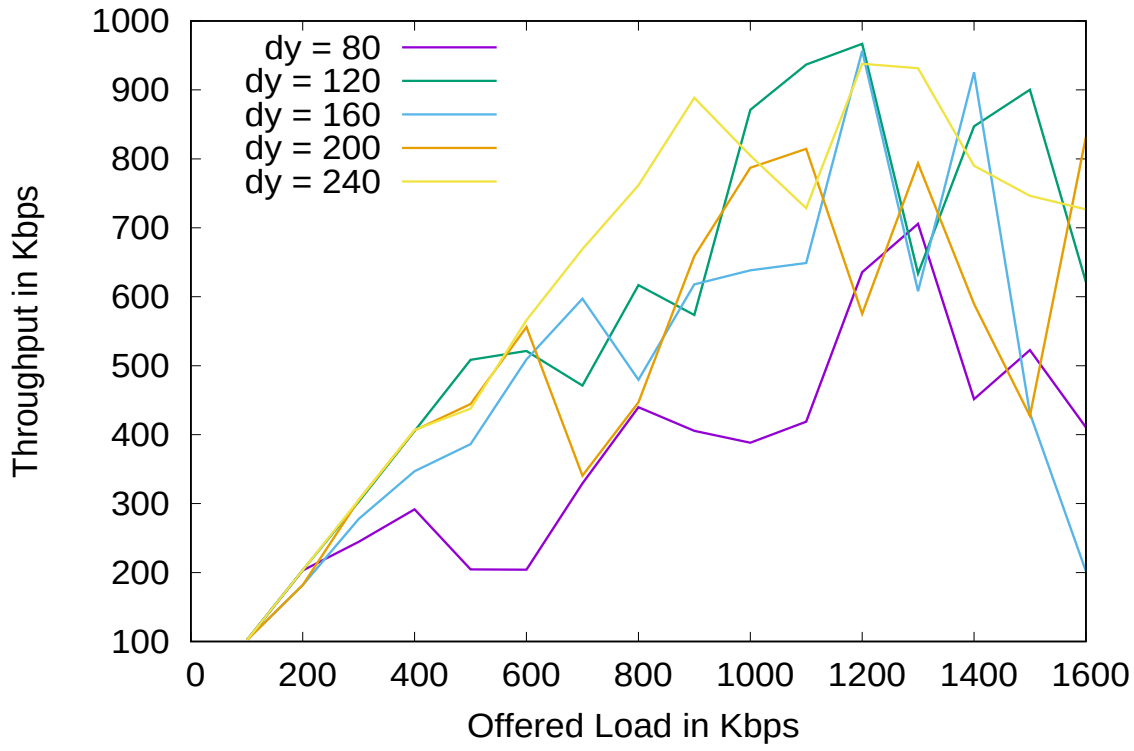

Figure 9: 7×5 OLSR flow 0 throughput with varying load and d_x.

3.8 Grid with OLSR routing

We conclude the analysis of the grid by presenting the results obtained with OLSR instead of static routing. ARP bindings are pre-resolved as before. Throughput and latency are shown in figures 9 through 12

The anomalous better than expected behavior at $d_y = 120$ persists with OLSR routing but overall throughput is significantly lower and far more erratic because of the periodic loss of connectivity in OLSR.

With OLSR routing, significant packet loss was observed at low loads, but the loss rate was strongly dependent on offered load. All of the low load losses are due to loss of routing capability either at the source or along the path. We do not yet understand the periodic-like correlation between offered load and loss of routing.

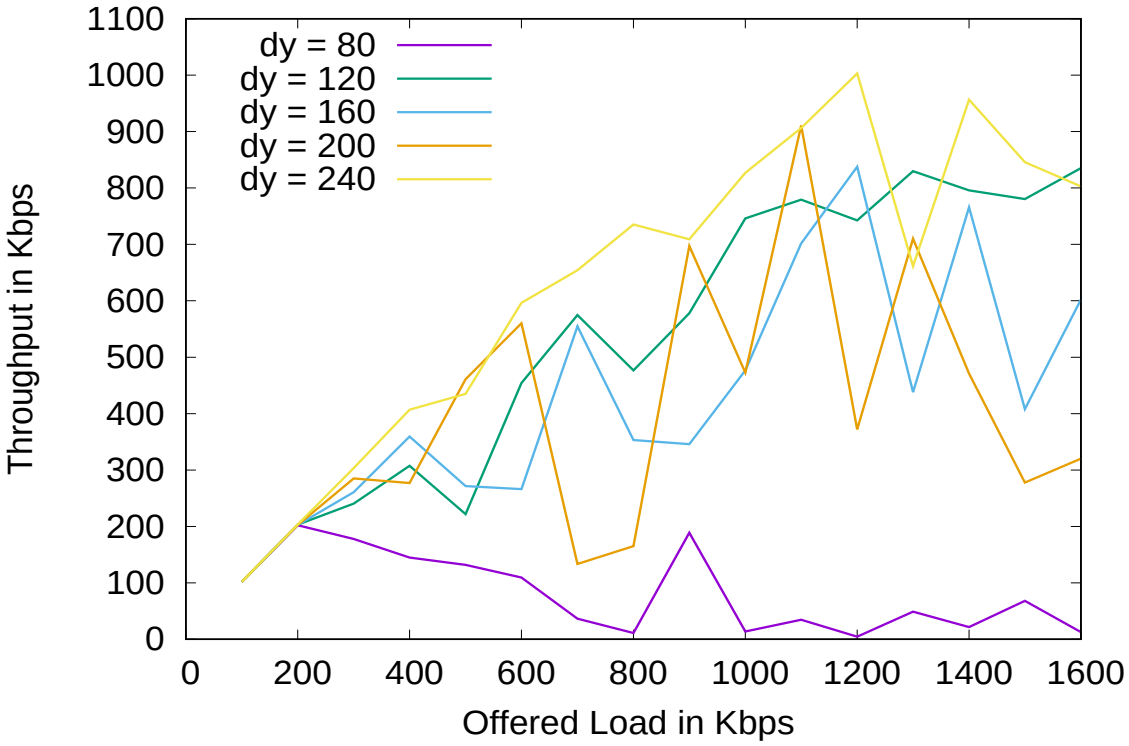

Figure 10: 7×5 OLSR flow 2 throughput with varying load d_x.

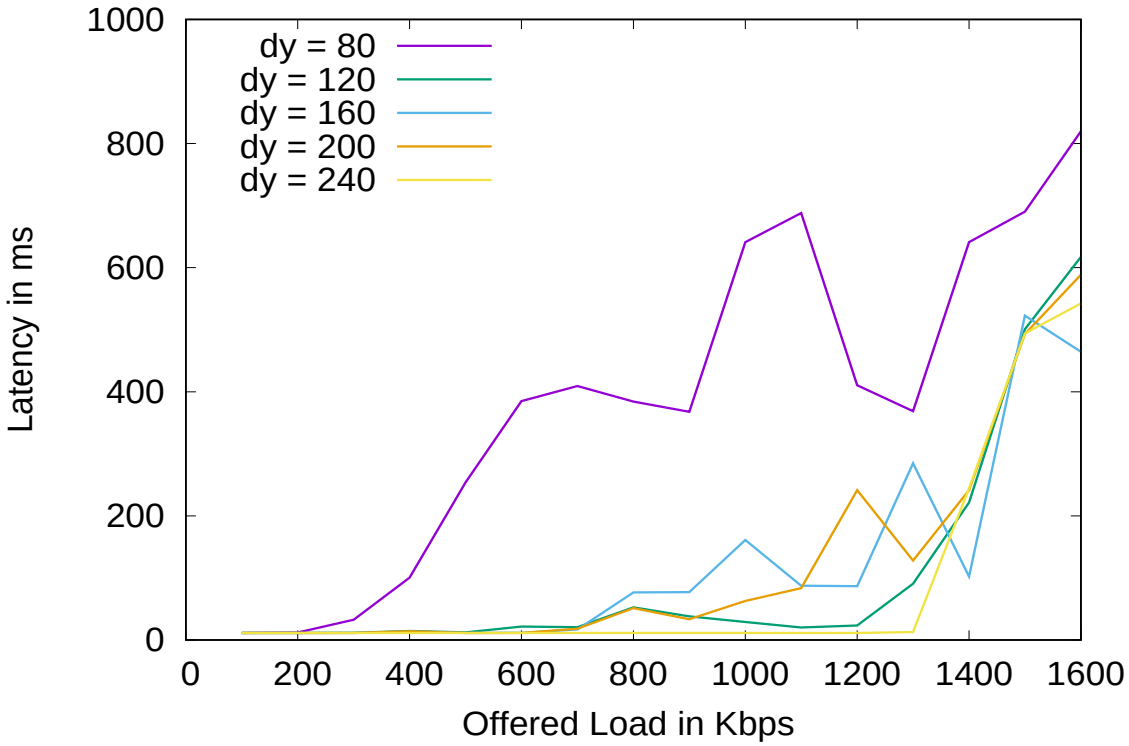

Figure 11: 7×5 OLSR flow 0 latency with varying load and d_x.

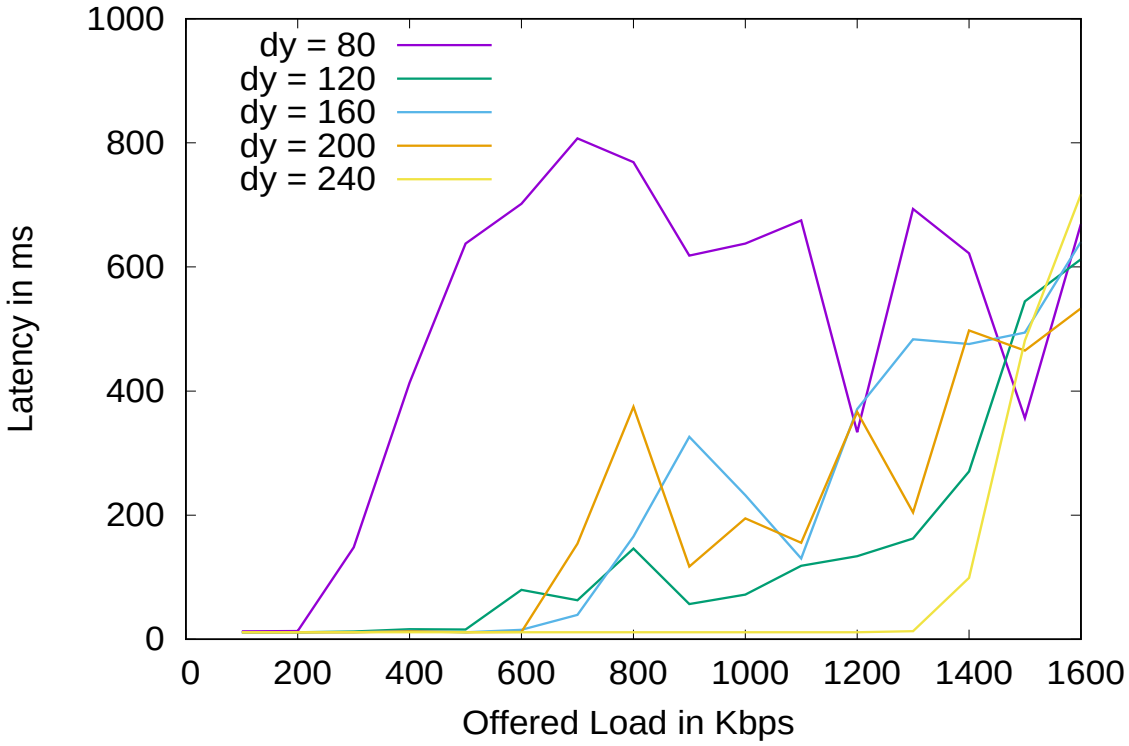

Figure 12: 7×5 OLSR flow 2 latency with varying load d_x.

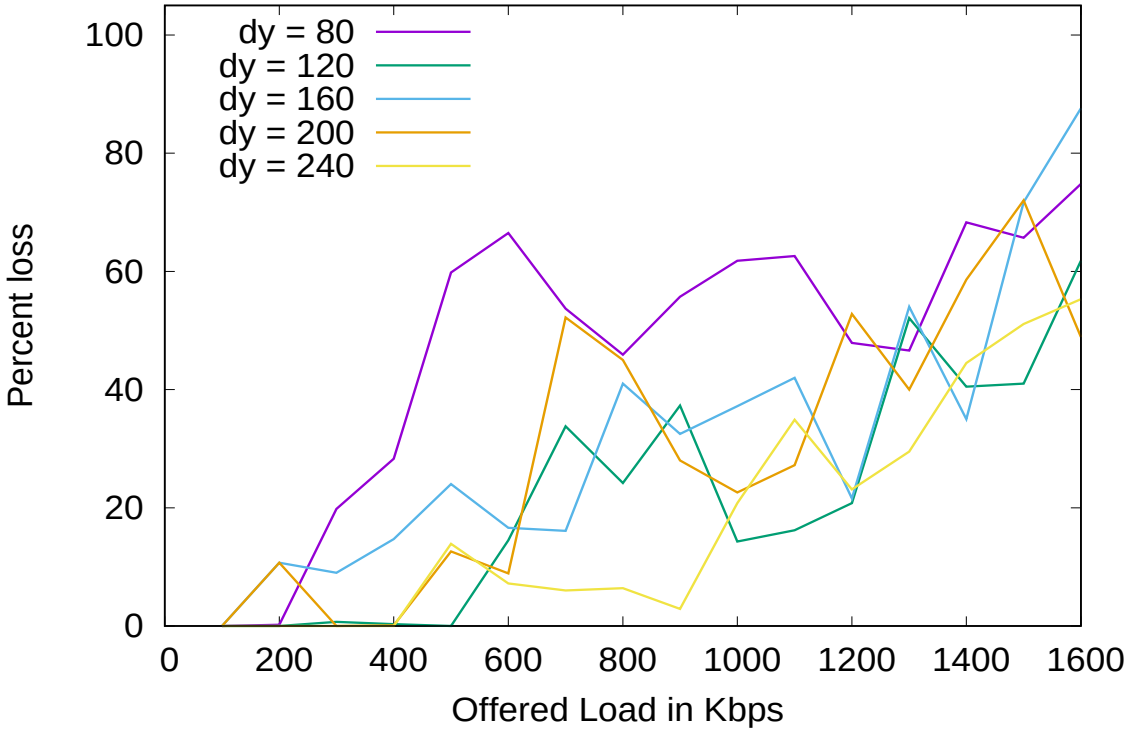

Figure 13: 7×5 OLSR flow 0 loss with varying load and d_x.

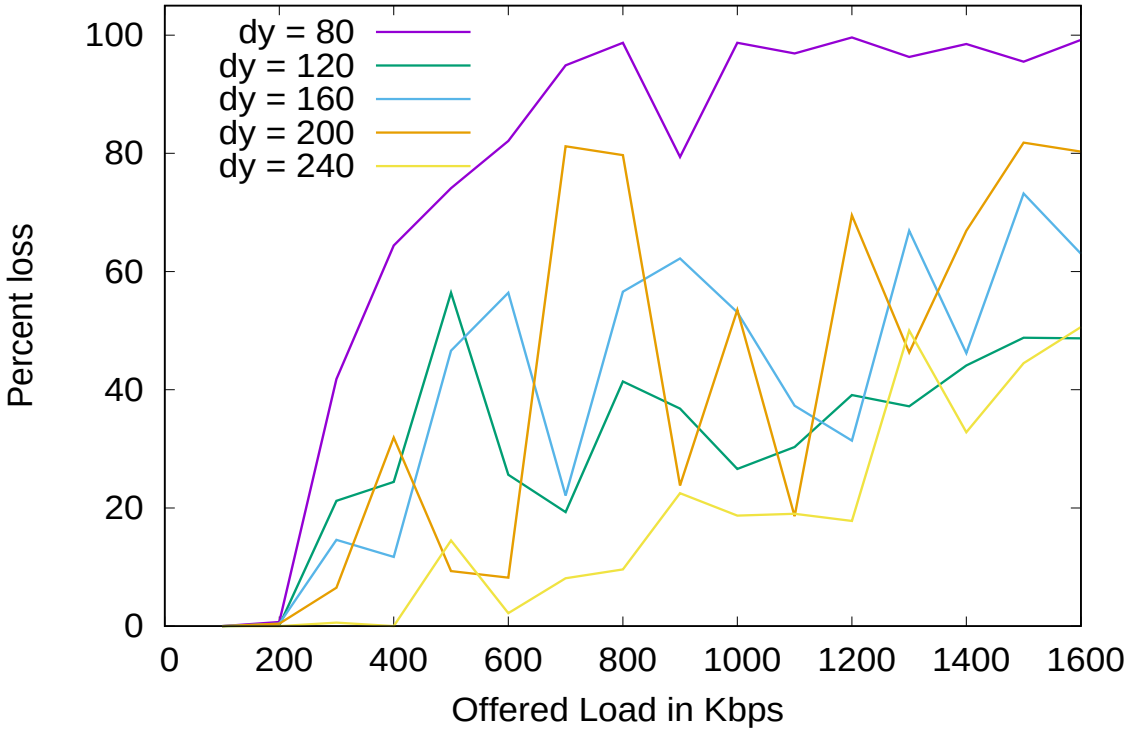

Figure 14: 7×5 OLSR flow 2 loss with varying load and d_x.

4 Conclusion

The results presented in the analysis of the single channel grid networks represent a best case scenario with static routing and ARP pre-resolution It is clear that a regular grid is a very poor choice for linear transport of high demand flows. Nevertheless, if $d_y > Md_x$ for relatively small (3 in this case) M the grid can operate as n_y independent linear flows.

However, it was also shown that the maximum throughput in a 6 hop linear network at 1.4 Mbps was only 23% of maximum point to point throughput of 6.1 Mbps. This calls into question the viability of grid based CSMA-CA networks for supporting (near) CBR linear traffic flows even if a multi-channel approach is employed.

A Appendices

A.1 802.11n PHY and MAC details

The 802.11n protocol employs Orthogonal Frequency Division (OFDM) modulation with standard unbonded channel bandwidth of 20 Mhz and The full channel is subdivided into 64 sub-channels, and 52 of these are used for data transport. The duration of a modulation state also known as the *symbol time* is 4 or $3.6\mu sec$ depending on whether or not the long intersymbol gap (800 ns) or the short gap (400 ns) is enabled.

In the section of this document the focus upon 20 Mhz bandwidth with 400 ns gap with a single spatial stream (no MIMO). Depending on signal quality up to 8 different combinations of modulation and coding rates a The nominal bit rate of the slowest (BPSK 1/2) is 7.2 Mbps while the fastest (QAM-64 5/6) is 72.2 Mbps. Characteristics of the 8 combinations are shown in table 22.

Table 22: 802.11n MCS data

MCS	Name	Code Rate	Coded Bits	Coded Bit Rate	Nom Bit Rate	Effective Bit Rate	Efficiency
0	BPSK 1/2	0.500	1	14.4	7.2	6.46	0.89
1	QPSK 1/2	0.500	2	28.9	14.4	11.97	0.83
2	QPSK 3/4	0.750	2	28.9	21.7	16.79	0.77
3	Q-16 1/2	0.500	4	57.8	28.9	20.95	0.73
4	Q-16 3/4	0.750	4	57.8	43.3	27.89	0.64
5	Q-64 2/3	0.667	6	86.7	57.8	33.50	0.58
6	Q-64 3/4	0.750	6	86.7	65.0	35.98	0.55
7	Q-64 5/6	0.833	6	86.7	72.2	38.20	0.53

Assume s = symbol time, k = number of data channels, b = coded bits per channel per symbol, and cr = coding rate. Then the nominal bit rate r in Mbps is given by:

$$r = k \times b \times cr/s.$$

For example, with $s = 3.6$, $k = 52$, $b = 4$, $cr = 3/4$, we find $r = 43.3$ Mbps.

A.2 Channel Access Delay

There are two timing parameters defined in the standard that underlie the interframe space and arbitration delays. The short interframe space $SIFS$ is the minimum amount of time that *must* elapse between the end of one transmission and the start of a subsequent transmission. The second is the arbitration slot time. Times for the 802.11n 23.

Table 23: Interframe Timing Data in μs

	SIFS	Arb slot
802.11n	16	9

A.3 Interframe space delay

The first element of the inter-packet delay is the time based short interframe space $SIFS$. It commences when the channel state goes from busy to idle. It must also be observed when a station becomes ready to send and senses that the channel is already idle. We first consider the simpler case in which the channel is already idle. In 802.11n arbitration, arbitration slot time is $9\mu s$. The short interframe space (SIFS) is 16 μs. For certain specific control frames (e.g. Ack, CTS), only the recipient of the previous transmission is eligible to send, and the Ack or CTS can be sent at the completion of the SIFS. In the absence of hidden stations, these frame types will never experience a collision.

A.4 Arbitration delay

In 802.11 networks that support polling, the PCF interframe space (PIFS) is equal to SIFS + 1 arbitration slot time). For all 802.11 networks the DCF interframe space (DIFS) is SIFS + 2 slots or (32 + 26 = 58 μs) for 802.11p. Polling requires a capable Access Point (AP) and is not supported in DSRC networks.

The Enhanced Distributed Channel Access ($EDCA$) standard supported by 802.11n identifies four arbitration classes commonly called background (BK), best effort (BE), video (VI) and voice (VO). The default AIFS values are 7, 3, 2, 2 slots respectively, but these may be reconfigured by the network administrator. Thus, the default AIFS for VO and VI access classes is the same as the DIFS.

Therefore, when a station with traffic of access class AC wishes to send and it senses the channel to be idle it must first wait for a delay = SIFS + AIFS[AC]. If the channel remains idle when this time has elapsed, the station executes the back off delay.

A.5 Backoff delay

The back off delay is designed to randomize transmission attempts among flows of the same or different arbitration classes. It is based upon binary exponential back off using back off windows whose sizes depend on access class as shown in table 24. A station maintains a current congestion window (CW) for each access class. The CW will always be between the CWMin and CWMax values shown in the table. At initialization and after a successful transmission all CW values are reset to the appropriate CWMin. Hence, the CW > CWMin condition occurs only after an unsuccessful transmission and the station will stay in the CW > CWMin state until the transmission succeeds or the packet is dropped.

When the interframe space delay has elapsed then the station will generate a random number, CWDelay, in the range [0, CW] and then wait CWDelay additional arbitration slots. After the CWDelay has elapse, if the channel remains idle the station will transmit.

Therefore, in the default configuration the mean total delay for the initial transmission of a frame

Table 24: Congestion window and arbitration spacing slots

	CWMin	CWMax	AIFSN
Background (BK)	15	1023	7
Best Effort (BE)	15	1023	3
Video (VI)	7	15	2
Voice (VO)	3	7	2

Table 25: Spacing, arbitration, and back off data

	Value
SIFS(us)	16
ArbSlot(us)	9
AIFS[AC] (slots)	2
AIFS delay	26
CWmin (slots)	7
E[Bkoff] (slots)	3.5
E[Bkoff] us	31.5
E[total delay]	65.5

in the VI class will be:

16 (SIFS) + 2 × 9 + (AIFS) + 3.5 × 9 (Backoff Delay) = 65.5 μs.

A.6 Framing Overhead

Each PHY level frame commences with a preamble that is transmitted at BPSK 1/2 encoding. There are several optional preambles whose use depends in requirements for interoperation with non 802.11n nodes. We assume a 32.6 μs preamble in our model. The premable is followed by a 16 bit field called the service and 6 bit field called the tail is appended to the MPDU that is encapsulated.

Each MPDU commences with a 3-address field 24 byte IEEE 802.11 MAC Header and followed by a 4 byte CRC. Consequently, a 1500 byte NPDU creates a 1528 byte MPDU of 12224 bits. This the addition of service and tail produces a 12246 bit PPDU that is transmitted at the active MCS encoding.

A.7 Computing throughput efficiency

Throughput efficiency at the application layer is the ratio of the time spent sending the bits of the APDU to the total time spent sending the frame that contained the APDU. We use a 1472 byte APDU sent using the UDP-IP protocols in the VI access class over a QAM-16 3/4 channel as an example. We assume only an single sender is active and so the channel is always sensed idle and each transmission succeeds on the first try.

As described previously, the physical layer PDU (PPDU) consists 12446 bits. It is also obviously necessary to send an integral number of complete symbols. So some padding may be required to fill out the last symbol.

For QAM-16 3/4, the number of coded bits per symbol is $52 \times 4 = 204$. The number of coded bits required for 12246 data bits at a coding rate of 3/4 is $\lceil 4/3 \times 12246 \rceil = 16238$. Consequently the number of symbols required is $\lceil 16328/204 \rceil = 79$.

Therefore, the time to send the PPDU is:

$$3.6\mu s \times 79 = 284.4\mu s$$

In the previous sections we have seen that the preamble requires 32.6 μs, and that the arbitration overhead including both AIFS(VI) and mean back off delay is 65.50 μs.

Hence the total time to send the frame is $284.4 + 32.6 + 65.5 = 382.5$ μs. Consequently the effciency is 284.4 / 382.5 = 74.4. The effective bit rate is then computed by multiplying the nominal bit rate by the efficiency.

A.8 Unicast message acknowledgments

The previous discussion is a proper analysis of broadcast packets. However, MAC layer Acks are used in unicast messages in 802.11n and further reduce the efficiency.

A MAC layer ack is sent as a 16 byte Mac Control Frame embedded in a standard 28 byte MAC header. Consequently the Ack is 352 bits. When the service and tail are included the resulting PPDU is 374 bits or 3 symbols at QAM-16 3/4. The transmission time is $3 \times 3.6\mu s = 10.8\mu s$.

Addtional overhead is limited to the preamble time (32.6) and the SIFS 16 μs yielding a total overhead of 48.6 μs. The total time to send the ack frame is thus 10.8 + 48.6 = 59.2 μs. The total time for frame plus ack is thus 382.5 + 59.2 = 441.9, and the effciency at the physical layer is reduced to 284.4 / 441.9 = 0.644. To compute efficiency at application layer it suffices to compute $8 \times 1472/12246 \times 0.664 = 0.62$.

Therefore, expected throughput at nominal rate $43.\bar{3}$ is 27.86 at the PHY layer and 26.80 at the APP layer. The preceding analysis is summarized in table 26.

A.9 APDU length effects

For a given modulation technique all of the elements below the APDU time are independent of the APDU size. Thus making the APDU small can dramatically reduce efficiency.

Table 26: QAM-16 3/4 protocol efficiency

PPDU (us)	284.4
Framing (us)	32.6
Mean arb (us)	65.5
E[PPDU + framing + arb (us)]	382.5
Bcast efficiency	0.744
Ack PDU (us)	10.8
Ack framing (us)	32.6
Ack arb (SIFS) us	16.0
Ack total (us)	59.4
2-way total (us)	441.9
PPDU efficiency	0.644
APDU size (bits)	11776
PPDU size (bits)	12246
APDU :: PPDU size ratio	0.962
APDU efficiency	0.619
Nominal Rate	43.333
E[Max PPDU bit rate] (Mbps)	27.888
E[Max APDU bit rate] (Mbps)	26.818

A.10 Modulation and coding rate effects

Keeping the APDU size at 1472 but changing the modulation or coding rate also alters efficiency. Because the preamble and arbitration overhead are fixed in the time domain, any change that increases the nominal bit rate will reduce efficiency and any change that reduces the nominal bit rate will increase efficiency.