Accuracy Enhancements of the 802.11 Model and EDCA QoS Extensions in ns-3

Timo Bingmann

Decentralized Systems and Network Services Research Group
Institute of Telematics, University of Karlsruhe

February 6, 2009
Roadmap

1. ns-3 Basics
   - Introduction
   - Showcase: Design Patterns
   - Current State

2. Wifi in ns-3
   - State of 802.11
   - PHY Layer
   - Signals, Noise and Interference
   - Short Recapitulation of DCF
   - QoS with EDCA

3. Conclusion
ns-3 Introduction

ns-3 is

- a discrete-event network simulator.
- intended to replace ns-2.
- not backwards compatible to ns-2.
ns-3 Introduction

ns-3 Goals

- Create tools aligned with needs of modern networking research.
- Work as open-source project with active community participation.
- Improve repeatability of results in research papers.
ns-3 and ns-2

ns-3 is not based on ns-2: drop ns-2’s historic burdens.

- ns-3 is fully C++.
- Leverage up-to-date features of C++.
- Create optional language bindings like Python for interpreter frontends.
Design Patterns

Utilize modern design patterns in C++:

- Object and attribute system.
- Smart Ptr<> automatic memory management.
- Callbacks to decouple modules.
- COM-like object aggregation and interface querying.
- Decouple trace sources from sinks.

Requires advanced C++ knowledge.
Design Pattern: Tracing

Tracing needs vary greatly in different simulations.

ns-2:

- Trace objects inserted as network elements.
- Fixed trace file format for further statistical processing.
- Not easily customizable to own experiment.
- Also available: queue monitors.
Design Pattern: Tracing

- Models export `TraceSources`. Examples: Node packet reception, 802.11 PHY state changes, TCP congestion window values.
- `TraceSources` can be connected to own callback functions
- or to predefined trace files generators for output in pcap/tcpdump format or ascii text.
### 1 ns-3 Basics

#### 1.3 Current State

<table>
<thead>
<tr>
<th>Existing core ns-2 models</th>
<th>Existing ns-3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Applications</strong></td>
<td>OnOffApplication, asynchronous socket API, packet sockets</td>
</tr>
<tr>
<td>ping, vat, telnet, FTP, HTTP, probabilistic and trace-driven traffic generators, webcache</td>
<td><strong>Transport layer</strong></td>
</tr>
<tr>
<td><strong>Network layer</strong></td>
<td>TCP (many variants), UDP, SCTP, XCP, TFRC, RAP</td>
</tr>
<tr>
<td>Unicast: IP, MobileIP, generic distance vector and link state, IPinIP, source routing</td>
<td>Multicast: PGM, SRM, RLM</td>
</tr>
<tr>
<td>MANET: AODV, DSR, DSDV, TORA, IMEP</td>
<td><strong>Link layer</strong></td>
</tr>
<tr>
<td><strong>Link layer</strong></td>
<td>ARP, HDLC, GAF, MPLS, LDP, Diffserv</td>
</tr>
<tr>
<td>MACs: CSMA, 802.11b, 802.15.4 (WPAN), satellite Aloha.</td>
<td>PointToPoint, CSMA, 802.11 MAC low, high and rate control algorithms</td>
</tr>
<tr>
<td><strong>Physical layer</strong></td>
<td>TwoWayGround, Shadowing, OmniAntennas, EnergyModel, Satellite Repeater</td>
</tr>
<tr>
<td>TwoWayGround, Shadowing, OmniAntennas, EnergyModel, Satellite Repeater</td>
<td><strong>Core Support</strong></td>
</tr>
<tr>
<td><strong>Core Support</strong></td>
<td>RNGs, tracing monitors, mathematical support, test suite, animation (nam)</td>
</tr>
<tr>
<td>RNGs, unit tests, logging, callbacks, mobility visualizer</td>
<td><strong>Existing ns-3</strong></td>
</tr>
<tr>
<td></td>
<td>UDP, TCP</td>
</tr>
<tr>
<td></td>
<td>Unicast: IPv4, global static routing</td>
</tr>
<tr>
<td></td>
<td>Multicast: static routing</td>
</tr>
<tr>
<td></td>
<td>MANET: OLSR</td>
</tr>
<tr>
<td></td>
<td><strong>Physical layer</strong></td>
</tr>
<tr>
<td></td>
<td>802.11a, Friis propagation loss, log distance loss, basic wired (loss, delay)</td>
</tr>
<tr>
<td></td>
<td><strong>Core Support</strong></td>
</tr>
<tr>
<td></td>
<td>RNGs, unit tests, logging, callbacks, mobility visualizer</td>
</tr>
</tbody>
</table>

Based on tables from [1] and [2].
# SLOC of ns-2.33 and ns-3.3

<table>
<thead>
<tr>
<th></th>
<th>ns-2.33</th>
<th></th>
<th>ns-3.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>C/C++</td>
<td>162,208</td>
<td>58%</td>
<td>77,270</td>
</tr>
<tr>
<td>Tcl</td>
<td>103,419</td>
<td>37%</td>
<td>Python¹</td>
</tr>
<tr>
<td>Other</td>
<td>13,341</td>
<td>5%</td>
<td>Other</td>
</tr>
<tr>
<td>Total</td>
<td>278,968</td>
<td></td>
<td>Total</td>
</tr>
</tbody>
</table>

¹ excludes automatically generated code
Statistics generated using David A. Wheeler’s 'SLOCCount'.
### SLOC of ns-2.33 and ns-3.3

<table>
<thead>
<tr>
<th></th>
<th>ns-2.33</th>
<th></th>
<th>ns-3.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>C/C++</td>
<td>162,208</td>
<td>58%</td>
<td>77,270</td>
</tr>
<tr>
<td>Tcl</td>
<td>103,419</td>
<td>37%</td>
<td>Python(^1)</td>
</tr>
<tr>
<td>Other</td>
<td>13,341</td>
<td>5%</td>
<td>Other</td>
</tr>
<tr>
<td>Total</td>
<td>278,968</td>
<td></td>
<td>Total</td>
</tr>
<tr>
<td>802.11</td>
<td>6,067</td>
<td>2%</td>
<td>802.11</td>
</tr>
</tbody>
</table>

\(^1\) excludes automatically generated code

Statistics generated using David A. Wheeler’s 'SLOCCount'.
UML of ns-2’s Wifi Classes

Mac802_11

TXC
+handleMsgFromUp(p:Packet*): void

Mac802_11Ext

RXC
+handleMsgFromBelow(p:Packet*): void

BackoffTimer_t

BackoffMgr

ChannelStateMgr

IFSTimer

NAVTimer

WirelessPhyExt
+sendDown(p:Packet*): void
+sendUp(p:Packet*): void

WirelessPhy

interf
+Pt: double
+end: double

PowerMonitor
+recordPowerLevel(power:double, duration:double): void
+getPowerLevel(): double

Propagation

Mac802_11Ext::MAC_MIBExt

Mac802_11Ext::PHY_MIBExt

WirelessChannel
UML of ns-3’s Wifi Classes
Thesis Goals

Goals

- ns-3 wireless simulations give equal or accountably different results like equivalent ns-2 simulations.
- Extend ns-3 with EDCA for 802.11e QoS.
Modelling 802.11 in ns-3

MAC
- DcaTxop
  - DcfManager
  - StationManager
  - Queue
  - MacHigh
  - MacLow
  - MacRxMiddle

PHY
- WifiPhy
  - ErrorRateModel
  - InterferenceHelper

Medium
- WifiChannel
  - PropagationLossModel
State of 802.11 in ns-3

PHY layer:

- Currently only 802.11a rates supported.
- No simulation of capture effect.
- No Nakagami propagation loss model.
+ BER/PER reception criterion.
PHY Layer

Goal: compatibility with ns-2 WirelessPhyExt.

Required components

- PowerMonitor for cumulative noise
- SINR reception criterion
- Capture effect
- Nakagami propagation loss model
Signals, Noise and Interference

Time

Power

Time

Power
Signals, Noise and Interference
Signals, Noise and Interference

![Power vs Time Diagrams]

802.11 Enhancements in ns-3

Timo Bingmann - 19/44
University of Karlsruhe
Signals, Noise and Interference

\[
\text{SINR} = \frac{\text{Signal}}{\text{Noise} + \text{Interference}}
\]

Power (dBm)

-100
-90
-80

SINR (dB)

20
10
0
-10

Time

University of Karlsruhe

802.11 Enhancements in ns-3

Timo Bingmann - 20/44
SINR Threshold

Time

SINR (dB)

20
10
0

BPSK 5dB QAM-16 15dB
QPSK 8dB QAM-64 25dB
SINR Threshold

- **BPSK**: 5dB
- **QPSK**: 8dB
- **QAM-16**: 15dB
- **QAM-64**: 25dB

The diagram shows the SINR (Signal-to-Interference-plus-Noise Ratio) threshold for different modulation schemes. The thresholds are as follows:

- **BPSK** at 5dB
- **QPSK** at 8dB
- **QAM-16** at 15dB
- **QAM-64** at 25dB

The graph illustrates the SINR levels over time, with segments labeled as **H** for header and **PAYLOAD** for payload data.
BER/PER Criterion

\[
\text{BER}_{\text{BPSK}} \left( \frac{E_b}{N_0} \right) = Q \left( \sqrt{\frac{2E_b}{N_0}} \right)
\]
Capture Effect

The diagram illustrates the capture effect in Wi-Fi signals, where the power levels of two signals are shown over time. The green signal has a power level of -90 dBm, and the blue signal has a power level of -80 dBm. The combination of these signals results in a superposition in the combined signal, as shown in the lower graph. The combined signal power level is close to the blue signal power level, indicating the capture effect.
Capture Effect

Power (dBm)

-100
-90
-80

SINR (dB)

-10
0
10
20

Time

802.11 Enhancements in ns-3

Timo Bingmann - 24/44
University of Karlsruhe
Without Capture Effect

![Diagram showing SINR (dB) over time with different states for RX and BUSY]

- Receiver states: IDLE, RX, BUSY, IDLE
- SINR (dB) over time with RxError
With Capture Effect

![Graph showing SINR (dB) over time with different states for RX and IDLE, and RxError and RxOk events.]}
Thesis Goals

Goals

- ns-3 wireless simulations give equal or accountably different results like equivalent ns-2 simulations.
- Extend ns-3 with EDCA for 802.11e QoS.
Modelling 802.11 in ns-3

- MAC
  - DcaTxop
  - Queue
  - DcfManager
  - StationManager
  - MacHigh
  - MacLow
  - MacRxMiddle

- PHY
  - WifiPhy
  - InterferenceHelper
  - ErrorRateModel

- Medium
  - WifiChannel
  - PropagationLossModel
Short Recapitulation of DCF

Radio transmission using CSMA/CA: Carrier sense multiple access with collision avoidance

802.11 has two carrier sense mechanisms:

- physical - CCA_BUSY
- virtual - NAV (network allocation vector)
Physical Carrier Sense

Stations always listen to the radio channel.

CCA_BUSY indication is raised if radio energy level is above a CS threshold.
Virtual Carrier Sense

Stations hear and decode **all packet headers** on the radio channel.

Header contains a duration field. Reserves channel for time after packet by updating NAV.
RTS/CTS using NAV
RTS/CTS using NAV

A

RTS

SIFS

DATA

B

SIFS

CTS

SIFS

ACK

Other

NAV (RTS)

NAV (CTS)

?
**IFS - Interframe Spaces**

- **SIFS**: Short IFS for direct answers to frame sequences.
- **PIFS**: PCF IFS for Point Coordination Function
- **DIFS**: DCF IFS for Distributed Coordination Function
- **EIFS**: Extended IFS for error backoff.
- **AIFS[i]**: Arbitration IFS for QoS.

**BUSY**
Backoff Procedure

DIFS

backoff=9

backoff=5

backoff=8
Backoff Procedure

- **Backoff=8**
- **Backoff=5**
- **Backoff=9**

Each backoff represents a random delay before a node attempts to transmit a packet. The diagram shows the progression of backoff values with DIFS intervals in between each attempt.
Backoff Procedure

- **Backoff** = 9
- **Backoff** = 5
- **Backoff** = 8
Backoff Procedure

DIFS

backoff=9

DIFS

backoff=4

DIFS

backoff=8

DIFS

backoff=5
Backoff Procedure

- **DIFS**

- **backoff = 9**

- **backoff = 4**

- **backoff = 8**
Contention Window

- **DIFS**
- **aSlotTime** = 9µs (802.11a)

Initial attempt: [0..15]

1st retransmission: [0..31]

2nd retransmission: [0..63]

Backoff is uniform random integer from [0...CW].
Problems of DCF for QoS

DCF is not good for time-critical traffic:
- Any STA may transmit arbitrarily large frames.
- All traffic stored in one queue.

PCF does not handles these issues:
- Contention-free period may be delayed.
EDCA Access Categories

AC_VO

AC_VI

AC_BE

AC_BK

AIFS[VO] CW[VO]

AIFS[VI] CW[VI]

AIFS[BE] CW[BE]

AIFS[BK] CW[BK]

Internal Collision Resolution

Transmit
## Default EDCA Parameters

### 802.11p (Draft 4.02)

<table>
<thead>
<tr>
<th></th>
<th>VO</th>
<th>VI</th>
<th>BE</th>
<th>BK</th>
<th>DFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWmin</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>CWmax</td>
<td>7</td>
<td>7</td>
<td>15</td>
<td>1023</td>
<td>1023</td>
</tr>
<tr>
<td>AIFSN</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>AIFS</td>
<td>34µs</td>
<td>43µs</td>
<td>70µs</td>
<td>97µs</td>
<td>34µs</td>
</tr>
</tbody>
</table>
DCF Backoff Probability

- VO
- VI
- BE
- BK
- DCF

DCF Backoff Probability Diagram

- SIFS
- PIFS
- DIFS
Default EDCA Parameters of 802.11p
Work Status

Already finished:

- Ported NakagamiPropagationLossModel including dependencies.
- Implemented Ns2ExtWifiPhy for SINR reception and capture effect.
Outlook

Further Plans:

- Backport capture to BER/PER model.
- Implement and verify 802.11e EDCA QoS.
- Compilation and speed improvements with icc.
- Theoretical discussion of parallel or distributed 802.11 simulation.
End

Thank you for your attention.
Bibliography

ns-3 project goals.
In WNS2 ’06: Proceeding from the 2006 Workshop on
ns-2: the IP network simulator, page 13, New York, NY,
USA, 2006. ACM.

ns-3 overview, December 2008.