

Programming Assignment #8

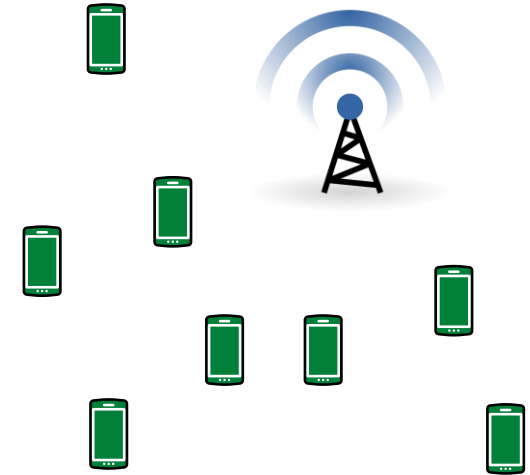
MAC Performance Analysis: Slotted ALOHA

Mu He

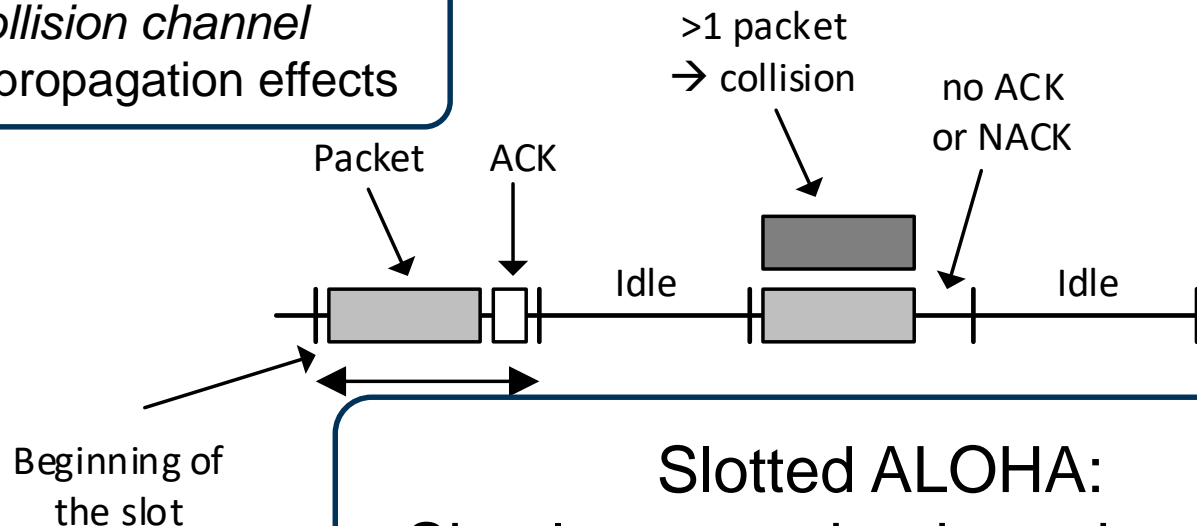
Mu.he@tum.de

Slotted ALOHA

- **Possible scenario:**
multiple mobile stations (MSs), uplink to BS
- MSs are synchronized to the **slot**
(e.g., via SYNC broadcast from BS)
- All transmissions occur within *a slot*
- **Three states:**
singleton (1 MS), idle (0 MSs), collision (>1 MS)



collision channel
→ no propagation effects



Slotted ALOHA:
Simple contention-based protocol

Why do we use ALOHA-based MAC?

Contention-based

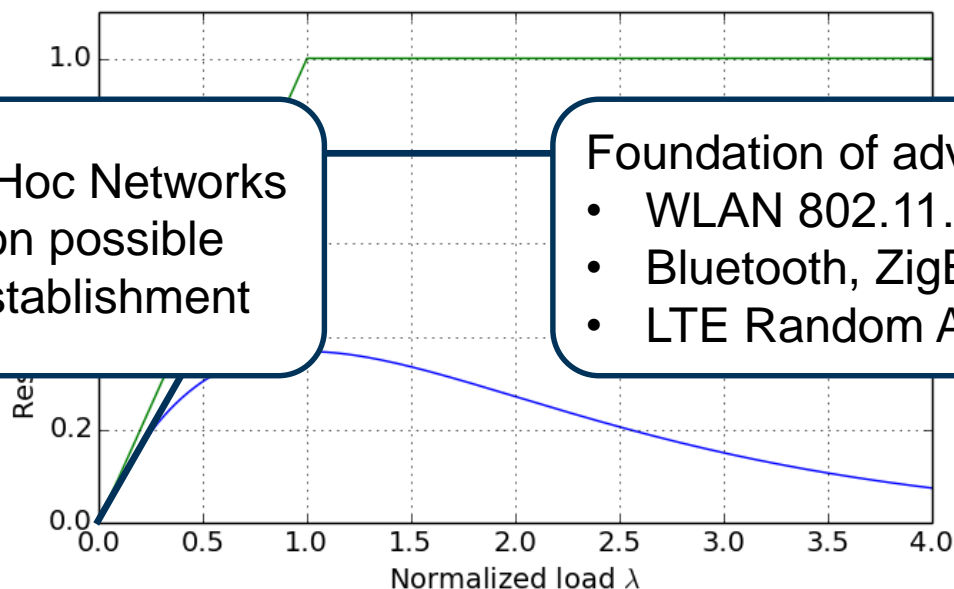
- + Low overhead
→ **No Information Exchange!**
- + Low complexity
- + Lower delay (if under-loaded)
- Low resource utilization: overprovisioning
- Non-linear degradation of the network performance (if over-loaded)

Contention-free

- + Higher resource utilization (high load)
- + Linear scaling of parameters with the load
- High overhead
→ **Information Exchange!**
- High delay (if under-loaded)

- Low load, Ad-Hoc Networks
- No coordination possible
- Connection establishment

- Foundation of advanced protocols:
- WLAN 802.11...
 - Bluetooth, ZigBee, 6LoWPAN,
 - LTE Random Access Procedure



Basic analytical model

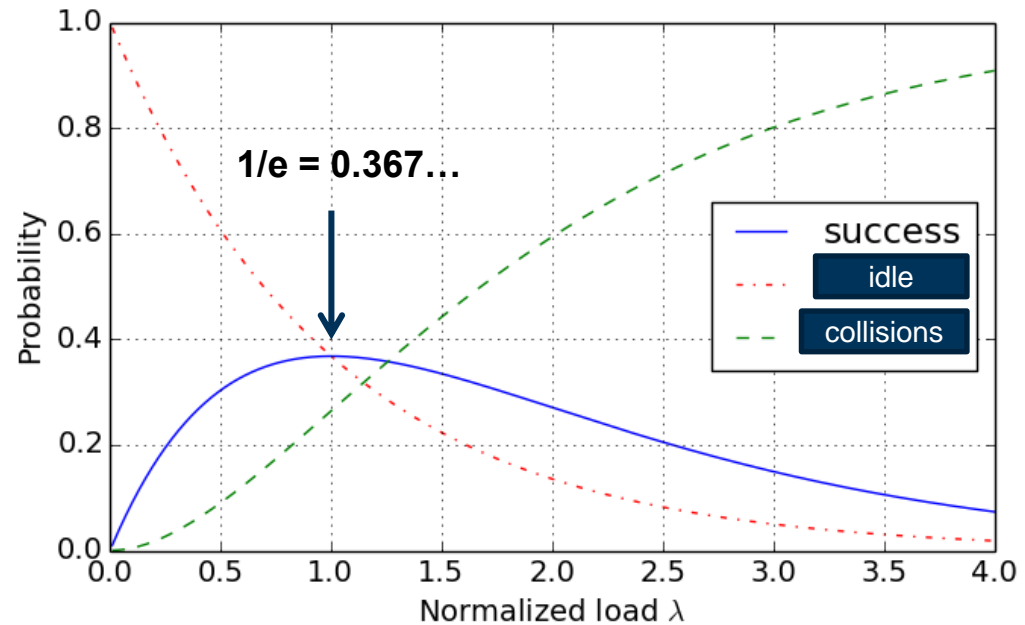
- **Poisson** arrivals with rate λ (**load per slot**)
- **Three states:** Success (1), Idle (0), Collisions (e)
- **Throughput T** – ratio of successful receptions (1) to the total number of slots (1+0+e)
= resource utilization

Probability of exactly one transmission in a slot is (success probability):

$$P[k = 1] = \frac{\lambda^k e^{-\lambda}}{k!} = \lambda e^{-\lambda} = T$$

$$P[k = 0] = \frac{\lambda^k e^{-\lambda}}{k!} = e^{-\lambda} = \text{Idle}$$

$$P[k > 1] = 1 - P[k = 0] - P[k = 1] = \text{Collision}$$

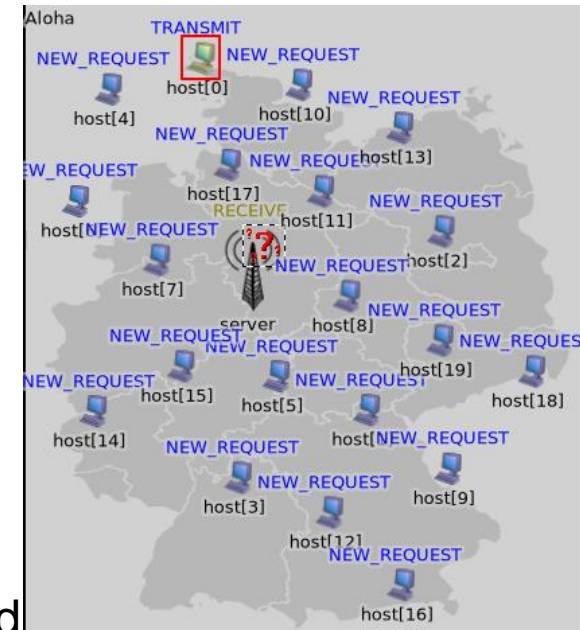


In real applications: retransmissions!

Simulated network



- N MSs (**hosts**)
- One BS (**server**)
- Exponentially distributed *inter-arrival time*
- **Host behavior:**
 - No retransmission
 - Retransmission until acknowledgement is received
 - Uniform / exponential back-off interval between retransmissions
- **Server:**
 - No acknowledgement
 - With acknowledgement



Project structure



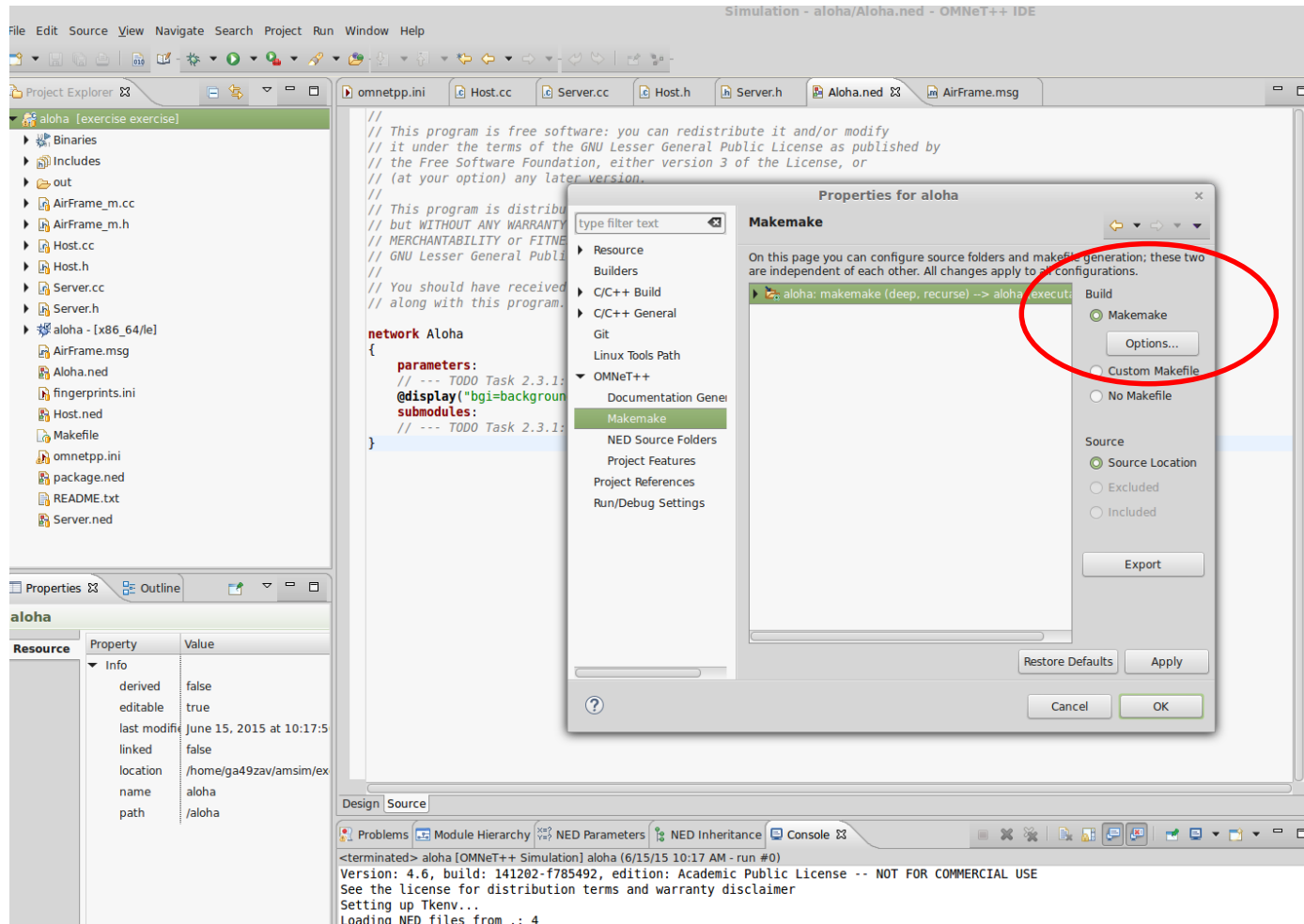
The screenshot shows an IDE with the following components:

- Project Explorer:** Displays the project structure for 'aloha [exercise exercise 12]'. The files listed are: Includes, AirFrame_m.cc, AirFrame_m.h, Host.cc, Host.h, Server.cc, Server.h, AirFrame.msg, Aloha.ned, fingerprints.ini, Host.ned, Makefile, **omnetpp.ini** (selected), package.ned, README.txt, and Server.ned.
- Properties Panel:** Located at the bottom left, it has tabs for 'Properties' and 'Outline'. The 'Properties' tab is active, showing a table with 'Property' and 'Value' headers.
- Code Editor:** Displays the code in 'Host.cc'. The code is a C++ implementation of a network protocol, featuring comments in green and code in purple. It includes a task for 'TODO Task 3.1.5: your code goes here'.

```
/* --- end of the task --- */  
  
}  
  
else if (msg==retryTxTimer){  
    EV << "Waited for backoff, going for retransmission " << endl;  
    ASSERT(state==RETRY);  
    /* --- TODO Task 3.1.5: your code goes here --- */  
    /* --- end of the task --- */  
}  
  
else {  
    error ("Shouldn't end up here");  
}  
  
}  
else {  
    //must be ACK  
    if (state==NEW_REQUEST || state==TRANSMIT){  
        delete msg;  
    }  
    else {  
        ASSERT(state==RETRY);  
  
        if (!checkAck( msg ))  
            error("Invalid ack");  
  
        /* --- TODO Task 3.1.5: your code goes here --- */  
        // add an ACK processing  
  
        /* --- end of the task --- */  
        delete msg;  
    }  
}  
}
```

Project structure (2)

Project-> Properties -> Omnet++ -> Makemake
Set the Build as Makemake



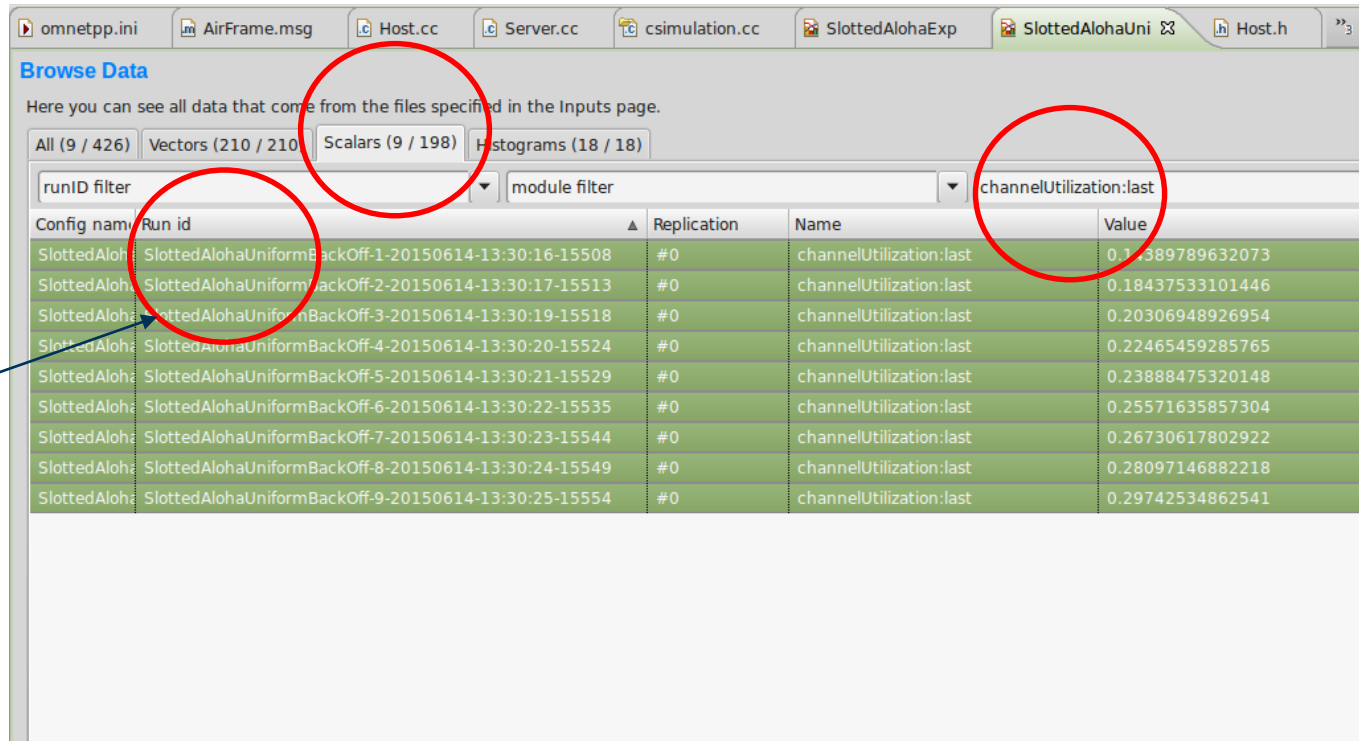
- Create a network and run the simulation ~24 Points
- Add retransmissions to the model ~64 Points
- Randomize back-offs ~12 Points

- Bonus:
 - Re-run 30 times and plot with the 0.95 confidence interval ~10 Points
 - Implement the statistic collection for the ratio of dropped packets ~5 Points
 - Implement exponential back-offs ~10 Points
 - Early bird ~5 Points



Questions?

Exporting the results



Browse Data

Here you can see all data that come from the files specified in the Inputs page.

All (9 / 426) Vectors (210 / 210) Scalars (9 / 198) Histograms (18 / 18)

runID filter module filter channelUtilization:last

Config name	Run id	Replication	Name	Value
SlottedAloha	SlottedAlohaUniformBackOff-1-20150614-13:30:16-15508	#0	channelUtilization:last	0.1389789632073
SlottedAloha	SlottedAlohaUniformBackOff-2-20150614-13:30:17-15513	#0	channelUtilization:last	0.18437533101446
SlottedAloha	SlottedAlohaUniformBackOff-3-20150614-13:30:19-15518	#0	channelUtilization:last	0.20306948926954
SlottedAloha	SlottedAlohaUniformBackOff-4-20150614-13:30:20-15524	#0	channelUtilization:last	0.22465459285765
SlottedAloha	SlottedAlohaUniformBackOff-5-20150614-13:30:21-15529	#0	channelUtilization:last	0.23888475320148
SlottedAloha	SlottedAlohaUniformBackOff-6-20150614-13:30:22-15535	#0	channelUtilization:last	0.25571635857304
SlottedAloha	SlottedAlohaUniformBackOff-7-20150614-13:30:23-15544	#0	channelUtilization:last	0.26730617802922
SlottedAloha	SlottedAlohaUniformBackOff-8-20150614-13:30:24-15549	#0	channelUtilization:last	0.28097146882218
SlottedAloha	SlottedAlohaUniformBackOff-9-20150614-13:30:25-15554	#0	channelUtilization:last	0.29742534862541

Ordered by

Right click -> export as .csv
Uncheck "adding headers"
Plot using matplotlib - recommended