UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Omnet++

UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Overview

Omnet ++

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Definition:

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators.

- Extensible
  - Easy to add new parts of the simulation framework
- Modular/Component-based
  - Can rearrange existing building blocks to define multiple scenarios
- Library
  - Set of components to be re-used
- Framework
  - Support for running simulations and processing data

Omnet ++

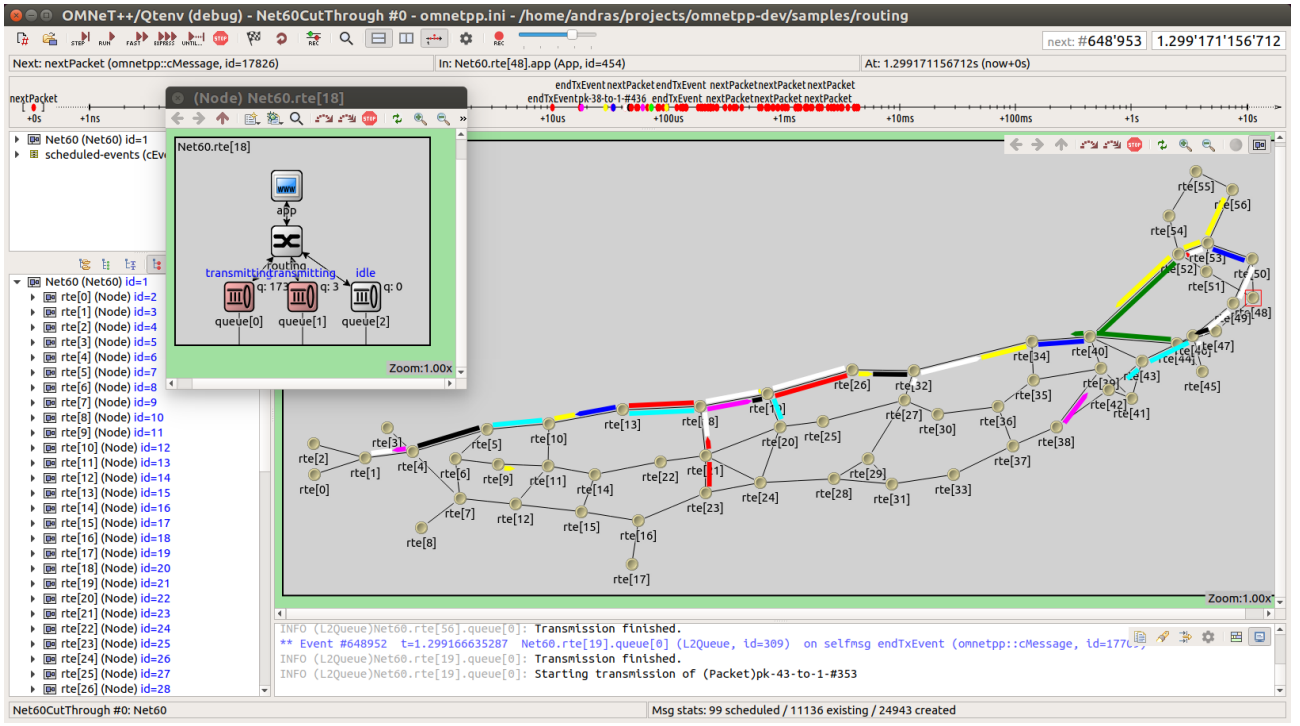
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Two versions
- OMNeT++
  - Open source
  - Free for non-commercial use
- OMNeST++
  - Commercial support
  - Automated install
  - Better support for parallel execution
  - Additional tools

Some screenshots

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- GUI for running simulations
  - Based on QT (since v5)
  - Earlier versions based on Tk
- Support for interactive run/inspection
  - Stop/Start/Step/…
  - Animations
- Details drill-down

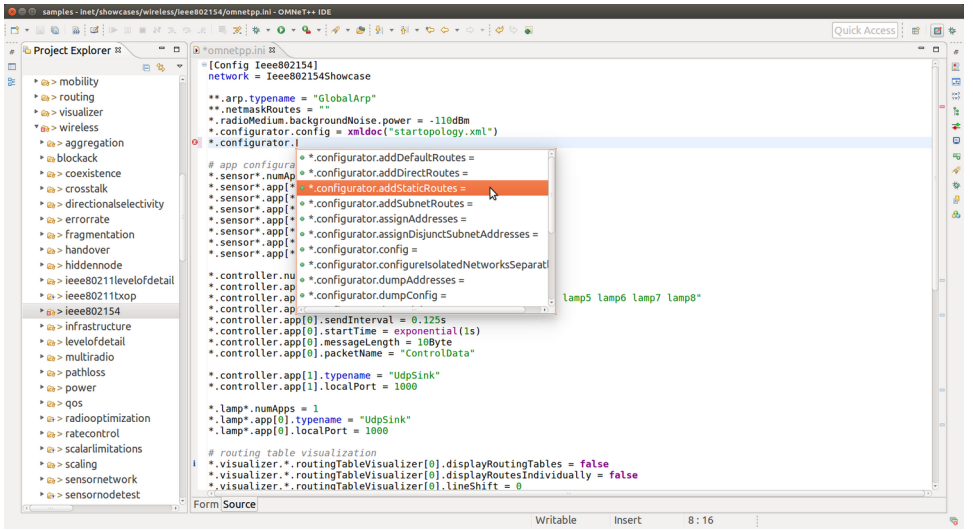


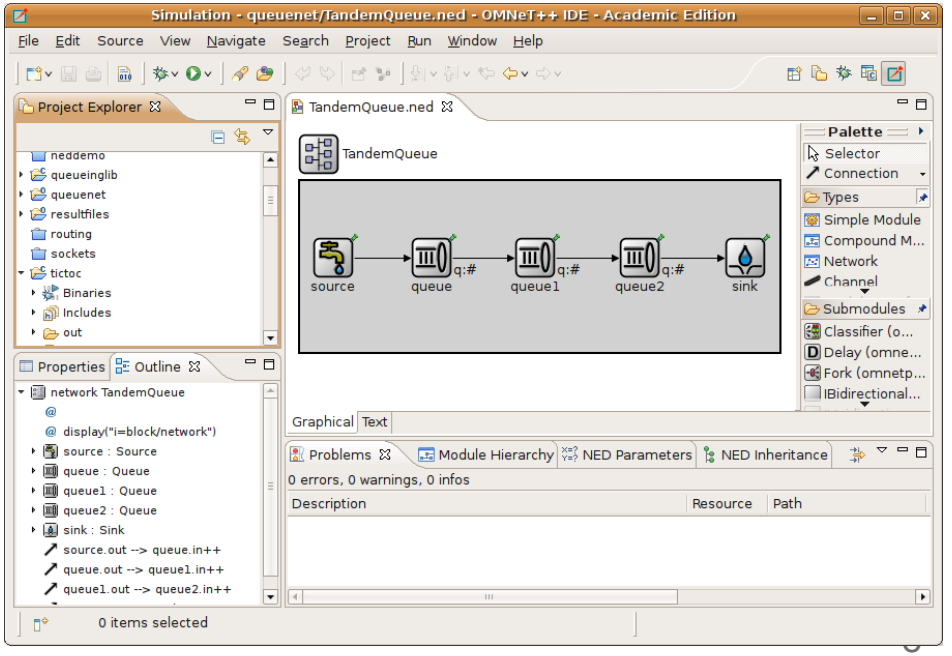
Some screenshots

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

UNIVERSITAS STUDIORUM MUTINENSIS ET REGIENSIS 1175

- IDE for development
- Based on Eclipse
- Support for C/C++ development
  - Writing code
  - Auto-completion
  - Includes debugging
- Support for Omnet-specific files
  - .ini/.ned files
  - Supports also graphical editing

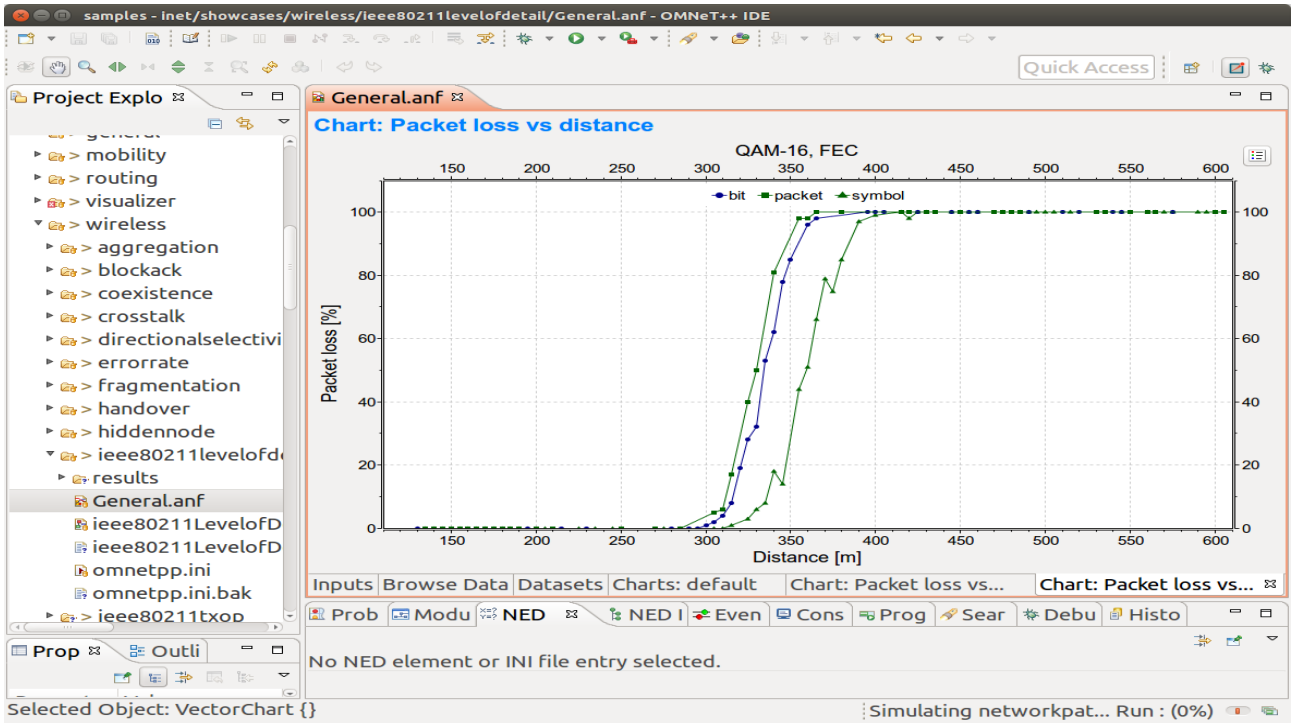




Some screenshots

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

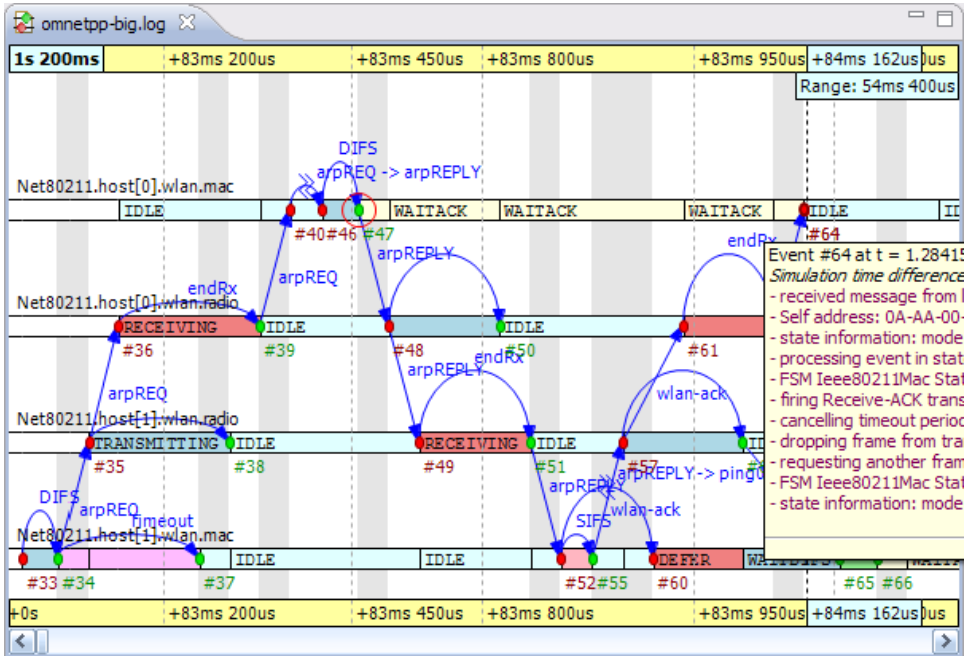
- GUI for data analysis
  - Integrated in IDE
  - Basic analysis tools
- Can be integrated with other methods
- Since version 6 (upcoming)
  - Support for python-based data analysis
  - Python wrappers for some omnet tools



Some screenshots

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Other data representation
- Support for protocol analysts
- Sequence charts
- Management of event logs



UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

(

Compiling Omnet++

Basic instructions

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Install required packages and python resources
  - `$ sudo apt-get install build-essential clang lld gdb bison flex perl python3 python3-pip qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools libqt5opengl5-dev libxml2-dev zlib1g-dev doxygen graphviz libwebkit2gtk-4.0-37`
  - `$ pip3 install numpy pandas matplotlib scipy seaborn posix_ipc`
- Unpack sources
- Setup environment
  - `$ . setnev`

Basic instructions

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Invoke `./configure`
  - Disable unwanted (troublesome) features: OSG and OSG_EARTH
  - `$ ./configure WITH_OSG=no WITH_OSGEARTH=no`
- Compile
  - `$ make -j $(nproc)`
- Run IDE
  - `$ omnetpp`

Directory overview

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Main directory layout
- System directories
  - Binaries: bin
  - Include files: include
  - Icons for sims: images
  - Documentation: docs
  - Sources: src
  - Unit testing: test

```
omnetpp/          OMNeT++ root directory
  bin/            OMNeT++ executables
  include/        header files for simulation models
  lib/            library files
  images/         icons and backgrounds for network graphics
  doc/            manuals, readme files, license, APIs, etc.
    ide-customization-guide/ how to write new wizards for the IDE
    ide-developersguide/ writing extensions for the IDE
    manual/       manual in HTML
    ned2/         DTD definition of the XML syntax for NED files
    tictoc-tutorial/  introduction into using OMNeT++
    api/          API reference in HTML
    nedxml-api/   API reference for the NEDXML library
    parsim-api/   API reference for the parallel simulation library
  src/            OMNeT++ sources
    sim/          simulation kernel
      parsim/     files for distributed execution
      netbuilder/files for dynamically reading NED files
    envir/        common code for user interfaces
    cmdenv/       command-line user interface
    tkenv/        Tcl/Tk-based user interface
    qtenv/        Qt-based user interface
    nedxml/       NEDXML library, nedtool, opp_msgc
    scave/        result analysis library
    eventlog/     eventlog processing library
    layout/       graph layouter for network graphics
    common/       common library
    utils/        opp_makemake, opp_test, etc.
  test/           regression test suite
    core/         tests for the simulation library
    anim/         tests for graphics and animation
    dist/         tests for the built-in distributions
    makemake/     tests for opp_makemake
    ...
```

Directory overview

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Contributors code

```
contrib/      directory for contributed material
  akaroa/     Patch to compile akaroa on newer gcc systems
  topologyexport/  Export the topology of a model in runtime
  ...
```

- IDE

```
ide/            Simulation IDE
  features/     Eclipse feature definitions
  plugins/      IDE plugins (extensions to the IDE can be dropped here)
  ...
```

- Additional tools (scave, splitvec, etc…)

```
tools/        Platform specific tools and compilers (e.g. MinGW/MSYS on Windows)
```

- Samples (working code to study and modify)
- Focus on queuenet example

```
samples/      directories for sample simulations
  aloha/      models the Aloha protocol
  cqn/        Closed Queueing Network
  ...
```

Install analyzer

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Retrieve source
  - `$ git clone https://github.com/rlancellotti/omnet_analyzer.git`
- Install additional packages
  - `$ sudo apt-get install libsqlite2-dev sqlitebrowser`
  - `$ pip3 install -r requirements.txt`
- Install
  - `$ install_tools.py`

The easy way...

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Download the VM disk image from the course site
  - QCow2 disk images (KVM)
    http://web.ing.unimo.it/rlancellotti/materiale/omnet.qcow2.bz2
- Set up a Virtualization system
  - VMWare, Virtualbox, KVM
- Prepare disk (possibly convert qcow → vmdk)
  - `$ qemu-img convert -p -f qcow2 -O vmdk <IMG>.qcow2 <IMG>.vmdk`
- Create VM with disk image

vmware®

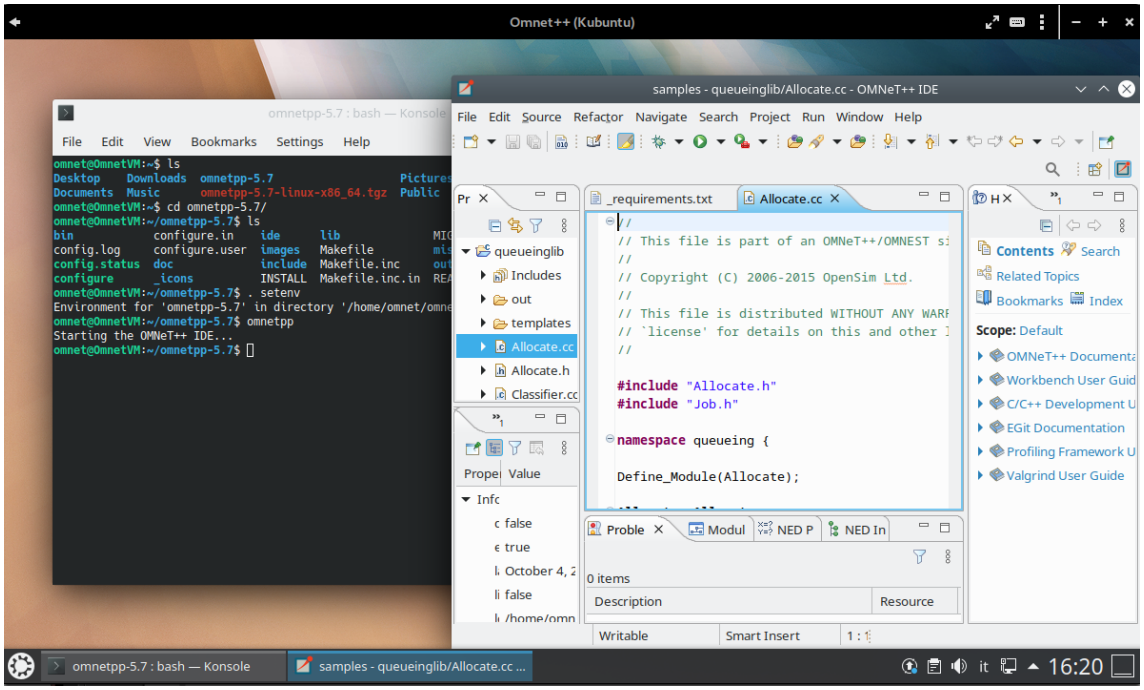
The easy way...

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Run the VM and enjoy Omnet++
  - User: omnet
  - Password: omnet
- To run omnet:

```
$ cd omnetpp-6.0.1
$ . setenv
$ omnetpp
```

- Import examples
- Forget about Inet++



Simulator structure

Parts of a simulator

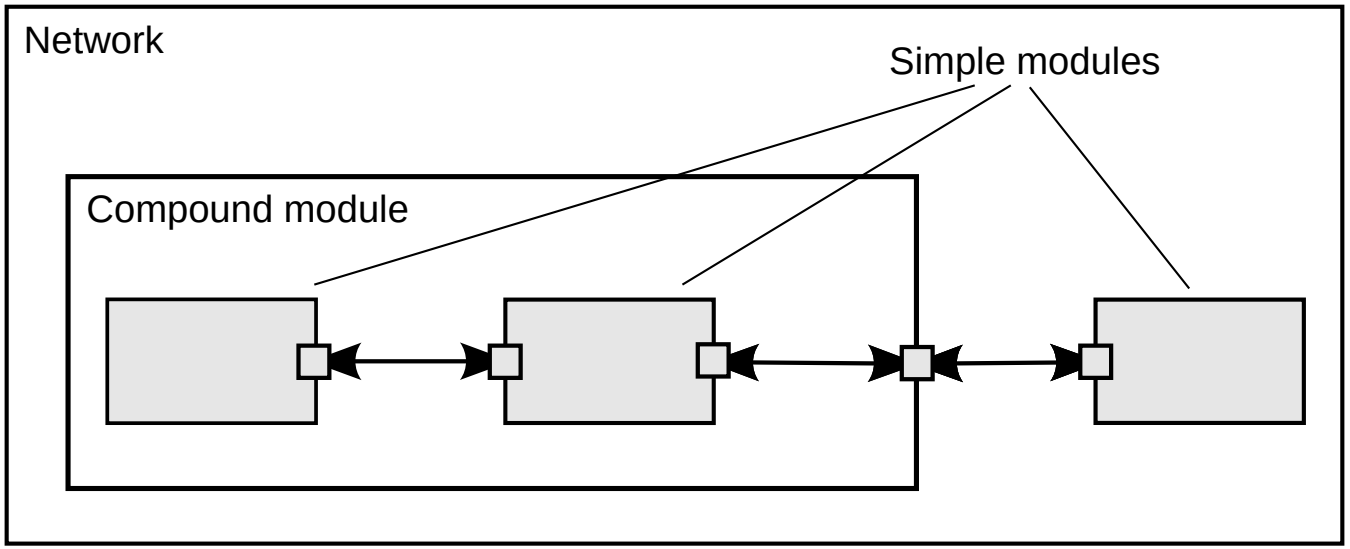
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Basic elements of Omnet++ framework
  - Modules
  - Gates
  - Channels
  - RNG & Data collectors
  - Messages
- Simulation description
  - Network topology
  - Configuration(s)

Modules

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Two types of modules
  - Simple modules
  - Compound modules
- Compound module
  - Composed of multiple modules
- Simple module
  - Atomic entity
  - Written in C++
- Network
  - Top level compound module



Simple Module

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Two sources:
- C++ Source
  - Defines module behavior
- NED source
  - NED = NEtwork Description
  - Defines module interface
  - Can support additional meta-data

Simple Module

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Two types of sources:
- C++ Source (.cpp, .h)
  - Class extending `cSimpleModule`
  - Defines module behavior
  - Main method: `handleMessage(msg)`
- Relevant example:
  - Look at code for module source
  - In `samples/queueinglib/`

Simple Module

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA


```
#ifndef __QUEUEING_SOURCE_H
#define __QUEUEING_SOURCE_H

#include "QueueingDefs.h"

namespace queueing {

class Job;

/**
 * Abstract base class for job generator modules
 */
class QUEUEING_API SourceBase : public cSimpleModule
{
    protected:
        int jobCounter;
        std::string jobName;
        simsignal_t createdSignal;
    protected:
        virtual void initialize() override;
        virtual Job *createJob();
        virtual void finish() override;
};


/**
 * Generates jobs; see NED file for more info.
 */
class QUEUEING_API Source : public SourceBase
{
    private:
        simtime_t startTime;
        simtime_t stopTime;
        int numJobs;

    protected:
        virtual void initialize() override;
        virtual void handleMessage(cMessage *msg) override;
};
```

- Classes extends `cSimpleModule`
- Relevant methods
  - `intialize()`
  - `handleMessage()`
- Focus on Source module
  - Event-driven programming
- Relevant functions:
  - `send()`
  - `emit()`
  - `scheduleAt()`


```
void SourceBase::finish()
{
    emit(createdSignal, jobCounter);
}

//----

Define_Module(Source);

void Source::initialize()
{
    SourceBase::initialize();
    startTime = par("startTime");
    stopTime = par("stopTime");
    numJobs = par("numJobs");

    // schedule the first message timer for start time
    scheduleAt(startTime, new cMessage("newJobTimer"));
}

void Source::handleMessage(cMessage *msg)
{
    ASSERT(msg->isSelfMessage());

    if ((numJobs < 0 || numJobs > jobCounter) && (stopTime < 0 || stopTime > simTime())) {
        // reschedule the timer for the next message
        scheduleAt(simTime() + par("interArrivalTime").doubleValue(), msg);

        Job *job = createJob();
        send(job, "out");
    }
    else {
        // finished
        delete msg;
    }
}
```

Simple Module

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- NED source
  - NED = NEtwork Description
  - Defines module interface
    - Parameters
    - Connections for other modules (gates)
  - Can support additional meta-data
    - Es. explicit declaration of C++ class
      → useful in libraries or when inheritance is used

```
simple Queue
{
    parameters:
        int capacity;
        @class(mylib::Queue);
        @display("i=block/queue");
    gates:
        input in;
        output out;
}
```


```
simple Source
{
    parameters:
        @group(Queueing);
        @signal[created](type="long");
        @statistic[created](title="the number of jobs created";record=last;interpolationmode=none);
        @display("i=block/source");
        int numJobs = default(-1);               // number of jobs to be generated (-1 means no limit)
        volatile double interArrivalTime @unit(s); // time between generated jobs
        string jobName = default("job");         // the base name of the generated job (will be the module name if left empty)
        volatile int jobType = default(0);       // the type attribute of the created job (used by classifers and other modules)
        volatile int jobPriority = default(0);   // priority of the job
        double startTime @unit(s) = default(interArrivalTime); // when the module sends out the first job
        double stopTime @unit(s) = default(-1s); // when the module stops the job generation (-1 means no limit)
    gates:
        output out;
}
```

- `@signal`: name of signal sent
- `@statistic`: how to collect signals
- `@display`: icon, color, etc...
- Parameters (with type and units of measure)

Gates

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Gates
  - Ports for message exchange
- Cardinality
  - Single port
  - Array of ports
- Direction
  - In
  - Out
  - InOut

```
simple Classifier {
    parameters:
        int numCategories;
    gates:
        input in;
        output out[numCategories];
}
```

Channels

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Channels
  - Links that connect gates
- Characteristics of a channel
  - Ideal
  - Channel with delay
  - Channel with delay and bandwidth
  - Error rate can also be specified
- Channels can have a direction
  - Directed flow
  - Bidirectional
- Channel support inheritance

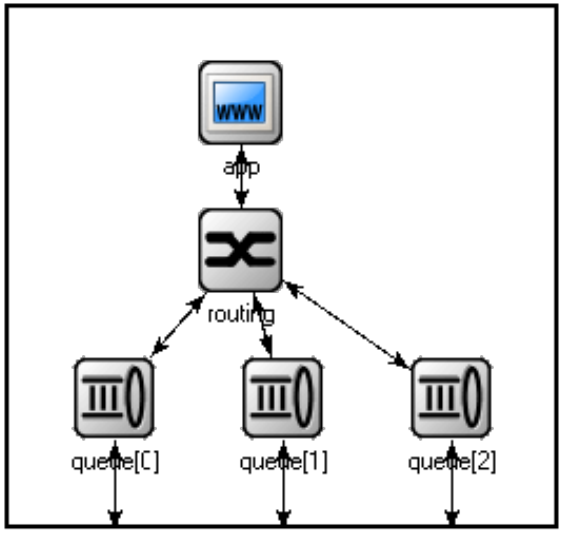
```
channel Ethernet100 extends ned.DatarateChannel
{
    datarate = 100Mbps;
    delay = 100us;
    ber = 1e-10;
}
```

```
a.g++ <--> Ethernet100 <--> b.g++;
a.g++ <--> Backbone {cost=100; length=52km; ber=1e-8;} <--> b.g++;
a.g++ <--> Backbone {@display("ls=green,2");} <--> b.g++;
```

Compound node

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- A node defined using other modules
- Defines
  - Sub-modules
  - Interconnections
- No C++ code
- Only NED code
  - Can use a powerful syntax




```
module Node
{
    parameters:
        int address;
        @display("i=misc/node_vs,gold");
    gates:
        inout port[];
    submodules:
        app: App;
        routing: Routing;
        queue[sizeof(port)]: Queue;
    connections:
        routing.localOut --> app.in;
        routing.localIn <-- app.out;
        for i=0..sizeof(port)-1 {
            routing.out[i] --> queue[i].in;
            routing.in[i] <-- queue[i].out;
            queue[i].line <--> port[i];
        }
}
```

RNG

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Parameters can define random variables
  - Use of `volatile` keyword

```
simple App
{
    parameters:
        string protocol;        // protocol to use: "UDP" / "IP" / "ICMP" / ...
        int destAddress;        // destination address
        volatile double sendInterval @unit(s) = default(exponential(1s));
                                // time between generating packets
        volatile int packetLength @unit(byte) = default(100B);
                                // length of one packet
        volatile int timeToLive = default(32);
                                // maximum number of network hops to survive
    gates:
        input in;
        output out;
}
```

- Additional expressions can be used

```
**.serviceTime = simTime()<1000s ? 1s : 2s  # queue that slows down after 1000s
```

Data collections

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Omnet++ supports data collection and aggregation
  - Vectors (time series)
  - Histograms
  - Quantile estimation (PSquare algorithm)
- Support from C++ code
  - Direct access to classes in modules
- Can create data histogram from simulation definition
  - Modules emit events
  - Events can be collected and aggregated
  - Definition from NED code

Vector files

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

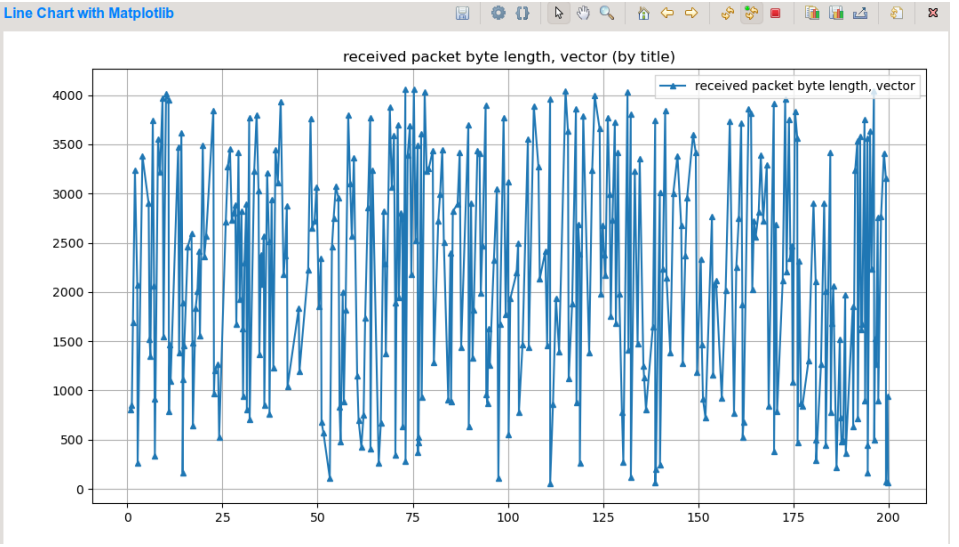
- Column-base file
- Can become huge
  - Main reason for simulation failure
  - Disk full of .vec files
- Can be analyzed using external tools
- Built-in tool
  - In Omnetpp 6 → python-based

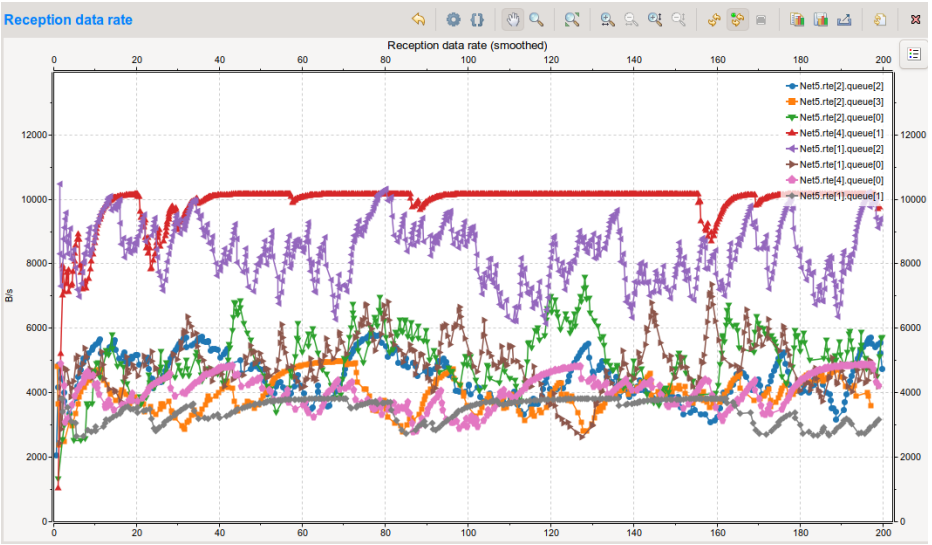
```
# vector 3 FogNaive.cloud[0].sink lifeTime:vector ETV
4       0.023772122963  0.015987500417
10      0.06209147189   0.019643986444
17      0.07611450023   0.029026615012
25      0.089211438456  0.024731157796
28      0.101594410192  0.036910655029
34      0.107927160112  0.021334313679
37      0.112338267093  0.038865135537
39      0.118570585276  0.041834692358
46      0.124403878325  0.018287277716
47      0.126370439707  0.049531881061
55      0.138142315187  0.033182163189
61      0.147263698427  0.027906351601
62      0.148009214901  0.018377387159
64      0.157761064522  0.037953012724
71      0.169610013804  0.039342490285
74      0.178196945305  0.055100527776
77      0.187060357165  0.052354140642
81      0.199063518235  0.060642278146
```

Vector files

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Built-in tool for analysis
- Can plot simple data series
- An provide more complex plotting features



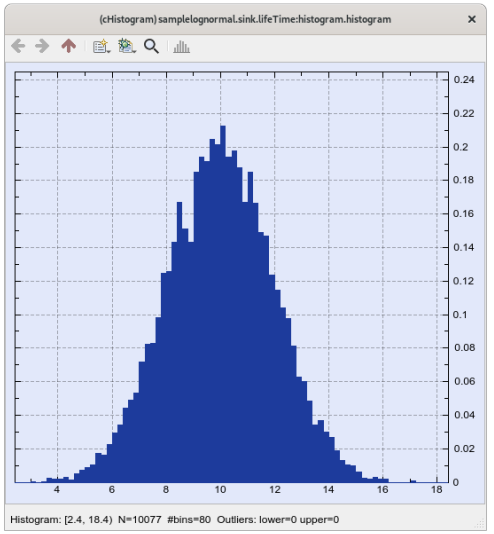


Histograms

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Compact representation of probability distribution
- Can describe stochastic process
- Automatic mechanism for determining
  - Number of bins
  - Ranges
- Operation on data
  - Export (including additional info)
  - Visualization

```
statistic FogMeshNetworkDelay.fog[0].server[0] queueLength:histogram
field count 44475
field mean 2.4617425519955
field stddev 2.1991489549599
field min 0
field max 8
field sum 109486
field sqrsum 484614
attr interpolationmode sample-hold
attr title "queue length, histogram"
bin     -inf    0
bin     0       10007
bin     1       8816
bin     2       7050
bin     3       5768
bin     4       4236
bin     5       3292
bin     6       2408
bin     7       1770
bin     8       1128
bin     9       0
bin     10      0
```



Histograms

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Reference class:
  - Class `cLongHistogram`
- Can be declared as a a class property
- Easy to use
  - Just invoke `record()` method
- Typically used together with vectors

```
class Txc15 : public cSimpleModule
{
  private:
    long numSent;
    long numReceived;
    cLongHistogram hopCountStats;
    cOutVector hopCountVector;

  protected:
```

```
hopCountVector.record(hopcount);
hopCountStats.collect(hopcount);
```

Histograms

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Signal-based histograms
- In .ini file: enable histograms
  - `result-recoding-modes`
- Statistics for signal must be defined in `.ned` file

```
**.source[15].sendInterval = exponential(1s/54.00009000000001)
**.source[16].sendInterval = exponential(1s/18.00009)
**.source[17].sendInterval = exponential(1s/72.00009)
**.source[18].sendInterval = exponential(1s/108.00009000000001)
**.source[19].sendInterval = exponential(1s/54.00009000000001)
**.source[*].suggestedTime = 1s * lognormal(log((1/100)*(1/sqrt(2))), sqrt(log(2)))
**.source[*].packetLength = 1B
**.fog[*].queue=9
**.fog[*].nServers = 1
**.fog[*].dispatcher.load.result-recording-modes = +histogram
**.fog[*].server[*].queueLength.result-recording-modes = +histogram
**.fog[*].server[*].load.result-recording-modes = +histogram
**.fog[*].server[*].timeSlice = -1s
```

```
simple FogPU
{
    parameters:
        //@group(Queueing);
        @display("i=block/queue;q=queue");
        @signal[dropped](type="long");
        @signal[queueLength](type="long");
        @signal[load](type="double");
        @signal[queueingTime](type="simtime_t");
        @signal[busy](type="bool");
        @statistic[dropped](title="drop event";record=vector?,count;interpolationmode=none);
        @statistic[queueLength](title="queue length";record=vector,timeavg,max;interpolationmode=sample-hold);
        @statistic[load](title="advertised load";record=vector,timeavg,max;interpolationmode=sample-hold);
        @statistic[queueingTime](title="queueing time at dequeue";record=vector?,mean,max;unit=s;interpolationmode=none);
        @statistic[busy](title="server busy state";record=vector?,timeavg;interpolationmode=sample-hold);
```

Quantiles

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Reference class:
  - Class cPSquare
- Similar to histogram
- Can record a list of pre-defined quantiles
- Used like an histogram
- Less frequently used compared to other recording facilities

Messages

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Message: unit of information exchanged among modules
- Typical behavior:
  - Exits from a *out* gate
  - Transits over a channel
  - Enters a *in* gate
- There can be exceptions (e.g. for radio packets)
- Messages can have a content
  - Fields defined in a .msg file
  - Used to generate C++ code
- Packet
  - Special message with a size (for BW channels)

A simple Network

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

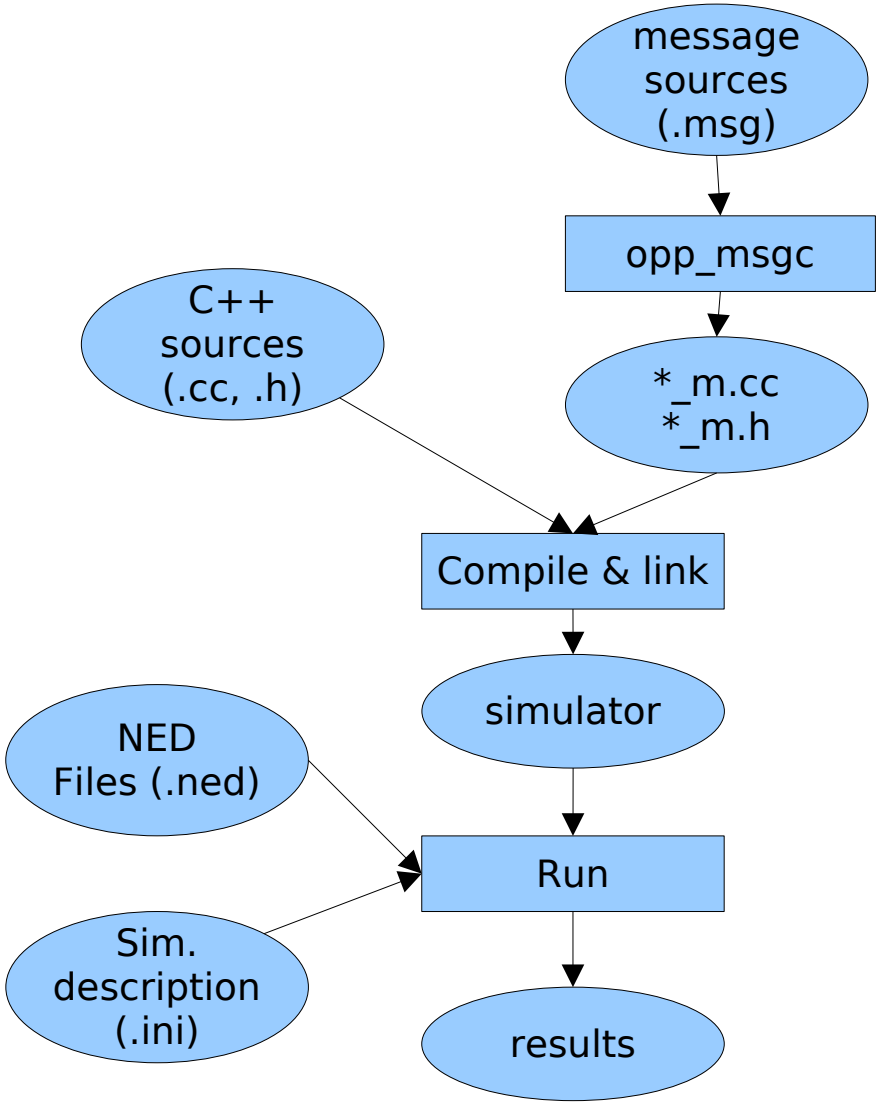
- Network
- Top level element of a simulation
- Like a compound module

```
//
// A network
//
network Network
{
    submodules:
        node1: Node;
        node2: Node;
        node3: Node;
        ...
    connections:
        node1.port++ <--> {datarate=100Mbps;} <--> node2.port++;
        node2.port++ <--> {datarate=100Mbps;} <--> node4.port++;
        node4.port++ <--> {datarate=100Mbps;} <--> node6.port++;
        ...
}
```

Running a simulation

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Define modules
  - .ned files
  - .cc, .h sources
- Create messages classes
  - .msg files
- Compile and link
- Define scenarios and parameters
  - .ned (network definition)
  - .ini (parameter values, multiple configurations)
- Run simulation(s)
- Analyze data

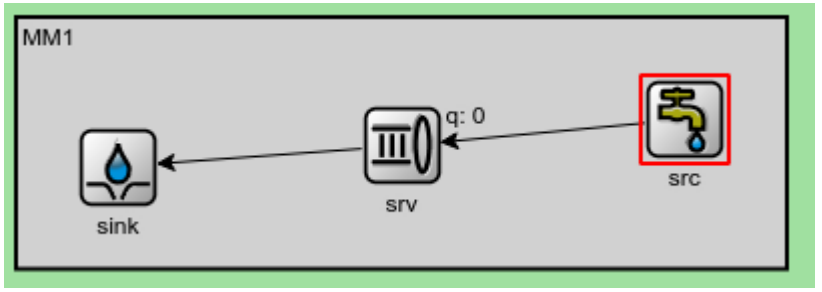


Hands-on

Goal

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Evaluate a M/M/1/K Queue
  - Different load values $\rho = \lambda/\mu$
  - Different queue length K
- Define Queuing-based model
  - Data source (src)
  - Server = server+queue (srv)
  - Data collector (sink)



Look at relevant files

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Network definition
  - .../samples/queuenet/MM1.ned
- Example of simple model
  - .../samples/queueinglib/Source.{cc,h,ned}
- Configuration
  - .../samples/queuenet/MM1.ini
- Too many configurations to handle
  → Use of mako templating engine
  - Utility update_template.py
  - .../samples/queuenet/sample_MM1.ini.mako

MM1.ned

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA


```
import org.omnetpp.queueing.Queue;
import org.omnetpp.queueing.Sink;
import org.omnetpp.queueing.Source;

network MM1
{
    parameters:
        int K=default(10);
        double rho=default(0.8);
        double mu =default(10);
        double lambda = mu * rho;
        src.interArrivalTime = exponential(1s/lambda);
        srv.serviceTime = exponential(1s/mu);
        srv.capacity = K;
    submodules:
        src: Source;
        srv: Queue;
        sink: Sink;
    connections:
        src.out --> srv.in++;
        srv.out --> sink.in++;
}
```

MM1.ini

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

```
[General]
ned-path = .;../queueinglib
network = MM1
#cpu-time-limit = 60s
cmdenv-config-name = MM1Base
tkenv-default-config = MM1Base
qtenv-default-config = MM1Base
repeat = 5
sim-time-limit = 10000s
#debug-on-errors = true
**.vector-recording = false

# parameters of the simulation
[Config MM1Base]
description = "Global scenario"
**.srv.queueLength.result-recording-modes = +histogram
**.sink.lifeTime.result-recording-modes = +histogram

[Config MM1_rho01_K5]
extends=MM1Base
**.K = 5
**.rho=0.1

[Config MM1_rho02_K5]
extends=MM1Base
**.K = 5
**.rho=0.2
```

Sample_MM1.ini.mako

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

```
[General]
ned-path = .;../queueinglib
network = MM1
#cpu-time-limit = 60s
cmdenv-config-name = MM1Base
tkenv-default-config = MM1Base
qtenv-default-config = MM1Base
repeat = 5
sim-time-limit = 10000s
#debug-on-errors = true
**.vector-recording = false

# parameters of the simulation
[Config MM1Base]
description = "Global scenario"
**.srv.queueLength.result-recording-modes = +histogram
**.sink.lifeTime.result-recording-modes = +histogram
%for K in [5, 7, 10, -1]:
%for rho in [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]:

[Config MM1_rho${"%02d" % int(rho*10)}_K${K if K>0 else "inf"}]
extends=MM1Base
**.K = ${K}
**.rho=${rho}
%endfor
%endfor
```

Running the simulation

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Several configurations
- Several runs for each configuration
- Runs should be
  - Automatic
  - Parallel
- Use of utility make
- Need a script to invoke simulator:
- Script `run`

```
#!/bin/sh
./queuenet $*
```

Running the simulation

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Make script executable
  - `$ chmod a+x run`
- Creating Runfile
  - `$ make_runfile.py -f MM1.ini`
- Running experiments
  - `$ make -j $(nproc) -f Runfile`

Data analysis

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Output files
  - .sca files (scalar)
  - .vec files (vectors)
- Use of batch processing capabilities
- Use of multiple runs to compute confidence intervals

Look at relevant files

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Data and analysis definition
  - .../samples/queuenet/configMM1.json
- Relevant info
  - Parameters defining a scenario
  - Metrics of interests
  - Analyses to carry out
- Collect data
  - `$ parse_data.py -c configMM1.json -d MM1.db \ -j $(nproc) -r results/MM1_*-#0.sca`
- Look at the database
  - MM1.db

ConfigMM1.json

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA


```
{
    "scenario_schema": {
        "rho": {"pattern": "**.rho", "type": "real"},
        "K": {"pattern": "**.K", "type": "int"}
    },
    "metrics": {
        "TotalJobs": {"module": "**.src", "scalar_name": "created:last" ,"aggr": ["none"]},
        "DroppedJobs": {"module": "**.srv", "scalar_name": "dropped:count" ,"aggr": ["none"]},
        "QLen": {"module": "**.srv", "scalar_name": "queueLength:timeavg" ,"aggr": ["none"]},
        "Utilization": {"module": "**.srv", "scalar_name": "busy:timeavg" ,"aggr": ["none"]},
        "PQueue": {"module": "**.sink", "scalar_name": "queuesVisited:mean" ,"aggr": ["none"]},
        "ServiceTime": {"module": "**.sink", "scalar_name": "totalServiceTime:mean" ,"aggr": ["none"]},
        "WaitingTime": {"module": "**.sink", "scalar_name": "totalQueueingTime:mean" ,"aggr": ["none"]},
        "ResponseTime": {"module": "**.sink", "scalar_name": "lifeTime:mean" ,"aggr": ["none"]}
    },
    "analyses": {
        "SensRho-Kinf": {
            "outfile": "results/loadcurve_inf.data",
            "scenarios": {
                "fixed": {"K": "-1"},
                "range": ["rho"]
            },
            "metrics": [
                {"metric": "TotalJobs", "aggr": "none"},
                {"metric": "DroppedJobs", "aggr": "none"},
                {"metric": "PQueue", "aggr": "none"},
                {"metric": "ServiceTime", "aggr": "none"},
                {"metric": "WaitingTime", "aggr": "none"},
                {"metric": "ResponseTime", "aggr": "none"}
                    ]
        },
```

Producing output

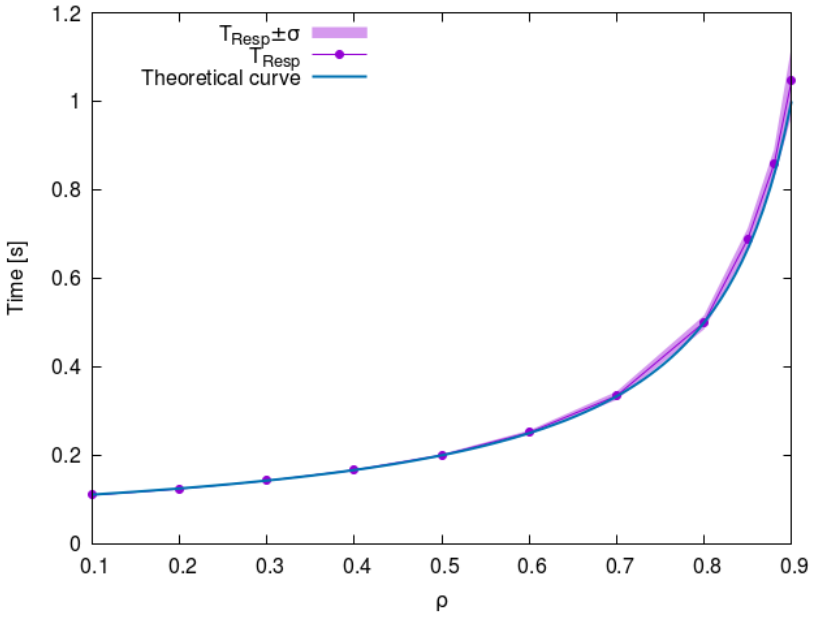
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

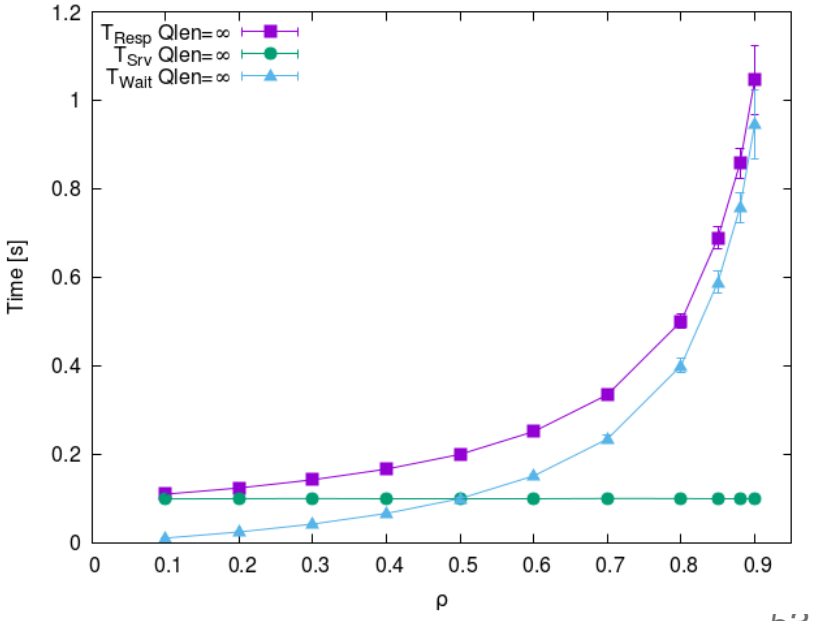
- Analyze data
  - `$ analyze_data.py -c configMM1.json -d MM1.db`
- Look at the relevant files
  - .../samples/queuenet/results/*.data
- Create plots using gnuplot
  - .../samples/queuenet/MM1.gnuplot

Simulator validation

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Loss of precision for high utilization
  - Look a theoretical curve of response time
  - Still within confidence intervals
- Analysis of contributions to response time
  - Service time constant
  - Queuing time grows with utilization
  - Queuing is also source of variance

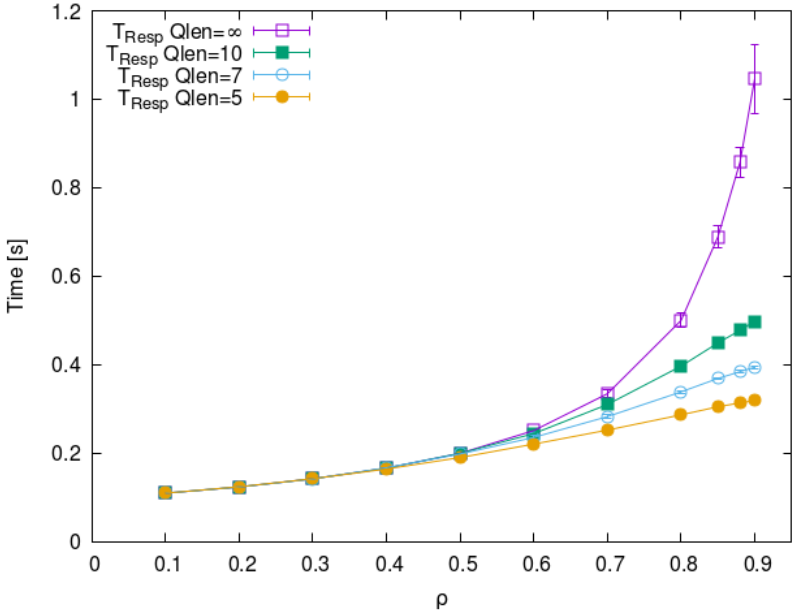


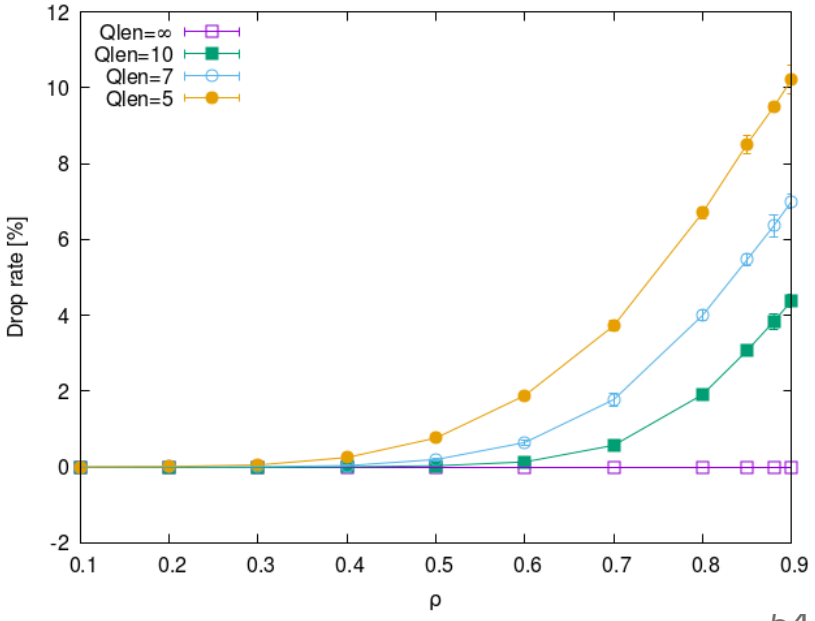


Impact of queue length

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Impact of queue length
- Response time
  - Smaller queue → less variance
- Drop rate
  - Smaller queue → lower drop rate
- Small percentage of dropped jobs
  - es. Q=10 → drop rate <5%
  - High impact of response Time

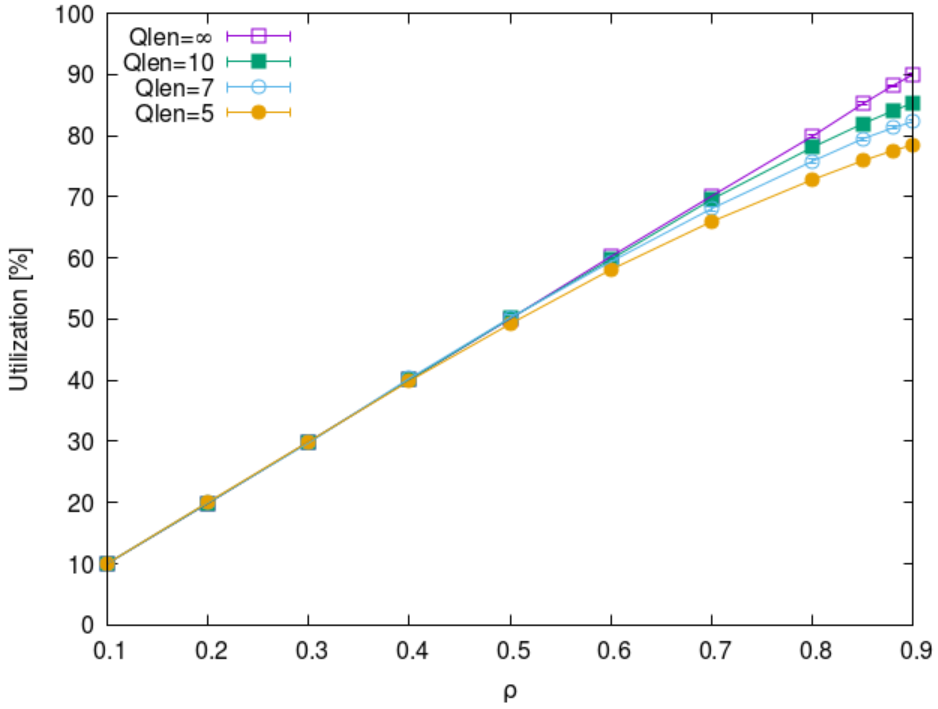




Results analysis

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Utilization to explain lower response time
- For high utilization:
  - Small reduction in utilization
    → high impact on response time



Additional exercises

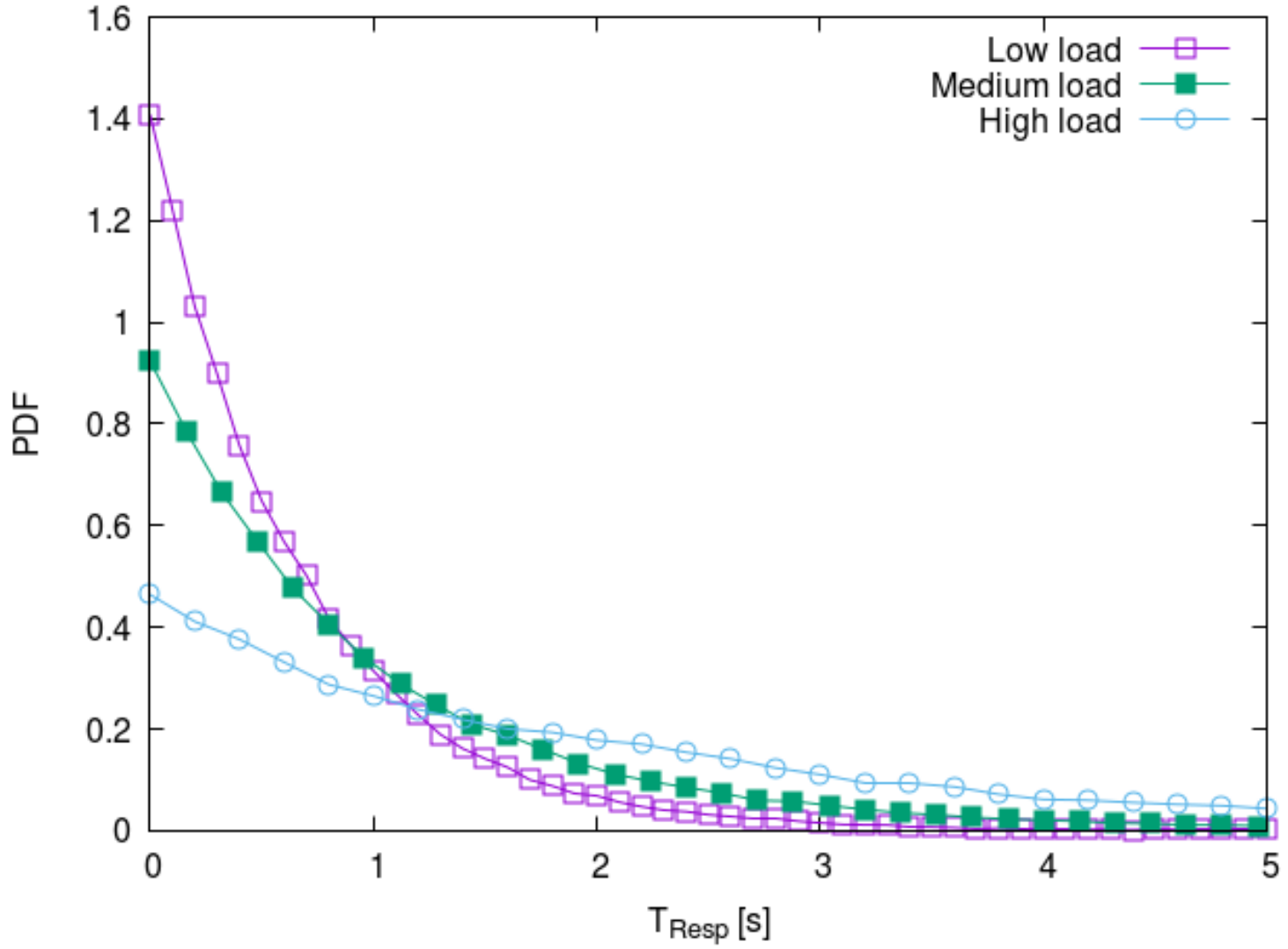
More exercises on M/M/1

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Additional simulations to carry out
- Load balanced M/M/1 systems
  - 2 sources, 2 servers, 1 sink
  - For each source and server: λ=4 req/s, μ=5 req/s
  - Response time (average and histogram)
- Load un-balanced M/M/1 systems
  - Same topology
  - λ_1=4.5 req/s, λ_2=3.5 req/s, $\mu_1=\mu_2$=5 req/s
  - What happens to response time?

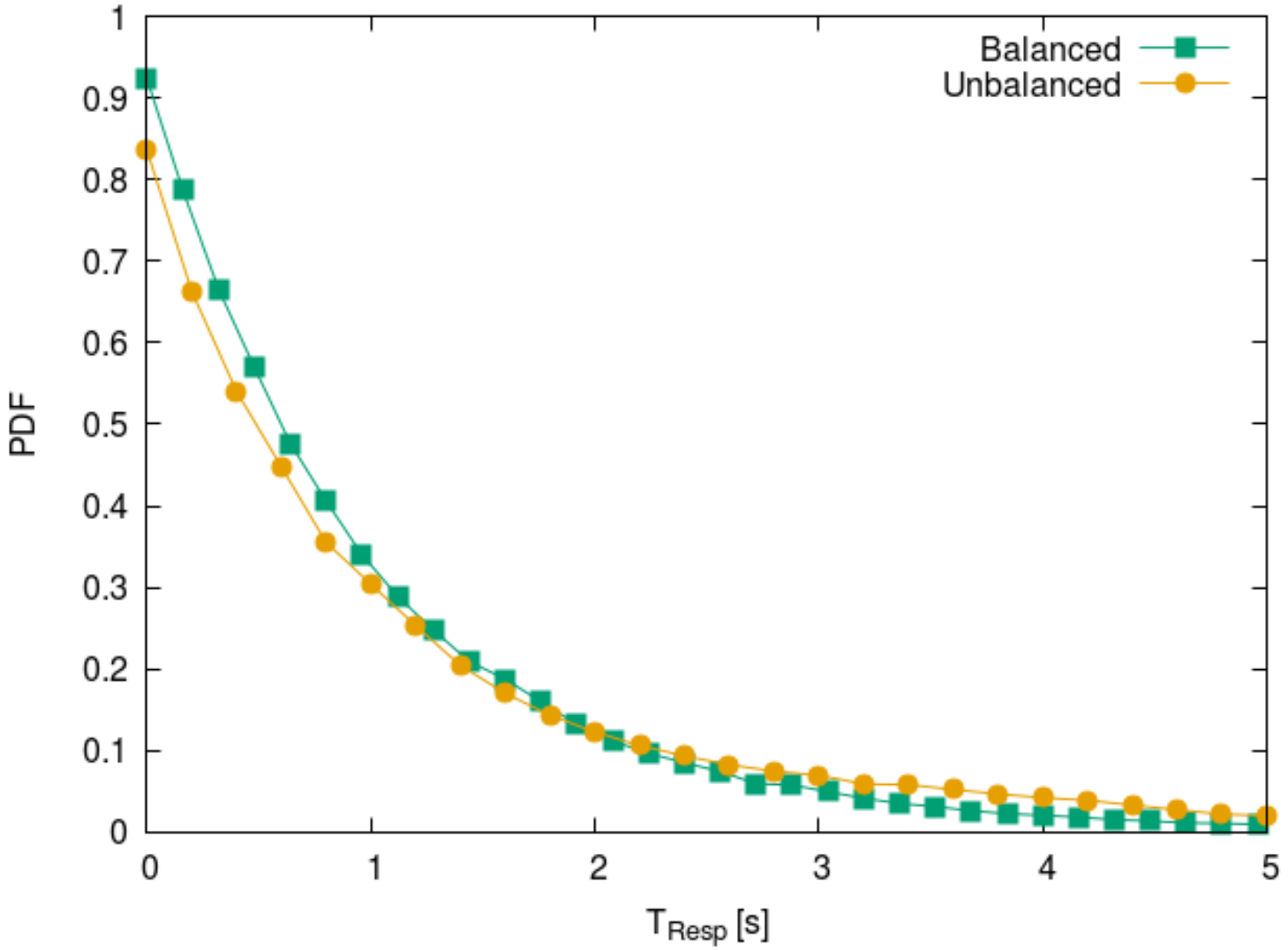
Some results

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA



Some results

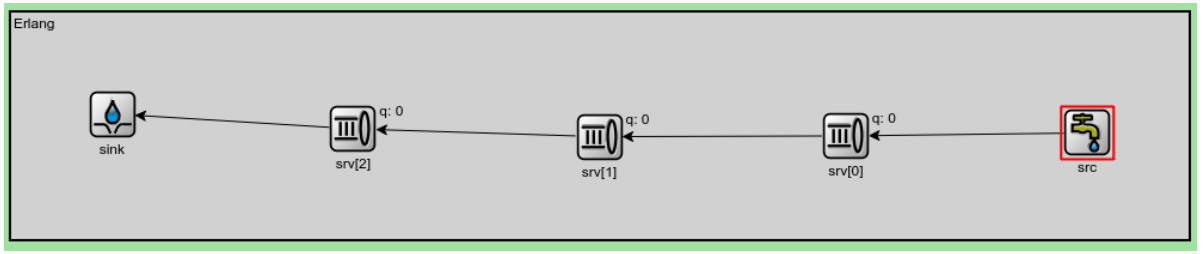
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA



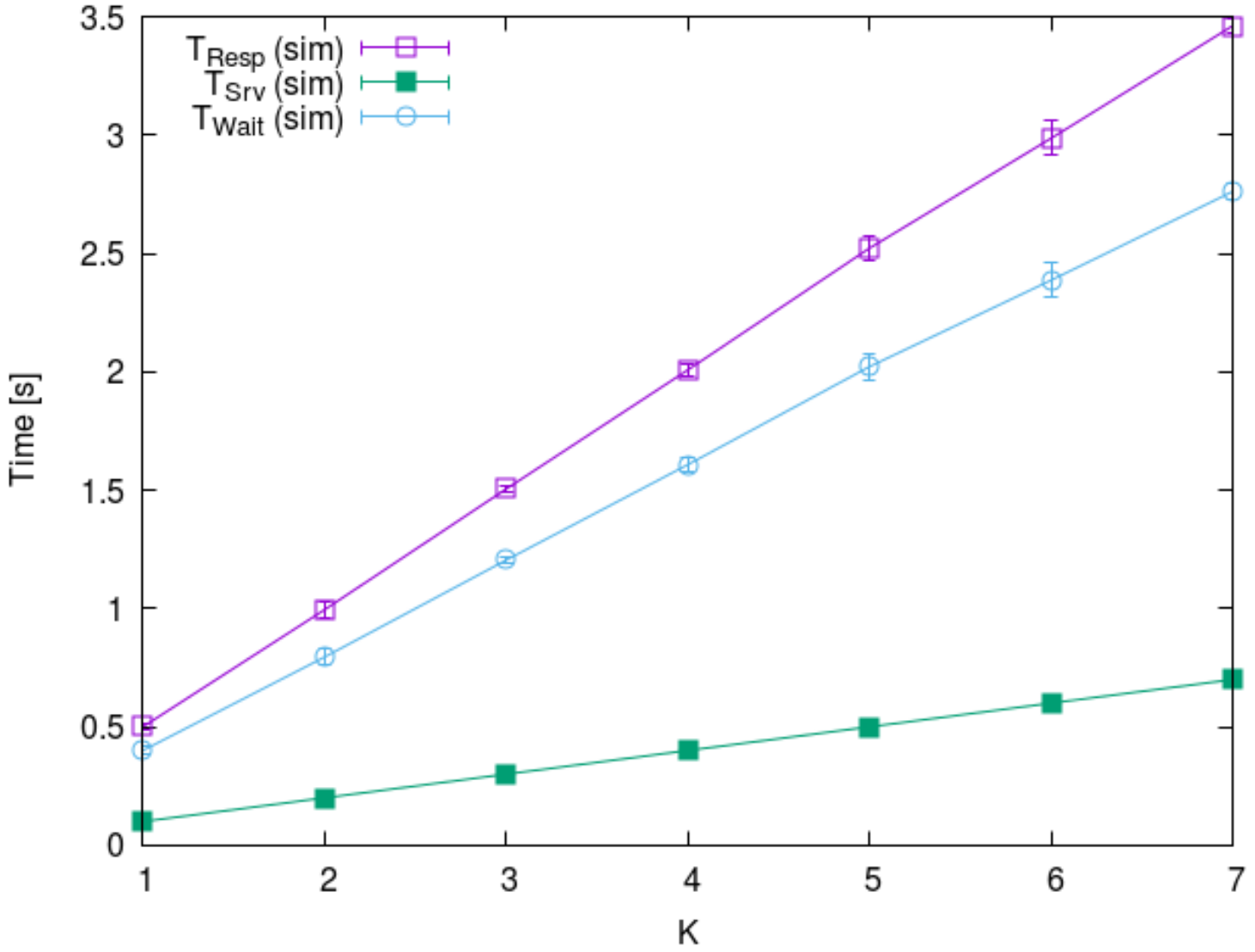
Erlang model

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Chain of M/M/1 systems
  - 1 sources
  - N servers
  - 1 sink

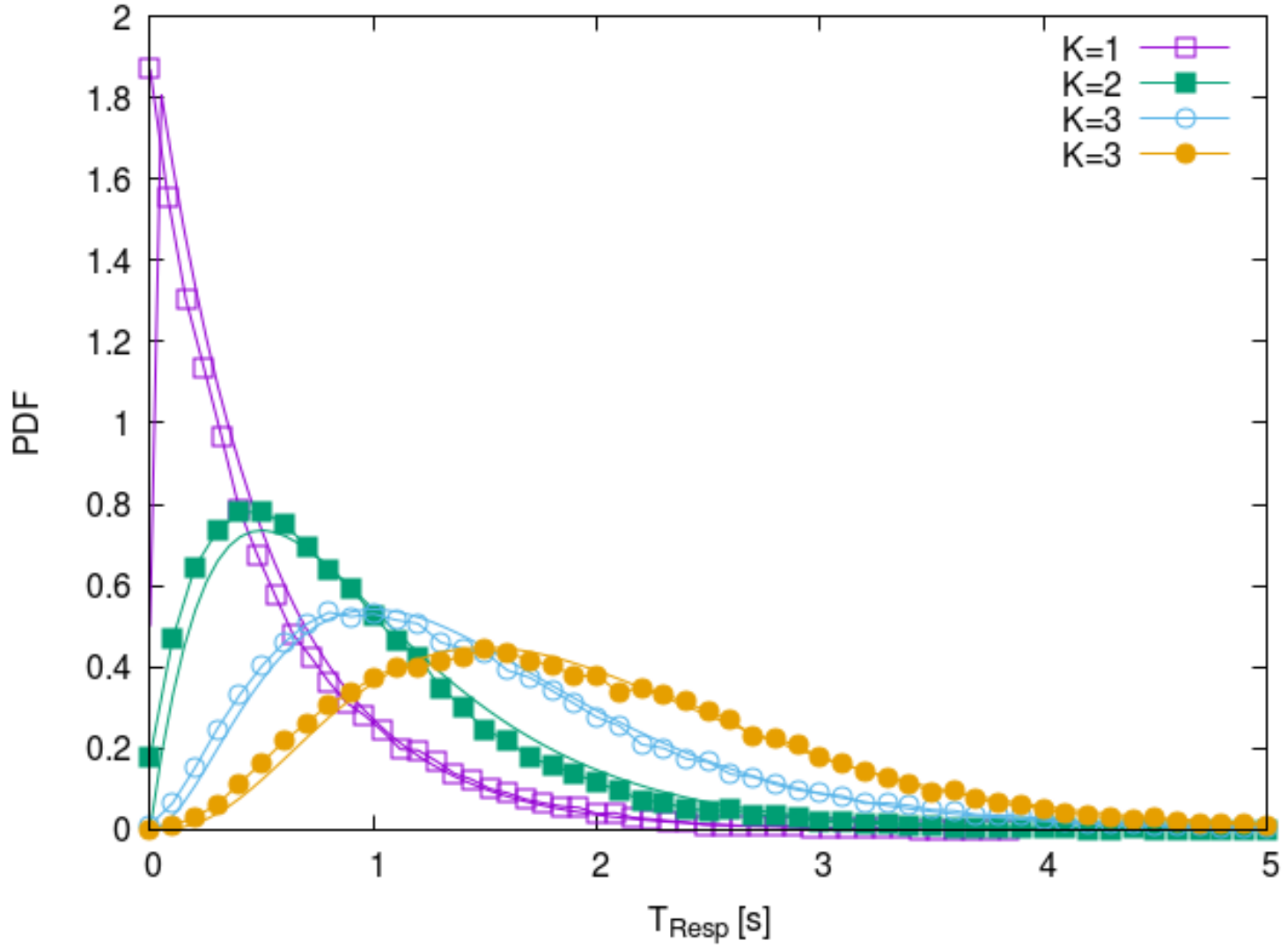


- Arrays of Modules
  - srv[N]: Source;
- Iteration to connect nodes
- Evolution of response time as N changes
  - Average values
  - Histogram of response time distribution
  - Erlang distribution


T_{Resp} (sim)
T_{Srv} (sim)
T_{Wait} (sim)
Time [s]
K
0
0.5
1
1.5
2
2.5
3
3.5
1
2
3
4
5
6
7

Some results

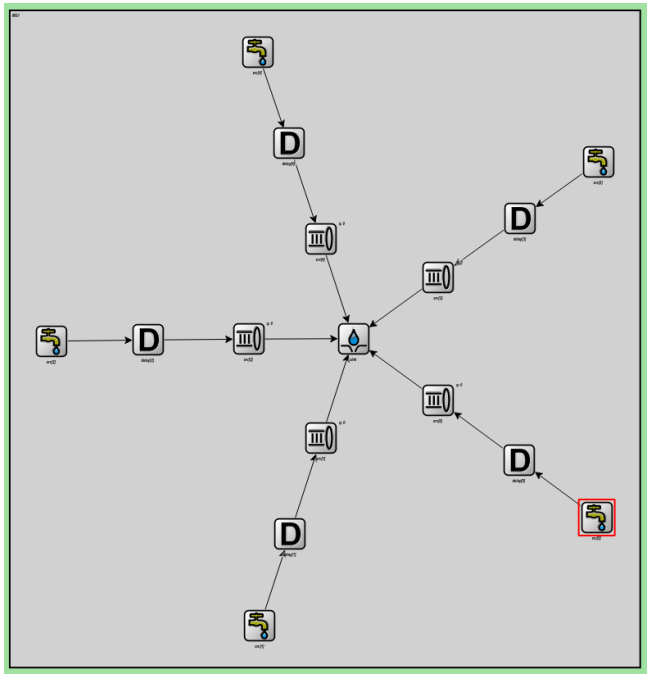
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA



Complex topology with delays

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Create a complete topology
  - N sources
  - N servers
  - Network delay
  - 1 sink
- Arrays of Modules
  - `src[N]: Source;`

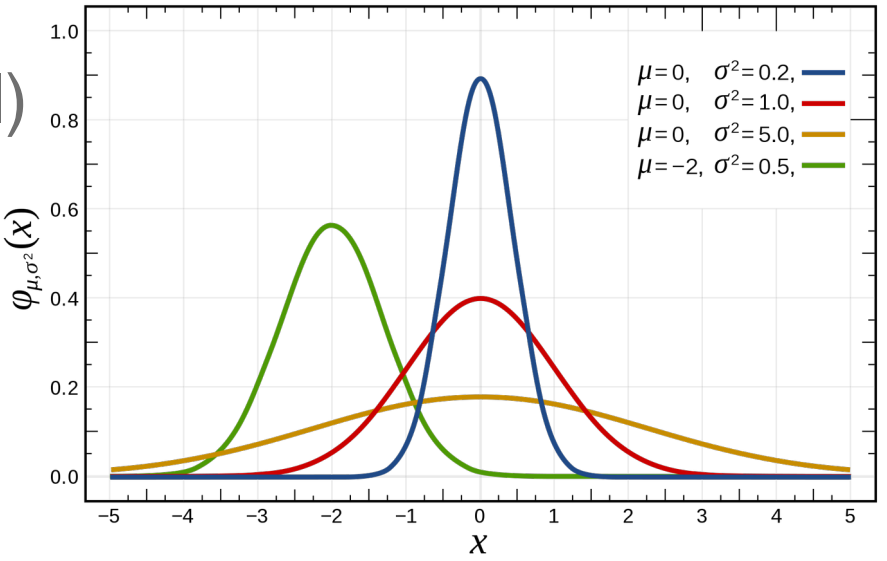


- New Delay Module
  - `import org.omnetpp.queueing.Delay;`
- Use of wildcards
  - `delay[*].delay=1s*truncnormal(1/mu, 0.1/mu);`

Exercises

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

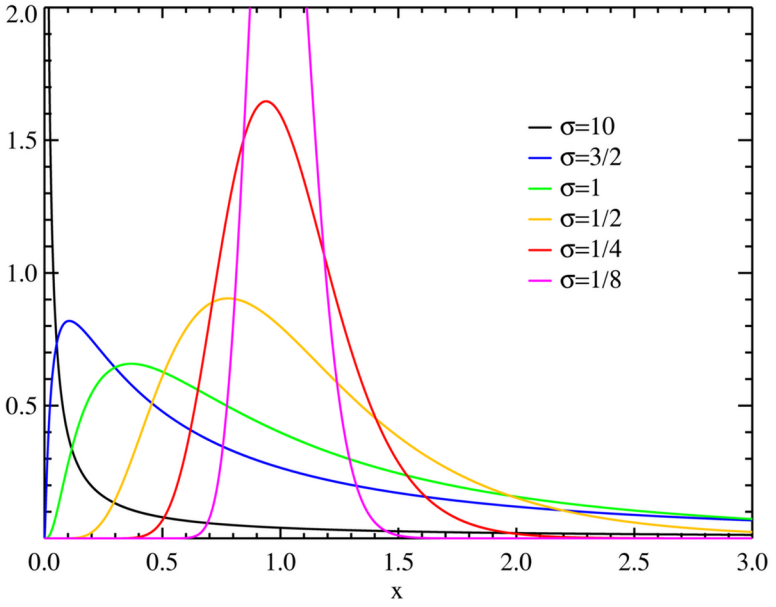
- Scenario: ρ=0.8
  - Medium-high utilization
- Delay is a Gaussian (truncnormal) distribution
  - Avg delay=Avg service time
  - Standard deviation= 10% average value
- Fog systems
  - Processing and network delay are comparable
  - Interesting scenario



Beyond exponential

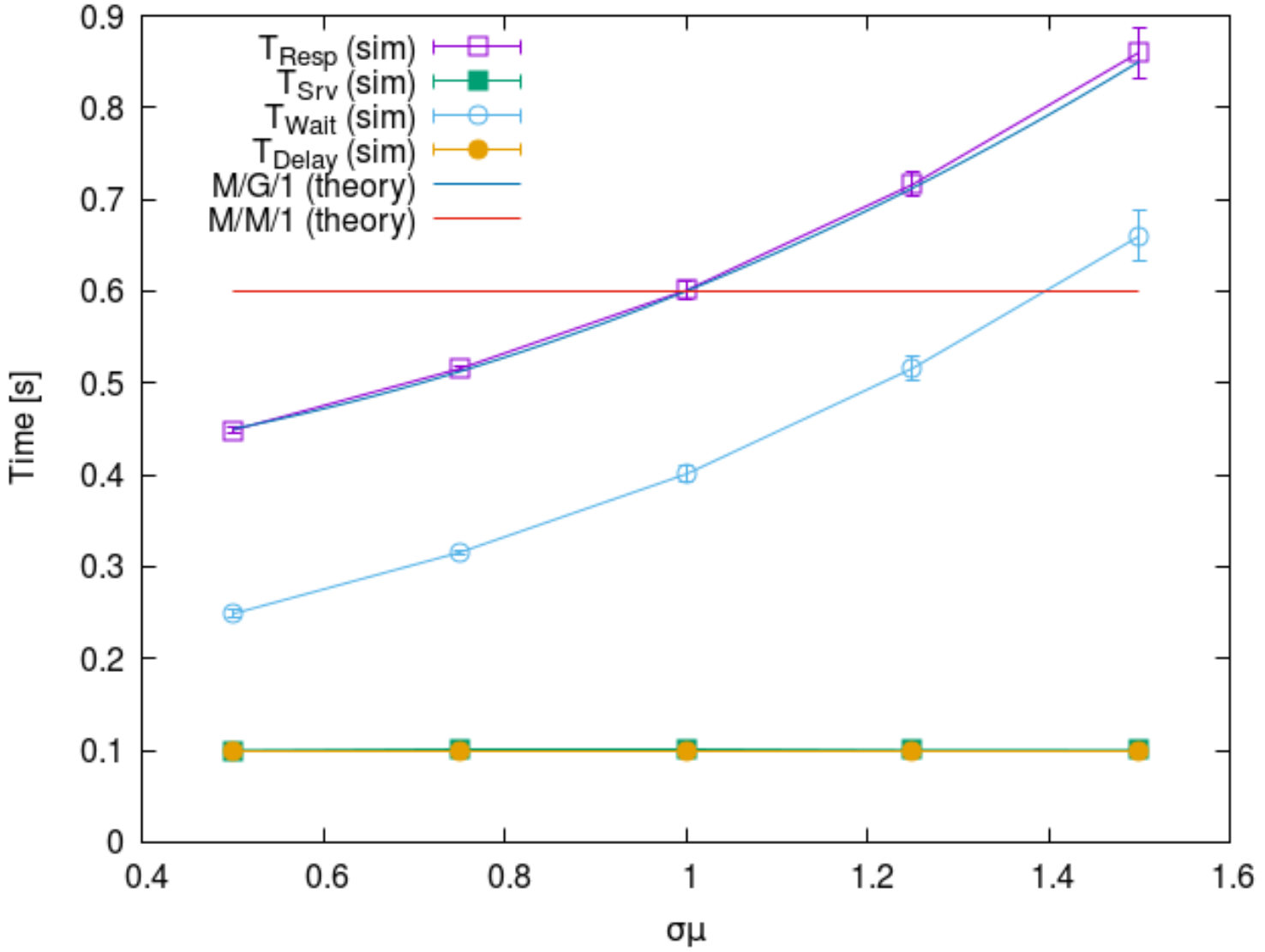
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Service time is a log-normal distribution
  - Common in computing systems
  - P(x)=lognorm(*m*, *w*) in .ned file
  - P(x) with average=*a* (=1/μ), std. dev.=*s*
  - → $m=\ln(a^2/\mathrm{sqrt}(a^2+s^2))$
    $=\ln(1/(\mu^2*\mathrm{sqrt}(1/\mu^2+s^2)))$
  - → $w=\mathrm{sqrt}(\ln(1+(s^2/a^2)))$
    $=\mathrm{sqrt}(\ln(1+(s^2*\mu^2)))$
- Interesting analysis as s changes



Some results

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA



A final case

Real systems

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Example of a realistic Fog systems
  - Sensors
  - Fog nodes
  - Cloud data center
- Mapping sensor→fog based on optimization problem
  - Heuristic solution
- Focus on representation
  - Placement of nodes according to coordinates
  - Colorized result
  - Map as background

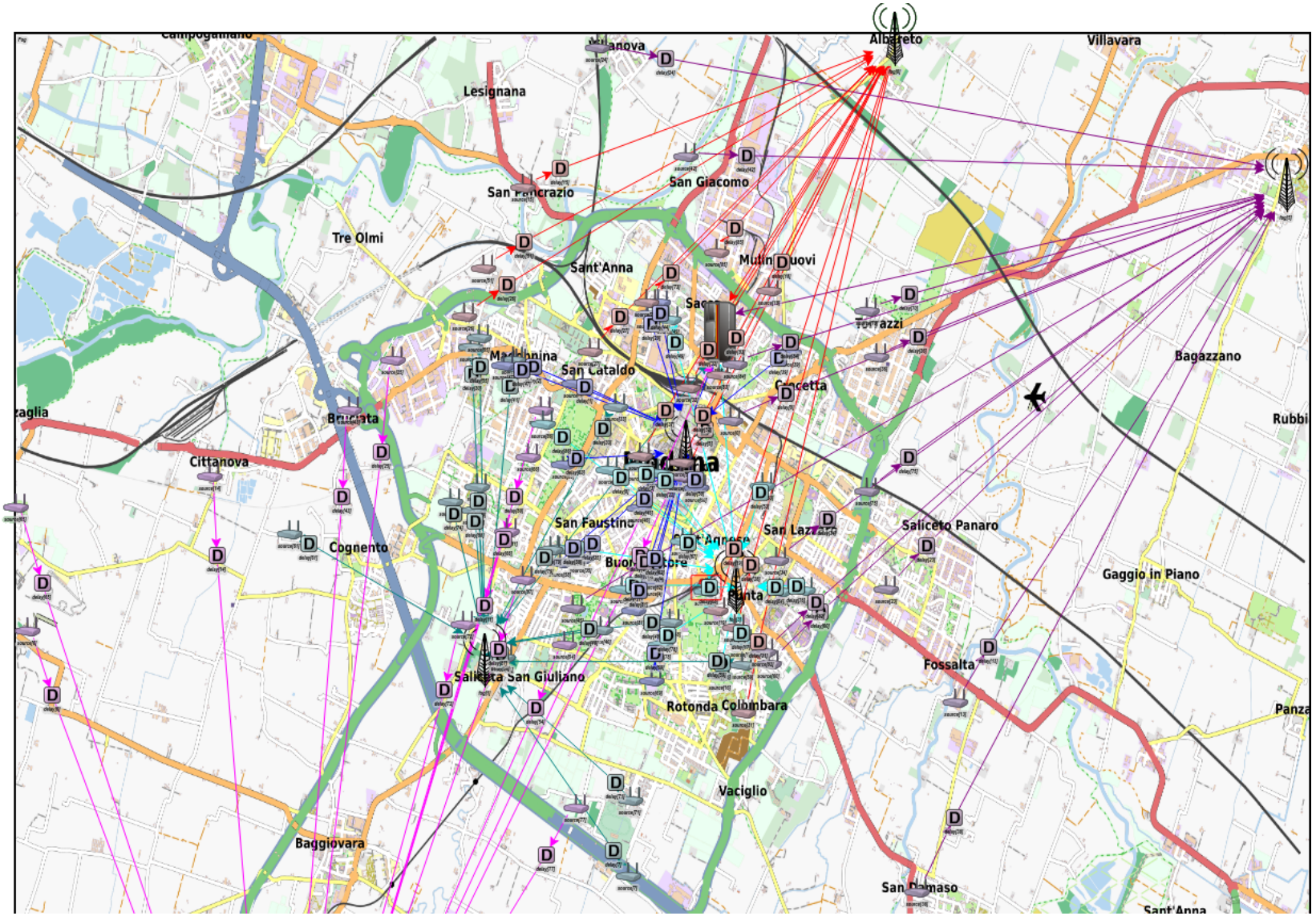
Real systems

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Definition of new models to support visualization
  - Add support for xpos, ypos, color attributes
  - Impact on Cloud, Fog and Sensor modules
- Describe service times
  - Multiple scenarios supported
- Add background map
  - Created from Open Street Map
  - File in images/maps
- Analysis for multiple workloads

Map representation

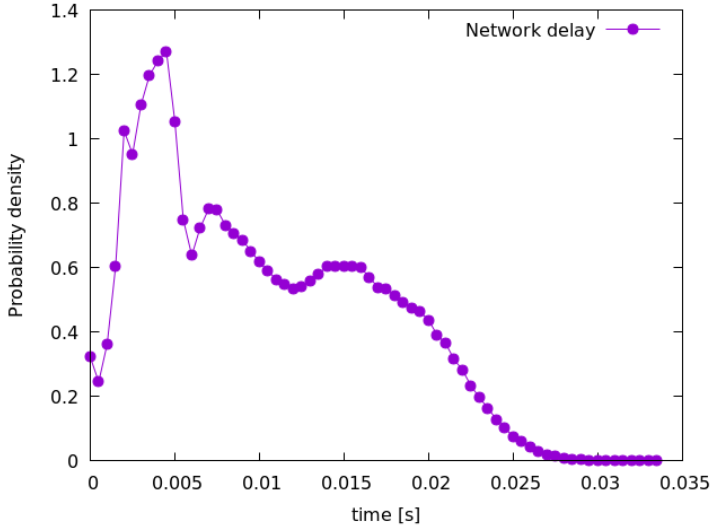
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

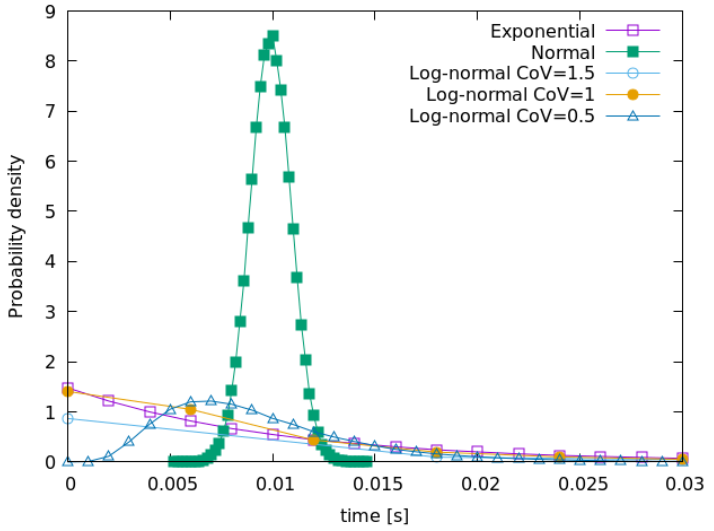


Some results

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Network delay based on topology
  - Implemented using delay modules
- Service time
  - Exponential
  - Gaussian
  - Log-normal

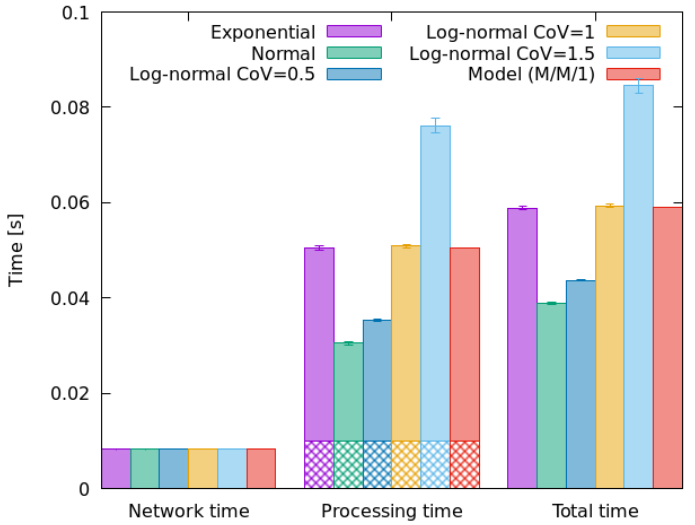


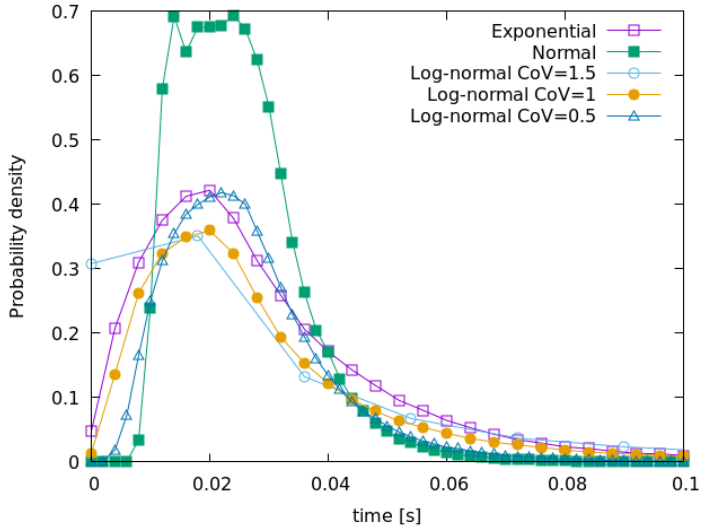


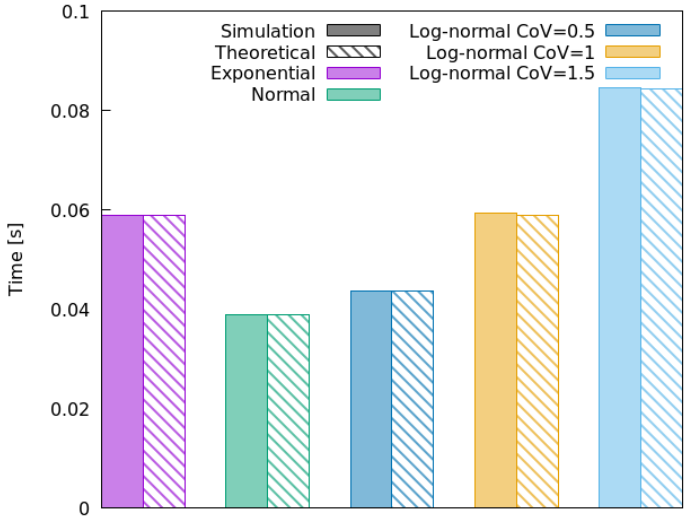
Some results

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Average resp. time
  - Breakdown
  - Fitting with models
- Histogram







Exercises for home

The problem of truncation

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Consider a M/G/1 system
- Execution time is a truncnormal distribution
  - Mean value = 1/μ
  - Std. Dev. = σ
- Inter-arrival time is λ
- Consider a scenario where
  - λ=8 req/s
  - μ=10 req/s
  - σ=[0.01, 0.05, 0.1, 0.5]
- Evaluate server utilization ρ as a function of σ

Load unbalancing

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Consider a M/M/1 system with 2 servers
- Incoming load is:
  - For server 1: $\lambda-\delta$, $\lambda=8$ req/s
  - For server 2: $\lambda+\delta$, $\lambda=8$ req/s
- Processing rate is μ for both servers
  - Evaluate impact of load difference δ
  - Repeat analysis for different average utilization $\rho=\lambda/\mu$

For the fearless...

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Development of new module
- Create load balancer
  - Accepts requests from multiple sources
  - Send requests to multiple outputs
- Load balancing of outgoing requests
  - Round robin
  - Random
- Application to previous topology