Software and Hardware Tools Used in the SAN Group

SAN PhDs and PostDocs

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Outline

- Milosh: Contiki, Cooja, IoT-Lab
- Jingyue: SimEvents Toolbox
- Aleksandra: KOALA, SPARK, SWIM
- Hrishikesh: Heracles
- Mike: Grasp, Virtual Sensors
- Ehsan: Weka, RapidMiner
- Martijn: Tool for evaluating scheduling analysis

My workflow (Milosh)

- **Develop in Contiki**
- **Debug in Cooja/MSPSim**
- **Simulate in Cooja**
- **Evaluate on FIT IoT-Lab test bed**

Contiki: Open-source OS for IoT

- **Low memory footprint (standard C)**
- **Implements many standards**
- **Good hardware support**
- **Backed by industry and academia**

Contiki: usage

- **Provides full low-power IP networking**
  - **6LowPAN, RPL, CoAP, MQTT**
- **Software energy estimation**
- **Easy to prototype and evaluate networking protocols**

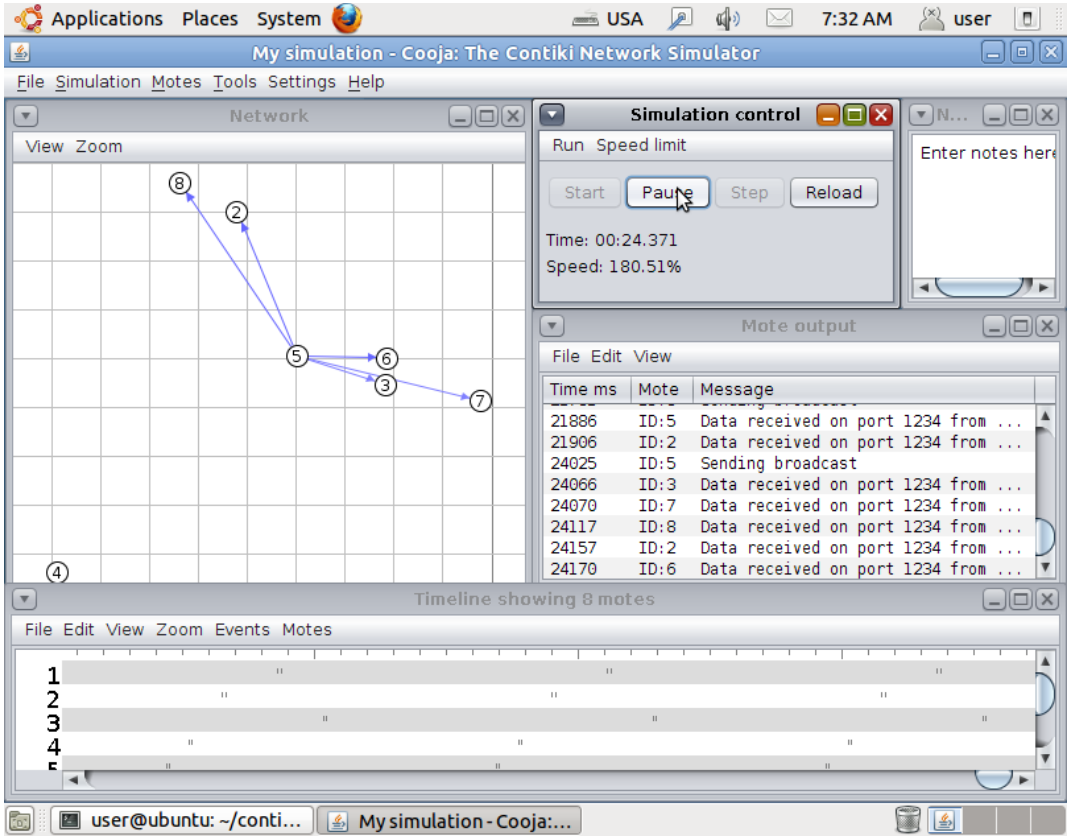
Contiki: availability

- **Homepage: http://www.contiki-os.org/**
- **Source code: https://github.com/contiki-os/contiki/**
- **Virtual Machine (Instant Contiki): http://goo.gl/ozkUfb**
- **Tutorials: https://github.com/contiki-os/contiki/wiki**
- **Mailing list: http://goo.gl/8UEqJq**

Cooja: Cross-layer simulator

- Part of Contiki distribution
- Device emulator (only MSP430, ARM under way)
- Network simulator
- GUI and command line support
- Can log and inspect radio traces

Cooja: how it looks like



Cooja: usage

- **Fast debugging (MSPSim)**
- **Explore parameter space**

FIT IoT-Lab: large scale open testbed

- 2700 sensor nodes in 6 sites
  - MSP430, STM32 (Cortex-A3), Cortex-A8
  - 800 MHz & 2.4 GHz radios (802.15.4)
- Synthetic test bed
- Support for Contiki, FreeRTOS, Riot, TinyOS, Linux, OpenWSN

FIT IoT-Lab: how it looks like

Lille Test-Bed (https://www.iot-lab.info/deployment/lille/)

FIT IoT-Lab: usage

- **Evaluate networking protocols in real-life**
- **Link-layer assessment on radio traffic**
- **Provides tools for measuring energy consumption**

FIT IoT-Lab: availability

- **Homepage: https://www.iot-lab.info/**
- **Tools: https://github.com/iot-lab/iot-lab**
- **Tutorials: https://github.com/iot-lab/iot-lab/wiki**

Introduction of SimEvents Toolbox

Jingyue Cao

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Introduction of SimEvents Toolbox

- Simulink: SimEvents Toolbox
- Key Feature: Discrete Event Simulation
- Model Example:
  - Manufacturing
  - Controls
  - Network Communications
  - System Architecture
  - General Applications

SimEvents Libraries

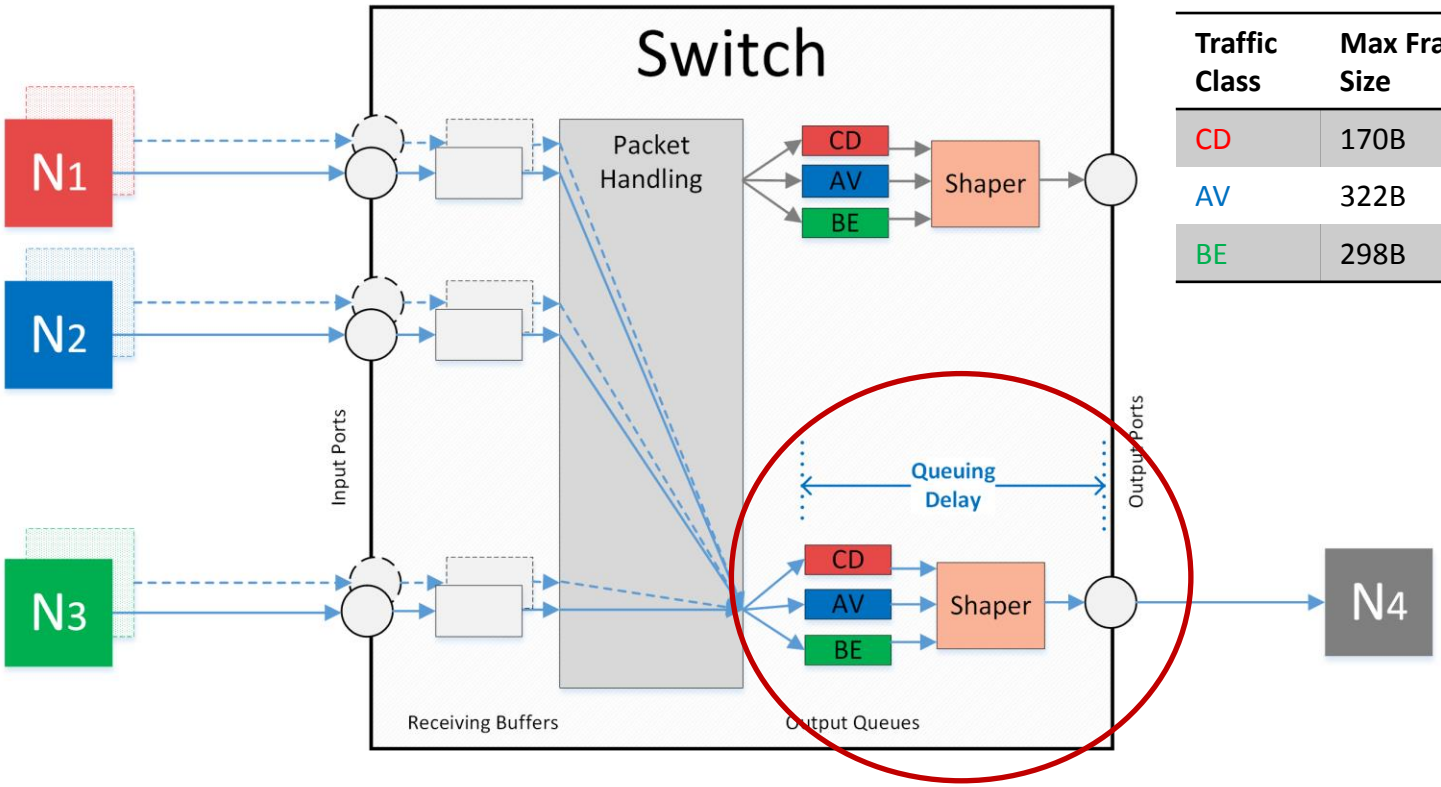
- Main Functionalities

Term	Example: Network Communication	Relevant Libraries
Entity	Ethernet Packet	Generators Sinks
Attribute	Timing, Destination/Source Address, Size, Class/Priority, ID, etc.	Attributes (Set/Get)
Events (Processing of Entities)	Queuing, Routing, Prioritization, Selection, Transmission, etc.	Entity Management (Combiner/Splitter) Queues (FIFO/LIFO/Priority) Routing (Input/Output Switch) Gates Servers

- Other Functionalities
  - Integration SimEvents with Simulink: Gateways(time-based <-> event-based)
  - Integration SimEvents with Matlab workspace: Sinks(to workspace)
  - Observations: Sinks(attribute/signal scope)

Example: Network Communication (1)

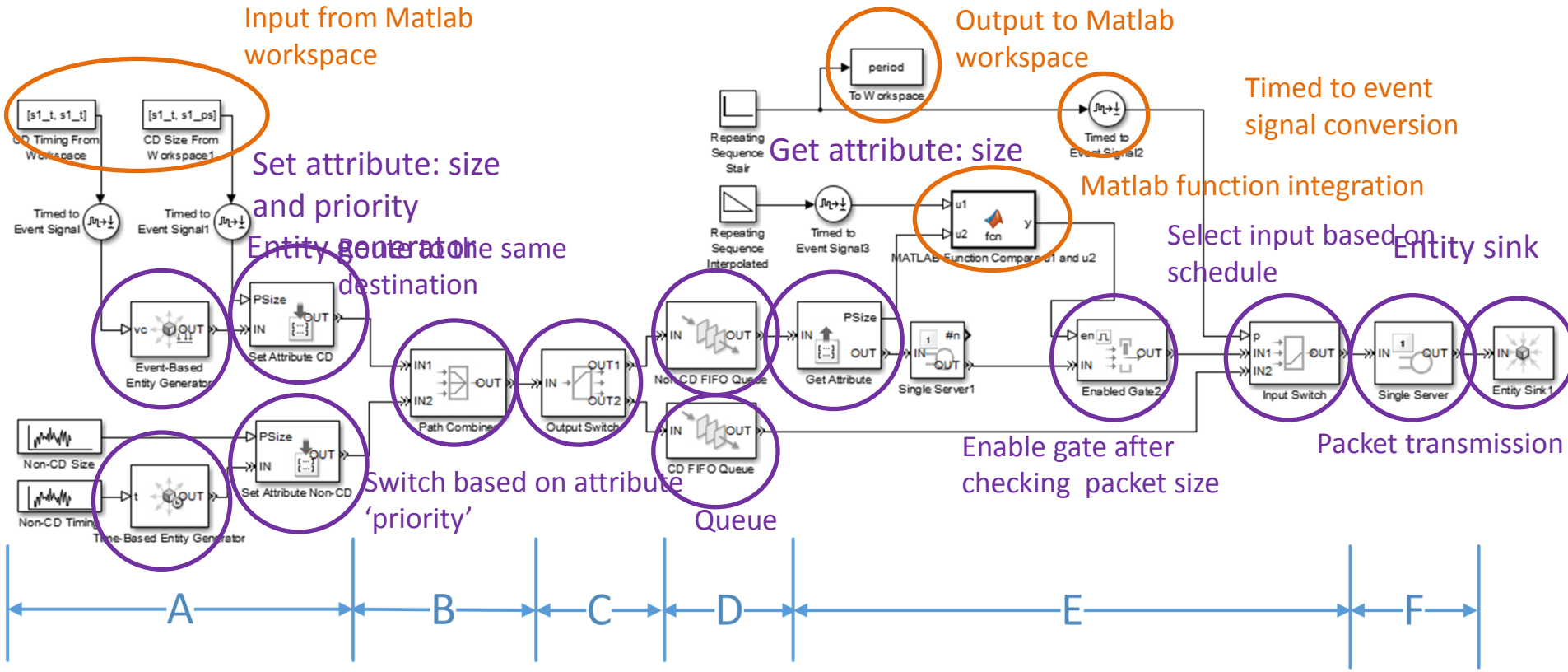
- Switched Ethernet
- Packets: CD (Control Data), AV (Audio/Video), BE (Best Effort)
- Shapers: Packet selection at the output ports



Traffic Load

Traffic Class	Max Frame Size	Transmission Period
CD	170B	500us
AV	322B	125us
BE	298B	#

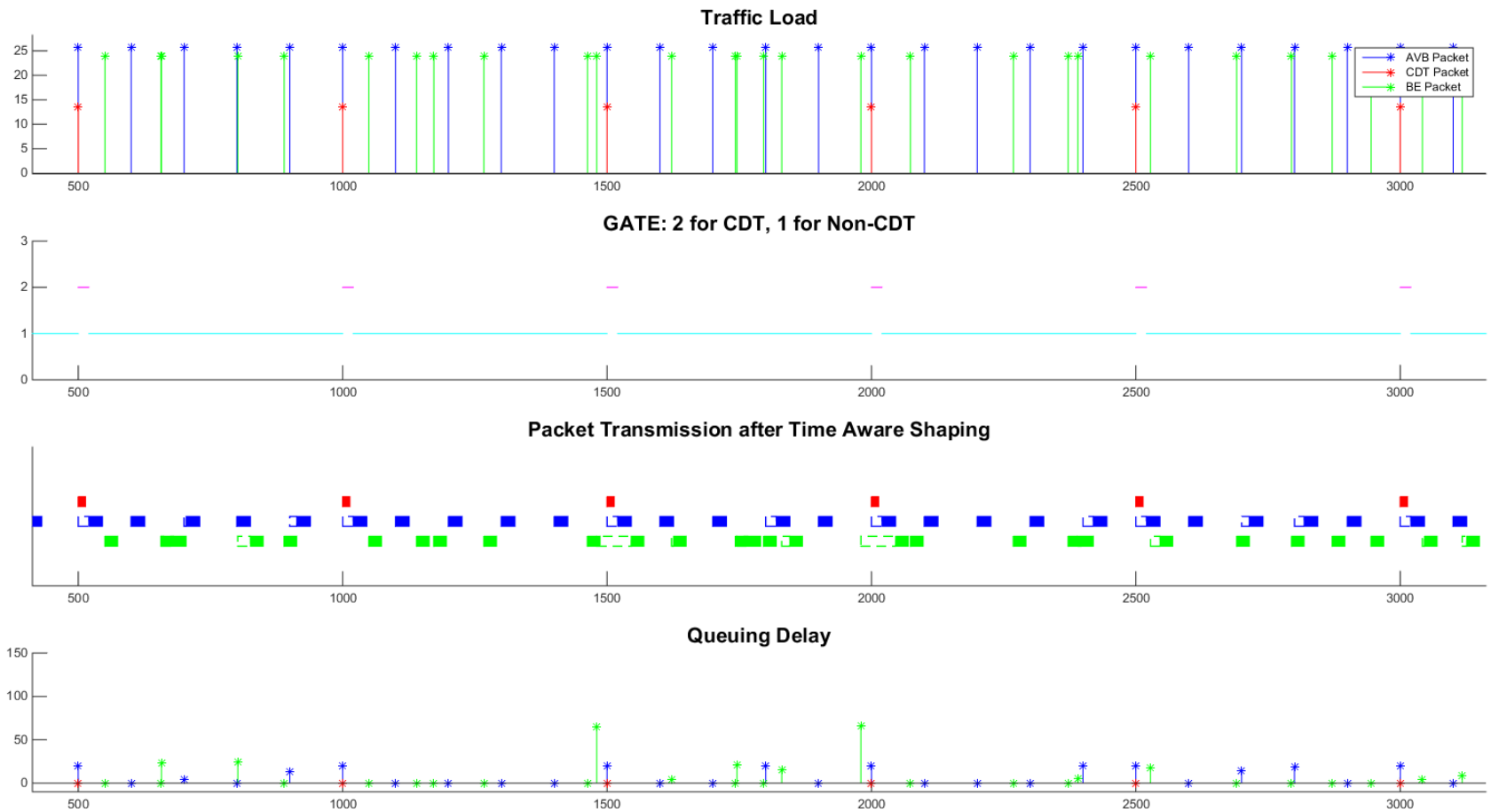
Example: Network Communication (2)



A: Packet generation
B: Packet routing
C: Packet classification
D: Packet queuing
E: Packet selection
F: Packet transmission

Example: Network Communication (3)

Graphic Representation of Simulation Result



Software and Hardware Tools Used in the SAN Group

Aleksandra Kuzmanovska

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Tools

- KOALA resource manager
- SPARK framework
- SWIM workload generator

KOALA

- Resource manager for cluster/grid/cloud
- Support various application types
  - single parallel applications, workflows, frameworks
  - provisioning across cluster and cloud environments
- Deployed on DAS4
- Usage : active development
  - Scheduling heterogeneous frameworks
  - Provisioning policies

KOALA availability

- More information : http://www.pds.ewi.tudelft.nl/koala/
- Use KOALA on DAS4 :
  - request DAS-4 account: das-account@cs.vu.nl
- Download KOALA :
  - private repository: https://svn.st.ewi.tudelft.nl/koala/
  - koala@st.ewi.tudelft.nl

SPARK

- General purpose execution engine with distributed in-memory data storage
- Support variety of applications
  - Machine learning (iterative MapReduce)
  - SQL and structured data processing
  - Graph and stream processing
- Usage: one of the frameworks scheduled with KOALA

SPARK availability

- **More information : https://spark.apache.org/**
- **Download SPARK : git://github.com/apache/spark.git**
- **Mailing lists : {user, dev}@spark.apache.org**

SWIM : Statistical workload injector for MapReduce systems

- Suites of workloads
  - real life workloads from production systems (MapReduce)
  - thousands of jobs,
  - complex data, arrival, and computation patterns
- Generate representative workloads
  - sampling historical MapReduce cluster traces
- Execute workloads with low performance overhead
- Usage: to generate Hadoop workloads

SWIM availability

- **More information:**

  https://github.com/SWIMProjectUCB/SWIM/wiki

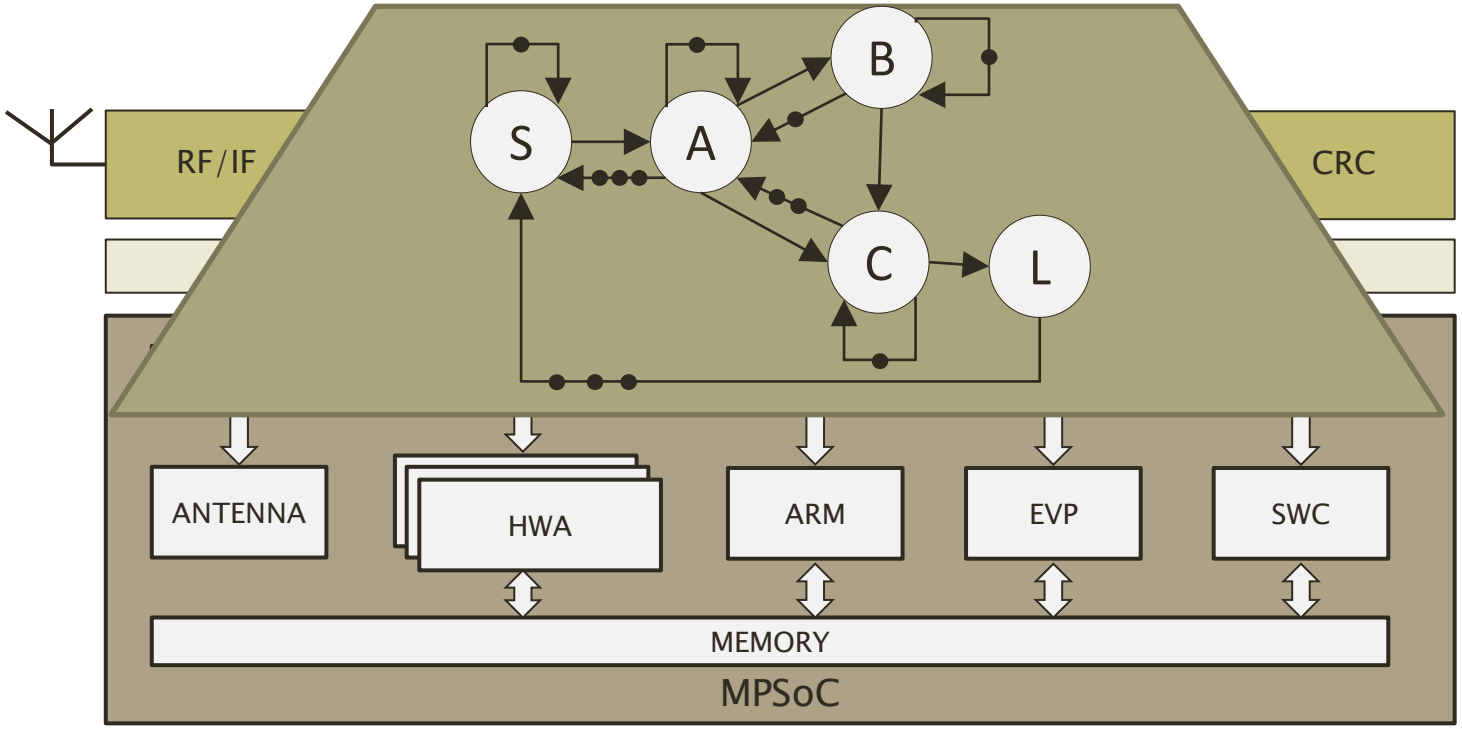
- **Download SWIM:**

  https://github.com/SWIMProjectUCB/SWIM.git

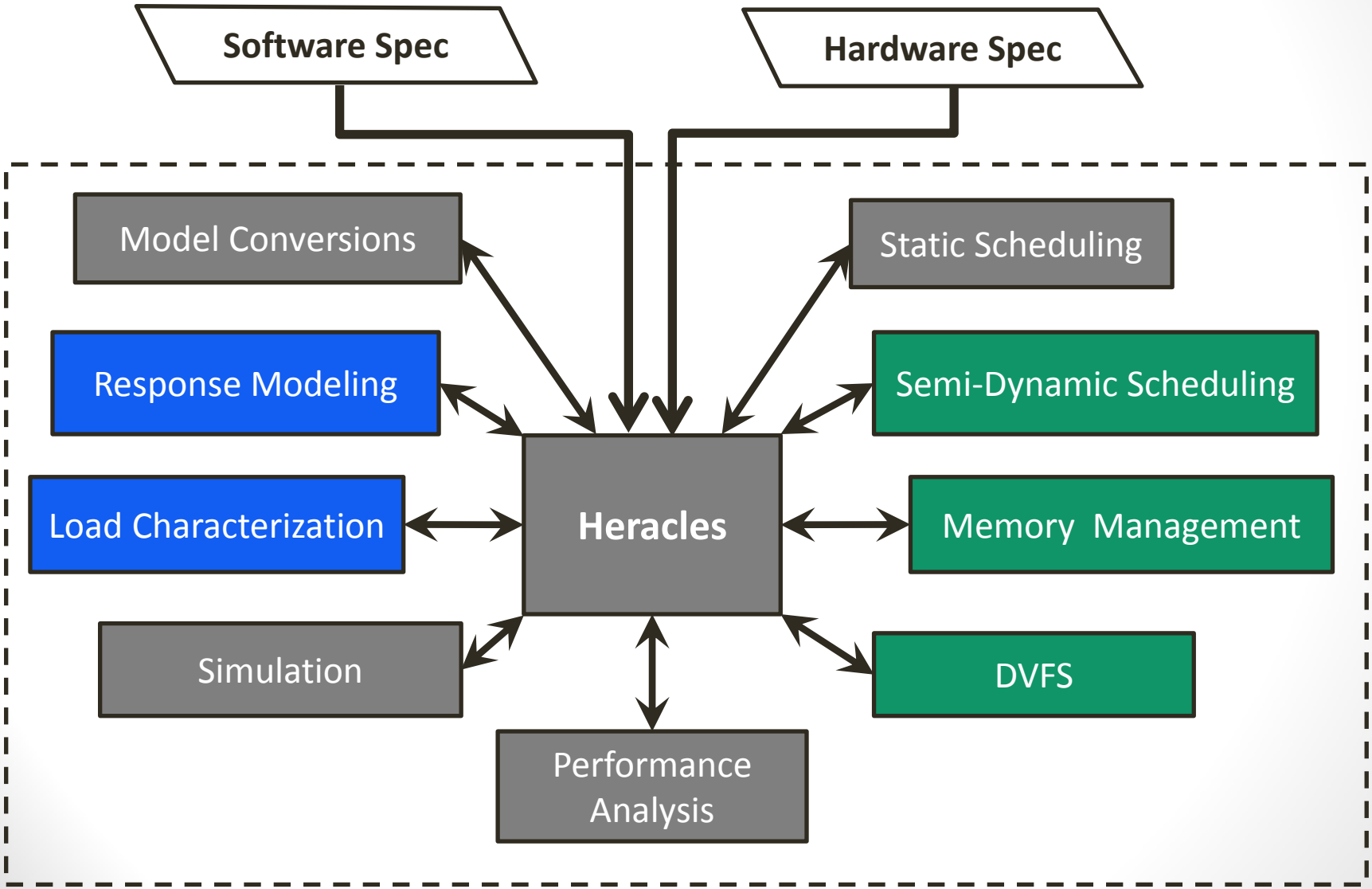
Heracles

The Hero of Dataflow Analysis

Software Defined Radio



Heracles Tool



Resources

- Source Code: will be made available soon…

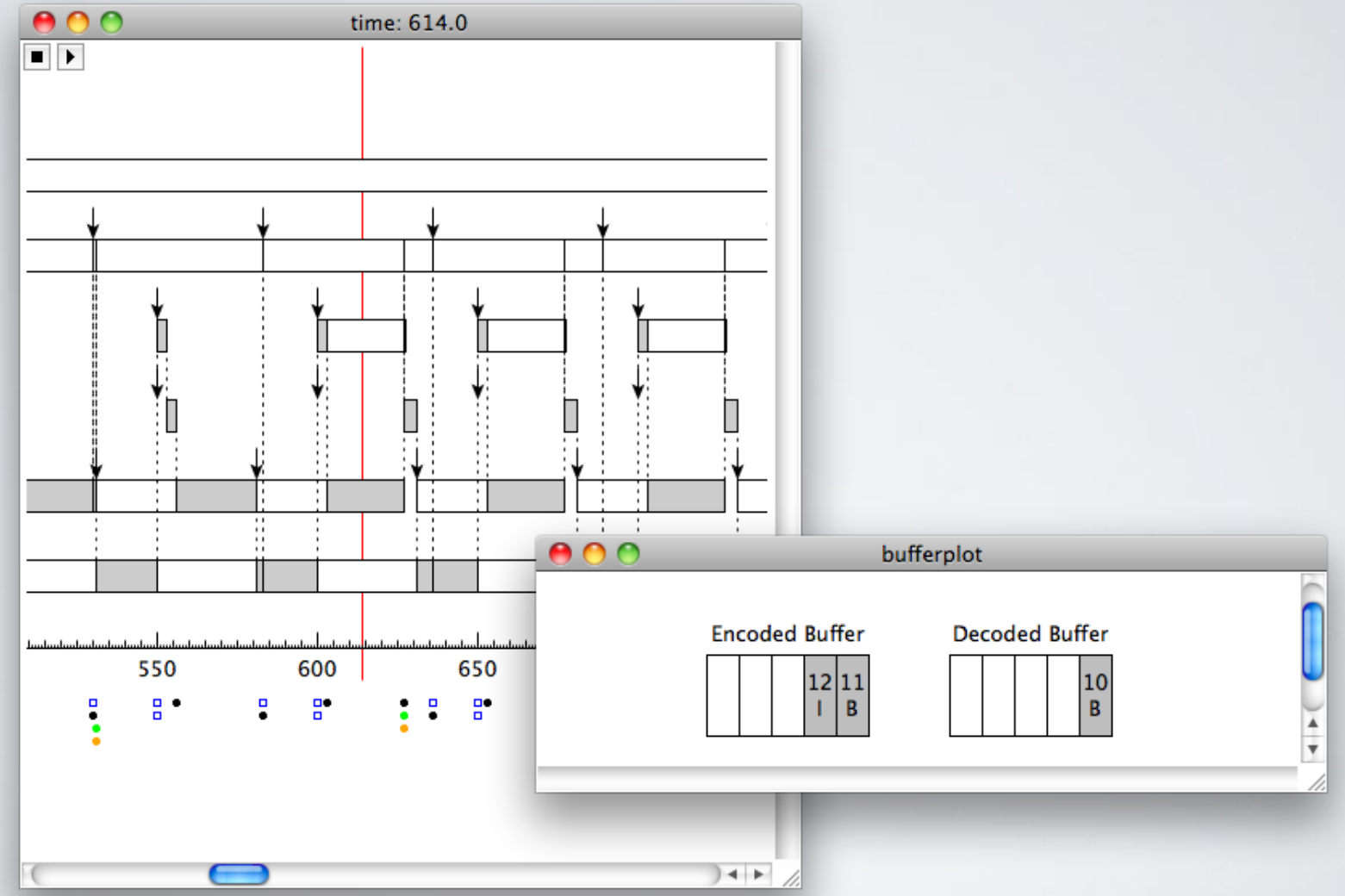
Grasp: trace visualization

Available online:

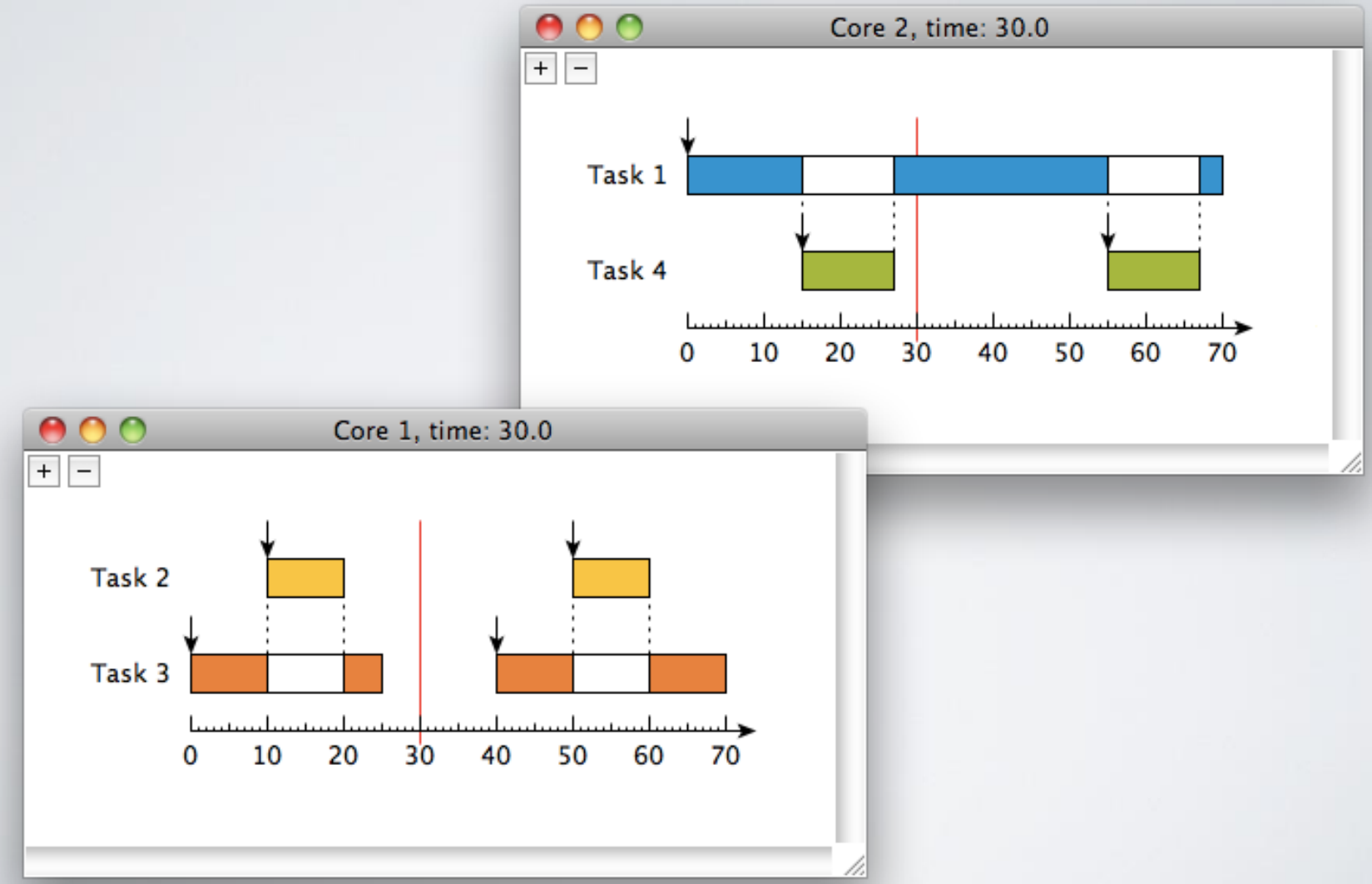
http://www.win.tue.nl/~mholende/grasp

for Linux, Mac, Windows

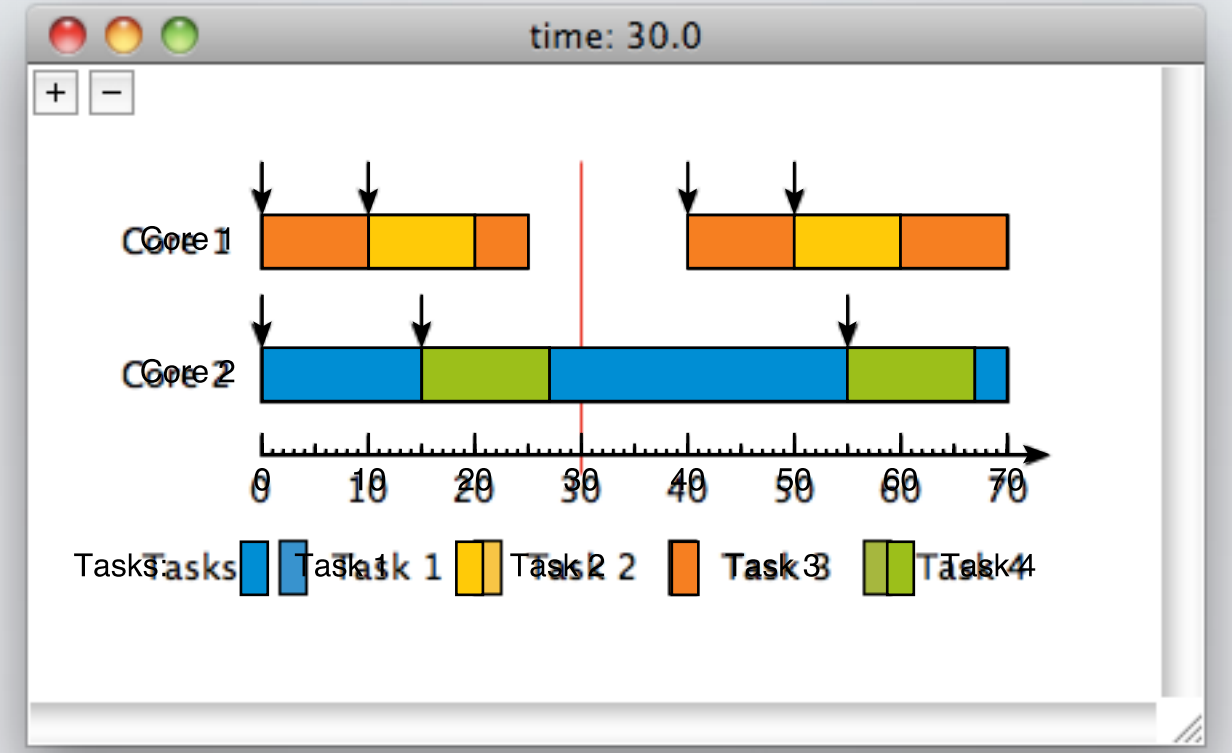
Example


time: 614.0
550
600
650
bufferplot
Encoded Buffer
12
I
11
B
Decoded Buffer
10
B

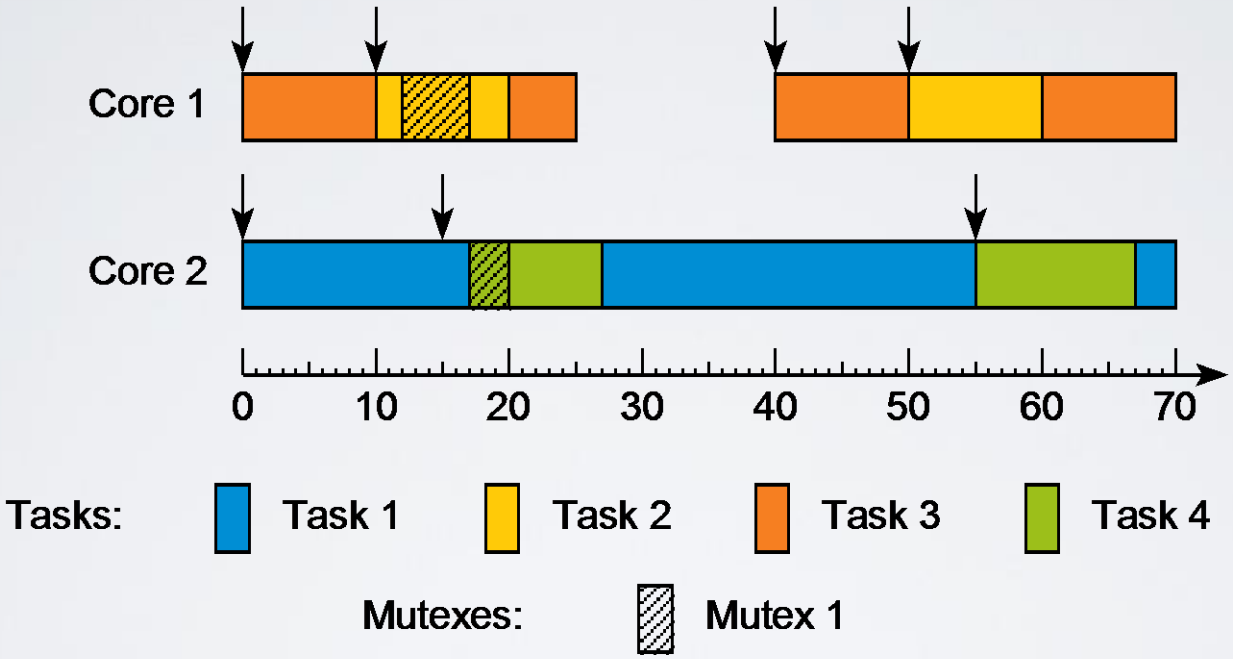
Example


Core 2, time: 30.0
Task 1
Task 4
0
10
20
30
40
50
60
70
Core 1, time: 30.0
Task 2
Task 3
0
10
20
30
40
50
60
70

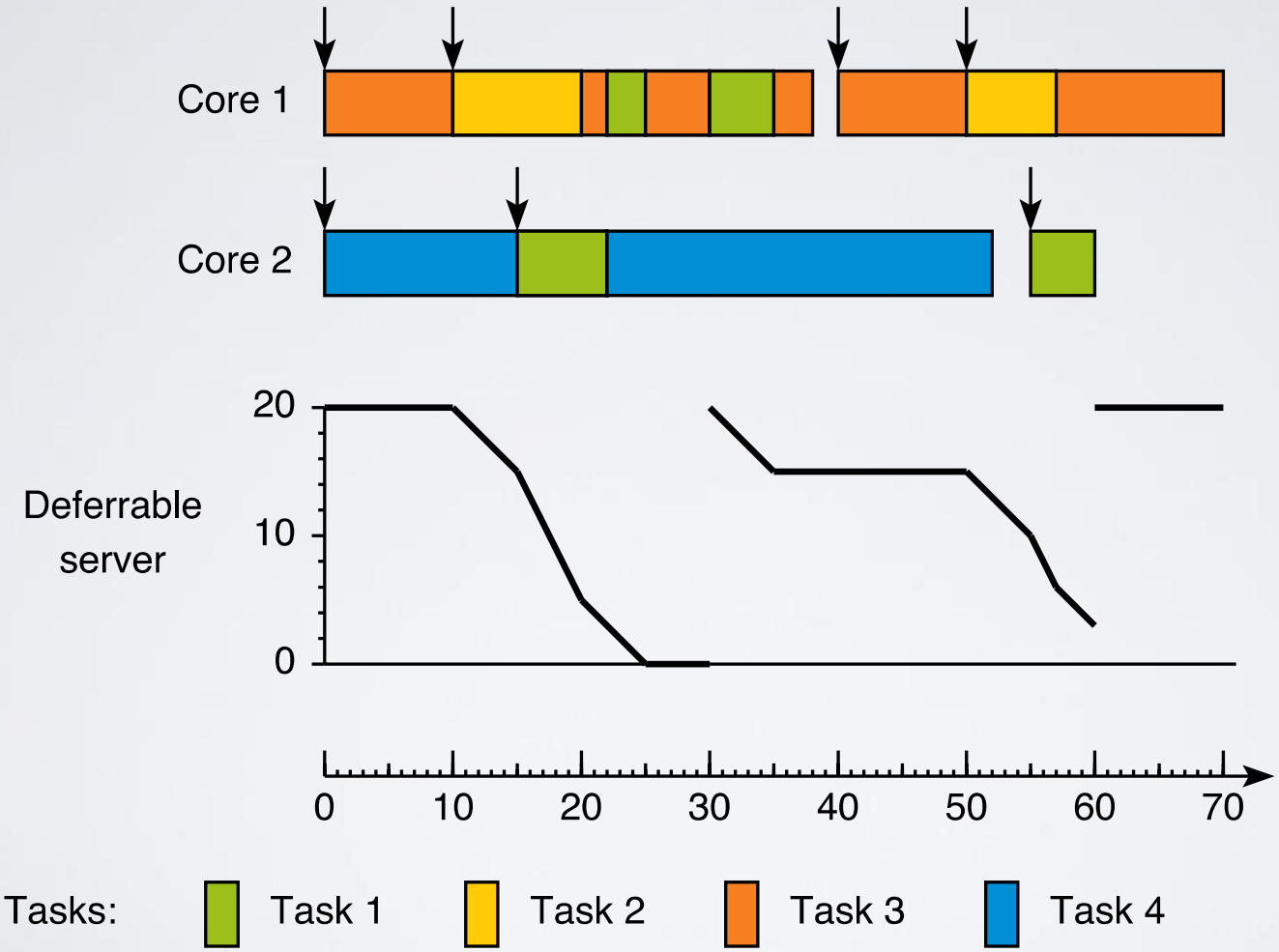
Example


time: 30.0
+
−
Core 1
Core 2
0
10
20
30
40
50
60
70
Tasks:
Task 1
Task 2
Task 3
Task 4

Example

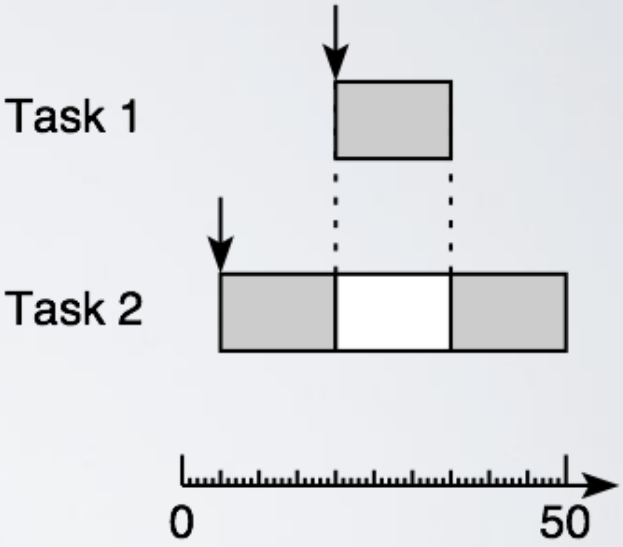

Core 1
Core 2
0
10
20
30
40
50
60
70
Tasks:
Task 1
Task 2
Task 3
Task 4
Mutexes:
Mutex 1

Example


Core 1
Core 2
Deferrable server
20
10
0
0
10
20
30
40
50
60
70
Tasks:
Task 1
Task 2
Task 3
Task 4

Grasp trace example

```
newTask task1 -priority 7 -name "Task 1"
newTask task2 -priority 8 -name "Task 2"
plot 5 jobArrived job2.1 task2
plot 5 jobResumed job2.1
plot 20 jobArrived job1.1 task1
plot 20 jobPreempted job2.1 -target job1.1
plot 20 jobResumed job1.1
plot 35 jobCompleted job1.1 -target job2.1
plot 35 jobResumed job2.1
plot 50 jobCompleted job2.1
```



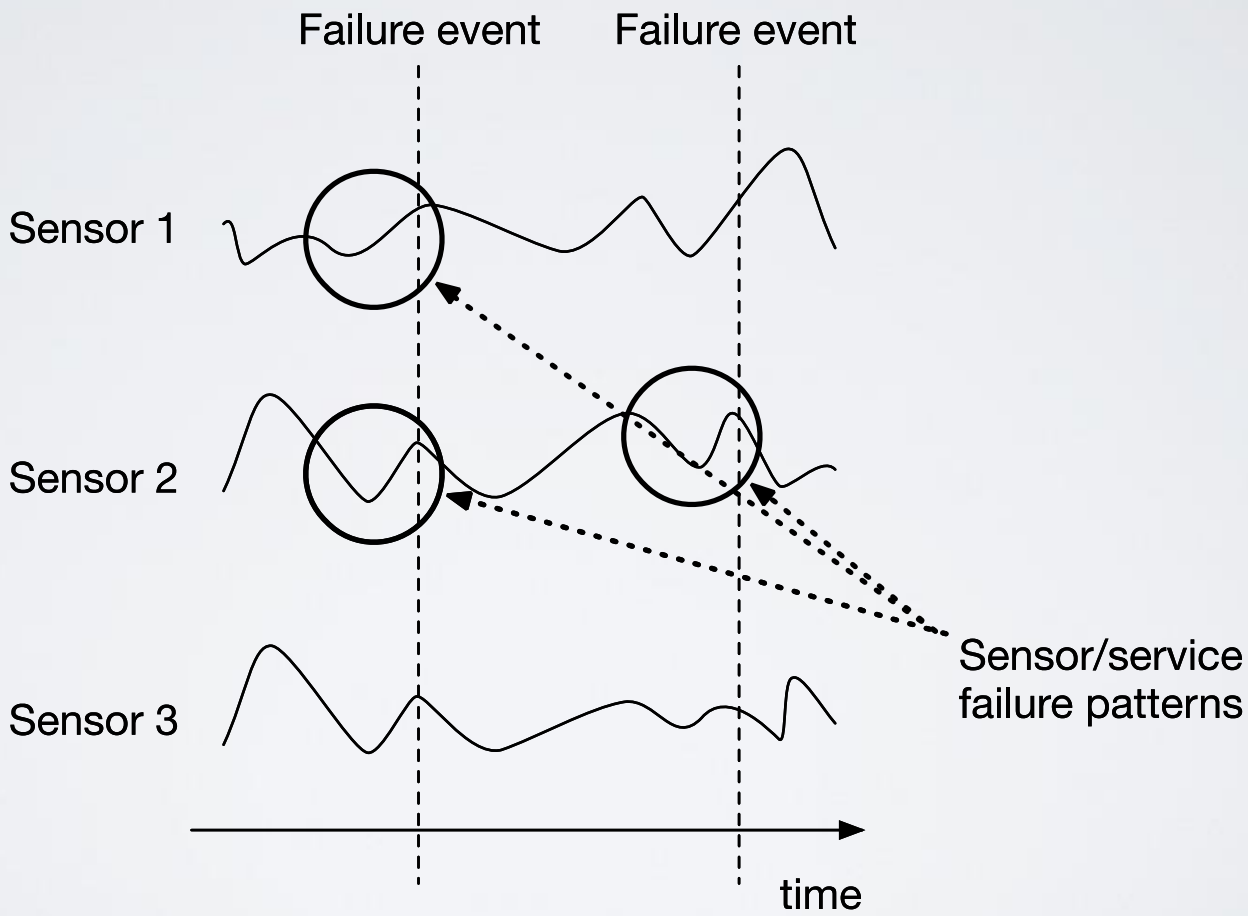
Virtual Sensors

Available online:

http://www.win.tue.nl/~mholende/sensors

for Linux, Mac, Windows

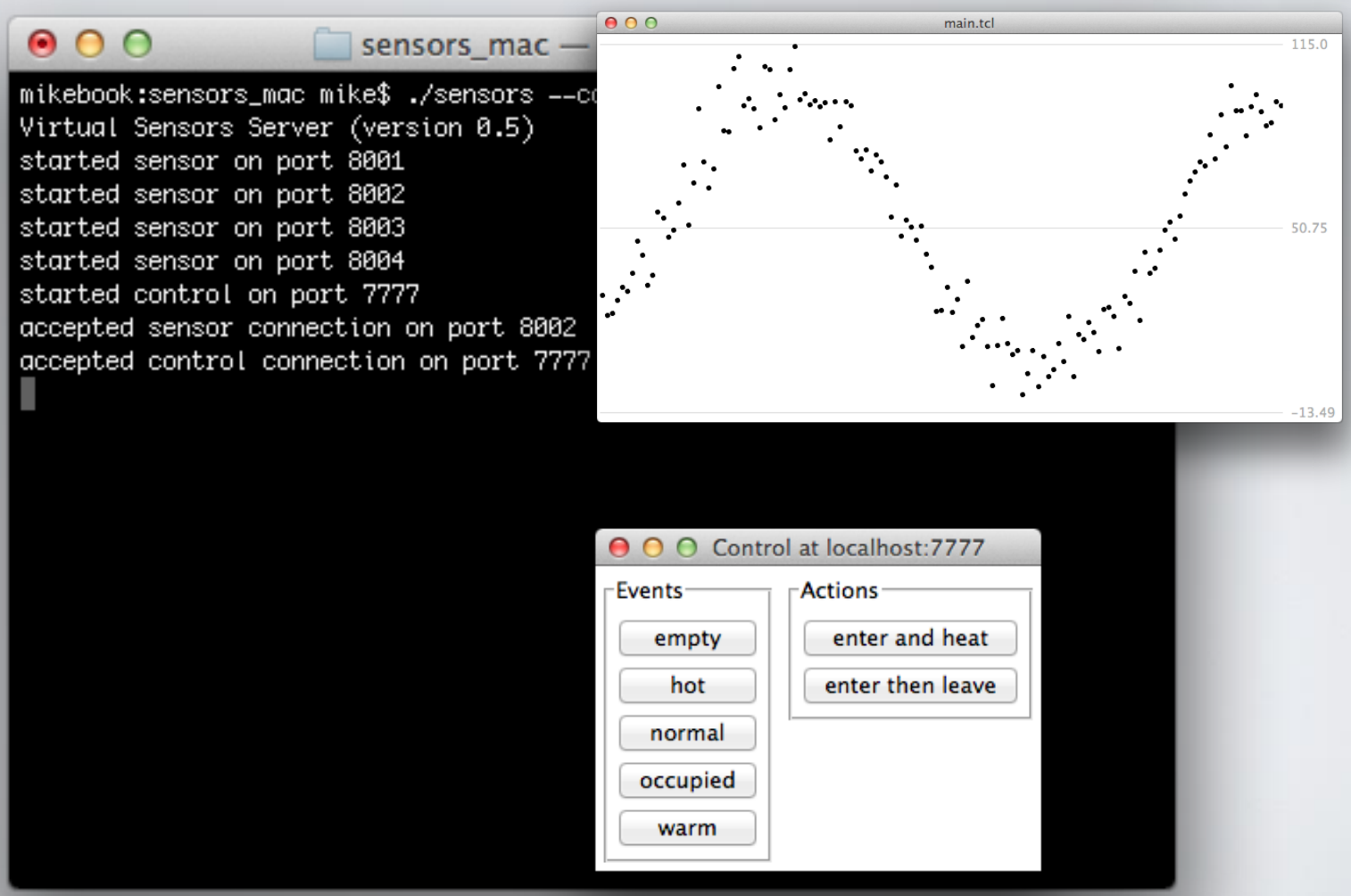
Project: Mantis



Reasons for synthetic data

- Gain better **insights** into the performance of data analytics algorithms
- Avoid labeling **unlabeled** data sets
- **Evaluate** and **demonstrate** data analytics implementations

Virtual sensors



Virtual sensors

- What's provided:
  - Easy way for **specifying sensor behavior**
  - Infrastructure for **streaming sensor data** and **controlling sensor behavior**

- What's missing:
  - Realistic sensor models

WEKA & RapidMiner
Data Mining/Machine Learning Tools

Ehsan Ullah Warriach

SAN Seminar

System Architecture and Networking Group

SAN

Technische Universiteit Eindhoven University of Technology

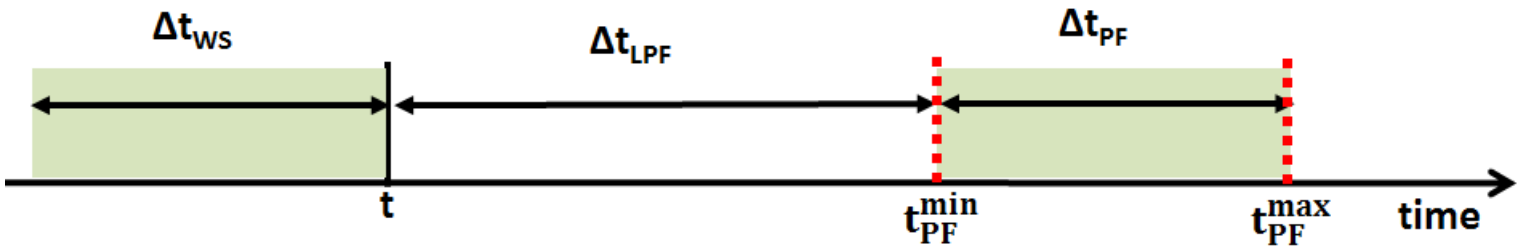
Where innovation starts

WEKA (Waikato Environment for Knowledge Analysis) & RapidMiner

- A collection of machine learning algorithms
  - Logistic regression, decision tree, neural networks, support vector machines, Bayes' nets, k-means, etc.;
    - Classification – arranged into predefined classes,
    - Regression – to find a function that models the data with least error,
    - Clustering – similar to classification but classes are not predefined,
    - Association rule learning – search for relationship.
- Data preprocessing and visualization
  - Discretization, normalization, resampling, attribute selection, transforming, and combining attributes.
- WEKA supports 100 operators, RapidMiner supports 500 operators.
- Java based.
- GUI interfaces.
- Data can be imported from a file in various formats:
  - WEKA → ARFF, CSV. RapidMiner → 15-18 different file formats.

Time series analysis

- Time series analysis is the process of using machine learning algorithms to model a time-dependent series of data points;
  - Predictions are made using generated model for future events based on known past events.



- Comparison of machine learning algorithms:
  - multiple linear regression (MLR)
  - artificial neural networks (ANNs)

Research Problems

- WEKA
  - Time series data analysis (SAN)
    - Fault prediction models
      - predict remaining battery lifetime of the MyriaNed (WSN platform) using real traces of battery consumption.
  - Intelligent Lighting (Aravind - SAN)
    - Missing data treatments.
- RapidMiner
  - Fault detection (DS-RUG)
    - Outlier, spikes.
  - Activity recognition (DS-RUG)
    - Presence, absence, meeting, working with PC, working without PC.
  - Vehicle types classification (DS-RUG)

WEKA resources

- WEKA website: http://www.cs.waikato.ac.nz/~ml/weka/index.html
- WEKA Data Mining Books:
  - Ian H. Witten and Eibe Frank, Data Mining: Practical Machine Learning Tools and Techniques (Second Edition).
  - Jiawei Han and Micheline Kamber, Data Mining: Concepts and Techniques, 2nd ed.
- WEKA Wiki: http://weka.sourceforge.net/wiki/index.php/Main_Page.
- Download
  - http://www.cs.waikato.ac.nz/ml/weka/downloading.html
  - WEKA (>= 3.7.3) - time series analysis environment.
- Support multiple platforms (written in java):
  - Windows, Mac OS X and Linux.
- Online tutorials
  - For beginners: https://www.youtube.com/watch?v=m7kpIBGEdkI

RapidMiner resources

- RapidMiner website: https://rapidminer.com/
- Download
  - An individual can manage online account to download RapidMiner (software license keys),
  - RapidMiner (>=5.3) - time series analysis environment.
- Support multiple platforms (written in java)
  - Windows, Mac OS X and Linux.

Technische Universiteit
Eindhoven
University of Technology

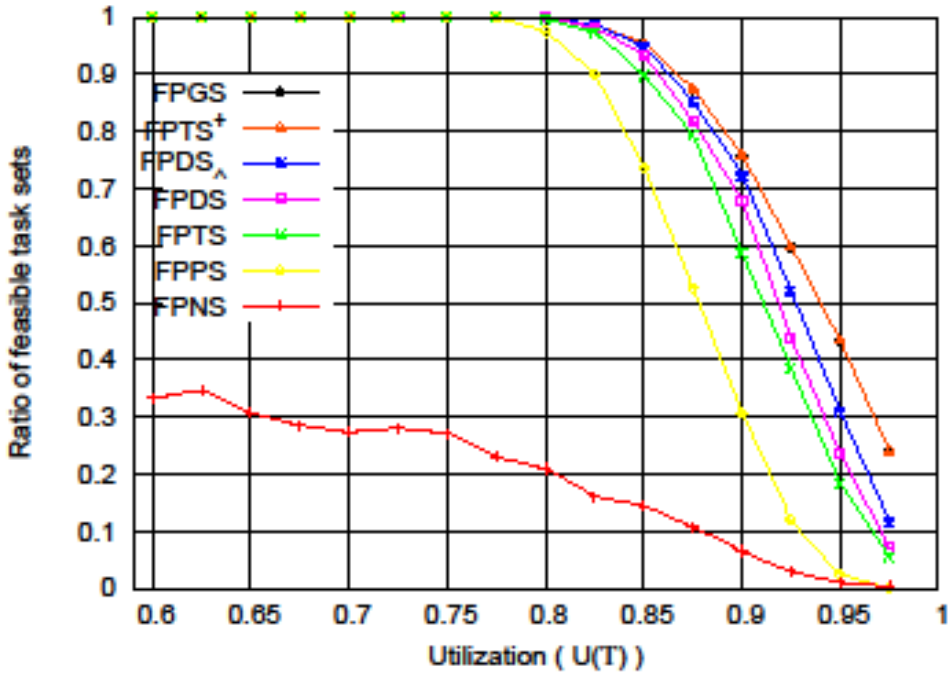
SAN

Tools for the evaluation of scheduling analysis

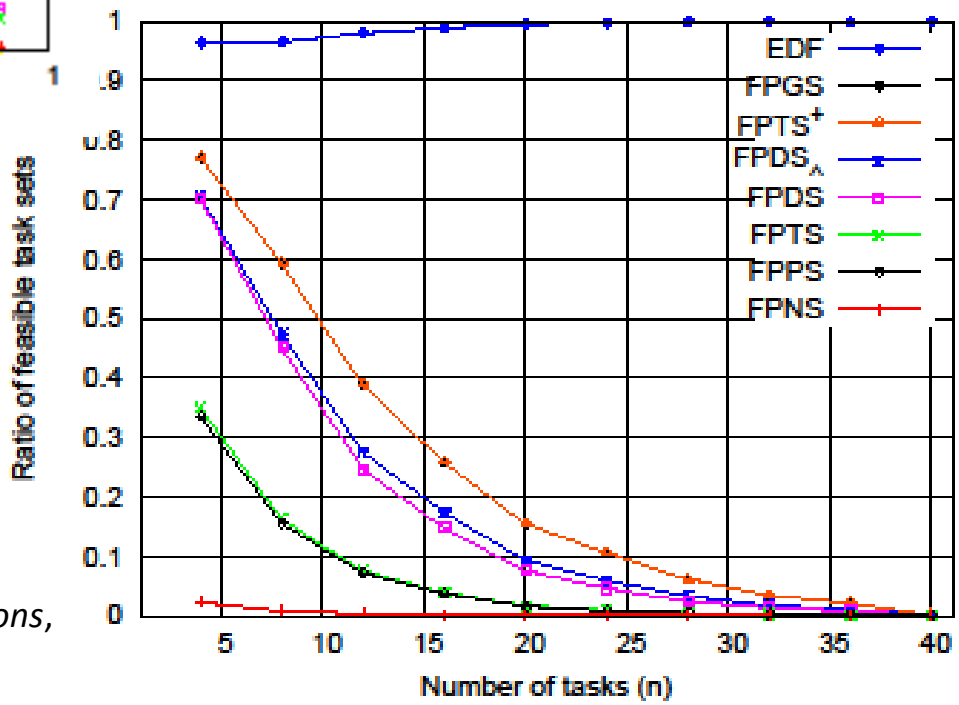
Martijn van den Heuvel

Systems Architecture and Networking (SAN) group
Department of Mathematics and Computer Science
Eindhoven University of Technology

Example evaluation



- **Generate random samples**
- Perform scheduling tests



Source:
R.J. Bril, M.M.H.P. van den Heuvel, U. Keskin, J.J. Lukkien, *Generalized fixed-priority scheduling with limited preemptions*, ECRTS, pp. 209--220, July 2012

Tools

- GnuPlot:
  - opensource data plotting (Win/Linux/Mac)
  - scalable graphics
  - automated plots through scripts
  - http://www.gnuplot.info/

- Generate random samples:
  - **Uunifast algorithm:**

    E. Bini and G. Buttazzo,
    "Biasing effects in schedulability measures,"
    in ECRTS, July 2004, pp. 196–203

  - http://retis.sssup.it/~bini/resources/matlab/UUniFast.m

Generate random samples

- Generate n values adding up to value U:

$$U = \sum_{i=1}^{n} x_i$$

Example:

– Processor utilization is U = 0.8;

– Total of n = 10 tasks.

Random output:

values $x_1 \ldots x_{10}$

Plot:

– Perform test

– Repeat many times

– Count successes