

TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers

Pablo Abad, Pablo Prieto, Lucia Menezo, Adrian
Colaso, Valentin Puente, Jose-Angel Gregorio

University of Cantabria



Interconnects Research: Simulation Tool

What makes a Simulation Tool better than others?

Flexibility

- Heterogeneous field, from supercomputers to CMP.
- Highly Configurable



Accuracy Vs. Comp. Effort

- Avoid slow simulations for first stages of research process.
- But provide accurate enough results at last stages.



Ease-of-Use

- Fast learning is essential.
- MAX: 1-day delay for user-mode



TOPAZ

- Interfaz to Full-System Simulation.
- Multithreaded simulation for massive number of routers.
- Simple & Complex models.
- Dynamic accuracy simulation.
- Many out-of-the-box components.
- Very modular code, easy to understand.

Outline

- **Simulator Description**
- **Out-of-the-Box**
- **Utilization Examples**
- **Support & Collaboration**

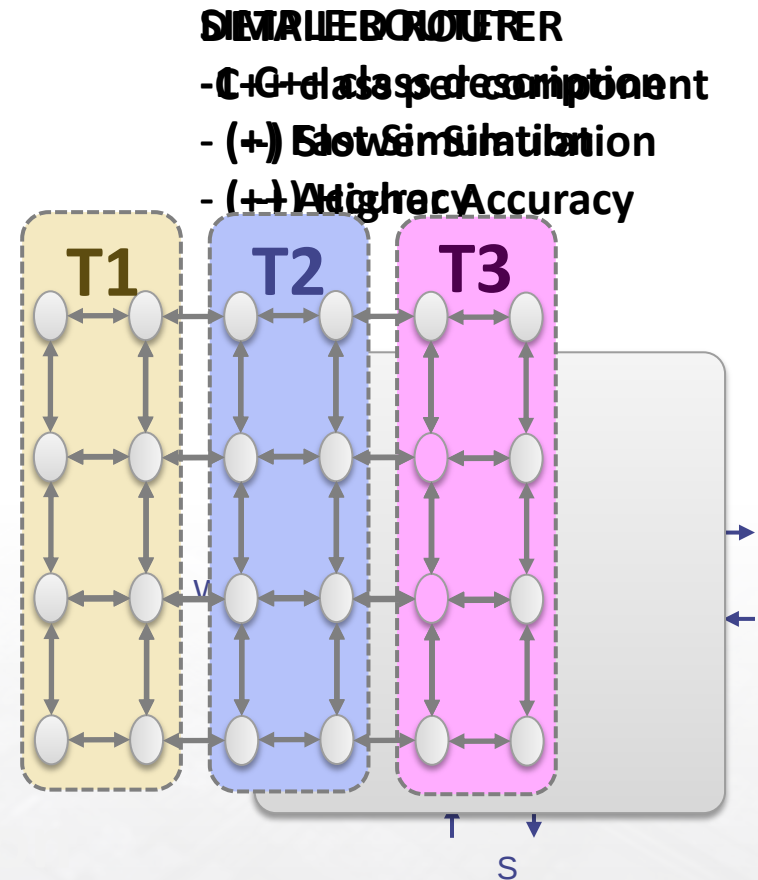
Main Features

- Evolution of SICOSYS
- Object-oriented Design
 - Implemented in C++
 - 100 classes / 50,000 lines of code
 - High portability (C++ standard compiler)
- Different levels of detail
- Support for different network topologies

[REF] V.Puente, J.A. Gregorio, R. Bevide, SICOSYS: an integrated framework for studying interconnection network performance in multiprocessor systems. IEEE Comput. Soc, 2000.

Main Features

- Evolution of SICOSYS
- Object-oriented Design
- Different levels of detail
- Support for different levels of detail



Using TOPAZ (Building)

```
>./TPZSimul -s SIMUL_DETAILED
```

TPZSimul.ini

```
<RouterFile id="../sgm/Router.sgm">  
<NetworkFile id="../sgm/Network.sgm">  
<SimulationFile id="../sgm/Simula.sgm">
```

Simula.sgm

```
<Simulation id="SIMUL_DETAILED">  
  <Network id="TORUS">  
  <SimulationCycles id=100000>  
  <DiscardTraffic id=10000>  
  <TrafficPattern id="MODAL" type="RANDOM">  
  <Load id=0.1>  
  <PacketLength id=2>  
</Simulation>
```

Network.sgm

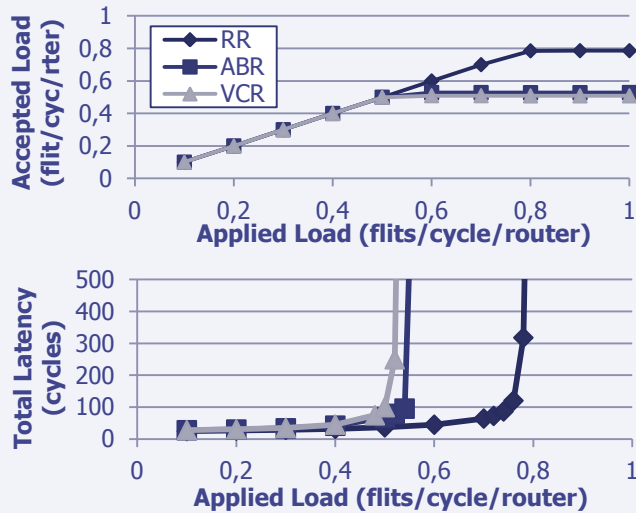
```
<TorusNetwork id="TORUS" sizeX=8 sizeY=8 router="DETAILED" delay=1>  
<MeshNetwork id="MESH" sizeX=8 sizeY=8 router="DETAILED" delay=1>
```

Router.sgm

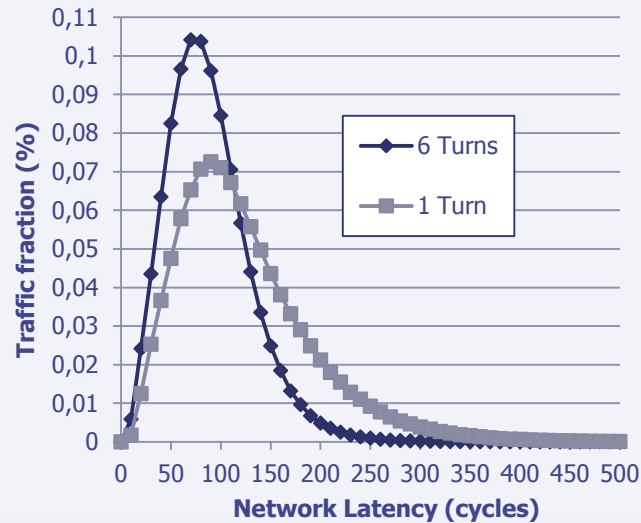
```
<Router id="DETAILED" inputs=5 outputs=5 bufferSize=64  
bufferControl="ST" routingControl="ROUTING_ALG">  
  <Injector id="INJ">  
  <Consumer id="CONS">  
  <Buffer id="BUF1" type="X+" headerDelay=2>  
  <Buffer id="BUF2" type="X-" headerDelay=2>  
  .....  
  <Buffer id="BUF5" type="Node" headerDelay=2>  
  <Routing id="RTG1" type="X+" headerDelay=1>  
  .....  
  <Routing id="RTG5" type="Node" headerDelay=1>  
  <Crossbar id="XBAR" inputs="5" outputs="5" type="CT">  
    <Input id=1 type="X+">  
    .....  
    <Output id=5 type="Node">  
  </Crossbar>  
  
  <Connection id="C01" source="INJ" destination="BUF5">  
  .....  
  <Connection id="C20" source="RTG.1" destination="XBAR.1">  
  .....  
</Router>
```

Using TOPAZ (Printing)

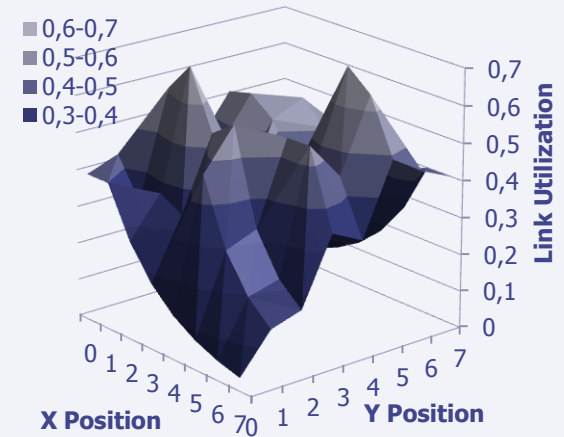
Standalone



Throughput/Latency curves

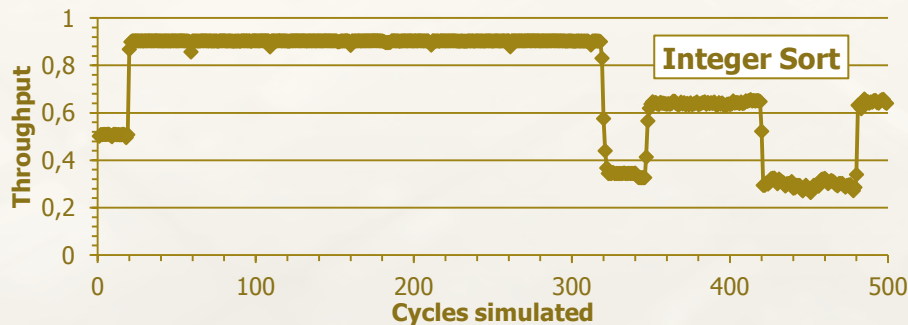


Latency Histogram



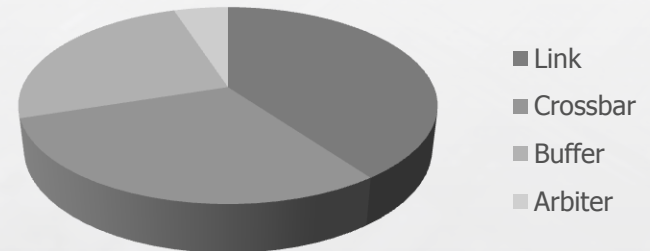
Injection/Consumption/Link map

+ Gems (or Gem5)



Throughput/Latency evolution

+ Orion



Power Breakdown

Outline

- Simulator Description
- **Out-of-the-Box**
- Utilization Examples
- Support & Collaboration

Out of the Box

1. Configuration Parameters

Router	Flow Control	Topology	Traffic
Buffer Size	Virtual Cut Through	Ring	Random
Buffer Delay			Bit-Reversal
Packet Size	Bubble Flow Control	Mesh (2D & 3D)	Perfect-Shuffle
# Virtual Channels			Transpose Matrix
#Physical networks	Wormhole	Torus (2D & 3D)	Tornado
Message Types			Hot-Spot
Router Pipeline	Virtual Channel flow Control	Midimew (2D)	Local
Link Delay			Trace-Based
		Square Midimew (2D)	

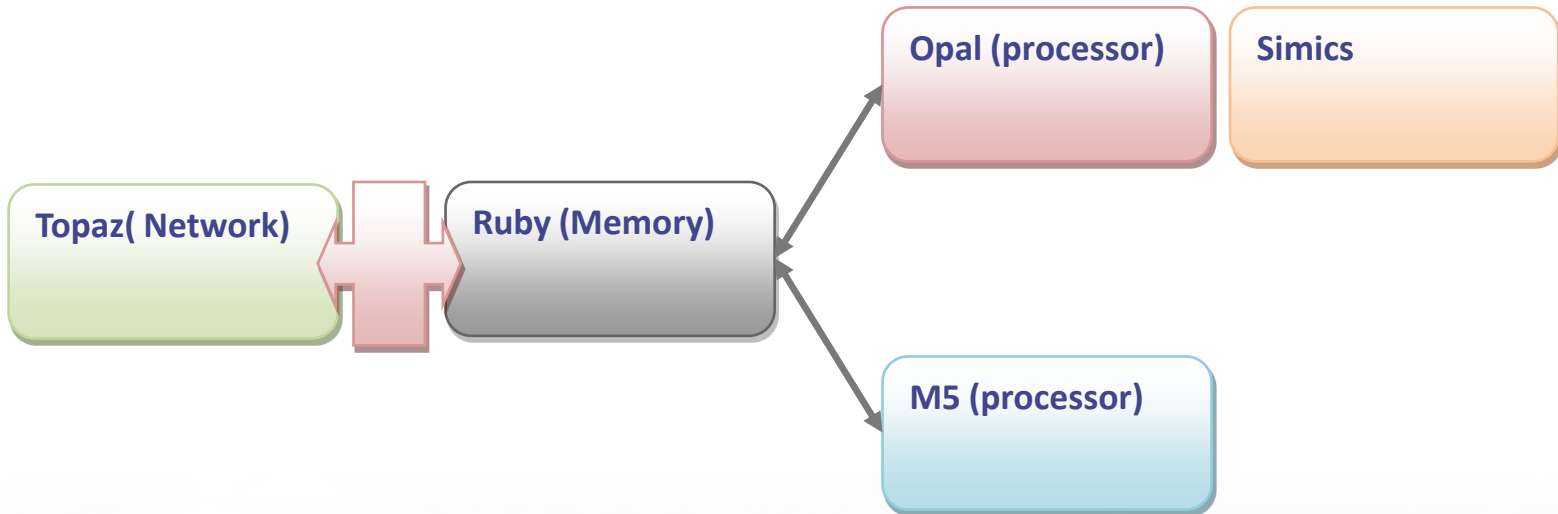
Out of the Box

2. Available Routers

Router	REF	Year	Level of Detail
Adaptive Bubble Router	[14]	2001	Complex & simple
Deterministic Bubble Router	[15]	1998	Complex & simple
Deterministic with VC (Dally)	[16][17]	2001	Complex & simple
VCTM (Dally + MC Support)	[18]	2008	Complex & simple
Pipeline Optimized	[24]	2008	Complex & simple
Rotary Router	[19]	2007	Complex
Bufferless Router	[21]	2010	Simple
Bidirectional Router	[22]	2009	Simple
Buffered Crossbar	[23]	1987	Complex

Out of the Box

3. Integration with Full-System Simulation Tools



Wisconsin Multifacet Gems: <http://research.cs.wisc.edu/gems/>

Gem5 simulator system: http://gem5.org/Main_Page

Outline

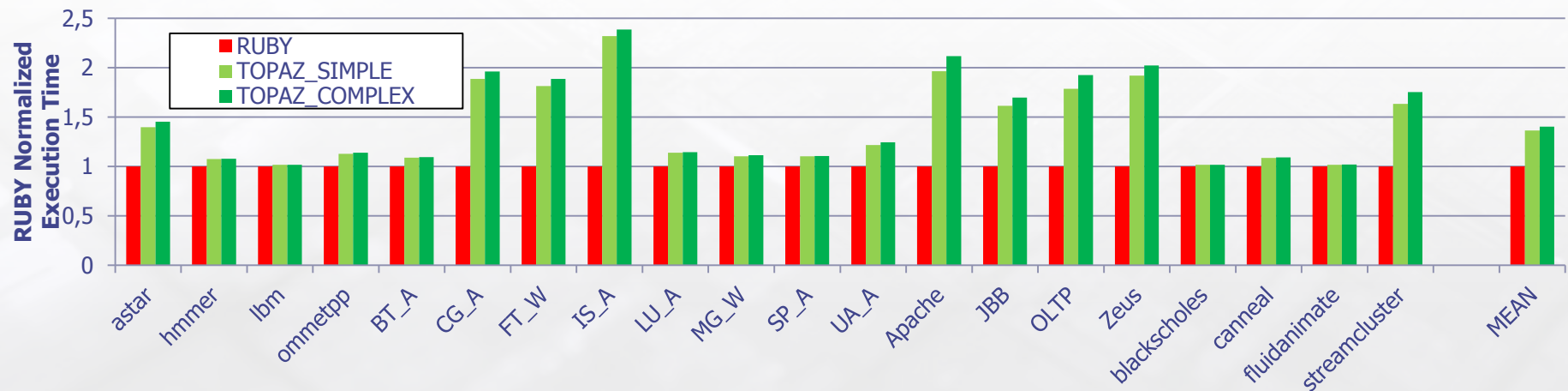
- Simulator Description
- Out-of-the-Box
- **Utilization Examples**
- Support & Collaboration

Increasing Full-System simulation accuracy

Main System Parameters

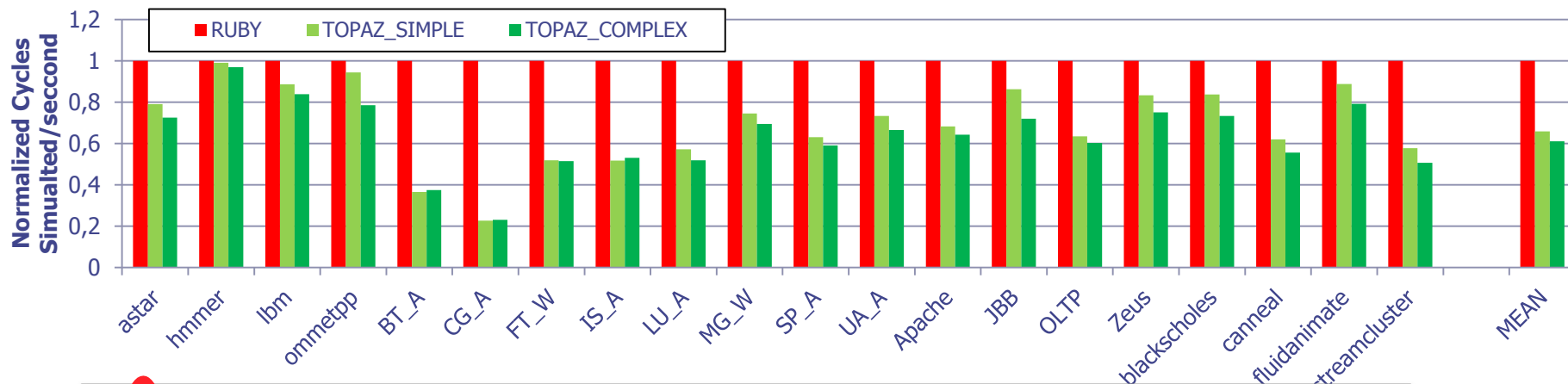
System				Network	
Cores	16 Cores, @4GHz, OOO, 4-wide issue, 64-entry IW, 16 outstanding Mem. Req	L2	16 MB, SNUCA, Token(B) coherence protocol, 6 msg. dependence chain	Topology	4x4 Mesh
L1	Independent I/D caches, 32KB, 4-way, 1 cycles	L2 Bank	1MB, 16-way, 5 cycles, pseudo LRU	Links	1 cycle, 128bits wide
Memory	4GB, 320GB/s, 260 cycles	OS	Solaris 10		

Broadcast Coherence Protocol (Execution Time)

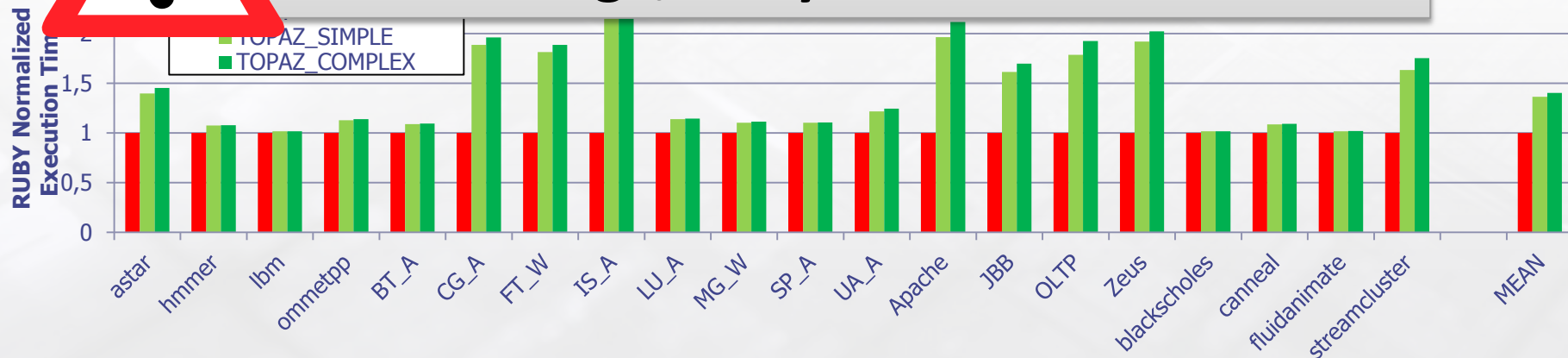


Increasing Full-System simulation accuracy

Simulation speed (cycles/second)

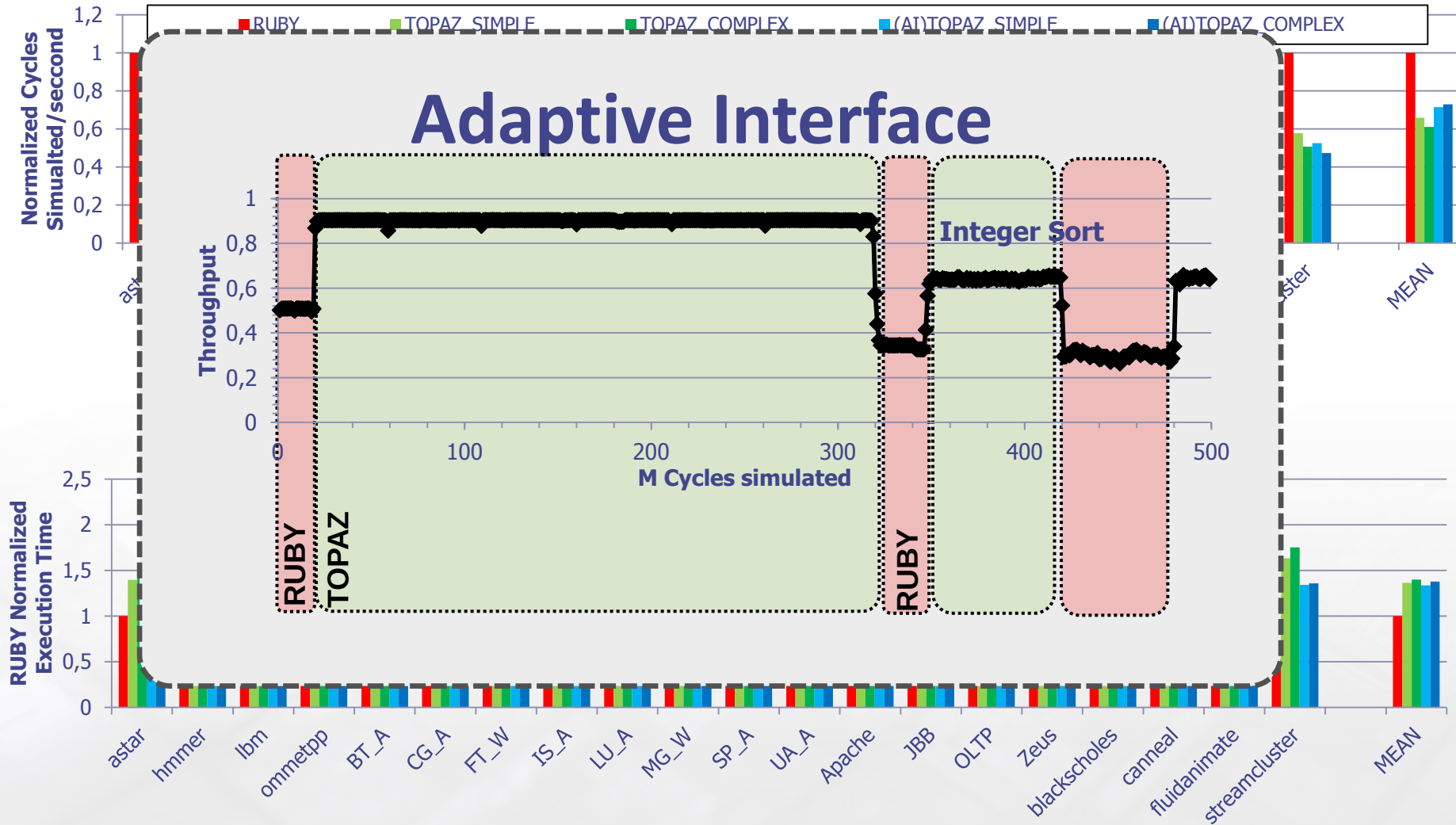


More Accuracy => Slower simulations
On average, Ruby is $\approx 2X$ faster



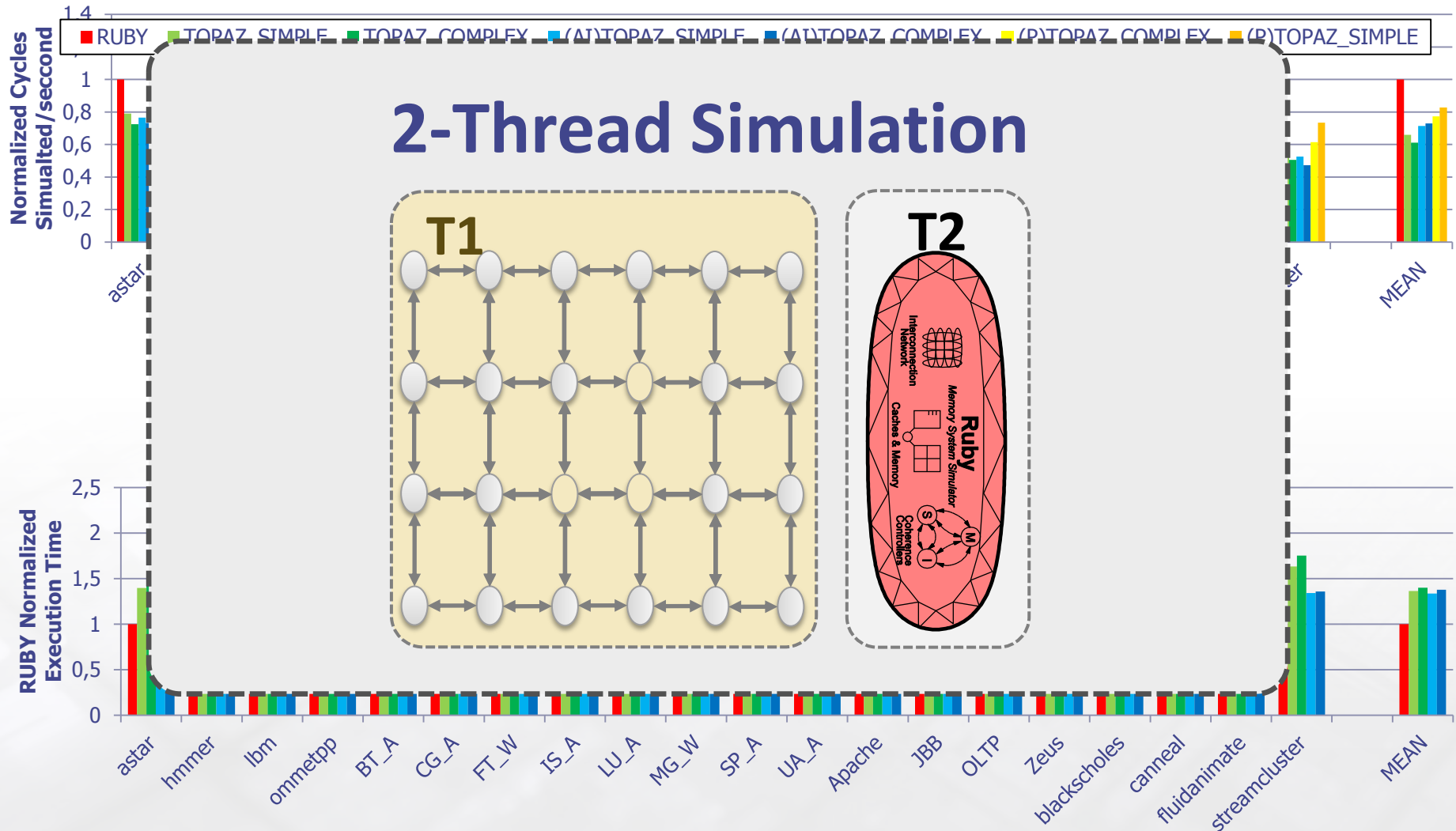
Improving Simulation Speed (I)

Simulation speed (cycles/second)



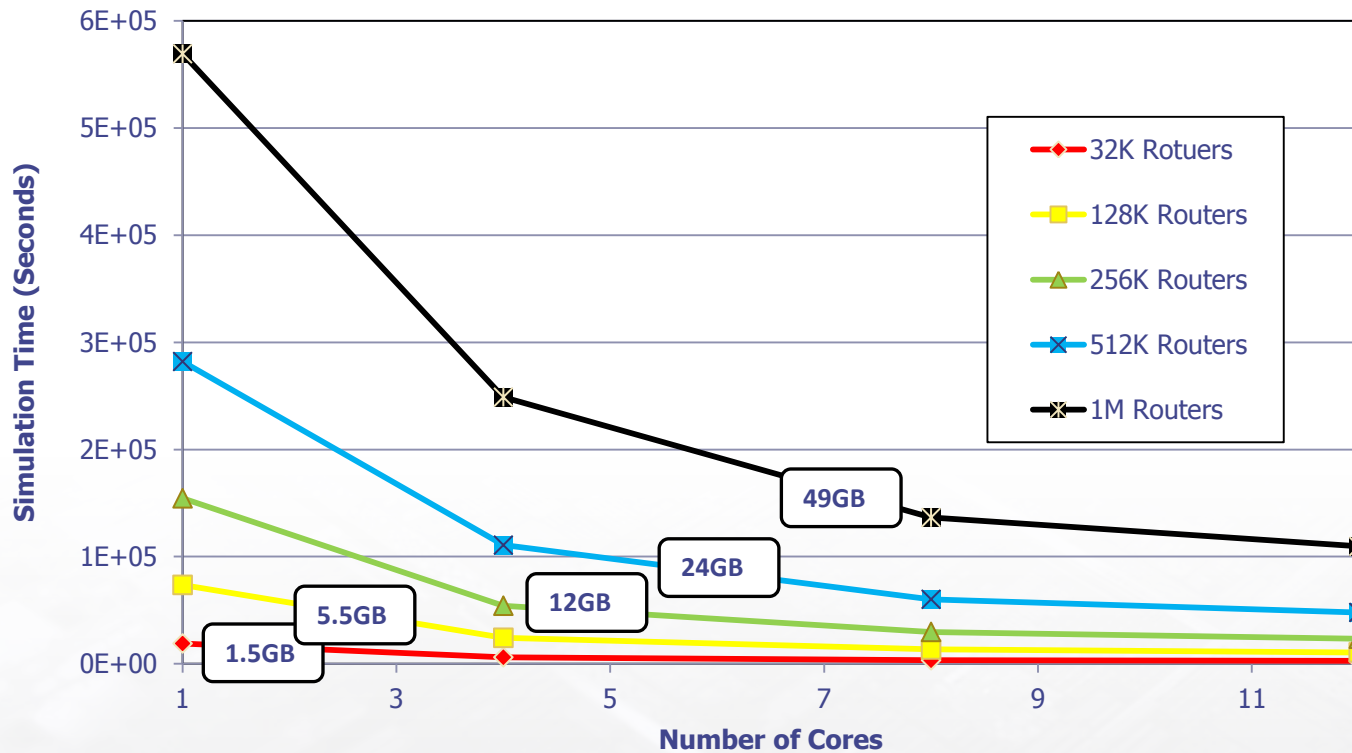
Improving Simulation Speed (II)

Simulation speed (cycles/second)



Simulating thousand-node Networks

12-Core (Xeon E5645) server with 54GBytes of main memory.



- 3D Torus, Bubble Router (simple), similar to IBM Blue Gene.
- Multithreaded implementation takes advantage of multicore server
- Good speedup for 1 Million routers

Outline

- Simulator Description
- Out-of-the-Box
- Utilization Examples
- **Support & Collaboration**

Support & Collaboration



The screenshot shows the Google Code project page for 'tpzsimul'. The browser address bar displays 'code.google.com/p/tpzsimul'. The page title is 'tpzsimul' with the subtitle 'TOPAZ Interconnection network simulator'. Navigation links include 'Project Home', 'Wiki', 'Issues', and 'Source'. The 'Summary' tab is selected, showing 'Project Information' and 'Overview'. The 'Project Information' section includes a recommendation button, star status, project feeds, code license (GNU GPL v2), and labels (Academic, Simulator, Interconnection networks, Network on Chip, CMP, Architecture, GEMS, GEM5, Full system simulator, Large scale networks). The 'Members' section lists 'vpunte' as the maintainer, with 2 committers and 3 contributors. The 'Overview' section describes TOPAZ as a general-purpose interconnection network simulator. The 'Announcements' section highlights the initial public release. The 'Documentation' section lists links for getting started, configuration, and integration. The 'References' section cites a paper from NOCS 2012.

code.google.com/p/tpzsimul

tpzsimul
TOPAZ Interconnection network simulator

Project Home [Wiki](#) [Issues](#) [Source](#)

Summary [People](#)

Project Information

 Recommend this on Google

★ Starred by 1 user
[Project feeds](#)

Code license
[GNU GPL v2](#)

Labels
Academic, Simulator, Interconnection networks, Network on Chip, CMP, Architecture, GEMS, GEM5, Full system simulator, Large scale networks

Members
[vpunte](#)
2 committers
3 contributors

Your role
[Committer](#)

Links

External links
[Galerna](#)
[Unican](#)

Overview

TOPAZ is a general-purpose interconnection network simulator that allows the modeling a wide variety of message routers with different tradeoffs between speed and precision. TOPAZ comes from [SICOSYS](#) simulator, which was originally conceived to obtain results are very close to those obtained by using HDL description of networks components by hardware simulators but at lower computational cost. In order to make the tool easily comprehensible, extensible and reusable, the design of the tool is object-oriented and its implementation is in C++ language. For the models provided, approximately 110 classes, distributed in about 50,000 lines of code have been necessary. The simulator has support for parallel execution using standard POSIX threads. The portability is very high and can be used in any UNIX platform with a C++ standard compiler.

Announcements

- [10/02/2012] Initial release publicly available (See [repositories](#))

Documentation

- [Getting Started](#) A quick guide to understand how to get the sources and compile the simulator.
- [Simulator Configuration](#) How to configure the simulator and most common command line options
- [SGML Configuration](#) A quick review of SGML configuration files. How to define Routers, Networks and Simulations
- [Components Available](#) A list of the components availables in the *tip*
- [GEMS Integration](#) A small review of GEMS integration and a guide to compile GEMS with TOPAZ networks
- [GEM5 Integration](#) A small review of GEM5 integration and a guide to compile GEM5 with TOPAZ networks

References

If you use this tool in your research, please cite the following paper:

P. Abad, P. Prieto, L. Menezes, A. Colaso, V. Puente, J. A. Gregorio. TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers, **NOCS 2012** ([pdf](#))

Support & Collaboration



The screenshot shows a web browser window with the address bar displaying `code.google.com/p/tpzsimul/wiki/GStarted?tm=6`. The page title is "tpzsimul" with the subtitle "TOPAZ Interconnection network simulator". The navigation bar includes links for "Project Home", "Wiki", "Issues", and "Source". The "Wiki" tab is active, showing a list of pages: "Getting Started", "First Steps", "Configuration", "Out of the Box", "GEMS Integration", and "GEM5 Integration". The "Getting Started" page is selected, displaying a "GStarted" section with the subtitle "Simple guide for download and compile TOPAZ" and a "Updated Feb 3, 2012 by" note. The main content area is titled "Guide to Compile TOPAZ network simulator" and includes a "Step 1: Get It" section. This section explains that a Mercurial client is needed and provides a terminal command to clone the repository: `hg clone https://code.google.com/p/tpzsimul/ tpzsimul`. A "Step 2: Compile It" section follows, stating that the code is structured in 4 directories: **src** (implementation of all classes, i.e., `.cpp`), **inc** (specification of all classes, i.e., `.hpp`), **sgm** (SGML samples of different router microarchitectures, networks, etc...), and **mak** (where the simulator has to be compiled).

code.google.com/p/tpzsimul/wiki/GStarted?tm=6

tpzsimul
TOPAZ Interconnection network simulator

Project Home Wiki Issues Source

Edit << ☆ GStarted
Simple guide for download and compile TOPAZ Updated Feb 3, 2012 by

- [Guide to Compile TOPAZ network simulator](#)
 - [Step 1: Get It](#)
 - [Step 2: Compile It](#)
 - [Step 3: Use It](#)

Guide to Compile TOPAZ network simulator

Step 1: Get It

You will need a [Mercurial](#) client in order to do so. If you have little idea about DVCS, we suggest to take a look at the best [Mercurial tutorial](#) out there.

```
hg clone https://code.google.com/p/tpzsimul/ tpzsimul
```

Step 2: Compile It

The code is structured in 4 directories:

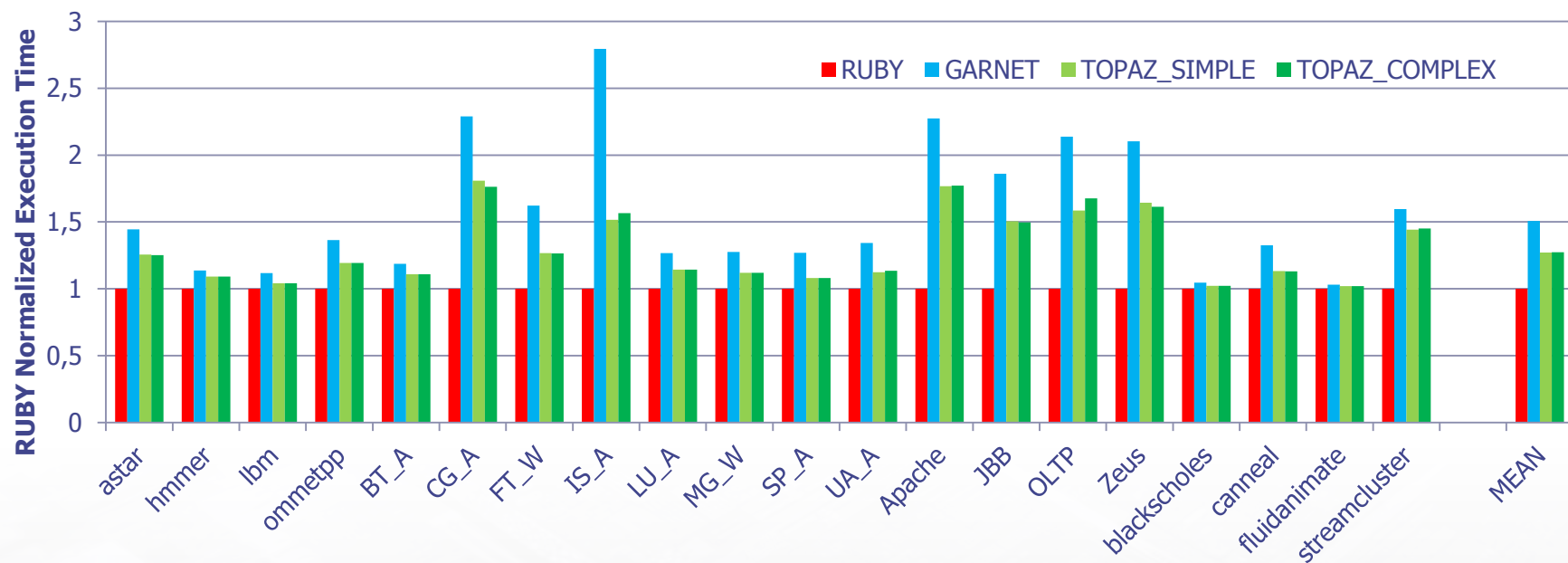
- **src** includes the implementation of all classes, i.e., `.cpp`
- **inc** includes the specification of all classes, i.e., `.hpp`
- **sgm** includes SGML samples of different router microarchitectures, networks, etc...
- **mak** is where the simulator has to be compiled

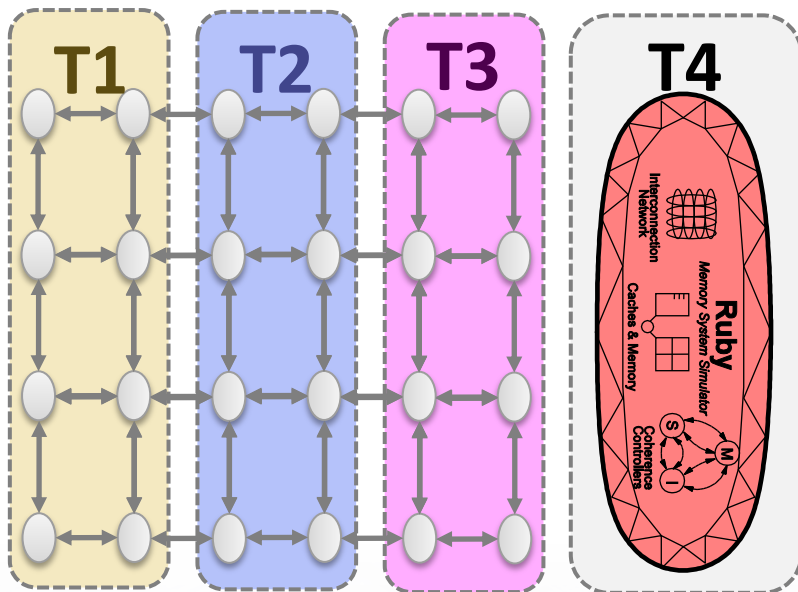
<http://www.atc.unican.es/galerna/index.html>

Thanks for your attention

Questions?

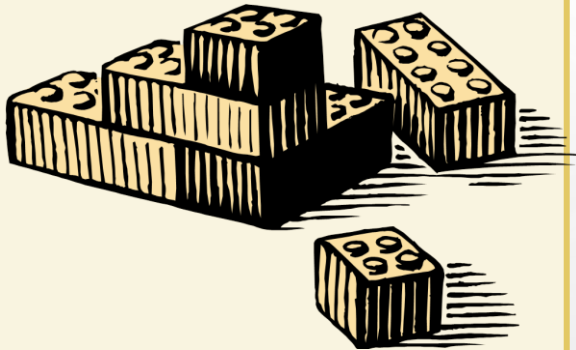
GARNET



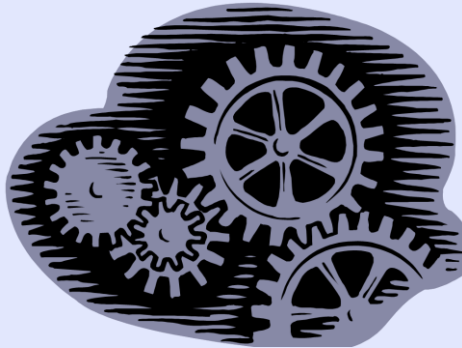


Using TOPAZ

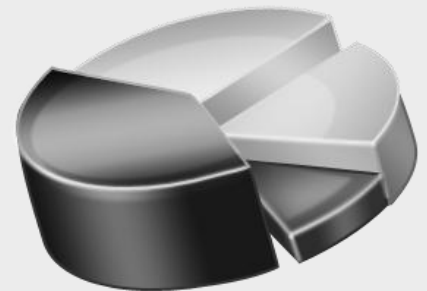
BUILDING



RUNNING



PRINTING

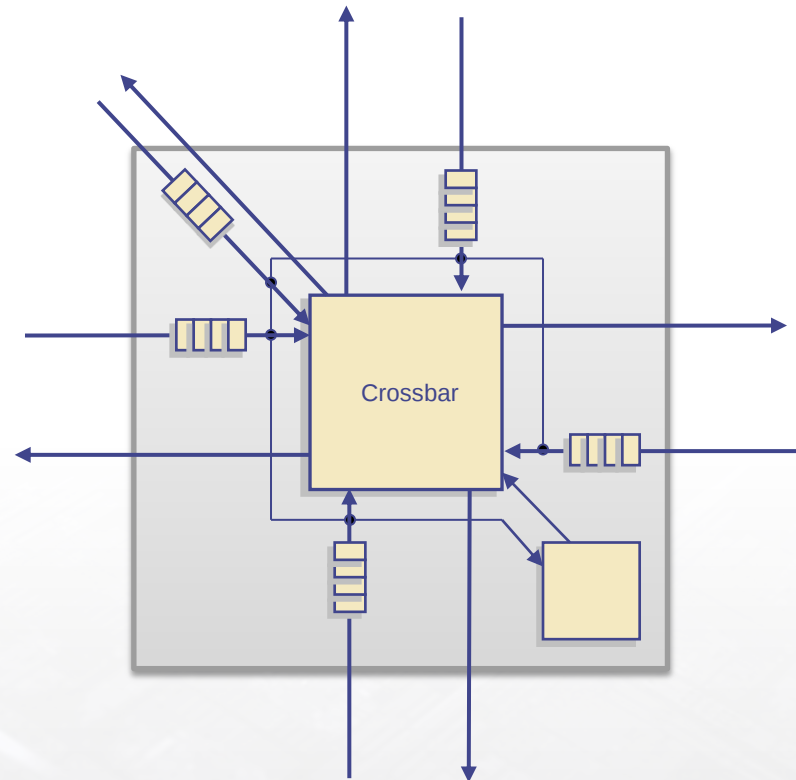


Using TOPAZ (Building)

Router.sgm

- Router & Crossbar Ports
- Buffer Size
- Routing & Flow Control Policies

Network.sgm



Using TOPAZ (Building)

Router.sgm

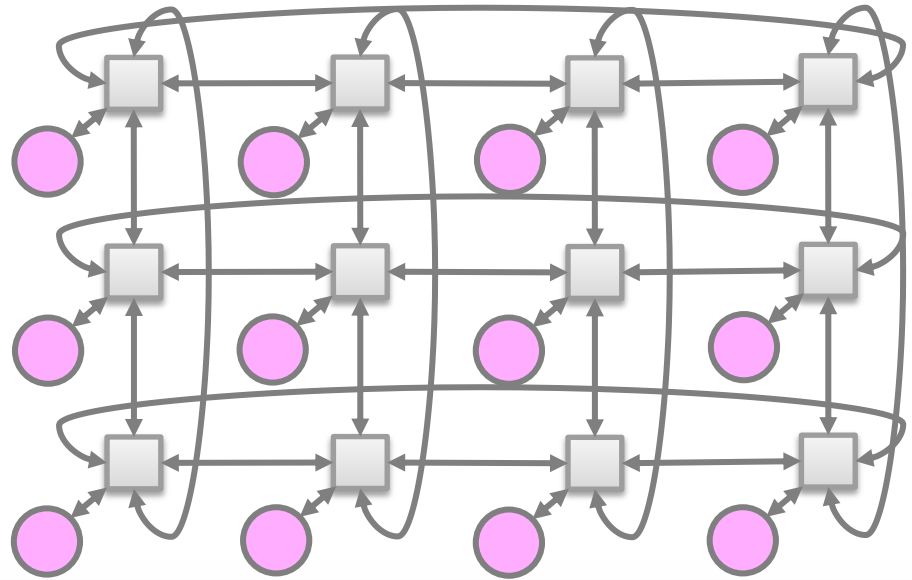
- Router & Crossbar Ports
- Buffer Size
- Routing & Flow Control Policies

Network.sgm

- Network Size
- Network Topology
- Link Delay

Simula.sgm

- Traffic Pattern
- Message Size
- Simulation Cycles



Using TOPAZ (Running)

No need to re-compile

- Only need to add new configurations at **sgml files**.
- Each configuration identified by a tag at **Simula.sgm**. Option **-s** at command line to choose a specific configuration.

Different Execution Modes:

- Run your simulation for **XXX Cycles**.
- Run your simulation until **YYY Messages** reach their destination.

Command Line Options:

- Many sgml parameters can be overwritten through command line options.
- Example:
- Useful for **scripting**.