Interconnection networks simulations for computing systems dedicated to scientific applications at the exascale

Flavio Pisani

ISHEP 2017
University of Bologna

05/04/2017

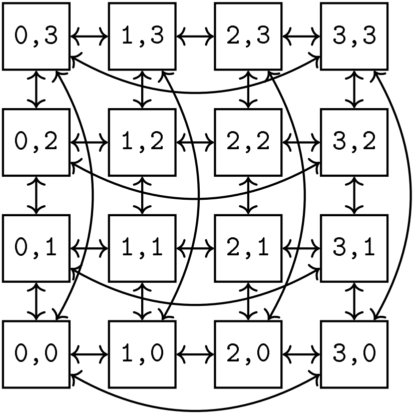
Growth of computing power and exascale

- The available computing power is exponentially growing
- The next frontier is the exascale 10^{18} FLOPS (Floating Point Operation Per Second)
- This computing power will be available interconnecting together $\sim 10^6$ computing nodes
- The interconnection network becomes extremely complex and fundamental for the system

$$\Downarrow$$

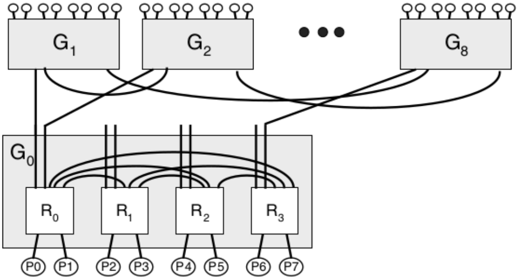
Simulations of the network system are required

N-dimensional Tori



- N-dimensional grid with periodic boundaries connections
- Every node has $2N$ neighbours
- Good scalability
- Optimized for short ranged traffic

Dragonfly



- Three layer hierarchical network: router, group e system
- P_i end nodes
- R_i routers in every group
- h channel for inter groups communication
- G_i groups in the system
- More complex scalability

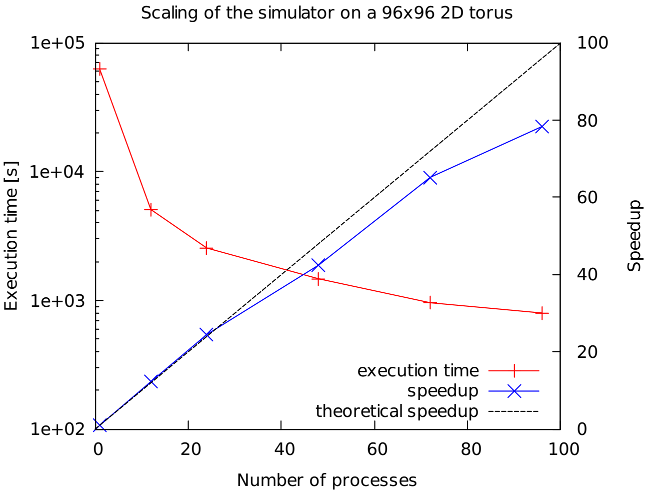
Simulation's requirements

- Low level network description
- Parallel computing capabilities
- Quick implementation of new topologies
- Quick swap of routing algorithms
- Capability of generating both synthetic and real application traffic
- Perform statistical analysis of network performances

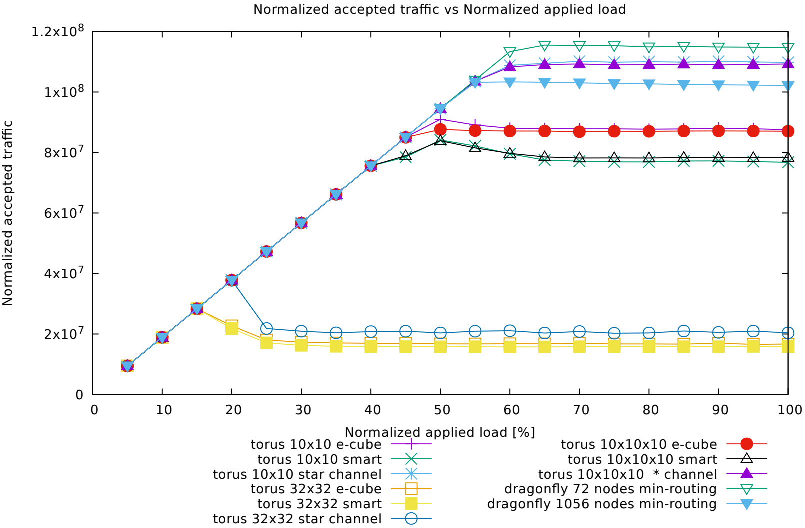
By using the OMNet++ framework we can fulfill all the requirements

Scaling of the simulator

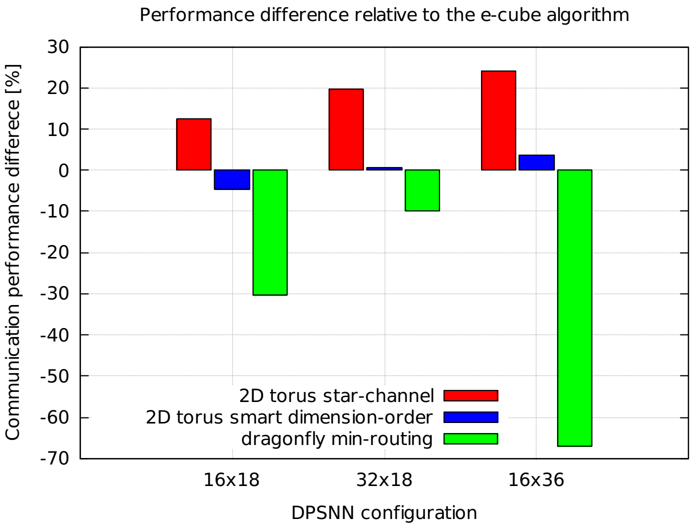
Scaling of the simulator on an intel cluster with infiniband interconnection.



Random uniform traffic



DPSNN traffic



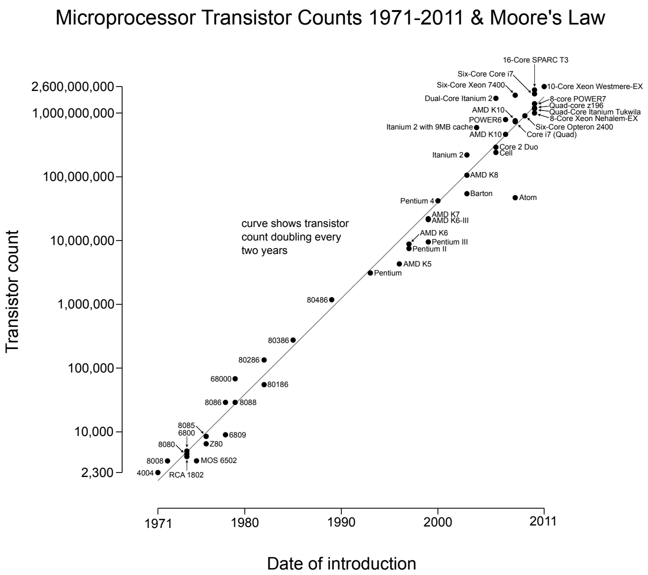
Conclusions

- Development of an accurate, flexible and scalable network simulator
- N-dimensional tori can sustain uniform traffic
- Adaptive routing algorithms improve significantly the performances
- Non-minimal partly adaptive routing algorithms do not provide significant improvements
- Dragonfly networks are efficient for uniform traffic
- DPSNN traffic is more efficient on tori than on dragonfly networks

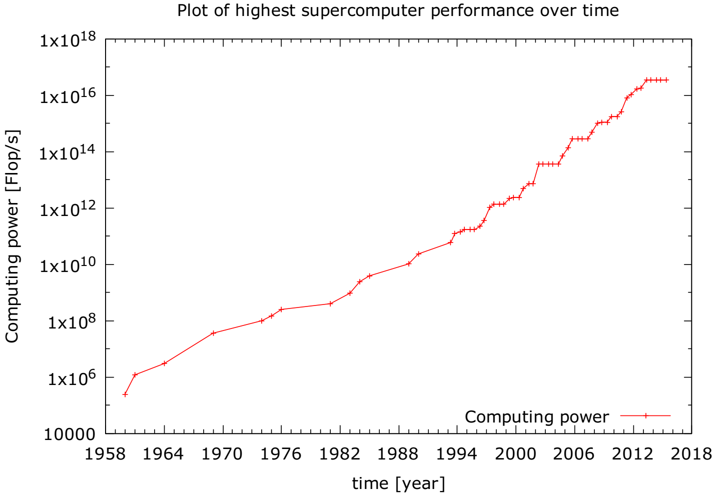
THANK YOU FOR YOUR ATTENTION

BACKUP SLIDES

Moore's law


Microprocessor Transistor Counts 1971-2011 & Moore's Law
Transistor count
2,600,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
2,300
Date of introduction
1971
1980
1990
2000
2011
curve shows transistor count doubling every two years
16-Core SPARC T3
Six-Core Core i7
Six-Core Xeon 7400
10-Core Xeon Westmere-EX
Dual-Core Itanium 2
8-core POWER7
Quad-core z196
Quad-Core Itanium Tukwila
8-Core Xeon Nehalem-EX
AMD K10
POWER6
Six-Core Opteron 2400
Itanium 2 with 9MB cache
AMD K10
Core i7 (Quad)
Core 2 Duo
Cell
Itanium 2
AMD K8
Barton
Atom
Pentium 4
AMD K7
AMD K6-III
AMD K6
Pentium III
Pentium II
AMD K5
Pentium
80486
80386
80286
68000
80186
8086
8088
8085
6800
6809
8080
Z80
8008
MOS 6502
4004
RCA 1802

top500



Plot of highest supercomputer performance over time
Computing power [Flop/s]
1x10^18
1x10^16
1x10^14
1x10^12
1x10^10
1x10^8
1x10^6
10000
1958
1964
1970
1976
1982
1988
1994
2000
2006
2012
2018
time [year]
Computing power

Neuronal models

LIFCA dynamic model for spikes generation

$$V_m < V_{th} \begin{cases} \dot{V}_m & = -\frac{V_m - E_L}{\mathrm{T}} - \frac{g_w w}{C_m} + \frac{I_e}{C_m} \\ \dot{w} & = -\frac{w}{\tau_w} \end{cases}$$

$$V_m \geq V_{th} \begin{cases} V_m & = V_{reset} \\ w & = w + A_C \end{cases}$$

Column connection model

$$Ae^{-\frac{r}{\lambda}}$$

OMNet++

What OMNeT++ is?

Discrete event simulation Framework written in C++

What does it offer?

- A flexible and configurable object-oriented structure
- A scripting language for easy definition of network topologies
- The possibility of collecting statistics during the simulation
- Native support to parallel processing through OpenMPI

Cluster specifications

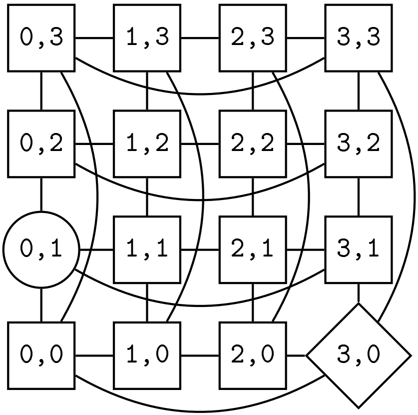
CPU	2 x Intel Xeon CPU E5620 @ 2.40 GHz
Memory	48 GB
Network card	Mellanox MT26428 Connectx2
OpenMPI version	1.10.3
Linux version	CentOS 7.2 kernel 3.10.0-327.22.2

Routing algorithms

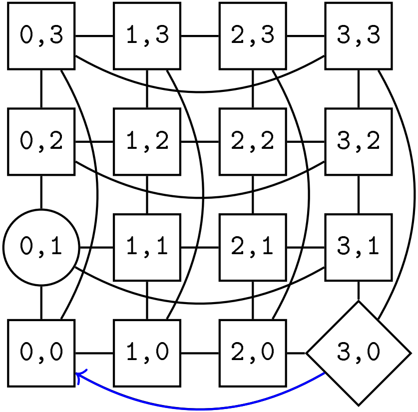
Routing algorithm's classification

- Minimal: it selects only the shortest path between two nodes.
- Deterministic: It selects only one among the available paths.
- Partly adaptive: It selects several of the available paths.
- Fully adaptive: It selects several of available paths.

Examples of routing algorithms for tori



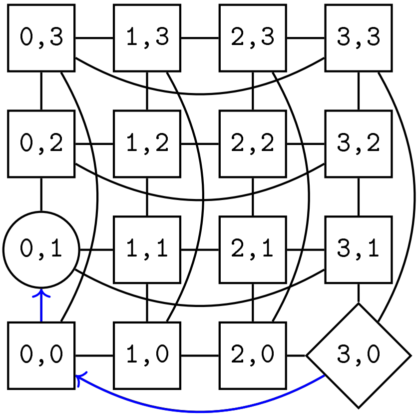
Examples of routing algorithms for tori



Minimal deterministic

e-cube

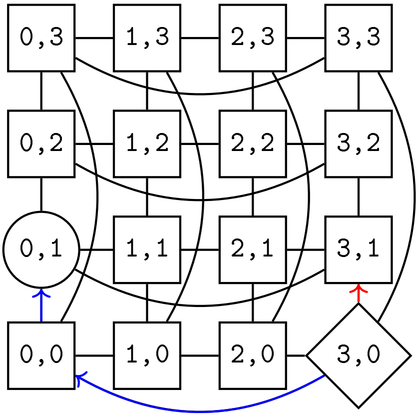
Examples of routing algorithms for tori



Minimal deterministic

e-cube

Examples of routing algorithms for tori



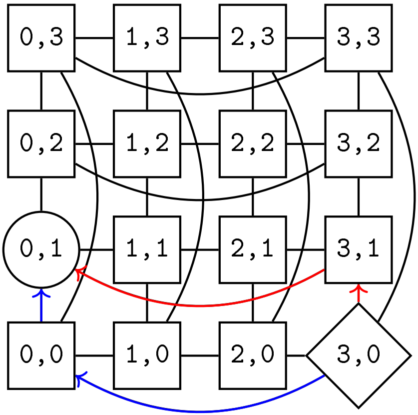
Minimal deterministic

e-cube

Minimal fully adaptive

star-channel

Examples of routing algorithms for tori



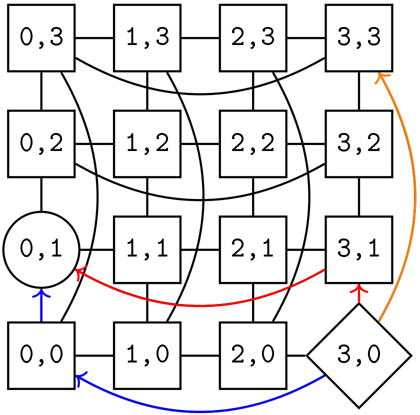
Minimal deterministic

e-cube

Minimal fully adaptive

star-channel

Examples of routing algorithms for tori



Minimal deterministic

e-cube

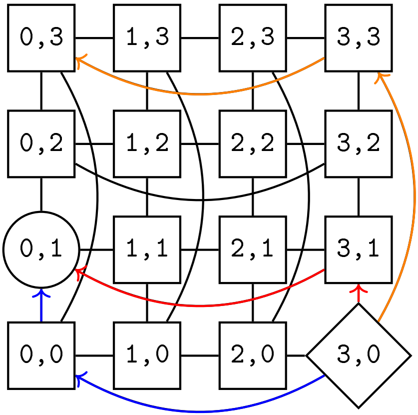
Minimal fully adaptive

star-channel

Non minimal partly adaptive

smart dimension-order

Examples of routing algorithms for tori

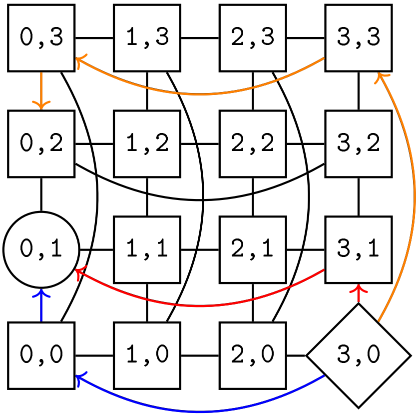


Minimal deterministic
e-cube

Minimal fully adaptive
star-channel

Non minimal partly adaptive
smart dimension-order

Examples of routing algorithms for tori



Minimal deterministic

e-cube

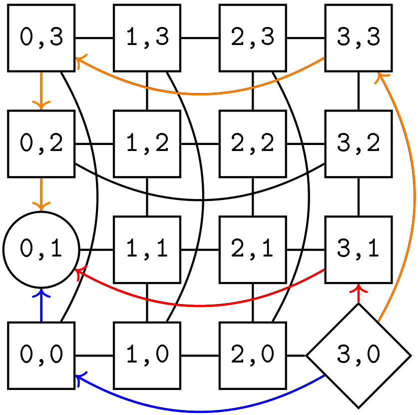
Minimal fully adaptive

star-channel

Non minimal partly adaptive

smart dimension-order

Examples of routing algorithms for tori



Minimal deterministic

e-cube

Minimal fully adaptive

star-channel

Non minimal partly adaptive

smart dimension-order

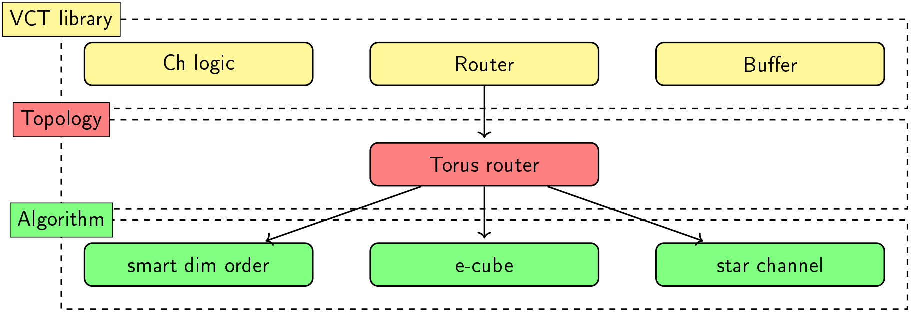
Simulation's requirements

- Low level network description
- Parallel computing capabilities
- Quick implementation of new topologies
- Quick swap of routing algorithms
- Capability of generating both synthetic and real application traffic
- Perform statistical analysis of network performances

By using the OMNet++ framework we can fulfill all the requirements

Structure of the simulation software

Three levels hierarchical software structure



Random uniform traffic