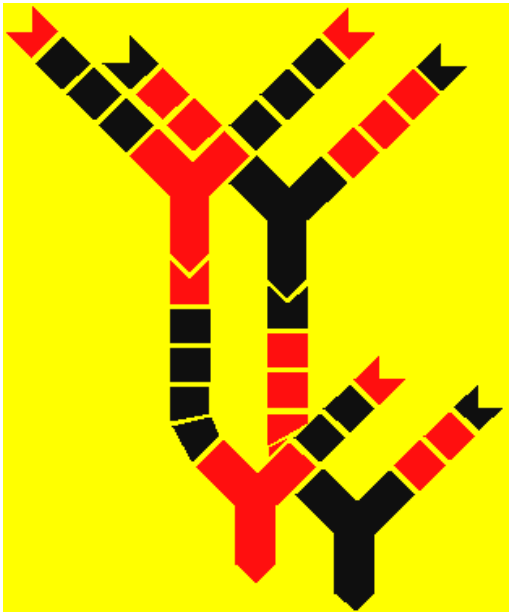
CHALLENGES AND ROADMAP FOR SCIENTIFIC APPLICATIONS AT EXASCALE



STEFANO MARKIDIS

KTH ROYAL INSTITUTE OF TECHNOLOGY

HPC2014 – CETRARO

11/7/2014

OUTLINE

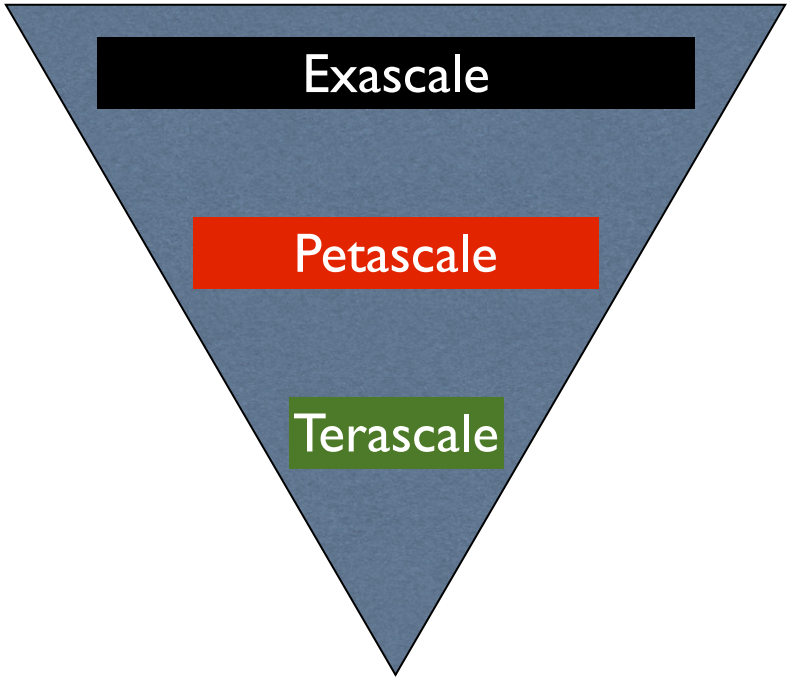
- **THE EXASCALE "REGIME" FOR APPLICATIONS**
- **EXASCALE CHALLENGES FOR APPLICATIONS**
  - THE EXASCALE UNKNOWN
  - ADOPTION LAG
  - MPI + X DILEMMA
  - ADVANCED ALGORITHMS NOT DESIGNED FOR EXASCALE
  - NOISE PROPAGATION
- **SCIENTIFIC APPLICATIONS (EPiGRAM APPLICATIONS)**
  - COMMUNICATION KERNEL
  - PERFORMANCE MODELING OF APPLICATIONS TOWARDS EXASCALE
- **ROADMAP FOR EPiGRAM APPLICATIONS**
- **CONCLUSIONS**

THE EXASCALE "REGIME"

10^{18} FLOPS

10^{15} FLOPS

10^{13} FLOPS



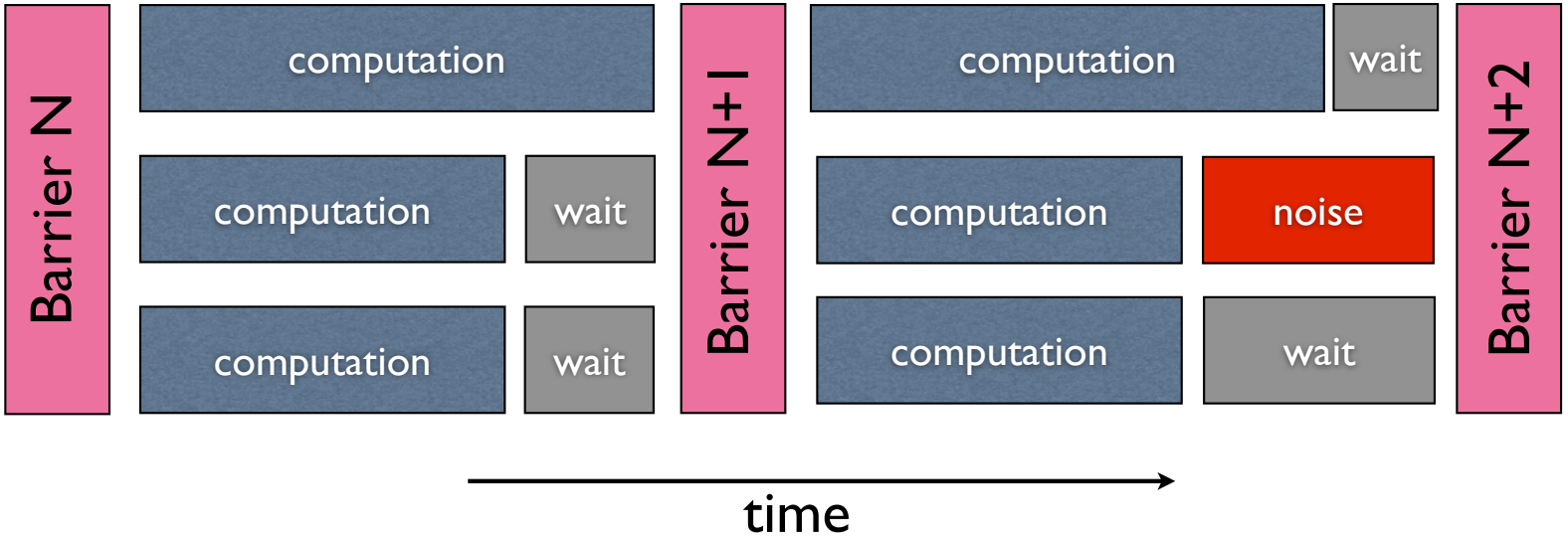
Synchronization effects become not negligible

Network latency and bandwidth bottleneck

Local memory access bottleneck

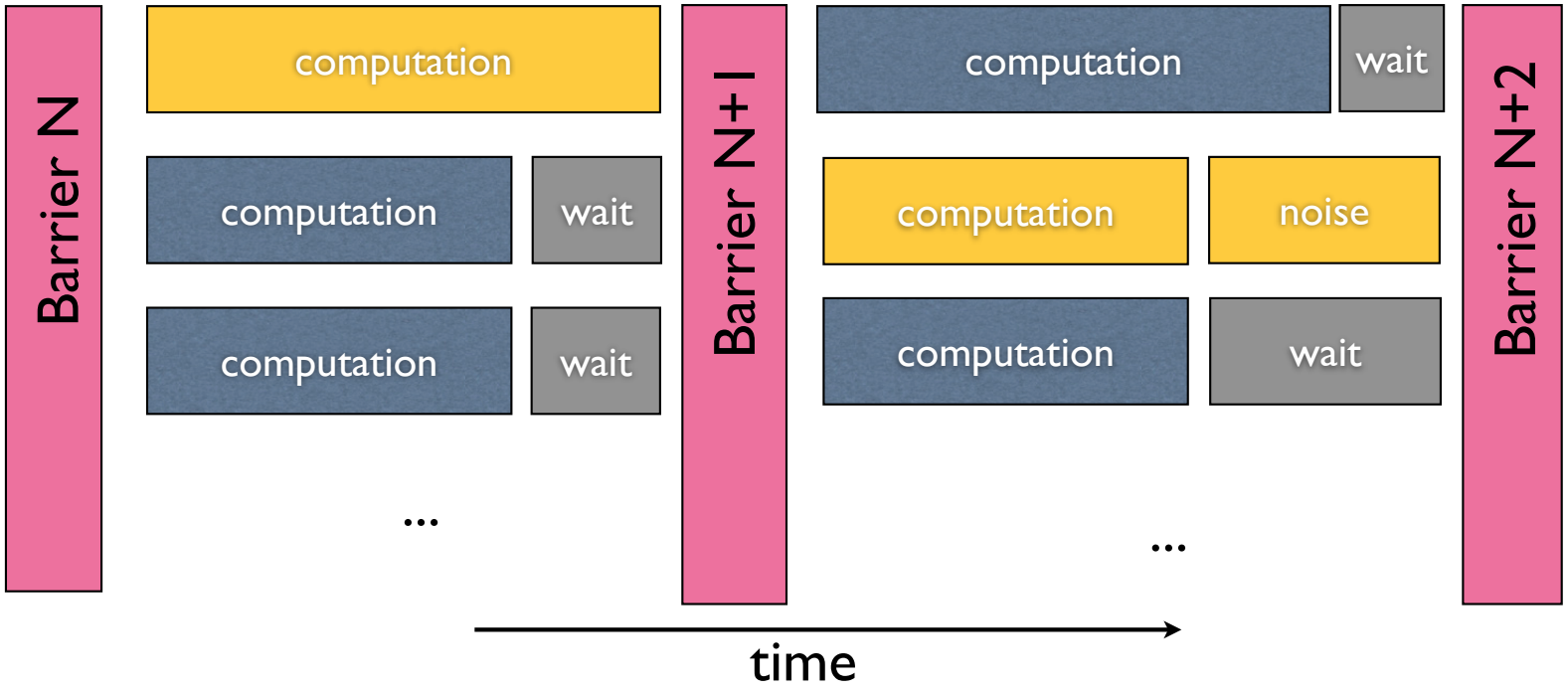
CURRENT HPC APPLICATIONS

- The vast majority of current HPC applications is *bulk-synchronous* which means each participating process is composed of iterative computation phases separated by barrier (synchronization) where speedy processes wait for lagging ones to catch up.
- Different computation times and OS/architectural overheads (noise) lead to different process execution time (unbalance)



SYNCHRONIZATION

Due to synchronization, each computation phase is as fast as the slowest process.



On billion processes, this issue will become even more serious

IMPLICIT SYNCHRONIZATION IN MP

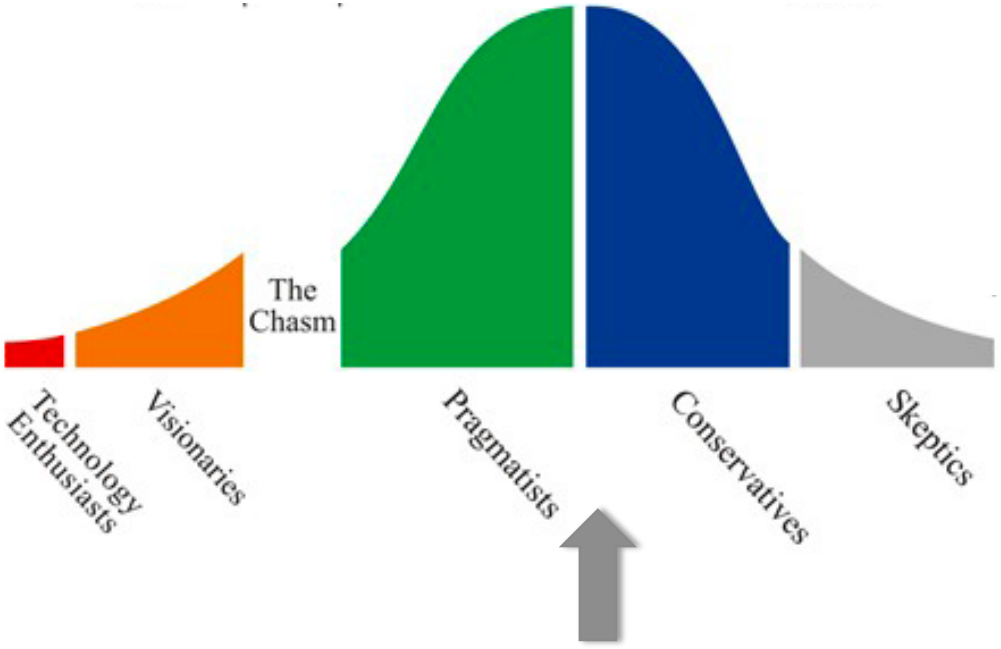
- In Message Passing PM, processes are often **synchronized implicitly** by remote data dependency. For instance, a receive can't finish before the corresponding matching send has been posted and the transmission cost has been paid.
- Synchronization properties of an application depend of the collective type algorithm, point to point messaging and the system network parameters.
- Collective communication includes several communication steps. Blocking collectives are typically synchronizing operations.

CH1: THE EXASCALE HW AND SW SYSTEM

We are preparing applications for running on machine which we don’t know the details of. More importantly, we don’t know which programming systems will be the most efficient at exascale.

CH2: ADOPTION LAG

New parallel programming systems are emerging. Traditional programming systems are evolving fast also. Application developers are in many cases conservative when it comes to embrace new programming systems.



MPI + OpenMP

Numerical libraries are faster in adopting new features of programming systems

CH3: MPI + X DILEMMA

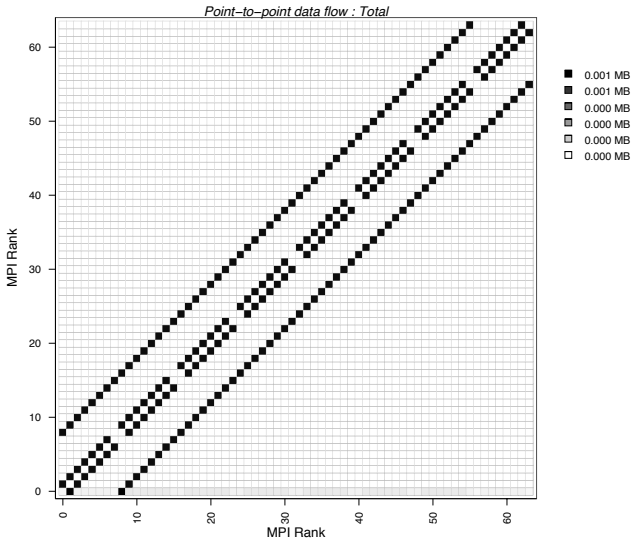
It is likely that MPI will be still the preferred approach for inter-node communication (safest choice).

How we deal with intra-node parallelism?

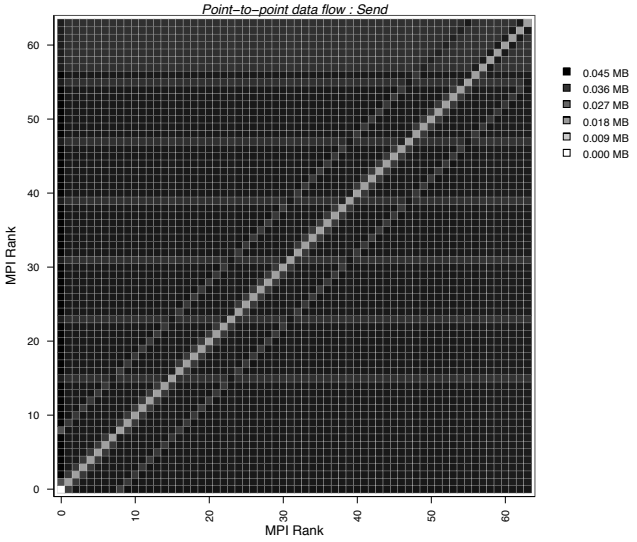
- + OpenMP.
- + PGAS. Interoperability issues with MPI.
- + MPI. MPI 3.0 and 3.1 provide shared memory mechanisms (MPI Win allocate shared and MPI endpoints).

CH4: ALGORITHMS NOT DESIGNED FOR EXASCALE

SIMPLE PETSc JFNK (GMRes)

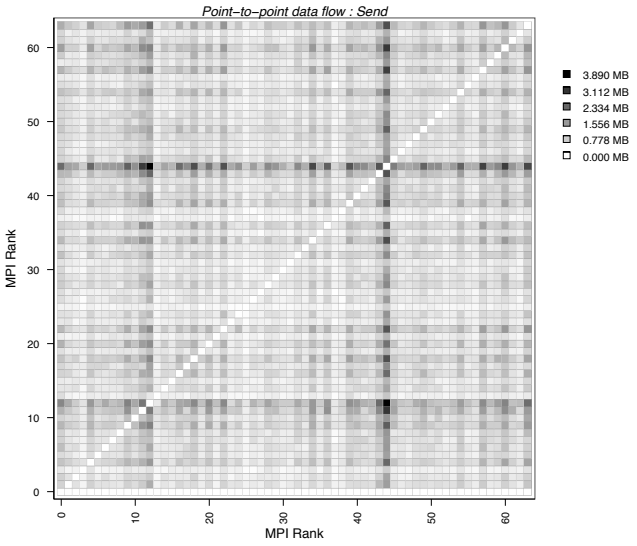


PETSc JFNK (GMRes) + MG PC + ..



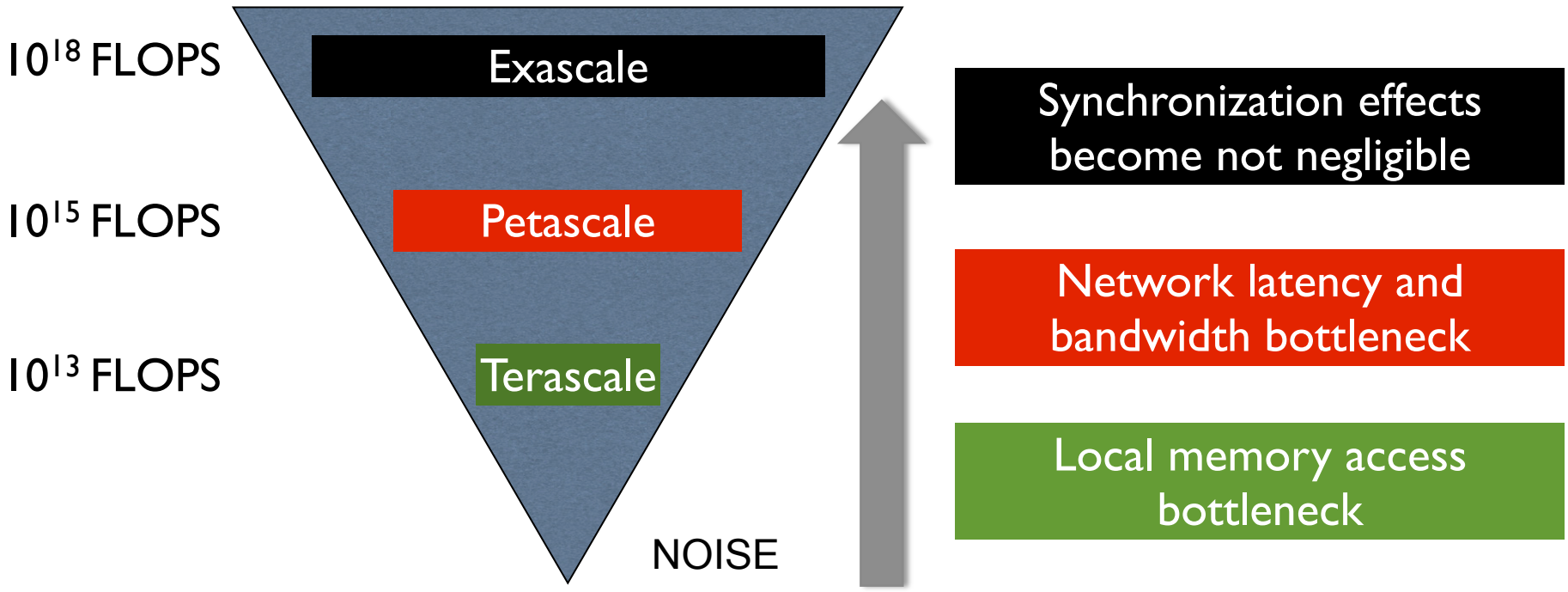
Sophisticated and advanced algorithms are often not designed for concurrency on billions of processes

GRAPH500



CH5: NOISE PROPAGATION

As the error in computations, noise (local delays on compute nodes with little impact on process < 0.25%) can propagate during the same simulation.



CH5: NOISE PROPAGATION

- Noise can be **amplified**: the delay increases linearly/logarithmically increasing the number of processes. Blocking communication often amplifies noise.
- Noise can be **absorbed**: noise can be consumed in existing synchronization delays and disappears. Non-blocking communication has high potential of absorbing noise.

Noise characteristics and properties are strongly system-dependent and interact non-linearly with collective communications.

EPIGRAM APPLICATIONS

EPiGRAM is an EC-funded project working on programming models for exascale (presented on Tuesday by E.Laure). We look at MP (MPI) and PGAS (GPI). New concepts in the programming models are tested in two applications:

- iPIC3D: Particle-in-Cell code for space physics applications.
- Nek5000: CFD code for modeling uncompressible flow (Paul Fischer) .

They are bulk-synchronous and MPI-based applications.

As first step of the project, we focused on studying communication kernel, suggesting improvement and preparing a roadmap (EPiGRAM deliverable available).

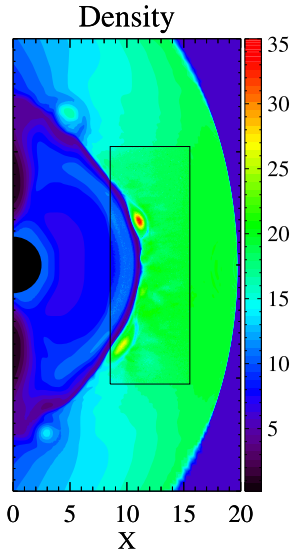
EPiGRAM

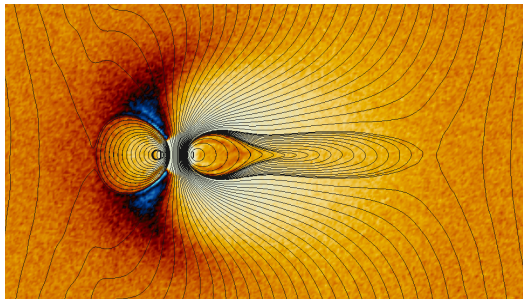
IPIC3D PARTICLE-IN-CELL CODE

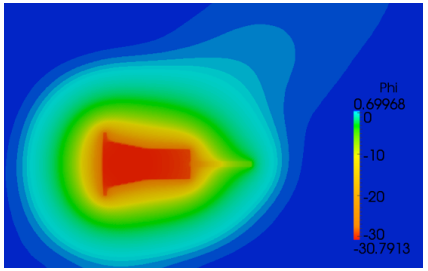
iPIC3D is an implicit Particle-in-Cell code for simulation of space plasmas. Plasma electrons and ions are mimicked by computational particles that move under the effect of a self-consistent electromagnetic field.

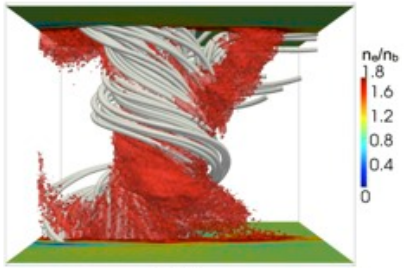
In each computational cycles, particles move, Maxwell's equations are solved, and interpolation from/to particle **are computed.**

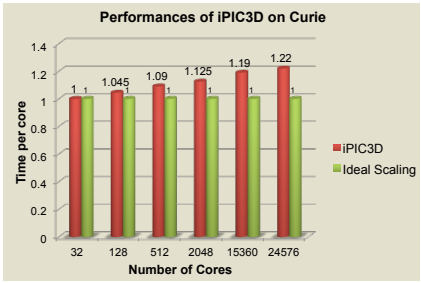
20,000 LOC C++ and MPI, open source: https://github.com/CmPA/iPic3D



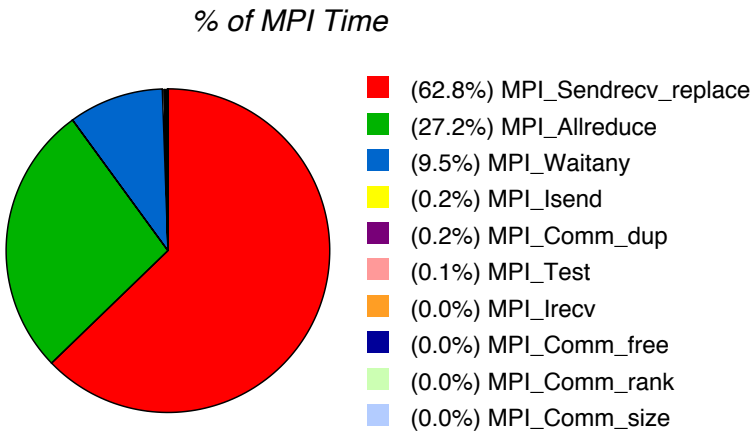


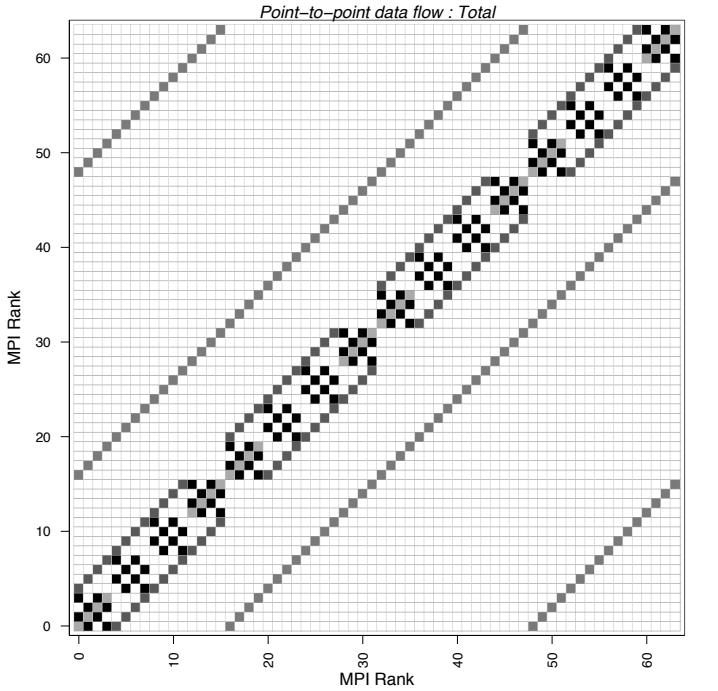


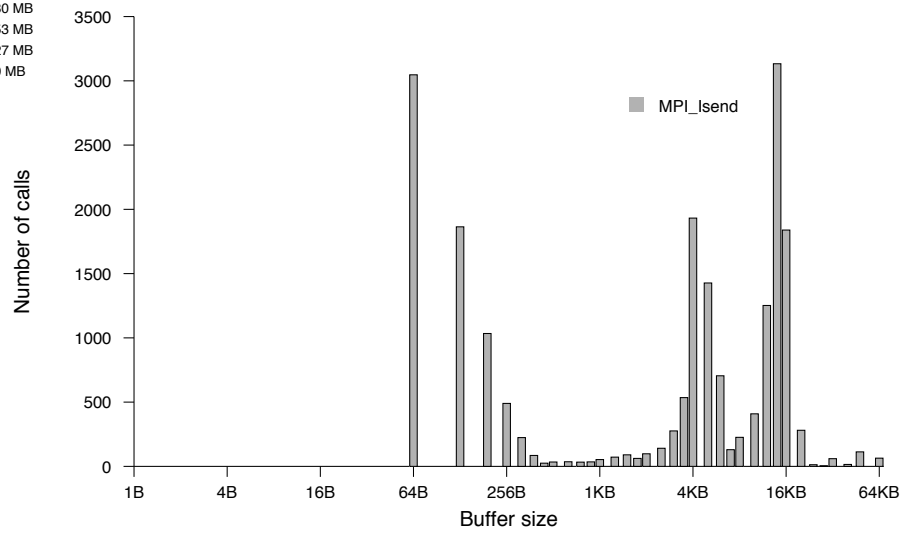




UNDERSTANDING IPIC3D COMMUNICATION KERNEL



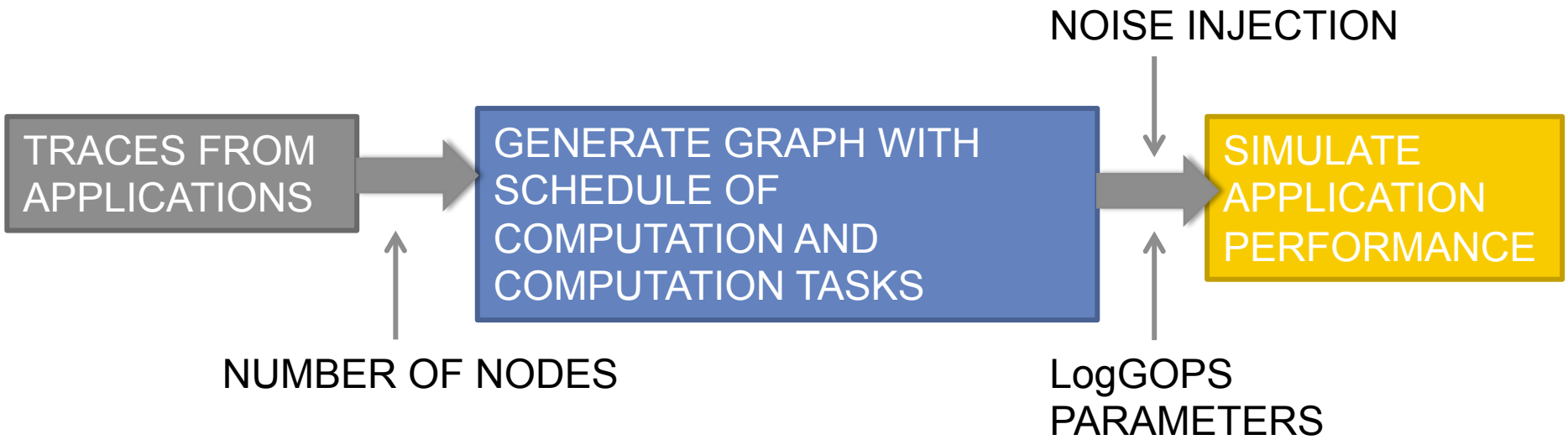




PERFORMANCE MODELING OF APPLICATIONS

In EPiGRAM, we use the LogGOPS model to simulate parallel performance of the applications. Point-2-point communication is modeled using L= latency, g = 1/bandwidth, … Collectives are modeled as a series of point-2-point communications.

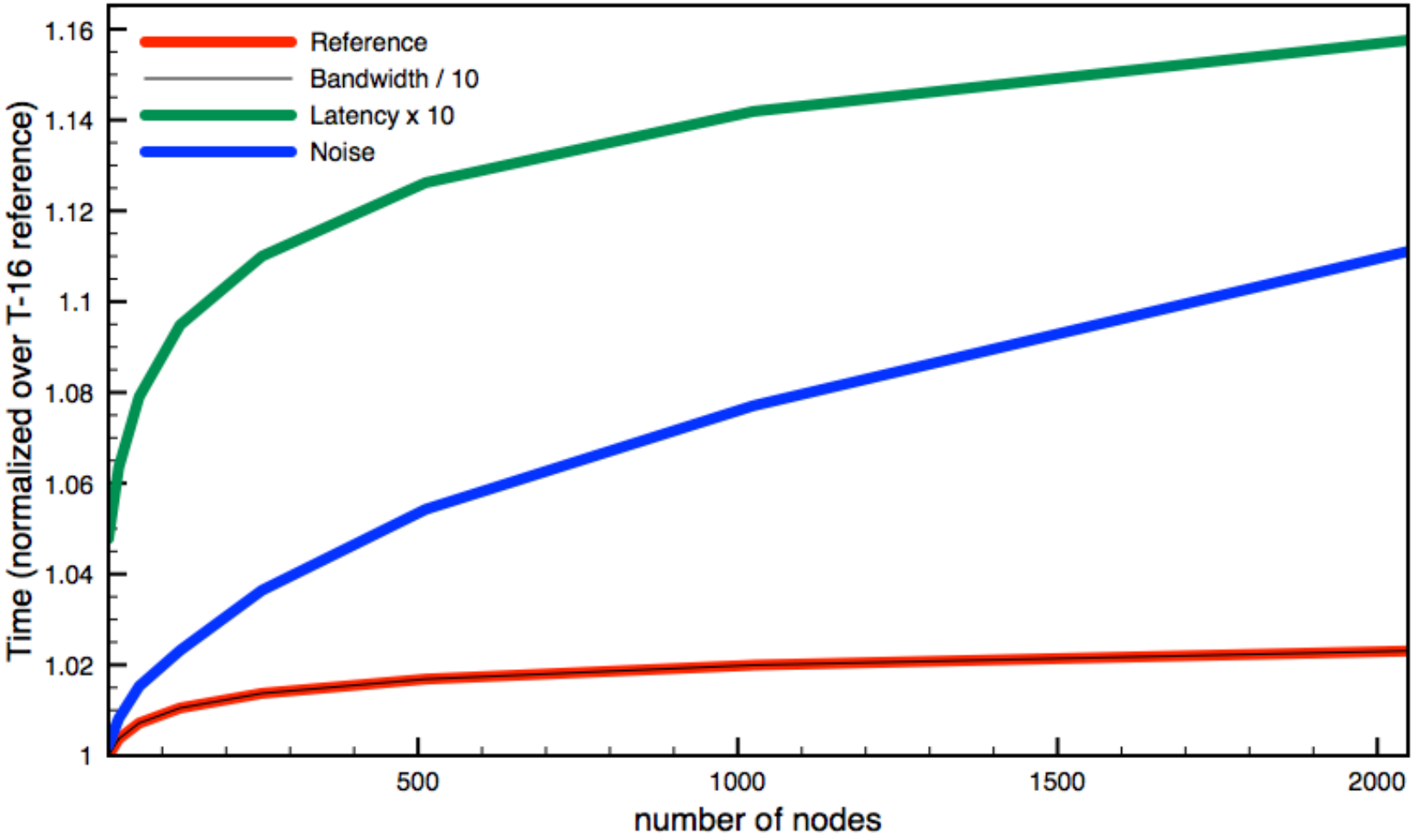
We use LogGOPSim simulator developed by T. Hoefler.



Hoefler, Torsten, Timo Schneider, and Andrew Lumsdaine. "LogGOPSim: simulating large-scale applications in the LogGOPS model." Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing. ACM, 2010.

PERFORMANCE MODELING OF IPIC3D

WEAK SCALING SIMULATION



ROADMAP FOR EPIGRAM APPLICATIONS

NEW FEATURES IN PROGRAMMING SYSTEMS

- Investigation interoperability of MPI and PGAS (GPI) in real-world codes
- Linear solvers using non-blocking sparse collectives. This requires to reformulate the solvers.
- Use of one-sided communication for particle communication in iPIC3D

PERFORMANCE MODELING

- Performance modeling of applications on billion processes.
- Study noise sensitivity of applications. Different kind of noise and different frequencies.

CONCLUSIONS

- **Entering in the new exascale "regime"**
- **From applications, many challenges ahead.**
  - Main challenge is choice of programming system. Likely that MPI will still be the main choice
  - Noise and load balance will bite hard at exascale
  - Not exascale-friendly algorithms.
- **Studying two MPI-based applications in EPiGRAM:**
  - Focus on performance modeling towards exascale.
- **Roadmap:**
  - New features in MPI in applications.
  - Focus on interoperability of MP and PGAS.
  - Study of noise sensitivity possibly on billion processes.

"All I'm saying is now is the time to develop the technology to deflect an asteroid."

from http://b612foundation.org/just-for-fun/these-dinosaurs-had-the-right-idea/