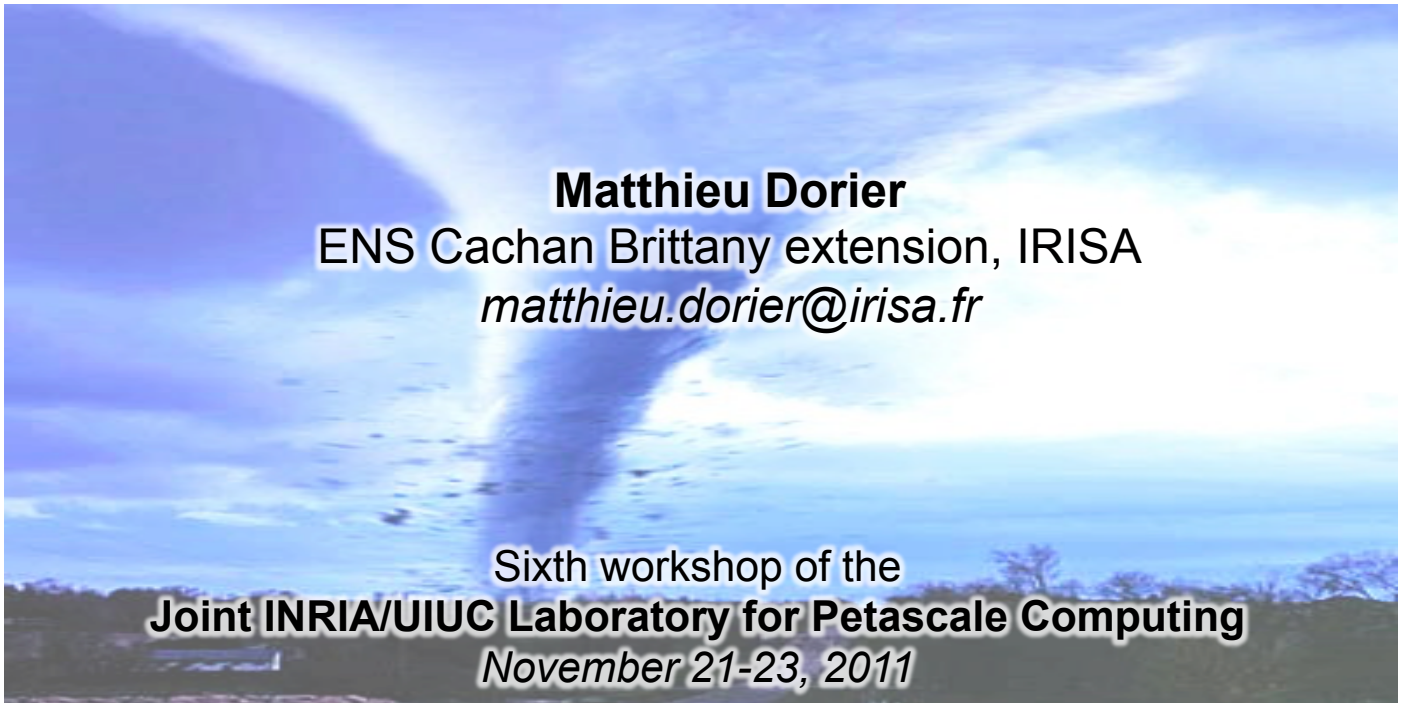
Joint Laboratory for Petascale Computation

Update on Damaris:
Making CM1 scaling linearly up to 10K cores



Matthieu Dorier
ENS Cachan Brittany extension, IRISA
matthieu.dorier@irisa.fr

Sixth workshop of the
Joint INRIA/UIUC Laboratory for Petascale Computing
November 21-23, 2011

Outline

- Recall on the motivation
- Introduction to Damaris
- New results on Kraken with CM1
- Work in progress: visualization
- Conclusion

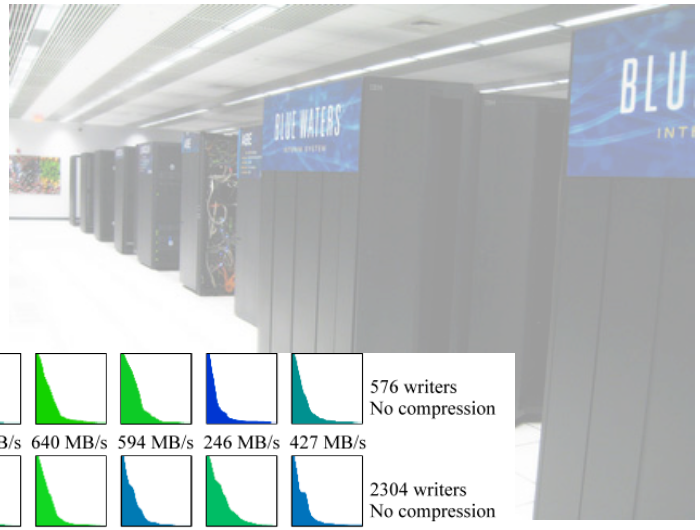
Outline

- **Recall on the motivation**
- Introduction to Damaris
- New results on Kraken with CM1
- Work in progress: visualization
- Conclusion

Motivations: I/O in HPC

- **Large-scale climate simulations on Blue Waters** (e.g. Georges Bryan's Cloud Model 1)

- **With great powers comes ~~great responsibility~~ more data, always more data**:
  - Terabytes of new data every minute;
  - More I/O jitter, more unpredictable run time;
  - More communication, higher pressure on I/O servers;
  - More difficulties to read back, analyze, visualize…

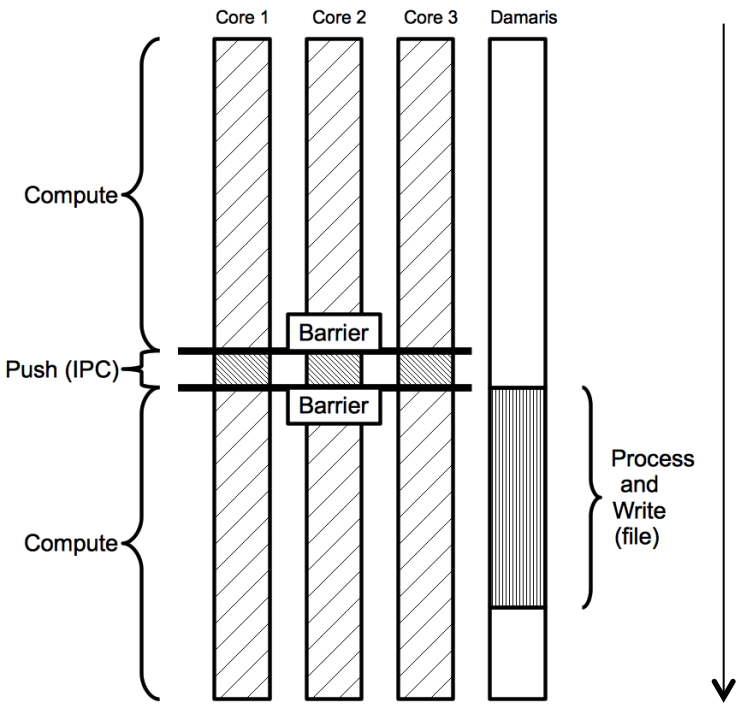


- Requirement for **new I/O approaches** allowing **end-to-end scientific process** on the same supercomputer, with **negligible I/O and data processing overhead**.

Outline

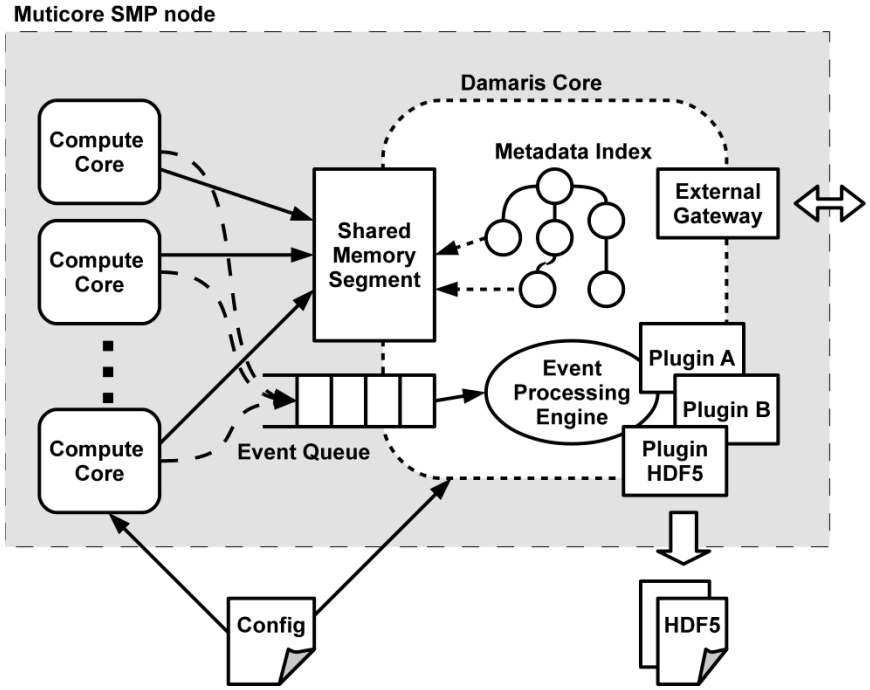
- Recall on the motivation
- **Introduction to Damaris**
- New results on Kraken with CM1
- Work in progress: visualization
- Conclusion

Damaris: overview



- **On multicore SMP nodes,** using all the cores for computation may **not** lead to the best **performance.**
- Use **service-dedicated cores** on each SMP node to…
  - **Perform I/O, data compression, filtering…**
  - **Data analytics and visualization**

Damaris: overview



- **On multicore SMP nodes,** using all the cores for computation may **not** lead to the best **performance.**
- Use **service-dedicated cores** on each SMP node to…
  - **Perform I/O, data compression, filtering…**
  - **Data analytics and visualization**

Design principles

- Uses **Shared-memory** between computation cores/dedicated cores
- **Avoids copy of data** as much as possible
- **Describes** the data externally
- Provides a **plugin system** to adapt to the user's needs

Damaris: current state

- **Version 0.3.1** available at ***http://damaris.gforge.inria.fr*** along with **tutorials** and **documentation**;

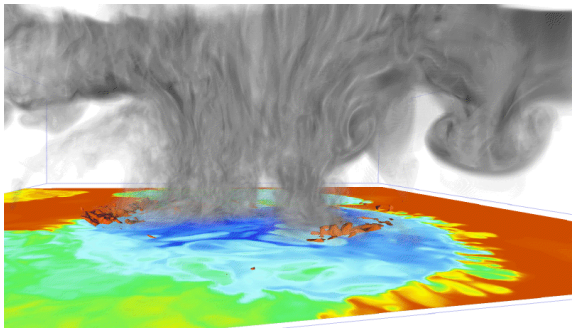
- **Works** on **Linux** (including **Linux Cray**);
- Adapts to simulations written in **C, C++** and **Fortran;**
  - Extremely simple API (write, allocate, signal);
  - Helper functions to integrate in **MPI applications** (communicators splitting, etc.);
- Uses **XML** files to describe the data and the plugins;
  - Along with an XSD validation schema;
- Plugins can be written in **C++** and in **Python**
  - Allows to use Python libraries such as **NumPy, SciPy, Matplotlib, YT**, … to perform data analysis and visualization.
  - Allows to change your plugins very easily by changing a **script file**, without recompiling, even without shutting down the application.

Outline

- Recall on the motivation
- Introduction to Damaris
- **New results on Kraken with CM1**
- Work in progress: visualization
- Conclusion

The Kraken Cray XT5

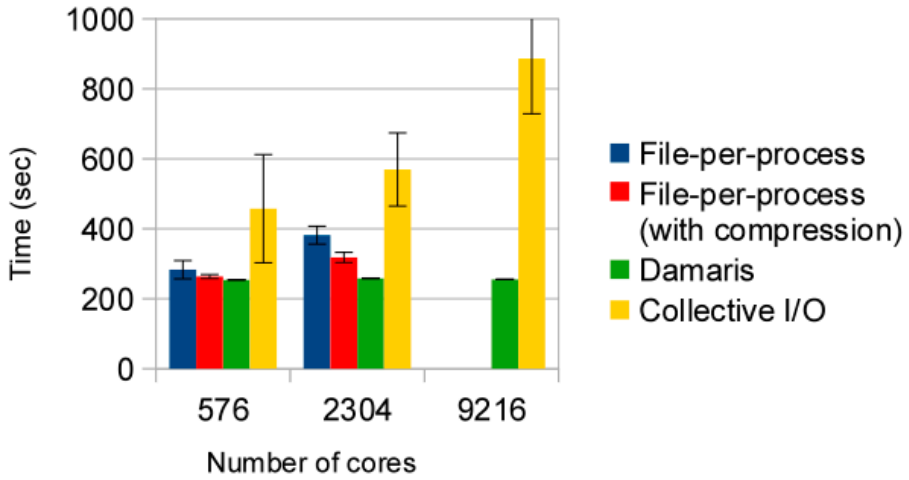
- **CM1: Cloud Model 1, one of the targeted BlueWaters application**
- **Kraken:**
  - **Currently 11th in the Top500 (1,17 PetaFLOPs)**
  - **12 cores per node (112,896 cores), 16 GB local memory per node**
  - **Lustre parallel files system: 48 OSS (160 OST), 1 MDS**



- **Note:** just like Blue Waters, Kraken is provided by Cray, has a Lustre file system and works with Cray Linux Environment. Moreover CM1 is a targeted BW application. Results are thus highly relevant with regard to expected I/O behavior on Blue Waters.

Results on Kraken with CM1

- **File-per-process:** creates to many files, becomes intractable starting from 2K cores;
- **Collective I/O:** imposes huge I/O overhead (70% time spent in I/O), does not allow compression;
- **Damaris:**
  - 70% improvement with 9216 cores
  - No more I/O jitter
    ➔ Predictable run time
  - 12 times fewer files, no synchronization
  - I/O scheduling
    ➔ 15 times higher throughput
  - Overhead-free compression (600%)



CM1 weak scaling on Kraken

Outline

- Recall on the motivation
- Introduction to Damaris
- New results on Kraken with CM1
- **Work in progress: visualization**
- Conclusion

In-situ visualization with Damaris

- **Goal: shorten the path from the running simulation to the retrieval of relevant scientific results, avoid moving data across different machines.**

- **Automate image generation using Python+MatplotLib, YT, SciPy**
  - Data published by the simulation to Damaris is visible to Python as a NumPy array;
  - NumPy can transform the data, perform diagnosis, compute images using MatplotLib;
  - By analyzing data, Damaris can select and draw what is interesting for the scientist.

- **Interactive visualization using VisIt**
  - Visualization on demand: connecting VisIt to the running simulation WITHOUT perturbing it;
  - Damaris acting as a rendering engine (e.g. use of NVIDIA's GPU on Blue Waters' compute nodes);
  - Capability to re-load old data to perform multi-simulations comparison.

Outline

- Recall on the motivation
- Introduction to Damaris
- New results on Kraken with CM1
- Work in progress: visualization
- **Conclusion**

Conclusion

- **Damaris has proven to be very efficient on Kraken**
  - 70% speedup on 9216 cores using the CM1 application;
  - 15 times higher throughput, 600% overhead-free compression, I/O jitter hidden;
  - CM1 now scales perfectly well, as opposed to using standard I/O approaches.
  - Recall: Kraken is very similar to Blue Waters' new Cray architecture.
- **Work in progress targets in-situ visualization**
  - Using Python libraries to ease the design of visualization plugins
  - Using VisIt for interactive visualization
- **Future work**
  - Simplifying end-to-end scientific process using a configurable workflow from scientific simulations to visualization and analysis tools.

Conclusion

- **Damaris has proven to be very efficient on Kraken**
  - 70% speedup on 9216 cores using the CM1 application;
  - 15 times higher throughput, 600% overhead-free compression, I/O jitter hidden;
  - CM1 now scales perfectly well, as opposed to using standard I/O approaches.
  - Recall: Kraken is very similar to Blue Waters' new Cray architecture.
- **Work in progress targets in-situ visualization**
  - Using Python libraries to ease the design of visualization plugins
  - Using VisIt for interactive visualization
- **Future work**
  - Simplifying end-to-end scientific process using a configurable workflow from scientific simulations to visualization and analysis tools.

Thank you, questions?