Joint Laboratory for Petascale Computation

In Situ Data Analysis and Visualization: Results with *Damaris* and Application to Ensemble Simulations

Matthieu Dorier – ENS Cachan Brittany, IRISA
Catalina Nita – U. Politehnica Bucuresti, INRIA

9th workshop of the Joint Lab for Petascale Computing
Lyon, June 2013

Joint work with: Roberto Sisneros (UIUC). Dave Semeraro (UIUC). Gabriel Antoniu (INRIA). Tom Peterka (ANL)

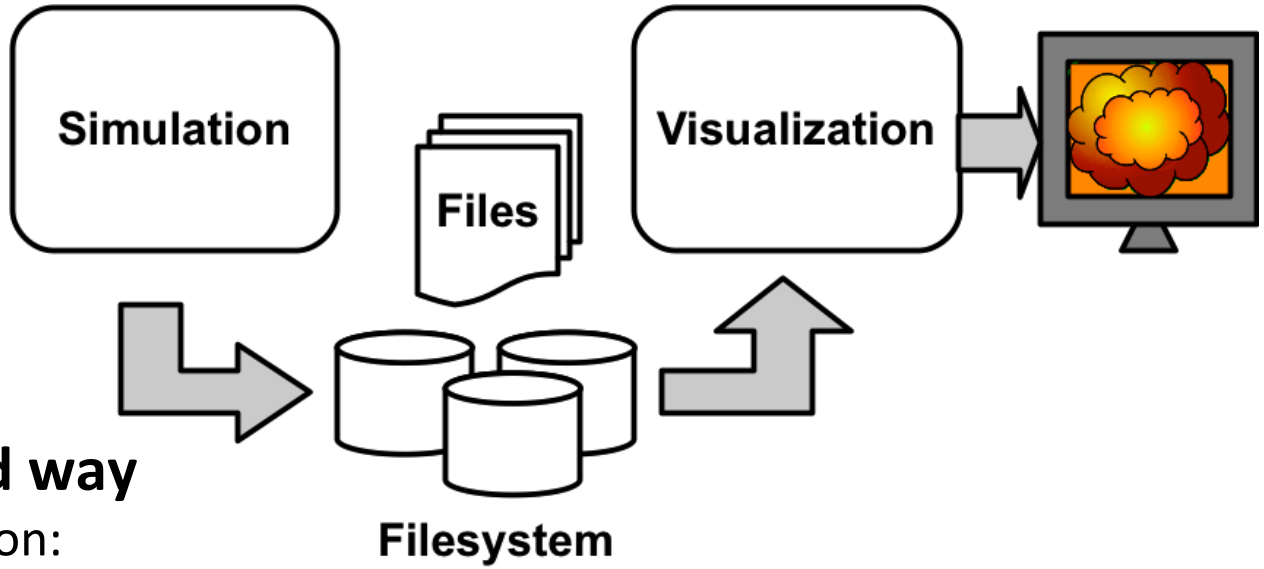
Outline

- In situ analysis: why and how?
- Results with ***Damaris***, CM1 and Nek5000
- In situ analysis of ensemble simulations
- Conclusion

Outline

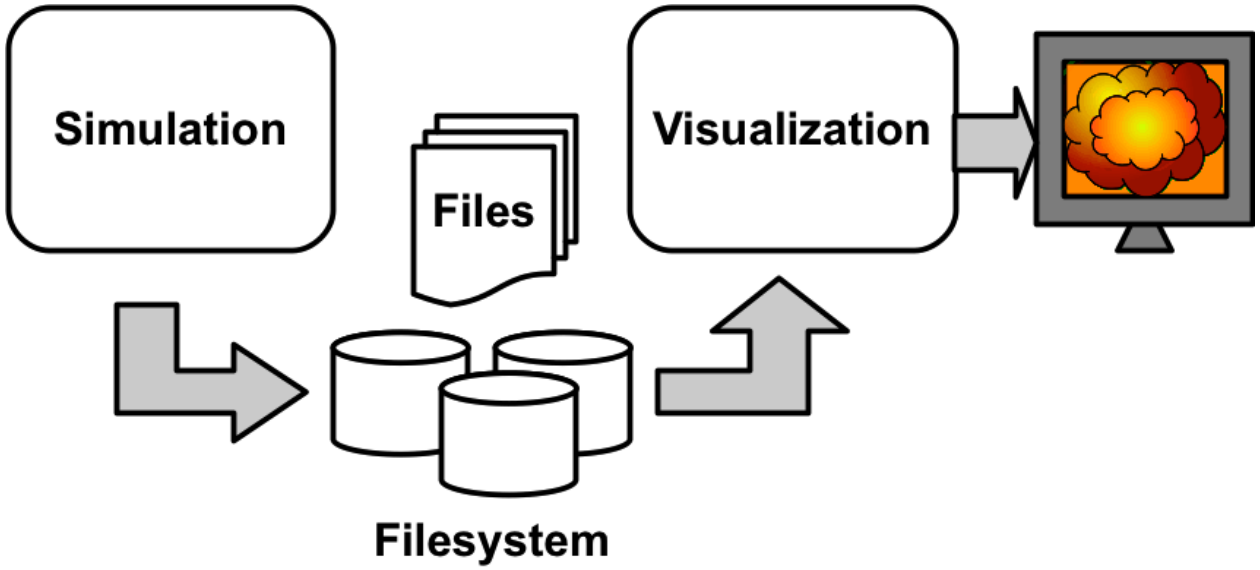
- In situ analysis: why and how?
- Results with ***Damaris***, CM1 and Nek5000
- In situ analysis of ensemble simulations
- Conclusion

In Situ analysis: **Why?**



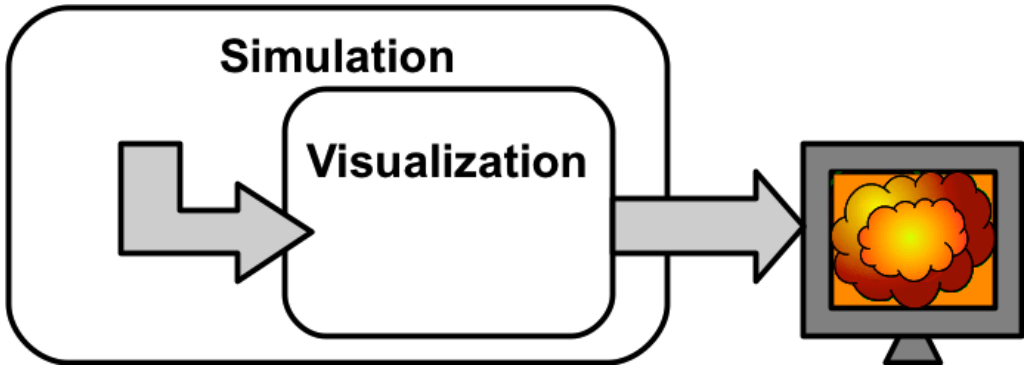
- **The old fashioned way**
  - Offline visualization:
    - Run your simulation for days
    - Write a bunch of files periodically, using HDF5, NetCDF, etc.
    - Move the files to an analysis cluster
    - Analyze your data
    - ~~Find something scientifically relevant~~ notice the simulation didn't behave as expected

In Situ analysis: **Why?**



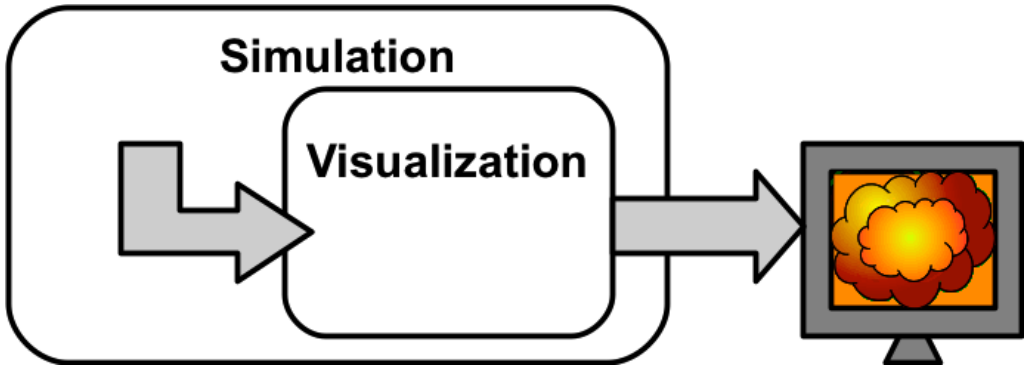
- **Motivations**
  - I/O becoming a bottleneck: need to drastically reduce storage demands
  - Migrations of data to a visualization cluster become intractable
  - Need to adapt the output format
  - Longer, more complex simulations: need to reduce the time-to-insight
  - Visualization software also suffer from the I/O bottleneck

In Situ analysis: **Why?**
Earlier analysis = Faster science



- **In situ visualization:**
  - Bypassing storage
  - Integrating visualization algorithms in the code of the simulation
    - Benefits from data locality, reduced data movements
    - Benefits from already deployed resources
  - Send results (images) to the user
    - Direct and immediate access to the results
    - Ease simulation diagnosis

In Situ visualization: the challenges



- **In situ visualization should**
  - Be efficient and scalable
  - Efficiently use resources co-located with the simulation
  - Be easy to integrate in existing simulations
  - Be adaptable to different simulations and visualization software

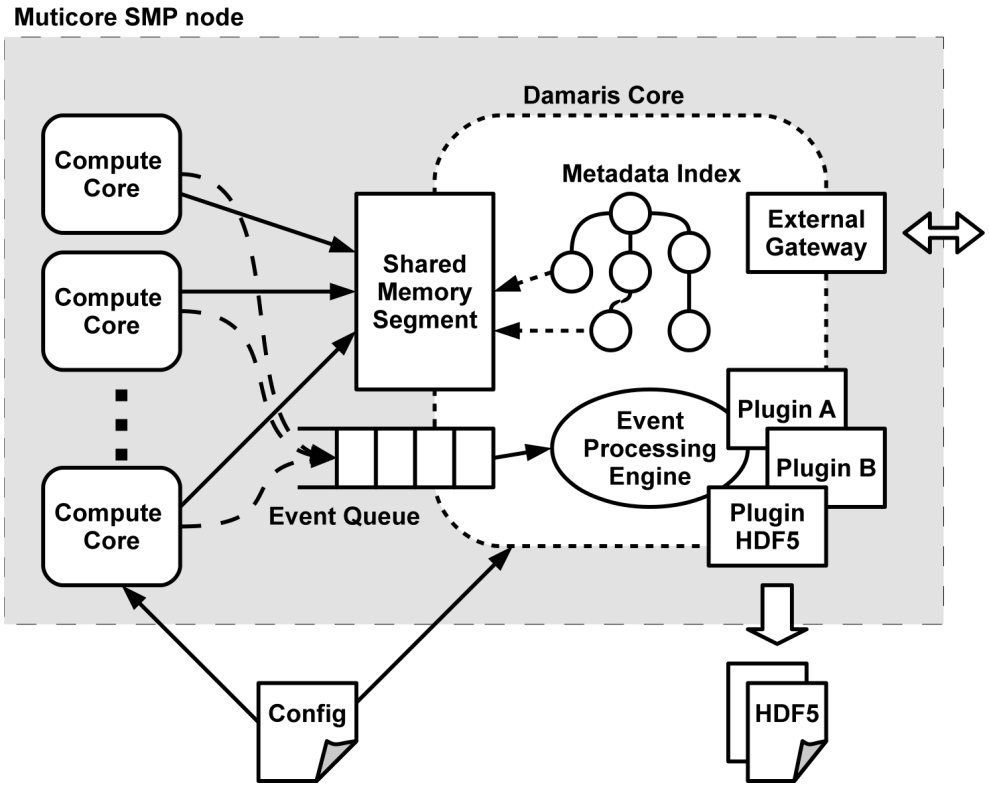
In Situ analysis: **How?**
The "traditional" in situ approach

- Time-partitioning
  - Simulation periodically stops
  - All cores are leveraged to do visualization

OR

- A better way: space-partitioning
  - Using dedicated cores
  - Perform visualization asynchronously

In Situ analysis: **How?**
Recall on the ***Damaris*** approach



- **Damaris**
  - Dedicated cores for data management
  - Shared-memory based communication model
  - Plugin system for data processing
  - XML description of data
  - Already proven efficient for I/O

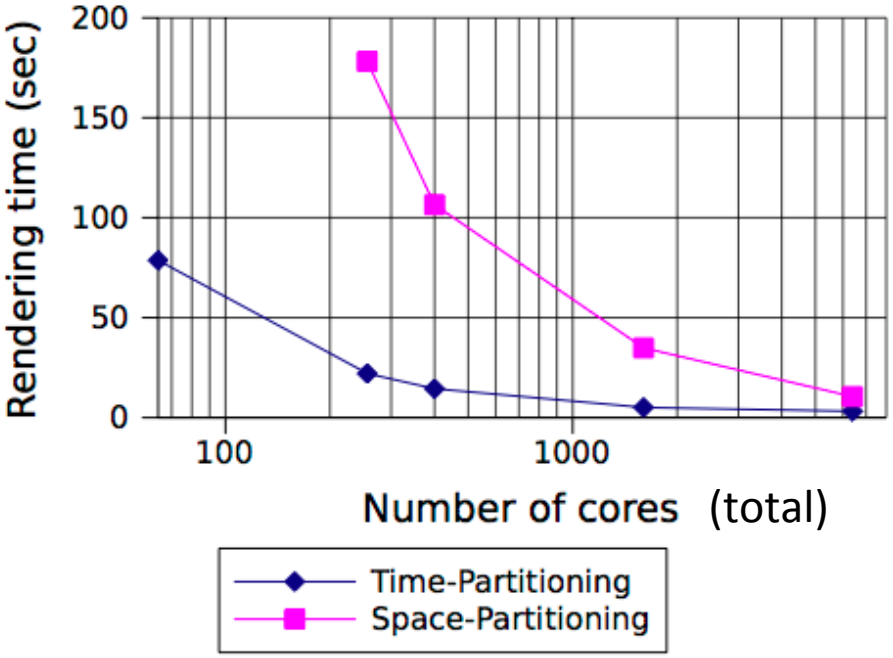
Damaris addresses all the challenges

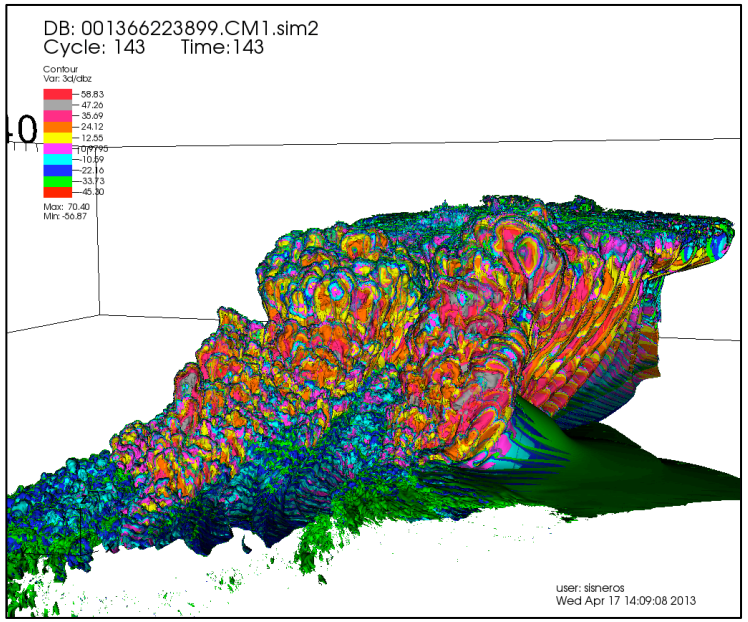
- **In situ visualization should**
  - Efficiently use resources co-located with the simulation
    - => Damaris uses of shared memory
    - => Damaris uses dedicated cores to reduce the impact on run time
  - Be easy to integrate in existing simulations
    - => Damaris uses a simple description of data in XML, including visualization-related structures (meshes, curves, domains…)
  - Be adaptable to different simulations and visualization software
    - => Damaris can be connected to VisIt (so far)
    - => Damaris has a plugin system accepting C++ codes or Python scripts
    - => Damaris+VisIt can even be used interactively!
  - Be efficient and scalable
    - => Results with Damaris presented in the next section

Outline

- In situ analysis: why and how?
- Results with ***Damaris***, CM1 and Nek5000
- In situ analysis of ensemble simulations
- Conclusion

In situ iso-surface of CM1

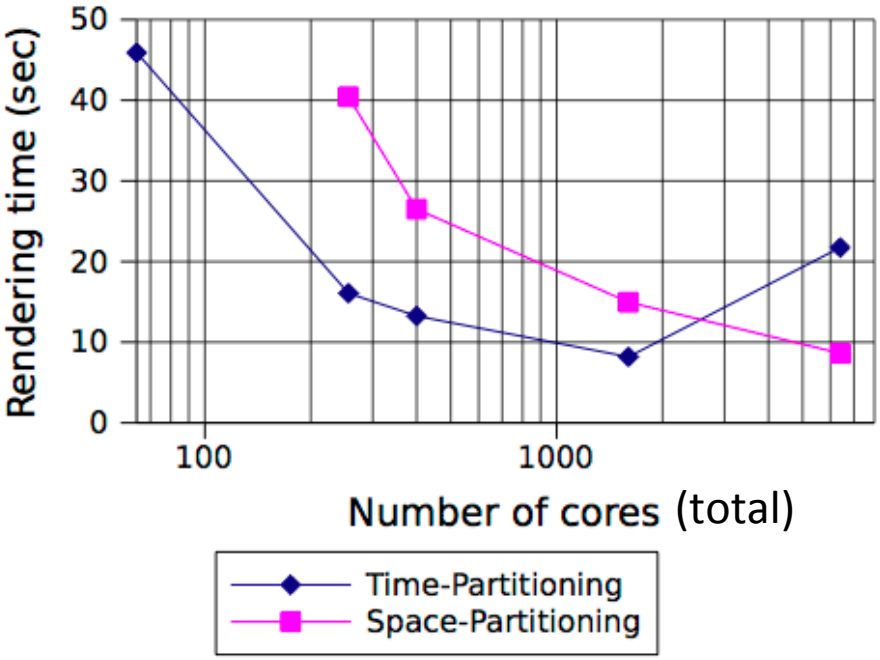


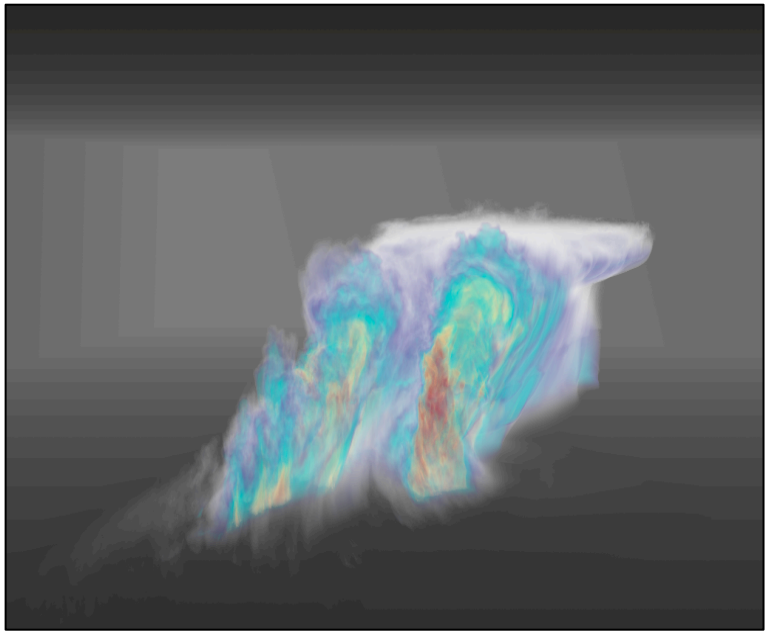


Experiments on Blue Waters with CM1 (up to 6400 cores / 400 nodes):

Using 400 dedicated cores is as efficient as using all 6400 cores!

In situ ray-casting of CM1

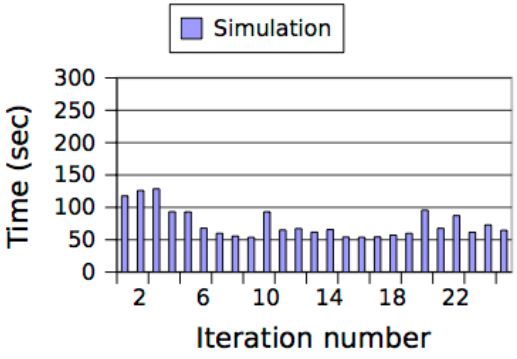
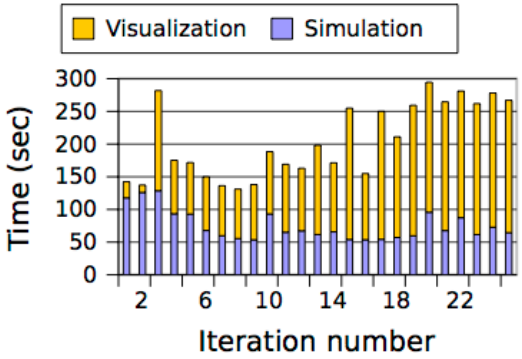
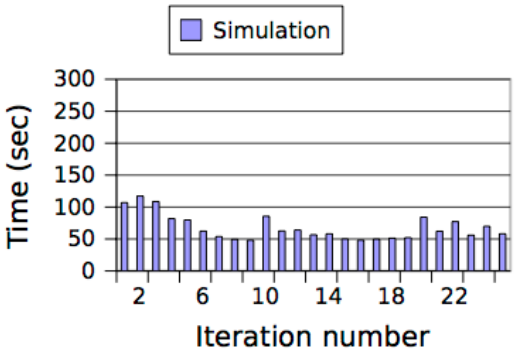
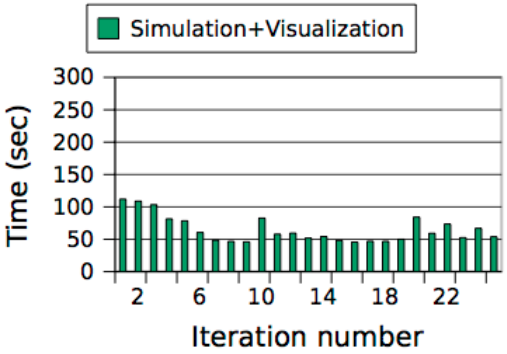




Experiments on Blue Waters with CM1 (up to 6400 cores / 400 nodes):

Using 400 dedicated cores is 2x more efficient than using all 6400 cores!

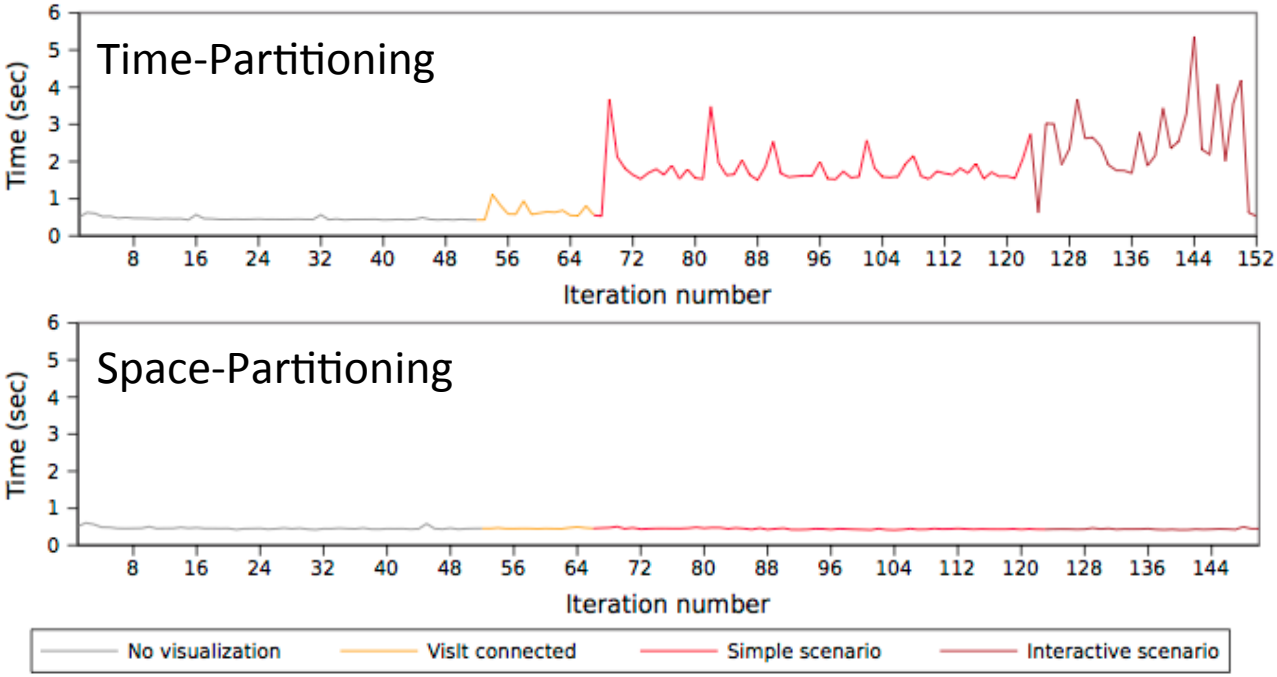
Run time at large scale

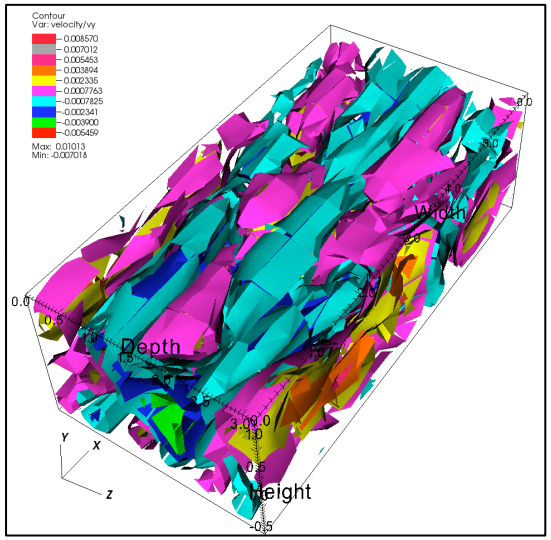
	Without Visualization	With Visualization
Time Partitioning		
Space Partitioning		

Experiments on Grid'5000 with Nek5000 (912 cores):

Using Damaris completely hides the performance impact of in situ visualization

Impact of interactivity





Experiments on Grid'5000 with Nek5000 (48 cores):

Using Damaris completely hides the run time impact of in situ visualization, even in the presence of user interactivity

Want a live demo? (just kidding!)

- Checkout a ***demo video*** at

  http://damaris.gforge.inria.fr

- Results submitted to LDAV 2013
  - Available as a research report: RR-8314 – *A nonintrusive, adaptable and user-friendly in situ visualization framework*. Matthieu Dorier, Roberto Sisneros, Tom Peterka, Gabriel Antoniu, Dave Semeraro
- Poster at the PhD forum of IPDPS 2013

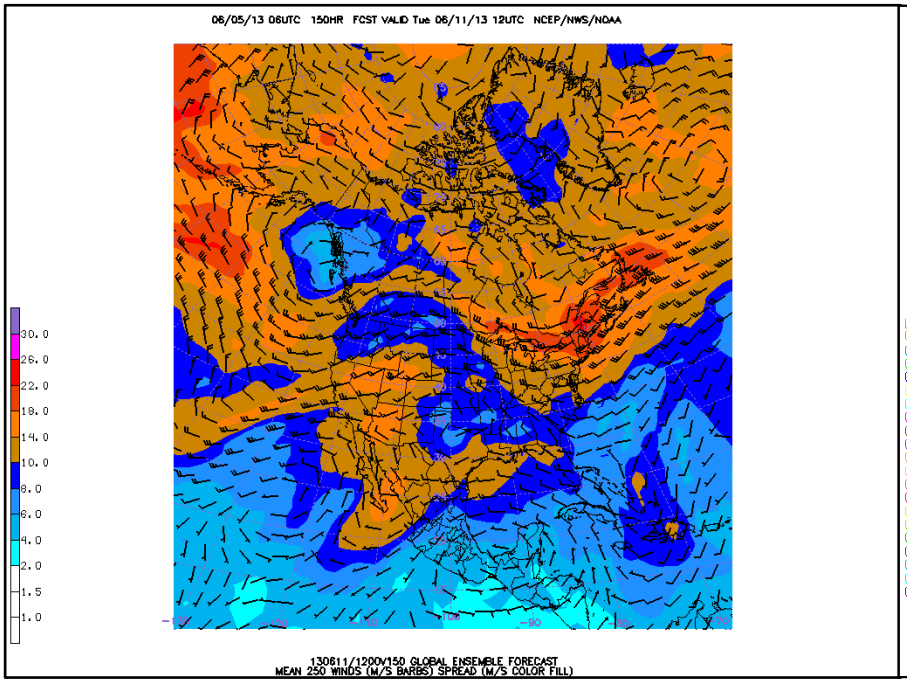
Outline

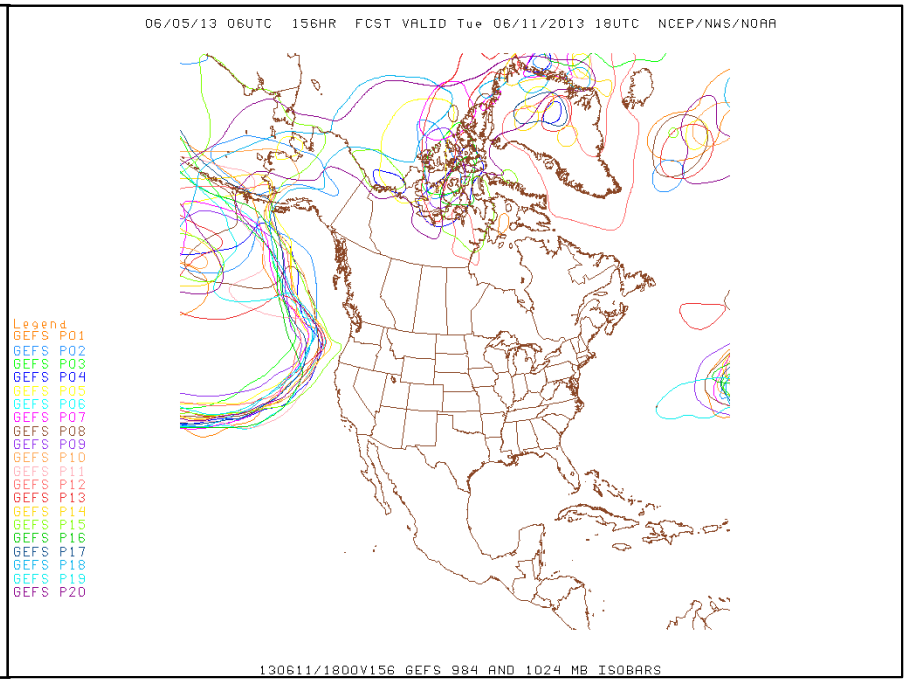
- In situ analysis: why and how?
- Results with ***Damaris***, CM1 and Nek5000
- In situ analysis of ensemble simulations
- Conclusion

Ensemble simulations

- Very common use of large-scale simulations
- Many runs,
  - Varying the parameters (most common, parameterized study)
  - Varying the model (model comparison)
  - Varying the numerical method (5^{th} to 6^{th} order, for instance)
  - Varying the scale (less likely)
- Important in forecasting:
  - Small or medium scale (1K – 10k cores), short runs (min – hours)
  - Important variations of the results
    → Model is not robust, forecast unreliable
  - Small variations
    → Model is robust, forecast reliable

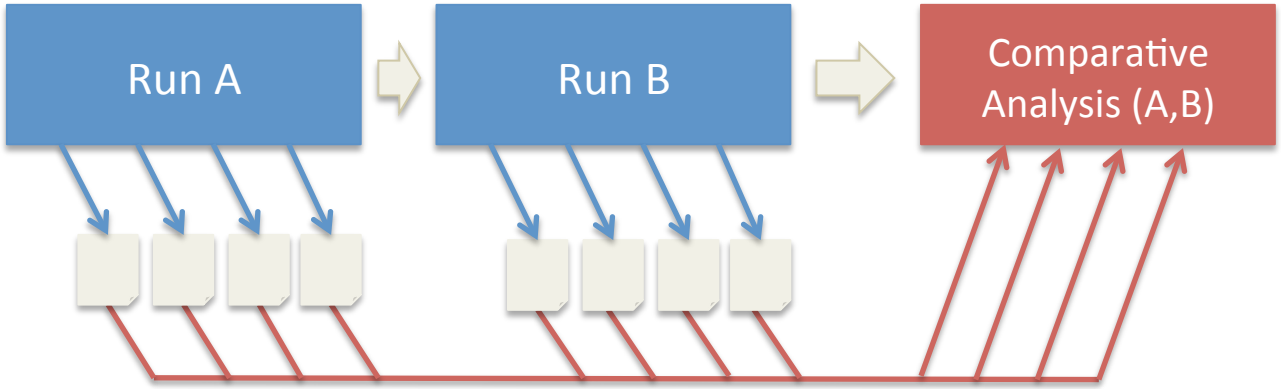
Examples in climate





Model Analysis and Guidance, Global Ensemble Forecast System - (GEFS-SPAG), National Weather Service (http://mag.ncep.noaa.gov/)

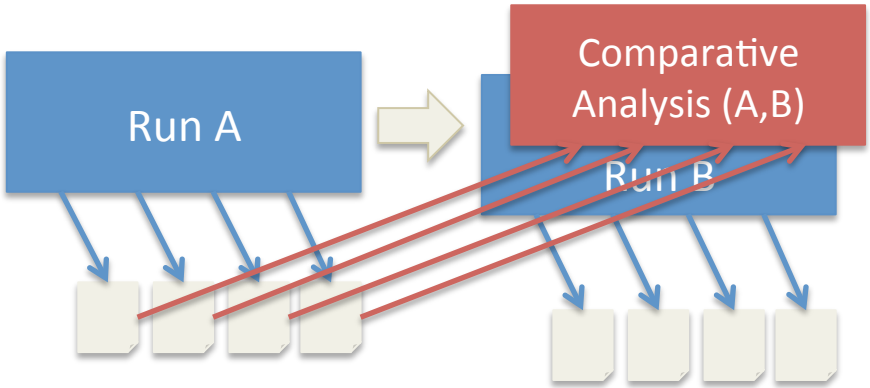
Analysis of Ensemble simulations: The "classical" approach



Example with 2 runs

1. Complete run A, simulation writes periodically
2. Complete run B, simulation writes periodically
3. Reload data from A and B into an analysis pipeline

Analysis of Ensemble simulations: The "in situ" approach



Example with 2 runs

1. Complete run A, simulation writes periodically
2. Start run B
   a. At iteration x of run B, reload data from iteration y = f(x) of run A
   b. In situ analysis of in-memory data from B and loaded data from A

Challenges of in situ analysis for ensemble simulations

- In situ analysis has been proposed as an alternative to storage
  - Ensemble simulations still require storage
- How to perform in situ analysis of finished runs?
  - Reloading data from previous runs within the current run
- How to efficiently manage resources when reloading data?
- How to make a model understand data from another model?
- How to accommodate for change of scales?
- How to provide an efficient and simple interface for the analysis of many runs?

Using Damaris for ensemble simulations

- Damaris already provides…
  - ✓ Easy integration in simulations (simple API, XML configuration)
  - ✓ Asynchronous data analysis using dedicated cores
  - ✓ Plugin system for in situ analysis
  - ✓ A connection to visualization tools (VisIt)
- Remaining challenges to address:
  - ❑ Building a metadata-rich persistency layer for Damaris
  - ❑ Making Damaris understand multiple models
  - ❑ “Smart” data management, “smart” resource usage

Outline

- In situ analysis: why and how?
- Results with ***Damaris***, CM1 and Nek5000
- In situ analysis of ensemble simulations
- Conclusion

Conclusion

- Damaris provides nonintrusive, efficient, and user-friendly in situ visualization framework
  - Evaluated with CM1 and Nek5000
  - On many platforms including Blue Waters
- Ensemble simulations:
  - Many challenges when it comes to efficiently write, re-load, compare data in a transparent and user-friendly manner
  - Damaris provides a good basis to address some of these challenges

Questions?