A propos du centre ou de la direction fonctionnelle

Contexte et atouts du poste

The increasing performance gap between computing elements and permanent storage in supercomputer re-shuffles the cards for optimization of simulation codes. Peripheral concerns such as I/O and post-processing data analytics that used to be developed independently with little care for performance are becoming more and more critical to optimize. Their integration in the main code can however come at a high cost in term of maintainability and performance portability. Simulations are also more and more coupled with data analytics processes when coupled with other codes, scientific instruments or sensors, as for Digital Twins applications. These analytics, like deep learning, can require significant compute resources leading to complex multi-level parallelizations and data stagings. Today’s approaches are mostly based on explicit and manual code assembly that are time consuming to write, intrusive in the simulation code and lack flexibility.

Building workflows that achieve high performance executions while keeping programming simple requires a framework enforcing modularity through a good separation of concerns, and automatic assembly processes for negotiating task placement and data extraction.

In situ data analytics is an approach where the simulation writes raw results to permanent storage that is later analyzed with independent software. By construction, this ensures a good separation of concern between the main simulation code and data analytics. This approach does however lead to severe performance issues where simulation is limited by the disk bandwidth and can not leverage supercomputers computing power.

Post-mortem data analytics is an approach where the simulation processes when coupled with other codes, scientific instruments or sensors, as for Digital Twins applications. These analytics, like deep learning, can require significant compute resources leading to complex multi-level parallelizations and data stagings. Today’s approaches are mostly based on explicit and manual code assembly that are time consuming to write, intrusive in the simulation code and lack flexibility.

In situ processing consists in using the same computing resources for data analytics and simulation. This reduces the performance impact as the data can be reduced before it is written to disk. Many variants of in situ data analytics exist ranging from a) sequential where the simulation code is interrupted for data processing, b) dedicated core where some cores of each node are reserved for analytics or even c) task-based where the runtime dynamically interleaves fine grain simulation and analytics tasks[7]. Depending on the approach chosen, memory overhead and cache trashing can however still badly impact performance. The approach is also typically rather invasive in the code and can be very complex to implement if the data distribution used in the analysis differs from that of the simulation proper. The data analytics can also be executed on dedicated nodes, often called staging nodes, distinct from the simulation nodes. This approach remove load from the simulation nodes that do not need to share their resources (compute units, cache, memory) with the analytics, but these node may become underused and it may require costly data transfers towards these staging nodes. Some frameworks like ADIOS exclusively support one option. Other frameworks like FlowVR, Damaris or Decaf support in situ and in transit processing, but task placement and data movements need to be manually defined. The choice is often the results of a multi-criteria tradeoff. The in transit storage system DataSpaces has recently investigated several approaches for the automatic data analytics placement, but on the staging nodes only [14, 2].
The goal for this Ph.D. thesis is to investigate a modular workflow design and automatic assembly processes capable of achieving high performance executions while keeping programming simple. This approach will have to minimize the impact on simulation code both in terms of performance and software complexity by ensuring a good separation of concerns. Based on some constraints expressed by the user, the system should be able to automatically assemble the different requested data processing steps, with in situ or in transit task placement and efficient data extraction and redistributions.

The proposed approach will have to take into account the specificity of data processing workloads for HPC: strong memory coupling of some operations, new deep memory hardware, different requirements (memory, CPU, ...) of different operations, etc. It will have to handle new types of workload expected to appear, including machine-learning based ones for example. It will have to handle workflow including logic based on the data produced by simulation to activate or disable some analytics steps. To reach these goals, it will likely have to extend approaches like lazy data copies, contract based data extraction [13], \(N \times M\) data redistributions [4].

**References**


[3] Matthieu Dorier, Matthieu Dreher, Tom Peterka, Justin M. Wozniak, Gabriel Antoniu, and Bruno Raffin. Lessons Learned from Building In Situ Coupling Frameworks. In Workshop on In Situ Infrastructures for Enabling Extreme-scale Analysis and Visualization (ISAV’15)- Held in conjunction with SC15, Austin, November 2015. ACM.


Compétences
Taste for research, curiosity, creativity, good programming skills and system knowledge.

To be eligible a candidate needs to justify for 2 years at least of studies outside of France.