D7.2

EESI2 Second Annual Report

2014 Update Vision & Recommendations

D7.2

2014 Update

Vision & Recommendations

CONTRACT NO	EESI2 312478
INSTRUMENT	CSA (Support and Collaborative Action)
THEMATIC	INFRASTRUCTURE

Due date of deliverable:

Actual submission date: 30th July 2014

Publication date:

Start date of project: 1 September 2012 Duration: 34 months

Name of lead contractor for this deliverable: TOTAL SA, Philippe RICOUX

Revision: V1.0

TABLE of CONTENTS

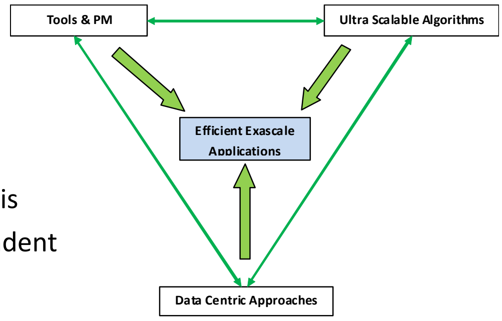
Summary

The document presents the update, from European Exascale Software Initiative Experts, of the roadmap/vision and the recommendations for critical R&D challenges to be funded in order to develop efficient applications at Exascale computing.

These vision and recommendations are focused on the issues and challenges at Extreme Computing and Extreme Data, which are much more complex than classical HPC and which cannot be tackled only by pursuing the development of known HPC technologies and tools.

The roadmap towards the implementation of efficient Exascale applications and the consecutive recommendations are gathered in three large pillars:

- Tools & Programming Models
- Ultra Scalable Algorithms
- Data Centric Approaches



Note that the Data Centric vision is very new in Europe but is essential for approaching the ultra complex and interdependent challenges of Extreme Computing and Extreme Data.

As already advised by EESI2 (and by US DOE), Exascale requires a new and different approach compared to classical HPC. There is an urgent need for specific and disruptive R&D programs targeting Exascale software.

All EESI2 recommendations are aligned with this approach: they are coherent and aimed at assuring the efficiency of tools and applications at Exascale.

European Exascale Software Initiative 2014 Recommendations

Section I
Vision & Introduction

The EESI2 Initiative is clearly oriented toward the development and implementation of efficient Exascale applications, algorithms and software for enabling the emergence of a new generation of data intensive and extreme computing applications. The driver for this is the disruptive nature of Exascale computing with its potential for massive return on investment by addressing huge economic, societal and scientific challenges. New thinking is required to develop new programming models, new algorithms, new tools, new data processing methods ... not only bigger than the present ones, which will remain useless, but far beyond the required innovation. Exascale creates fundamental new opportunities, but it also brings fundamental new challenges.

The EESI vision is in coherence with International R&D programs funded on Exascale in particular in the US and Japan. Europe clearly has strengths (applications, scalable algorithms, couplers...) but also is clearly late on some Exascale key issues (languages, programming tools ...). EESI is very active in the International Initiative Big Data & Extreme Computing (BDEC) towards Exabytes and Exaflops from EU, US and Japan.

The principles underlying the recommendations are:

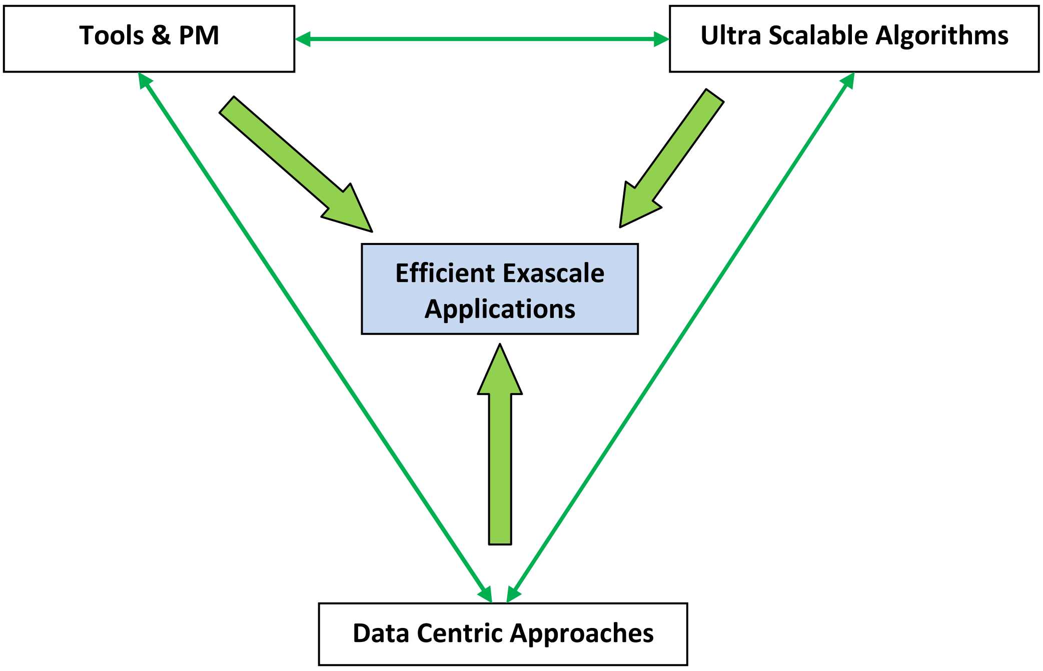
- Exascale is not only a “bigger HPC”. There is an urgent need for new specific and disruptive ultra-scalable improvements in order to realise its full potential

- Extreme computing and Extreme Data should be tackled simultaneously.
  At Exascale, Extreme Computing and Extreme Data (or Big Data) are intrinsically linked since supercomputers become mandatory to analyze efficiently huge flows of data generated by large scale instruments or by massive complex simulations. Exascale applications will be efficient only through developments by multidisciplinary teams, optimizing the interactions between architecture (nodes, cores, memories, interconnect, power, resilience ...), algorithms (programming, ultra scalable numerical methods, asynchrony, fault tolerance ...), and applications (discretization of problems, engineering tools, data processing ...).
- It is urgent that the EU funds large projects focusing on complex specific Exascale challenges in particular in the domains where the EU has established strengths. Exascale 2020 (or 2022) is approaching rapidly and the Exascale issues are really challenging.

In this context, and clearly in accordance with the EESI roadmap towards Exascale, the experts of the EESI initiative have developed a set of coherent recommendations. Each recommendation proposes an initiative to tackle one of the key issues. As a whole the initiatives comprise a unique and coherent objective: to supply all necessary actions towards efficient Exascale software and applications.

The EESI 2014 recommendations can be grouped into three pillars: Tools and Programming Models including resilience, Ultra Scalable Algorithms, and Data Centric Approaches. Each EESI 2014 recommendation is detailed in the next section of this document.

Exascale comprises both Exaflops and Exabytes. Power efficiency, computing and overall performance are heavily dependent on data locality and data movement. Therefore, Data Centric Approaches to Extreme Computing should be in the core of the Exascale strategic roadmap.



In the **Tools & Programming Models pillar**, recommendations concern programming models and methods, heterogeneity management, software engineering and cross-cutting issues like resilience, validation and uncertainty quantification with a strong focus on the specificity of Exascale in these domains.

The following recommendations are proposed for funding by the European Commission:

- High productivity programming models for Extreme Computing
- Holistic approach for extreme heterogeneity management of Exascale supercomputers
- Software Engineering Methods for High-Performance Computing
- Holistic approach to resilience
- Verification Validation and Uncertainties Quantifications tools evolution for a for better exploitation of Exascale capacities

In the **Ultra Scalable Algorithms pillar** recommendations concern specific and disruptive algorithms for Exascale computing, taking a step-change beyond "traditional" HPC. It will lead to the design and implementation of extremely efficient scalable solvers for a wide range of applications.

The following recommendations are proposed for funding by the European Commission:

- Algorithms for Communication and Data-Movement Avoidance
- Parallel-in-Time: a fundamental step forward in Exascale Simulations (disruptive approach)

The **Data Centric pillar** links Extreme Computing and Extreme Data. For the transition to Exascale, current data life cycle management techniques must be fully rethought, as described in the first joined document "Software for Data Centric Approaches to Extreme Computing" which is more a vision than a concrete recommendation. This pillar gathers together key strategic issues for Exascale applications which are not enough addressed until now in Europe.

Ensuing from the EESI holistic vision of "Software for Data Centric Approaches to Extreme Computing", the following recommendations, all new at European level, should be supported and funded by European Commission:

- Towards flexible and efficient Exascale software couplers (direct or not, exchange of big data)
- In Situ Extreme Data Processing and better science through I/O avoidance in High-Performance Computing systems
- Declarative processing frameworks for big data analytics, extreme data fusion e.g. identification of turbulent flow features from massively parallel Exaflops and Exabytes simulations

This document reflects the vision and the recommendations of more than 120 worldwide experts in scientific and industrial applications and in all sciences and technologies required for Exascale.

It takes into account existing strengths in the European HPC and ICT communities. It addresses key strategic areas for which there is an urgent need for funded programs of work, beyond the classical and conservative HPC approaches, to develop and improve European competitiveness and to achieve leadership.

Not all of these recommendations are at the same level of generalization but they are complementary and linked to each other by their global common objective: enabling the emergence of a new generation of intensive data and extreme computing applications. Some of them are fully disruptive; all need to go beyond known HPC technologies and methods.

All these recommendations should be supported and funded.

Some of these recommendations could be addressed in part by being strategic themes for new Centres of Excellence (CoEs).

The section below gives full details of the EESI 2014 recommendations.

Section II
Description of the EESI 2014 Recommendations

Recommendations of the Tools & Programming Models pillar:

Tools&PM

- High productivity programming models for Extreme Computing
- Holistic approach for extreme heterogeneity management of Exascale supercomputers
- Software Engineering Methods for High-Performance Computing
- Holistic approach to resilience
- Verification Validation and Uncertainties Quantifications tools evolution for a for better exploitation of Exascale capacities

Recommendations of the Ultra Scalable Algorithms pillar:

Ultra Scalable Algorithms

- Algorithms for Communication and Data-Movement Avoidance
- Parallel-in-Time: a fundamental step forward in Exascale Simulations (disruptive approach)

Recommendations of Data Centric Approaches pillar:

Data Centric Approaches

***Vision* “*Software for Data Centric Approaches to Extreme Computing*”**

- Towards flexible and efficient Exascale software couplers (direct or not, exchange of big data)
- In Situ Extreme Data Processing and better science through I/O avoidance in High-Performance Computing systems
- Declarative processing frameworks for big data analytics, extreme data fusion e.g. identification of turbulent flow features from massively parallel Exaflops and Exabytes simulations

High productivity programming models

Tools&PM

Recommendation Brief Description

Exascale systems are posing manifold challenges to scientific applications due to: requirements on scalability to very large number of nodes (hundreds of thousands), heterogeneity of the nodes (different architectures, such as GPUs and general purpose CPUs, but also heterogeneity in the same architecture with big-little architectures), very strict constraints in energy and requirements on the efficient management or huge amounts of data. What is more, all the complexity of Exascale systems cannot be exposed to the application and as well it is not sustainable anymore the porting of applications every time a new architecture appears. Since the availability of applications that can exploit the features of these Exascale systems is crucial, it is necessary to be able to have **high-productivity programming models** with a high level of **programmability** that are able to express in a few lines the scientific algorithms, with high **portability achieved through high level of abstraction** of the underlying hardware and that rely on **powerful runtimes** to ensure **efficient execution** and enable **dynamic load balancing.**

Recommendation Context

Innovation cycles in high performance computing platforms are characterized by moving towards more and more complex architectures accompanied with a crucial need for efficient programmability. Just for computing resources, aspects such as larger number of resources organized in different hierarchies (hundreds of thousand nodes, several sockets per node, hundreds of cores per socket…) of heterogeneous nature (different ISA, different variability in the availability of these resources, different performance/frequency, faults …), make difficult the programming of such machines. However, it is not sustainable to write and re-write applications for new infrastructures, since the most expensive cost is the application developer and the time invested in re-writing and porting prevents to develop new functionalities. In this sense, programming models that enable to abstract the logic of the applications from the actual infrastructure details are a must. In such programming models, the application developed can focus in the problem to be solved independently of the actual architecture of the system. Examples that follow this idea are: approaches that enable sequential programming with annotations (OpenMP, OmpSs), embedded Domain Specific Languages (eDSLs) or programming languages/scripting languages that enable rapid prototyping (i.e., Python). The importance here is to provide such programming interfaces with intelligent runtimes that are able to fill the gap between the application and the actual hardware, implementing efficient methodologies that exploit the features of the systems.

Also with larger number of nodes and core count, global synchronicity is not acceptable anymore and programming models and runtimes that support local synchronizations, such as task-based approaches guided by data dependences where the typical fork-join scheme is avoided are a must. In terms of communications, the programming model should be able to support and efficiently manage non-blocking and asynchronous collective operations. The programming model is going to be optimally linked with intelligent runtime systems that are able to take into account aspects such as resource management for tasks, exploitation of the data locality, moving computation to the data and enable other optimizations such as automatic load balancing between different processes. A crucial aspect to be considered is also the inclusion of energy efficiency criteria both at the programming model and runtime decisions. The programming environment would optimally come with a set of tools that support the development and analysis of the applications.

With Europe being very well positioned in programming models, it is the right moment to fund initiatives which can unify good ideas from Europe, and get a strong European programming model.

Objectives of the Recommendation

The goal of this recommendation is to fund R&D programs in order to explore

- New approaches in task-based asynchronous execution models, being able to hide the details of the HW platform
- Automatic exploitation of parallelism enabling scalability at very large number of nodes
- Communication hiding programming in heterogeneous architectures
- Embedded Domain Specific Languages to improve productivity of HPC heterogeneous environments (prototyping programming languages or scripting languages)
- Tools for automatized detection of data- and task-dependencies for multi-threaded task based programming
- Programming environments where it is possible, for example, to interplay between compute platforms and data-bases, to design and execute workflows for high-throughput computing or multi-stage computational refinements.
- Intelligent runtimes that perform efficient resource management, exploit data locality, automatic load balancing, and that are energy aware, between other features.

Tools and implementations should demonstrate their applicability in numerical kernels, mini-apps and large scale simulations. Cases such as in situ extreme data processing were new data transformations and compressions that reduce drastically extreme raw data generated during HPC simulations are applied, are examples of the applicability of this High Productivity programming models.

Holistic approach for extreme heterogeneity management of Exascale supercomputers

Tools&PM

Design and develop new efficient HW/SW APIs for the integrated management of heterogeneous systems, near-data technologies and energy-aware devices, to enable exascale-ready applications.

Scientific perimeter:

Energy efficiency is today a key concern in supercomputers (SC) design. Exascale SCs are expected to be feasible with a power envelope of 20MW[1] . This requires a 50x gain in energy efficiency w.r.t. the today top500 SC[2]. In addition energy cost accounts twice in the SC facility cost: as computation power and cooling costs. Disruptive technologies are needed to achieve at the same time a significant energy and performance gain.

Erreur ! Source du renvoi introuvable. shows that today supercomputers already embed HW accelerators to improve energy-efficiency and performance and there is a clear trend on increasing their usage in future systems as well as an increase in the heterogeneity of the accelerator brand and functionality.

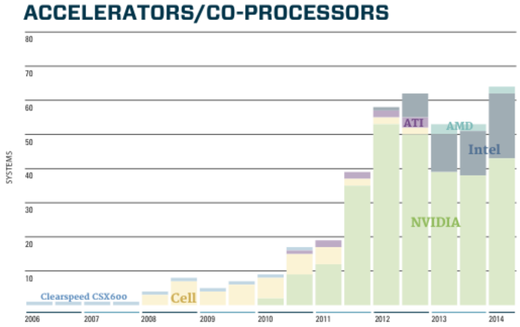


Fig. 1: Trend in heterogeneous system market share in Top500 [3]supercomputers

Coherently with the European Technology Platform for High Performance Computing (ETP4HPC) vision[4] Exascale will require heterogeneity at an unprecedented level embedding in the same system near-data, data-parallel, function specific accelerators, general purpose multicores with multiple ISA, with same ISA but multiple energy-performance trade-offs, networking accelerators as well as volatile and non-volatile memories. The progress in integration technologies (3D, 2.5D) will lead to tightly-coupled heterogeneous systems. Non-volatile memories as well as novel virtual memory sharing paradigm will lead to an unprecedented level of operating modes with performance and energy efficiency trade-offs. The entire software stack is needed to evolve to hide the complexity of this wide design space supporting the programmer as well as the system administrator.

Heterogeneous computing aims to increase of orders of magnitude the energy efficiency and performance of the computational part of the workload. Indeed general purpose computers are designed to support a large set of operations and this leads to several degrees of inefficiency on the specific operation execution. To overcome this limitation today embedded systems embed different HW accelerators to efficiently run complex and repetitive kernel patterns. Today supercomputers embed data-parallel and thread-parallel accelerators as expansion cards. GPUs and the Intel Xeon Phi are two examples of these accelerators. In addition not only complex computational kernels can have benefits from HW heterogeneity as also low complexity computational kernels run inefficiently in general purpose processing cores which are optimized for performance. This is mainly due to design overheads that need to be placed during synthesis to achieve

[1] K. Bergman, et al. Exascale computing study: Technology challenges in achieving exascale systems. Technical report, 09 2008.

[2] J. Dongarra. Visit to the National University for Defense Technology Changsha, China. Technical report, University of Tennessee, 06 2013.

[3] http://s.top500.org/static/lists/2014/06/TOP500_201406_Poster.png

[4] http://www.etp4hpc.eu/wp-content/uploads/2013/06/Joint-ETP-Vision-FV.pdf

the final peak performance. Arm big.LITTLE architecture[5] is an example in which two clusters of homogenous cores are integrated in the same die, with one cluster composed by complex A15 cortex core with nominal frequency of 1.6 GHz and the second cluster with simpler A7 cortex with nominal frequency of 1.2 GHz. The O.S. (i.e. the scheduler) can switch dynamically in between the two clusters to follow the instantaneous computational requirement.

When a large data access is required general purpose architectures became inefficient due to the limited memory and I/O bandwidth and by energy spent to move data in and out the CPU. Performance became limited by the data access latency while energy by the bit traveling distance. The growth of 3D integration technologies make possible to stack memory dies on top of each other and to integrate them in the same package with logic. In true 3D integration the logic die can be stacked at the bottom of the memory whereas on 2.5D both the memory stack and logic die can be stacked on a common silicon interposer. Unfortunately these stacks cannot be directly integrated with general purpose processors as they would lead to thermal hazards. Indeed error and refresh rate in DRAM increases with temperature. On the other side ultra-low power parallel processors which are emerging in embedded systems are characterized by simpler cores and L1 memory architectures and extremely high energy efficiency. For these reasons they are good candidates to act as programmable engine directly in the memory. These new architectures will be tailored to perform complex data-movement and data-stream/aggregation kernels directly in the memory. This will mitigate the today bandwidth and energy efficiency limits by reducing the core to memory communication to the only computational kernel results instead of the entire raw data. At the same time the computation is achieved with large data parallelism and high efficiency directly inside the memory. The development of non-volatile memory technologies will enable to share the same near-data computing principle to storage elements.

Today heterogeneous supercomputers are the first step toward the development of massively complex tightly-coupled heterogeneous systems which aim to devise HW level energy-efficiency and performance. Communication overheads as well as the increased complexity induced by the variety of design choices are the looming threats which mine heterogeneity potential gain. Hardware supported virtual memory sharing is one of the foreseen approaches to this issue.

The entire SW stack (programming models, run-time, OS and system support software) needs to be fully innovated to support programmability and efficient performance/energy usage of the large set of different resources and computational modes. HW support for efficient communication, coherency and offload control needs to be co-designed with the SW stack to exploit the energy-efficiency and performance enabled by the extreme heterogeneity. Moreover novel abstraction layers and optimization strategies must be developed to cope with the extreme heterogeneity and parallelism of Exascale systems and finding the optimal resource and operating mode usage. Synergic integration with novel programming models strategies and software engineering methods, as promoted in the "*High Productivity Programming Models for Extreme Computing"* and "*Software Engineering Methods for High-Performance Computing"* recommendations, is envisioned and requires the development of novel management and control APIs which can be exploited by the programmer and by the runtime to deploy at Exascale the potential energy-efficiency of novel architectures in the different application domains.

Sub project decomposition:

The recommendation aims at foster the research and development of:

- HW/SW APIs to manage the complexity and the programmability gap inherent of extreme heterogeneous Exascale level supercomputers;
- Design strategies for scalable and efficient heterogeneous-aware exascale applications;
- Scalable and efficient community scientific applications for exascale;
- System software to support efficient usage of exascale heterogeneous supercomputers in production.

[5] Greenhalgh, P., "Big.LITTLE Processing with ARM Cortex-A15 and Cortex-A7," ARM White Paper, 2011.

Impact (scientific, technological, societal & environmental):

Research projects in this area are expected to improve the performance and energy-efficiency scalability of the entire ecosystem of supercomputers applications which includes different aspect of the daily society as well as industrial competitiveness and social security. In addition being capable of handling extreme heterogeneous exascale supercomputing means being able of sustain exascale ICT infrastructure. This is in line with the ETP4HPC research agenda and is foreseen to achieve a synergic integration with the other research pillars identified by the EESI recommendations.

Funding:

We suggest collaborative projects as founding scheme, with duration of 3 years and total budget of 7-10M€.

Software Engineering Methods for High-Performance Computing

Tools&PM

Recommendation Brief Description:

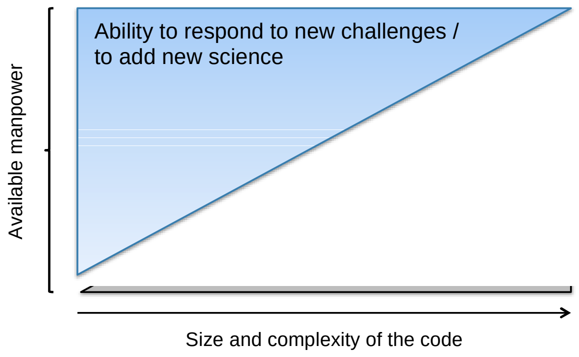
To be able to cope with the complexity of exascale systems in terms of very large number of nodes, heterogeneity of resources, energy awareness, or fault tolerance, to mention just a few challenges, new productivity-enhancing methods and tools are needed to use the manpower available for the development and maintenance of HPC software more effectively, maximizing the potential of applications under given resource constraints.

Recommendation Context

Starting from very vague high-level requirements, high-performance computing (HPC) applications are often developed incrementally, evolving through many versions, and rarely reaching a final version as the frontier of sciences in the field and in computing hardware is constantly advancing. The software process is often described as informally agile. Many codes begin their life in academia as the product of a small team. As the functionality and the demand grow, the team becomes bigger. Initially, users and developers are usually identical. Once the code has grown to reach a certain level of maturity, scientists or engineers external to the development team start joining the user base. Successful codes are often commercialized or continued as community projects, especially if the resources and the expertise of a single organization do not suffice. The larger the user base and the more critical the application, the more important become non-functional requirements such as performance on high-end parallel platforms, robustness, and ease-of-use. Correctness is always the most important requirement, but the effects of errors are more disastrous if more people depend on correct results. In general, the lifespan of successful HPC applications can be in the order of multiple decades, typically surviving several generations of hardware. In the context of exascale, aspects such the availability of tools for debugging and performance analysis on very large numbers of nodes, tools to support the development of applications for complex and heterogeneous systems, and tools to predict the behaviour of applications on those systems need to be considered.

Objective

In addition to the cost of adding new features, software has also a cost of ownership – even if the functionality remains stable. Bugs have to be fixed and the code has to be ported to new platforms at least before the current ones are decommissioned, which usually happens every few years. The less maintainable an application is, the higher its cost of ownership becomes and the less manpower is available for the creation of new scientific capabilities. New developments also become more challenging and thus more expensive the more complex the original code base is. Just like a major portion of the operational costs of hardware have been distilled into the term power envelope, it is helpful to think of software as having a manpower envelope that places a limit on its size and complexity. If size and complexity exceed the limit of what the manpower envelope can support, the code is destined to stagnate (see figure). The objective must therefore be to lower the cost of ownership and utilize the remaining manpower as efficiently as possible to maximize the potential of the code.


Ability to respond to new challenges / to add new science
Available manpower
Size and complexity of the code

Given the long lifetime of HPC codes, development rarely means development from scratch. Instead, it more than often means extending or restructuring existing code, which was often written by people different from those who apply the changes. This is true in particular for codes being developed in academic institutions with their natural staff fluctuation. For a grown code base, even prototypical changes are expensive and require careful planning. Especially for large industry and community codes, regression testing emerges as a major cost factor. Regression testing has to cover functional as well as non-functional requirements such as performance and scalability. Given that simulation codes often aim at scenarios that are hardly accessible to experiments, the generation of test cases itself can become challenging. In industries such as automotive and aerospace, testing may have legal implications where safety certificates can only be issued if simulation software has been verified. Any changes to the software can therefore be particularly time-consuming and costly.

To reach exascale, applications will have to address numerous technical hurdles simultaneously, including (i) scalable and energy efficient algorithm design, (ii) hardware faults which may compromise scalability, (iii) soft errors which may compromise correctness, and (iv) the pre- and post-processing of vast amounts of input and output data. Moreover, the problems exascale applications are supposed to solve are also more complex from a scientific viewpoint, often requiring the coupling of methods across many length and time scales. As a consequence, the development costs of applications that strive to run at this level will rise significantly, creating a need to reassess software processes from an economic perspective. Otherwise, the capability of our software base will be left way behind the potential of our hardware.

Impact

Improved software engineering methods have the potential to lower development costs and make development more sustainable. In this context, sustainability means the ability to grow or change. This is necessary to preserve the value of our investments in software. If development is not sustainable, an application may never reach exascale. Thus, proper software engineering goes beyond budgetary considerations, and indeed it is almost certainly an essential ingredient on the path towards exascale. While not at the frontier of domain science, software engineering methods should be regarded as a critical link of the supply chain without which this frontier cannot advance. In this sense, its contribution will significantly help justify the investments in exascale hard- and software.

Level of innovation

The required level of innovation has two dimensions – one cultural and one technological. Changing the culture means to move away from a field split into the sub-disciplines domain science, numerical analysis, systems architecture, and programming towards a more integrated view that also encompasses people and processes. From a technological viewpoint, the required methods and tools will face the same two categories of challenges that application developers face. There is a genuine methodological challenge of solving the problem at hand and there is the exascale challenge of making the solution work with hundreds of millions of threads.

Proposal

In this recommendation, we concentrate on key aspects related to design and quality management, aiming at providing appropriate methods and tools to support them.

- ➢ Develop predictive methods and **tools to assist software re-design and co-design** of scientific application software for future platforms. They need to predict the effects of changes in the software, especially with respect to performance, scalability, energy efficiency, and fault tolerance on emerging exascale systems. We further require lightweight predictive tools that estimate the effects of changes of the hardware as a basis for the co-design of future platforms or to detect when communication bottlenecks will prevent applications to scale, to give an example. While such predictions are already being made, the methods being applied are still ad-hoc and tools, where they exist, lack the desired level of automation and robustness.

- Develop **scalable debugging and performance analysis tools** for exascale systems. While tools for correctness and performance analysis exist, they lack the required scalability and offer only limited insight. Both debuggers and performance analyzers need to be enabled to cope with even more massive amounts of information in a scalable manner. Both static and dynamic approaches should be considered. Moreover, modeling tools are needed that anticipate performance bottlenecks before they become manifest at the target scale. Such tools will also be useful in assessing the potential of existing codes as a prerequisite for planning and prioritizing changes. With their strong record in tools, the European community is optimally prepared.
- Carry out a **survey of current software engineering practices and processes** across a wide range of academic and industrial HPC software development activities. While development practices have been surveyed in the past, it is safe to assume that technological progress has changed the landscape since then and will continue to do so. Based on the outcome, we need to review current practices and **define and validate optimal software engineering** processes for exascale applications. In this context, emphasis should be given to re-use techniques such as software families and product lines, e.g., in combination with domain-specific languages (DSLs).
- Develop new and efficient ***resilience tools*** specifically dedicated to Exascale figures, for supporting the transformation of codes into resilient software. This topic should be fully integrated into the EESI recommendation ***"Holistic approach to resilience for simulations and data analytics"*** (chapter of this document).

Holistic approach to resilience for simulations and data analytics

Tools&PM

Summary

The evolution of the software and hardware technologies will lead to an increase of fault rates that will translate into higher error and failure rates at Exascale. HPC hardware itself cannot detect and correct all errors and failures. Maintaining European resilience capabilities is mandatory to be able to develop efficient Exascale applications. But besides the development of resilience techniques, a holistic detection/recovery approach covering and orchestrating all layers from the hardware to the application appears necessary to be able to run simulations and data analytics executions to completion and produce correct results.

Context

The HPC community is facing difficult challenges concerning resilience at extreme scale. Namely, it is projected that Exascale systems will suffer more frequent failures (process, node, network, and component) and more frequent Silent Data Corruptions (SDCs) than current systems due to an increase in complexity and in the number of components. Current solutions are already responsible for 20% of loss in computing capacity. All experts of resilience reports consider that the current most popular solution that is application level checkpointing on parallel file system will become a serious bottleneck as systems increase in size and complexity and will not be applicable at Exascale.

Improvement in hardware detection and recovery is needed but will not suffice to cover efficiently all error and failure scenarios.

First, software errors like file system failures and other system software failures cannot be covered by hardware.

Second, at hardware level, hardening the main expected sources of soft errors (latchs and flip flops) may lack supportive markets to justify the extra cost.

Third, the last generation of Petascale systems (beyond 10 Petaflops) shows that systems built from non proprietary components, for example systems using commodity CPUs and accelerators are seeing up to a dozen of failures per day that are not handled by hardware.

Research in system software, numerical libraries and application codes has produced principles and some times academic software prototypes.

However

1) all these techniques (multi-level checkpointing, fault tolerant protocols, alternative to checkpointing, failure prediction, resilient algorithms, detection of silent data corruptions) still need significant research to be ready for the Exascale

2) these software solutions are sparse and not designed to be integrated and work in concert in a resilience software stack.

The development of separate resilience techniques and knowledge will most probably not respond adequately to the Exascale challenge. The articulation of resilience solutions appears to be a key issue to strengthen the capacity of the community (including system vendors) to produce efficient and effective resilience solutions for Exascale systems. There is an opportunity window for Europe to develop this approach which will be needed anyway in short future.

Recommendation

The community can address now this situation by proposing the development of ***resilience API*** (Application Programming Interfaces) that will provide the required integration of resilience techniques and coordination of software resilience mechanisms and by improving critical resilience mechanisms:

- understanding and modeling of fault propagation,
- error detection,
- failure prediction,
- roll-back and roll-forward recovery.

This API defined in concert by resilient experts and application developers will provide efficient resilience for Exascale simulations and data analytics executions.

To that goal, it is particularly important to form an ecosystem involving system hardware developers through ETP4HPC, system software developers, library developers as well as application developers and end users through PRACE.

Verification Validation and Uncertainty Quantification tools evolution for a better exploitation of Exascale capacities

Tools&PM

Recommendation brief description

The mathematical background behind uncertainty analysis is very strong and comes from the field of statistics. Europe can claim world leading experts on these topics, but the link between this community and the HPC community must be strengthened. The recommendation aims at preparing an unified European VVUQ package for Exascale computing by identifying and solving problems limiting usability of these tools on many-core configurations; facilitating access to the VVUQ techniques to the HPC community by providing software that is ready for deployment on supercomputers; and making methodological progresses on the VVUQ methods for very large computations.

Context

In the field of large scale scientific simulations of complex phenomena involving multi-scale and multi-physics models, use as input of massive datasets acquired from large scale instruments (network of sensors, (radio) telescopes, satellites, …) or analysis of large amount of data generated by computer simulations, the associated uncertainties in the numerical simulation process can arise from different sources:

- Lack of knowledge on a physical parameter (epistemic uncertainty),
- Parameter with a random nature (aleatory uncertainty),
- Uncertainty related to the model (model error, too simplified model),
- Uncertainty related to the numerical errors (numerical errors of the model, to the input and output data …).

Taking into account these uncertainties is essential for the acceptance of numerical simulation for decision making. These uncertainties must be integrated in the verification and validation process of the simulation codes. This process is now commonly called VVUQ (Verification, Validation and Uncertainty Quantification). Verification consists in checking that the equations underlying the code are correctly solved. Validation is the stage during which the predictive capability of the numerical model is checked against experimental data or a reference model. Uncertainty quantification consists in defining the uncertainties that taunt the output of the simulation code.

The rise of Exascale with future simulations spanning over billions of threads executed on many core heterogeneous devices, dealing with complex storage hierarchies and software stack will reduce dramatically the deterministic nature of simulations, models and results generated. In order to being able to take into account such upcoming technologies and continue to trust into results obtained (with regards to the reduction of experiments) it become mandatory to develop solid VVUQ methodologies and tools and Europe have a strong card to play in this field like OpenTurns [1] and URANIE [2] regarding international competition in US with DAKOTA [3] and PSUADE Uncertainty Quantification Project by Lawrence Livermore National Labs [4].

This unified framework shall need at the same time:

- multidisciplinary skilled teams (statistics & probability, numerical analysis, PDE, physicians),
- access to high computational power, as the statistical methods for calibration and validation need to evaluate several times a (possibly) costly numerical code.

This recommendation is very connected with the other EESI2 recommendations related to Software Engineering Methods for High-Performance Computing.

Objectives of the Recommendation

R1 - Ultra-scalable tools for VVUQ:

Verification tools: Verification procedures cover two aspects: software unit tests and mathematical verification of the numerical resolution. For unit tests, the software tools that are used mostly come from the broad software engineering community, and do not take into account the specific needs of Exascale computing. The proposal is to improve those tools to make them suitable.

Validation and UQ: The VVUQ tools should evolve to make use of the different levels of parallelism in a smooth manner: batch scheduling system, distributed parallelism, in-node parallelism strategies, and on coupled simulations involving several codes.

R2 – Accessibility of the software:

Numerical/software improvements on VVUQ tools to facilitate 'black-box' usage and therefore facilitate the dissemination of techniques on all software, be it academic or industrial, open source or commercial.

Setting up an unified European-wide package for VVUQ, with deployment and promotion of the software on the PRACE infrastructures and efficiency benchmarks.

R3 – Methodological progresses:

Model errors in the validation process: Most of the UQ techniques make the assumption that numerical models are perfect and propagate parametric uncertainties, but they of aleatory or epistemic nature. Emerging techniques provide solutions for improved predictability of the simulations taking into account the existence of model errors.

Surrogate models and reduced basis models: These models used to replace actual codes with less computationally intensive numerical models are essential elements for uncertainty analysis and optimisation of large simulations. The work will consist in improving the learning stages of these models by taking into account the objective (finding an extremum, defining a failure point, evaluating uncertainty) at the learning stage.

In that context it's proposed that targeted IP funding tools over 4 years could host this project. It should mean an approximate 15 million Euros budget, 6M on R1, 6M on R2, and 3M on R3. Deliverables will come mostly in the form of Open Source software with adequate support and deployment structures, ready for use for the scientific and industrial community.

References

[1] OpenTURNS : http://www.openturns.org

[2] URANIE : http://sourceforge.net/projects/uranie/

[3] DAKOTA (Design Analysis Kit for Optimization and Terascale Applications) toolkit developed by Sandia National Laboratory http://dakota.sandia.gov/software.html

[4] PSUADE Uncertainty Quantification Project http://computation.llnl.gov/casc/uncertainty_quantification/

Algorithms for Communication and Data-Movement Avoidance

Ultra Scalable Algorithms

Summary

Explore novel algorithmic strategies, far beyond the well-known communication hiding techniques, to minimize data movement as well as the number of communication and synchronization instances in extreme computing; minimization should happen at both local (e.g. within a multiprocessor, across a memory hierarchy) and remote (e.g. network) levels.

Context and Background

As it is well recognized, in parallel computing there are three key factors: data movement, data movement and data movement. The cost of moving data is consistently recognized as the biggest obstacle to Exascale computing, both within and between computing units (processors).

Researchers in HPC have long been aware of this general principle, see e.g. [7]; in 2004, David Patterson [6] outlined the situation for computing networks in general, and discussed design techniques of *caching*, *replication* and *prediction* to mitigate the effects of network latency. Related techniques include *cache-oblivious* algorithms; and in the case of HPC, there is a growing trend to *recompute* quantities whenever this is cheaper than retrieving them.

Avoiding data movement has also very relevant *power efficiency* implications; some data on this was presented in e.g. [2], where many comparisons were made between data retrieval from caches in both GPUs and CPUs. Since one of the critical factors for Exascale computing is energy efficiency, efficient data movement impacts both computation speed and energy consumption.
A unified treatment of intra-processor, memory hierarchy and network data movement is a necessary step towards the development of a sustainable programming model for an Exascale computing infrastructure.

Most of the research in the past has addressed the communication problem as a scheduling or a tuning problem. Indeed one approach is to overlap communication with computation, but such an approach reaches its limits very fast. Hence, even if the gap between communication speed and computation speed is a well-known problem since several years, the progress achieved by using scheduling or tuning techniques is not sufficient.

Techniques for communication and data movement avoidance have far reaching algorithmic and mathematical implications. In a quest to address the communication problem, European researchers co-invented communication avoiding algorithms for dense and sparse linear algebra, e.g. [3, 4, 5]. These algorithms are able to provably minimize communication and are based on novel numerical algorithms and techniques. These first papers have shown that it is necessary to address the communication problem directly during the design of a numerical algorithm. This leads to the design of a new generation of algorithms that reduce the number of communication and synchronization instances to a minimum, and hence drastically reduce the communication cost with respect to classic algorithms.
In addition, European researchers have been extensively exploring hierarchical algorithms based on compression techniques as *H*-matrices and fast multipole methods, which are very relevant in this context. Hierarchical algorithms are inherently capable of reducing communication as well as synchronization at all levels of the on-chip and off-chip network system. Other research groups are involved in techniques for handling data movement within hybrid computing nodes [1].

Objectives of the Recommendation

One of the objectives of this recommendation is to coordinate the multiple groups working on many different aspects and in different algorithmic areas.
The funding from the EU should cover programs to:

- Explore the design of a new generation of algorithms, for both dense and sparse linear algebra and beyond, that are able to drastically reduce communication costs, and even provably minimize it in some cases;

- Focus on operations that are at the intersection with the data mining community, as for example computing the low rank approximation of a very large matrix, an important tool in different areas;
- Focus on a comprehensive treatment of data movement within and between computing nodes;
- Enable the development of sustainable software that implements this new generation of communication avoiding numerical algorithms;
- Enable leadership of European researchers in selected areas;
- Enable the coordination and federation of multiple efforts to reach a critical mass.

Dense and sparse linear algebra operations are a prime candidate for such efforts, because they are key algorithmic areas for a wide range of scientific applications, and because their rich mathematical structure enables modeling the behavior of computing systems in a compact and precise manner. However communication avoiding algorithms should be developed beyond numerical linear algebra, for all critical stages of computationally intensive applications, e.g. mesh generation algorithms, parallel in time methods.

References

[1] V. Cardellini, S. Filippone, D. Rouson: Design Patterns for sparse matrix computations on hybrid CPU/GPU platforms. Scientific Programming, 22 (2014).

[2] B. Dally: GPU Computing to Exascale and beyond, SC2010, available from http://www.nvidia.com/content/PDF/sc 2010/theater/Dally SC10.pdf

3] J. Demmel, L. Grigori, M. Hoemmen, and J. Langou, Communication-Optimal Parallel and Sequential QR and LU Factor- izations, SIAM J. Sci. Comput. Vol. 34, No. 1, pp. A206A239, 2012, available since 2008.

[4] L. Grigori, J. Demmel, and H. Xiang, Communication avoiding Gaussian elimination, Proceedings of the IEEE/ACM SuperComputing SC08 Conference, November 2008.

[5] L. Grigori, J. Demmel, and H. Xiang, CALU: a communication optimal LU factorization algorithm, SIAM J. Matrix Anal. & Appl., 32, pp. 1317-1350, 2011.

[6] D. Patterson Latency Lags Bandwidth, Comm. of the ACM., 47(10)):71–75, October 2004.

[7] W. Wulf and S. McKee. Hitting the wall: Implications of the obvious. ACM SIGArch Computer Architecture News, 23(1):20–24, Mar. 1995.

Parallel-in-Time: a Major Step Forward in Parallel Simulations

Ultra Scalable Algorithms

Recommendation Brief Description

The **efficient exploitation of Exascale systems** will require massive increases in the parallelism of simulation codes, and today most time-stepping codes make little or no use of **parallelism in the time domain**; the time is right for a coordinated research program exploring the **huge potential of Parallel-in-Time methods** across a wide range of application domains.

Recommendation Context

Exascale systems which are achievable in the next 5-10 years will contain millions of cores. In order to make efficient use of these systems, high-performance applications must have sufficient parallelism to support parallel execution across millions of threads of execution. The development of more efficient and more highly parallel scalable solvers is therefore at the forefront of Exascale applications research and development.

Most simulations which are expected to deliver economic, societal and scientific impact from Exascale systems contain time-stepping in some form and present-day codes make little or no use of parallelism in the time domain; time stepping is currently treated as a serial process.

There is a class of Parallel-in-Time methods which through parallelization of the time domain have the potential to extract very large additional parallelism from a wide range of time-stepping application codes. This is a disruptive technology which will deliver performance speed-ups of between 10 and 100. By comparison, optimizations of current algorithms typically yield benefits in the range of tens of percent, or at most a factor of 2-3 improvement.

Potential application areas include: climate research, computational fluid dynamics, life sciences, materials science, nuclear engineering, etc. These include applications of HPC with the highest return on investment in terms of economic, societal and scientific impact.

European researchers are leading Parallel-in-Time developments. There has been a series of three international workshops dedicated to these algorithms held in Europe (Lugano, 2011, Manchester, 2013, and Jülich, 2014) with 21 European speakers from six EU countries as well as invited speakers from the US, Japan and Russia. This is indicative of a diverse, thriving and world-leading European research community.

The time is now right for a coordinated research program exploring the huge potential of Parallel-in-Time methods across a wide range of application domains. The Parallel-in-Time concept is now firmly established from an applied mathematics point of view, there is a developing European community and some applications have been deployed (e.g. in the field of neutron transport). However, it remains necessary to address some key issues:

i. the applicability to fluid dynamics, with the particular problem of energy conservation for the convection part;

ii. the application of Parallel-in-Time methods to highly oscillatory non-linear systems, e.g. through a numerically computed locally asymptotic solution;

iii. new Parallel-in-Time approaches based on the extension of classical multigrid methods, yielding multiscale representations in both space and time (space-time multigrid);

iv. the feasibility of massively parallel implementations, including the potential of coupling these Parallel-in-Time algorithms with other iterative solvers like Picard or Newton type fixed points procedures for non-linear issues or parallel domain decomposition techniques;

v. the impact of Parallel-in-Time methods on the memory requirements, data management and energy efficiency of applications; and

vi. Develop a Parallel-in-Time algorithm focusing on the accuracy of time-averages, since for some applications where long time propagations are required, it is not the exact trajectory that is of interest but some average properties like ergodicity.

Parallel-in-Time methods have some common features with earlier methods for which the introduction of a hierarchy makes parallelization easier, e.g. parallel multigrid, fast multipole. Synergies with these methods should be examined, with the possible sharing of ideas and HPC tools.

Objectives of the Recommendation

The goal of this recommendation is to fund R&D programs with the following objectives:

- Application of Parallel-in-Time methods to a wide range of application domains should now be tackled, either directly where energy conservation issues are not limiting e.g. for some life sciences problems or molecular dynamics, or after solving this issue, e.g. by projection like approaches for geophysical fluids (ocean and atmosphere), climate, seismology etc.
- The establishment of multi-disciplinary consortia to work on the deployment of Parallel-in-Time methods and applications to new fields, combining the expertise of applied mathematicians, application scientists, computational scientists and HPC technology specialists, following a co-design approach.
- A number of different options should be looked at when preparing projects: having a rather large project, under which a number of different applications would be studied and synergistically developed, and/or a coordinated cluster of smaller projects, each of which looks at a single application, but exchanging knowledge and experience.
- A series of benchmarks and test cases should be established and maintained in order to have a clear view of the advantages, disadvantages and quality of the different Parallel-in-Time methods. The test cases should be at the same time close to the real applications and simple enough to allow both the integration and test of Parallel-in-Time methods.
- In order to maximize their exploitation across different application domains, Parallel-in-Time software should be encapsulated in reusable scalable libraries. This should be at a timely point in the development of the software and following the establishment of a stable and robust methodology born out of experience with a range of applications. Such libraries would then accelerate and facilitate the further deployment of Parallel-in-Time methods in new application areas and targeting new libraries for Exascale computing.
- We envisage a number of world-leading projects utilizing the Parallel-in-Time method but focusing on different approaches and on different application areas. With 2-4 projects between €2M and €4M each, we recommend that the total amount of support should be in the range of €5 million to €10 million.

The ultimate goals of the Parallel-in-Time initiative are to establish a coordinated multi-disciplinary community of scientists and technologists with skills in the Parallel-in-Time methods, and to develop and deploy those methods to deliver significant improvements in the performance and hence in the impact of important simulation codes exploiting future high-performance systems.

Software for Data Centric Approaches to Extreme Computing

Efficiency at Exascale level requires breaking with the traditional scientific workflow where simulation data are stored on disk for later analysis. This disruption comes in sync with new memory technologies, new photonic networks as well as the increasing cost of transistors. For instance new non-volatile memories (e.g. Memristors) hold the promise of providing persistent memories close to the CPU that are fast, large, energy efficient and at a reasonable cost. On the software side, big data and other in memory computing technologies may be providing new solutions to help scientists facing the coming deluge of data. Holistic approaches considering all data cycles from sensors capture to visualization, encompassing simulation, code coupling, in-situ, pre and post analysis can guarantee that no bottlenecks are introduced in the scientific discovery process. In particular, it is strongly wished that new systems simplify human-in-the-loop workflows.

Data management is at the core of the design of Exascale applications. This is illustrated in Figure 1. In this figure, data may follow any paths (blue arrows) in the ecosystem, each component having its how performance profile, quality of service and cost. For instance while HPC technology optimizes writing in parallel the data, data mining techniques favor reading. Choosing to use one or the other technology must be carefully planed according to a global view of the workflow.

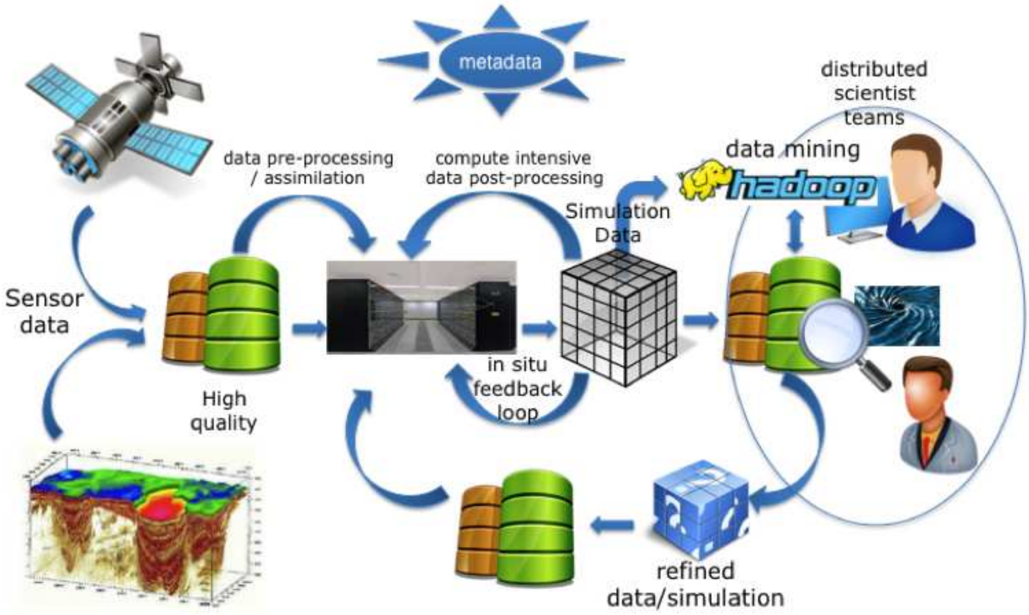


Figure 1: *complex work flow of Exascale application*

On one hand one must consider the rising price of IO systems. On the other hand as a deluge of data is to be expected synergies between big data and traditional HPC techniques has to be well thought-out. Data categories are also an important concern. For instance, data from sensors cannot be regenerated and must be stored safely while some data produced by simulation may be easier to re-compute when combine with in-situ data processing techniques. Each data must be stored and organized to use the proper resources. As well, metadata and provenance must be kept consistent all the way. This likely will strongly disrupt current practices.

The deluge of data requires new data analysis techniques. Big data (extreme data) technologies may provide new disruptive methods for such tasks. These techniques need to be extended to take advantage of highly scalable parallel infrastructure. This may be a return contribution of HPC to the big data field. Behind this topic lies many complex and holistic issues such as: serialization/deserialization of data, design of data structures able to cope with highly asynchronous execution as well as compute / IO activities interleaving.

More generally, data mining techniques must be extended to fit the file formats used in HPC (e.g. HDF5, netCFD) and bridges must be established between HPC and big data usual formats.

Metadata management and specification is also a critical challenge. They are keys elements in the science discovery process. Their design is particularly important to obtain a consistent end-to-end use of the data. Furthermore, they impact on sharing policy management implementation (e.g. at the core of the decision process concerning data to be set public, what storage migration, etc.).

Analysis and visualization of data produced by large-scale simulations are often sidelined in favor of pure computation performance. As we foresee Exascale systems in the next decade, the offline analysis approach shows its limits: more and more scientists see the scalability of their simulations dropping because of unmatched computation and I/O performance as well as higher I/O variability. However, in-situ[6] approaches (potentially more efficient) have difficulties in getting accepted, as scientists fear to dive into fundamental code changes in a simulation they have used for years. Defining the right tradeoff here is a challenge. Also related to the same limitation in I/O performance, HPC scientists predict fundamental changes in the way I/O and data management will be handled in the near future. In particular, the heterogeneous processor environment and memory hierarchy of the new platforms, together with the increasing use of GPU and accelerators, open new alternatives for data analysis.

Application development for Exascale systems is of a rare complexity. Complexity lays in achieving scalability in all steps and in roadmapping a software in an uncertain environment. On one hand it is best if legacy codes can be reused, on the other hand it is likely that many codes will have to be deeply re-designed / re-developed. Figure 2 illustrates this tradeoff. Domain specific approaches may be able to hide complexity to users but as they are mode specific they address a smaller community. In the end, a tradeoff must be made between development cost (including the tools, API, maintenance, etc.) and the potential user's base. As application software moves much slower than hardware technology we believe that anticipation is extremely crucial.

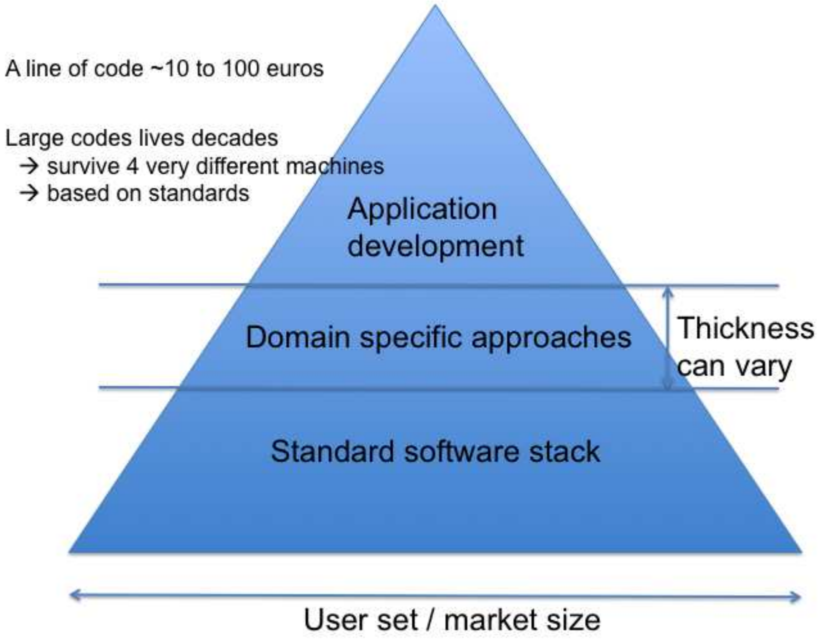

A line of code ~10 to 100 euros
Large codes lives decades
→ survive 4 very different machines
→ based on standards
Application development
Domain specific approaches
Thickness can vary
Standard software stack
User set / market size

Figure 2: Domain specific approach trade-off.

[6] See EESI recommendation on this specific topic.

This topic cannot be viewed only under the technology angle. Indeed, designing the applications requires finding tradeoffs between in-situ vs. ex-situ processing, selecting data format, access policy, data relocation, format changes, etc. These tradeoffs are not only driven by technology and performance but also by the ecosystem exposed to the researchers. Furthermore, it is important to note that a global efficient use of the Exascale resources can be contradictory with the objectives of individual research teams. **Understanding the full cycle of data** is probably the most important question to drive Exascale technology development.
Unfortunately, making rational and efficient uses of communication, compute, storage resources requires engineer skills that are currently in shortage or just not available to scientists.

EESI aims at encouraging the community to form multi-disciplinary research groups capable of handling the complete set of concepts necessary to design data centric approaches to Exascale computing. It is particularly important to also integrate ecosystem and economical issues. **For instance, energy cost is a growing concern that may lead to move from "charge by core-hours” to “charge by kilowatt-hours” in order to capture the entire complexity of a data centric approach to Exascale.**

Towards flexible and efficient Exascale software couplers

Data Centric Approaches

Summary

As stated by applications experts in many recent reports, the rise of extreme computing with data intensive capacities will allow only few “hero” applications to scale out to such full systems in capability mode (one simulation scaling on billions of cores). The major potential of such upcoming architectures will rely capacity simulation based on multi-scale and multi-physics scientific codes running individually on hundreds thousands of cores and smartly coupled together with highly loaded models that exchange data with a high frequency.

Such coupled models are challenging to develop due to the need to coordinate execution of the independently developed model components while resolving both scientific and technical heterogeneities.

Despite some existing specific initiatives, there is a crucial need to develop new and common European-wide coupling methodologies and tools in order to support major scientific challenges in research (evolution of the climate, astrophysics and materials) and engineering (combustion, catalysis, energy, …).

Context

In many scientific and industrial communities (aerospace, aeronautical, climate, automotive, astronomy, electronics …), some of the most complex scientific simulations requiring extreme scale computing are based on multi-physics and multi-scale models that simulate multiple interacting physical processes by combining independently developed modeling components into a single model. Development of coupled models is an inherently multi-disciplinary effort, requiring scientific expertise from multiple domains as well as technical expertise in developing high performance software. One source of uncertainty is whether the state-of-the-art software libraries used today for model coupling will scale efficiently on Exascale platforms.

The component models (i.e., the constituents that are linked together) often utilize different spatial representations, requiring interpolation from one model’s output to another model’s input, evolve at different timescales, requiring temporal averaging or statistical sampling, and use incompatible data structures, requiring on-the-fly data structure conversion.

The efficiency of these different coupling requirements in conjunction with the load balancing of the component models is therefore crucial for Exascale computing. The main design constraints underlined for efficient Exascale computing are of course mandatory for coupled applications: use more parallelism, less memory and less communication.

The current coupling technologies used for multi-physics and multi-components simulations can roughly be split into two main categories. The first one, *direct coupling*, is designed to offer faster implementation while the second, *coupling via top-level interfaces*, one focuses on high performance via architectural conformity. There is not a clear view of which coupling architecture will survive to Exascale computing. Both are submitted to almost the same challenges (use more parallelism, less communications and less memory) and offer alternative solutions to address them.

Objectives of the Recommendation

To improve the performances of coupled applications in terms of usability and scalability on Exascale machines, the recommendations are to work on coupling libraries as well as on coupled models and their environment.

Recommendations on the coupler improvements:

Based on recent experiences on actual computers, several essential improvements of coupling paradigms have to be investigated:

- It is of primary interest to design standard coupling API in order to enable interoperability, ease the integration of new models and cross disciplinary exchanges and sharing of performance analysis,

- The methods implemented in the software have to avoid centralization of data and prefer distribution even for high order interpolation methods,
- The performance of localization process (needed for the weight and address calculations required for the interpolation of the data between the components meshes) at the beginning of the coupled computations, and optimization of this process for geometrical or mesh changes during the simulations as well as of communication performances between the coupled models have to be increase by use of asynchronous processes, hide of operations by computations, intelligent search algorithms…
- As for the next generation of codes, it is important to investigate new programming models and systems for parallelism (PGAS, hybrid MPI/OpenMP …) at the level of the component models and of the coupler,
- Finally, in order to benefit from scalability performances of component models on different hardware, investigations in the possibility to couple efficiently component models running on heterogeneous architectures have to be achieved.

Recommendations on the coupled models improvements:
From actual performance measurements, an important path to efficient Exascale coupled simulations is integrated deeper the coupling process inside the component model environments:

- An advanced comparison between single and multi executables have to be done on several component models types in order to extract the pro and cons of each methodology in the context of Exascale computing.
- It is essential to improve the coupling algorithms in order to hide communications with computations in the models, exchange only relevant information with reasonable frequency …
- The communication costs between the models have to be reduced by optimizing the balance between processes that are concerned by the coupling and other ones that are not, as well as the corresponding data distribution, optimizing the communication schemes between the coupled models via co-partitioning methods …

Recommendations on the software environment:
Going to Exascale simulations brings new challenges in setting up and post-process multi-physics and multi-components computations on complex geometries in order to produce accurate and exploitable results. Hence, tools dedicated to ease to set up coupled computations and to post-process them have to be proposed (mesh connection between model, quick verifications of conformity, evaluation of physical quantities during computations, joint exploitation of massive results …).

In that context it is proposed that targeted IP funding tools over 4 years could host this project. It should mean an approximate 25 people and 8-12 million Euros budget.

In Situ Extreme Data Processing and Better science through I/O avoidance in High-Performance Computing systems

Data Centric Approaches

Recommendation Brief Description

Data analysis framework from a post-process centric to a close to real-time concurrent approach based on **either in-situ or in-transit processing of the raw data** of numerical simulations, **processed as they are computed** during massively parallel post-petascale and Exascale applications.

Recommendation Context

With the onset of extreme-scale computing (at Exascale for example), challenges related to extreme data, energy and I/O constraints are becoming dominating concerns. It will become impossible for scientists to save a sufficient amount of raw simulation data to persistent storage for subsequent processing

Algorithms must adapt to machines with extreme concurrency, low communication bandwidth, and high memory latency, while operating within the time constraints prescribed by the simulation.

So, it is necessary to move away from an only post-process centric data analysis paradigm towards a complementary as close as possible to real time concurrent analysis framework, in which raw simulation data is processed as it is computed in order to maximize data utilization while loaded into processing units and minimize rough data movements to remote storage units in order to save energy. The quite real time in situ processing could also allow computational steering of simulations (backward, pause, forward).

In this context computations are considered in-situ if they utilize the primary compute resources, while in-transit processing refers to offloading computations to a set of secondary resources using asynchronous data transfers. Both approaches are intermediate steps towards the end goal of achieving real-time extreme data analysis in high-performance computing systems.

Objectives of the Recommendation

The goal of this recommendation is to fund R&D programs in order to explore:

- Real Time (In situ) data-related energy/performance trade-offs for end-to-end simulation workflows running at scale on current high-end computing systems, with a power model based on system power and data exchange patterns.
- The design and implementation of common and new analysis techniques typically performed on large-scale scientific simulations: topological analysis, descriptive statistics, surrogate data model, filtering, compression, pattern/feature discovery, error analysis …
- The approximations which will be used to adaptively determine frequency at which full (no compression) analyses are computed, …
- The promising new approach of sub-linear algorithms addressing the fundamental mathematical problem of understanding global features of a data set using limited resources.
- The algorithmic developments, co-scheduling system to coordinate the execution of various analysis workflows.
- The ultra scalable and the most flexible implementation of a framework that support efficient data movement between in-situ and in-transit computations taking into account tiered storage architectures and asynchronous data transfers.
- The use of multi-precision computations, reconsidering most algorithms and studying (using e.g. backward error analysis) places where such data compression can be safely implemented, and providing necessary error indicators for these optimized routines, as currently done in sparse direct solvers.

- The use of in situ data analysis for tracking and checking fault or error propagation into the simulation, associating a resilience aspect with the execution of workflows on parallel systems.

With the necessary following actions:

- All these developments must be associated to software improvements and actions of data processing should be scheduled with new programming models.
- All new in situ algorithms, real time methods should be validated by an iterative way on real applications through at least post processing comparison.

The ultimate goal of the in situ extreme data processing is to promote new data transformations and compressions that reduce drastically extreme raw data, generated during HPC simulations, by preserving the information required for a particular analysis while sacrificing most everything else and store the only relevant data.

All these theoretical ideas should be aligned with practical challenges of in-situ, in-transit and real-time high-performance computation where extreme data must be processed under severe communication and memory constraints.

Declarative processing frameworks for big data analytics

Data Centric Approaches

Recommendation Brief Description

Exascale systems provide an incredible huge amount of *synthetic* data that need to be processed (e.g. for visualization) in order to get a full understanding of what they simulate, conversely, data acquisition system gather an incredible amount of diverse *real* data that need to be processed to get a better understanding of the situations they have been gathered from. Computer scientists and specialists of statistics used to manage and treat these data. The current Variety, Volume and Velocity of data imply a synergy and collaboration between different fields of science in order to extract full intelligence and knowledge from these data in close to real time.

Recommendation Context

The last decade was marked by the digitalization of virtually all aspects of our daily lives. Today, businesses, government institutions, and science and engineering organizations face an avalanche of digital data on a daily basis. All due in part to the decline in disk storage costs, the ever-increasing popularity of cloud storage services, and the ubiquitous availability of networked devices. At first glance this appears to be favorable for our increasingly networked society. However, in many ways it is a burden. Data is neither information, nor knowledge. Instead, data is of great value once it has been refined and analyzed, in order to address well-formulated questions, concerning problems of interest. It is only then that economic and social benefits can be fully realized.

Most of modern big data analytics questions can only be solved using techniques drawn from varying fields, including graph and network analysis, machine learning, mathematical modeling, numerical simulations, mathematical analysis, statistics, signal processing, and text processing, among others.

Reciprocally simulations on Exascale platforms of mathematical models produce similar avalanche of in silico data that are images of the situation they simulate. The interpretation, visualization allowing immersion and retroaction on the simulation require new tools at the level of the precision the models and their current simulations allow on Exascale platforms. The interpretation of these data, the understanding of their coherence does not rely enough on geometric analysis, topological analysis and tools developed in fields that are far from numerical simulations and use of computers.

Before the "big data" era, the few programmers with MPI expertise, predominantly located in supercomputing centers were sufficient in number. For many decades, software engineers and general users in varying domains did not have to worry about scalability issues in their computing systems, thanks in part to higher-level programming languages, compilers, and database systems. In contrast, today's existing technologies have reached their limits due to big data requirements, which involve data volume, data rate and heterogeneity, and the complexity of the analysis algorithms, which go beyond relational algebra, employing complex user-defined functions, iterations, and distributed state.

Data locality and volume are a primary concerns. The internal data structures must be designed in order to facilitate adaptation to the architecture of the computer.

In the era of many-core processors, cloud computing, and NoSQL, we must ensure that well-established declarative language concepts (inherent in relational database systems) make their way into big data systems. In order to make this a reality, the research community will need to address the related challenges. For example, (i) designing a programming language specification that does not require systems programming skills, (ii) mapping programs expressed in this programming language to a computing platform of their own choosing, and (iii) executing these in a scalable manner. In particular, this means devising execution strategies that are distributed, parallelized, and support both in-memory technologies and out-of-core execution for data-intensive algorithms.

In order to meet this challenge the data management community will have to come together with the HPC community. We will have to develop novel scalable algorithms and systems that are able to organize the data deluge and intelligently distill information to create value. To achieve this, declarative query languages must now be extended to support the declarative specification of varying analysis methods (e.g., anomaly detection, classification, and clustering). This will particularly require making iterations and limited forms of (distributed) state first class citizens of an extended relational algebra.

Furthermore, the power of declarative languages, namely, automatic optimization, parallelization, and adaptation of the same program to varying distributed systems and novel hardware architectures (depending on data distribution, data size, data rate, and system load) must be preserved. In this way, we will be able to overcome the current “stone age” in big data analytics. That is, algorithm specifications in systems that do not automatically optimize (e.g., MPI, MapReduce, and Hadoop), imperative languages (e.g., C), object-oriented languages (e.g., Java), and relational-oriented languages (e.g., SQL, XQuery, Pig, Hive, and JAQL) with non-tunable external driver programs, and technical computing systems (e.g., R and MATLAB) that do not scale.

Recommendation Objectives

Processing iterative, stateful data analysis programs on vast amounts of “data in motion” under low-latency while leveraging a declarative specification and ensuring data independence requires novel methods and techniques both from a systems and an algorithmic point-of-view. As research community, we will have to build on our existing results to considerably advance the state-of-the-art in designing and building systems for optimizing and executing complex data analysis programs on potentially evolving datasets under the constraint of significantly reducing latency (i.e., the time until first analysis results are available). In particular, we see the following major research and development challenges:

- **declarative specification and automatic deployment of complex data analysis programs:** we need to extend the declarative approach of relational databases to describe, plan, optimize, and execute iterative data analysis programs (DAPs) with complex user defined functions and mutable state;
- **declarative scalable data analysis libraries:** we need to cooperate with the signal processing, machine learning, and the general data analysis community on declarative, scalable algorithms in order to enable deep analysis of big data;
- **continuous, workload-aware optimization and execution of data analysis programs over evolving data:** data analysis usually is a multi-user scenario, where multiple DAPs run concurrently on the same system, each with the requirement for low-latency answers, and each competing for the same, scarce resources; this scenario requires the system to recognize and leverage synergies during the concurrent execution of long-running or standing queries;
- **adaptive, seamless deployment:** a wider application of data analysis techniques requires that a data analysis system that fits seamlessly into an existing computer architecture and can cope with and automatically optimize for specific properties of the underlying hardware;
- **trading-off virtualization:** modern data analysis systems must properly exploit the increasing availability of multi-core machines despite operating in a virtualized environment; achieving this goal requires us to identify crucial resources and make those components transparent to the runtime system, potentially using methods of paravirtualization in contrast to full virtualization; this aspect will be particularly important when accessing huge, distributed states (I/O and network transparency), as well as the efficient execution of CPU-bound operations;
- **first results fast:** low-latency processing of data streams despite high data ingest rates is a key requirement for the analysis of data from sensor networks, internet of things applications, robotics, or long-running simulations; from the systems side, this requires low startup costs for DAPs with respect to both query compilation and execution (which is particularly challenging for many core CPUs or compute clusters/clouds consisting of a large amount of parallel nodes), avoidance or intelligent hedging of

blocking/batching operators, and exploitation of state in long-running queries, in particular in conjunction with novel, algorithmic fault-tolerance;

- **Better account of models** : for many data frame there exist a large number of models, that are validated by a long term expertise and that allow already to provide knowledge for control, optimization, decision. These models are generally not used to process data that, on the opposite, use ad'hoc simple models that are built and improved from the data. The intelligence of involved and certified models does not help enough the data mining, a reason for this being the different time scale between the numerical simulation and data processing. However, for a decade now, approaches known as "model reduction techniques" have been proposed that allows performing accurate simulations from a two stage procedure: an offline learning process and a fast real time online procedure. These reduced approaches provide as good solutions as standard simulation and the on line process is within times compatible with data acquisition and mining. Some connection should be promoted between these different fields of research
- **provide automatic mathematical based procedure to recognize in raw data features that will help in better interpreting and/or visualizing and/or interact with data.**

The data management community **has already** been working on several of these challenges in isolation and has built systems beyond Hadoop**,** such as the European effort Stratosphere/Flink [4,5], next to systems developed in the US or Asia, such as Spark [1], ePIC [2], Asterix [3], among others. These systems represent an inspiring basis for future innovation, but will need to be adapted and partially redesigned to deal with the high complexity of the operators and data, in order to enable the broad adoption these systems through data independence and declarative specification.

[1] *M. Zaharia, M. Chowdhury, M. J. Franklin, et al: "Spark: cluster computing with working sets," HotCloud (2010).*

[2] *D. Jiang, G. Chen, B. C. Ooi, K.-L. Tan, S. Wu: "epiC: an Extensible and Scalable System for Processing Big Data," PVLDB 7(7): 541-552 (2014).*

[3] *S. Alsubaiee, Y. Altowim, H. Altwaijry, et al: "ASTERIX: An Open Source System for Big Data Management and Analysis." PVLDB 5(12): 1898-1901 (2012).*

[4] *Stratosphere, http://www.stratosphere.eu, last checked Jul 7, 2014*

[5] *Apache Flink Incubator Project, http://flink.incubator.apache.org/last checked Jul 7, 2014*

Identification of turbulent flow features into massively parallel Exascale simulations

Data Centric Approaches

Summary

The rise of multi petascale and upcoming Exascale HPC facilities will allow to turbulent simulations based on LES and DNS methods to address high fidelity complex problems in climate, combustion, astrophysics or fusion. These massive simulations performed on tens to hundreds thousands of threads will generate a huge volume of data, which is difficult and inefficient to post process asynchronously later after by a single researcher. The proposed approach consists on post processing this rough data on the fly by smart tools able automatically to extract pertinent turbulent flow features, store only a reduced amount of information or provide feedback to application in order to steer its behaviour.

Context

Large-Eddy Simulation and Direct Numerical Simulation are increasingly attractive approaches for the modelling of turbulent flows due to the rise of multi petascale and Exascale supercomputers. The increase in CPU power is a strong driving mechanism in the Computational Fluid Dynamics (CFD) community because it enables to enhance the fidelity of the simulations either increasing the mesh resolution or the simulation physical time, or adding more physics. As a result, large-scale simulations with meshes of several billion cells are currently generated on massively parallel machines using tens of thousands processors.

The analysis of billion-cell simulations is highly challenging because it requires handling a large amount of data. Traditional data processing tools usually need to be redesigned in order to cope with this amount of data. This challenge, which is shared with many other scientific domains and also with experimentalists, is often referred to as the "big data" challenge. The technical solutions that are used to alleviate this problem are well known: data partitioning, data ordering, parallel processing... There is therefore a strong convergence of the techniques implemented in parallel Navier-Stokes solvers and the "big data" post-processing tools.

This recommendation is an extension to the development of in-situ and transit post processing which will allow to post process on the fly the data just after being computed in order to maximise data reuse (for performance) and minimize data movement (for energy saving).

Objectives of the Recommendation

Data mining in large-scale turbulent simulations applied climate, combustion, traditional CFD, astrophysics, fusion … to become more and more difficult because of the size and the complexity of data generated. The extraction of large vortices or large-scale wrinkling of premixed flames for instance in combustion both require to analyse large portions of the computational domain or to perform time averages or modal decompositions of the flow field (Proper Orthogonal Decomposition and Dynamic Mode Decomposition). These techniques have to be conducted in parallel because of the large amount of data to post-process and they often imply the inversion of large linear systems to find the system eigenvalues or to provide selective filtering. Moreover, these techniques have to be versatile enough to be applied to realistic geometries.

To address this data mining challenge, it is mandatory to develop a complete toolbox of efficient parallel algorithms based on:

1. Massively parallel high-order low-pass and band-pass filters
2. Conservative high-order interpolation kernels for the interpolation of fine grids to coarser grids
3. Massively parallel Singular Value Decomposition algorithms for Dynamic Mode Decomposition of large sets of data
4. Highly efficient linear solvers for symmetric matrices as those encountered in implicit filters

----