PACO 2019: 3rd Workshop on Power-Aware Computing

Tuesday 5 November 2019 - Wednesday 6 November 2019

Max Planck Institute for Dynamics of Complex Technical Systems

Book of Abstracts

The organization committee would like to express its gratitude to

- Computational Methods in Systems and Control Theory (CSC) research group at Max Planck Institute

and

- Megware

for sponsoring the workshop.

Contents

1

Opening

Author: Peter Benner[1]

[1] *Max Planck Institute for Dynamics of Complex Technical Systems*

Corresponding Author: benner@mpi-magdeburg.mpg.de

7

Parallel solution of large sparse systems by direct and hybrid methods

Author: Iain S. Duff[1]

[1] *STFC RAL, UK and Cerfacs, France*

Corresponding Author: iain.duff@stfc.ac.uk

We discuss a range of algorithms and codes for the solution of sparse systems that we have developed in an EU Horizon 2020 Project, called NLAFET, that finished on 30 April 2019.

We used two approaches to get good single node performance. For symmetric systems we used task-based algorithms based on an assembly tree representation of the factorization. We then used runtime systems for scheduling the computation on both multicore CPU nodes and GPU nodes [4]. The second approach was to design a new parallel threshold Markowitz algorithm 2 based on Luby' s method [5] for obtaining a maximal independent set in an undirected graph. This was a significant extension since our graph model is a directed graph.

We then extended the scope of these two approaches to exploit distributed memory parallelism. In the first case, we base our work on the block Cimmino algorithm [3] using the ABCD software package coded by Zenadi in Toulouse [6]. The kernel for this algorithm is the direct factorization of a symmetric indefinite submatrix for which we use the above symmetric code.

To extend the unsymmetric code to distributed memory, we use the Zoltan code from
Sandia 1 to partition the matrix to singly bordered block diagonal form and then use the above unsymmetric code on the blocks on the diagonal. We show the performance of our codes on industrial strength large test problems on a heterogeneous platform. Our codes that are available on github are shown to outperform other state-of-the-art codes.

1 E. BOMAN , K. D EVINE , L. A. F ISK , R. H EAPHY , B. H ENDRICKSON, C. V AUGHAN , U. C ATALYUREK , D. B OZDAG , W. M ITCHELL , AND J. TERESCO , Zoltan 3.0: Parallel Partitioning, Load-balancing, and Data Management Services; User's Guide, Sandia National Laboratories, Albuquerque, NM, 2007.
Tech. Report SAND2007-4748W http://www.cs.sandia.gov/Zoltan/ug html/ug.html.
2 T. A. D AVIS, I. S. D UFF , AND S. NAKOV, Design and implementation of a parallel markowitz threshold algorithm, Technical Report RAL-TR-2019-003, Rutherford Appleton Laboratory, Oxfordshire, England, 2019. NLAFET Working Note 22. Submitted to SIMAX.
[3] I. S. D UFF , R. G UIVARCH , D. RUIZ , AND M. Z ENADI, The augmented block Cimmino distributed method, SIAM J. Scientific Computing, 37 (2015), pp. A1248–A1269.
[4] I. S. D UFF , J. H OGG , AND F. LOPEZ, A new sparse symmetric indefinite solver using a posteriori threshold pivoting, Tech. Rep. RAL-TR-2018-012, Rutherford Appleton Laboratory, Oxfordshire, England, 2018. NLAFET Working Note 21. Submitted to SISC.
[5] M. L UBY , A simple parallel algorithm for the maximal independent set problem, SIAM J. Computing, 15 (1986), pp. 1036–1053.
[6] M. Z ENADI, The solution of large sparse linear systems on parallel computers using a hybrid implementation of the block Cimmino method., Thése de Doctorat, Institut National Polytechnique de Toulouse, Toulouse, France, décembre 2013.

Day I / **15**

Exploiting Nested Task-Parallelism in the LU Factorization of Hierarchical Matrices

Authors: Rocío Carratalá-Sáez[1]; Enrique S. Quintana Ortí[2]

[1] *Universitat Jaume I*

[2] *Universitat Politècnica de València*

Corresponding Author: rcarrata@uji.es

Hierarchical matrices (H-matrices) lie in-between dense and sparse scenarios. Therefore, it is natural to tackle the LU factorization of H-Matrices via a task-parallel approach, which has recently reported successful results for related linear algebra problems. In this work, we will describe how to discover the data-flow parallelism intrinsic to the operation at execution time, via the analysis of data dependencies based on the memory addresses of the tasks' operands. This is especially challenging for H-matrices, as the data structures dimensions vary during the execution.

Day I / **19**

Unleashing the sptrsv method in FPGAs

Authors: Federico Favaro[1]; Ernesto Dufrechou[2]; Pablo Ezzatti[3]; Juan Pablo Oliver[1]

[1] *Facultad de Ingeniería, Universidad de la República*

[2] *Universidad de la República*

[3] *HCL, INCO, FING, UDELAR*

Corresponding Author: ffavaro@fing.edu.uy

Field–Programmable Gate Arrays (FPGAs) as hardware accelerators offer great flexibility and performance, and recently are emerging as a more energy–efficient alternative than other many–core devices.

The traditional methods for FPGA design involve the use of low–level Hardware Description Languages such as VHDL or Verilog. These impose a vastly different programming model than standard software languages, and their use requires specialized knowledge of the underlying hardware, which is why FPGAs are not massively adopted by the High Performance Computing (HPC) community. However, more recently manufacturers are making efforts to adopt High Level Synthesis languages like C/C++, System C or OpenCL.

In this context, the purpose of this work is to explore the use of FPGAs in HPC applications involving Numerical Linear Algebra (NLA) operations, specifically in the special case of sparse NLA field. We include a brief review of the state of the art in this matter and, as a starting point, we develop and evaluate the sparse triangular linear system solver ($sptrsv$) using OpenCL. The experimental evaluation performed includes both, the runtime and the energy–consumption perspectives.

Day I / **20**

Towards an efficient many-core implementation of the IRKA

Authors: Matías Valdés[1]; Ernesto Dufrechou[1]; Pablo Ezzatti[2]; Jens Saak[3]

[1] *Universidad de la República*

[2] *HCL, INCO, FING, UDELAR*

[3] *Max Planck Institute for Dynamics of Complex Technical Systems*

Corresponding Author: mvaldes@fing.edu.uy

The modeling of physical phenomena as Linear Time Invariant systems is a common practise across science and industry. It is often the case that the order of these models is so large that it renders them unuseful at the time of simulating the studied system. In these cases, practitioners can appeal to Model Order Reduction (MOR) techniques, which departing from the original model produce a reduced one with much lower order and similar behaviour.

One of the most well known techniques is the Iterative Rational Krylov Algorithm (IRKA). The method departs from an initial Reduced Order Model, represented by a set of shifts, and iteratively refines this set until the shifts converge to an optimal ROM. This refinement involves, in each step, the solution of two shifted linear systems $(\sigma_i E - A)x = b$ and $(\sigma_i E - A)^T y = c$.

In this work we study the use of the BiCG algorithm to solve those sequences of dual pairs on heterogeneous CPU-GPU platforms. With this purpose, we compare the use of a direct solver, a CPU and a GPU implementation of the BiCG, from the runtime and energy consumption points of view.

Our initial results using the Rail test case, and fixing the amount of work for the BiCG solvers per IRKA iteration, seem to indicate that, for large test cases, using the accelerator can effectively reduce the runtime of the BiCG, increasing the energy consumtption only slightly. This result is promising, and motivates the work on the series of BiCG applications to improve their convergence. For example, the use of preconditioners and recycled Krylov spaces should be explored in the future.

Day I / 21

Automatic selection of GPU sparse triangular solvers based on energy consumption

Authors: Raúl Marichal[1]; Marcos Pastorini[1]; Ernesto Dufrechou[1]; Pablo Ezzatti[2]; Enrique S. Quintana Ortí[3]

[1] *Universidad de la República*

[2] *HCL, INCO, FING, UDELAR*

[3] *Universitat Politècnica de València*

Corresponding Author: rmarichal@fing.edu.uy

Preconditioned Krylov-subspace methods to solve general sparse linear systems are the computational bottleneck of the solution of many science and engineering problems. In particular, it is the preconditioner application on each iteration of the solver, the stage that concentrate the most of the processing time. Many times this stage implies the solution of a number of sparse triangular linear systems, which has motivated their study by the NLA and HPC community.

The parallelization of this procedure in accelerators such as GPUs is widely addressed in the literature. Up to recently, the dominant approach to tackle this operation was the the \textit{level-set} strategy, which relies on preprocessing the sparse matrix to determine sets of independent unknowns that can be solved for in parallel. The most established and mature example of this approach is the routine distributed with the cuSparse library of NVIDIA. In recent work some authors have proposed a number of routines based on the \textit{self-scheduling} paradigm. In this paradigm the execution schedule is decided dynamically as threads have to wait until their data dependencies are resolved by other threads. The experimental results comparing both approaches are not conclusive, and indicate that both paradigms are well suited for certain matrices and not for others.

In previous work we have used machine learning models to attempt to predict which will be the best performing variant of the routine for a given sparse matrix. Now we are interested in incorporating the dimension of energy consumption, studying the relation between runtime and power

consumption of each variant and using machine learning to decide which solver to use considering both runtime and enregy.

Day I / 25

Gingko's load-balancing COO SpMV on NVIDIA and AMD GPU architectures

Authors: YU-HSIANG TSAI[1]; Hartwig Anzt[1]; Terry Cojean[1]; Thomas Grützmacher[1]; Pratik Nayak[1]; Tobias Ribizel[1]

[1] *Karlsruhe Institute of Technology*

Corresponding Authors: yu-hsiang.tsai@kit.edu, terry.cojean@kit.edu

Efficiently processing unbalanced and irregular matrices on manycore architectures is a challenging problem. With the load-balancing Sparse Matrix Vector Multiplication (SpMV) based on the coordinate format (COO), we have designed an SpMV kernel that provides attractive performance across a wide range of matrices. In this contribution, we present the load-balancing COO SpMV kernel, elaborate on architecture-specific tuning knobs, and evaluate the performance and efficiency across different GPU architectures from AMD and NVIDIA.In this evaluation, we do not exclusively focus on runtime performance, but also consider metrics like bandwidth utilization, energy consumption, and performance-per-$. We also discuss the Ginkgo library in which the presented SpMV functionality is available. Ginkgo is a next-generation sparse linear algebra library able to run on multi- and manycore architectures. The library design is guided by combining ecosystem extensibility with heavy, architecture-specific kernel optimization. The library design is guided by combining ecosystem extensibility with heavy, architecture-specific kernel optimization. The software development cycle ensures production-quality code by featuring unit testing, automated configuration and installation, Doxygen code documentation, as well as a continuous integration and continuous benchmarking framework. Ginkgo is an open source effort licensed under BSD 3-clause and ships with the latest version of the xSDK package (v.0.5.0).

24

Iterative Refinement in Three Precisions

Authors: Erin Carson[1]; Nicholas J. Higham[2]

[1] *Charles University*

[2] *School of Mathematics, University of Manchester*

Corresponding Author: carson@karlin.mff.cuni.cz

Support for floating point arithmetic in multiple precisions is becoming increasingly common in emerging architectures.
For example, half precision is now available on the NVIDIA V100 GPUs, on which it runs twice as fast as single precision with a proportional savings in energy consumption. Further, the NVIDIA V100's half-precision tensor cores can provide up to a 16x speedup over double precision.

We present a general algorithm for solving an n-by-n nonsingular linear system Ax = b based on iterative refinement in three precisions. The working precision is combined with possibly different precisions for solving for the correction term and for computing the residuals. Our rounding error analysis of the algorithm provides sufficient conditions for convergence and bounds for the attainable normwise forward error and normwise and componentwise backward errors, generalizing and unifying many existing rounding error analyses for iterative refinement.

We show further that by solving the correction equations by GMRES preconditioned by the LU factors the restriction on the condition number can be weakened to allow for the solution of systems which are extremely ill-conditioned with respect to the working precision. Compared with a standard $Ax = b$ solver that uses LU factorization in single precision, these results suggest that on architectures for which half precision is efficiently implemented it will be possible to solve certain linear systems $Ax = b$ in less time and with greater accuracy.

We present recent performance results on the latest GPU architectures which show that this approach can result in practical speedups and also discuss recent work in extending this approach to iterative refinement for least squares problems.

Day I / **22**

Parallel multiprecision iterative Krylov subspace solver

Author: Xenia Rosa Volk[1]

Co-authors: Hartwig Anzt [1]; Thomas Grützmacher [1]

[1] *Karlsruhe Institute of Technology*

Corresponding Author: xenia-rosa.volk@web.de

The use of multiprecision numerics is becoming increasingly attractive as modern processor architectures often achieve significantly higher performance and throughput rates when using lower precision than IEEE double precision. Error analysis aims at investigating how rounding errors introduced by using different precision formats propagate throughout the algorithms and potentially impact the convergence of iterative schemes.

In my talk I present experimental results on using a multiprecision Arnoldi algorithm inside the GMRES iterative solver. Specifically, I analyze the scenario where the memory format for storing the Krylov-building system matrix is decoupled from the arithmetic format, and the memory access to the system matrix uses a more compact, lower precision format while preserving IEEE double precision in all arithmetic.

Day I / **11**

Energy-Time Analysis of Heterogeneous Clusters for EEG Classification

Authors: Juan José Escobar[1]; Julio Ortega[1]; Antonio F. Díaz[1]; Jesús González[1]; Miguel Damas[1]

[1] *University of Granada, Granada, Spain*

Corresponding Author: jortega@ugr.es

Power-aware computing introduces an additional dimension in the development of efficient parallel codes for heterogeneous computing architectures. Along with experimental frameworks that facilitate the accomplishment of experimental measures, there is a need for data analysis strategies and programming guidelines and strategies that jointly consider speed and consumption performance, among others. Our communication in the present edition of the PACO workshop describes the work developed on time-energy analysis of an electroencephalogram (EEG) classification problem implemented on heterogeneous clusters with nodes including CPU and GPU architectures, and the techniques proposed by our research group of the Department of Architecture and Computer Technology of the University of Granada in this topic [2, 3, 4, 5].

Our approach analyses the energy consumption and runtime behaviors of a parallel master-worker evolutionary algorithm according to the workload distribution among the GPU and CPU cores and their operating frequencies. In the kind of evolutionary procedure here considered, most of the workload corresponds to the fitness evaluation of the solutions that constitute the population evolved across the generations (iterations) of the algorithm. Thus, the way the solutions are allocated to GPU and CPU cores will determine the efficiency in performance and energy behavior of the parallel algorithm once a suitable workload distribution among the available cores is attained. We have defined models [3] for runtime and energy consumption that have been fitted to the experimental results by multiple linear regression with values of the R^2-statistics that show high statistical significance in all the cases.

The implemented program is an *OpenCL* (version 1.2) code (compiled with *GCC* 4.8.5) for a multi-objective feature selection problem corresponding to a BCI task [6] applied to a dataset recorded in the BCI Laboratory of the University of Essex. Each pattern is an electroencephalogram (EEG) described by 3600 features corresponding to 12 features per each of the 20 temporal segments and 15 electrodes [6].

From these models, it is possible to define a workload distribution that takes into account both the runtime and the energy consumption through a bi-objective cost function that properly weights both objectives. As many useful bioinformatics and data mining applications are tackled by programs with a similar profile to that of the here considered parallel master-worker procedure, the conclusions about energy consumption and speedups can be taken into account many times, and the proposed energy-aware scheduling approach could be frequently applied.

Regarding the operating mode control, the standard ACPI (Advanced Configuration and Power Interface) [9] includes mechanisms to manage and save energy, provides a suitable control of the BIOS functioning and gives information about the configuration and the control of the processor states related to the energy consumption $(C_0, C_1, C_2, C_3, \ldots, C_n)$ and performance $(P_0, P_1, \ldots, P_n)$. Likewise, the *Linux* kernel implements the CPUFreq (CPU Frequency scaling) subsystem [7], which allows the operating system, either automatically through the events generated by the ACPI, or through user program calls, to change the operating frequency of the processor for energy saving. The so-called governors [8] are included in the CPUFreq subsystem, to implement specific policies for controlling the processor clock. The interface to use these services at the user level can be found in the header file linux/cpufreq.h 1.

Acknowledgments
This work was supported by project PGC2018-098813-B-C31 (Spanish *Ministerio de Ciencia, Innovación y Universidades*), and by European Regional Development Funds (ERDF).

References

1. Brodowski, D.: The linux/cpufreq.h header file. https://github.com/torvalds/linux/blob/master/tools/power/cpupow accessed: 2019-10-07

2. Escobar, J.J., Ortega, J., Damas, M., Savran, R., Gan, J.Q.: Energy-time analysis of convolutional neural networks distributed on heterogeneous clusters for EEG classification. In: Advances in Computational Intelligence. IWANN 2019. pp. 895–907. Lecture Notes in Computer Science, Springer, Gran Canaria, Spain (June 2019). https://doi.org/10.1007/978-3-030-20518-8_74

3. Escobar, J.J., Ortega, J., Díaz, A.F., González, J., Damas, M.: Energy-aware load balancing of parallel evolutionary algorithms with heavy fitness functions in heterogeneous CPU-GPU architectures. Concurrency and Computation. Practice and Experience 31(6), e4688 (2019). https://doi.org/10.1002/cpe.4688

4. Escobar, J.J., Ortega, J., Díaz, A.F., González, J., Damas, M.: A power-performance perspective to multiobjective electroencephalogram feature selection on heterogeneous parallel platforms. Journal of Computational Biology 25(8), 882–893 (2019). https://doi.org/10.1089/cmb.2018.0080

5. Escobar, J.J., Ortega, J., Díaz, A.F., González, J., Damas, M.: Time-energy analysis of multilevel parallelism in heterogeneous clusters: the case of EEG classification in BCI tasks. Journal of Supercomputing 75(7), 3397–3425 (2019). https://doi.org/10.1007/s11227-019-02908-4

6. Ortega, J., Asensio-Cubero, J., Gan, J.Q., Ortiz, A.: Classification of motor imagery tasks for BCI with multiresolution analysis and multiobjective feature selection. BioMedical Engineering OnLine 15(1), 73 (2016). https://doi.org/10.1186/s12938-016-0178-x

7. The Linux Kernel: CPU performance scaling. https://www.kernel.org/doc/html/v4.15/admin-guide/pm/cpufreq.html, accessed: 2019-10-07
8. The Linux Kernel: CPUFreq governors. https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt, accessed: 2019-10-07
9. UEFI Forum, Inc.: Advanced configuration and power interface (ACPI) specification. https://uefi.org/sites/default/files/resou accessed: 2019-10-07

Day I / 9

Convolutional Neural Nets for the Run Time and Energy Consumption Estimation of the Sparse Matrix–Vector Product

Authors: Maria Barreda[1]; Manuel F. Dolz[1]; M. Asunción Castaño[1]; Enrique S. Quintana-Ortí[2]

[1] *Universitat Jaume I*

[2] *Universitat Politècnica de València*

Corresponding Author: dolzm@uji.es

Introduction. Modeling the execution time and the energy efficiency of the Sparse Matrix-Vector product (SpMV) on a current CPU architecture is especially complex due to i) irregular memory accesses; ii) indirect memory referencing; and iii) low arithmetic intensity. While analytical models may yield accurate estimates for the total number of cache hits/misses, they often fail to predict accurately the total execution time and energy consumption. In this work, we depart from the analytic approach to instead leverage Convolutional Neural Networks (CNNs) in order to provide an effective estimation of the run time and energy consumption of the SpMV operation.

Modeling. The memory-bound nature of SpMV operation (y = Ax) is mainly due to the low nonzero densities and the irregular sparsity patterns in matrix A which, in general, dictate a considerable volume of cache misses and DRAM memory accesses to the x vector. Therefore, the vector storing the column indices (*vpos*) from the CSR format array can be regarded as a key element to understand the distinct arithmetic-to-memory access intensities and predict both the execution time and energy consumption. With this idea in mind, we design a CNN where the inputs are the values of the *vpos* array as a one-dimensional image of the sparse matrix A that captures the order in which the entries of x are retrieved from memory. Once trained, the filters in the convolutional layers should be capable of capturing meaningful features of the *vpos* array in order to yield useful run time and energy consumption estimations via the relation between flops and cache hits/misses. In the methodology proposed to tackle the SpMV modeling problem, a CNN receives the *vpos* array as an input. However, given that sparse matrices may present large variations on their size and number of nonzero entries (nnz), we split the *vpos* input array into chunks (or blocks) of size b=250 to design a CNN with a constant number of inputs (neurons). With it, the estimated total execution time and energy consumption for a matrix can be calculated as the aggregation of the time and energy estimations for each of the blocks.

Evaluation. We tackle the performance and energy consumption estimation as a regression problem, resulting in a same CNN model that is trained once per metric (time and energy). The training of the networks was performed on a set of blocks of sparse matrices from the SuiteSparse matrix collection labeled with the corresponding SpMV execution time and energy consumption drawn by an Intel Xeon Haswell core, running at 2.4 GHz. These metrics were measured using Intel RAPL performance counters. The results of the testing process reveal that the relative mean error for the set of testing matrices for the proposed models is lower than 1.7% for the execution time and less than 3.1% for the energy consumption. On a separated experiment, we extended our analysis on varying frequencies, ranging from 1.2 to 2.4 GHz. These results reveal that decreasing the processor frequency slightly improves the relative mean error.

Conclusions. We proposed a CNN models to estimate the execution time and energy consumption of the SpMV operation. The designed models incorporate a blockwise strategy which makes

the CNN architecture independent of the sparse matrix size, allowing users to obtain run time and energy consumption estimations prior the execution or even when the target architecture is no accessible.

Day I / 16

Revisiting the idea of multiprecision block-Jacobi preconditioning - where do we stand 2 years later?

Authors: Hartwig Anzt[1]; Terry Cojean[1]; Thomas Grützmacher[1]; Enrique Quintana-Orti[2]

[1] *Karlsruhe Institute of Technology*

[2] *Polytechnic University of Valencia*

Corresponding Author: hartwig.anzt@kit.edu

At the PACO workshop 2017, we presented the idea of decoupling the memory precision from the arithmetic precision, and storing a block-Jacobi preconditioner such that the precision format of each diagonal block is optimized to the numerical characteristics. The idea is to reduce the pressure on the memory bandwidth while preserving regularity of the preconditioner and the convergence of the top-level iterative solver. This is expected to render attractive resource savings as the performance and energy footprint of memory-bound applications strongly correlates to the data transfer volume. Two years later, we review the effectiveness of this idea by evaluating a sophisticated high performance implementation of the adaptive precision block-Jacobi preconditioner for GPU architectures that is implemented in the Ginkgo numerical linear algebra library. We present performance results revealing the attractiveness of the approach, with the runtime savings even exceeding the expectations. We also consider the use of non-standard precision formats, and the problem-specific trade-off between preconditioner application cost and preconditioner effectiveness.

Day I / 26

In-application energy measurement on Megware SlideSX systems

Author: Martin Köhler[1]

Co-authors: Jens Saak [1]; in cooperation with MEGWARE [2]

[1] *Max Planck Institute for Dynamics of Complex Technical Systems*

[2] *MEGWARE Computer Vertrieb und Service GmbH*

Corresponding Author: koehlerm@mpi-magdeburg.mpg.de

The design of energy efficient applications requires a proper energy measurement inside the compute servers. The Megware SlideSX chassis for HPC servers provide an energy measurement directly between the power supply and the mainboard of the system with a sampling rate up to 100Hz. This enables the users to detect the energy consuming parts of their applications. In order to obtain the energy measurements from the operating system a plugin for the SLURM workload manager is developed. This plugin uses a server-client model to communicate with the users' applications. Beside the design of the measurement plugin with SPANK (SLURM Plug-in Architecture for Node and job Kontrol) the presentation will cover the usage in some example applications.

Massively Parallel & Low Precision Accelerator Hardware as Trends in HPC - How to use it for large scale simulations allowing high computational, numerical and energy efficiency with application to CFD

Author: Stefan Turek[1]

[1] *TU Dortmund*

Corresponding Author: stefan.turek@tu-dortmund.de

The aim of this talk is to present and to discuss how modern, resp., future High Performance Computing (HPC) facilities regarding massively parallel hardware with millions of cores together with very fast, but low precision accelerator hardware can be exploited in numerical simulations so that a very high computational, numerical and hence energy efficiency can be obtained. Here, as prototypical extreme-scale PDE-based applications, we concentrate on nonstationary flow simulations with hundreds of millions or even billions of spatial unknowns in long-time computations with many thousands up to millions of time steps. For the expected huge computational resources in the coming exascale era, such type of spatially discretized problems which typically are treated sequentially, that means one time after the other, are still too small to exploit adequately the huge number of compute nodes, resp., cores so that further parallelism, for instance w.r.t. time, might get necessary.

In this context, we discuss how "parallel-in-space simultaneous-in-time" Newton-Multigrid approaches can be designed which allow a much higher degree of parallelism. Moreover, to exploit current accelerator hardware in low precision (for instance, GPUs or TPUs), that means mainly working in single precision or even half precision, we discuss the concept of "prehandling" (in contrast to "preconditioning") of the corresponding ill-conditioned systems of equations, for instance arising from Poisson-like problems. Here, we assume a transformation into an equivalent linear system with similar sparsity but with much lower condition numbers so that the use of low-precision hardware might get feasible. In our talk,
we provide for both aspects preliminary numerical results as "proof-of-concept" and discuss the open problems, but also the challenges, particularly for incompressible flow problems.

12

Parallel Algorithms for CP, Tucker, and Tensor Train Decompositions

Author: Grey Ballard[1]

[1] *Wake Forest University*

Corresponding Author: ballard@wfu.edu

Multidimensional data, coming from scientific applications such as numerical simulation, can often overwhelm the memory or computational resources of a single workstation. In this talk, we will describe parallel algorithms and available software implementations for computing CP, Tucker, and Tensor Train decompositions of large tensors. The open-source software is designed for clusters of computers and has been benchmarked on various supercomputers. The algorithms are scalable, able to process terabyte-sized tensors and maintain high computational efficiency for 100s to 1000s of processing nodes. We will detail the data distribution and parallelization strategies for the key computational kernels within the algorithms, which include the matricized-tensor times Khatri-Rao product, computing (structured) Gram matrices, and tall-skinny QR decompositions.

Day II / 14

Energy Efficiency of Nonlinear Domain Decomposition Methods

Author: Axel Klawonn[1]

Co-authors: Martin Lanser [1]; Oliver Rheinbach [2]; Gerhard Wellein [3]; Markus Wittmann [3]

[1] *Universität zu Köln*

[2] *TU Bergakademie Freiberg*

[3] *Friedrich-Alexander-Universität Erlangen*

Corresponding Author: axel.klawonn@uni-koeln.de

A nonlinear domain decomposition (DD) solver is considered with respect to improved energy efficiency. In this method, nonlinear problems are solved using Newton's method on the subdomains in parallel and in asynchronous iterations. The method is compared to the more standard Newton-Krylov approach, where a linear domain decomposition solver is applied to the overall nonlinear problem after linearization using Newton's method. It is found that in the nonlinear domain decomposition method, making use of the asynchronicity, some processor cores can be set to sleep to save energy and to allow better use of the power and thermal budget. Energy savings on average for each socket up to 77% (due to the RAPL hardware counters) are observed compared to the more traditional Newton-Krylov approach, which is synchronous by design, using up to 5120 Intel Broadwell (Xeon E5-2630v4) cores. The total time to solution is not affected. On the contrary, remaining cores of the same processor may be able to go to turbo mode, thus reducing the total time to solution slightly. Last, we consider the same strategy for the ASPIN (Additive Schwarz Preconditioned Inexact Newton) nonlinear domain decomposition method and observe a similar potential to save energy

Day II / 13

Domain decomposition methods in FreeFEM with ffddm

Authors: Pierre-Henri Tournier[1]; Pierre Jolivet[2]; Frédéric Hecht[1]; Frédéric Nataf[1]

[1] *Sorbonne Université, CNRS, Université de Paris, Inria, Laboratoire Jacques-Louis Lions, F-75005 Paris, France*

[2] *IRIT, CNRS, Toulouse, France*

Corresponding Author: tournier@ljll.math.upmc.fr

The idea behind ffddm is to simplify the use of parallel solvers in the open source finite element software FreeFEM. The ffddm framework is entirely written in the FreeFEM language. Thanks to ffddm, FreeFEM users have access to high-level functionalities for specifying and solving their finite element problems in parallel using scalable two-level Schwarz domain decomposition methods. The coarse space correction can be built either from a coarse mesh or using the GenEO (Generalized Eigenvalue in the Overlap) approach 1. One can also use ffddm for learning and prototyping domain decomposition methods without compromising efficiency.
The presentation includes numerical experiments for diffusion, elasticity and wave propagation problems 2. In particular, large scale experiments illustrate the use of multilevel methods for the iterative solution of the coarse problem when it gets too large. This is a joint work with Frédéric Nataf, Pierre Jolivet and Frédéric Hecht. Numerical results are obtained using HPC resources from GENCI-CINES (Grant 2017- A0020607330).

1 Spillane, N. and Dolean, V. and Hauret, P. and Nataf, F. and Pechstein, C. and Scheichl, R. Abstract Robust Coarse Spaces for Systems of PDEs via Generalized Eigenproblems in the Overlap. Numerische Mathematik, 2014.
2 M. Bonazzoli, and V. Dolean, and I. G. Graham and E. A. Spence and P.-H. Tournier Domain decomposition preconditioning for the high-frequency time- harmonic Maxwell equations with absorption. Mathematics of Computation, 2019.

Day II / 23

Evaluating asynchronous Schwarz solvers for Exascale.

Authors: Pratik Nayak[1]; Hartwig Anzt[1]

[1] *Karlsruhe Institute of Technology*

Corresponding Author: pratik.nayak@kit.edu

With the commencement of the exascale computing era, we realize that the majority of the leadership supercomputers are heterogeneous and massively parallel even on a single node with multiple co-processors such as GPU's and multiple cores on each node. For example, ORNL's Summit accumulates six NVIDIA Tesla V100's and 42 core IBM Power9's on each node.

At this scale of parallelism, the traditional bulk-synchronous programming model will not be able to leverage the compute power of the hardware efficiently. Hence, it is necessary to develop and study asynchronous algorithms that circumvent this issue. The Schwarz methods are a class of Domain decomposition solvers which allow for decomposing the main domain into smaller subdomains, solve the sub-domain problems in parallel and exchange the information between iterations.

In this study, we examine the asynchronous version of one of these (Restricted Additive Schwarz) solvers where we do not explicitly synchronize, but allow for communication of the data between the subdomains to be completely asynchronous. Thereby, we remove the bulk synchronous nature of the algorithm. We accomplish this using the RDMA functions of MPI. We study the benefits of using such an asynchronous solver over its synchronous counterpart on both multi-core and on multiple GPU's. Detecting convergence in these asynchronous solvers is tricky, and careful consideration of hardware is necessary to make efficient use of the hardware. We show that this concept can nevertheless render attractive runtime benefits over the synchronous counterparts.

Day II / 6

S-Step Enlarged Conjugate Gradient Methods

Author: Sophie Moufawad[1]

[1] *American University of Beirut (AUB)*

Corresponding Author: sm101@aub.edu.lb

In many numerical simulations, there is a need to solve a sparse linear system ($Ax = b$) at every iteration. The solution of these linear systems, using iterative methods such as Krylov Subspace Methods, consumes around 80% of the simulation's runtime on modern architectures. Recently, enlarged Krylov subspace methods were introduced in the aim of reducing communication and speeding-up the convergence of Krylov subspace methods, thus minimizing the energy consumption. These enlarged Krylov subspace methods consist of enlarging the Krylov subspace by a maximum of t vectors per iteration based on a domain decomposition of the graph of A. In this talk, we present s-step enlarged Krylov subspace methods, whereby s iterations of enlarged Krylov subspace methods are merged to further reduce communication. We introduce several s-step enlarged CG versions (SRE-CG, MSDO-CG) and discuss their numerical stability.