Scalable Hierarchical Algorithms for stochastic PDEs and UQ

Alexander Litvinenko[1], **Gustavo Chávez**[2], **David Keyes**[2], **Hatem Ltaief**[2], **Rio Yokota**[2]

SRI Uncertainty Quantification[1] and Extreme Computing Research Centers at KAUST[2]

alexander.litvinenko@kaust.edu.sa

1. Abstract

$\mathcal{H}$-matrices and Fast Multipole (FMM) are powerful methods to approximate linear operators coming from partial differential and integral equations as well as speed up computational cost from quadratic or cubic to log-linear ($\mathcal{O}(n \log n)$), where n number of degrees of freedom in the discretization. The storage is reduced to the log-linear as well. This hierarchical structure is a good starting point for parallel algorithms. Parallelization on shared and distributed memory systems was pioneered by Kriemann [1,2]. Since 2005, the area of parallel architectures and software is developing very fast. Progress in GPUs and Many-Core Systems (e.g. XeonPhi with 64 cores) motivated us to extend work started in [1,2,7,8].

2. Fast Multipole (FMM) and Hierarchical ($\mathcal{H}$-) matrices

Communication is the bottleneck for any algorithm as it approaches the limit of its parallel scalability. We provide new upper bounds for the communication complexity of FMM

Reference	Processes		Data per Process		Communication complexity
Teng 98	$\mathcal{O}(P)$		$\mathcal{O}\left((N/P)^{2/3}(\log N+\mu)^{1/3}\right)$		$\mathcal{O}\left(P(N/P)^{2/3}(\log N+\mu)^{1/3}\right)$
Lashuk 09	$\mathcal{O}(\sqrt{P})$		$\mathcal{O}\left((N/P)^{2/3}\right)$		$\mathcal{O}\left(\sqrt{P}(N/P)^{2/3}\right)$
Yokota *et al.* 14	Global	Local	Global	Local	Global + Local
	$\mathcal{O}(\log P)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}\left((N/P)^{2/3}\right)$	$\mathcal{O}\left(\log P+(N/P)^{2/3}\right)$

Table 1: *Communication complexity of FMM, N is problem size, P is number of processors.*

Aim:
To improve results of Kriemann [1,2] about parallel $\mathcal{H}$-matrix implementation on shared memory systems on new architectures.

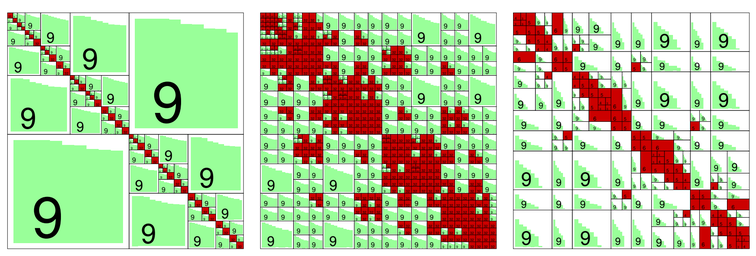
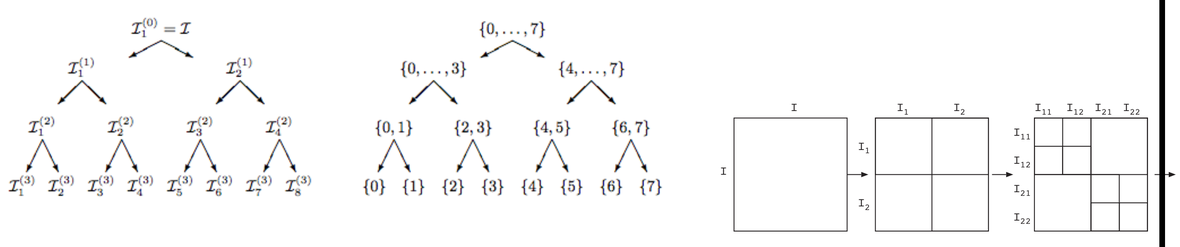


Figure 1: *Examples of $\mathcal{H}$-matrix approximation of exponential covariance matrix with weak and standard admissibility conditions, BEM matrix*

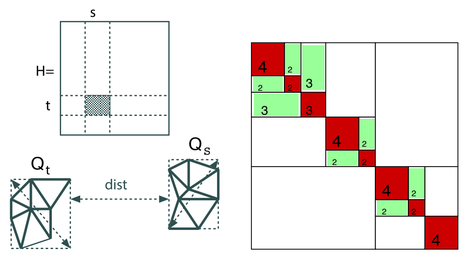
Methodology:
1. Build cluster tree T_I and block cluster tree $T_{I\times I}$.



2. For each $(t \times s) \in T_{I\times I}$, $t, s \in T_I$, check admissibility condition

$\min\{diam(Q_t), diam(Q_s)\} \leq \eta \cdot dist(Q_t, Q_s)$.

if(adm=true) then $M|_{t\times s}$ is a rank-k matrix block
if(adm=false) then divide $M|_{t\times s}$ further or define as a dense matrix block, if small enough. Grid $\rightarrow$ cluster tree (T_I) + admissibility condition $\rightarrow$ block cluster tree ($T_{I\times I}$) $\rightarrow$ $\mathcal{H}$-matrix $\rightarrow$ $\mathcal{H}$-matrix arithmetics.



Operation	Sequential Complexity (Hackbusch et al. '99-'06)	Parallel Complexity (Kriemann '05)
$storage(M)$	$N=\mathcal{O}(kn\log n)$	$\frac{N}{P}$
Mx	$N=\mathcal{O}(kn\log n)$	$\frac{N}{P}$
$M_1\oplus M_2$	$N=\mathcal{O}(k^2n\log n)$	$\frac{N}{P}$
$M_1\odot M_2$, M^{-1}	$N=\mathcal{O}(k^2n\log^2 n)$	$\frac{N}{P}+\mathcal{O}(n)$
$\mathcal{H}$-LU	$N=\mathcal{O}(k^2n\log^2 n)$	$\frac{N}{P}+\mathcal{O}(\frac{k^2n\log^2 n}{n^{1/d}})$
$\mathcal{H}$-matrix conversion	$N=\mathcal{O}(k^2n\log^2 n)$	$\frac{N}{P}$

Further Applications: Matrix exponential allows us to solve ODEs

$$\dot{x}(t)=Ax(t), \quad x(0)=x_0, \quad \rightarrow x(t)=\exp(tA)x_0$$

Other matrix function: use representation by the Cauchy integral

$$f(A)=\frac{1}{2\pi i}\oint_\Gamma f(t)(A-tI)^{-1}dt f(A)\approx\sum_{j=1}^{k} w_j f(t_j)(A-t_jI)^{-1}$$

[Hackbusch, TR 4/2005, MPI]
Matrix equations (Arises in the context of optimal control problems)

(Lyapunov) $AX+XA^T+C=0$, (Sylvester) $AX-XB+C=0$, (Riccati) $AX+XA^T-XFX+C=0$. (1)

where A, B, C, F are given matrices and X is the solution (a matrix).
Solution of the Ricati equations Let $A\in\mathbb{R}^{n\times n}$, $B\in\mathbb{R}^{m\times m}$, and let $\sigma(A)<\sigma(B)$, then

$$X:=\int_0^\infty \exp(tA)C\exp(-tB)dt, \;\; X_k:=\sum_{j=0}^{k} w_j\exp(t_jA)C\exp(-t_jB) \qquad (2)$$

solves the Riccati equation [Hackbusch, TR 4/2005, MPI].

3. HiCMA: Hierarchical Computations on Multicore with hardware Accelerators

HiCMA aims to tackle the challenge that is facing the linear algebra community due to an unprecedented level of on-chip concurrency, introduced by the manycore era. **HiCMA** is a high performance numerical library designed for efficient compressions and fast implementations of ($\mathcal{H}$-matrix) algorithms across a range of architectures: multicore, accelerators and ARM processors. The core idea is to redesign the numerical algorithms (as implemented in $\mathcal{H}$-Lib) and to formulate them as successive calls to computational tasks, which are then scheduled on the underlying system using a runtime system to ensure load balancing. The algorithm is then represented as a Directed Acyclic Graph (DAG), where nodes represent *hierarchical* tasks and edges show the data dependencies between them. There are various paramount factors such as admissibility condition (block partition), the minimal leaf size (diagonal block) and the $\mathcal{H}$-matrix rank (numerical accuracy), which influence the performance of HiCMA. An auto-tuning framework is therefore critical to select the optimal parameters for a given application (FEM, BEM), which exhibits low rank matrix computations.

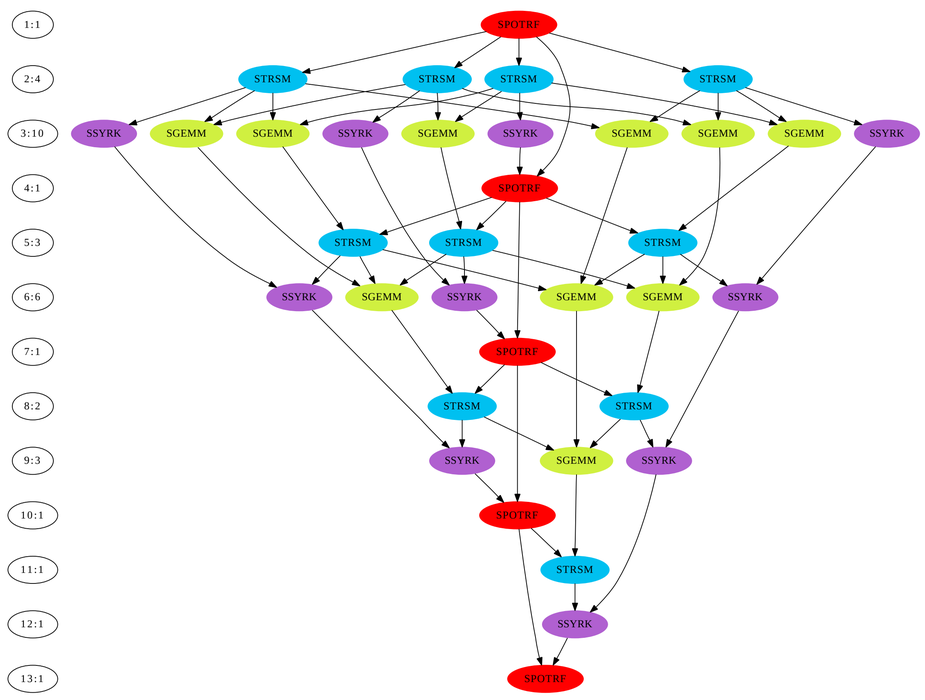


Figure 3: *Directed Acyclic Graph for a Cholesky factorization: DAG height corresponds to the length of the critical path and the DAG width to the degree of concurrency.*

4. New algorithmic ideas for parallel $\mathcal{H}$-matrices

An hierarchical matrix is composed of a mixture of low-rank blocks (green) and dense blocks (red). For maximum performance (c.f. Figure 4 left), the size of the dense blocks is typically a small constant (8, 16, 32, or 64); the definitive number is processor-dependent.

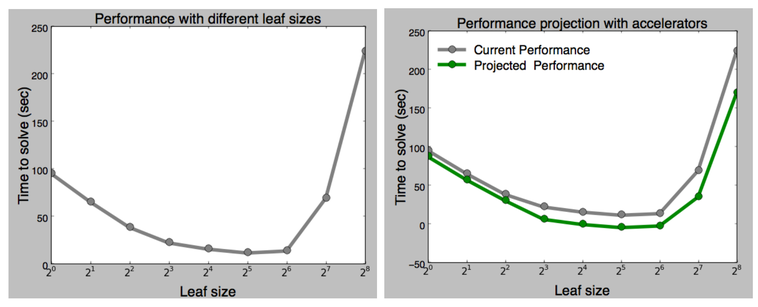


Figure 4: *Performance effect for different leaf sizes (left), and the projected performance using accelerators (right). $\mathcal{H}$-LU factorization of the 2D Poisson model problem on a 1024^2 points grid.*

The main observation is that multi-processors are coupled with accelerators. Furthermore, there are mature dense linear algebra libraries that run in such hardware, and in some cases the low-rank and dense computations can be done concurrently, so there is an opportunity of hiding the communication with the accelerator. This will enable to work with larger dense blocks, at minimal burden to the overall performance for kernels that demand larger dense blocks, or with improved performance for the typical type of kernels (c.f. Figure 4 right).

5. Applications

Parallel $\mathcal{H}$-matrix library will allow us to model very large, detailed covariance matrices which are used in simulation of oil reservoirs, climate forecast, data assimilation on irregular grids etc. Additionally we will able to solve numerically convection-diffusion, integral, acoustic, Helmholtz, Maxwell and matrix equations much faster with lower storage cost (all these examples are already available in HLIB [9] and HLIBPro [1]).

References

1. R. Kriemann, $\mathcal{H}$-LibPro documentation, http://www.hlibpro.com
2. R. Kriemann, Dissertation, University of Kiel, Germany, 2004.
3. R. Yokota, G. Turkiyyah, D. Keyes, Communication Complexity of Fast Multipole Method and its Algeb. Variants, 2014.
4. R. Yokota et al, Scalable Hierarchical Algorithms for eXtreme Computing', Supercomp. Frontiers and Innov., 2014.
5. R. Yokota, J. Pestana, H. Ibeid, and D. E. Keyes. Fast Multipole Preconditioners for Sparse Matrices Arising from Elliptic Equations. *arXiv:1308.3339v2*, 2014.
6. E. Agullo et al., Num. Lin. Alg. on Emerging Architec.: PLASMA and MAGMA Projects. J. of Ph. 180.1, 2009.
7. R. Kriemann, $\mathcal{H}$-LU Factorisation on Many-Core Systems, MIS MPG preprint 5, 2014.
8. M. Izadi, Dissertation, Hierarchical Matrix Techniques on Massive Parallel Computers, Uni Leipzig, Germany, 2012.
9. L. Grasedyck, S. Boerm, $\mathcal{H}$-matrix library, http://www.hlib.org