Acceleration of Electronic Structure Codes on Heterogeneous Hardware

Marcello Puligheddu, Ian J. Bush
Science and Technology Facilities Council

April 22nd 2024

GPUs in *ab initio* electronic structure codes

The state of the system is the solution of the Roothaan equation

$$F\epsilon = SC\epsilon$$

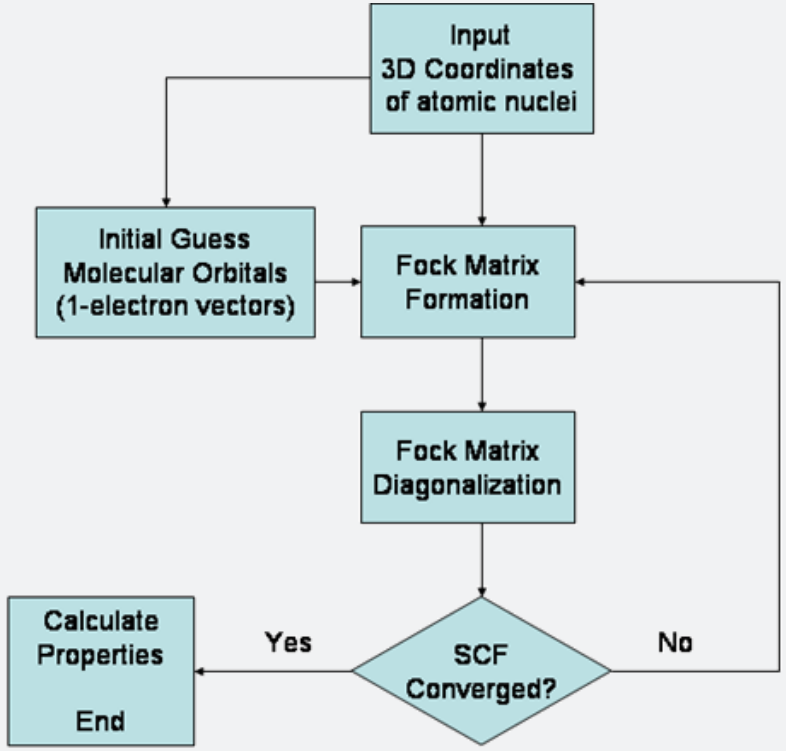
The majority of the computational work in these codes consists of two steps

- Using the density matrix C to form the Fock Matrix F
- Diagonalizing the Fock Matrix F to obtain the new density C

These steps are repeated in the Self-Consistent Fields (SCF) procedure

Step 1 requires the evaluation of an **enormous** number of integrals
Step 2 requires expensive linear algebra on large matrices

Make these calculations more affordable by exploiting the computational power provided by modern high-end GPUs



Benchmarks of Eigenvalues Solvers on GPUs

ELPA (Eigenvalue Solvers For ~~Peta~~Exaflop Applications) is a software library built to scale and to run on heterogenous architectures, including GPUs.

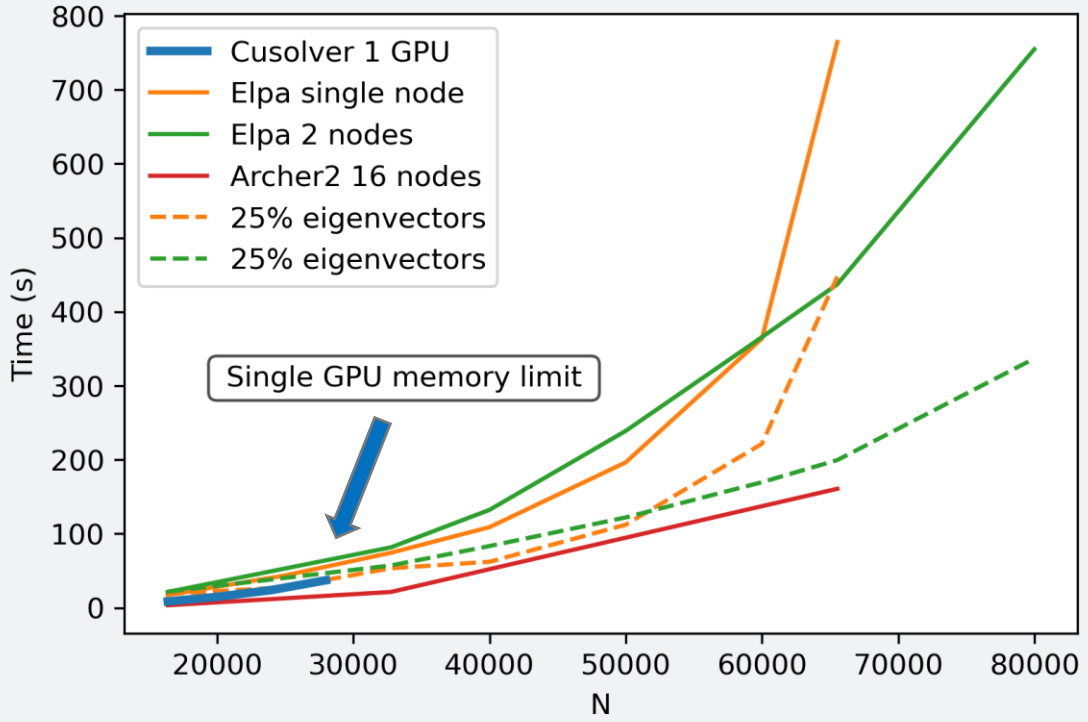
The interface is similar to ScaLAPACK, different routine for matrix diagonalization

ELPA can also compute a partial solution. This can result in a large saving of computational time

Tested on 2 nodes with 4 A100 40GB per node

- NVIDIA cuSolver can be faster than ELPA running on multiple GPUs (*)
- Solving the partial problem significantly reduces the time to solution.

Very quick project before the release of cublasMg



GPU Acceleration of the Calculation of the Fock Matrix

We focus on the contribution to the Fock matrix from the exact Hartree-Fock Exchange

$$F_{ik} \mathrel{-}= D_{jl}\ (ijkl)$$

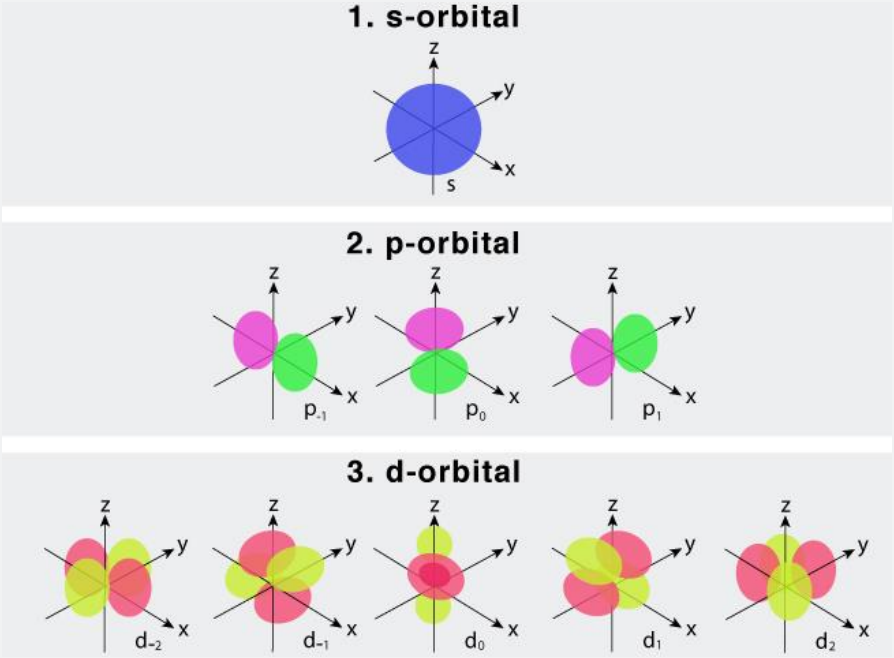
We wish to use to GPUs to accelerate the calculation of the 4 centres – 2 electrons integrals

$$(ijkl) = \int d^3x \int d^3x' \frac{\varphi_i(x)\varphi_j(x)\varphi_k(x')\varphi_l(x')}{|x - x'|}$$

(Also called more generally Electronic Repulsion Integrals ERI)

Where the φ are Gaussian orbitals of the type $\varphi = Y_L^m\ e^{-\alpha\, x^2}$

A basis function is a linear combination of these primitives



GPU Acceleration of the Calculation of the Fock Matrix

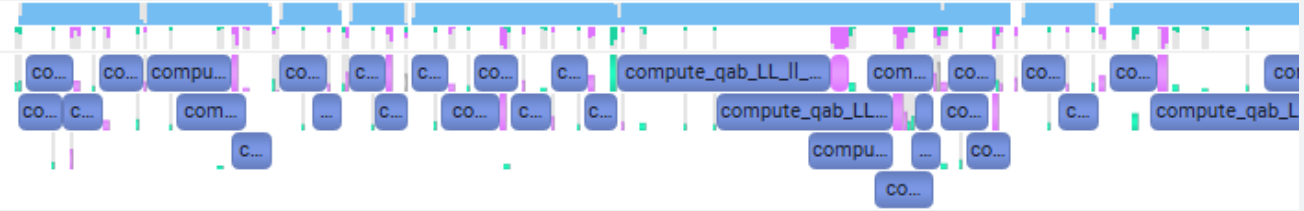
The calculation in the integrals is a complex process. Accelerating its using GPUs is not trivial.

What is the best approach to the efficient calculation on GPUs?
How to integrate into existing codes?
How to balance the load across multiple GPUs?

Start with the simpler processes and move our way up to the most complex components.

- First steps: use the Overlap matrix as a simple test-bed
- Keep the structure flexible. Python helps with prototyping
- At the end, compute the 4 centres – 2 electrons integrals on GPUs

No free lunch -> (often) the routines that take more time are also the harder to accelerate using GPUs



Profiler trace of the (much simpler) calculation of the Overlap matrix in CP2K

GPU Acceleration of the Calculation of the Fock Matrix

Integrals of the type $(ssss)$ can be done analytically.
Integrals of higher moments are iteratively built from the $(ssss)$ and their ~derivative
We use the Obara-Saika recurrence relations and Head-Gordon-Pople method

E.g. $(psss) = \left(\bar{R}_{ij} - \bar{R}_i\right)(ssss) + \left(\bar{R}_{ijkl} - \bar{R}_{ij}\right)(ssss)'$

We repeatedly apply these and other operations to increase the moment of the integrals

E.g. The $(dddd)$ integrals require 86 steps

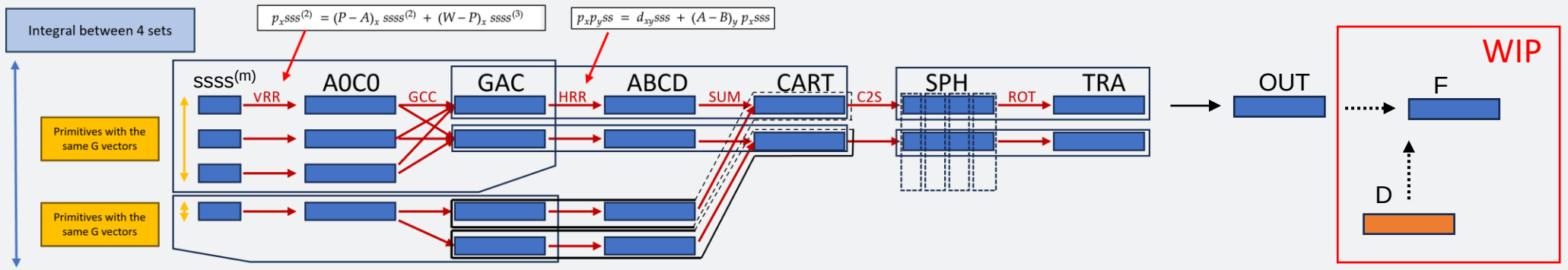
We precompute a plan of execution and refine it to:

- Fuse similar operations to reduce the number of steps
- Precompute memory offsets and array sizes
- Identify transformations that can be done concurrently -> only synchronize when necessary
- Contract primitives into the generalized contracted basis set

We prepare this plan on the CPU, save it and run it on the GPU.

GPU Acceleration of the Calculation of the Fock Matrix

The control logic on the GPU code itself is relatively simple, the complex code is kept on the CPU side.



The calculation is done in phases. The parallelisation of work between SM is not trivial.

Some operations can be split naturally to 1 Integrals -> 1 SM
Other operations are better split between SMs

- 1 Streaming Multiprocessor per GAC, 1-32 threads per Primitive
- 1 Streaming Multiprocessor per HRR transformation, 1 thread per element

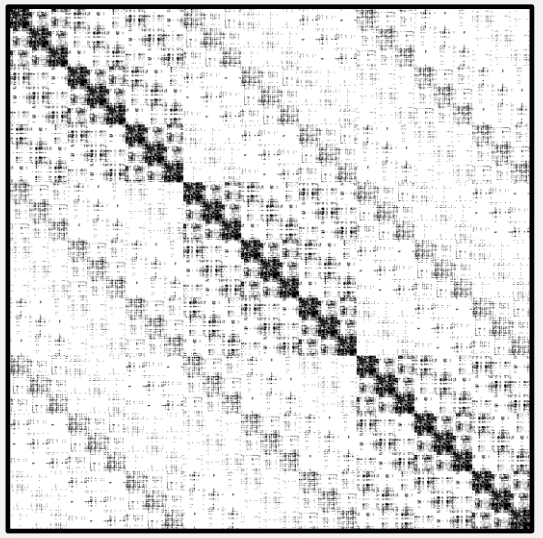
GPU Acceleration of the Calculation of the Fock Matrix

I only described the innermost core of the ERIs calculation

(1) I described a simplified view. Real ERIs are symmetric, periodic, screened linear combinations of integrals
- Each integral is relatively small - nanoseconds to maybe fraction of a millisecond –
- How we pack the integrals to expose parallelism on the GPU may be just as important
- Complexity hinders parallelism (2 body -> 4 body transition)

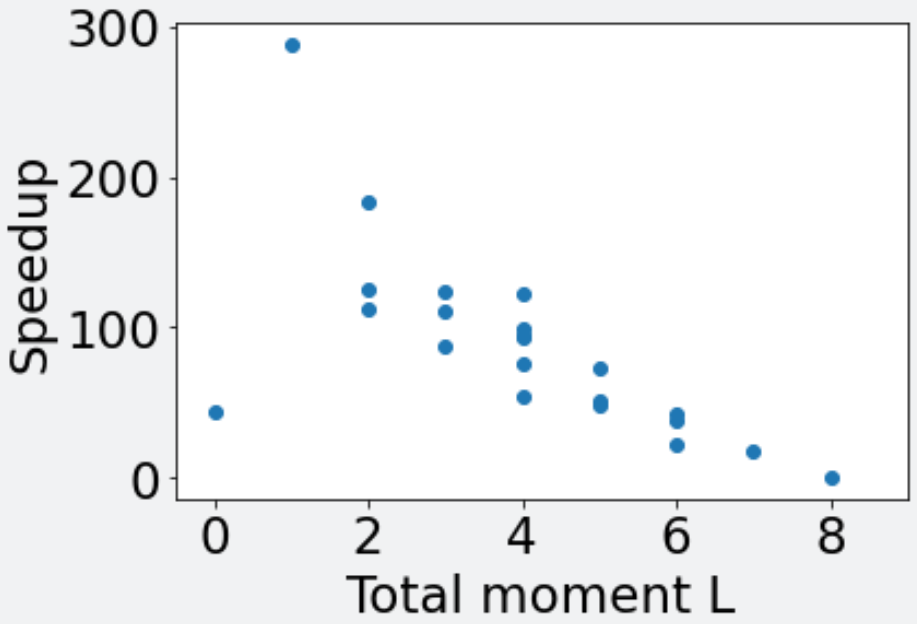
(2) We don't exactly need the ERI themselves. We need $F_{ik} -= D_{jl}\ (ijkl)$
- Doing this contraction on the GPU significantly reduces memory transfer
- Should also makes the code more streamlined

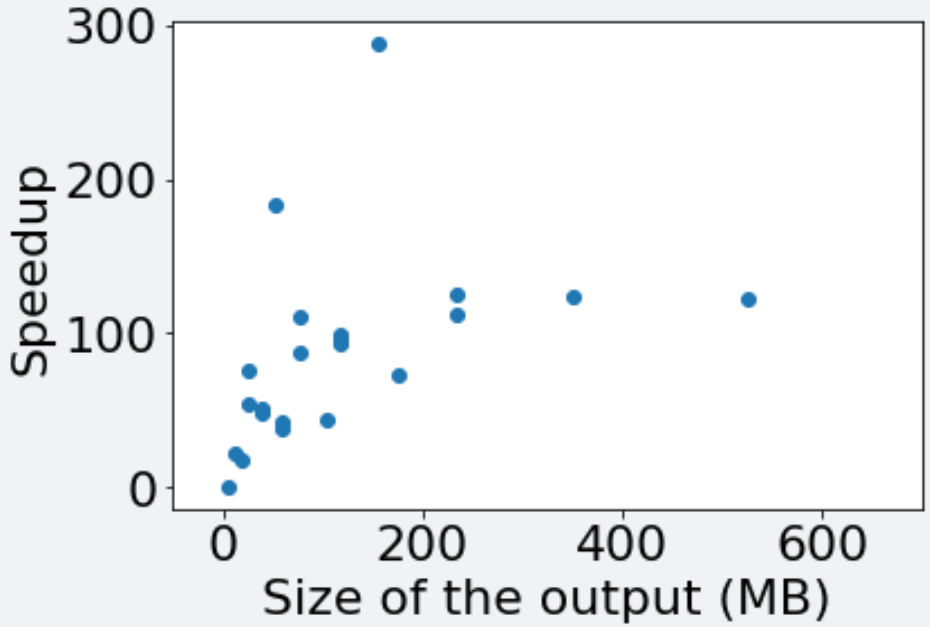
(3) Each code uses its own unique data-structure
- Packing integrals is not trivial (and not very efficient at the moment)
- Contraction with a distributed sparse matrix is *interesting*



Sparsity pattern of F (and D) for an 864 H_2O system before load balance

Some preliminary results



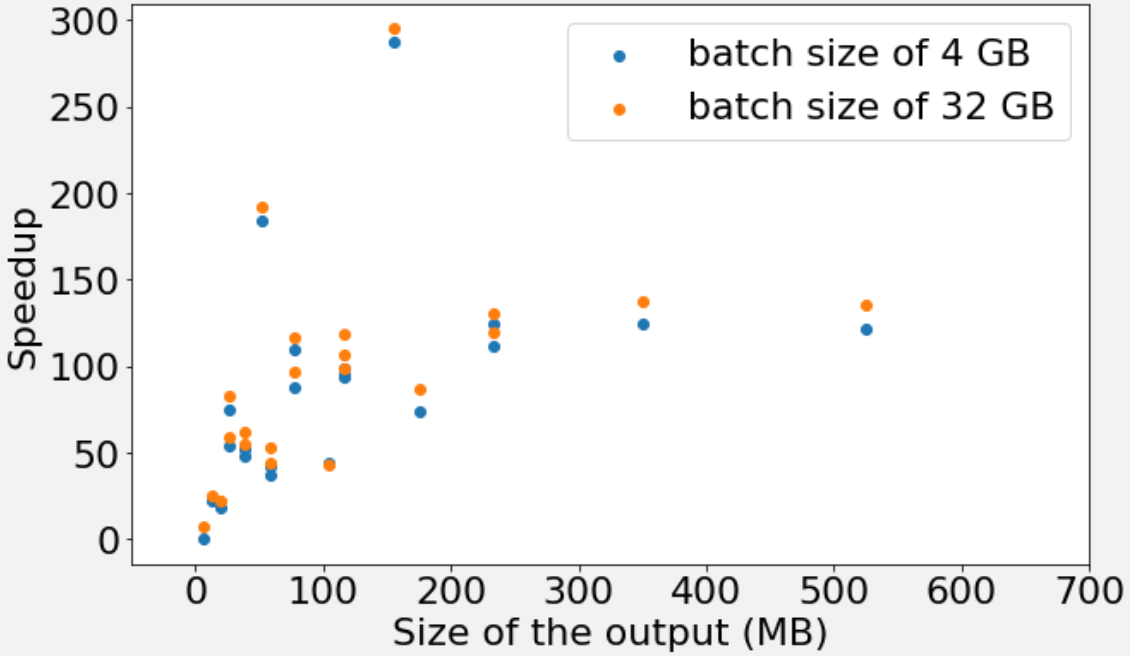


Speedup on A100 against a single core of an intel Xeon 5218.

Speedup appears to increase after L=0, followed by a gradual decline.

Part of this behaviour is due to the limited number of integrals at higher L

Some other preliminary results

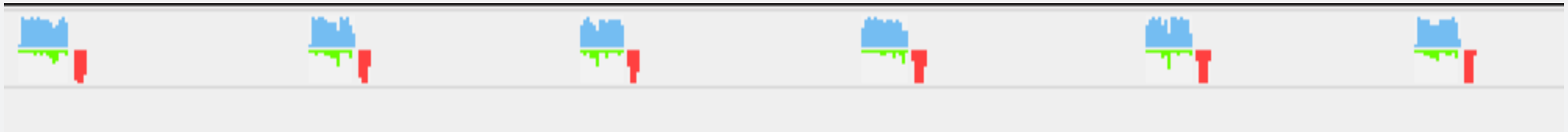


The calculation is done in batches. Once an integral needs too much memory, all integrals are computed

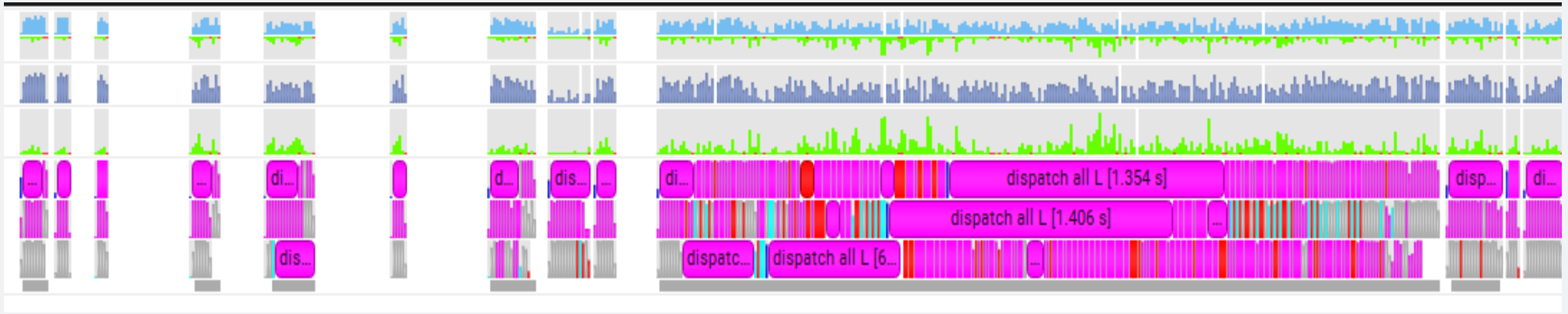
Increasing the max batch size from 4 to 32 GB gives an average speedup of ~ 10%

The dddd case was 14x faster, due to the very small number of integrals

Preparing the batches



Profiler of the batched calculation for 6 H2O. The large gaps between runs are spent preparing the input arrays



Used openMP to parallelize the packaging. Little shared memory -> reduced the batch size
Significant improvement, but far from saturating an A100
N.B. this is for a much larger benchmark with 12 H2O

Current State

Packaging is the limiting factor at the moment.

Different screening, allowing for a simpler packaging may improve time **significantly**

However, this is true in our synthetic benchmarks ran on our code, for a few test cases

Currently working with integration into cp2k, to get some time on a more realistic set

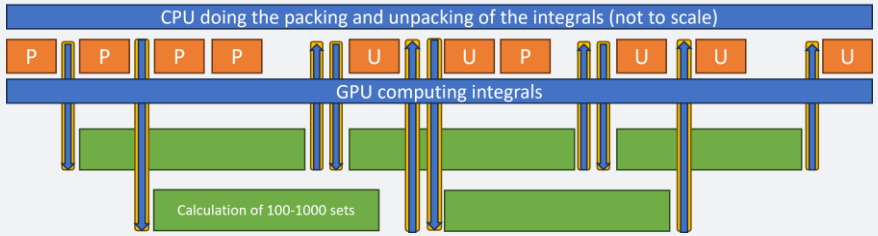
And to look for other limiting factors not visible in out benchmarks

Conclusions

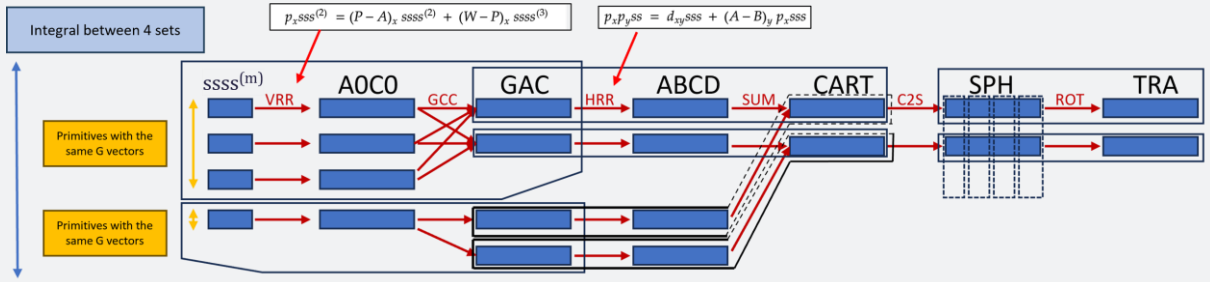
We discuss the use of Graphical Processing Units (GPUs) in local basis set *ab initio* electronic structure codes CP2K and CRYSTAL

We briefly discussed some of the ways GPUs can be used to accelerate these codes

We discussed some of the opportunity and challenges in the calculation of the exact Hartree-Fock Exchange



Scheme for the buffered calculation of the HF exchange



Simplified representation of a ERI calculation on a generalized contracted basis set

UK Research and Innovation

ExCALIBUR

PAX-HPC

PAX-HPC

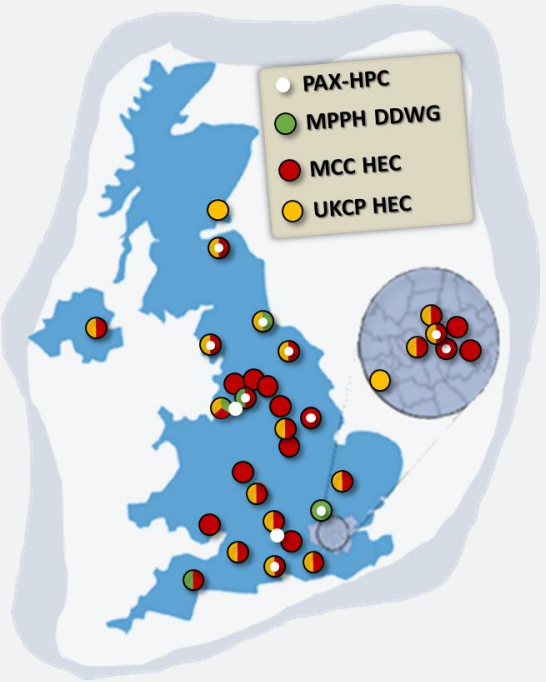
Particles at the eXascale on HPC

LM: Ian Bush

Boss: Gilberto Teobaldi

Alin Marin Elena

Thomas Keal



CCP-QC UKCP CSCS Centro Svizzero di Calcolo Scientifico Swiss National Supercomputing Centre DiRAC nag MATERIALS AND MOLECULAR MODELLING HUB NVIDIA NREL Universiteit Leiden Duke UNIVERSITY MCC intel CCP5

UKRI Engineering and Physical Sciences Research Council

University of York

Durham University

UNIVERSITY OF LINCOLN

Lancaster University

UKRI Science and Technology Facilities Council Scientific Computing

UCL

University of Hertfordshire UH

University of Southampton

King's College LONDON

MANCHESTER 1824 The University of Manchester

epcc

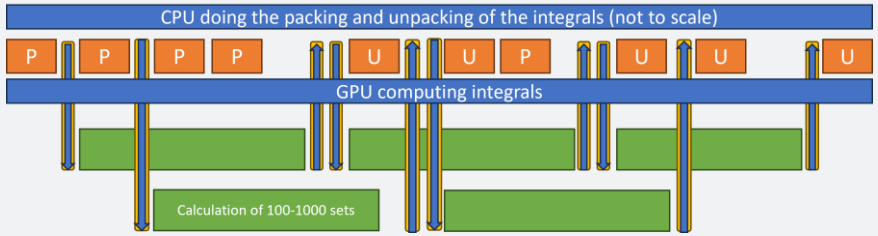
Conclusions

We discuss the use of Graphical Processing Units (GPUs) in local basis set *ab initio* electronic structure codes CP2K and CRYSTAL

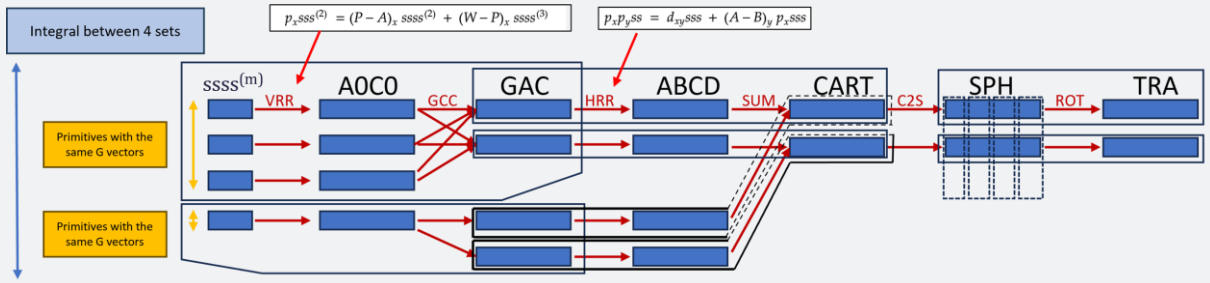
We briefly discussed some of the ways GPUs can be used to accelerate these codes

We presented some results about the diagonalization of the Kohn-Sham / Fock matrix

We discussed some of the opportunity and challenges in the calculation of the exact Hartree-Fock Exchange



Scheme for the buffered calculation of the HF exchange



Simplified representation of a ERI calculation on a generalized contracted basis set

UK Research and Innovation

ExCALIBUR

PAX-HPC

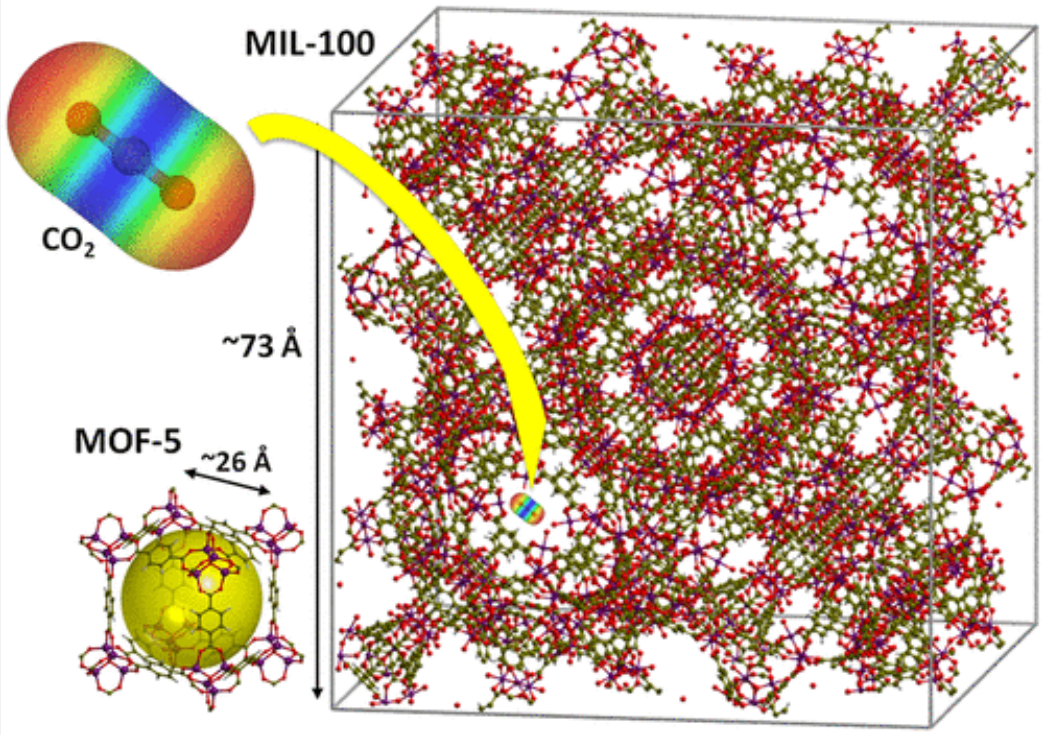
Ab initio Electronic Structure codes

Electronic structure codes compute the properties of the system under study from first principles.

They are used for:

- Modelling and rational design of materials
- Catalysis, batteries, solar cells, pharmaceuticals
- Novel materials.

Our work is focused on CP2K and CRYSTAL



Giant Metal Organic Framework MIL-100 investigated with CRYSTAL for use in carbon sequestration (https://doi.org/10.1021/acs.jpcc.9b06533)

Profiler of the batched calculation for 6 H2O. The large gaps between runs are spent preparing the input arrays