MAX DRIVING THE EXASCALE TRANSITION

Description

CP2K is a quantum chemistry and solid state physics software package that can perform atomistic simulations of solid state, liquid, molecular, periodic, material, crystal, and biological systems. CP2K provides a general framework for different modeling methods such as DFT using the mixed Gaussian and plane waves approaches GPW and GAPW. CP2K can do simulations of molecular dynamics, metadynamics, Monte Carlo, Ehrenfest dynamics, vibrational analysis, core level spectroscopy, energy minimization, and transition state optimization using NEB or dimer method detailed overview of features. CP2K is written in Fortran 2008 and can be run efficiently in parallel using a combination of multi-threading, MPI, and CUDA. It is freely available under the GPL license. It is therefore easy to give the code a try, and to make modifications as needed.
This factsheet presents the CSCS group's CP2K achievements within MAX.

Level of theory

Supported theory levels include DFTB, LDA, GGA, MP2, RPA, semi-empirical methods (AM1, PM3, PM6, RM1, MNDO, ...), and classical force fields (AMBER, CHARMM, ...) features.

Case studies

CP2K provides state-of-the-art methods for efficient and accurate atomistic simulations. Some of the key parts of CP2K are Quickstep, FIST, and QM/MM.

HPC performance

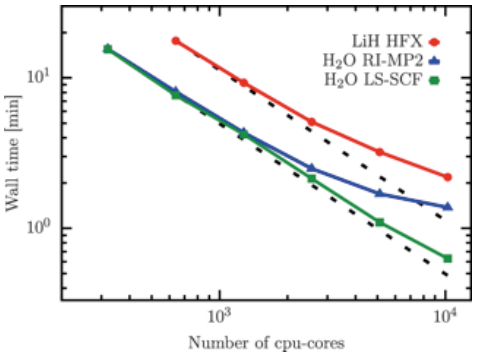
The CP2K benchmark suite provides performance which can guide users towards the best configuration (e.g. machine, number of MPI processors, number of OpenMP threads) for a particular problem. Below, one of such benchmarks:

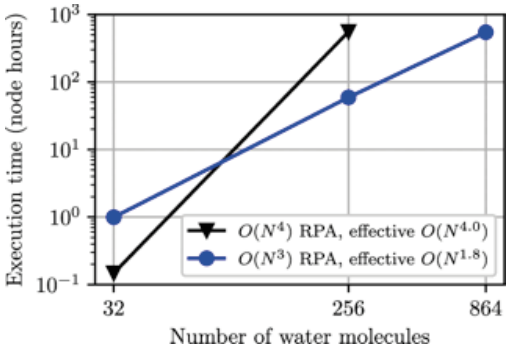
H20-64-RI-MP2

Description
This benchmark is a single-point energy calculation using 2nd order Møller-Plesset perturbation theory (MP2) with the Resolution-of-the-Identity approximation to calculate the exchange-correlation energy. The system consists of 64 water molecules in a 12.4 Å3 cell. This is exactly the same system as used by H20-64 but using a much more accurate model, which is around 100 times more computationally demanding than standard DFT calculations.

Availability
The benchmark is in the CP2K github at:
cp2k/benchmarks/QS_mp2_rpa/64-H20

Results
The best configurations are shown below.





Machine Name	Architecture	Date	Git Commit	Fastest time (s)	Configuration		Detailed results
HECToR	Cray XE6	13/01/2014	82b8204	141.633	49152 cores	8 OMP threads per MPI task	hector-h2o-64-ri-mp2
ARCHER	Cray XC30	09/01/2014	292a983	83.945	36864 cores	4 OMP threads per MPI task	archer-h2o-64-ri-mp2
Magnus	Cray XC40	04/11/2014	27eacee	63.891	24576 cores	6 OMP threads per MPI task	magnus-h2o-64-ri-mp2
Piz Daint	Cray XC30	12/05/2015	f439118	48.15	32768 cores	8 OMP threads per MPI task, no GPU	piz-daint-h2o-64-ri-mp2
Cirrus	SGI ICE XA	24/11/2016	989a92c	303.571	2016 cores	1 OMP thread per MPI task	cirrus-h2o-64-ri-mp2
Noctua	Cray CS500	25/09/2019	9f58d81	82.571	10240 cores	2 OMP thread per MPI task	noctua-h2o-64-ri-mp2

MAX DRIVING THE EXASCALE TRANSITION

COSMA in CP2K

COSMA is a parallel, high-performance, GPU-accelerated, matrix-matrix multiplication algorithm that is communication-optimal for all combinations of matrix dimensions, number of processors and memory sizes, without the need for any parameter tuning. The key idea behind COSMA is to first derive a tight optimal sequential schedule and only then parallelize it, preserving I/O optimality between processes. COSMA is integrated into the CP2K quantum chemistry simulator, through the provided ScaLAPACK API, which makes the integration trivial even when the simulation code is written in Fortran. In the production run, we ran Random-Phase Approximation (RPA) benchmark of 128 water molecules, using the Resolution of Identity (RI). The benchmark was run once on 1024 and once on 128 nodes of the GPU partition on Piz Daint supercomputer (Cray XC50). Computationally, this benchmark consists of 46 tall-and-skinny dense matrix multiplications, with the parameters shown in the upper table.

PDGEMM INFO				
DIMENSIONS			PROCESSOR GRID	
M	N	K	ROWS	COLS
17408	17408	3473408	128	1
BLOCK DIMENSIONS			TRANSPOSE FLAGS	
BLOCK M	BLOCK N	BLOCK K	MATRIX A	MATRIX B
8704	8704	13568	transposed (T)	non-transposed (N)

On **1024 nodes**, the performances of CP2K using COSMA and Cray-libsci_acc (version: 19.10.1), both GPU accelerated, for all dense matrix-matrix multiplications (pdgemm routine) were compared: the version with COSMA was approximately 2x faster (second table).

1024 nodes: Piz Daint Supercomputer (Cray XC50)

	GPU ACCELERATED	
ALGORITHM	CRAY-LIBSCI_ACC	COSMA-GPU
CONFIGURATION	1MPI x 12T	1MPI x 12T
CP2K RPA-RI 128-H20 [s]	317.68	175.12
46 x PDGEMM [s]	139.82	75.26
NODE GFLOP/s	676.37	1256.49
% PEAK PERF.	15.03%	27.92%
NODE TYPE (1024 nodes)	NVIDIA® Tesla® P100 16GB	
NODE PEAK PERF[GFLOP/s]	4500	

~2× faster

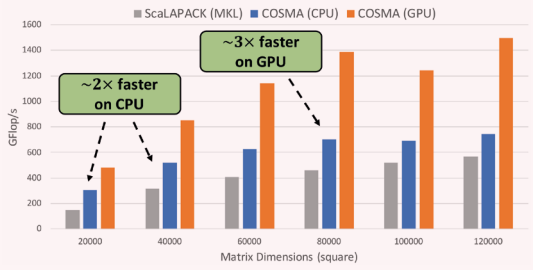
On **128 nodes**, we compared different performances of CP2K with algorithms for multiplying matrices (pdgemm routine): MKL (version: 19.0.1.144), Cray-libsci (version: 19.06.1), Cray-libsci_acc (version: 19.10.1, GPU accelerated) and COSMA (CPU-only and GPU-accelerated versions) libraries. The COSMA performances were the fastest on both CPU and GPU. The CPU version achieved the peak performance, whereas the GPU version achieved more than 65% of the peak performance of GPUs. Though the peak performance of GPUs assumes the data is already residing on GPUs, here matrices were initially residing on CPU. This is why the peak performance is not achieved with the GPU version. Still, this was 25-27% faster than the second best in this case. Results are summarized in the third table.

128 nodes: Piz Daint Supercomputer (Cray XC50)

	CPU-ONLY			GPU ACCELERATED	
ALGORITHM	CRAY-LIBSCI	MKL	COSMA-CPU	CRAY-LIBSCI_ACC	COSMA-GPU
CONFIGURATION	1MPI x 12T	1MPI x 12T	1MPI x 12T	1MPI x 12T	1MPI x 12T
CP2K RPA-RI 128-H20 [s]	6379.14	2305.41	2238.94	865.73	781.60
46 x PDGEMM [s]	5896.45	1836.85	1723.62	338.47	257.99
NODE GFLOP/s	128.30	411.87	438.92	2235.19	2932.44
% PEAK PERF.	25.70%	82.51%	87.92%	49.67%	65.17%
NODE TYPE (128 nodes)	Intel® Xeon® E5-2690 v3 @ 2.60GHz (12 cores, 64GB RAM)			NVIDIA® Tesla® P100 16GB	
NODE PEAK PERF[GFLOP/s]	499.2			4500	
	This is only using CPU nodes on the GPU partition of Piz Daint. However, CPU node peak perf is much higher on the CPU partition.			Max peak assumes the data is already on GPU, which explains why it is not fully achieved.	

~10% faster ~25% faster

With COSMA, even higher speedups are possible, depending on matrix shapes. To illustrate possible performance gains, we also ran different square matrix multiplications on the same number of nodes (=128) of Piz Daint supercomputer. The block size is 128x128 and the processor grid is also square: 16x16 (2 ranks per node). The performance of COSMA is compared against Intel MKL ScaLAPACK (version: 19.0.1.144). The results on Cray XC50 (GPU-accelerated) and Cray XC40 (CPU-only) are shown in the last table.



All the results from this section assumed matrices given in (block-cyclic) ScaLAPACK data layout. However, if the native COSMA layout is used, even higher throughput is possible.

Support

Reference manual

User Forum on Google groups

HOWTOs

Benchmark suite

References

- *T. D. Kühne et al., CP2K: An electronic structure and molecular dynamics software package - Quickstep: Efficient and accurate electronic structure calculations, J. Chem. Phys.* ***152****, 194103 (2020).*