OpenMP®

SC23 Booth Talk Series

Experiences in offloading GAMESS with OpenMP

Tosaporn Sattasathuchana, Peng Xu,
Ames National Lab

GAMESS ECP Team

Team member

Ames National Lab
- Mark Gordon (PI)
- Melisa Alkan
- Daniel Del Angel
- Dipayan Datta
- Taylor Harville
- Buu Pham
- Bryce Westheimer
- Peng Xu
- Federico Zahariev
- Tosaporn Sattasathuchana

Old Dominion University
- Masha Sosonkina (co-PI)
- Viabhav Sundriyal

EP Analytics
- Laura Carrington (co-PI)
- Ananta Tiwari
- Sarom Leang

Australian National U
- Giuseppe Barca (co-PI)
- Jorge Galvez

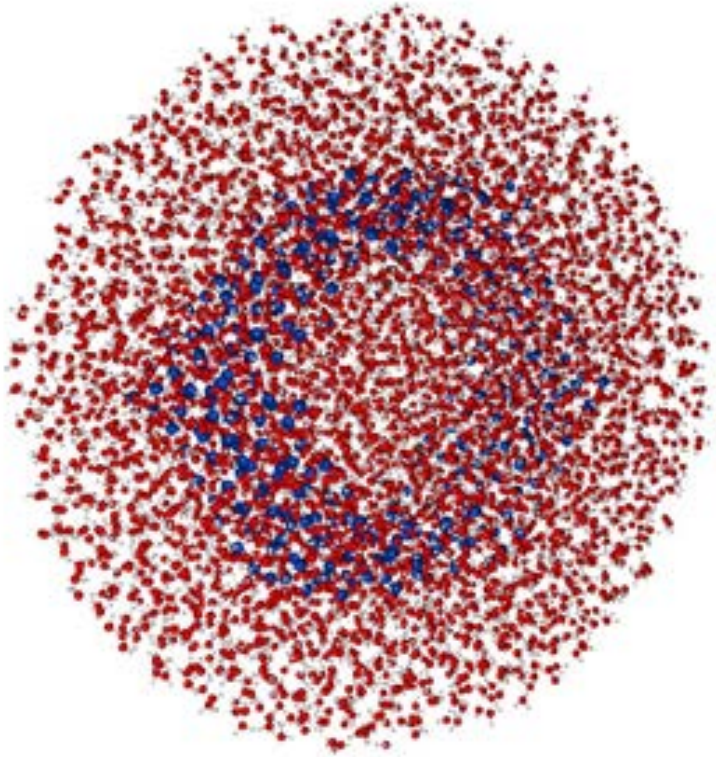
Georgia Tech
- David Sherril (co-PI)
- *David Poole*

U Texas, El Paso
- Shirley Moore (co-PI)
- Henry Moncada

Acknowledgement
- Argonne: Colleen Bertoni
- Brookhaven: Dossay Oryspayev
- Lawrence Berkeley: Jack Deslippe
- Oak Ridge: Dmytro Bykov
- AMD: Brian Cornille, Bob Robey
- HPE Cray: Luke Roskop, Marcus Wagner
- Intel: Brian Whitney, Xinmin Tian, Ravi Narayanaswamy
- HPCToolkit: John Mellor-Crummey, Wileam Phan, Laksono Adhianto

General Atomic and Molecular Electronic Structure System (GAMESS)

- General purpose electronic structure code
- Focus on *ab initio* quantum chemistry
- GAMESS ECP enables computation of large systems (> 10K atoms) using high-accuracy QM methods.
- Languages: Fortran + C library
- Programming models: CUDA/OpenMP-target



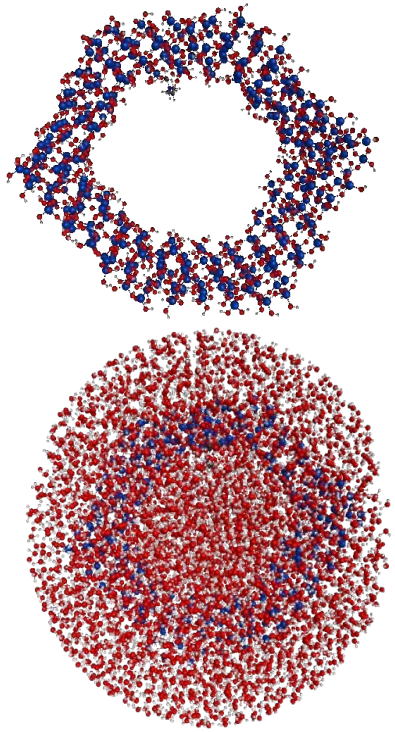
Introduction

Traditional *Ab initio* methods are expensive.

- HF & DFT ~1,000 atoms
- MP2 ~100 atoms
- CCSD(T) ~10-20 atoms

To achieve high chemical accuracy, these methods limit to small molecular systems.

Challenging to solve complex systems!



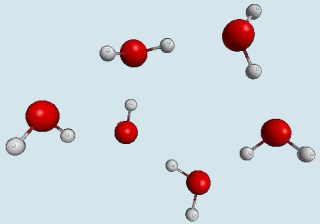
15,000++ atoms

Fragmentation methods

To enable direct connection with complex experiments with high accuracy

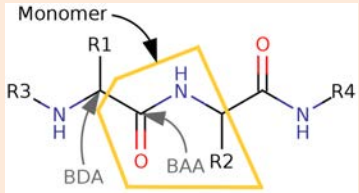
Effective fragment potential (EFP)

- A QM-based force field
- Designed to model solvent effects
- Noncovalent interactions
- Rigid model



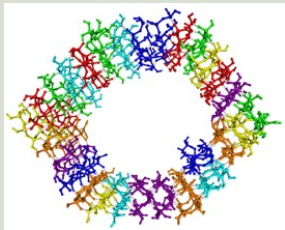
Fragment molecular orbital (FMO)

- Break system into smaller pieces
- Ab initio method based on many-body expansion
- Flexible fragments

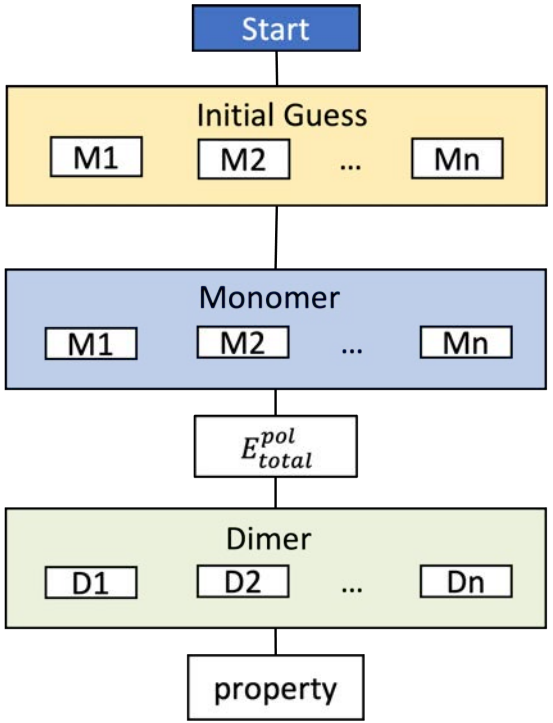


Effective fragment molecular orbital (EFMO)

- Merger of EFP+FMO
- Simplifies FMO and improves accuracy
- Ab initio method
- Flexible fragments



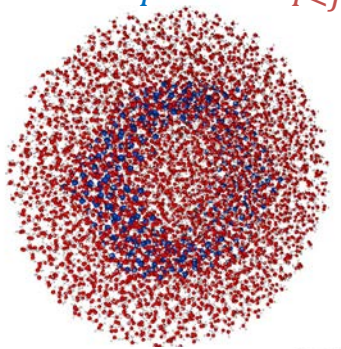
EFMO for heterogeneous catalysis

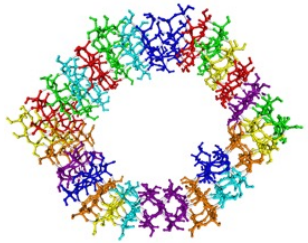


- QM level
  - Restricted Hartree Fock (RHF)
  - Resolution-of-the-identity MP2 (RI-MP2)
  - RI coupled cluster (RI-CC)
  - Density functional theory (DFT)
- MAKEFP → prepare EFP parameters
  - Couple-perturbed Hartree Fock (CPHF)
  - Time-dependent Hartree Fock (TDHF)

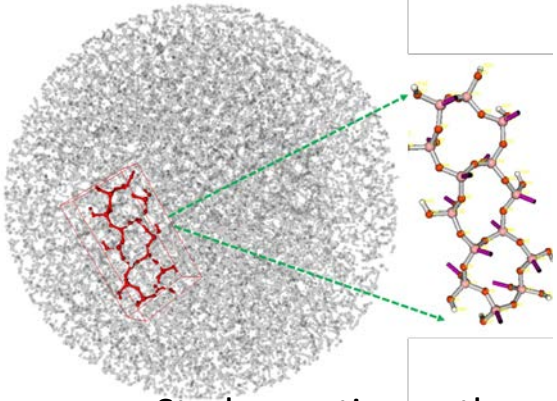
- QM level (RHF, RI-MP2, RI-CC, DFT)
- EFP-EFP (an ab initio force field)

$$E \cong \sum_{I}^{n_F} E_I + \sum_{I<J}^{n_F} \Delta E_{IJ}$$



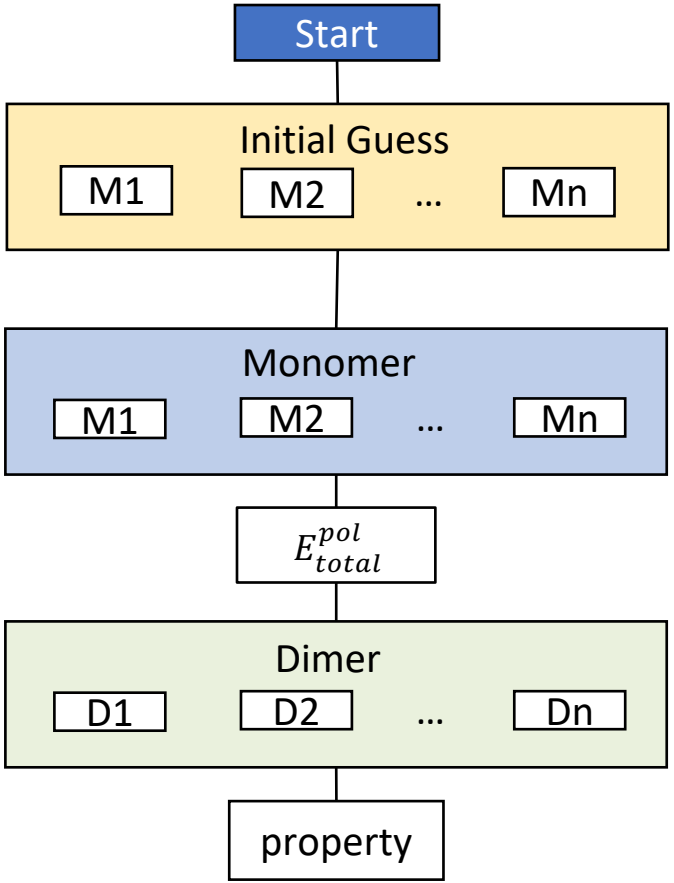


Divide into fragments



Study reaction pathway

EFMO for heterogeneous catalysis



- QM level
  - Restricted Hartree Fock (RHF)
  - Resolution-of-the-identity MP2 (RI-MP2)
  - RI coupled cluster (RI-CC)
  - Density functional theory (DFT)
- MAKEFP → prepare EFP parameters
  - Couple-perturbed Hartree Fock (CPHF)
  - Time-dependent Hartree Fock (TDHF)

- QM level (RHF, RI-MP2, RI-CC, DFT)
- EFP-EFP (an ab initio force field)

Porting QM codes to GPU

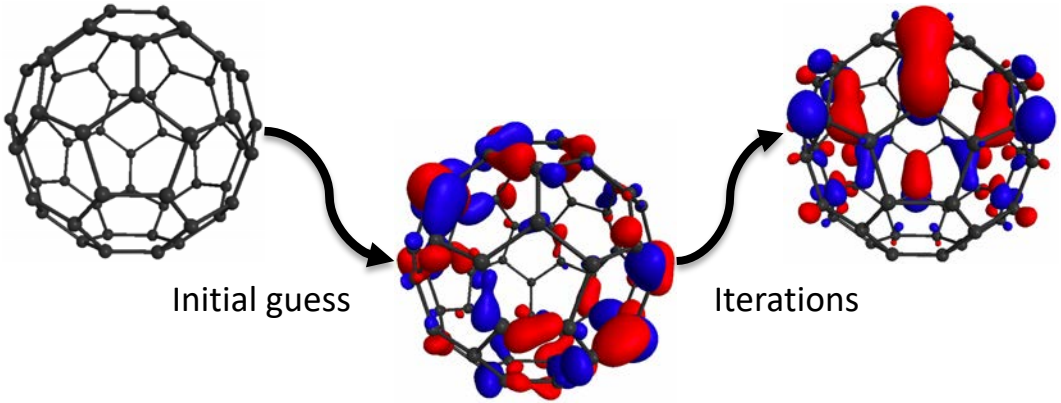
Improve time to solution

Initial Molecular Orbital Guess

Led by Daniel Del Angel

Initial Molecular Orbital Guess

- Electronic structure described in terms of Molecular Orbitals (MOs) – optimized iteratively
- A starting set of MOs (the initial guess)
- Matrix operations accelerated with GPU libraries (e.g., cuBLAS, cuSOLVER)

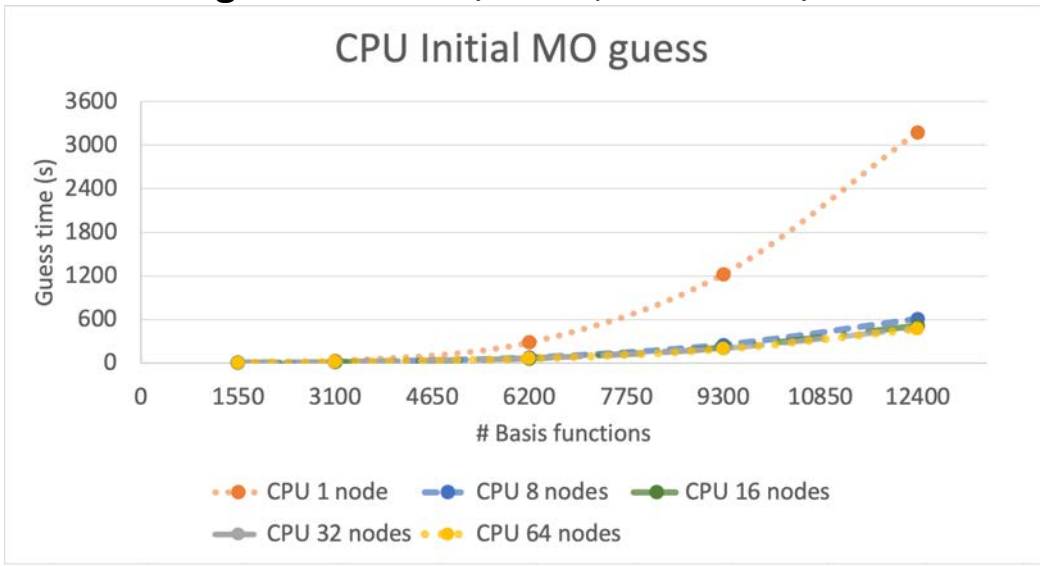



```
module offloaded_guess_mod
…
subroutine offloaded_DGEMM(matrices)
 use iso_c_binding,omp_lib
 …
 interface to cuBLAS
 …
 !$omp target data map (tofrom: matrices)
     call  cuBLAS_DGEMM(matrices)
 !$omp end target data
```

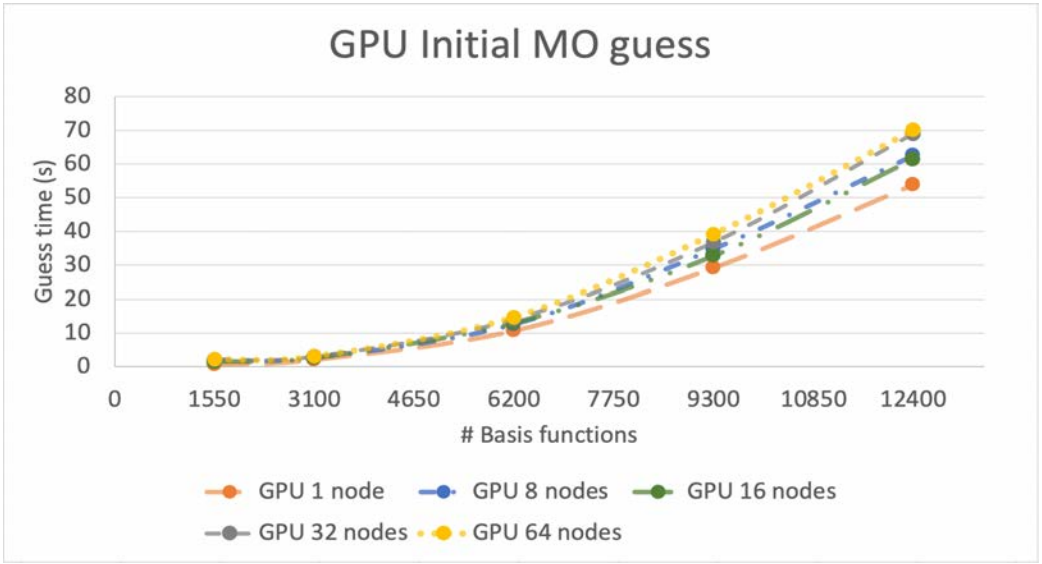
```
subroutine MO_GUESS
 use offloaded_guess_mod
   …
   if (offloading_enabled) then
       call offloaded_DGEMM(matrices)
   else
       call DGEMM(matrices)
   end if
   …
```

Initial guess offloading results

Using 8 MPI ranks/node, 8 threads/rank



Using 4 MPI ranks/node, 16 threads/rank



- Calculations performed on Perlmutter with 128 logical cores per node
- One 1 node, the speedup of the GPU code is 59x
- On 64 nodes, the speedup is 6.7x

Less speedup when increasing the number of nodes:

- Communication overhead
- The work is not distributed across GPUs.

Offloading HF/MAKEFP (CPHF and TDHF)

two—electron integral (TEI)-related codes

Led by Tosaporn Sattasathuchana, Peng Xu, Buu Pham, & Melisa Alkan

Offloading HF/MAKEFP (CPHF and TDHF)

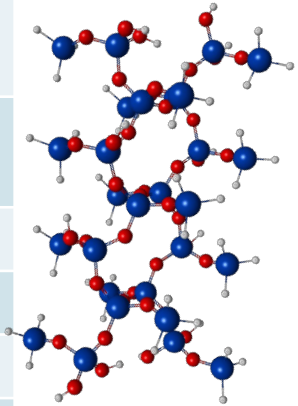
- CPHF and TDHF
  - Common to HF and MAKEFP
- Bottleneck
  - Evaluation of two-electron integrals
    - Some integral subroutines contain >1,000 lines of code.
  - Contraction with density matrix (DA) to form Fock matrix (FA)
- Modernize and restructure Fortran77 code
  - Remove common blocks
  - Remove large intermediate arrays

```
!$omp target distribute parallel do &
!$omp shared(FA,DA) private(ERIs) &
!$omp reduction(+:nintn)
do i,j,k,l
     call integral_j01(i,j,k,l,ERIs)
     call dirfck_rhf(i,j,k,l,ERIs,nintn,DA,FA)
```

```
subroutine dirfck_rhf(i,j,k,l,ERIs,nintn,DA,FA)
!$omp atomic update
     FA(IJ) = FA(IJ)+4*ERIs(ijkl)*DA(KL)
!$omp atomic update
     FA(KL) = FA(KL)+4*ERIs(ijkl)*DA(IJ)
!$omp atomic update
     FA(IK) = FA(IK)-ERIs(ijkl)*DA(JL)
!$omp atomic update
     FA(JL) = FA(JL)-ERIs(ijkl)*DA(IK)
!$omp atomic update
     FA(IL) = FA(IL)-ERIs(ijkl)*DA(JK)
!$omp atomic update
     FA(JK) = FA(JK)-ERIs(ijkl)*DA(IL)
```

Offloading Results (HF/MAKEFP)

Using 1GPU	Time from miniERIs (an iteration of RHF method) (s)						
	Summit (V100)		Perlmutter (A100)		Frontier (MI250X)		Sunspot (PVC)
Kernel	xlf/ 16.1.1-10	nvhpc/ 22.5	nvhpc/ 22.7	cce/ 15.0.1	amd/ 5.4.3	cce/16.0.0 rocm5.5.1	oneAPI
j01	0.26	0.03	0.02	0.01	0.02	0.01	0.01
j02	1.53	0.16	0.09	0.07	0.24	0.09	0.06
j03	1.98	0.15	0.08	0.2	0.72	0.16	0.12
j04	3.88	0.24	0.15	1.98	0.85	4.74	0.35
j05	16.96	1.09	0.54	5.95	15.8	20.88	1.98
j06	79.33	8.92	4.52	5.77	22.44	19.32	3.00
Sum	103.94	10.59	5.4	13.98	40.07	45.2	5.52



Portability of Offloaded Code

Accelerator	V100	A100	MI250X	PVC
Compiler	NVHPC	NVHPC CCE	CCE	oneAPI
Center	OLCF Summit	ALCF Polaris	OLCF Frontier	ALCF Aurora

Offloading RI-MP2

Led by Buu Pham

RI-MP2: Using Fortran Interfaces to Accelerated Libraries

Main bottleneck of RI-MP2
- the tensor contraction.

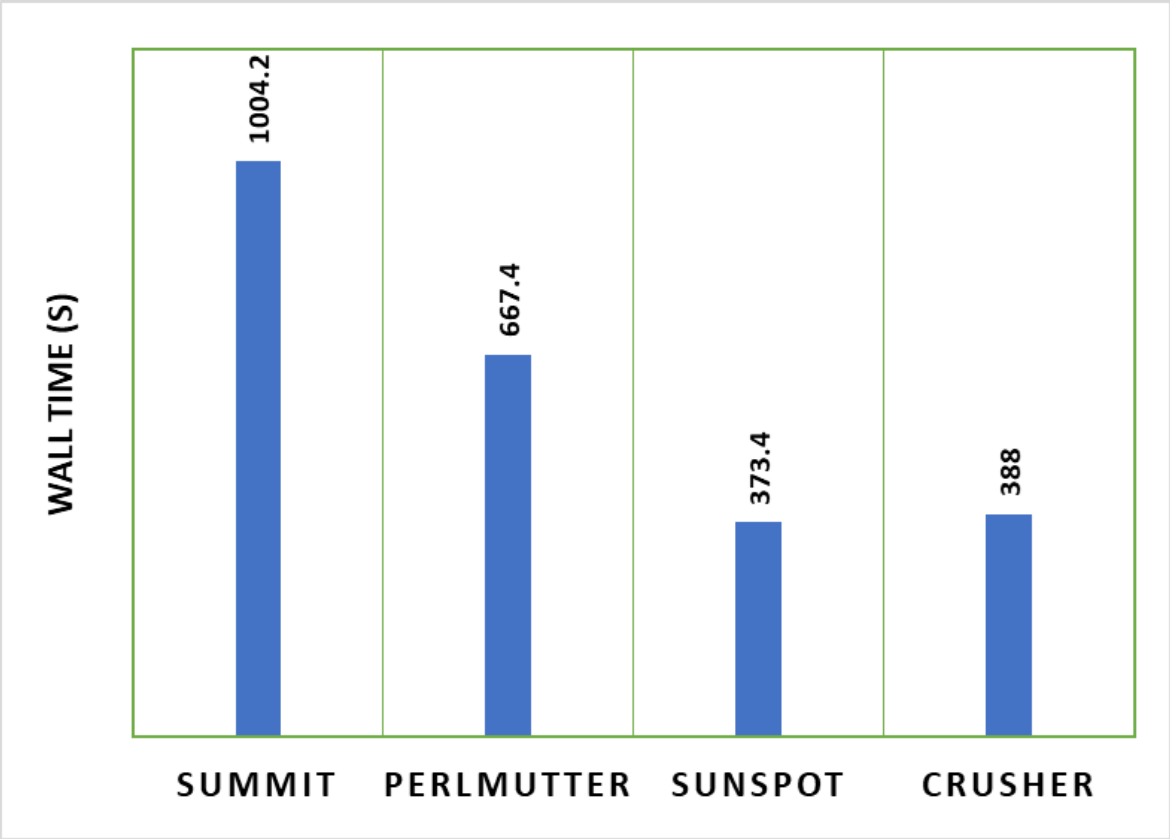
$$(ia|jb) \approx \sum_{P}^{X} B_{ia}^{P} B_{jb}^{P}$$

```
module cublasf
    use, intrinsic :: iso_c_binding
    type(c_ptr)    :: cublas_handle
    enum, bind(c)
       enumerator :: CUBLAS_OP_N = 0
       enumerator :: CUBLAS_OP_T = 1
       enumerator :: CUBLAS_OP_C = 2
    end enum
    integer(c_int) function cublasXtDgemm &
                    (handle,transa,transb,       &
                     m,n,k,alpha,dA,ldda,dB,lddb,&
                     beta,dC,lddc)       &
                     bind(c, name="cublasXtDgemm")
    type(c_ptr), value :: handle
    integer(c_int), value :: transa
    integer(c_int), value :: transb
    integer(c_int), value :: m
    integer(c_int), value :: n
    integer(c_int), value :: k
    real(c_double) :: alpha
    real(c_double),dimension(*) :: dA
    integer(c_int), value :: ldda
    real(c_double),dimension(*) :: dB
    integer(c_int), value :: lddb
    real(c_double) :: beta
    real(c_double),dimension(*) :: dC
    integer(c_int), value :: lddc
    end function cublasXtDgemm
end module
```

```
!$omp target data map(alloc:QVV)
!Loop over all slices of BI, BJ
!$omp target data map(to:BI,BJ)
!$omp target data use_device_ptr(BI,BJ,QVV)
   cublas_return =  cublasXtDgemm  &
      (cublas_handle,CUBLAS_OP_T,CUBLAS_OP_N, &
       NVIR*iQVV,NVIR,NAUXBASD,               &
       one,BI,NAUXBASD,                       &
           BJ,NAUXBASD,                       &
       zer,QVV,NVIR*iQVV)
   cublas_return = cudaDeviceSynchronize()
!$omp end target data
!$omp end target data
!$omp end target data
```

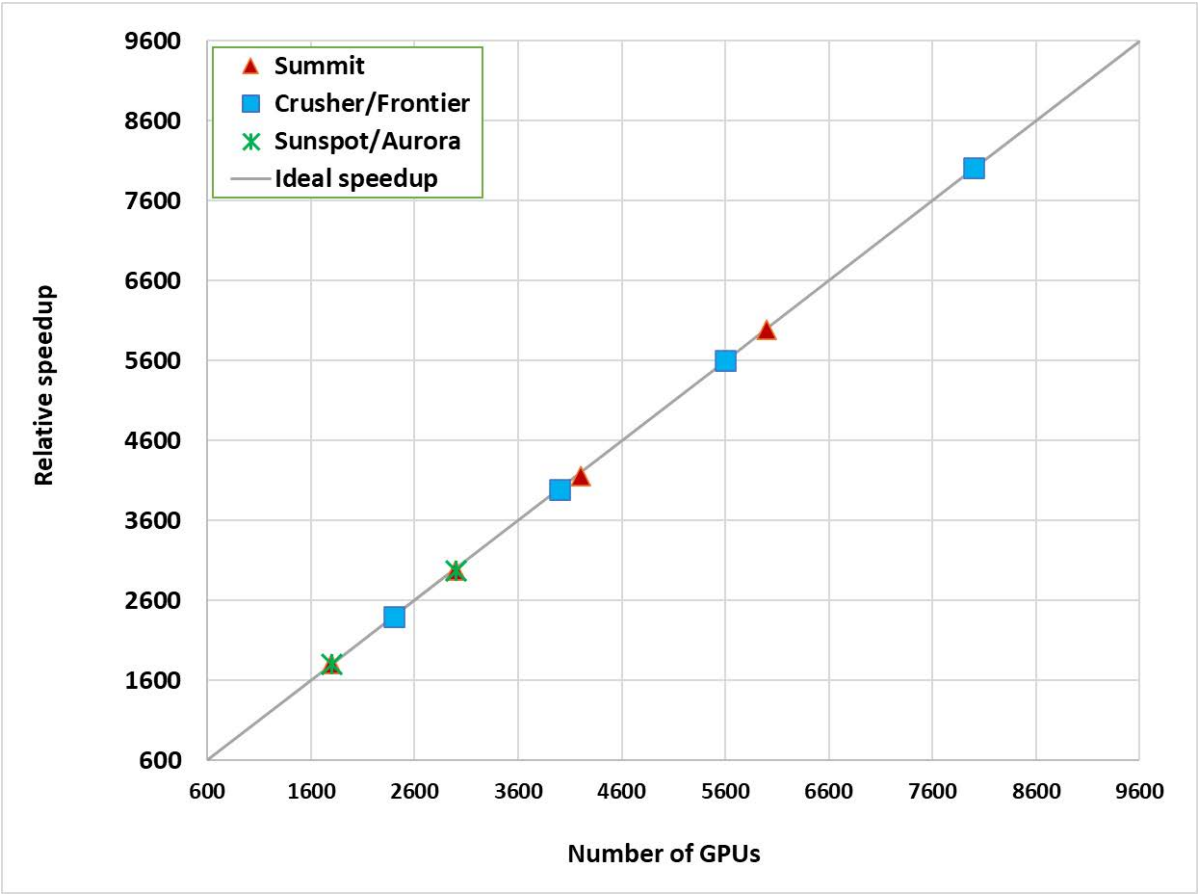
```
!$omp target data map(alloc:QVV)
!Loop over all slices of BI, BJ
!$omp target data map(to:BI, BJ)
!$omp target data use_device_ptr(BI,BJ,QVV)
call hipblasCheck(hipblasDgemm(
    HIPBLAS_handle,    & ! type(c_ptr), value
    HIPBLAS_OP_T,      & ! integer(kind(hipblas_op_n)), value
    HIPBLAS_OP_N,      & ! integer(kind(hipblas_op_n)), value
    NVIR*iQVV,         & ! integer(c_int), value
    NVIR,              & ! integer(c_int), value
    NAUXBASD,          & ! integer(c_int), value
    one,               & ! real(c_double)
    c_loc(BI(1)),      & ! type(c_ptr), value
    NAUXBASD,          & ! integer(c_int), value
    c_loc(BJ(1)),      & ! type(c_ptr), value
    NAUXBASD,          & ! integer(c_int), value
    zero,              & ! real(c_double)
    c_loc(QVV(1,1,1)), & ! type(c_ptr), value
    NVIR*iQVV          & ! integer(c_int), value
))
call hipCheck(hipDeviceSynchronize())
!$omp end target data
!$omp end target data
!$omp end target data
```

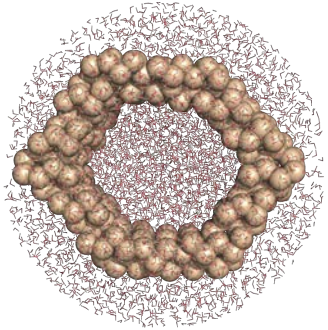
Relative Timing Across Different GPUs

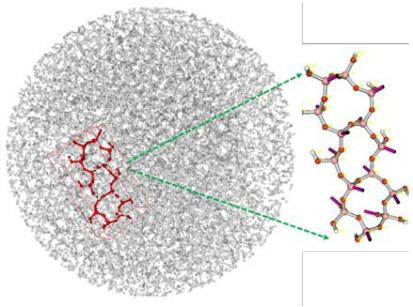


- $(H_2O)_{139}$
- 6-31G(d,p)
- Computed on one node
  - Summit: 6 V100s
  - Perlmutter: 4 A100s
  - Sunspot: 6 Intel GPUs
  - Crusher: 4 MI250Xs (8 GCDs)

EFMO/RIMP2 Strong Scaling







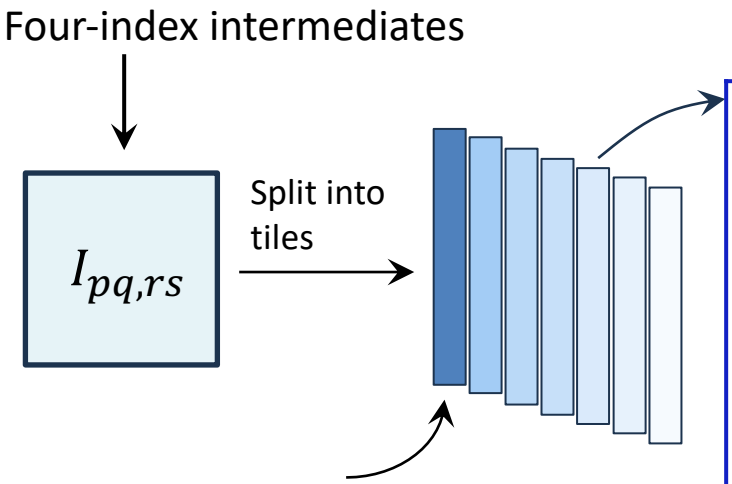
- $MSN@(H_2O)_{4597}$
- 6-31G/cc-pVDZ
- 74,277 basis functions
- Rcut=1.0
- 9,556 dimers
- 13,791 atoms

Offloading RI-CCSD(T)

Led by Dipayan Datta

The Generic Offloading Algorithm: Matrix Tiling to Reduce GPU Memory Allocation in RI-CCSD(T)

- **Density fitting approximation**: Allows factorizing CC equations in terms of the partially transformed three-index intermediates, e.g., $\mathcal{W}_{ij}^{P} = \sum_e (ie|P) t_e^j$.
- **Integral-direct algorithm**: 4-index intermediates are assembled on the fly.




```
!$omp target data use_device_ptr(I,T2,W)

#ifdef USE_CUBLAS
     cuBlas_status = cublasDGEMM(cublas_handle,
                     cublas_OP_N,cublas_OP_N,M,
                     N,K,1.0,c_loc(I),LDA,
                     c_loc(T2),LDB,0.0,c_loc(W),
                     LDC)
#elif USE_HIPBLAS
     call hipblasCheck(hipblasDGEMM(hipblas_handle,
                       hipblas_OP_N,hipblas_OP_N,M
                       N,K,1.0,c_loc(I),LDA,
                       c_loc(T2),LDB,0.0,c_loc(W),
                       LDC)
     call hipCheck(hipDeviceSynchronize())
#endif

!$omp end target data
```

$$^{a}t_{jb,i} \leftarrow t_{ab,ij}$$

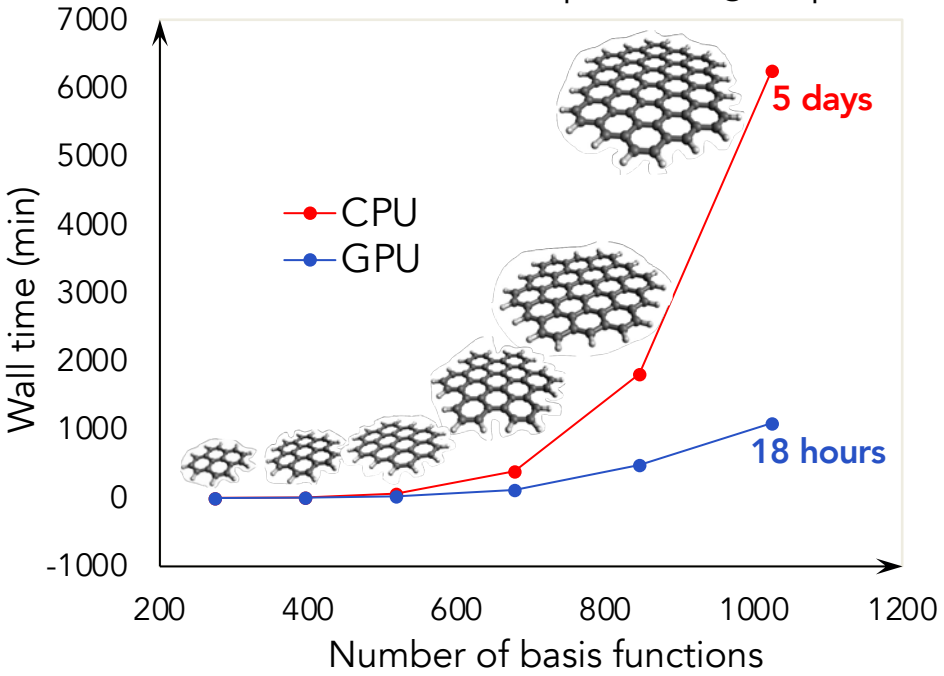
Sort cluster amplitudes on the CPU. Copy at most N^3 memory-scaling segments to the GPU.

DOI: 10.1021/acs.jctc.3c00876

Performance of the Offloaded Triples Correction in RI-CCSD(T)

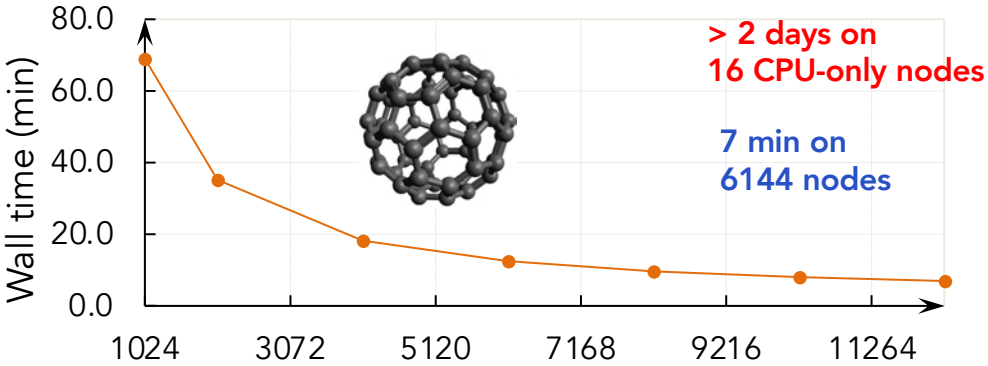
Perlmutter

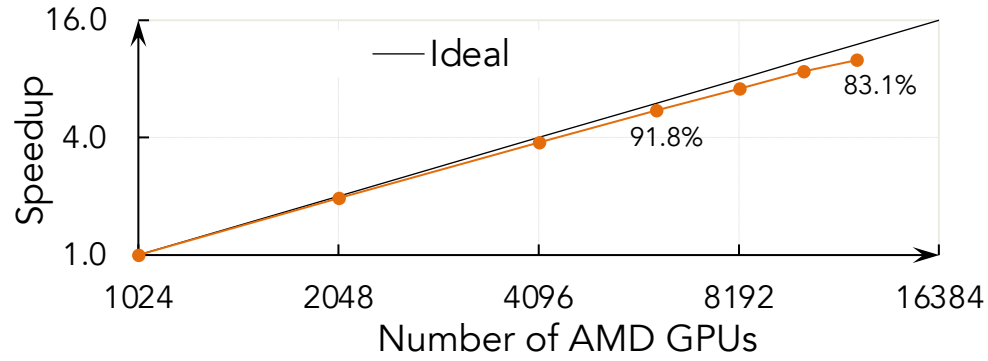
On 128 nodes
Basis: cc-pVDZ/aug-cc-pVTZ-RI



FRONTIER

C_{60}
Basis: cc-pVDZ/aug-cc-pVTZ-RI





Portability of the Offloaded RI-CCSD(T) Code

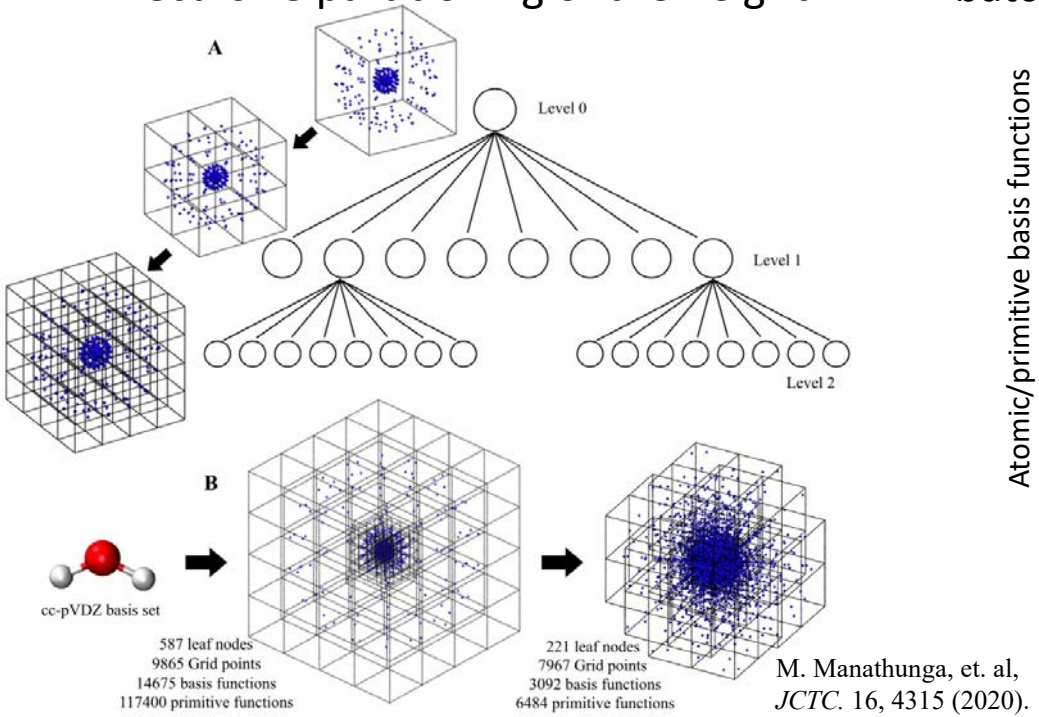
Accelerator	NVIDIA Tesla V100	NVIDIA A100	AMD Instinct MI250X
Compiler	NVHPC	NVHPC	CCE
Center	OLCF Summit	ALCF Polaris NERSC Perlmutter	OLCF Frontier/Crusher

Offloading DFT

Led by Federico Zahariev

Offload of the exchange-correlation (XC) part of the Fock matrix

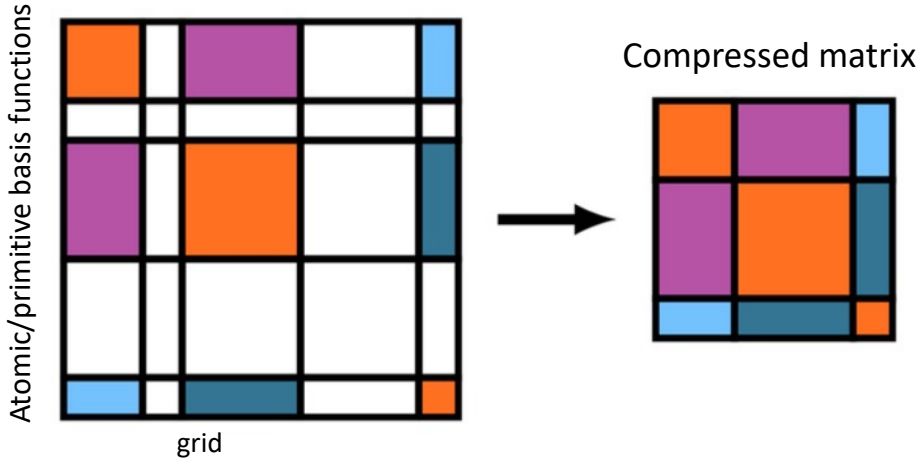
Recursive partitioning of the XC grid:



M. Manathunga, et. al, *JCTC.* 16, 4315 (2020).

Data preparation for DGEMM and DSYR2K with OpenMP offload to minimize the CPU ↔ GPU transfers

batch compression of matrices due to the sparse grid:



D. B. Williams-Young, et. al, *Front. Chem.* 8, 581058 (2020).

Batched (with variable dimensions) versions of DGEMM and DSYR2K (Magma library with HIP)

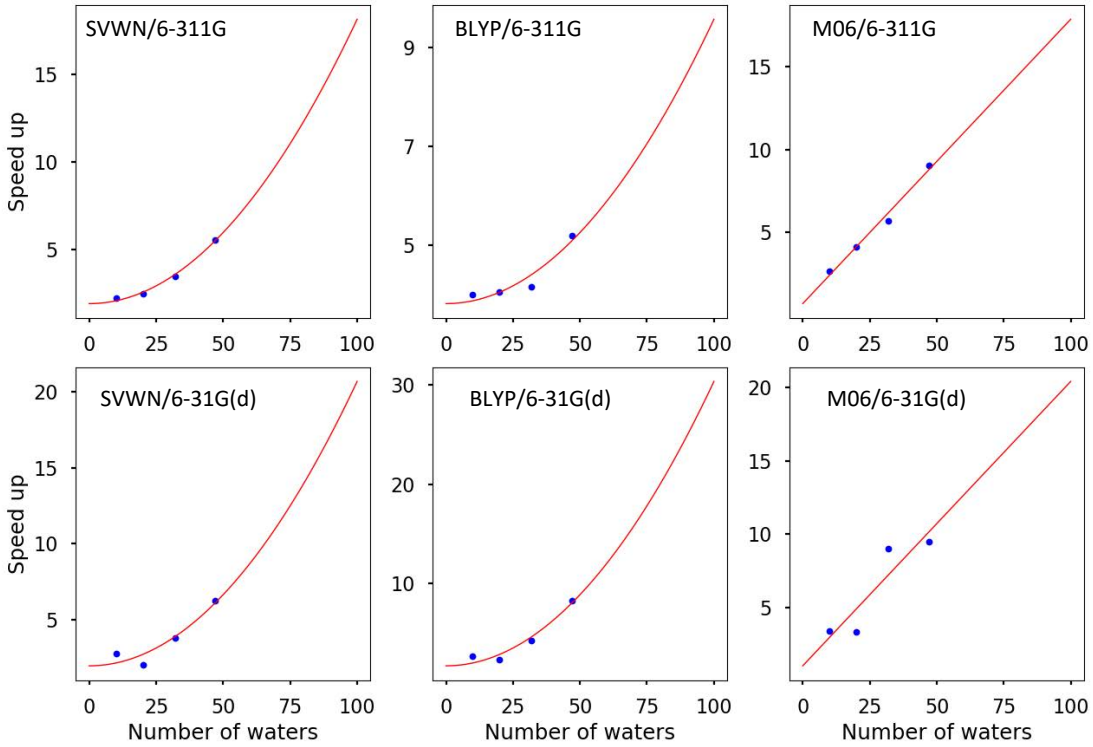
Offload of the exchange-correlation (XC) of the Fock matrix (results)

Speed-up versus the number of water molecules

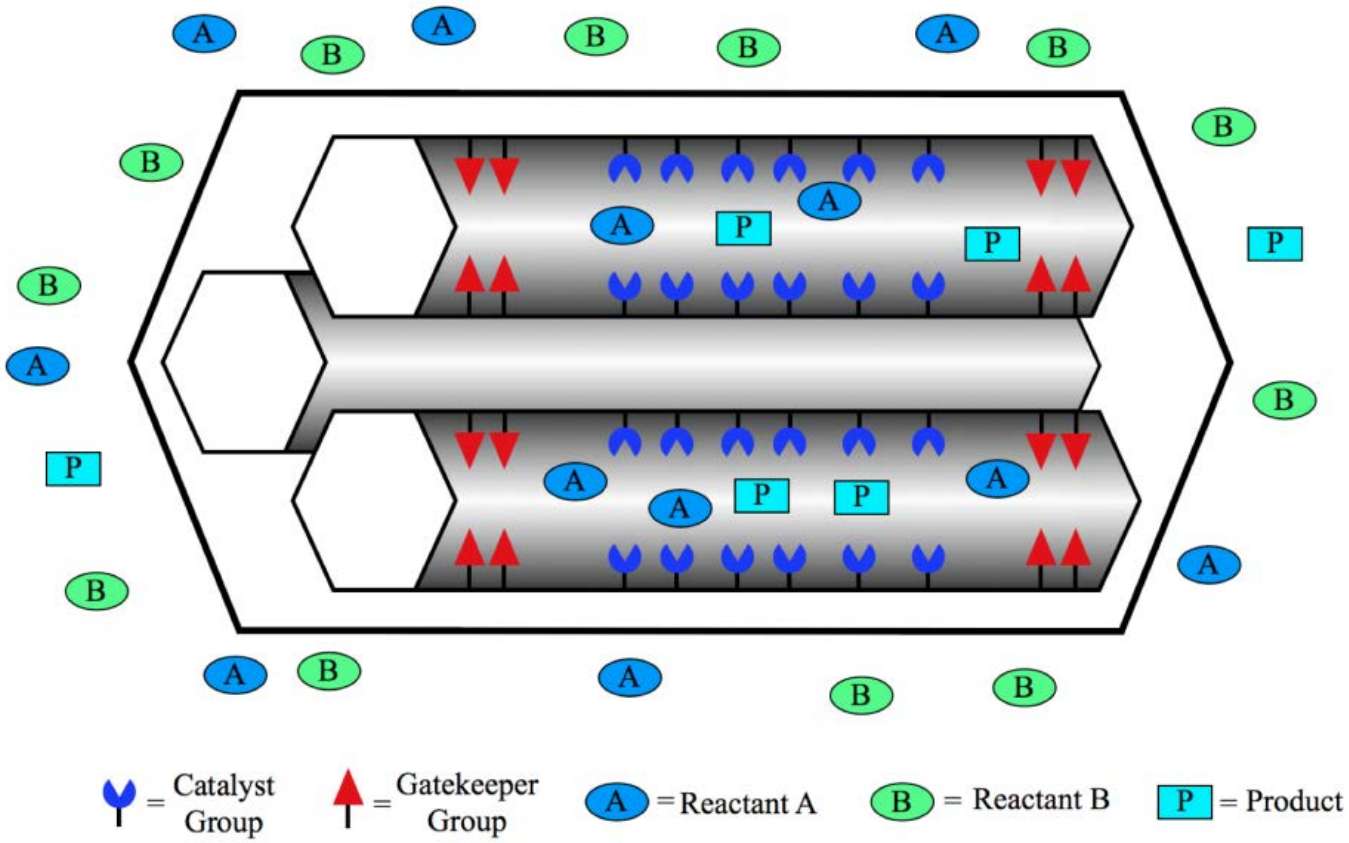
"speed up": ratio of OpenMP and GPU offload for the average times of the XC part of a SCF iteration on one Summit node

Speed Up	10 waters	20 waters	32 waters	47 waters
SVWN/6-311G	2.81	2.03	3.82	6.24
BLYP/6-311G	2.64	2.27	4.23	8.25
M06/6-311G	3.41	3.34	9.05	9.50
SVWN/6-31G(d)	2.19	2.47	3.43	5.55
BLYP/6-31G(d)	4.01	4.07	4.17	5.20
M06/6-31G(d)	2.62	4.10	5.69	9.06

Extrapolations of the speed-ups



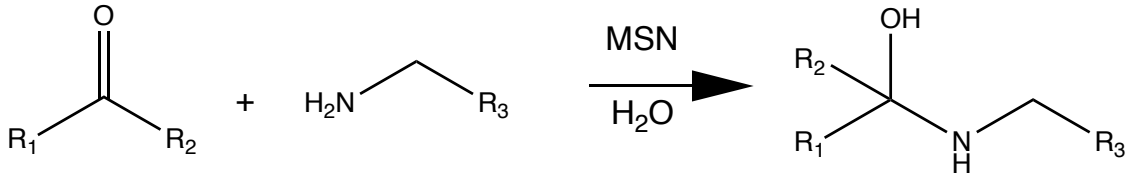
EFMO for Heterogeneous Catalysis



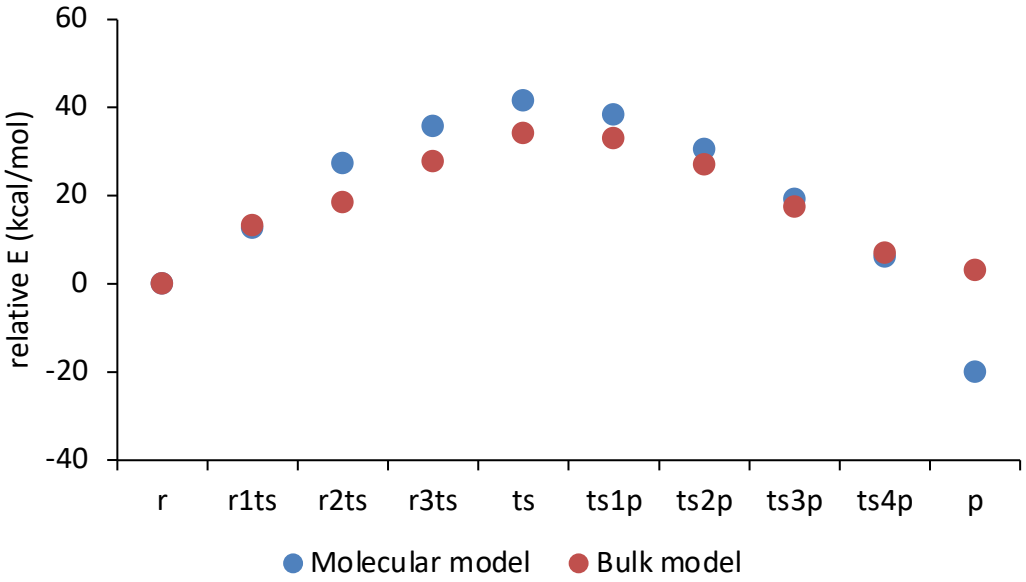
Reaction on mesoporous silica nanoparticles (MSNs)

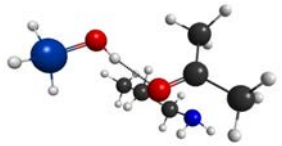
Gatekeeper groups selectively allow reactants “A” (not “B”) to enter and form product “P”

EFMO MSN Reaction Path

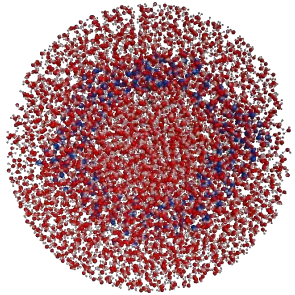


Reaction Pathway:
EFMO/RI-MP2/6-31G/cc-pVDZ-ri using Rcut=1





Molecular model

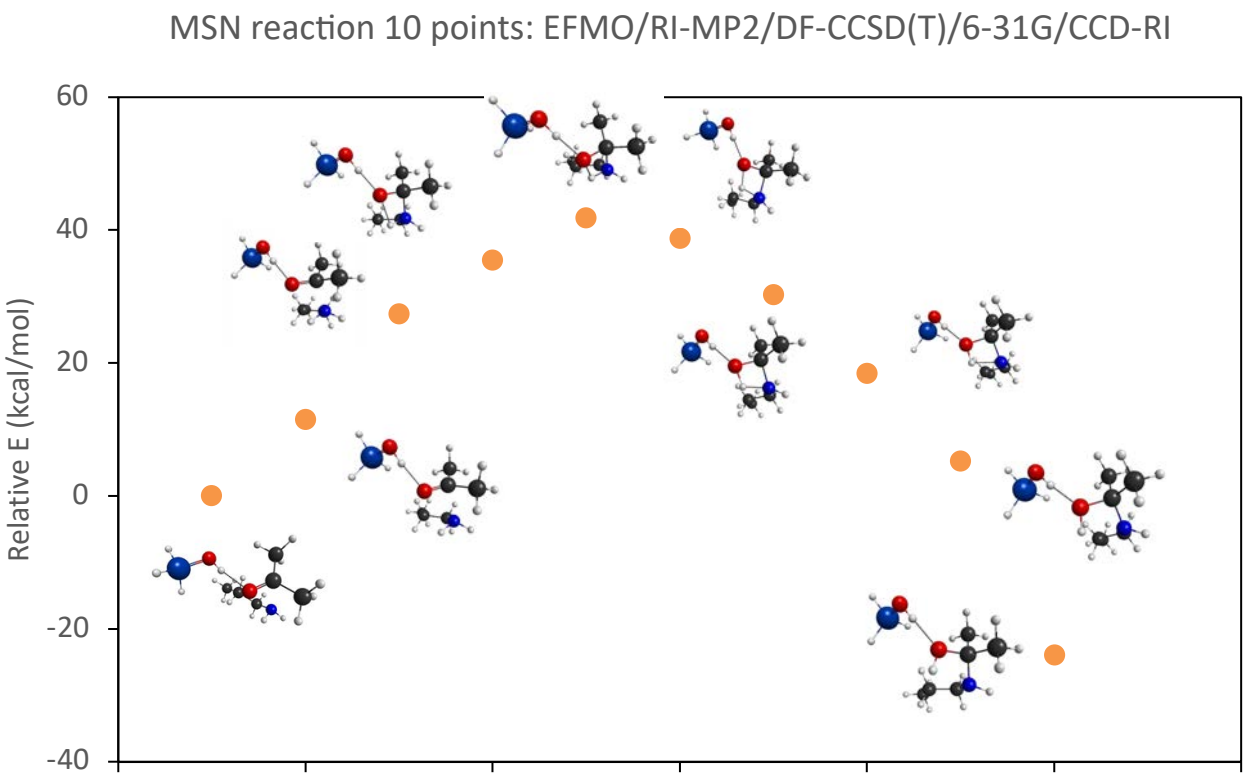


Bulk model (MSN + 4,600 H_2O)
15K atoms

#QM dimers= 9556
#EFP dimers= 6734

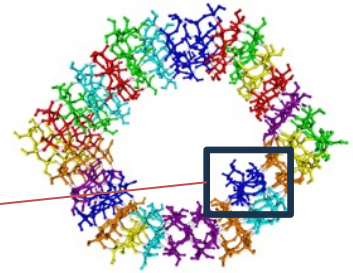
Multi-layer EFMO (ML-EFMO)

To achieve higher accuracy with insignificant increase of computational time

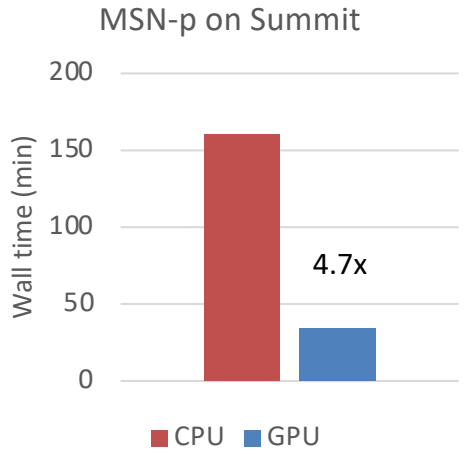


Layer 1: RI-MP2
MSN, water

Layer 2: RI-CCSD(T)
"Active region"
Reactant, TS, product



Offload: SCF, MAKEFP, RI-MP2, RI-CC



MSN-p on Frontier: speedup 2.7 X

GAMESS ECP Team

Team member

Ames National Lab
- Mark Gordon (PI)
- Melisa Alkan
- Daniel Del Angel
- Dipayan Datta
- Taylor Harville
- Buu Pham
- Bryce Westheimer
- Peng Xu
- Federico Zahariev
- Tosaporn Sattasathuchana

Old Dominion University
- Masha Sosonkina (co-PI)
- Viabhav Sundriyal

EP Analytics
- Laura Carrington (co-PI)
- Ananta Tiwari
- Sarom Leang

Australian National U
- Giuseppe Barca (co-PI)
- Jorge Galvez

Georgia Tech
- David Sherril (co-PI)
- *David Poole*

U Texas, El Paso
- Shirley Moore (co-PI)
- Henry Moncada

Acknowledgement
- Argonne: Colleen Bertoni
- Brookhaven: Dossay Oryspayev
- Lawrence Berkeley: Jack Deslippe
- Oak Ridge: Dmytro Bykov
- AMD: Brian Cornille, Bob Robey
- HPE Cray: Luke Roskop, Marcus Wagner
- Intel: Brian Whitney, Xinmin Tian, Ravi Narayanaswamy
- HPCToolkit: John Mellor-Crummey, Wileam Phan, Laksono Adhianto

OpenMP®

SC23 Booth Talk Series

openmp.org — OpenMP API specs, forum, reference guides, and more

link.openmp.org/sc23 — OpenMP SC'23 booth talk videos and presentations