OpenMP Offload in the GAMESS Quantum Chemistry Package

September 29, 2023
Funding

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy’s Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation’s exascale computing imperative.
GAMESS ECP Team

Mark Gordon (PI)

Melisa Alkan (Stanford)  
**Daniel Del Angel**  
Dipayan Datta

Taylor Harville  
**Buu Pham**  
**Tosaporn Sattasathuchana**

Bryce Westheimer  
Peng Xu  
Federico Zahariev

Giuseppe Barca (co-PI)  
**Jorge Galvez**  
Ananta Tiwari  
**Sarom Leang**

Laura Carrington (co-PI)  
David Sherril (co-PI)  
**David Poole**

Masha Sosonkina (co-PI)  
Viabhav Sundriyal (Intel)  
Eric Jensen

Shirley Moore (co-PI)  
Henry Moncada

*Former member of the Mark S. Gordon Research Group*
Acknowledgements

• Colleen Bertoni* (Argonne National Laboratory)
• Dossay Oryspayev (Brookhaven National Laboratory)
• Jack Deslippe (Lawrence Berkeley National Laboratory)
• Dmytro Bykov (Oak Ridge National Laboratory)
• AMD (Brian Cornille, Bob Robey)
• HPE Cray (Luke Roskop*, Marcus Wagner)
• Intel (Brian Whitney, Xinmin Tian, Ravi Narayanaswamy)
• HPC Toolkit (John Mellor-Crummey, Wileam Phan, Laksono Adhianto)

*Former member of the Mark S. Gordon Research Group
Acknowledgements

• **Hackathons**
  - GPU Hackathon, OLCF, 2019
  - ECP OpenMP, SOLLVE, 2020
  - GPU Hackathon, OLCF+NERSC, 2021
  - ECP OpenMP, SOLLVE, 2021
  - AMD, 2022
  - Open, NERSC, 2022 (Apr., Nov.)
  - Crusher, OLCF, 2022
  - Frontier, OLCF, 2023
  - Aurora, ALCF, 2023

• **Workshops**
  - ATPESC, ALCF, 2019
  - Aurora COE, ALCF, 2020
  - Frontier COE, OLCF, 2020

• **Training**
  - Spock, OLCF, 2021
  - Crusher, OLCF, 2022
  - Frontier, OLCF, 2023
  - HIP, OLCF + NERSC, 2023
Agenda

- GAMESS
- Electronic Structure Theory
- Initial Molecular Orbital Guess
- Resolution-of-Identity Møller-Plesset Second-Order Perturbation Theory (RI-MP2)
- Effective Fragment Molecular Orbital Method (EFMO)
- ECP Science Challenge
- Density Functional Theory (DFT)
- Issues, Workarounds, and Wishlist
- Questions
General Atomic and Molecular Electronic Structure System (GAMESS)

Principal Investigator: Mark S. Gordon (Ames Laboratory)
Maintainer/Release Manager: Sarom S. Leang (EP Analytics, Inc.)
GAMESS

• General purpose electronic structure code
• Focus on *ab initio* quantum chemistry
• Mark S. Gordon Quantum Theory Group/Iowa State University
• Lots of capabilities – too many to list
  [www.msg.chem.iastate.edu/gamess/capabilities.html](http://www.msg.chem.iastate.edu/gamess/capabilities.html)
• Two code bases:
  • GAMESS (Fortran 77/90/95/2003)
  • LIBCCHEM (C++)
• 2022 statistics: +9,000 downloads across 128 countries
• DOI: 10.1021/acs.jctc.3c00379 (accepted)
History

- Early 1980 - Fork of HONDO 5 from NRCC (Michel Dupuis)
- 1982 - Mike Schmidt joined Mark Gordon’s group at NDSU (80,000 LOC)
- 1991 - Parallelization (T. Windus)
- 1992 - Mark Gordon’s group moves to ISU
- 1996 - Distributed Data Interface (DDI) (G. Fletcher), EFP method (J. Jensen)
- 2004 - DDI for SMP using System V shared memory
- 2004 - DDI for subgroups, release of the FMO method (D. Fedorov) (650,000 LOC)
- 2007 - Release of the EFMO method (S. Pruitt)
- 2010 - Release of GAMESS-LIBCCHEM (A. Asadchev)
- 2017 - Mike Schmidt enters partial-retirement (759,000 LOC)
History (OpenMP Parallelization in GAMESS)

• 2016 – Two-electron integrals (V. Mironov)
• 2017 – Two-electron gradient code and HF algorithm w/ shared Fock (V. Mironov)
• 2017 – RI-MP2 energy code (B. Pham)
• 2018 – RI-MP2 gradient code (B. Pham)
• 2019 – One-electron integrals, DFT, FMO, PCM (V. Mironov)
• 2019 – QM-EFP2 energy code (P. Xu, T. Sattasathuchana)
• 2020 – RI-CC (D. Datta)
Citations by Year (References Tracked)

- **GAMESS**
  - 1993: https://doi.org/10.1002/jcc.540141112
  - 2005: https://doi.org/10.1016/B978-044451719-7/50084-6
  - 2020: https://doi.org/10.1063/5.0005188

- **NWChem**
  - 1995: https://doi.org/10.1002/qua.560560851
  - 2000: https://doi.org/10.1016/S0010-4655(00)00665-5
  - 2010: https://doi.org/10.1016/j.jcp.2010.04.018
  - 2020: https://doi.org/10.1063/5.0004997
  - 2021: https://doi.org/10.1021/acs.chemrev.0c00998

- **PSI4**
  - 2011: https://doi.org/10.1002/wcms.93
  - 2017: https://doi.org/10.1021/acs.jctc.7b00174
  - 2020: https://doi.org/10.1063/5.0006002

- **Dalton/LSDalton**
  - 2011: https://doi.org/10.1002/wcms.93

- **Molpro (Commercial)**
  - 2012: https://doi.org/10.1002/wcms.82
  - 2020: https://doi.org/10.1063/5.0005081

- **Turbomol (Commercial)**
  - 1989: https://doi.org/10.1016/0009-2614(89)85118-8
  - 2014: https://doi.org/10.1002/wcms.1162
  - 2020: https://doi.org/10.1063/5.0004635

- **Molcas**
  - 2003: https://doi.org/10.1016/S0927-0256(03)00109-5
  - 2010: https://doi.org/10.1002/jcc.21318
  - 2016: https://doi.org/10.1002/jcc.24221
  - 2019: https://doi.org/10.1021/acs.jctc.9b00532
  - 2020: https://doi.org/10.1063/5.0004835
Electronic Structure Theory

• Describe the motions of electrons
• Many-body problem with no closed solution
• Development and scaling of computational methods to describe electronic motion in atoms and molecules
Electronic Schrödinger Equation

\[ \hat{H}_{\text{elec}} \psi_{\text{elec}} = E_{\text{elec}} \psi_{\text{elec}} \]

\[ \hat{H}_{\text{elec}} = -\frac{1}{2} \sum_{i=1}^{N} \nabla_i^2 - \sum_{i=1}^{N} \sum_{A=1}^{M} \frac{Z_A}{r_{iA}} + \sum_{i<j}^{N} \frac{1}{r_{ij}} \]

- \( \hat{H}_{\text{elec}} \) is electronic Hamiltonian operator
- \( \psi_{\text{elec}} \) is electronic wavefunction of the system
- \( E_{\text{elec}} \) is the electronic energy of the system
- \( N \) is the total number of electrons
- \( M \) is the total number of nuclei
- Born-Oppenheimer approximation

Electronic Schrödinger Equation (cont.)

\[
\hat{H}_{\text{elec}} \Psi_{\text{elec}} = E_{\text{elec}} \Psi_{\text{elec}}
\]

\[
\hat{H}_{\text{elec}} = -\frac{1}{2} \sum_{i=1}^{N} \nabla_{i}^{2} - \sum_{i=1}^{N} \sum_{A=1}^{M} \frac{Z_{A}}{r_{iA}} + \sum_{i<j}^{N} \frac{1}{r_{ij}}
\]

- \( \Psi_{\text{elec}} \) contains all information about the system
  - Geometries (equilibrium, transition states)
  - Vibrational frequencies (IR spectra)
  - Excited states (UV/VIS spectra)
  - Dipole moment, polarizability
  - Barrier heights and reaction paths
  - Reaction rates with transition state theory
  - Thermodynamic properties with statistical mechanics
Hartree-Fock (HF or SCF or RHF) Method

\[ \hat{H}^{HF}_{elec} \Psi^{HF}_{elec} = E^{HF}_{elec} \Psi^{HF}_{elec} \]

\[ \hat{H}^{HF}_{elec} = -\frac{1}{2} \sum_{i=1}^{N} \nabla_i^2 - \sum_{i=1}^{N} \sum_{A=1}^{M} \frac{Z_A}{r_{iA}} + \sum_{i=1}^{N} V^{HF}_i \]

- Based on the independent particle model
- Electron-electron repulsion terms is replaced with an effective field produced from the averaged position of all other electrons
  - Results in an iterative process starting from a guessed set of functions (initial guess)
- Electron correlation energy,

\[ E_{correlation} = E - E_{HF} \]
Beyond Hartree-Fock

- Increased sophistication of methods to approach exact solution to the electronic Schrödinger equation
- Formal scaling of methods based on system size, $N$

<table>
<thead>
<tr>
<th>Scaling Behavior</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^4$</td>
<td>Hartree-Fock (HF), Density Function Theory (DFT)</td>
</tr>
<tr>
<td>$N^5$</td>
<td>Møller-Plesset Second-Order Perturbation Theory (MP2)</td>
</tr>
<tr>
<td>$N^7$</td>
<td>Coupled-Cluster with Singles and Doubles and Perturbative Triples excitations (CCSD(T))</td>
</tr>
<tr>
<td>$N^8$</td>
<td>Coupled-Cluster with Singles and Doubles and Triples excitations (CCSDT)</td>
</tr>
<tr>
<td>$N!$</td>
<td>Full Configuration-Interaction (Full-CI)</td>
</tr>
</tbody>
</table>
Initial Molecular Orbital Guess

Lead: Daniel Del Angel (Ames National Laboratory)
Initial Guess

- Electronic structure described in terms of molecular orbitals (MOs)
- MOs optimized through an iterative procedure
- Starting set of MOs (initial guess)
- Matrix operations used for initial guess generation
- Acceleration approach: use GPU libraries (e.g., cuBLAS and cuSOLVER)
Offloaded vs. Non-Offloaded Code

• Original CPU code uses BLAS calls for matrix operations
• Offloaded code contains a Fortran interface to the C-based cuBLAS and cuSOLVER libraries
• Conditional check to determine if CPU or GPU approach is taken
Offloading Results (Initial Guess)

- Calculations performed on Perlmutter with 128 logical cores per node
- GPU runs done with 4 MPI ranks per node, 16 threads per rank
- CPU runs done with 8 MPI ranks per node, 8 threads per rank
Resolution-of-Identity Møller-Plesset Second-Order Perturbation Theory (RI-MP2)

Lead: Buu Q. Pham (Ames National Laboratory)
Møller-Plesset Second-Order Perturbation Theory (MP2)

The MP2 correlation energy is:

\[ E^{(2)} = \sum_{i>j}^{\text{occ.}} (2 - \delta_{ij}) \sum_{ab}^{\text{vir.}} \frac{(ia|jb)[(ia|jb) - (ib|ja)]}{\varepsilon_i + \varepsilon_j - \varepsilon_a - \varepsilon_b} \]

where the MO-based four-index two-electron repulsion integrals (4-2ERIs) are,

\[ (ia|jb) = \sum_{\sigma}^{N} C_{\sigma b} \sum_{\nu}^{N} C_{\nu a} \sum_{\lambda}^{N} C_{\lambda j} \sum_{\mu}^{N} C_{\mu i}(\mu \nu|\lambda \sigma) \]

and AO-based 4-2ERIs are,

\[ (\mu \nu|\lambda \sigma) = \int \int \, dr_1 \, dr_2 \, \phi_{\mu}^*(r_1) \phi_{\nu} (r_1) \frac{1}{|r_1 - r_2|} \phi_{\lambda}^*(r_2) \phi_{\sigma} (r_2) \]
Resolution-of-Identity MP2 (RI-MP2)

\[(ia|jb) \approx \sum_P^X B_{ia}^P B_{jb}^P\]

\[B_{ia}^P = \sum_R (ia|R) W_{RP}^{\frac{1}{2}}\]

\[(ia|P) = \sum_\mu^{NB} \sum_\nu^{NB} C_{\mu i} C_{\nu a}(\mu\nu|P)\]

\[(\mu\nu|P) = \int dr_1 dr_2 \phi_\mu^*(r_1) \phi_\nu(r_1) \frac{1}{|r_1 - r_2|} \alpha_P(r_2)\]

\[W_{PQ} = \int dr_1 dr_2 \alpha_P(r_1) \frac{1}{|r_1 - r_2|} \alpha_Q(r_2)\]

- 4-2ERI is approximated with 2-2ERIs and 3-2ERIs
- Four main wall-time components
  - Evaluation of 2-2ERIs
  - Evaluation of 3-2ERIs
  - Contracting 3-2ERI with Cholesky vectors of the inverse 2-2ERI matrix
  - Distributed memory operations and data transfers

DOI: 10.1063/5.0143424

Compute: $N^4$
Memory: $N^3$
Fortran Interfaces to Accelerated Libraries

```fortran
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,lda,dB,ldb,
    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
  real(c_double) :: alpha
  real(c_double), dimension(*) :: dA
  integer(c_int), value :: ldaa
  real(c_double), dimension(*) :: dB
  integer(c_int), value :: ldb
  real(c_double) :: beta
  real(c_double), dimension(*) :: dC
  integer(c_int), value :: ldc
end function cublasXtDgemm
```

```fortran
!$omp target data map(alloq:QV)
!$omp target data map(to:BI,BJ)
!$omp target data use_device_ptr(BI,BJ,QVV)
cublas_return = cublasXtDgemm(handle,transa,transb,
  m,n,k,alpha,dA,lda,dB,ldb,
  beta,dC,lddc) &
  (cublas_handle,CUBLAS_OP_T,CUBLAS_OP_N, &
  NVIR*QVV,NVIR,NAUXBASD, &
  one,BI,NAUXBASD, &
  BJ,NAUXBASD, &
  zer,QVV,NVIR*QVV)
cublas_return = cudaDeviceSynchronize()!
!$omp target end data
!$omp target end data
!$omp target end data
```

```fortran
!$omp target data map(alloq:QV)
!$omp target data map(to:BI,BJ)
!$omp target data use_device_ptr(BI,BJ,QVV)
call hipblasCheck(hipblasDgemm( &
  HIPBLAS_handle, 4 ! type(c_ptr), value
  HIPBLAS_OP_T, 4 ! integer(kind(hipblas_op_t)), value
  HIPBLAS_OP_N, 4 ! integer(kind(hipblas_op_n)), value
  NVIR*QVV, 4 ! integer(c_int), value
  NVIR, 4 ! integer(c_int), value
  NAUXBASD, 4 ! integer(c_int), value
  one, 4 ! real(c_double)
  c_loc(BI(1)), 4 ! type(c_ptr), value
  NAUXBASD, 4 ! integer(c_int), value
  c_loc(BJ(1)), 4 ! type(c_ptr), value
  NAUXBASD, 4 ! integer(c_int), value
  zero, 4 ! real(c_double)
  c_loc(QVV(1,1,1)), 4 ! type(c_ptr), value
  NVIR*QVV 4 ! integer(c_int), value
))
call hipCheck(hipDeviceSynchronize())!
!$omp target end data
!$omp target end data
!$omp target end data
```

25
Single Node Capability

• Summit (512 GB/node):
  • \( \sim 5,000 \) atomic orbitals \& \( \sim 15,000 \) auxiliary basis functions
• Perlmutter (256 GB/node):
  • \( \sim 4,000 \) atomic orbitals \& \( \sim 12,000 \) auxiliary basis functions
• Aurora (1024 GB/node)*:
  • \( \gg 5000 \) atomic orbitals \& \( \gg 15,000 \) auxiliary basis functions
• Frontier (512 GB/node):
  • \( > 5000 \) atomic orbitals \& \( > 15,000 \) auxiliary basis functions

*https://www.alcf.anl.gov/aurora
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
- $R_{cut}=1.0$
- 9,556 dimers
- 13,791 atoms
Fragmentation Methods

Fragment Molecular Orbital (FMO)
Effective Fragment Potential (EFP)
Effective Fragment Molecular Orbital (EFMO)

Leads: Peng Xu and Tosaporn Sattasathuchana (Ames National Laboratory)
Fragment Molecular Orbital (FMO)
Fragment Molecular Orbital (FMO) Method

\[ E \cong \sum_{I}^{n_F} E_I + \sum_{I<J}^{n_F} \Delta E_{IJ} + \sum_{I<J<K}^{n_F} \Delta E_{IJK} + \cdots \]

Many-body expansion of the total energy

\[ \Delta E_{IJ} = E_{IJ} - E_I - E_J \]

E(2-body interaction)

\[ \Delta E_{IJK} = E_{IJK} - E_I - E_J - E_K - (E_{IJ} - E_I - E_J) - (E_{JK} - E_J - E_K) - (E_{IK} - E_I - E_K) \]

E(3-body interaction)
Effective Fragment Potential (EFP)
Effective Fragment Potential (EFP) Method

- *ab initio* force field method designed to capture intermolecular interactions
- \( E_{interaction} = E_{Coul} + E_{Pol} + E_{Disp} + E_{EXREP} + E_{CT} \)
- Includes many-body polarization: induced dipoles are iterated to self-consistency

<table>
<thead>
<tr>
<th>EFP components</th>
<th>Required Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coulomb (Coul)</td>
<td>Multipole moments up to octopoles located at atom centers and bond midpoints</td>
</tr>
<tr>
<td>Polarization (Pol)</td>
<td>Static dipole polarizability tensors distributed at localized MO (LMO) centroids</td>
</tr>
<tr>
<td>Dispersion (Disp)</td>
<td>Imaginary frequency dependent polarizability tensors distributed at LMO centroids</td>
</tr>
<tr>
<td>Exchange Repulsion (EXREP)</td>
<td>LMOs and LMO centroid coordinates, Fock matrix in LMO basis</td>
</tr>
<tr>
<td>Charge Transfer (CT)</td>
<td>Occupied + virtual molecular orbitals, eigenvalues of Fock matrix</td>
</tr>
</tbody>
</table>
Effective Fragment Molecular Orbital (EFMO)
Effective Fragment Molecular Orbital (EFMO) Method

\[ E^{\text{EFMO}} = \sum_I E_I^0 + \sum_{I<J}^{R_{IJ}<R_{\text{cut}}} \left[ (E_{IJ}^0 - E_I^0 - E_J^0) - E_{IJ}^{\text{pol}} \right] + \sum_{I>J}^{R_{IJ}>R_{\text{cut}}} \left( E_{IJ}^{\text{Coul}} + E_{IJ}^{\text{disp}} + E_{IJ}^{\text{ExRep}} + E_{IJ}^{\text{CT}} \right) + E_{\text{tot}}^{\text{pol}} \]

- Evaluate monomer energy in vacuo
- QM dimers: proximate dimer treated by QM method
- EFP dimers: distant dimer treated by EFP
- Many-body effect captured by EFP polarization

\[ R_{IJ} = \min_{i \in I, j \in J} \left\{ \frac{|\vec{r}_i - \vec{r}_j|}{r_i^{vdw} + r_j^{vdw}} \right\} \]

Relative minimum distance between fragments I and J

\( R_{\text{cut}} \): A user-defined, distance-related, dimensionless threshold value. The larger the \( R_{\text{cut}} \), the more dimers are treated as QM dimers.
Effective Fragment Molecular Orbital (EFMO) Method

**EFMO Calculation Overview**

**Monomer**
- SCF $\rightarrow$ HF method
- MAKEFP
- Correlated method if required

**Total Polarization**
- Total polarization from EFP

**Dimers**
- QM dimers
- EFP-EFP polarization for QM dimers
- EFP dimers

**Approach**
- Offload EFMO bottlenecks
  - SCF
  - MAKEFP (CPHF and TDHF)
  - Correlated method (e.g., RI-MP2)
- Keep on the CPU
  - EFP dimer
  - EFP polarization
Offloading HF/MAKEFP (CPHF and TDHF)

• CPHF and TDHF
  • Common to HF and MAKEFP

• Bottleneck
  • Evaluation of ERIs
    • 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

```fortran
!$omp target distribute parallel do &
!$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)
end do
```

```fortran
!$omp atomic update
   FA(IJ) = FA(IJ)+4*ERIs(ijkl)*DA(KL)
!$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)
```

subroutine dirfck_rhf(i,j,k,l,ERIs,nintn,DA,FA)
Offloading Results (HF/MAKEFP)

<table>
<thead>
<tr>
<th>Using 1GPU</th>
<th>Time from miniERIs (an iteration of RHF method) (s)</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Summit</td>
<td>Perlmutter</td>
<td>Frontier</td>
<td>Sunspot</td>
<td></td>
</tr>
<tr>
<td>Kernel</td>
<td>xlf/16.1.1-10</td>
<td>nvhpc/22.5</td>
<td>nvhpc/22.7</td>
<td>cce/15.0.1</td>
<td>cce/16.0.0 rocm5.5.1</td>
</tr>
<tr>
<td>j01</td>
<td>0.26</td>
<td>0.03</td>
<td>0.02</td>
<td>0.01</td>
<td>0.02</td>
</tr>
<tr>
<td>j02</td>
<td>1.53</td>
<td>0.16</td>
<td>0.09</td>
<td>0.07</td>
<td>0.24</td>
</tr>
<tr>
<td>j03</td>
<td>1.98</td>
<td>0.15</td>
<td>0.08</td>
<td>0.2</td>
<td>0.72</td>
</tr>
<tr>
<td>j04</td>
<td>3.88</td>
<td>0.24</td>
<td>0.15</td>
<td>1.98</td>
<td>0.85</td>
</tr>
<tr>
<td>j05</td>
<td>16.96</td>
<td>1.09</td>
<td>0.54</td>
<td>5.95</td>
<td>15.8</td>
</tr>
<tr>
<td>j06</td>
<td>79.33</td>
<td>8.92</td>
<td>4.52</td>
<td>5.77</td>
<td>22.44</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td><strong>103.94</strong></td>
<td><strong>10.59</strong></td>
<td><strong>5.4</strong></td>
<td><strong>13.98</strong></td>
<td><strong>40.07</strong></td>
</tr>
</tbody>
</table>
# Portability of Offloaded Code

<table>
<thead>
<tr>
<th>Accelerator</th>
<th>V100</th>
<th>A100</th>
<th>MI250X</th>
<th>PVC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compiler</td>
<td>NVHPC</td>
<td>NVHPC</td>
<td>CCE</td>
<td>oneAPI</td>
</tr>
<tr>
<td>Center</td>
<td>OLCF</td>
<td>ALCF</td>
<td>OLCF</td>
<td>ALCF</td>
</tr>
<tr>
<td></td>
<td>Summit</td>
<td>Polaris</td>
<td>Frontier</td>
<td>Aurora</td>
</tr>
</tbody>
</table>
ECP Science Challenge

Leads: Peng Xu, Tosaporn Sattasathuchana, and Buu Q. Pham (Ames National Laboratory)
ECP Science Challenge

Stretch problem \( \sim 150,000 \) basis functions

Base problem \( \sim 15,000 \) basis functions

Calculations of real heterogeneous catalysts at HF/RI-MP2/RI-CCSD(T) level of accuracy
ECP Science Challenge

Base problem ~15,000 basis functions

Stretch problem ~150,000 basis functions

Calculations of real heterogeneous catalysts at HF/RI-MP2/RI-CCSD(T) level of accuracy

High scaling methods
- Huge computational costs
- Enormous memory demands
- Scalability in practice (handling fault tolerance)
Approach

- Capture relevant QM effects
- Provide ideal scaling framework
- Preserve accuracy of underlying QM methods
- Reduce memory footprint
- Provide ideal data structure for GPU offloading
- Handling large numerical operation
- Enhance accuracy
## Offloading Results (Frontier)

EFMO/RI-MP2/6-31G/cc-pVDZ-ri

### CPU and CPU+GPU timings comparisons for MSN5-HYD on Frontier (128 nodes)

<table>
<thead>
<tr>
<th>Run</th>
<th>Rcut</th>
<th>Best FMO Energy (Hartree)</th>
<th>Total Wall Time (min)</th>
<th>Speed-up (x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>1</td>
<td>-382235.178290952</td>
<td>133.4</td>
<td></td>
</tr>
<tr>
<td>CPU+GPU</td>
<td>1</td>
<td>-382235.178290833</td>
<td>29.3</td>
<td>4.6</td>
</tr>
<tr>
<td>CPU</td>
<td>2</td>
<td>-382236.589264343</td>
<td>174.0</td>
<td></td>
</tr>
<tr>
<td>CPU+GPU</td>
<td>2</td>
<td>-382236.589266623</td>
<td>35.5</td>
<td>4.9</td>
</tr>
</tbody>
</table>

cpe/22.12, cce/15.0.0, rocm/5.3.0
128 nodes, 16 MPI ranks per node, 6 threads per rank
### Kernel Timings on Frontier (ROCPROF)

**EFMO/RI-MP2/6-31G/cc-pVDZ-ri**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Timings (s)</th>
<th>HIP API Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Kernels</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RHF</td>
<td>CPHF</td>
</tr>
<tr>
<td>0</td>
<td>114.9</td>
<td>50.6</td>
</tr>
<tr>
<td>1</td>
<td>111.5</td>
<td>53.7</td>
</tr>
<tr>
<td>2</td>
<td>107.3</td>
<td>52.3</td>
</tr>
<tr>
<td>3</td>
<td>93.2</td>
<td>44.2</td>
</tr>
<tr>
<td>4</td>
<td>96.7</td>
<td>44.9</td>
</tr>
<tr>
<td>5</td>
<td>100.8</td>
<td>46.5</td>
</tr>
<tr>
<td>6</td>
<td>98.7</td>
<td>45.4</td>
</tr>
<tr>
<td>7</td>
<td>101.5</td>
<td>50.5</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>103.1</td>
<td>48.5</td>
</tr>
</tbody>
</table>
Kernel Timings on Frontier (ROCPROF)

EFMO/RI-MP2/6-31G/cc-pVDZ-ri

Kernel and HIP API timing statistics for ranks 0-7 of a 128 node run of MSN5-HYDR on Frontier

<table>
<thead>
<tr>
<th>Rank</th>
<th>Kernels</th>
<th>HIP API Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RHF</td>
<td>CPHF</td>
</tr>
<tr>
<td>0</td>
<td>114.9</td>
<td>50.6</td>
</tr>
<tr>
<td>1</td>
<td>111.5</td>
<td>53.7</td>
</tr>
<tr>
<td>2</td>
<td>107.3</td>
<td>52.3</td>
</tr>
<tr>
<td>3</td>
<td>93.2</td>
<td>44.2</td>
</tr>
<tr>
<td>4</td>
<td>96.7</td>
<td>44.9</td>
</tr>
<tr>
<td>5</td>
<td>100.8</td>
<td>46.5</td>
</tr>
<tr>
<td>6</td>
<td>98.7</td>
<td>45.4</td>
</tr>
<tr>
<td>7</td>
<td>101.5</td>
<td>50.5</td>
</tr>
<tr>
<td>Average</td>
<td>103.1</td>
<td>48.5</td>
</tr>
</tbody>
</table>

GPU Utilization
- 1760.7 s for non-profiled run on 128 nodes
- 45% of total wall-time spent in kernels (797 s)
- 51% of total wall-time if DT time is included
## Offloading Results (Aurora) *Preliminary*

EFMO/RI-MP2/6-31G/cc-pVDZ-ri

<table>
<thead>
<tr>
<th>Run</th>
<th>Rcut</th>
<th>Best FMO Energy (Hartree)</th>
<th>Total Wall Time (min)</th>
<th>Speed-up (x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>1</td>
<td>-382235.178290892</td>
<td>48.7</td>
<td></td>
</tr>
<tr>
<td>CPU+GPU</td>
<td>1</td>
<td>-382235.178290870</td>
<td>15.0</td>
<td>3.2</td>
</tr>
<tr>
<td>CPU</td>
<td>2</td>
<td>-382236.589265166</td>
<td>62.4</td>
<td></td>
</tr>
<tr>
<td>CPU+GPU</td>
<td>2</td>
<td>-382236.589265446</td>
<td>18.5</td>
<td>3.4</td>
</tr>
</tbody>
</table>

Contact Colleen Bertoni (Argonne) for run details
## Offloading Results (Polaris)

EFMO/RI-MP2/6-31G/cc-pVDZ-ri

<table>
<thead>
<tr>
<th>Run</th>
<th>Rcut</th>
<th>Best FMO Energy (Hartree)</th>
<th>Total Wall Time (min)</th>
<th>Speed-up (x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>1</td>
<td>-382235.178290867</td>
<td>121.0</td>
<td></td>
</tr>
<tr>
<td>CPU+GPU</td>
<td>1</td>
<td>-382235.178290870</td>
<td>17.8</td>
<td>6.8</td>
</tr>
</tbody>
</table>

nvhpc/23.3, CUDA 11.8
128 nodes, 8 MPI ranks per node, 6 threads per rank
Density Functional Theory

Lead: Federico Zahariev (Ames National Laboratory)
Electronic Schrödinger Equation (Revisited)

\[ \hat{H}_{\text{elec}} \Psi_{\text{elec}} = E_{\text{elec}} \Psi_{\text{elec}} \]

\[ \hat{H}_{\text{elec}} = -\frac{1}{2} \sum_{i=1}^{N} \nabla_i^2 - \sum_{i=1}^{N} \sum_{A=1}^{M} \frac{Z_A}{r_{iA}} + \sum_{i<j} \frac{1}{r_{ij}} \]

- \( \Psi_{\text{elec}} \) is a function of 4\( N \) variables where \( N \) is the number of electrons in system.
- Hohenberg-Kohn theorem states that the ground state electronic energy is determined completely by the electron density (a function of 3 spatial variables),

\[ \rho(r) = N \int \int ds \, dx_2 \ldots dx_N |\Psi_{\text{elec}}(x_1, x_2, \ldots, x_N)|^2 \]

The energy is a functional of the electronic density.

Density Functional Theory

• Hohenberg-Kohn theorem establishes relationship between energy and electron density through the existence of a universal functional

• Universal functional is not known and gives rise to the development of density functional approximations (e.g., M06, PBE0, \(\omega\)B97-X)

• Kohn-Sham formalism of DFT enables leveraging existing machinery (HF) in quantum chemistry codes to perform a DFT calculation
  • Hamiltonian includes an exchange-correlation potential,
    \[
    v_{xc} = \frac{\delta E_{xc}}{\delta \rho}
    \]
    where,
    \[
    E_{xc} = \int \rho(r) f_{xc} \left( \rho_\alpha, \rho_\beta, \gamma_{\alpha\alpha}, \gamma_{\alpha\beta}, \gamma_{\beta\beta}, \tau_\alpha, \tau_\beta \right) \text{d}r
    \]
    Density Functional Approximation

Solved using numerical quadrature
Offload Exchange-Correlation (XC) Component of the Fock Matrix

Recursive partitioning of the XC grid:

Batch compression of matrices due to the sparse grid:

Data prep for DGEMM and DSYR2K to minimize DT


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

Offloading Results (DFT) *Preliminary*

**Speed-up vs. Number of Waters**

<table>
<thead>
<tr>
<th>Speed Up DFA/Basis Set</th>
<th>Number of Water Molecules</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
</tr>
<tr>
<td>SVWN/6-311G</td>
<td>2.8</td>
</tr>
<tr>
<td>BLYP/6-311G</td>
<td>2.6</td>
</tr>
<tr>
<td>M06/6-311G</td>
<td>3.4</td>
</tr>
<tr>
<td>SVWN/6-31G(d)</td>
<td>2.2</td>
</tr>
<tr>
<td>BLYP/6-31G(d)</td>
<td>4.0</td>
</tr>
<tr>
<td>M06/6-31G(d)</td>
<td>2.6</td>
</tr>
</tbody>
</table>

Speed-up = Ratio of OpenMP and OpenMP Offload timing for XC component on 1 Summit node.

**Extrapolations of the speed-ups**

- **SVWN**
- **BLYP**
- **M06**

![Graphs showing extrapolations of speed-ups for different basis sets and methods.](image_url)
Issues and Workarounds
Issues and Workarounds

• Intel
  • JIT is long (CMPLRLLVM-48607).  Use AOT.
  • Atomic update is very slow for Fortran (CMPLRLLVM-46117).  Use C code.
  • Issues with common blocks/modules (CMPLRLLVM-51444, incorrect result).
    Use C code.
  • Pass common block as argument list

• CCE
  • Only CCE needs 'simd' clause to enable multithread parallelism. Code portability issue down the road.
    This will have conflict with Intel compiler.

• AMD flang
  • NINT function is not implemented for GPU.  Write the function manually.
  • Can not do reduction for integer (compile error).  Remove it as it does not affect the correctness of the answer.
  • Can not have 'atomic update' in a call.
    Manually inline or use an INCLUDE.
Issues with Larger Kernels for SPD Integrals in GAMESS

• Intel
  • Issues with common blocks/modules (CMPLRLLVM-51444, incorrect result).
    ~ passing as argument list.
  • Compile and link times using AOT > 2 hours (CMPLRLLVM-48607).
  • Lots of spills for large kernels, which lead to incorrect answer (GSD-6123).
    ~ IGC_ForceOCLSIMDWidth=16

• CCE
  • Link time ~ 1 hour
Wish List
Wish List

• OpenMP function to use shared ‘\$omp shared (...)’
• Dynamic scheduling for GPU offloading with OpenMP
• Reduction for arrays
• Unify across compiler vendors
Outlook

• OpenMP offload focused GAMESS public release in November (SC23)
  • Summit, Polaris, Perlmutter, Frontier, Aurora support
  • https://www.msg.chem.iastate.edu/gamess/download.html

• Publications on OpenMP offload work coming soon!
  • RI-MP2
  • RI-CC (accepted)
  • EFMO
  • DFT
Thank You!

gamess@iastate.edu
Hidden Slides