Use case: Optimizing the Weather Research and Forecasting Model (WRF) with OpenMP Offload and Codee

NERSC Codee Training, September 6, 2024

Namo Wichitrnithed
Oden Institute, UT Austin

NERSC
Woo-Sun Yang
Helen He
Brad Richardson

Pacific Northwest National Laboratory
Koichi Sakaguchi
William I. Gustafson Jr.

Appentra Solutions S.L.
Manuel Arenaz
Ulises Costi Blanco
Alvaro Goldar Dieste

The Hebrew University of Jerusalem
Jacob Shpund

The Weather Research & Forecasting Model

- An atmospheric model written in Fortran written in the 1990's by a number of organizations such as the National Center for Atmospheric Research (NCAR) and the National Centers for Environmental Prediction (NCEP)
- Solves the 3D Euler equations using finite differences and explicit timestepping
- Used in both research and operational, real-time forecasting worldwide
- NERSC development branch: https://github.com/NERSC/WRF

Optimization goals

- Current parallelism: domain decomposition (MPI) into patches (`ims:ime, kms:kme, jms:jme`) and shared memory (OpenMP) among tiles (`its:ite, kts:kte, jts:jte`)
- MPI + GPU approach: offloading work from each patch to a GPU
- Programming workflow
  - Profilers (gprof, perftools, Nsight)
  - Static code inspection (Codee)

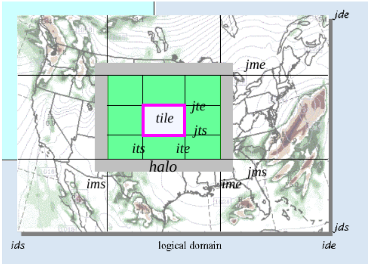


Figure 1: WRF decomposition layer. Image from Dudhia, J. "WRF Modeling System Overview".

Fast Spectral-Bin Microphysics (FSBM)

- Particle size spectrum divided into 33 intervals (bins)
- Computations required for each particle type and size at each grid point
- Require small timesteps (5-10 s)
- Universal; can be used for different atmospheric phenomena
- Current version in WRF: FSBM-2 (Shpund et al., 2019)

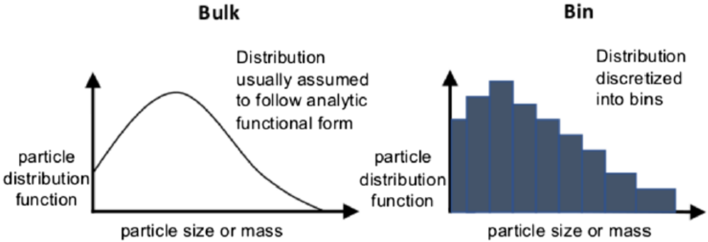
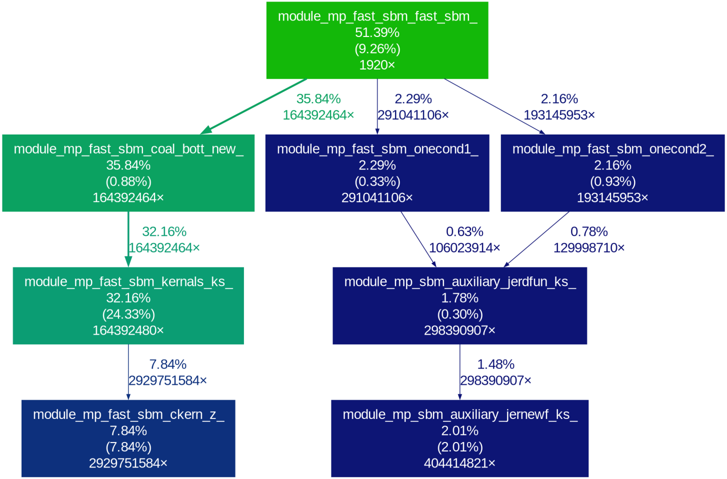


Figure 2: Image from Morrison et al., 2020.

Test case setup on Perlmutter

- Conus-12km test case
  - 425 x 300 x 50 grid
  - One-day restart
  - Time step: 5s
- Compilers: PrgEnv-nvidia/8.5.0 (NVHPC 23.9)
  - nvfortran, nvc, nvc++
  - Good GPU support for OpenMP, OpenACC, CUDA
  - GPU flags: `-mp=gpu -target-accel=nvidia80`
- WRF configure option: 4 (dm+sm) PGI (pgf90/gcc)

Finding time-consuming routines with Gprof



Inside the FSBM routine

- Subroutine Fast_SBM() in phys/module_mp_fast_sbm.F coal_bott_new()

```fortran
    do j = jts:jte
      do k = kts:kte
        do i = its:ite
          ! Collision-Coalescence process
          call COAL_BOTT_NEW(...)

          ! do stuff
        enddo
      enddo
    enddo


```

OpenMP GPU offloading

- A set of directives for C and Fortran that let the compiler generate GPU code
- Can manage parallelism and data transfer like CUDA API
- Directive-based: more portable and easier to port existing code to GPU, but less control

```
#pragma omp target teams
distribute parallel for
num_teams(3)
for (int i = 0; i < 12; ++i)
{
    C[i] = A[i] + B[i];
}
```

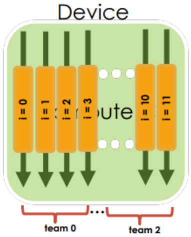


Figure 3: Image from https://www.olcf.ornl.gov/wp-content/uploads/2021/08/ITOpenMP_Day1.pdf

Inside the FSBM routine

- Subroutine `Fast_SBM()` in `phys/module_mp_fast_sbm.F`
- Parallelization granularity: number of grid points, assuming no race conditions inside `coal_bott_new()`

```fortran
do j = jts:jte
  do k = kts:kte
    do i = its:ite
      ! Collision-Coalesence process
      call COAL_BOTT_NEW(...)

      ! do stuff
    enddo
  enddo
enddo
```

Inside the Kernals_KS subroutine

- Global collision tables, e.g. cwlg,cwls are being modified at each grid point (i,k,j)

```
do n = 1,33
  do m = 1,33
    ckern_1 = ...
    ckern_2 = ...
    ! water - graupel
    cwlg(m,n) = (ckern_2 + (ckern1-ckern_2* ..)) * ...

    ckern_1 = ...
    ckern_2 = ...
    ! water - snow
    cwls(m,n) = (ckern_2 + (ckern1-ckern_2* ..)) * ...

    ! 18 more arrays

  enddo
enddo

```

Setting up Codee for WRF

```
# Capture compilation flags in JSON file
bear -- ./compile -j 8 wrf

# Initial screening report
codee screening --config compile_commands.json

# Checks report
codee checks --config compile_commands.json

# Example: in-place OpenMP offload insertion
codee rewrite --offload omp --in-place \
  module_mp_fast_sbm.f90:6293:4 \
  --config compile_commands.json
```

Codee analysis of Kernals_KS

- Codee implies there are no loop-carried dependencies, so the individual entries can be computed independently

```fortran
! Codee: Loop modified
!$omp target teams distribute parallel do &
!$omp private(n) map(from: cwlg, cwls, ...) ...
do n = 1,33
  ! Codee: Loop modified
  !$omp simd
  do m = 1,33
    ckern_1 = ...
    ckern_2 = ...
    cwlg(m,n) = (ckern_2 + (ckern1-ckern_2* ..)) * ...

    ckern_1 = ...
    ckern_2 = ...
    cwls(m,n) = (ckern_2 + (ckern1-ckern_2* ..)) * ...

    ! 18 more arrays
  enddo
enddo
```

Removing the global arrays

- Replace looking up m, n entry with computing as needed

```
    pure real function get_cwlg(..., m, n)
    pure real function get_cwls(..., m, n)

```

- No more shared arrays between grid points
- Speedup: around 1.4x
  - A lot of collision types are not used in FSBM
  - Not every entry m,n are used

Offloading the main loop

- Each grid point can now be assigned to a thread
- Further memory optimization allows a full collapse(3)

```
!$omp target teams distribute parallel do collapse(3)
do j = jts:jte
  do k = kts:kte
    do i = its:ite
      ! Collision - Coalescence
      call COAL_BOTT_NEW(...)

      ! do stuff
    enddo
  enddo
enddo
```

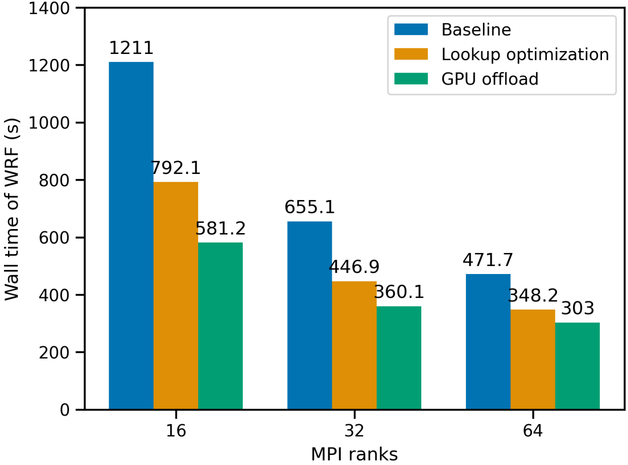
Speedup results

- 10-minute runs
- 1 OpenMP thread per MPI rank
- 1 GPU per MPI rank

Routine	Total speedup
coal_bott_new loop	66.6x
fast_sbm	2.99x
Overall	2.20x

Strong scaling

- No. of GPUs is fixed to 16, and no. of MPI ranks is varied from 16 to 64



Summary

- Accelerated a big part of the FSBM routine to GPUs through loop restructuring and OpenMP device offload
  - Codee's dependency analysis functionality exposed independence between computations among different grid points
- Achived an overall speedup of 2.2x for the 1 GPU per rank case for the CONUS-12km case
- A combination of runtime profiling and static code analysis is a very helpful aid in optimization efforts, especially for those not fully familiar with the context of the code

QUESTIONS