GE Global Research

Porting Computational Physics Applications to the Titan Supercomputer with OpenACC and OpenMP

Aaron Vose - Cray Inc.

GTC - 03/19/2015

Oak Ridge National Laboratory

Argonne National Laboratory

Porting - Overview

- **Porting methodology:**
  - Express underlying algorithmic parallelism.
    - Port to OpenMP first.
    - Port to OpenACC second.
- **Case studies / examples:**
  - TACOMA GE Global Research
  - Delta5D Oak Ridge National Laboratory
  - NekCEM Argonne National Laboratory

Porting - Overview

- **For each case study / example code: (TACOMA, Delta5D, and NekCEM)**

GE Global Research   Oak Ridge National Laboratory   Argonne National Laboratory

- Introduction to the code and example loop.
- OpenMP / OpenACC porting of the loop.
  - Express underlying algorithmic parallelism.
- OpenACC data motion with simplified call tree.
- Performance results.

Porting - OpenMP

- **Express existing loop-level parallelism with OpenMP directives.**
  - Cray's Reveal tool can do much of this automatically.
- **Port to OpenMP before OpenACC.**
  - OpenACC can reuse most OpenMP scoping.
  - OpenMP porting to CPU is easier than OpenACC porting to GPU.
  - Data motion can be ignored when porting to OpenMP.
- **Modify loops to expose more of underlying algorithms' parallelism.**

Porting - OpenACC

- **Identify candidate loops:**
  - Check loops’ trip/iteration count (CrayPAT).
- **Add OpenACC directives / Optimize Kernels:**
  - Check compiler listing for proper vectorization.
  - Ignore data motion (best performed once kernels are done and have known data requirements).
- **Finally, optimize device <-> host data motion.**
  - Perform bottom-up, hierarchical data optimization.

Porting - TACOMA

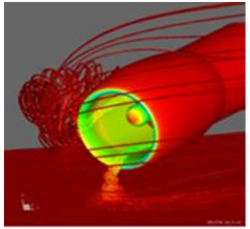
Case Study I: TACOMA

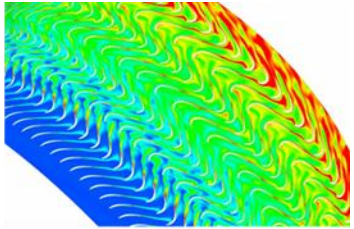
GE Global Research

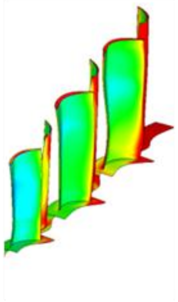
GE Global Research

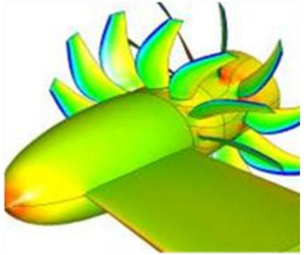
Porting - TACOMA

- **From GE’s Brian Mitchell.**
- **Computational fluid dynamics is essential to design jet engines, gas/steam turbines, and more.**
- **Finite-volume, block-structured, compressible flow solver, with stability achieved via JST.**









Porting - TACOMA

- **Example loop nest from TACOMA.**
  - Representative of a number of costly routines.
  - Can be made to parallelize on CPUs with OpenMP.
  - GPU vectorization requires more work.

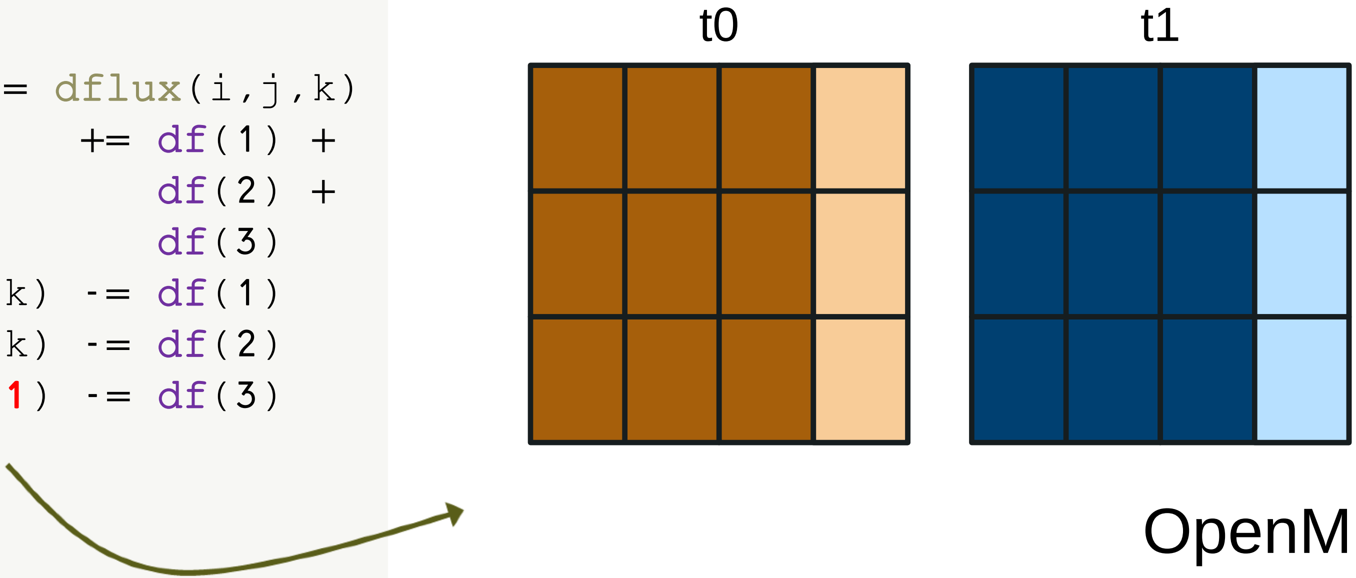
TACOMA - Algo. Parallelism

```
do k=1,n3
  do j=1,n2
    do i=1,n1
      df(1:3) = dflux(i,j,k)
      R(i,j,k)    += df(1) +
                     df(2) +
                     df(3)
      R(i-1,j,k) -= df(1)
      R(i,j-1,k) -= df(2)
      R(i,j,k-1) -= df(3)
    end do
  end do
end do
```

GE Global Research

TACOMA - Algo. Parallelism

```
do k=1,n3
  do j=1,n2
    do i=1,n1
      df(1:3) = dflux(i,j,k)
      R(i,j,k)    += df(1) +
                     df(2) +
                     df(3)
      R(i-1,j,k) -= df(1)
      R(i,j-1,k) -= df(2)
      R(i,j,k-1) -= df(3)
    end do
  end do
end do
```



OpenMP

GE Global Research

TACOMA - Algo. Parallelism

```
do k=1,n3
  do j=1,n2
    do i=1,n1
      df(1:3) = dflux(i,j,k)
      R(i,j,k)    += df(1) +
                     df(2) +
                     df(3)
      R(i-1,j,k) -= df(1)
      R(i,j-1,k) -= df(2)
      R(i,j,k-1) -= df(3)
    end do
  end do
end do
```

```
do k=ts3,tn3
  do j=ts2,tn2
    do i=ts1,tn1
      df(1:3) = dflux(i,j,k)
      if mycolor(i,j,k,tid)
        R(i,j,k)    += df(1) +
                       df(2) +
                       df(3)
      if mycolor(i-1,j,k,tid)
        R(i-1,j,k) -= df(1)
   end do
  end do
end do
```

OpenMP

TACOMA - Algo. Parallelism

GE Global Research

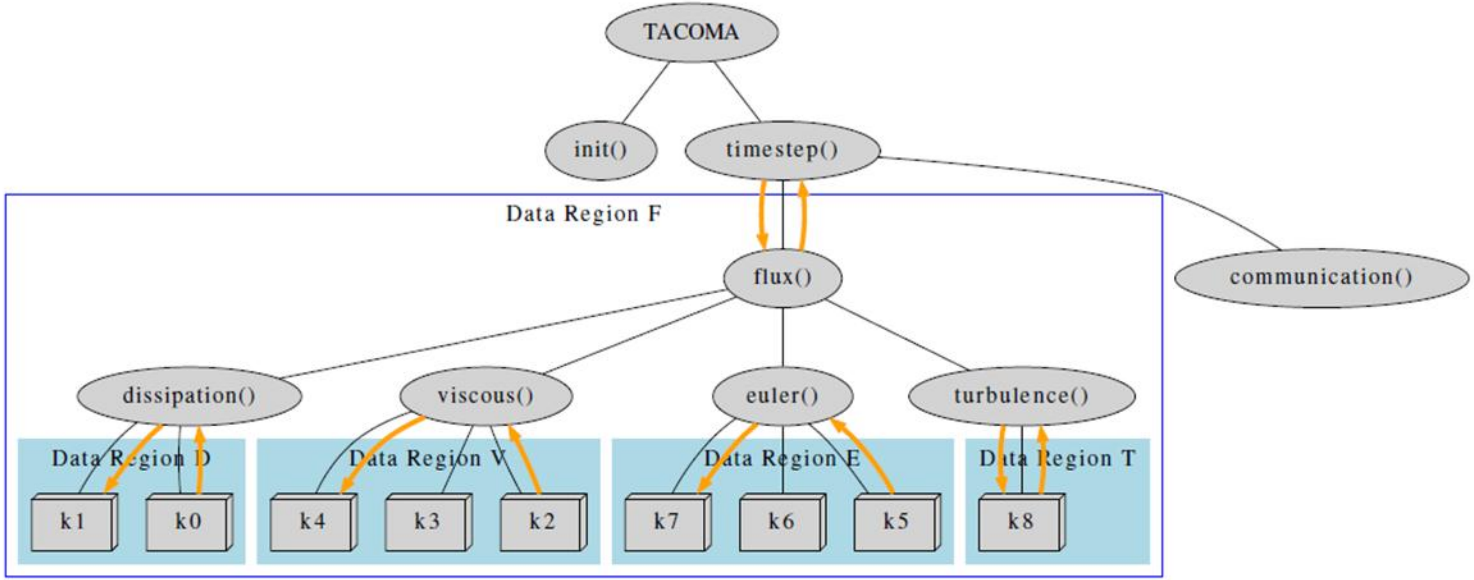
```
do k=1,n3
  do j=1,n2
    do i=1,n1
      df(1:3) = dflux(i,j,k)
      R(i,j,k)    += df(1) +
                     df(2) +
                     df(3)
      R(i-1,j,k) -= df(1)
      R(i,j-1,k) -= df(2)
      R(i,j,k-1) -= df(3)
    end do
  end do
end do
```

```
do
  df(i,j,k,1:3) = dflux(i,j,k)
end do

do
  R(i,j,k) += df(i,j,k,1) +
              df(i,j,k,2) +
              df(i,j,k,3)
  R(i,j,k) -= df(i+1,j,k,1) +
              df(i,j+1,k,2) +
              df(i,j,k+1,3)
end do
```

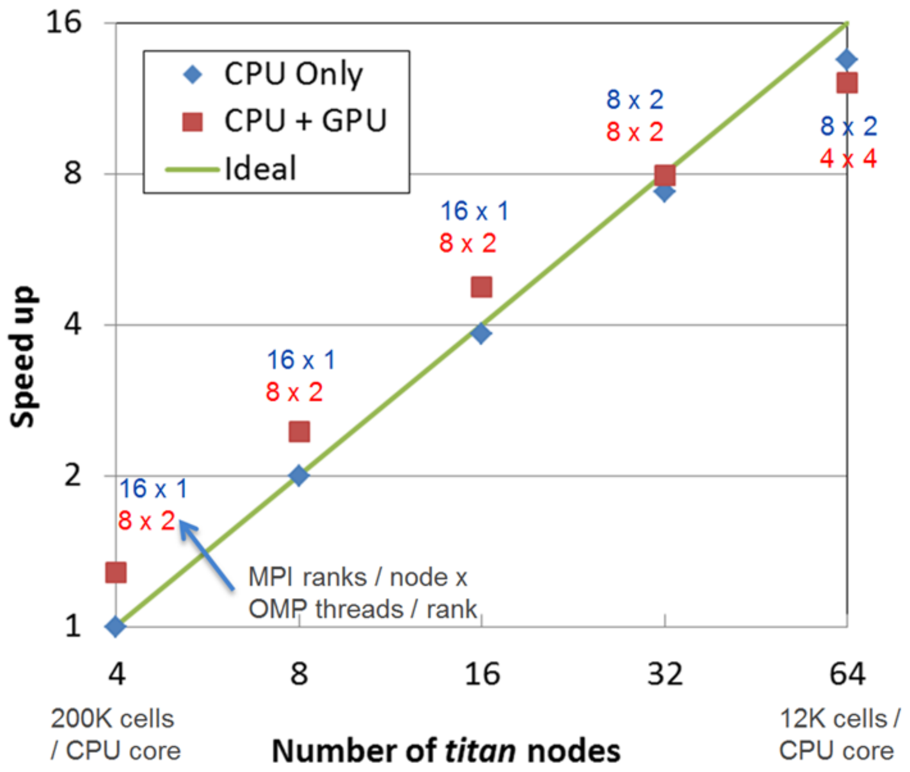
OpenACC

TACOMA - OpenACC Data



- **Create OpenACC data regions:**
  - Keep data on the GPU device as long as possible.
  - Create data regions in bottom-up, hierarchical fashion.

TACOMA - Performance

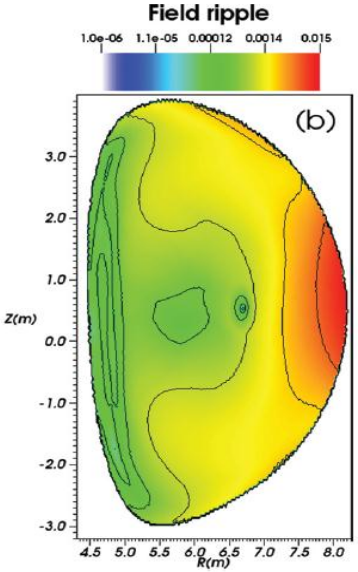


Porting - Delta5D

Case Study II: Delta5D

Porting - Delta5D

- From ORNL's Donald Spong.
- Monte-Carlo fusion code.
- Boozer space particle orbits.
- Hamiltonian guiding center equations solved with 4th order Runge Kutta.



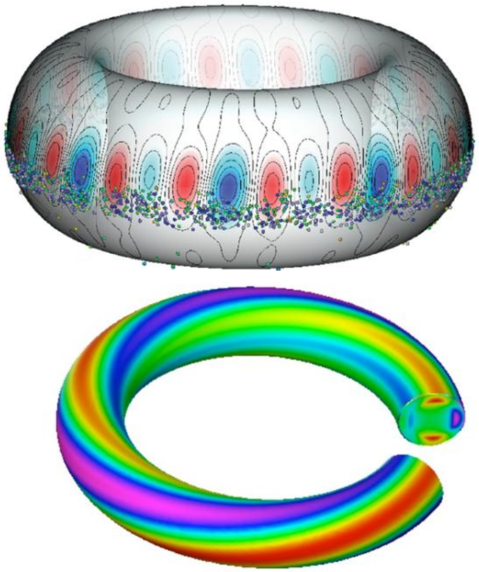


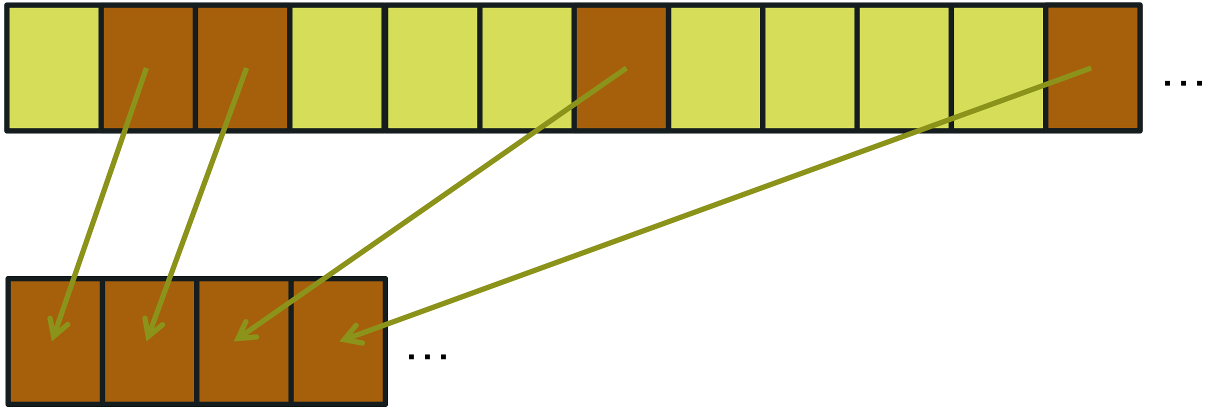
Figure 2 – TAE eigenmode structure projected onto an outer flux surface.

Porting - Delta5D

- **Example loop from Delta5D.**
  - Fast enough to run in serial on CPU; slow on GPU.
  - Data motion rules out running on CPU.
  - Needs to run in parallel on GPU.

Delta5D - Algo. Parallelism

- If a particle's trajectory takes it outside the confined plasma volume, append it to a list of escaped particles:



Delta5D - Algo. Parallelism

```
do i=1,maxorb
   ! -- Record this particle if it has "escaped".
   if(psinor(i) .gt. 1.) then
      iloss = iloss + 1
      phi_loss(iloss)   = y(6*i-3)
      psi_loss(iloss)   = y(6*i-4)/psimax
      thet_loss(iloss)  = y(6*i-5)
      elost = elost + hkin(i)/ejoule
   end if
end do
```

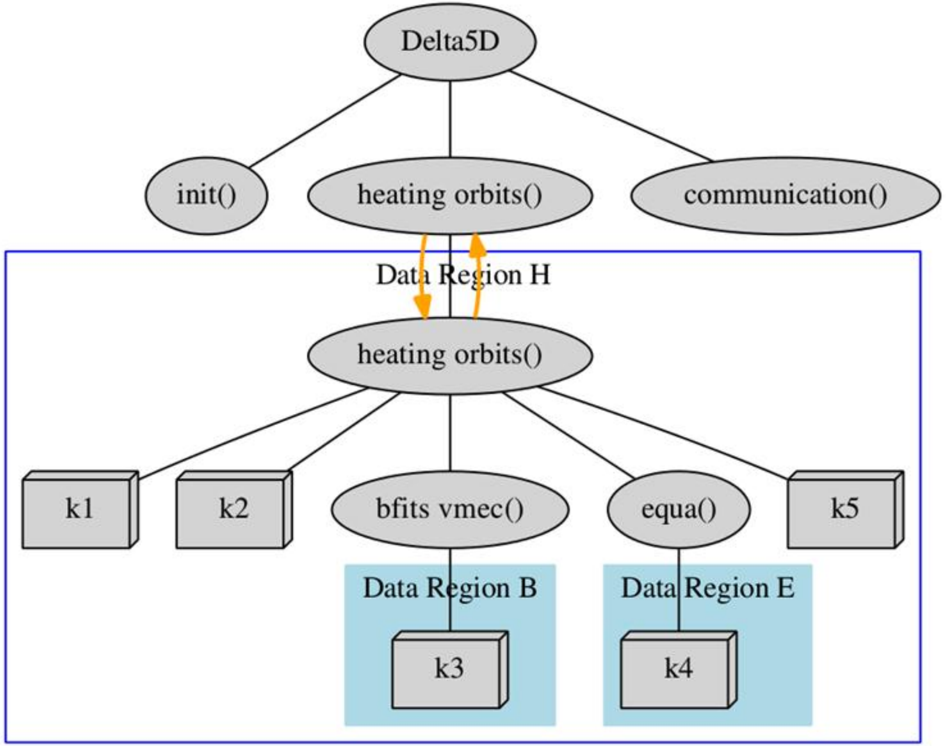
v1

Delta5D - Algo. Parallelism

v2

```
do i=1,maxorb
   ! -- Record this particle if it has "escaped".
   if(psinor(i) .gt. 1.) then
!$acc atomic capture
      iloss    = iloss + 1                 ! update-statement
      my_iloss = iloss                     ! capture-statement
!$acc end atomic
      phi_loss(my_iloss)   = y(6*i-3)
      psi_loss(my_iloss)   = y(6*i-4)/psimax
      thet_loss(my_iloss)  = y(6*i-5)
      elost = elost + hkin(i)/ejoule
   end if
end do
```

Delta5D - OpenACC Data



Delta5D - OpenACC Performance

OpenACC Sequential	OpenACC Atomics
19.446s	0.425s

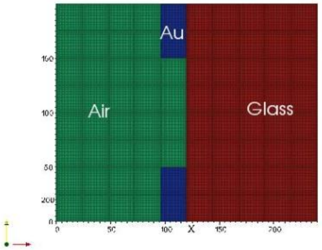
- 45x kernel speedup.
- Up to ~5-10% improvement in total runtime.

Porting - NekCEM

Case Study III: NekCEM

Porting - NekCEM

- From ANL's Mi Sun Min.
- Nekton for Computational Electro Magnetics.
- High-fidelity electro-magnetics solver based on spectral element methods.
- Written in Fortran and C.



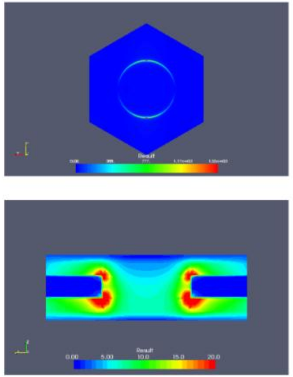


Figure 5. 3D nanohole with radius of 50 nm.

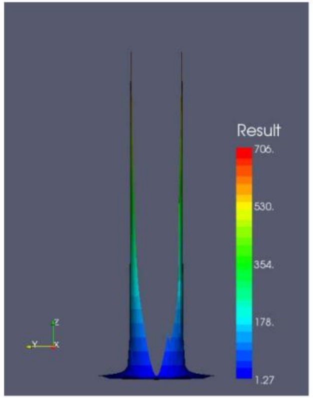


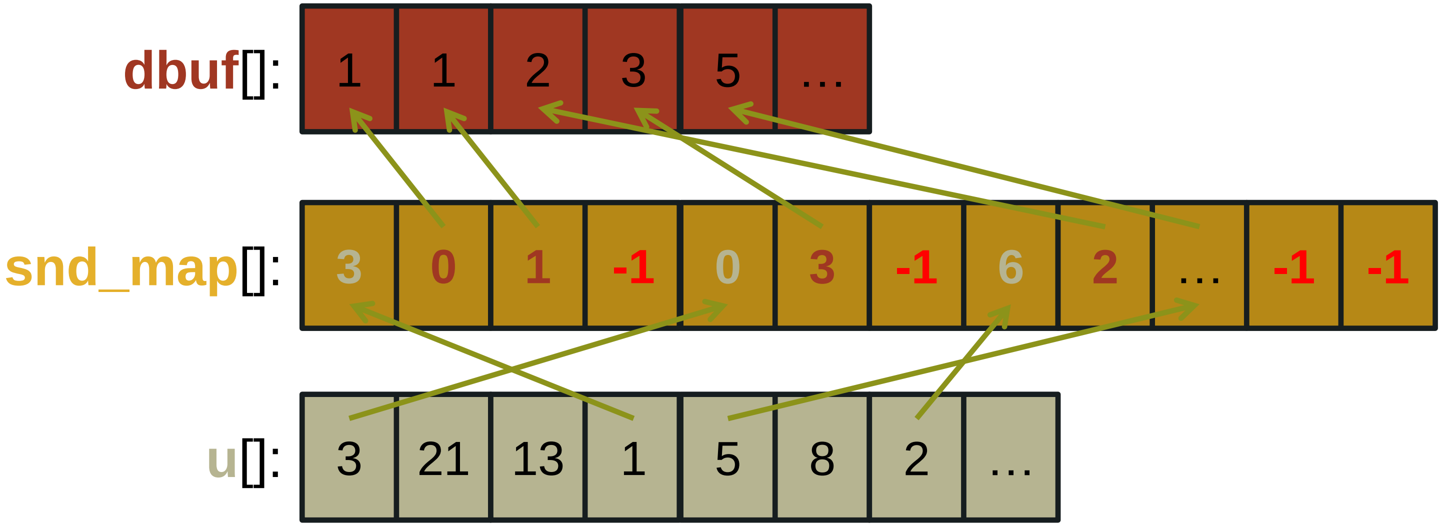
Figure 6. 3D nanohole with radius of 50 nm.

Porting - NekCEM

- **Example loop from NekCEM.**
  - Initial loop structure does not vectorize on GPU.
  - Gather/scatter benefits from high GPU bandwidth.
  - Data motion needed around MPI communication.

NekCEM - Algo. Parallelism

- Scatter from u to dbuf with indirect addressing using description vector snd_map internally terminated by -1.



NekCEM - Algo. Parallelism

Argonne NATIONAL LABORATORY

CRAY

v1

```
for(k=0; k<vn; k++) {
  l_map = snd_map;
  while( (i=*l_map++) != -1 ) {
    t = u[i+k*dstride];
    j = *l_map++;
    do {
      dbuf[j*vn] = t;
    } while( (j=*l_map++) != -1 );
  }
  dbuf++;
}
```

NekCEM - Algo. Parallelism

```
for(k=0; k<vn; k++) {
  for(i=0; snd_map[i]!=-1; i=j+1){
    for(j=i+1; snd_map[j]!=-1; j++){
      dbuf[k+snd_map[j]*vn] = u[snd_map[i]+k*dstride];
    }
  }
}
```

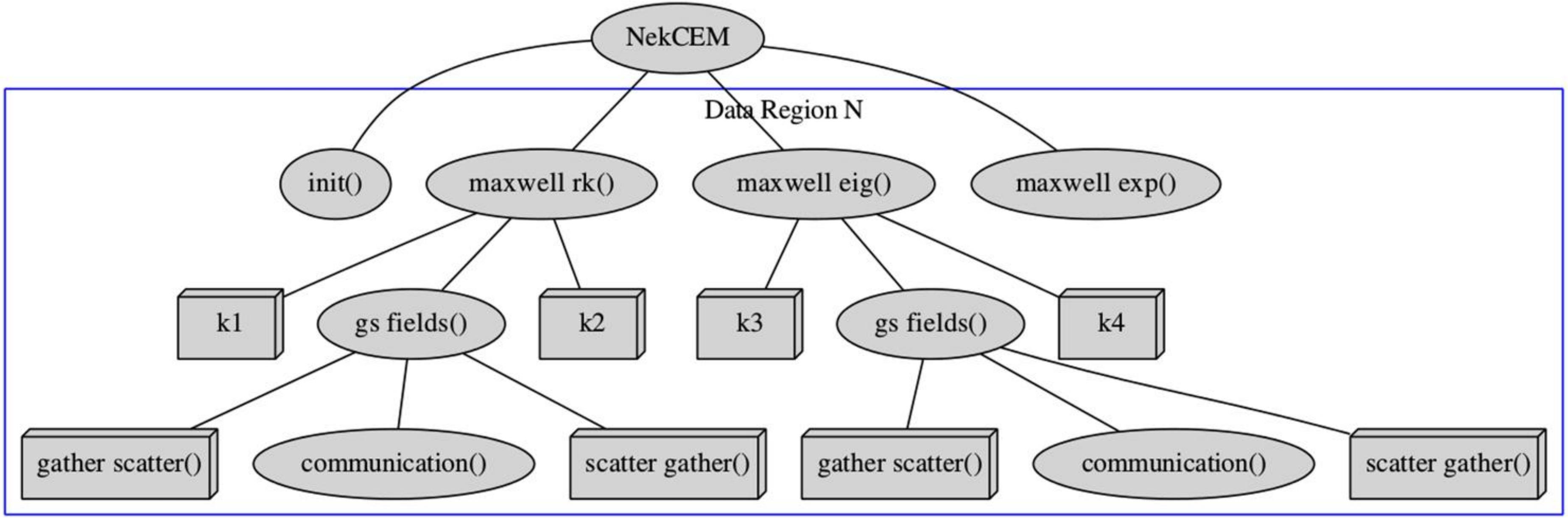
v2

NekCEM - Algo. Parallelism

```
for(k=0; k<vn; k++) {
  for(i=0; i<snd_m_nt; i++){
    for(j=0; j<snd_mapf[i*2+1]; j++) {
      dbuf[k+snd_map[snd_mapf[i*2]+j+1]*vn] =
         u[snd_map[snd_mapf[i*2]]+k*dstride];
    }
  }
}
```

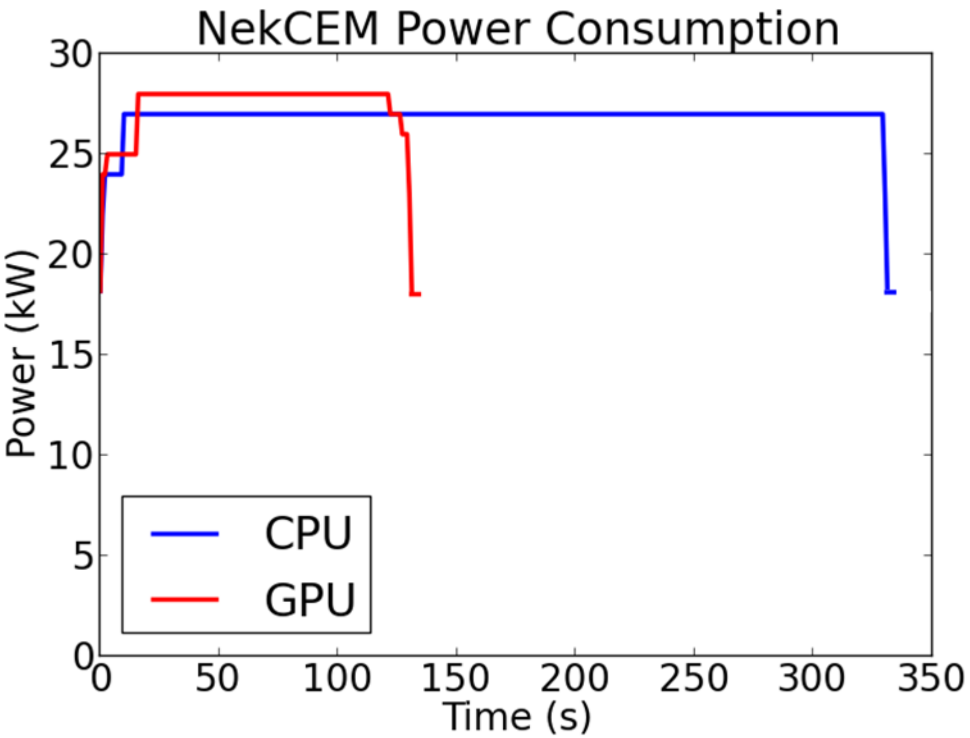
v3

NekCEM - OpenACC Data

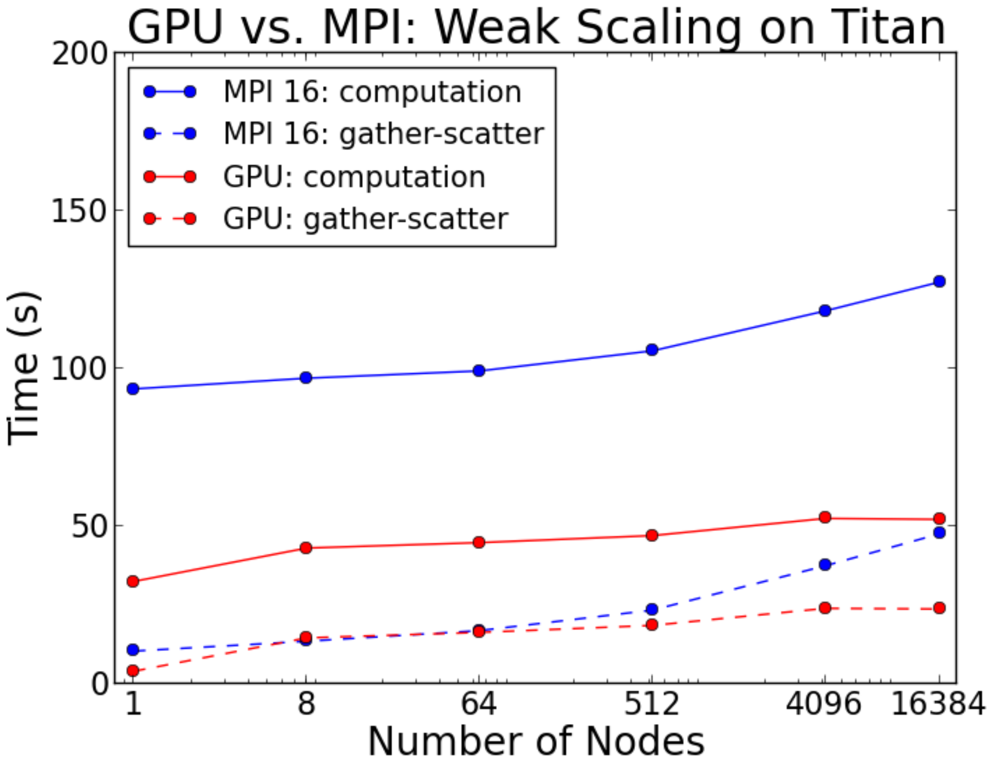


NekCEM - Performance

GPU run uses 39% the total energy of the CPU run!



NekCEM - Performance



Porting - Conclusion

- **Lessons learned:**
  - Port to OpenMP before OpenACC.
    - Reuse scoping work.
  - Optimize OpenACC data motion last.
    - Perform bottom-up, hierarchical data optimization.
  - Express underlying algorithm’s parallelism.
    - Don’t limit parallelism by existing implementation.

Porting - Legal

- **Contact Information**: Aaron Vose -- email: avose@cray.com -- phone: (865) 574-8140.

- **Acknowledgements**: This research used resources of the OLCF at ORNL, which is supported by the Office of Science of the U.S. DoE under Contract No. DE-AC05-00OR22725, as well as resources of the ALCF at ANL, which is supported by the Office of Science of the U.S. DoE under Contract No. DE-AC02-06CH11357. Also, many thanks to Brian Mitchell from GE for the TACOMA code, Donald Spong from ORNL for the Delta5D code, and Mi Sun Min from ANL for the NekCEM code.

-

-