



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

Aaron Vose - Cray Inc.
GTC - 03/19/2015

Porting - Overview

- **Porting methodology:**

- Express underlying algorithmic parallelism.
 - Port to OpenMP first.
 - Port to OpenACC second.

- **Case studies / examples:**

- TACOMA



GE Global Research

- Delta5D



- NekCEM



Porting - Overview

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



GE Global Research



- 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

CRAY®

Case Study I: TACOMA



GE Global Research

COMPUTE

| STORE |

ANALYZE

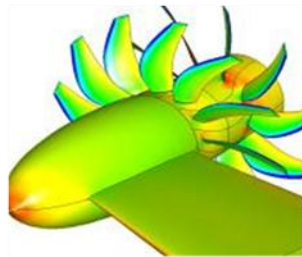
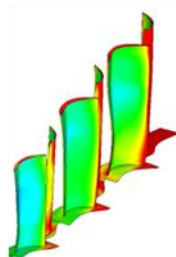
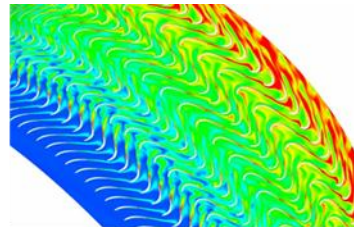
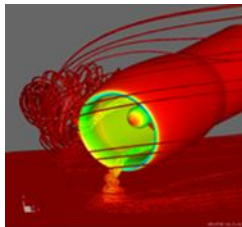
Copyright 2015 Cray Inc. & GE



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.

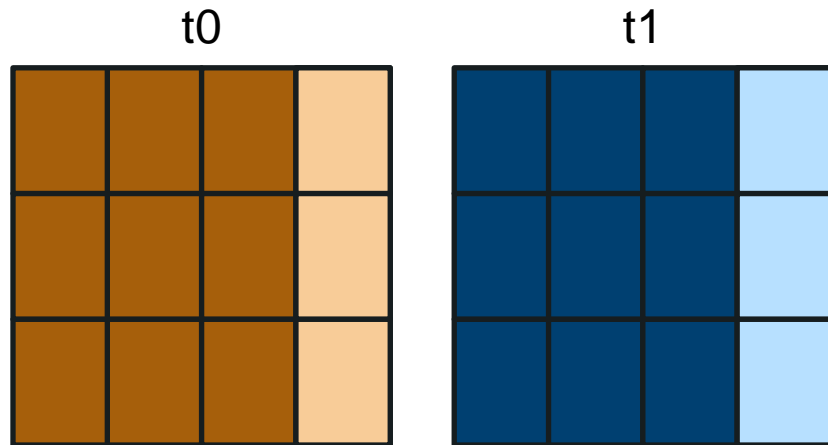



```
do k=1,n3
  do j=1,n2
    do i=1,n1
      df(1:3) = dflux(i,j,k)
      R(i,j,k) += df(1) +
                df(2) +
                df(3)
      R(i-1,j,k) -= df(1)
      R(i,j-1,k) -= df(2)
      R(i,j,k-1) -= df(3)
    end do
  end do
end do
```



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



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
end do
```

OpenMP



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
```

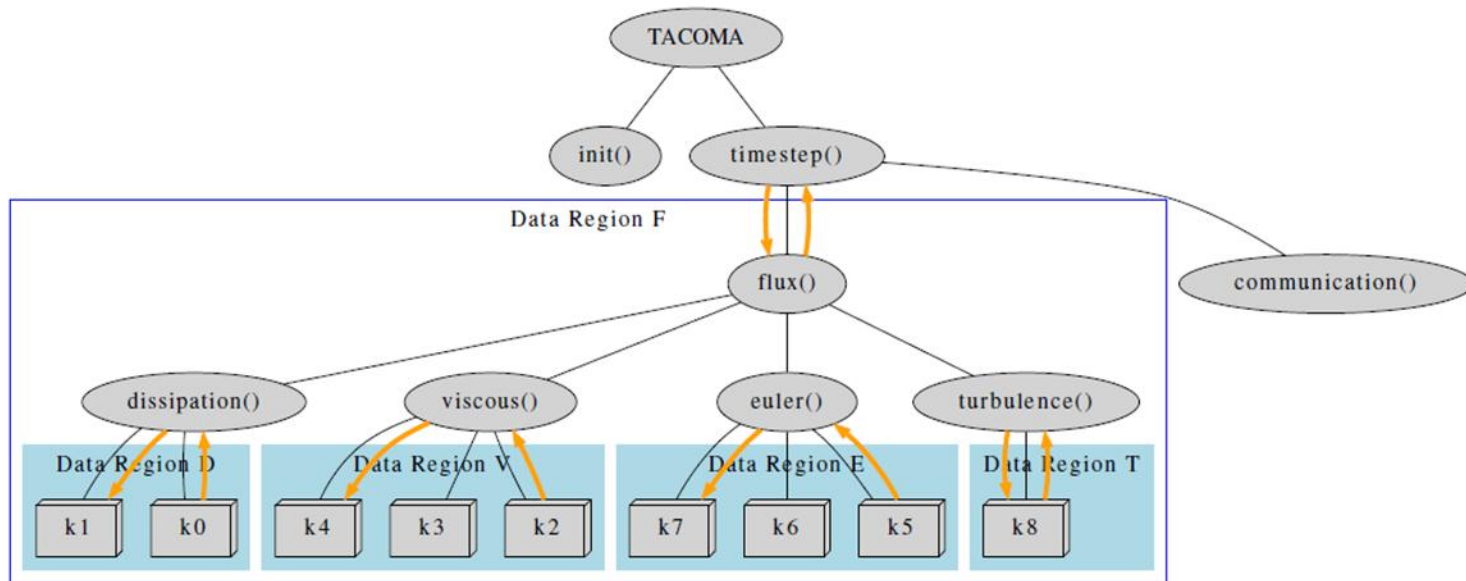
```
do
  df(i,j,k,1:3) = dflux(i,j,k)
end do

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



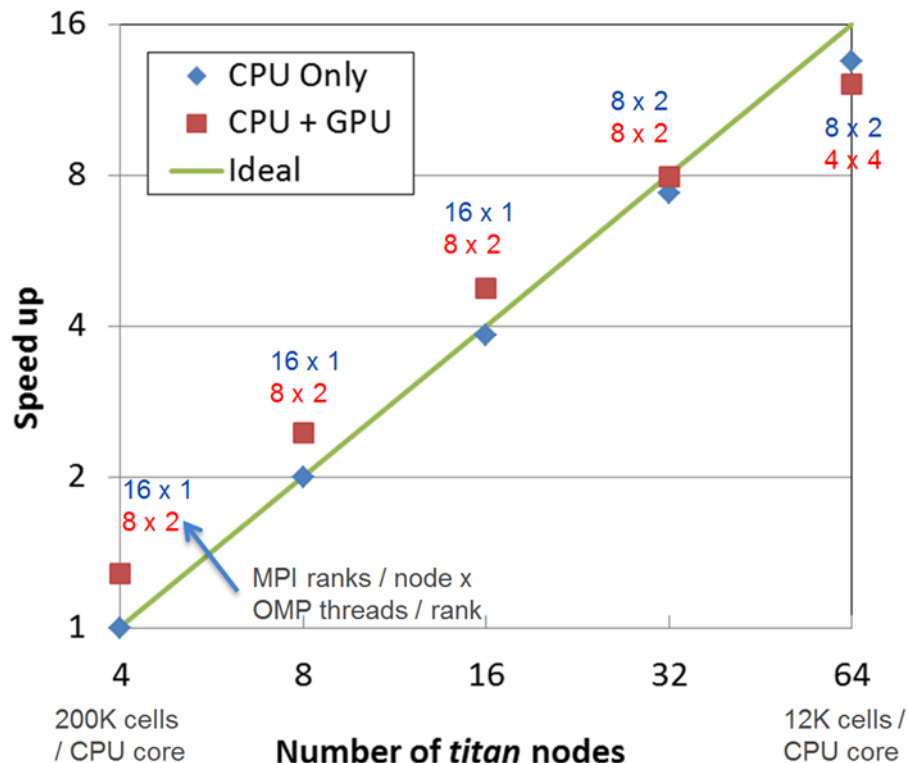
TACOMA - OpenACC Data



- **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

The Cray logo is located in the top right corner. It consists of the word "CRAY" in a blue, sans-serif font, followed by a registered trademark symbol (®). To the right of the text is a decorative graphic of a grid of small circles, some of which are colored in red, blue, and yellow.

Case Study II: Delta5D



COMPUTE

| STORE

| ANALYZE

Copyright 2015 Cray Inc. & GE

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

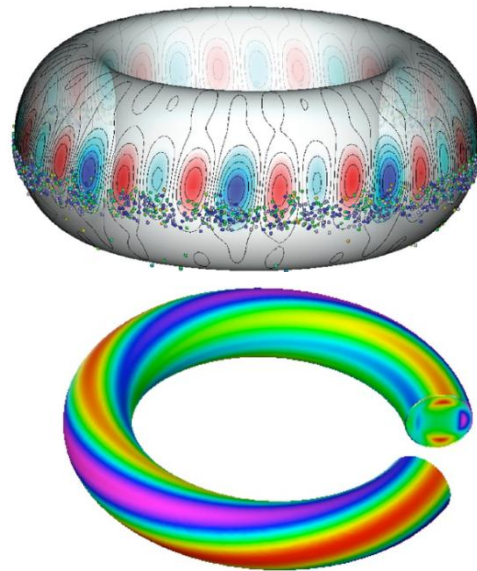
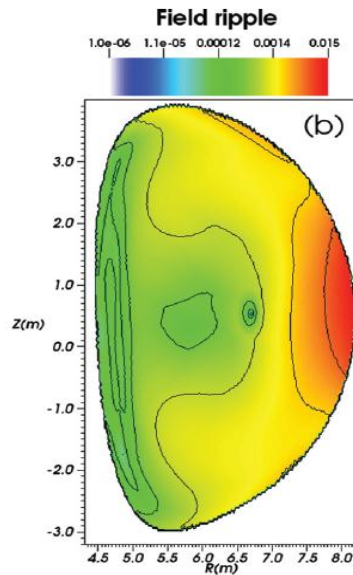


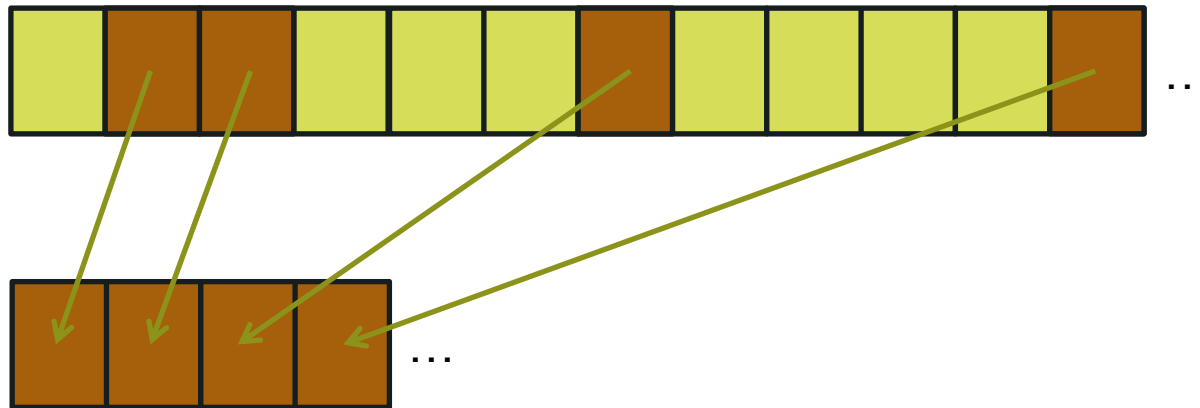
Figure 2 – TAE eigenmode structure projected onto an outer flux surface.

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.

- 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
```

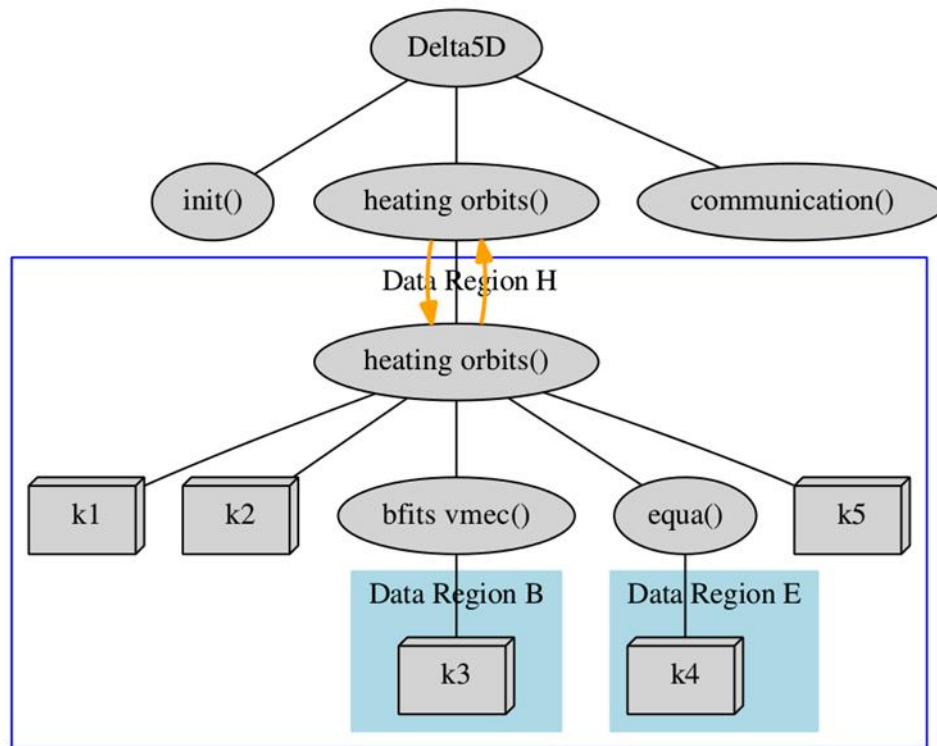
v1

Delta5D - Algo. Parallelism

```
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
```

v2

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



- 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.

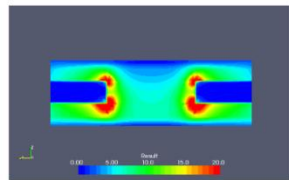
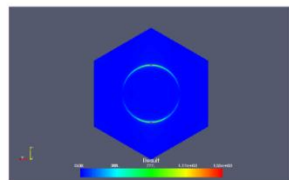
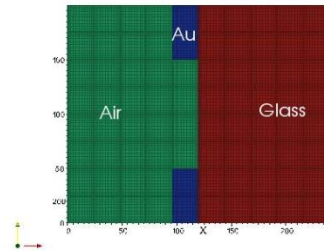


Figure 5. 3D nanohole with radius of 50 nm.

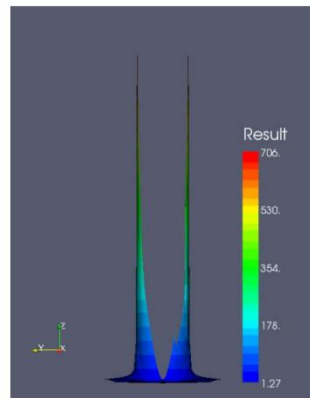
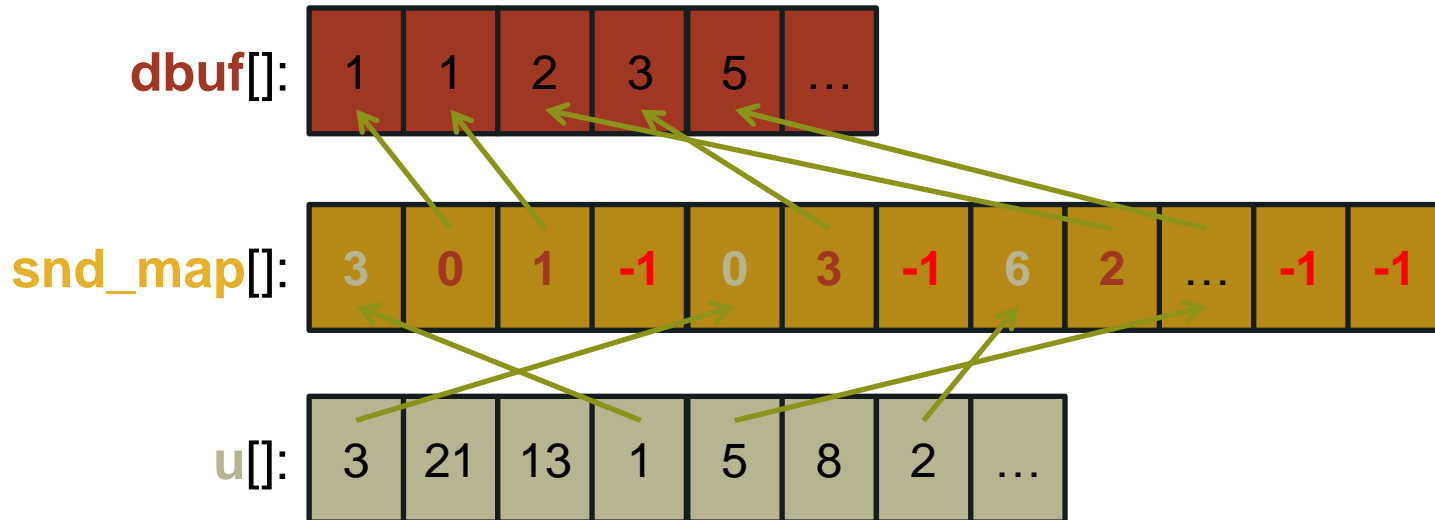


Figure 6. 3D nanohole with radius of 50 nm.



- **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.

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



```
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++;  
}
```

v1

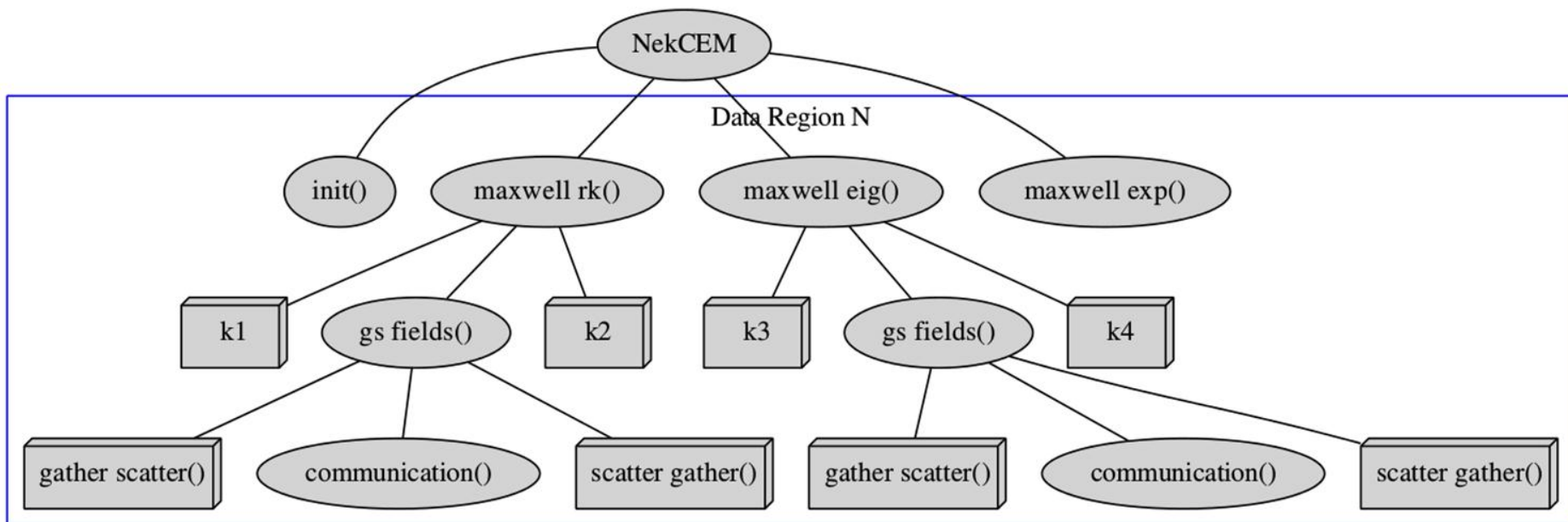
```
for(k=0; k<vn; k++) {  
  for(i=0; snd_map[i]!=-1; i=j+1){  
    for(j=i+1; snd_map[j]!=-1; j++){  
      dbuf[k+snd_map[j]*vn] = u[snd_map[i]+k*dstride];  
    }  
  }  
}
```

v2

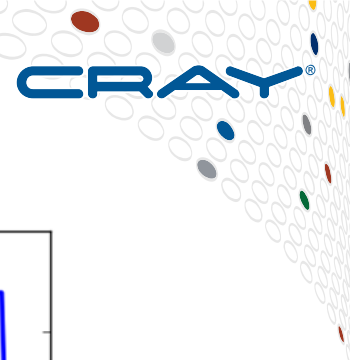
```
for(k=0; k<vn; k++) {
  for(i=0; i<snd_m_nt; i++){
    for(j=0; j<snd_mapf[i*2+1]; j++) {
      dbuf[k+snd_map[snd_mapf[i*2]+j+1]*vn] =
        u[snd_map[snd_mapf[i*2]]+k*dstride];
    }
  }
}
```

v3

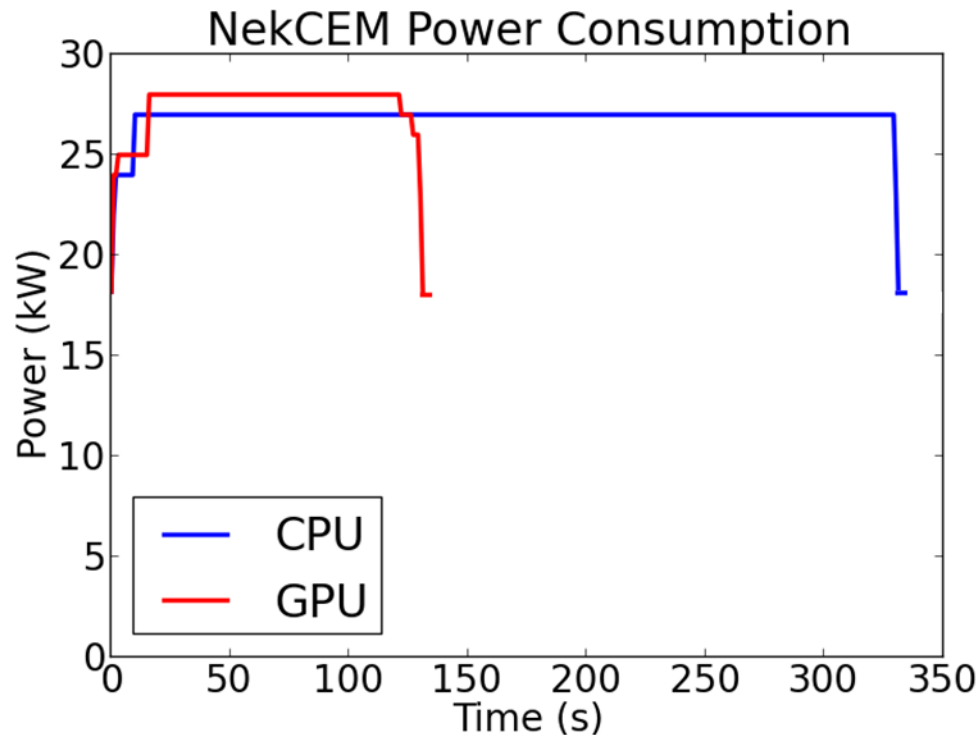
NekCEM - OpenACC Data



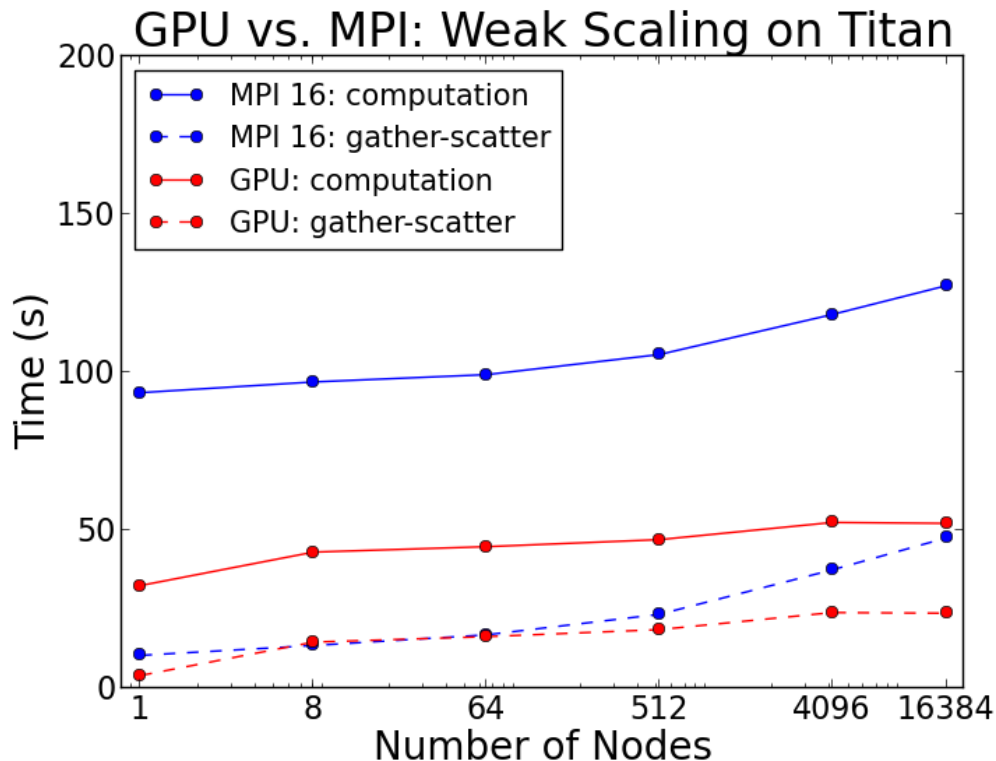
NekCEM - Performance



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.
- **Safe Harbor Statement:** This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.
- **Legal Disclaimer:** Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document. Cray Inc. may make changes to specifications and product descriptions at any time, without notice. All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice. Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request. Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user. Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, URIKA, and YARCDATA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.