Microscopic performance analysis using Extrae

John Dennis (dennis@ucar.edu)
How to Identify poorly performing code?

- Automatic performance identification
  - Barcelona Supercomputer Center (BSC)
  - Polytechnic University of Catalonia (UPC)
  - H. Servat, J. Labarta, J. Gimenez
- Utilize BSC tools
  - extrae: trace collection
  - paraver: visualization client
  - clustering & folding tools
- Enables very detailed tracing of application characteristics
- Creates a performance database
CAM5–SE: 2 degree (ne=16) on 384 cores
Computational burst clusters

Most expensive computational cluster
Total Instructions: Cluster 1

Task 22 Thread 1 - Cluster_1.0
Duration = 185.03 ms Counter = 584457.44 Kevents
Underperforming subroutines
Cluster 1

- **Group A:**
  - conden: 2.7%
  - compute_uwshcu: 3.3%
  - rtrnmc: 1.75%

- **Group B:**
  - micro_mg_tend: 1.36% (1.73%)
  - wetdepa_v2: 2.5%

- **Group C:**
  - reftra_sw: 1.71%
  - spcvmc_sw: 1.21%
  - vrtqdr_sw: 1.43%

Focus effort on one subroutine
wetdepa_v2 (CESM)

- CESM B-case, NE=16, 570 cores
- Yellowstone, Intel (13.1.1) -O2
- Original version:
  - 2.5% total time
  - 492.6 ms
- Modified version:
  - 0.73% total time
  - 121.1 ms
- Actual improvement: 4.07x
Vectorization Intensity Cluster #1

Increase in code vectorization
Stalls on Resources Cluster #1

Reduction in cycles stalled on resources
Conclusions

- Significant performance improvement possible for (SNB,MIC) architectures
- Possible to identify poorly performing code in CESM
- Possible to significantly increase performance through vectorization: 4x (SNB) – 9x (MIC)
- Next procurement question:
  - $35M on hardware
  - or
  - $32M on hardware + $3M optimizing code