Communication and Vectorization within HPGMG

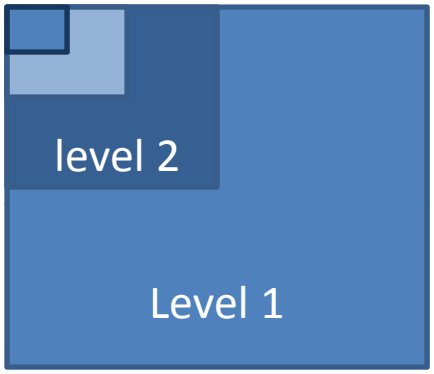
Vladimir Marjanovic

HPGMG in EU Project

- Mont Blanc 3 – hardware / software co-design
- HPGMG representative app used in app WP of MB3
- Profiling : Communication and Vectorization
  - Bandwidth (B/s)
  - Latency (s)
  - Message Injection Rate(1/s)

HPGMG Code

- Well written balanced codes MPI+OpenMP
- HPCG,HPGMG complexity:
  computational O(n) communication $O(n^{2/3})$
- MG many levels

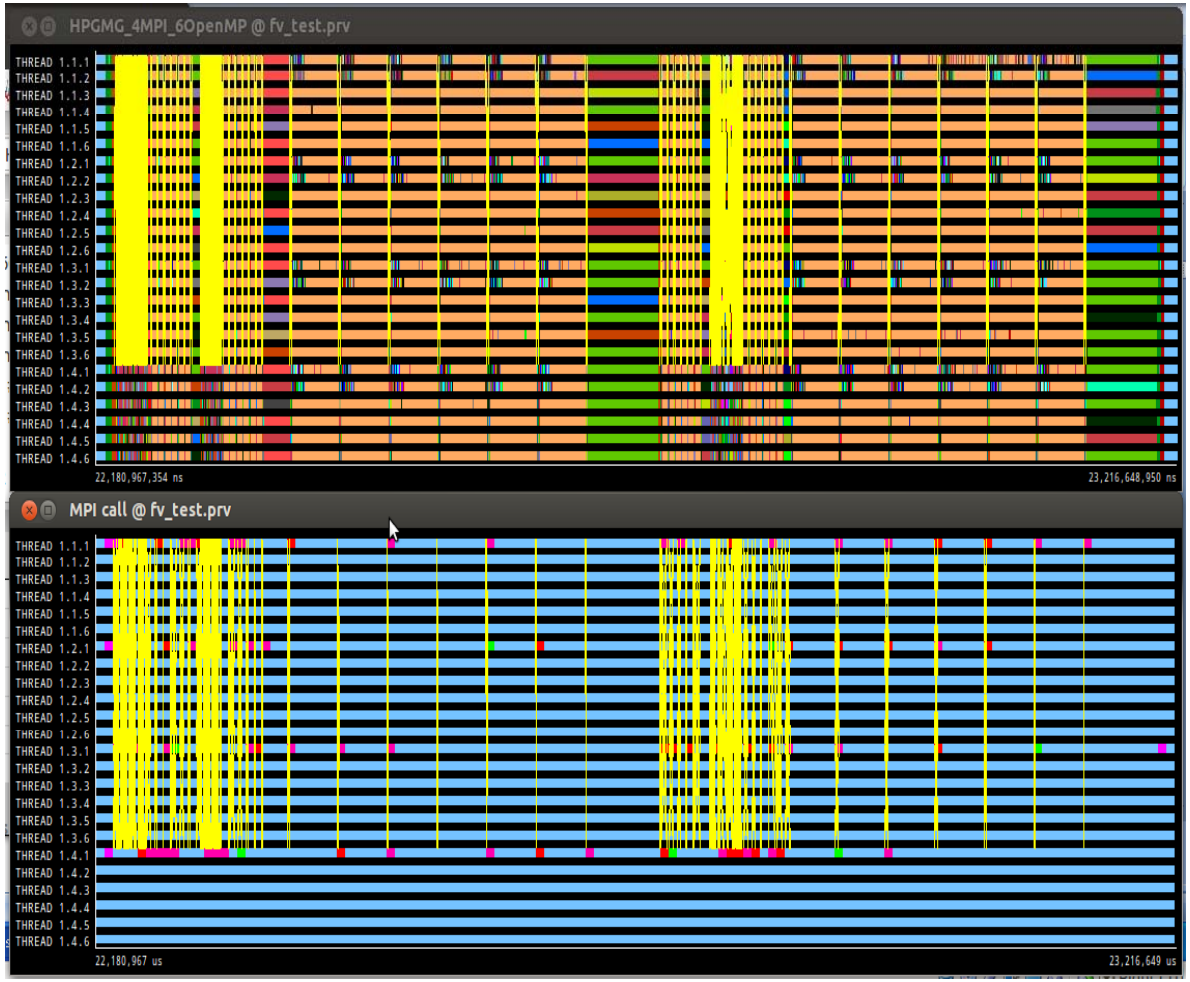


- mpirun ./hpgmg-fv SIZE_OF_BLOCK BLOCKs_PER_MPI

Perfect MPI data distribution :

(NUM_OF_MPIs * BLOCKs_PER_MPI) $^{1/3}$ **= INTEGER_NUMBER**

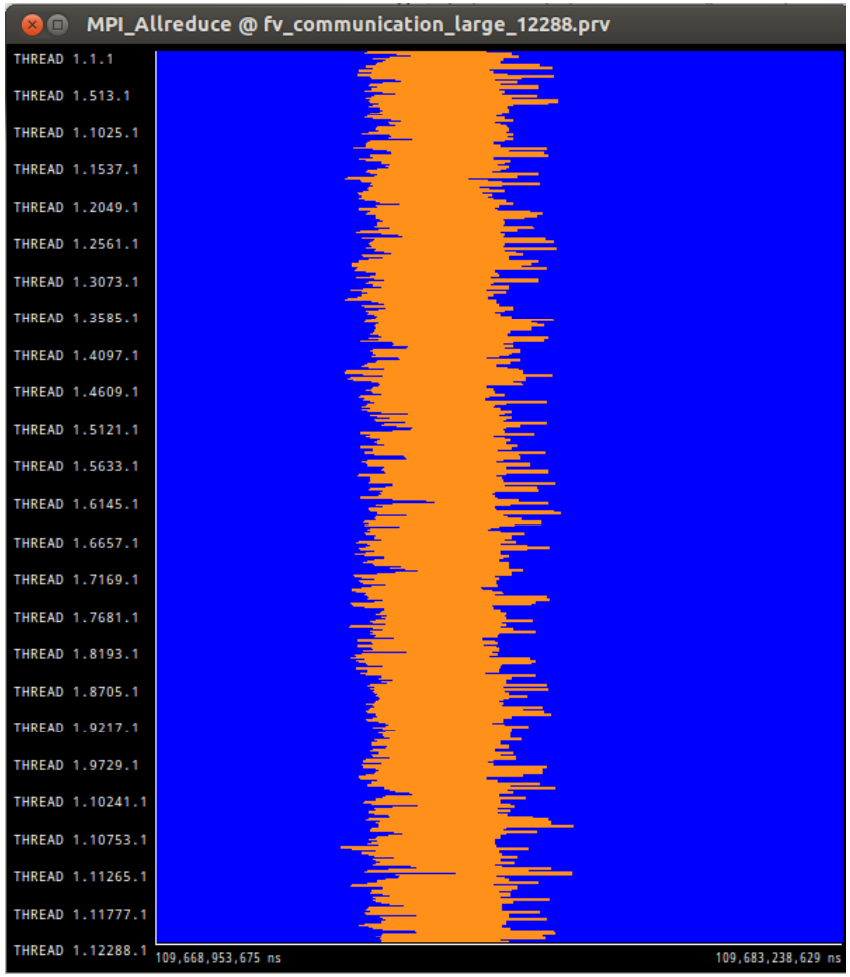
Communication Profiling: Tracing



- Trace 4MPI + 6OpenMP
- 25 OpenMP section
- MG – V cycle

- MPI:
  - MPI_Allreduce
  - ExchangeHalos

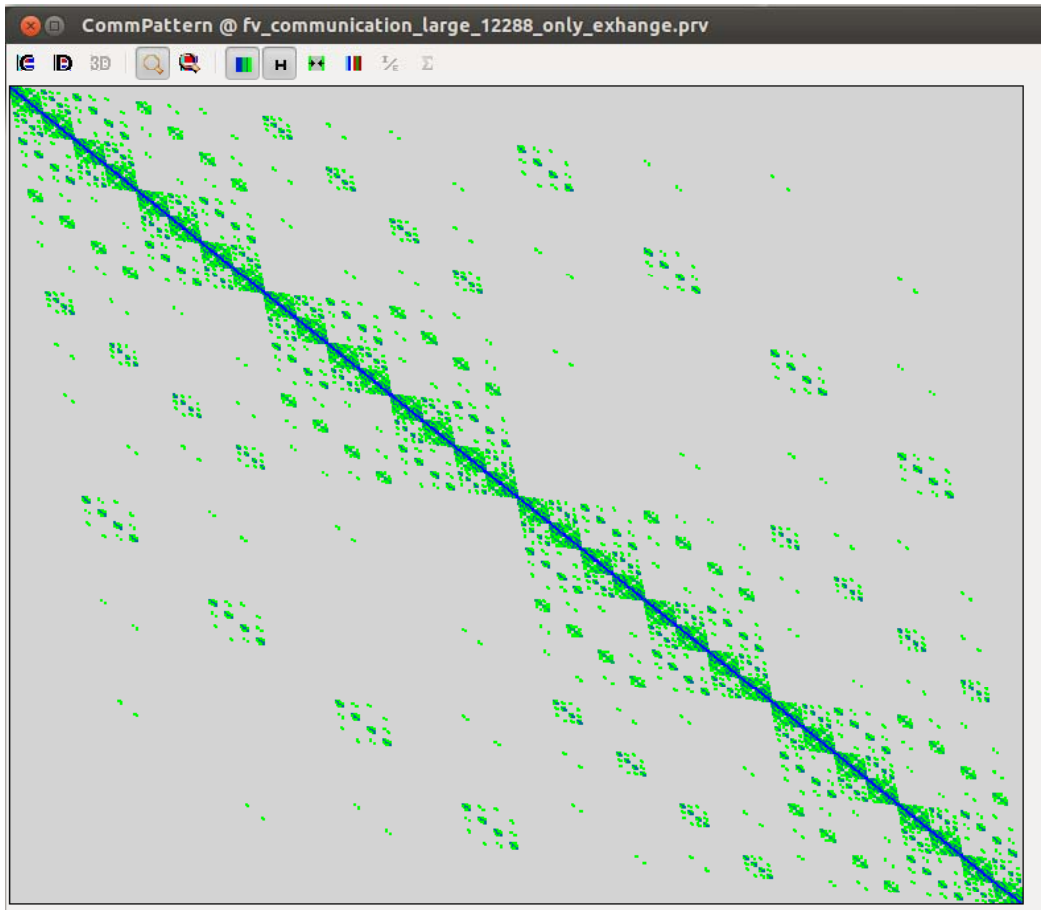
MPI_Allreduce 12288 MPIs



- XC40 - Intel Haswell E5-2680v3
- 24 cores per node
- HPGMG input 6 9
- 24*512 * 9 = 48*48*48 perfect cube

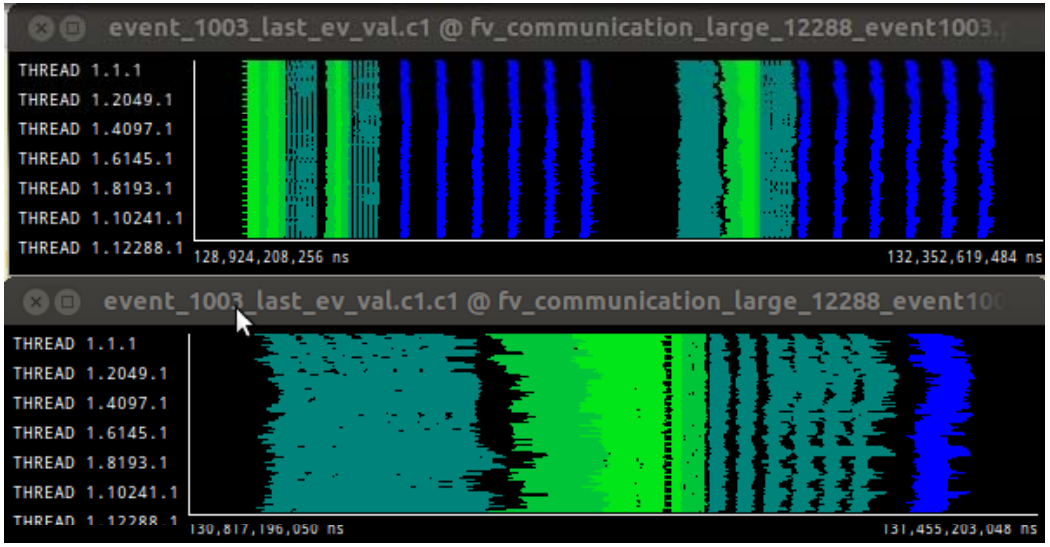
Average	2453728ns
Max	3907503 ns
Min	34976 ns
StDev	508181 ns
Avg/Max	0.63

Comm Pattern: ExchangeHalos 12288 MPIs



Max(bytes)	624640
Min(bytes)	2048
MaxNumOfNeighbor	24

Unbalancing: ExchangeHalos 12288 MPIs



2^6	12	All
2^5	24	All
2^4	26	All
2^3	108-360	35-all
2^2	108	9
2	120	3

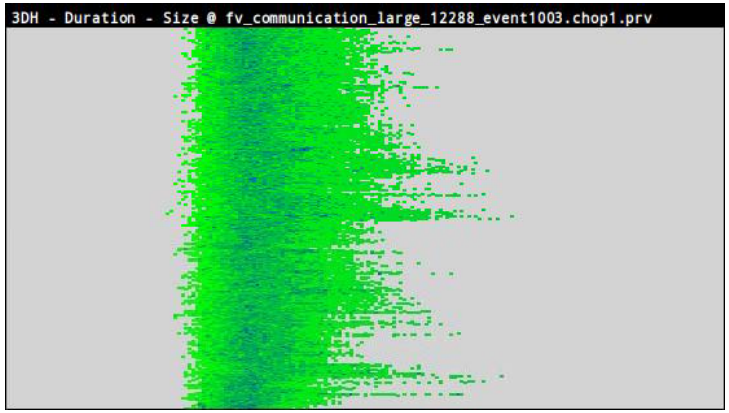
number of messages

Average	15514163ns
Max	20811766ns
Min	11503473ns
StDev	847420ns
Avg/Max	0.75

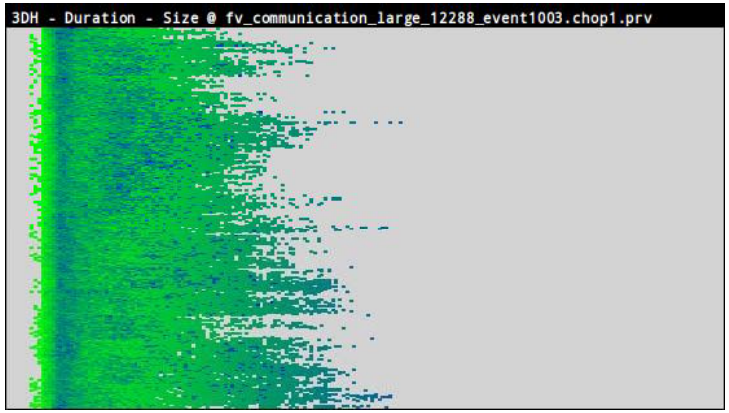
2^6 unbalancing issue ~ 25%

Exchange Halos: Histogram of time duration

2^6

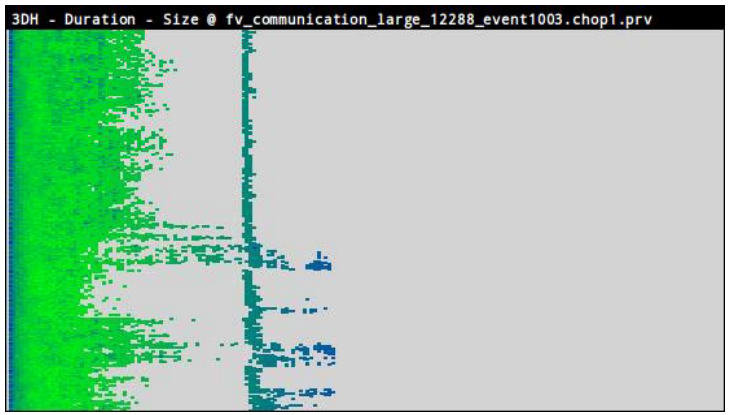


2^5



2^4

2^3



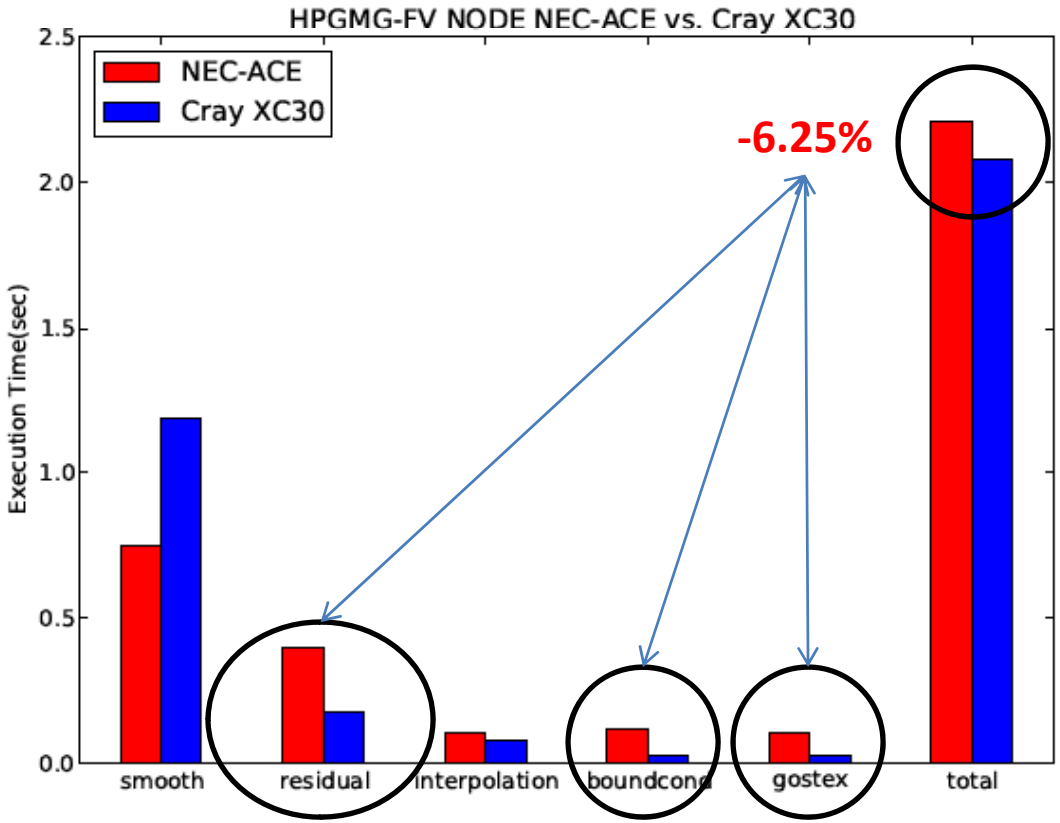
Conclusion: HPGMG Communication

- Large number of message.
- Stresses latency, routing
- Data distribution: perfect cube with input data
- Synchronization Issue

Vectorization of HPGMG : SX-ACE NEC

- SX – ACE NEC vector processor
- 4 cores for 256GFlops peak performance
- 1 GHz
- 16 vectore pipes per core
- 32 memory channels for 256GB/s bandwidth, one core or sharable
- Assignable Data Buffer (ADB) – Vector Cache 1MB

HPGMG-FV NEC-ACE versus CRAY XC30 no opt

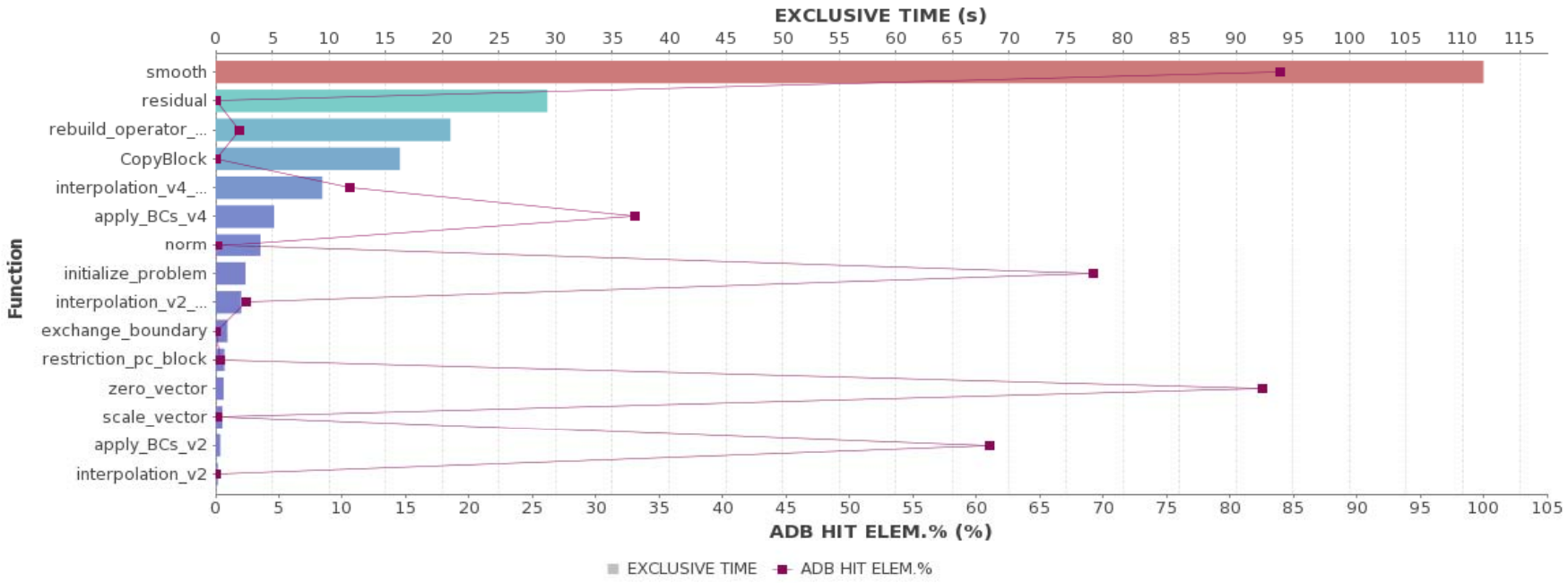


Ivy Bridge 5576

NEC SX-ACE

- Reference code
- no optimization
- Data distribution 8 2
- 4MPI vs 4MPI+6OpenM

NEC-ACE Profiler



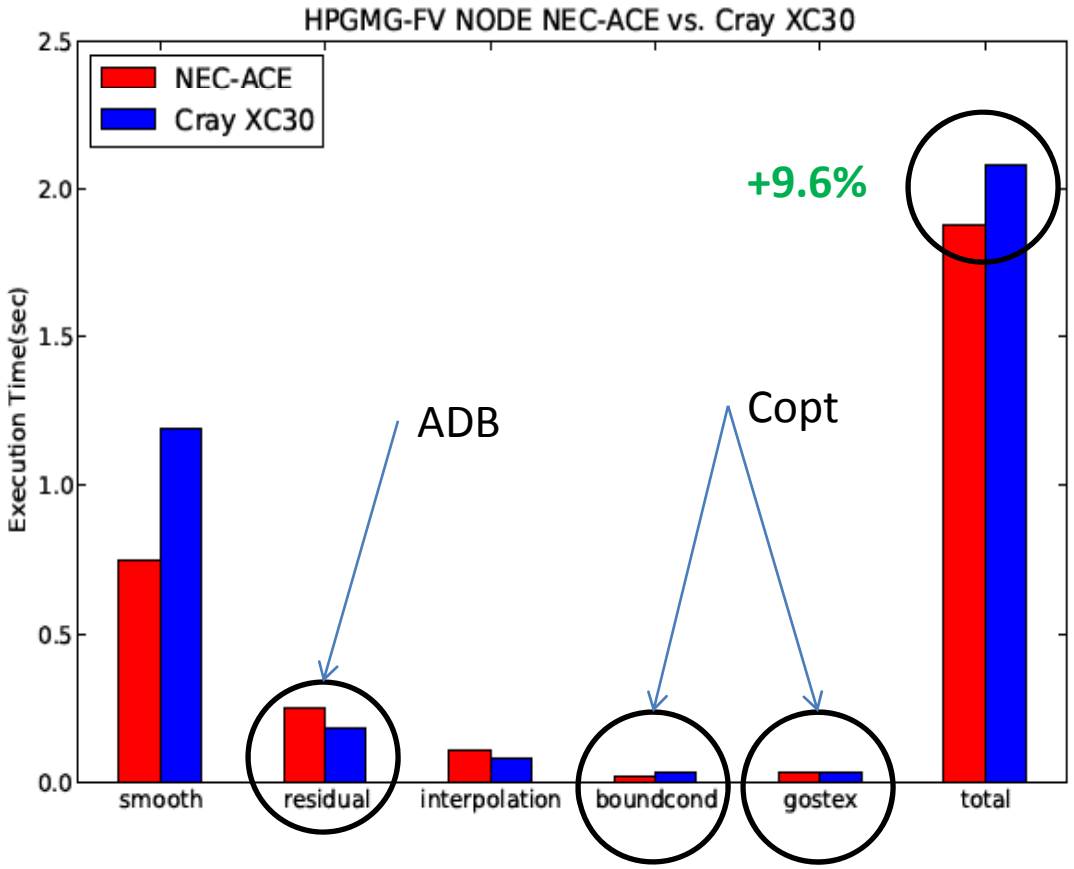
PROC.NAME	FREQUENCY	EXCLUSIVE TIME[sec](%)	AVER.TIME [msec]	MOPS	MFLOPS	V.OP RATIO	AVER. V.LEN	VECTOR TIME	I-CACHE MISS	O-CACHE MISS	BANK CONFLICT CPU PORT	BANK CONFLICT NETWORK	ADB HIT ELEM.%
smooth	25940	111.814(54.3)	4.311	34115.5	14634.4	97.67	58.6	108.605	0.173	0.928	0.002	31.051	83.93
residual	17588	29.260(14.2)	1.664	36589.9	16159.3	99.12	131.5	28.883	0.018	0.136	0.000	14.702	0.00

SX-ACE Code Optmization

```
//#pragma cdir on_adb(x)
    for(k=klo;k<khi;k++){
#pragma cdir on_adb(x)
    for(j=jlo;j<jhi;j++){
#pragma cdir on_adb(x)
    for(i=ilo;i<ihi;i++){
      int ijk = i + j*jStride + k*kStride;
      double Ax = apply_op_ijk(x);
      res[ijk] = rhs[ijk]-Ax;
    }}}
  }
```

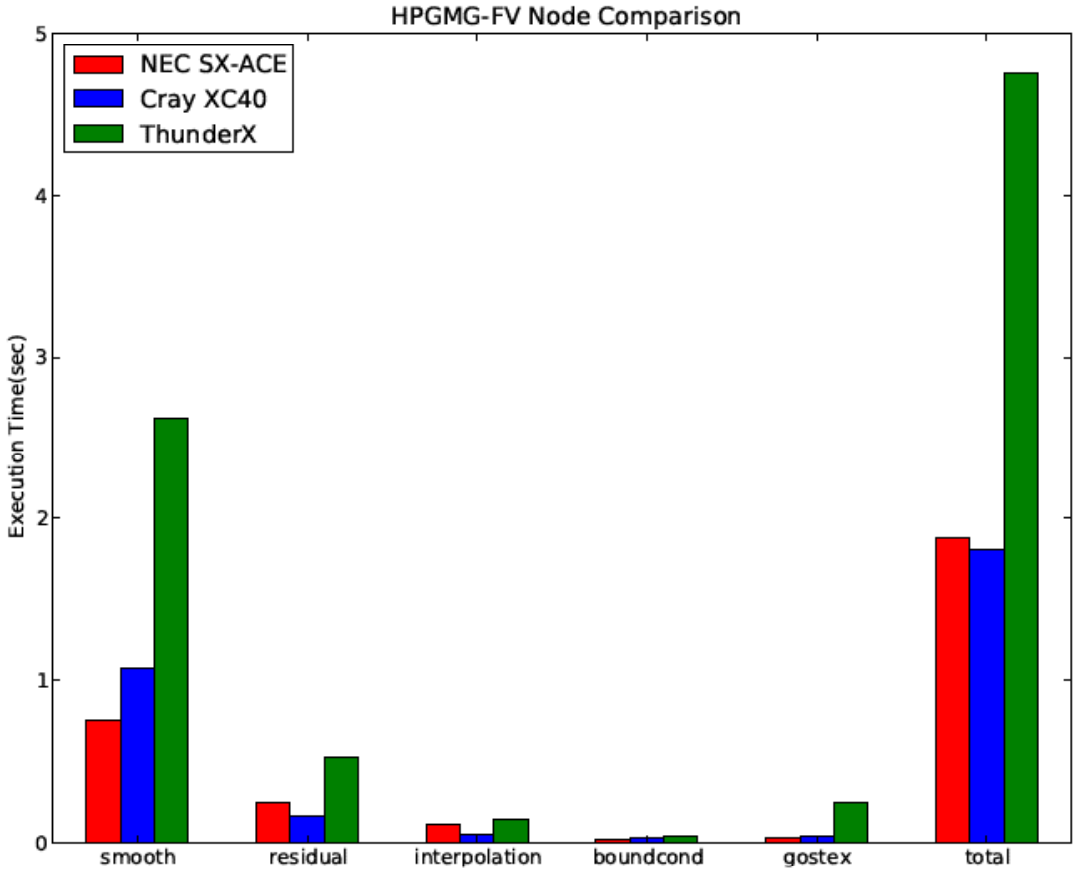
- Simple code modification
- Pragma ADB hint
- Keep data in vector cache

HPGMG-FV NEC-ACE versus CRAY XC30 with opt



- Copt
- use ADB
- 7 days
- NEC support
- 17% improvement

Node Comparison



- Memory BW
- Code Optimization
- Number of cores does not help
- Different performance for different routines

Conclusion: Vectorization

- Reference code very good well written for vectorization
- HPGMG stresses vector unit and compiler
- HPGMG shows performance issue on vectorziaon for small problem sizes