



# Performance of Density Functional Theory codes on Cray XE6

Zhengji Zhao, and Nicholas Wright  
National Energy Research Scientific  
Computing Center  
Lawrence Berkeley National  
Laboratory



National Energy Research  
Scientific Computing Center



Lawrence Berkeley  
National Laboratory



# Outline

- **Motivation**
- **Introduction to DFT codes**
- **Threads and performance of VASP**
- **OpenMP threads and performance of Quantum Espresso**
- **Conclusion**



## Motivation

- **Challenges from the multi-core trend**
  - Address reduced per core memory,
  - Make use of faster intra node memory access
- **Recommended path forward is to use threads/OpenMP**
- **Majority of the NERSC application codes are still in flat MPI**
- **Exam the performance implications from the use of threads in real user applications**



## Why DFT codes

- **Materials and Chemistry applications account for 1/3 of NERSC workflow.**
- **75% of them run various DFT codes.**
- **Among 500 application code instances at NERSC, VASP consumes the most computing cycles (~8%).**
- **VASP is in pure MPI, current status of majority user codes**
- **Quantum Espresso, an OpenMP/MPI hybrid codes, top #8 code at NERSC.**



# Density Functional Theory

- **What it solves**
  - Kohn-sham equation

$$\left\{-\frac{1}{2}\nabla^2 + V(r)[\rho]\right\}\psi_i(r) = E_i\psi_i(r)$$

$$\int \psi_i(r)\psi_j(r)dr = \delta_{ij}, \{\psi_i\}_{i=1,\dots,N}$$

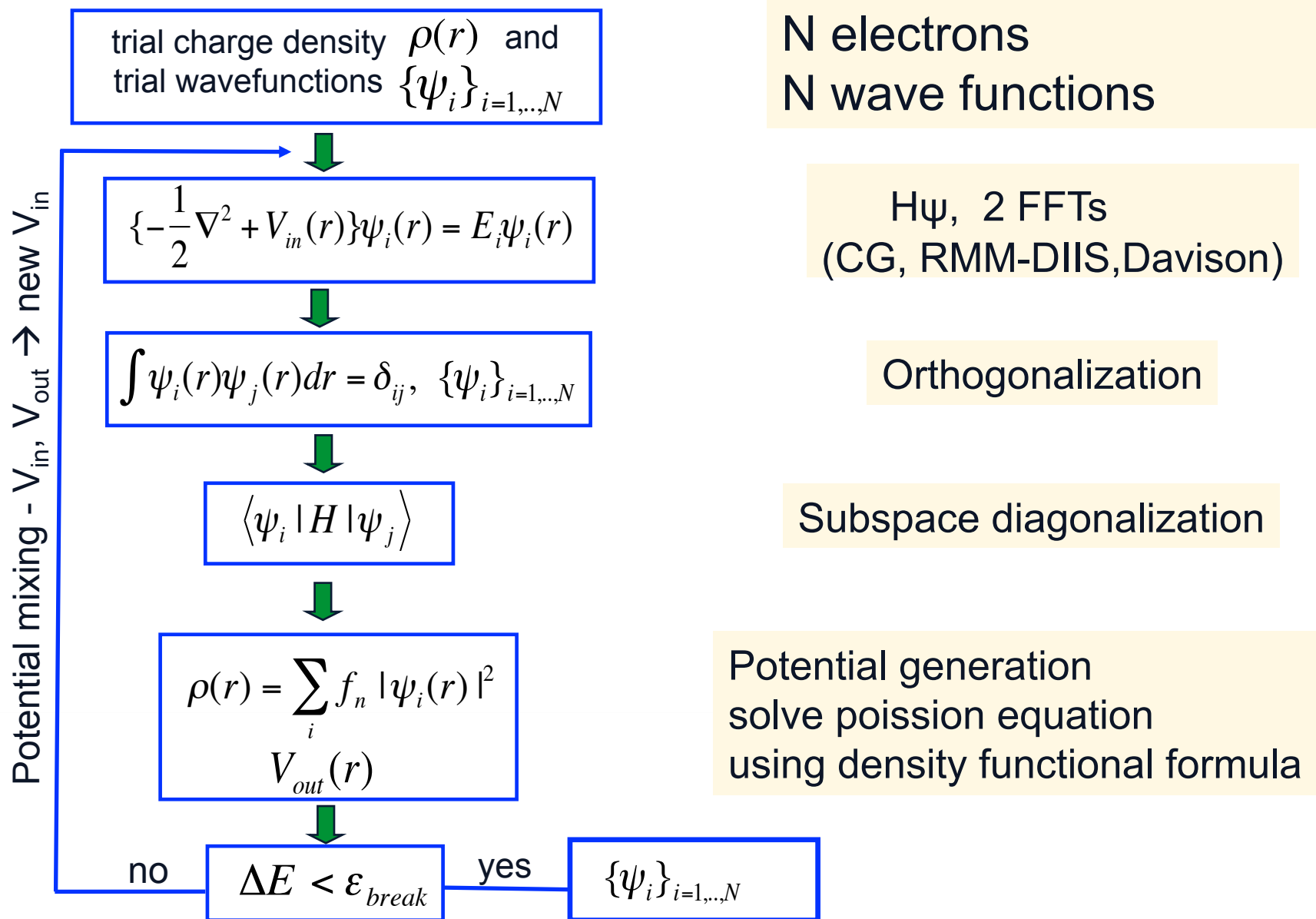
Local Density Approximation:

$$V(r)[\rho] = \sum \frac{Z_R}{|r-R|} + \int \frac{\rho(r')}{|r-r'|}d^3r' + \mu(\rho(r))$$

$$\psi_i(r) = \sum_G^R C_{i,G} e^{[i(k+G).r]}$$



# Flow chart of DFT codes





# Parallelization in DFT codes

## Level 1: Parallel over k-points

- The number of processors,  $N_{\text{tot}}$ , is divided into  $n_{\text{kg}}$  group, each group has  $N_k$  number of processors ( $N_{\text{tot}} = n_{\text{kg}} * N_k$ )
- Each group of processors deal with  $nk_{\text{tot}}/n_{\text{kg}}$  number of k points

$$\{\psi_{i,k}\}, i = 1, \dots, N; k = 1, nk_{\text{tot}}$$



U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science



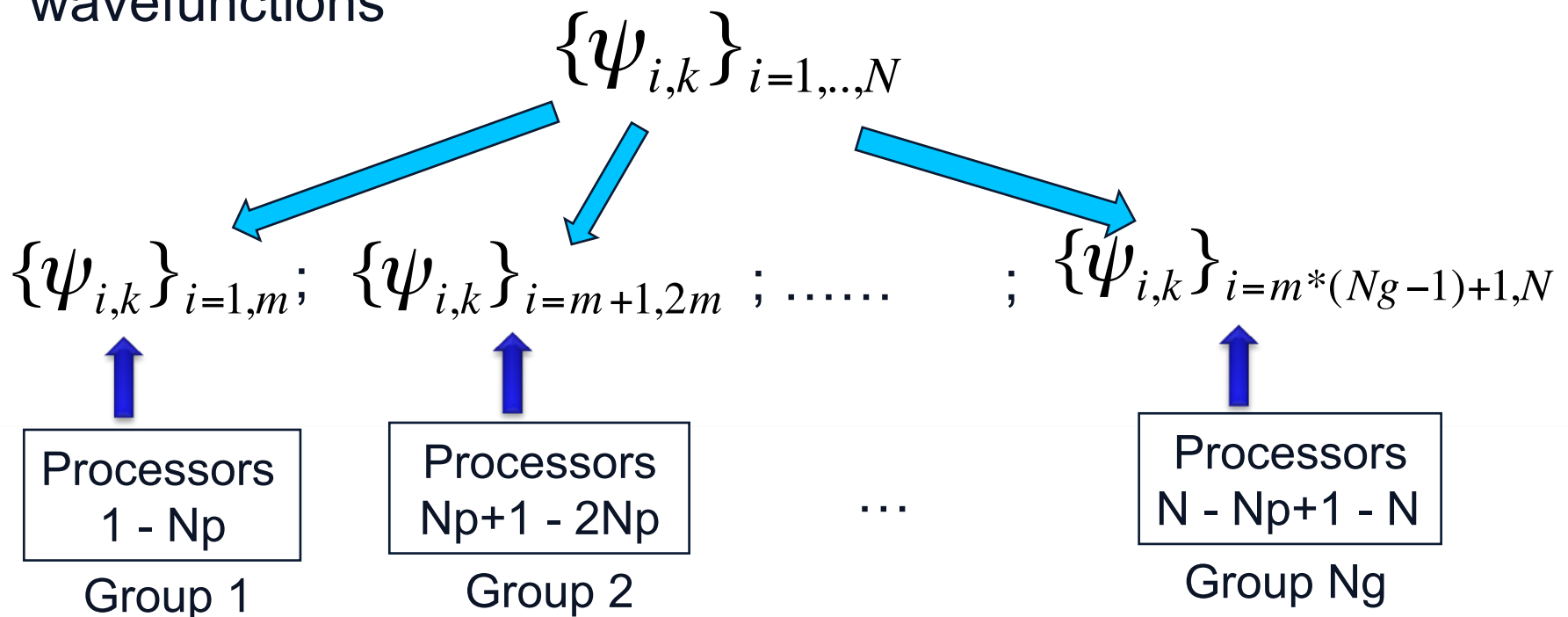
Lawrence Berkeley  
National Laboratory



# Parallelization in DFT codes

## Level 2: Parallel over bands

- The number of processors,  $N_k$ , is divided into  $N_g$  group, each group has  $N_p$  number of processors ( $N_{tot}=N_g*N_p$ )
- $N$  wavefunctions are also divided into  $N_g$  groups, each with  $m$  wavefunctions
- One group of processors deal with one group of wavefunctions



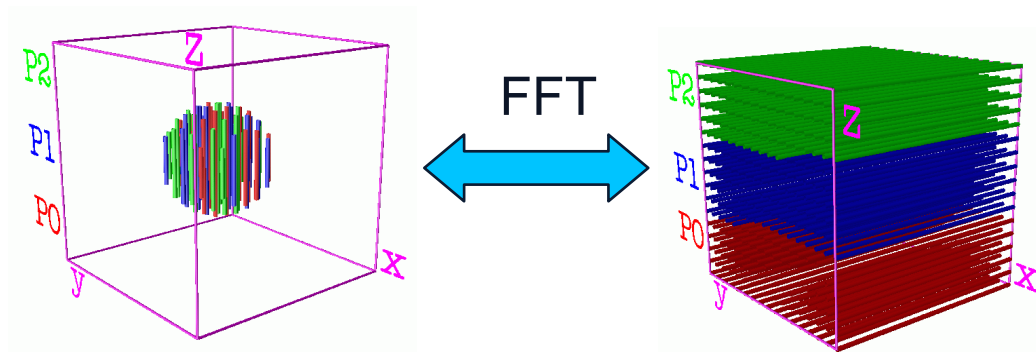




# Parallelization in DFT codes

## Level 3: Parallel over planewave basis set

Within each group of processors, the planewave basis is divided among the  $N_p$  number of processors:



$$\psi_{i,k}(r) = \sum_G C_{i,G} e^{i(k+G).r}$$

Divide the G-space into columns, and distribute them to the  $N_p$  processors

Real space



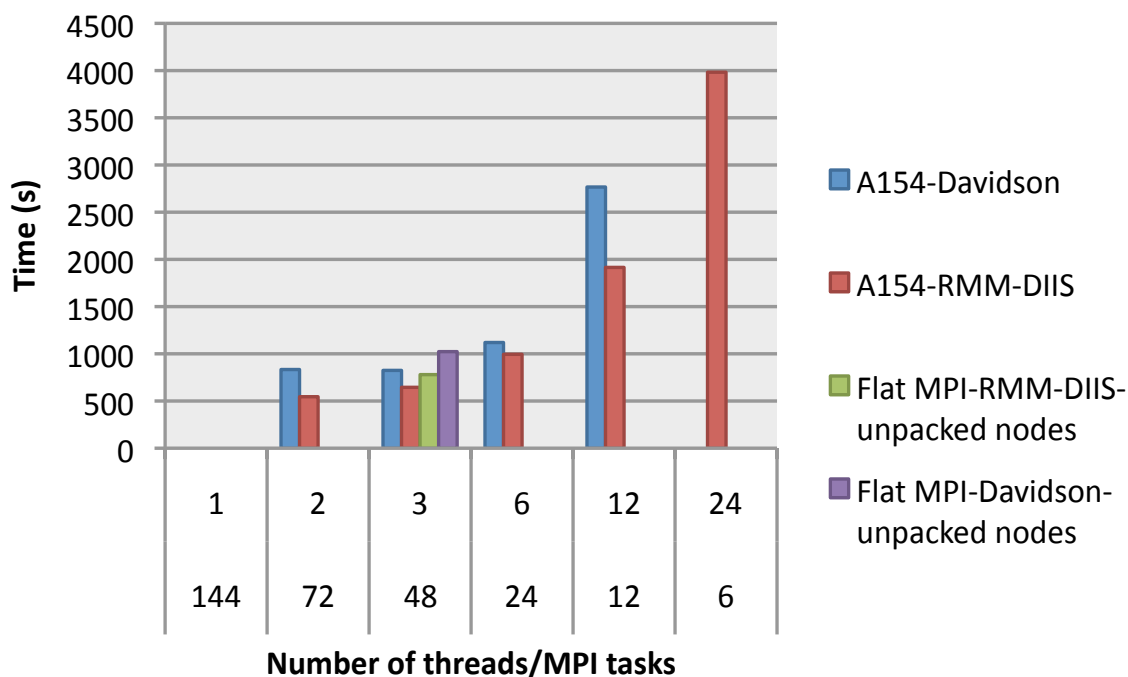
## VASP

- **A planewave pseudopotential code**
  - A commercial code from Univ. of Vienna
- **Libraries used**
  - BLAS, fft
- **Parallel implementations**
  - Over planewave basis set and bands
  - >1proc/atom scale
  - Flops 20-50% of peak (in real calculations)
- **VASP use at NERSC**
  - Used by 83 projects, 200 active users

<http://cmp.univie.ac.at/vasp>



# VASP: Performance vs threads



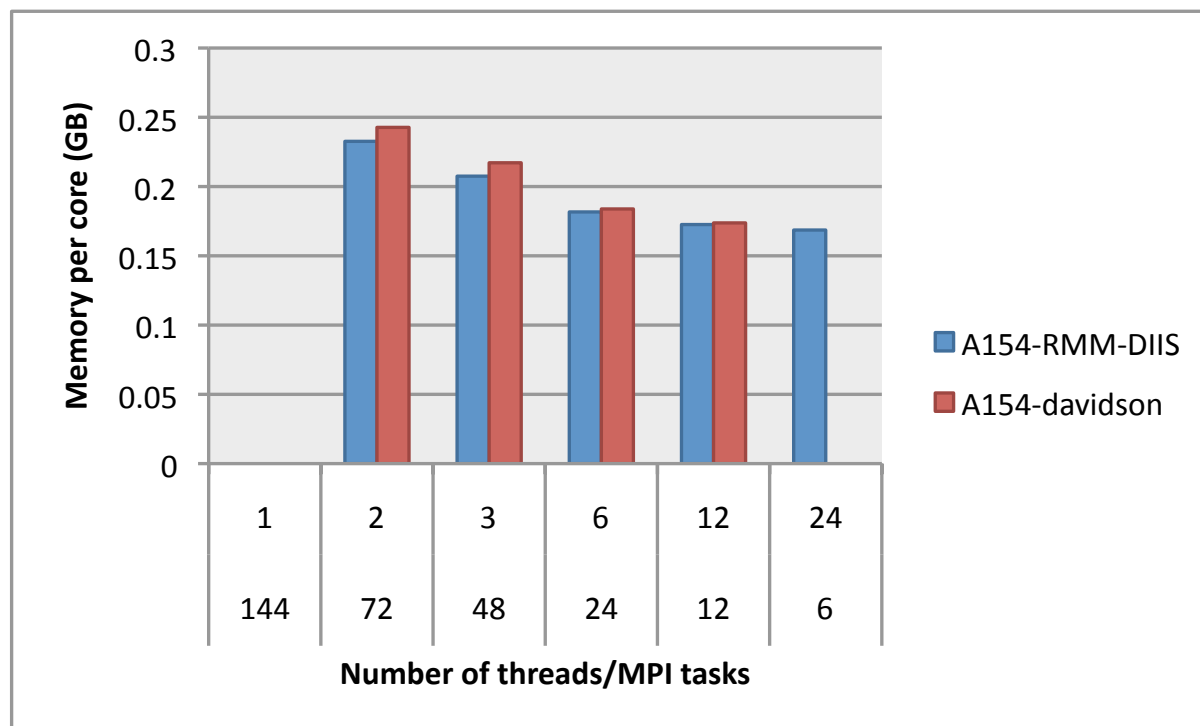
## Test case A154:

154 atoms  
998 electrons  
 $\text{Zn}_{48}\text{O}_{48}\text{C}_{22}\text{S}_2\text{H}_{34}$   
80x70x140 real-space  
grids;  
160x140x280 FFT  
grids  
4 kpoints

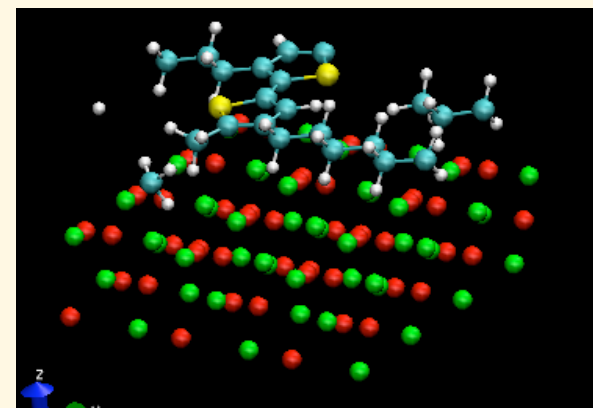
- When the number of threads increases, a little or no performance gain. Code runs slower.
- But in comparison to the flat MPI, at threads=3, VASP runs faster than the flat MPI on unpacked nodes by 20-25%



## VASP: Memory usage vs threads



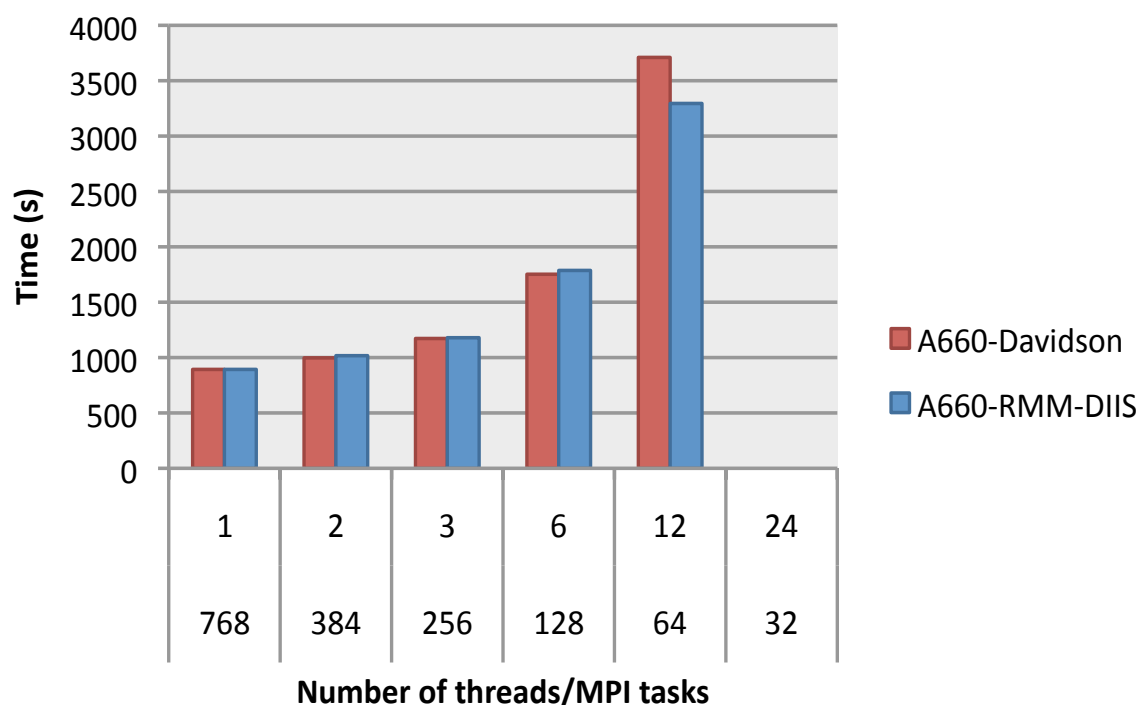
### Test case A154:



- Memory usage is reduced when the number of threads increases
- At threads=3, the memory usage is reduced by 10% compared to that of threads=2



# VASP: VASP runs slower when the number of threads increases



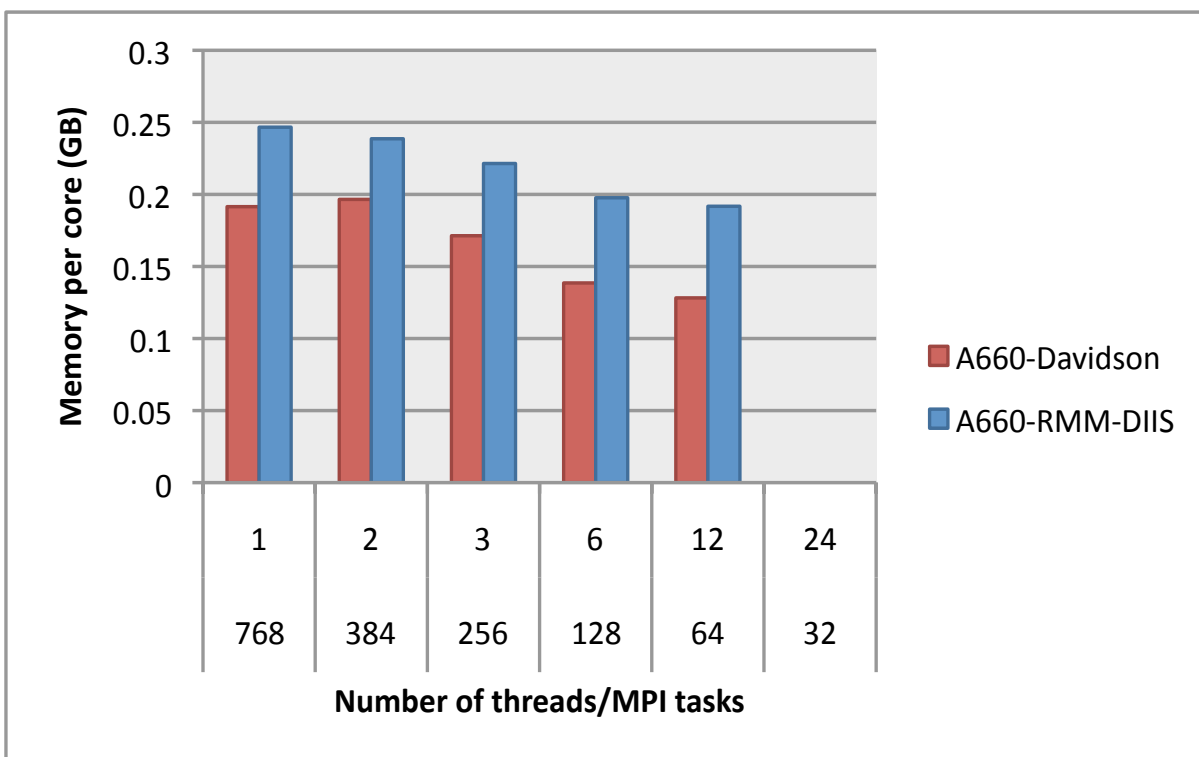
## Test case A660:

660 atoms  
2220 electrons  
 $C_{200}H_{230}N_{70}Na_{20}O_{120}P_{20}$   
240x240x486 real-space grids;  
480x380x972 FFT grids  
1 kpoint (Gamma point)  
Gamma kpoint only  
VASP

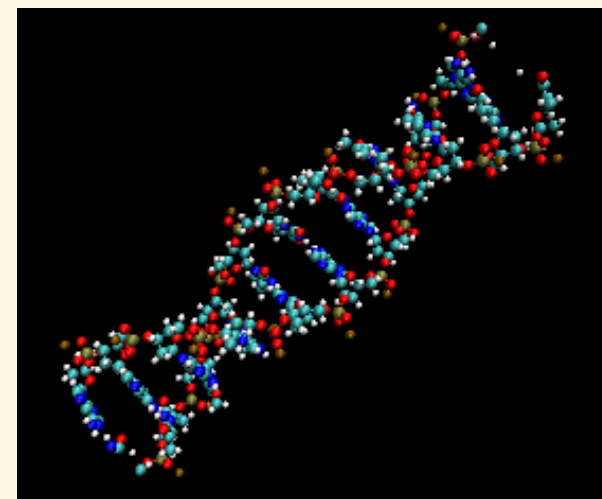
Threaded VASP at best (threads=2) is slightly slower (~12%) than the flat MPI



# VASP: Memory usage vs threads



## Test case A660:



Compare the memory usage for threads=2 and the flat MPI:  
For RMM-DIIS: there is a slight memory saving  
For Davidson: no memory saving at threads=2, slightly more use of memory (<3%)



# Quantum Espresso

- **A planewave pseudopotential code**
  - An open software DEMOCRITOS National Simulation Center and SISSA with collaboration with many other institutes
- **Libraries used**
  - BLAS, fft
- **Parallel implementations**
  - Over k-points, planewave basis and bands
  - >1proc/atom scale
- **QE use at NERSC**
  - Used by 21 projects



U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science

<http://www.quantum-espresso.org>



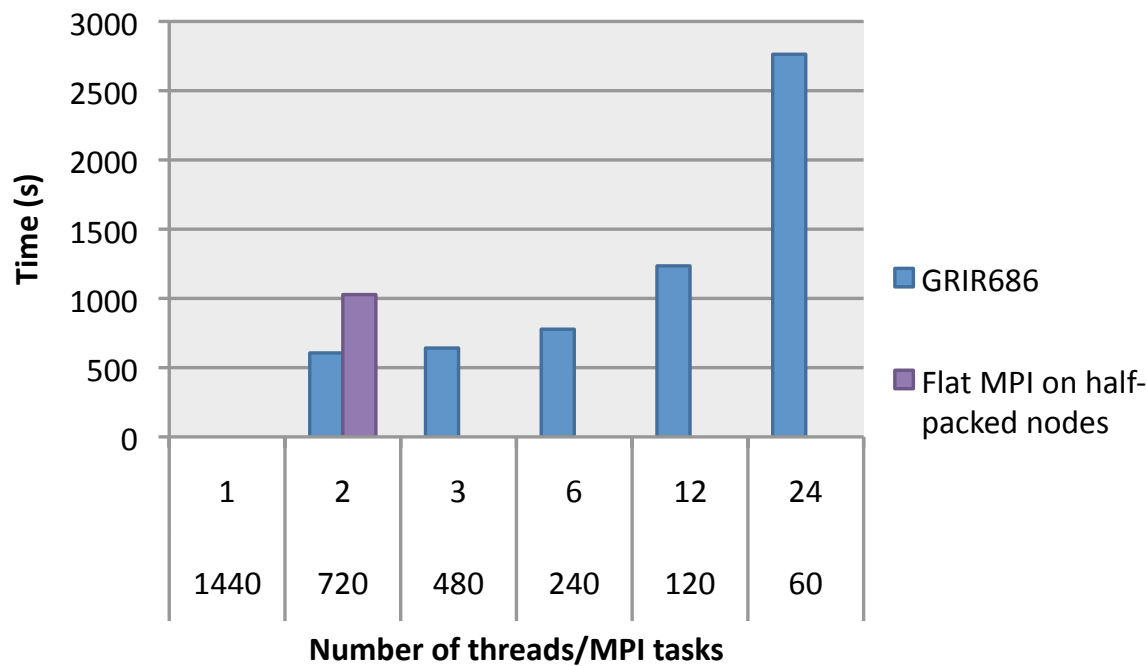
Lawrence Berkeley  
National Laboratory



# QE: The Hybrid OpenMP+MPI code runs faster than the flat MPI

## Test case GRIR686:

686 atoms  
5174 electrons  
 $C_{200}Ir_{486}$   
180x180x216 FFT  
grids  
2 kpoints

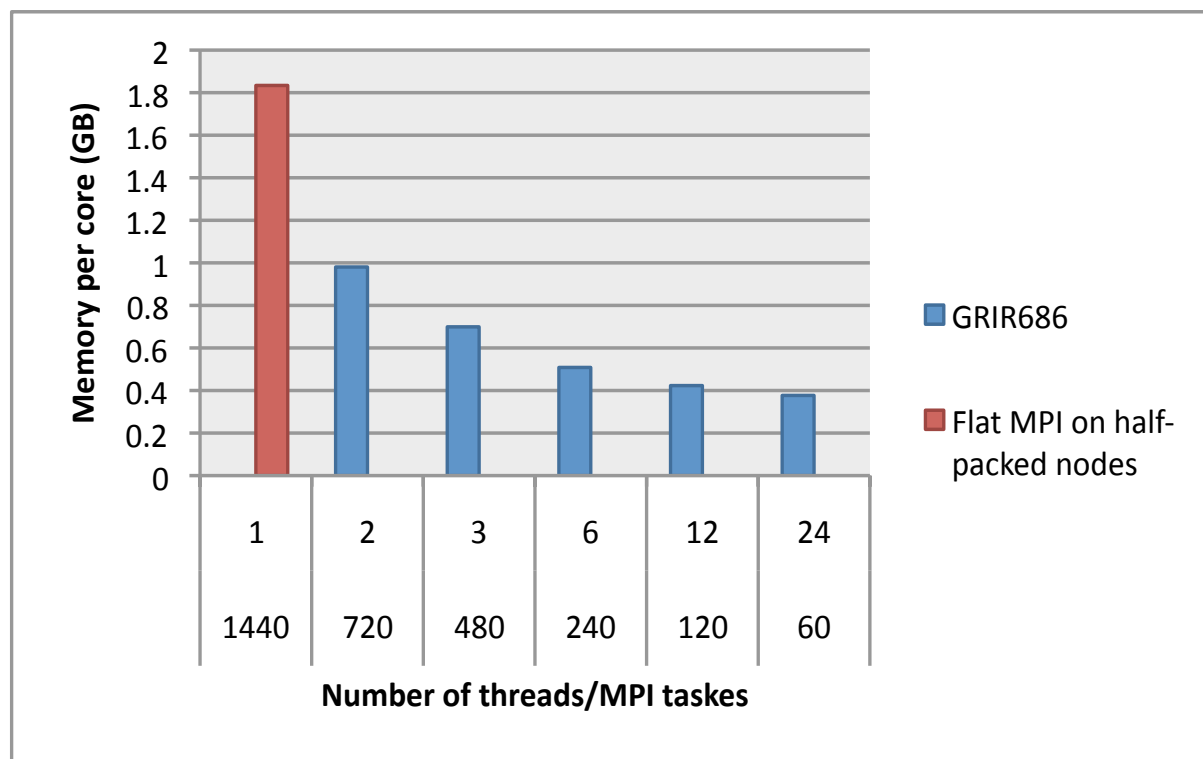


At threads=2, QE runs faster than the flat MPI on half-packed nodes by 38%





## QE: The OpenMP+MPI code uses less memory than the flat MPI



### Test case GRIR686:

686 atoms

5174 electrons

$C_{200}Ir_{486}$

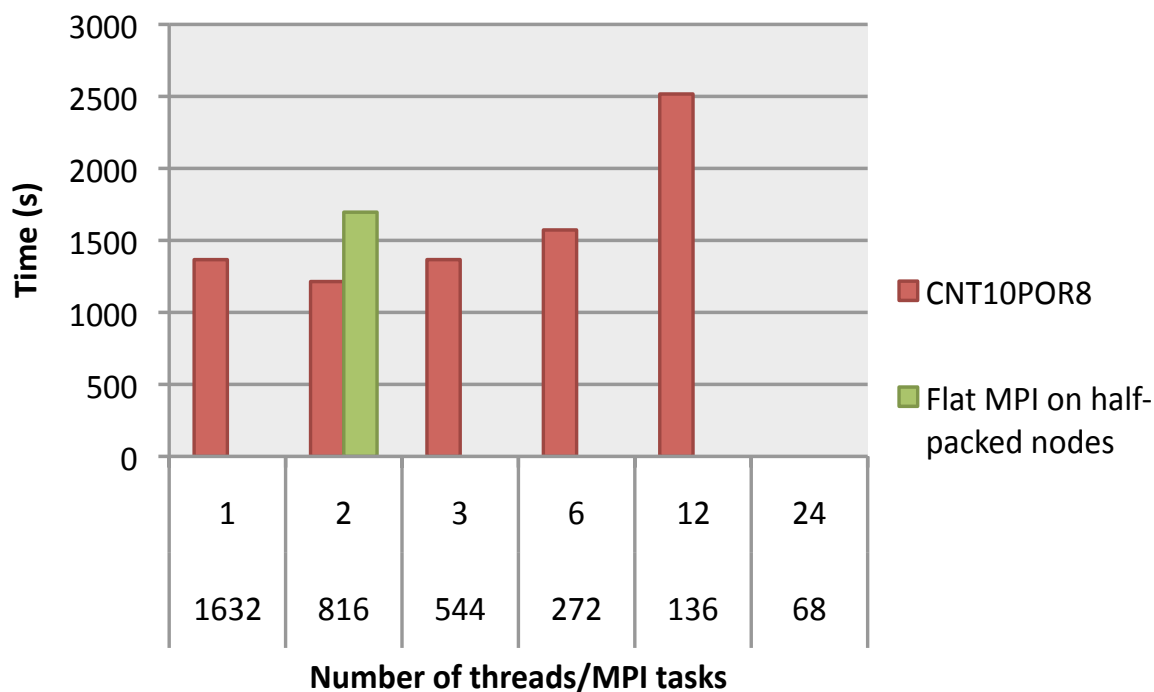
180x180x216 FFT  
grids

2 kpoints

At threads=2, the memory usage is reduced by 64%  
when compared to the flat MPI



# QE: The Hybrid OpenMP+MPI code runs faster than the flat MPI



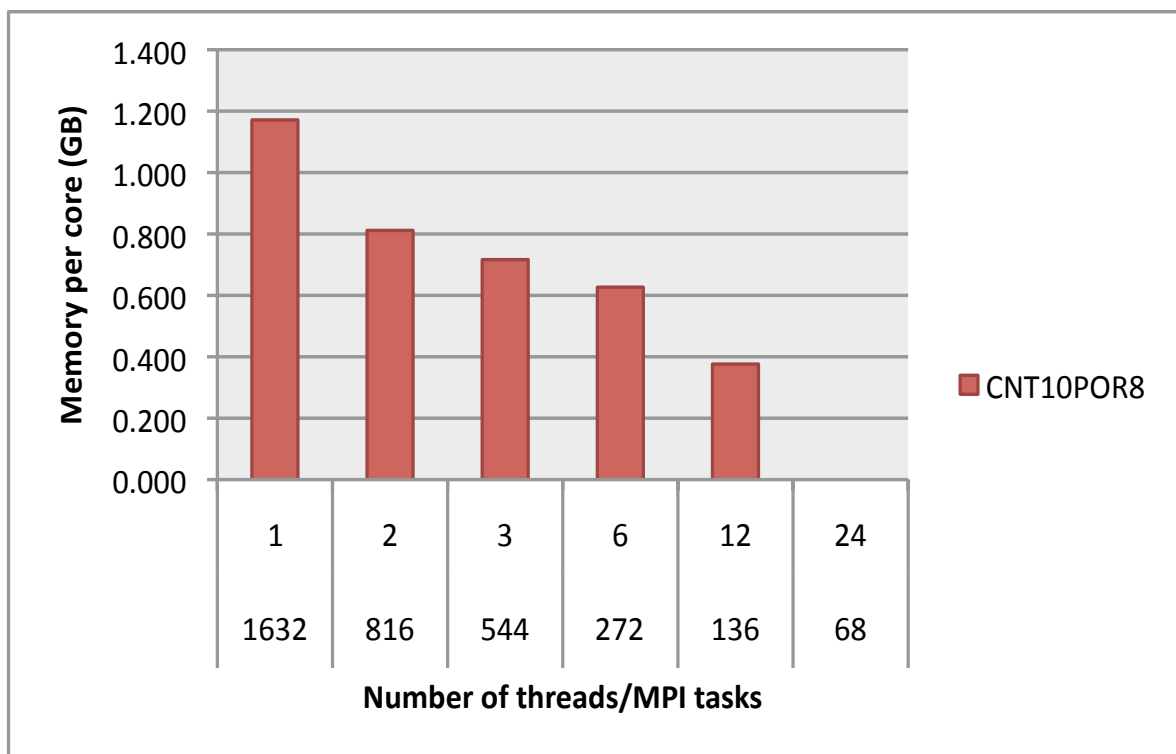
## Test case CNT10POR8:

1532 atoms  
5232 electrons  
 $C_{200}Ir_{486}$   
540x540x540 FFT  
grids  
1 kpoint (Gamma point)

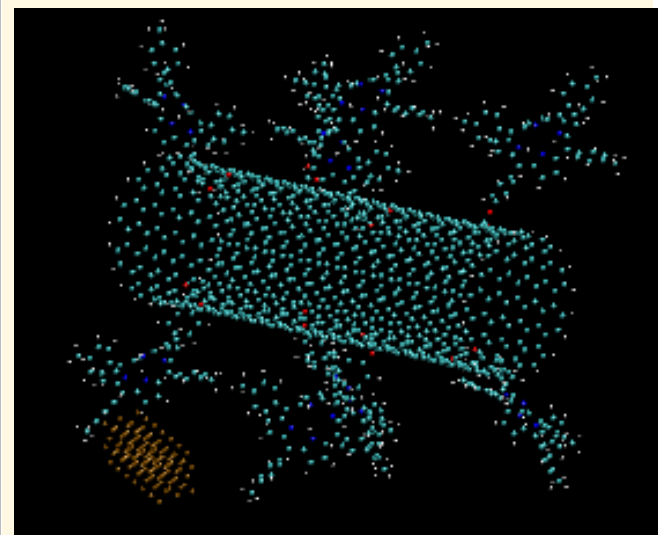
At threads=2, QE runs faster than the flat MPI on half-packed nodes by 28%



## QE: The OpenMP+MPI code uses less memory than the flat MPI



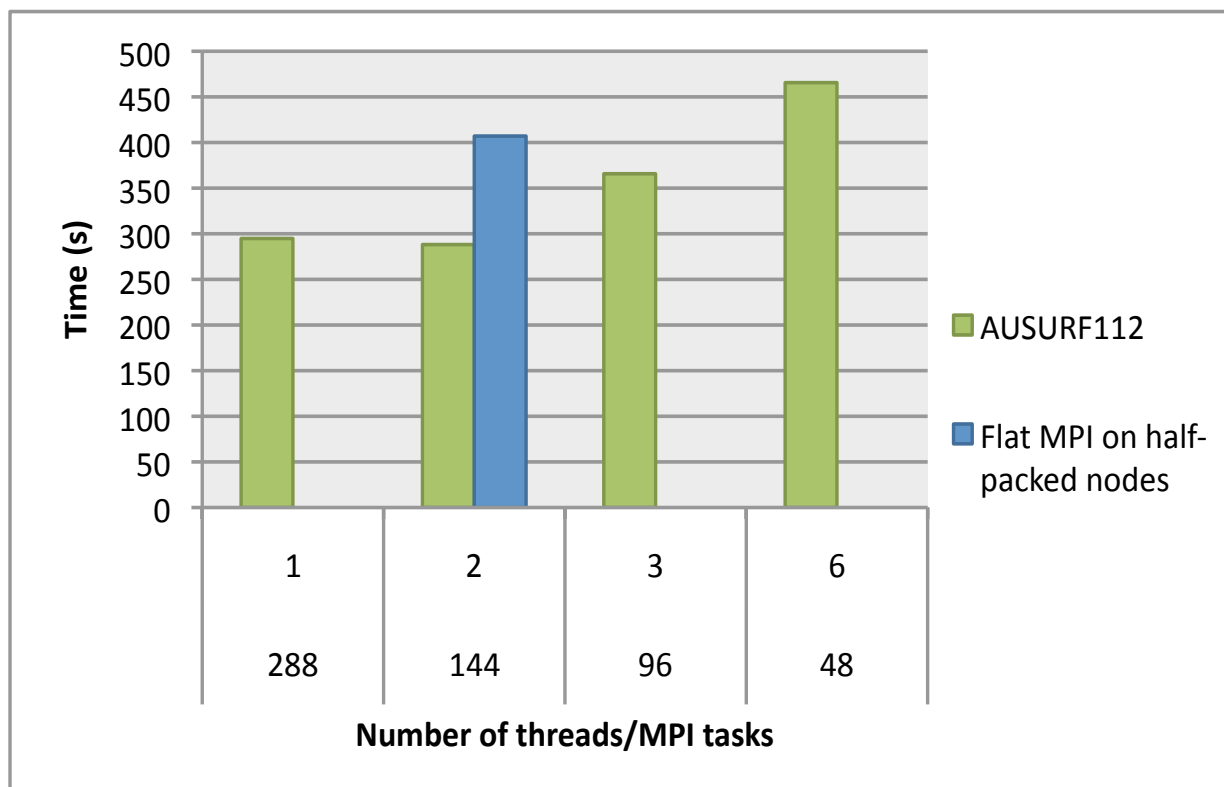
Test case  
CNT10POR8:



At threads=2, the memory usage is reduced by 30%



## QE: The Hybrid OpenMP+MPI code runs faster than the flat MPI



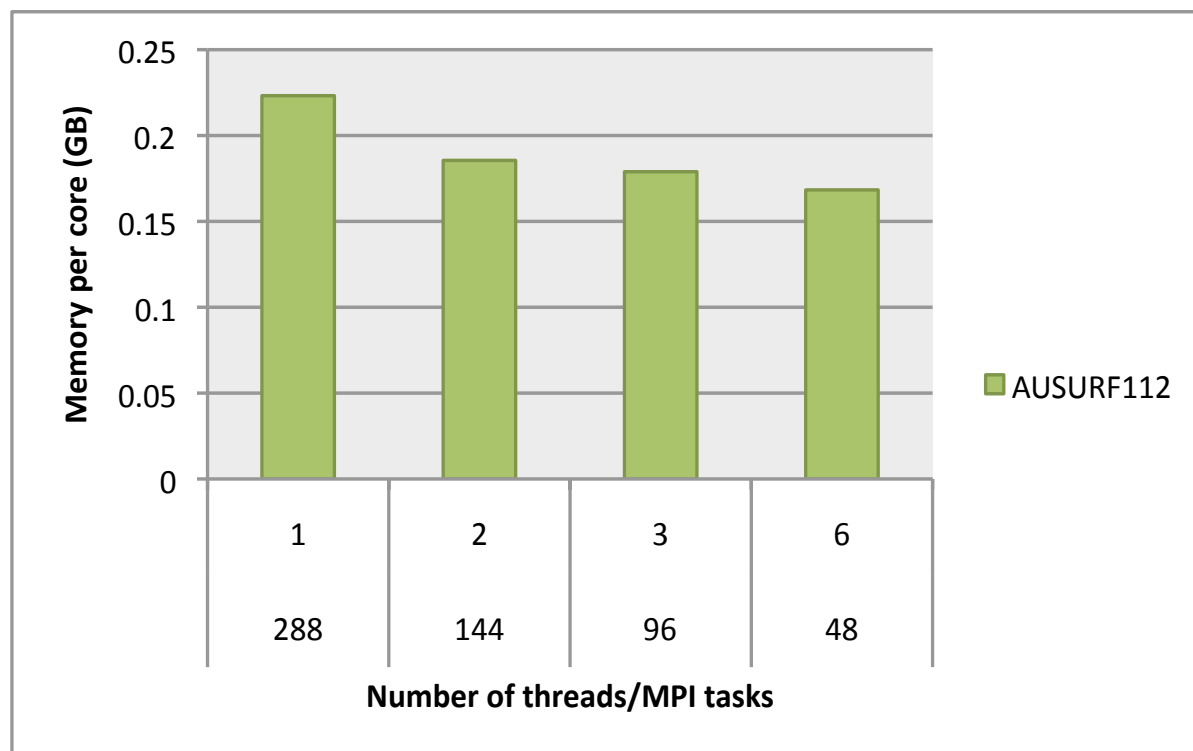
### Test case AUSURF112:

112 atoms  
5232 electrons  
 $C_{200}Ir_{486}$   
125x64x200 FFT grids  
80x90x288 smooth  
grids  
2 k-points

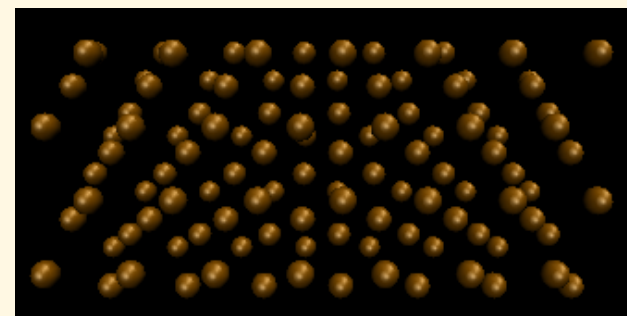
At threads=2, QE runs faster than the flat MPI on half-packed nodes by 22%



## QE: The OpenMP+MPI code uses less memory than the flat MPI



Test case  
**AUSURF112:**



At threads=2, QE runs faster than the flat MPI on half-packed nodes by 38%



## Conclusions

- **Performance of VASP from using MPI +OpenMP programming model**
  - A low-effort thread implementation - linked with the multi-threaded BLAS libraries
  - Slight performance gains in the order of 20-25%
  - Addition of OpenMP directives in the source code should help this situation.
  - Slight memory savings
  - Many optional parameters that affect the performance of VASP, our results are not all.



## Conclusions

--continued

- **Performance of QE from using MPI +OpenMP programming model**
  - OpenMP directives in the source code + linking to the multi-threaded libraries
  - Performance gains in the order of 40% in comparison to flat MPI, best performance achieved at threads=2
  - Significant memory savings, 20-40% per core when compared to the flat MPI.



## Conclusions

--continued

- **OpenMP+MPI is a promising programming model on Hopper**
  - Other DFT and other MPI codes which can make use of multi-threaded BLAS routines.





# Acknowledgement

- **NERSC user Wai-Yim Ching and Sefa Dag for providing VASP test cases**
- **NERSC resources**