Improving CASINO performance for models with large number of electrons

Dario Alfè, Mike Gillan
UCL, London, UK

Lucian Anton
NAG Ltd, Oxford, UK

HECToR Distributed CSE Support technical meeting
September 23-24, 2009

Outline

- Introduction
- Algorithms for distributed data
  - Shared memory
  - MPI two-sided
  - MPI one-sided & SHMEM
- Second Level Parallelism
  - MPI
  - OpenMP
- IO at large scale
- Conclusions

QMC and CASINO

Quantum Monte Carlo techniques are used to compute electronic structure of solids, large molecules, nano-clusters...

- Very precise results
- Good scaling with system size
- Good parallel efficiency

CASINO developed by Theory of Condensed Matter group, Cambridge University.

Fortran 95 +MPI

http://www.tcm.phy.cam.ac.uk/~mdt26/casino2.html

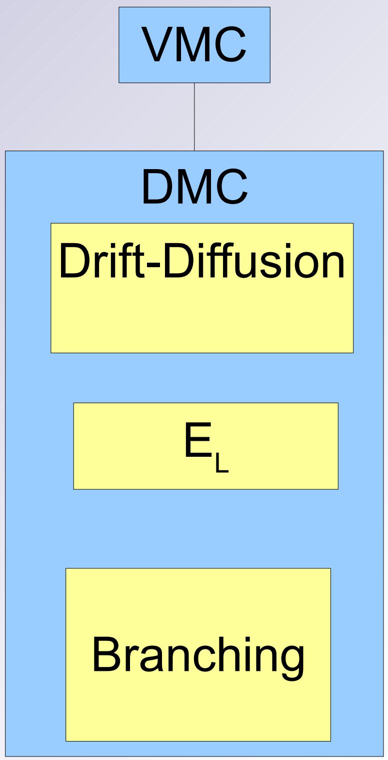
Background

- Quantum many-body systems: N_e electrons, N_I ions.
- Computationally challenging problem and of practical interest.

$$i\hbar \frac{\partial \Psi(R,t)}{\partial t} = -\frac{\hbar^2}{2\mathrm{m}} \sum_{i=1}^{N_e} \nabla_i^2 \Psi(R,t) + V(R, R_I)\Psi(R,t)$$

$$E = \frac{\langle \Psi | H \Psi \rangle}{\langle \Psi | \Psi \rangle}$$

CASINO QMC computational steps



$$\frac{1}{(2\pi\tau)^{3N/2}}\exp\left(-\frac{\left(\boldsymbol{R}-\boldsymbol{R}'-\tau\boldsymbol{v}(\boldsymbol{R}')\right)^2}{2\tau}\right)$$

$$E_L(\boldsymbol{R})=\Psi^{-1}\boldsymbol{H}\Psi$$

$$\exp\left(-\frac{\tau}{2}\left[E_L(\boldsymbol{R})+E_L(\boldsymbol{R}')-2E_T\right]\right)$$

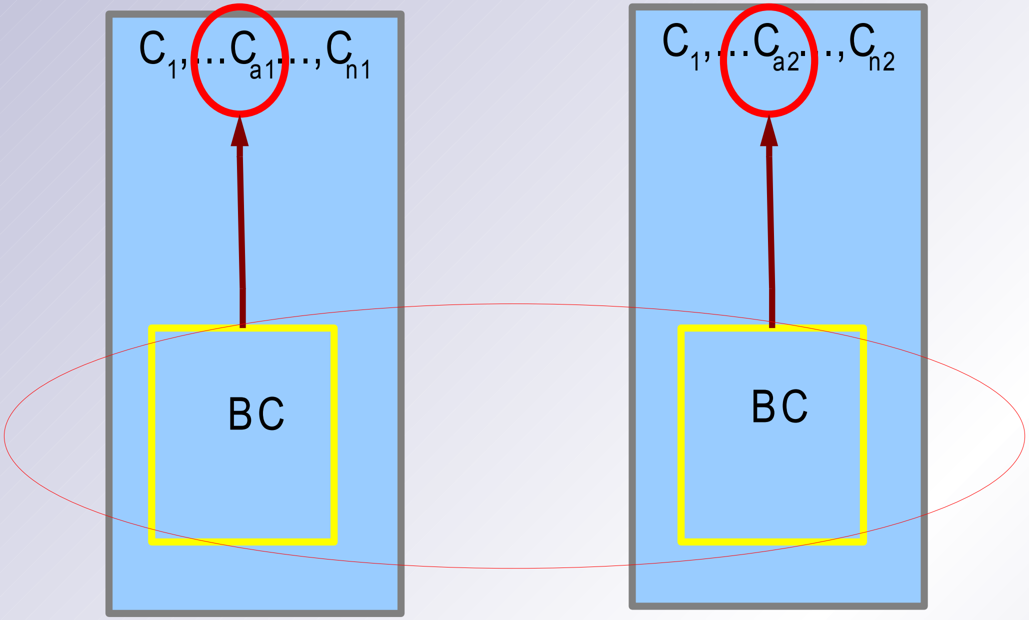
Blip coefficients

$$\Psi = f(t) e^{J} D_{\uparrow} D_{\downarrow}$$

$$D_{\uparrow} = \begin{vmatrix} \phi_1(r_1) & \phi_1(r_2) & \cdots & \phi_1(r_{N_\uparrow}) \\ \vdots & \cdots & \cdots & \vdots \\ \phi_{N_\uparrow}(r_1) & \phi_{N_\uparrow}(r_2) & \ldots & \phi_{N_\uparrow}(r_{N_\uparrow}) \end{vmatrix}$$

$$\mathrm{BC}(N_o, 0:N_{gx}-1, 0:N_{gy}-1, 0:N_{gz}-1, N_s)$$

The origin of the memory problem



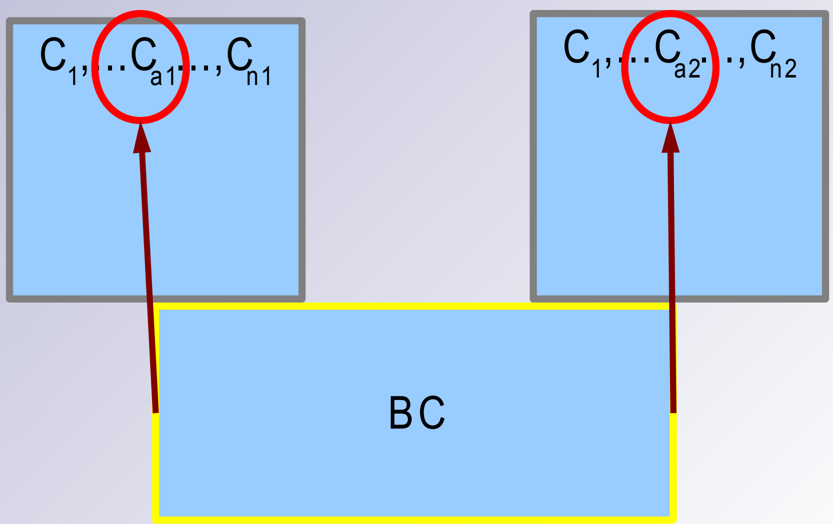
Can BC be shared
on a processor or a node?

1024 electrons need 512 OPO

80 grid points in each direction direction

>2GB in double precision

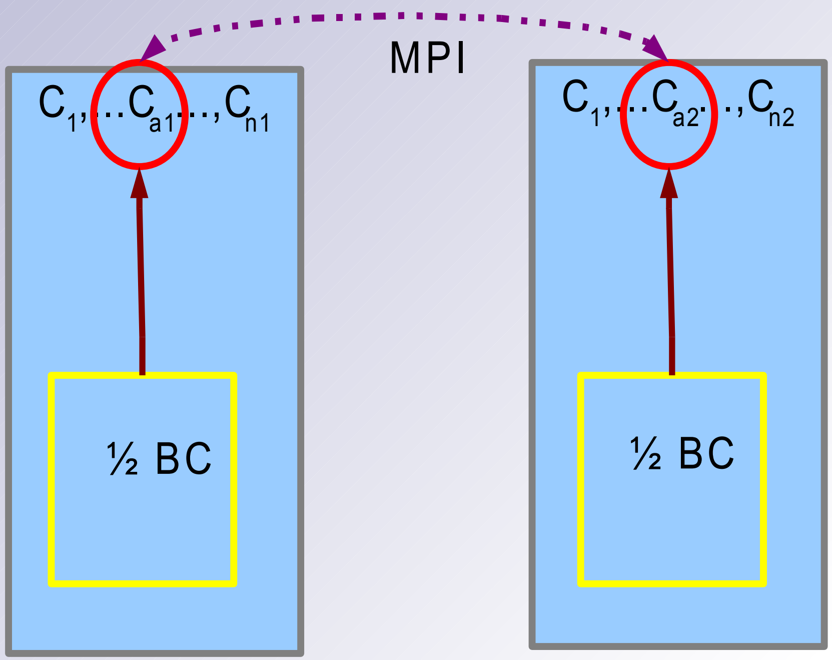
SHM



No MPI solution to share memory on a node, but one can use Unix inter process communication library:

- Easy to implement.
- Needs C functions to allocate the shared memory.
- Cray pointers to pass the reference to the FORTRAN pointers.

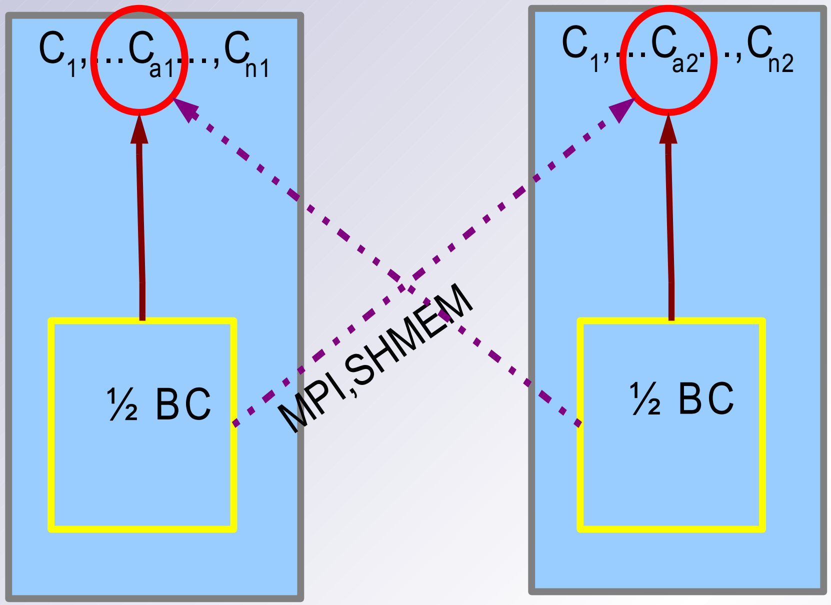
MPI-2Sided



- No need of shared memory
- Fully compliant with CASINO coding standard
- Call for orbital computation must be synchronous

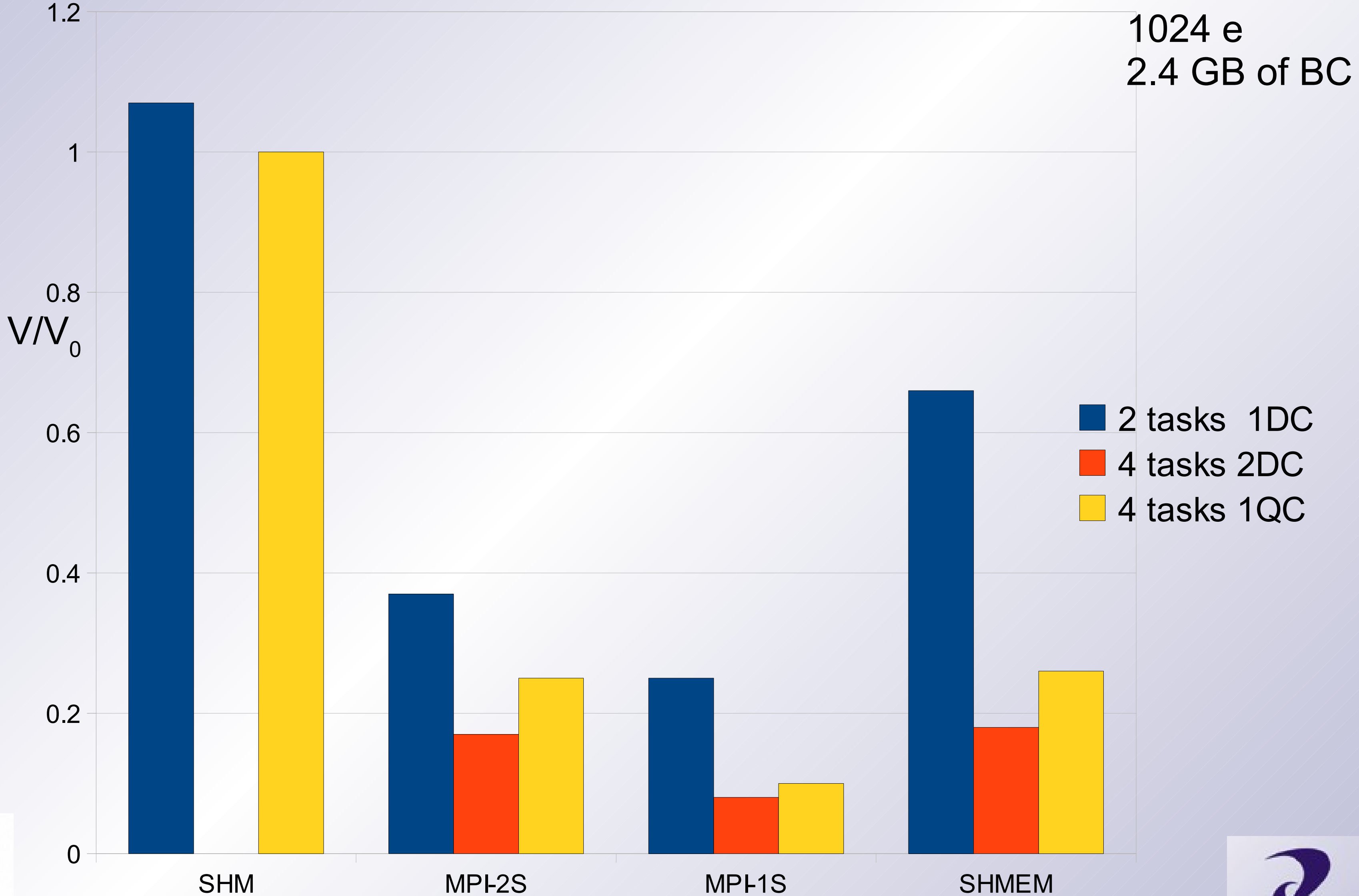
Implemented by Randolph Hood, LLNL, June 2008

MPI-1Sided



Can we avoid the synchronisation of MPI-2S with MPI one-sided or CRAY SHMEM library?

Timing results



Second level parallelism I

Why is needed?

$$t_{\text{total}} \approx N_{\text{step}} \times \frac{N_{\text{pop}}}{P} \times t_{\text{step}}$$

Second level parallelism II

For one configuration step we have to compute:

- The sums involved in computing the energy terms scales as N^2
- Slater matrix elements: N^2
- Slater determinant: N^3 (LU decomposition) or N^2(cofactor matrix)

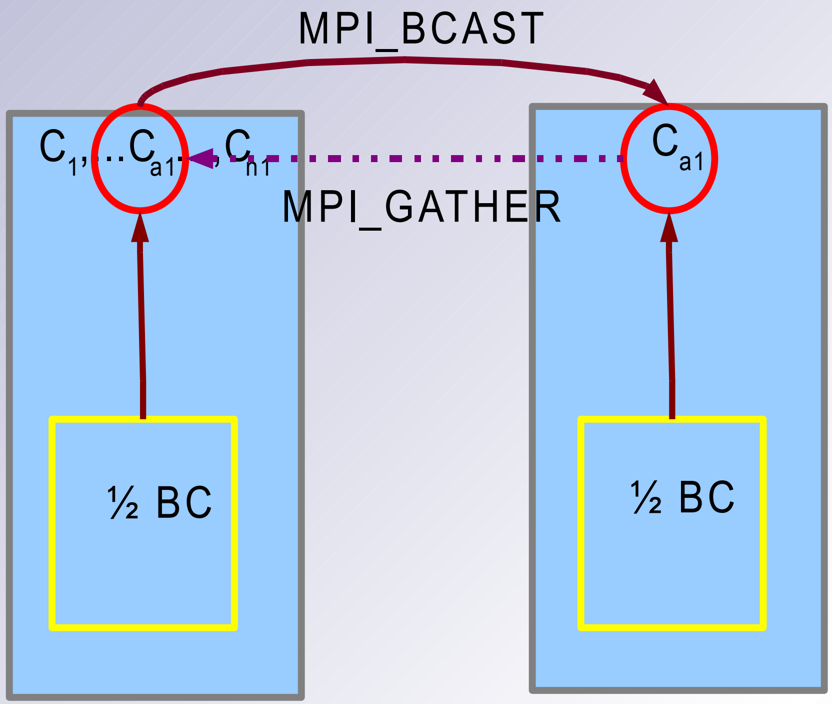
$$t_{\text{step}}[O(10^4)] \approx 10^{2.x} t_{\text{step}}[O(10^3)]$$

QMC algorithms for electronic structure at the petascale
K P Esler *et al*, J Phys: Conf Series, **125**(2008) 0122057

HECToR

RESEARCH COUNCILS UK

MPI second level parallelism



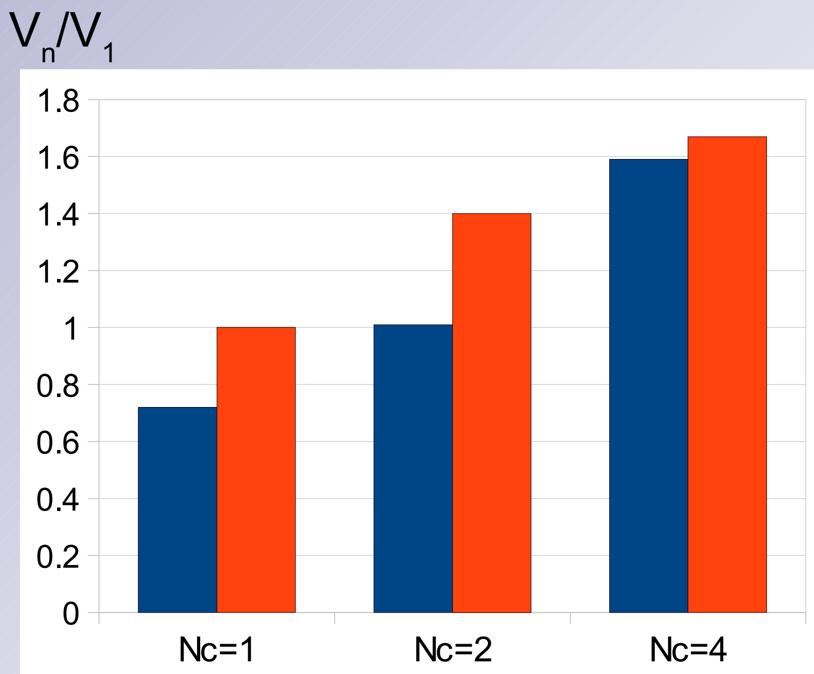
- The pool computes for the same configuration: OPO, Jastrow factor, energy components, Slater determinats.
- The computation is controlled by the pool head.

OpenMP second level parallelism

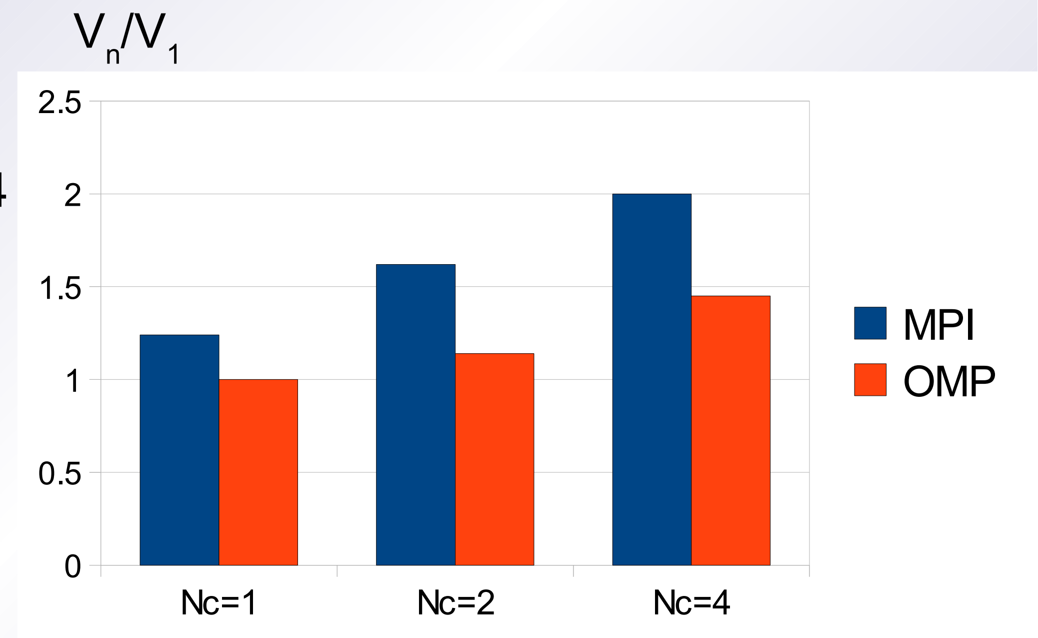
- Loops over the electron index abound in CASINO code and many of them are easy to adapt to parallel computation

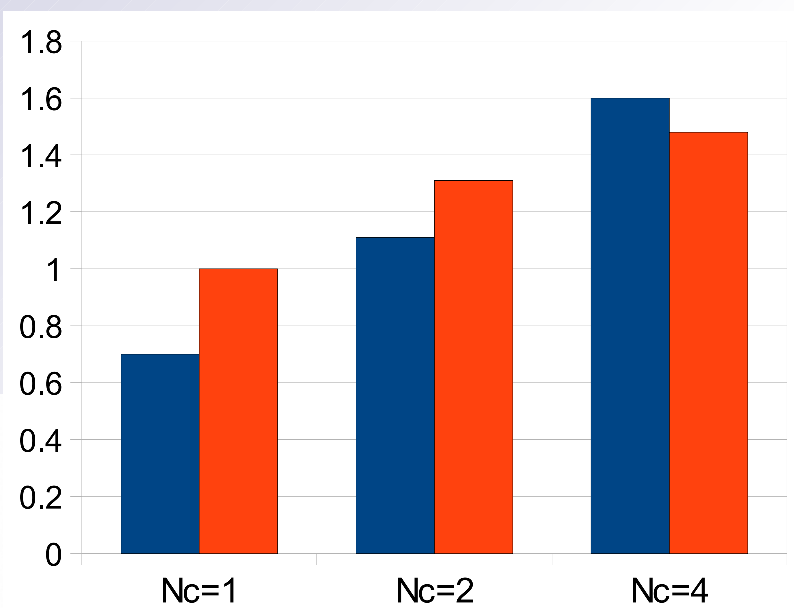
  Higher level OpenMP much harder to implement
- Dependencies in the code (bufferring)
- Compilers cannot handle module subroutines or variables in parallel regions(things are getting better though).

OPO & Jastrow

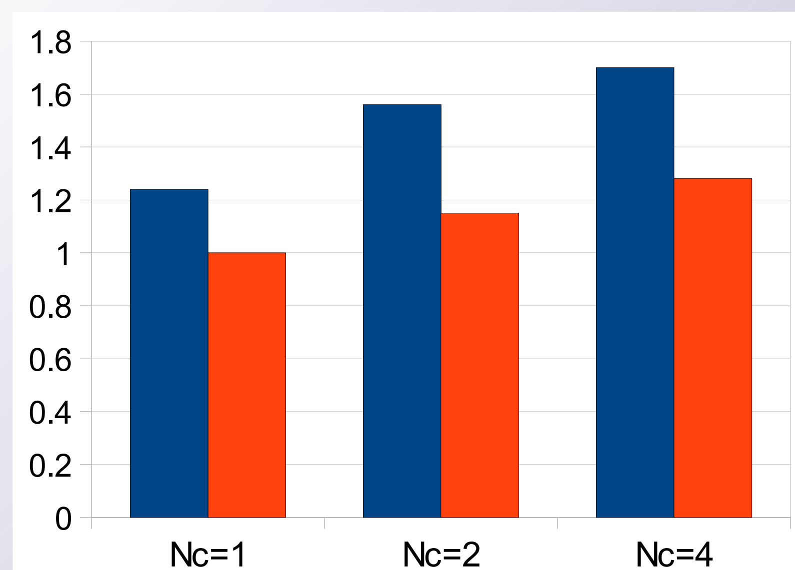


N_e=1024

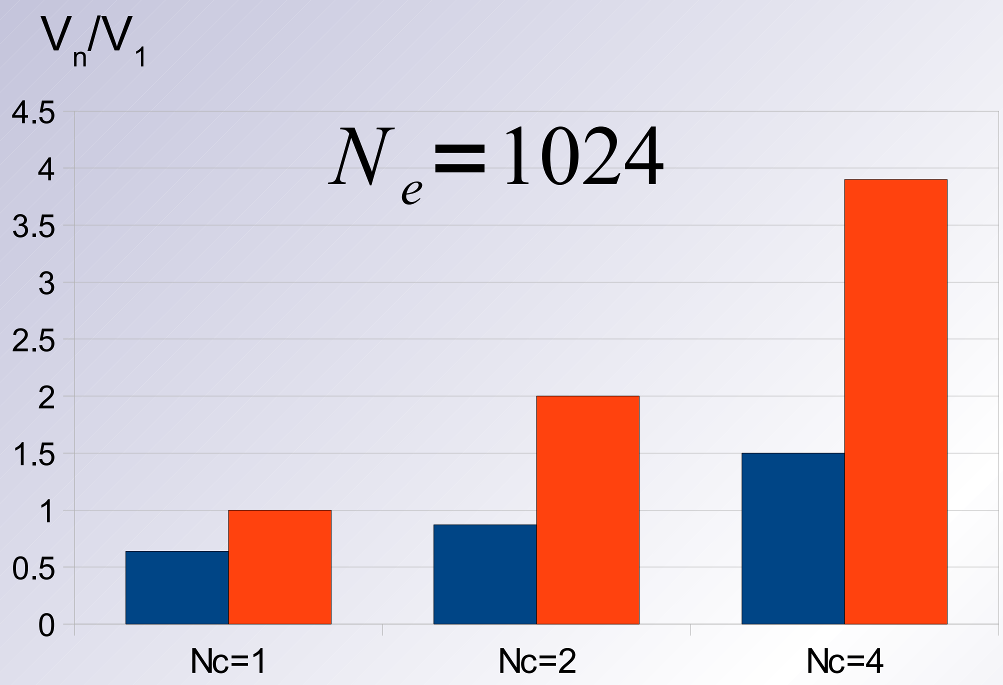




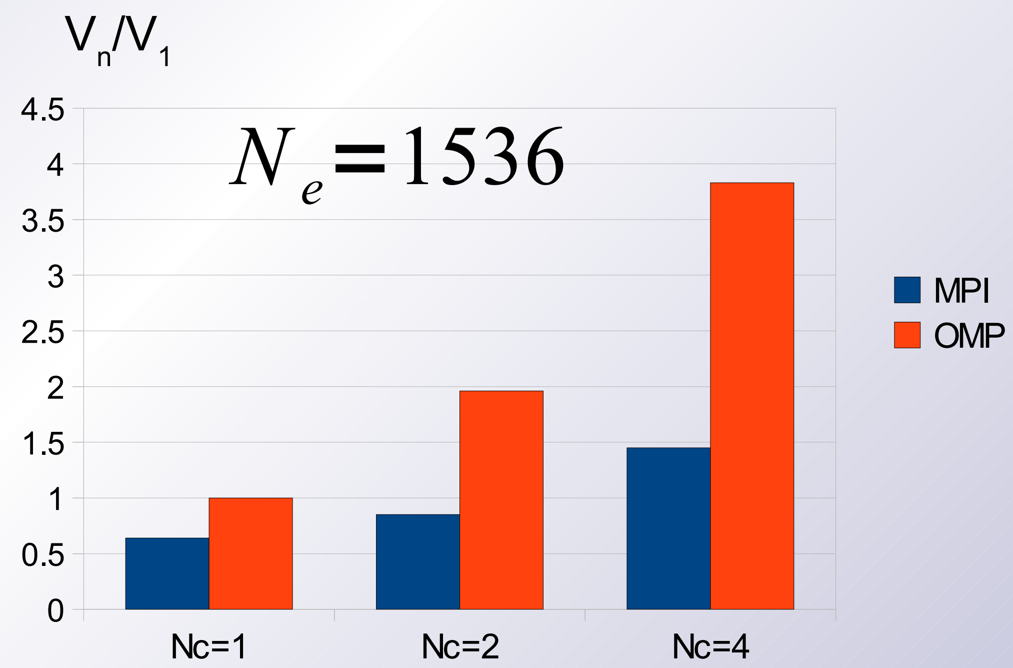
N_e=1536



Ewald sum

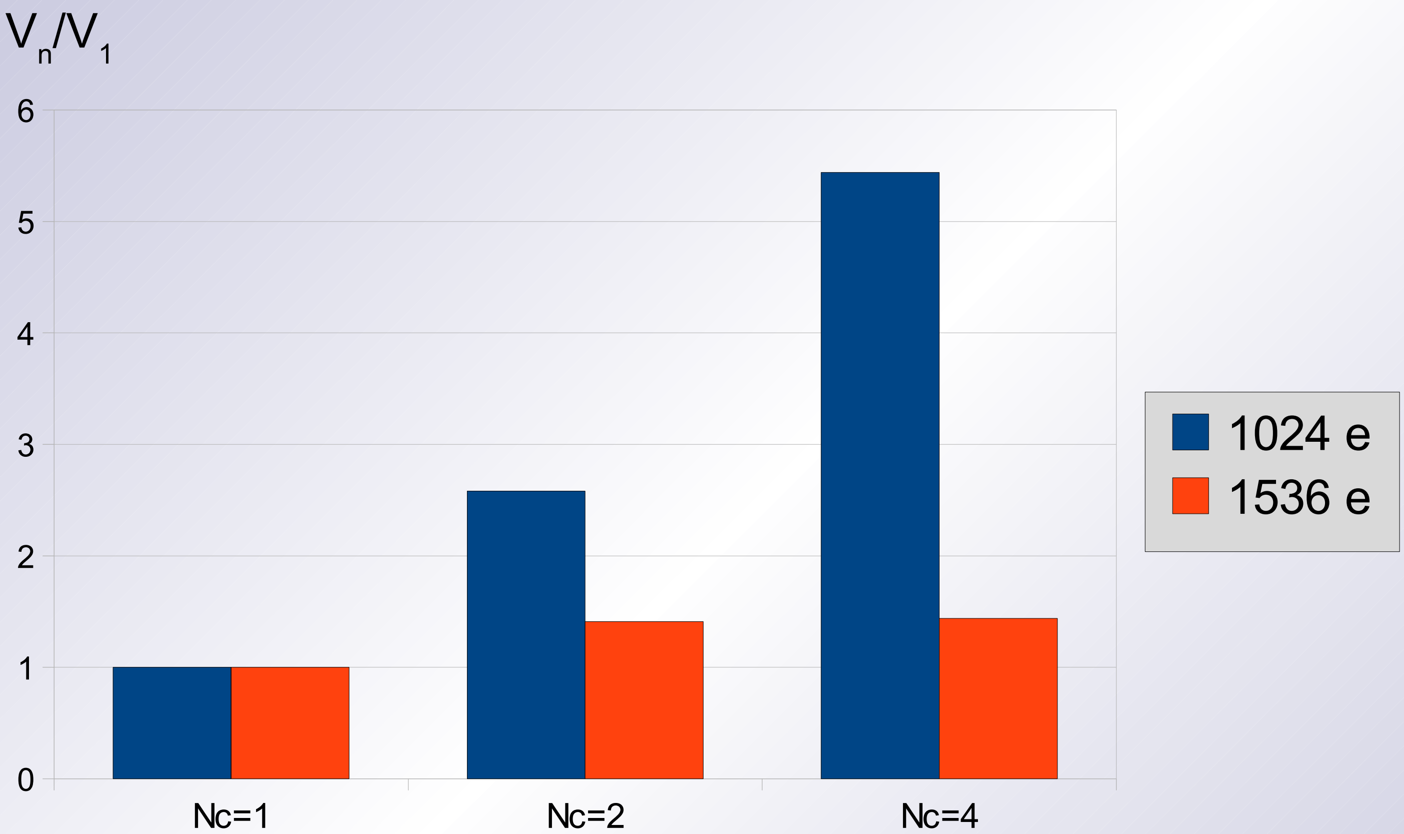


V_n/V_1
$N_e = 1024$
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Nc=1
Nc=2
Nc=4



V_n/V_1
$N_e = 1536$
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Nc=1
Nc=2
Nc=4
MPI
OMP

Update $\overline{D}$ & DMC (OpenMP only)



Update $\bar{D}$ for 1024 electrons

Ncores=2

```
USER / slater_update_dbar_.LOOP@li.2770
------------------------------------------------------------------------
  Time%                                   4.5%
  Time                            31.541876 secs
  Imb.Time                          0.415274 secs
  Imb.Time%                               5.2%
  Calls                  0.002M/sec      117869.0 calls
  PAPI_L1_DCM             23.775M/sec    1477514702 misses
  PAPI_TLB_DM              0.441M/sec      27410560 misses
  PAPI_L1_DCA            881.986M/sec   54811035690 refs
  PAPI_FP_OPS            997.309M/sec   61977759711 ops
  User time (approx)      62.145 secs  142933500000 cycles
100.0%Time
  Average Time per Call                     0.000268 sec
  CrayPat Overhead : Time     0.5%
  HW FP Ops / User time   997.309M/sec   61977759711 ops
10.8%peak(DP)
  HW FP Ops / WCT         997.309M/sec
  Computational intensity    0.43 ops/cycle     1.13 ops/ref
  MFLOPS (aggregate)       997.31M/sec
  TLB utilization         1999.63 refs/miss    3.906 avg uses
  D1 cache hit,miss ratios  97.3% hits          2.7% misses
  D1 cache utilization (M)  37.10 refs/miss     4.637 avg uses
```

Ncores=4

```
USER / slater_update_dbar_.LOOP@li.2770
------------------------------------------------------------------------
  Time%                                   2.4%
  Time                            13.466174 secs
  Imb.Time                          0.523796 secs
  Imb.Time%                               6.7%
  Calls                  0.002M/sec      117869.0 calls
  PAPI_L1_DCM             11.380M/sec     611074379 misses
  PAPI_TLB_DM              0.222M/sec      11933068 misses
  PAPI_L1_DCA            510.886M/sec   27431996968 refs
  PAPI_FP_OPS            577.128M/sec   30988879856 ops
  User time (approx)      53.695 secs  123498500000 cycles
100.0%Time
  Average Time per Call                     0.000114 sec
  CrayPat Overhead : Time     1.2%
  HW FP Ops / User time   577.128M/sec   30988879856 ops
6.3%peak(DP)
  HW FP Ops / WCT         577.128M/sec
  Computational intensity    0.25 ops/cycle     1.13 ops/ref
  MFLOPS (aggregate)       577.13M/sec
  TLB utilization         2298.82 refs/miss    4.490 avg uses
  D1 cache hit,miss ratios  97.8% hits          2.2% misses
  D1 cache utilization (M)  44.89 refs/miss     5.611 avg uses
```

Update $\bar{D}$ for 1536 electrons

Ncores=2

USER / slater_update_dbar_.LOOP@li.2770

Time% 12.9%
Time 25.522353 secs
Imb.Time 0.021617 secs
Imb.Time% 0.3%
Calls 464.8 /sec 24017.0 calls
PAPI_L1_DCM 14.540M/sec 751360178 misses
PAPI_TLB_DM 0.267M/sec 13780336 misses
PAPI_L1_DCA 483.841M/sec 25002479245 refs
PAPI_FP_OPS 549.334M/sec 28386821099 ops
User time (approx) 51.675 secs 118852500000 cycles
100.0%Time
Average Time per Call 0.001063 sec
CrayPat Overhead : Time 0.1%
HW FP Ops / User time 549.334M/sec 28386821099 ops
6.0%peak(DP)
HW FP Ops / WCT 549.334M/sec
Computational intensity 0.24 ops/cycle 1.14 ops/ref
MFLOPS (aggregate) 549.33M/sec
TLB utilization 1814.36 refs/miss 3.544 avg uses
D1 cache hit,miss ratios 97.0% hits 3.0% misses
D1 cache utilization (M) 33.28 refs/miss 4.160 avg uses

Ncores=4

USER / slater_update_dbar_.LOOP@li.2770

Time% 12.7%
Time 23.751722 secs
Imb.Time 0.118560 secs
Imb.Time% 0.9%
Calls 253.3 /sec 24017.0 calls
PAPI_L1_DCM 4.040M/sec 383077467 misses
PAPI_TLB_DM 0.075M/sec 7143074 misses
PAPI_L1_DCA 131.908M/sec 12508535695 refs
PAPI_FP_OPS 149.676M/sec 14193410550 ops
User time (approx) 94.828 secs 218103250000 cycles
100.0%Time
Average Time per Call 0.000989 sec
CrayPat Overhead : Time 0.1%
HW FP Ops / User time 149.676M/sec 14193410550 ops
1.6%peak(DP)
HW FP Ops / WCT 149.676M/sec
Computational intensity 0.07 ops/cycle 1.13 ops/ref
MFLOPS (aggregate) 149.68M/sec
TLB utilization 1751.14 refs/miss 3.420 avg uses
D1 cache hit,miss ratios 96.9% hits 3.1% misses
D1 cache utilization (M) 32.65 refs/miss 4.082 avg uses

Update $\bar{D}$ code

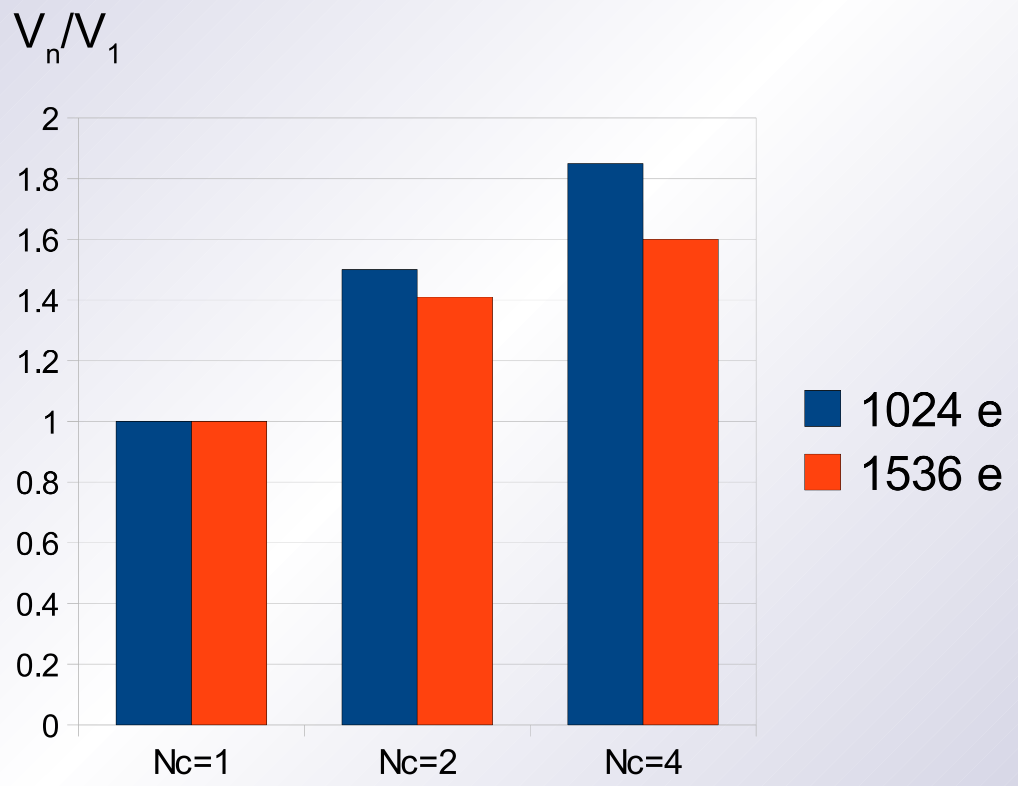
$$\bar{D}_{kj} = \begin{cases} \bar{D}_{kj}/q_i, & \text{if } j=i \\ \bar{D}_{kj} - \dfrac{\bar{D}_{ki}}{q_i}\left[\sum_{l=1}^{N} \bar{D}_{lj}\phi_l(\boldsymbol{r}_i)\right], & \text{if } j \neq i \end{cases}$$

$$q_i = \sum_{j=1}^{N} \bar{D}_{ji}\phi_j(\boldsymbol{r}_i)$$

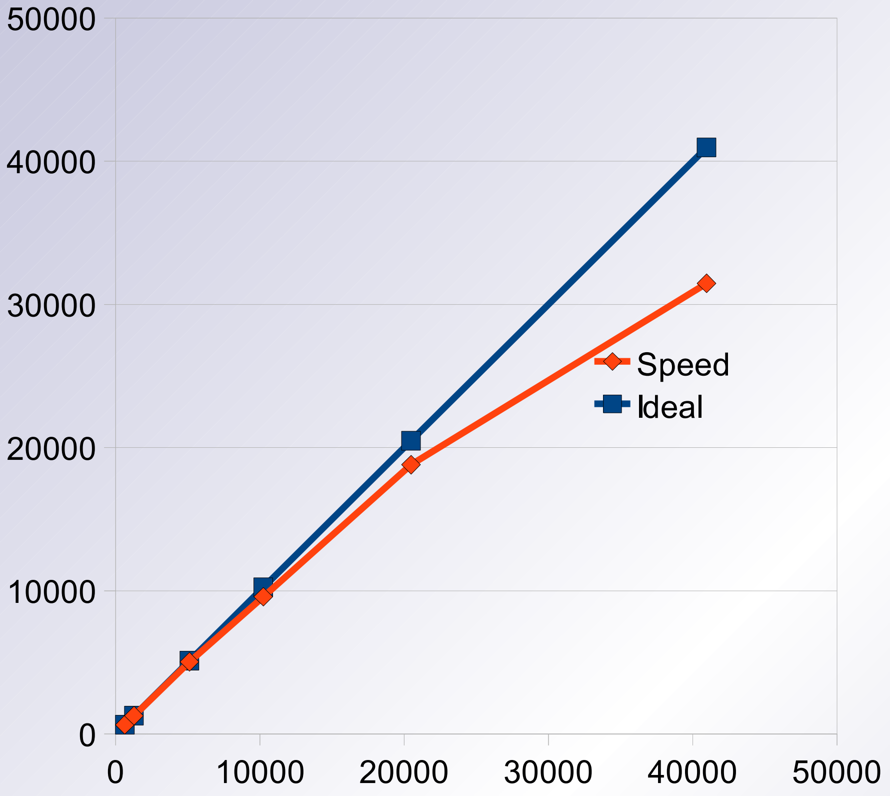
Code:

```
!$OMP PARALLEL DO DEFAULT(none) SHARED(ie,one_over_q_real,nele_uspin,&
!$OMP &dbar,uspin,rpsi) PRIVATE(je,tempr)
    do je=1,nele_uspin
     if(je==ie)cycle
     tempr=-one_over_q_real*ddot(nele_uspin,dbar(1,je,1,uspin),1,rpsi(1,1),1)
     call daxpy(nele_uspin,tempr,dbar(1,ie,1,uspin),1,dbar(1,je,1,uspin),1)
    enddo ! je
!$OMP END PARALLEL DO
```

Aggregated DMC performance



IO at large scale



Improvements:

- OPO data reorder and read/write in binary format (FORTRAN or MPIIO)
- config.in read only by small group of cores

  and data distributed with MPI.

CASINO scaling on Jaguar(ORNL) June 2009.

Conclusions

- The System V shared memory solution allows sharing of the orbitals data and therefore each core of a processor can run a computing task even for very large scale models.
- The input optimisations have eliminated unnecessary waiting time which wasted approximately 200 AU for each 1000 cores in a run.
- The computation using mixed mode second level parallelism reaches speed up factor close to 1.8 on quad core processor for models with a more than 1000 electrons.