The COLUMBUS Project:
General Purpose *ab-initio* Quantum Chemistry
Parallelization & Performance Issues

Thomas Müller

John von Neumann Institute for Computing
Central Institute for Applied Mathematics
Research Centre Jülich
D-52425 Jülich, Germany

http://www.fz-juelich/zam

Parallel Programming Model

- Data-centered model based on the **Global Array Toolkit** (PNNL) exploiting:

  ease of administration of distributed data while explicit exploitation of data locality is possible
  unified treatment of shared memory and distributed memory usage
  collective operations wrappers to MPI
  one-sided communication via ARMCI (low-level network support)
  user-level process-based

- Data may be classified as

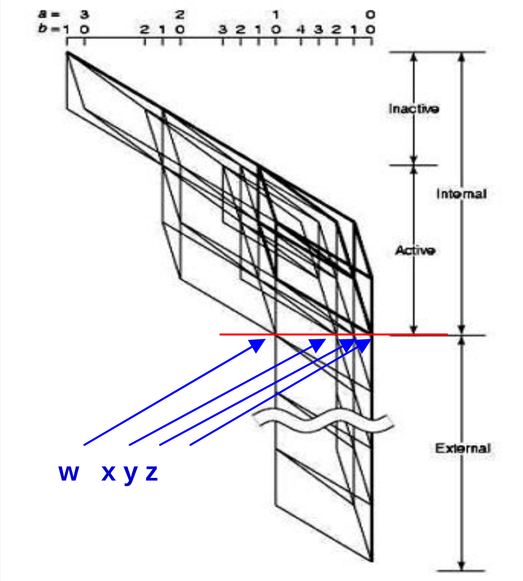
  fully distributed in global arrays (blocked & non-blocked one-sided access)
  one memory copy per SMP node (directly accessible in memory)
  one memory copy per process (virtual disk)
  local disk (either individually or shared by multiple processes)

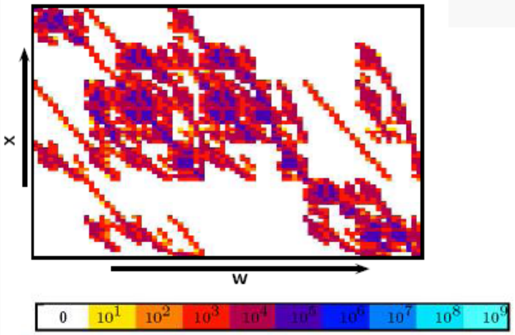
- Coarse-grain parallelization

  fine-grain parallelization exploiting parallel linear algebra highly inefficient here
  task definition arises naturally from the GUGA ansatz to MR-CI

- Supported platforms: all platforms supported by the Global Arrays Toolkit and MPI
- Languages: Fortran77/Fortran90, perl

Performance Model



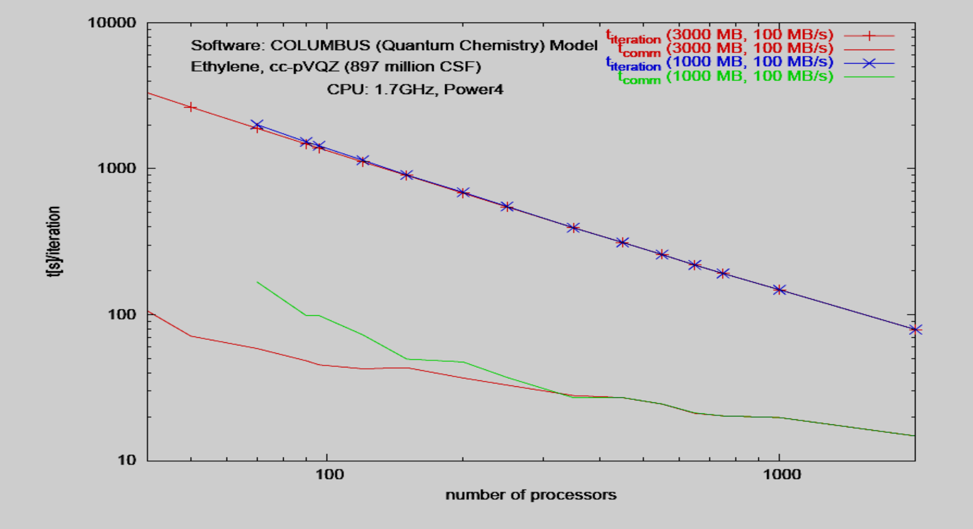


relative computational cost (2-ext)
(over blocks of valid internal walks)

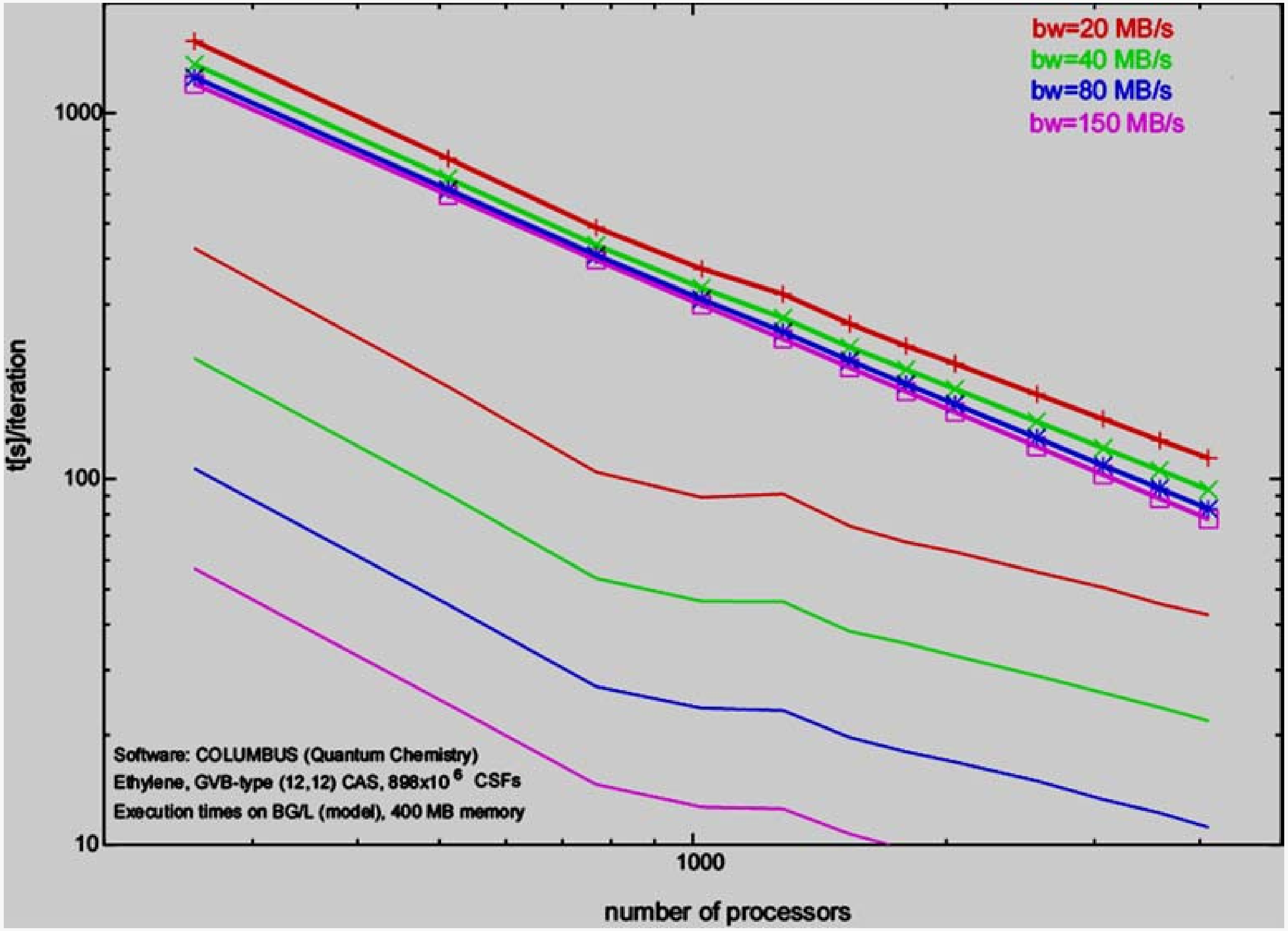
- extreme imbalance of computational due to sparsity of H
- forces dynamic load balancing
- sparsity reflected by the #valid internal walk *pairs* n_{iwp}
- computational cost

$$t_{total} = t_{comm}\,(v+w+I_{abcd}) + t_{internal} + t_{external}$$

- task definition: v, w segment, integral type I_{abcd}
- t_{comm}~ data volume/eff. averaged bandwidth
- $t_{internal}$ = n_{iwp} * average cost per valid internal walk pair
- $t_{contract}$ = n_{iwp}*average cost per valid internal walk pair

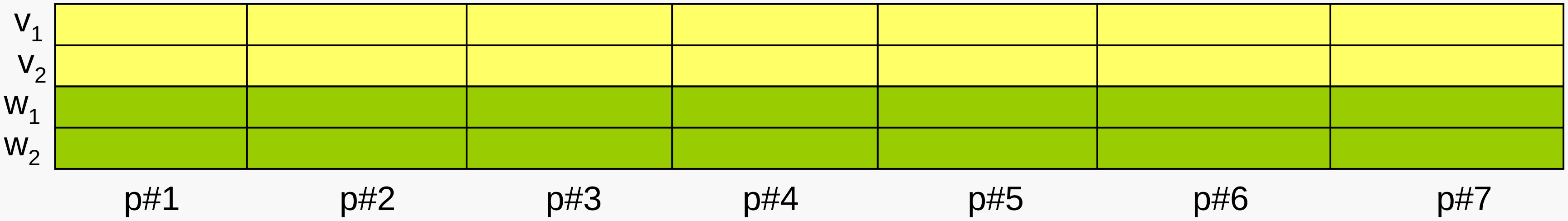


Performance Model

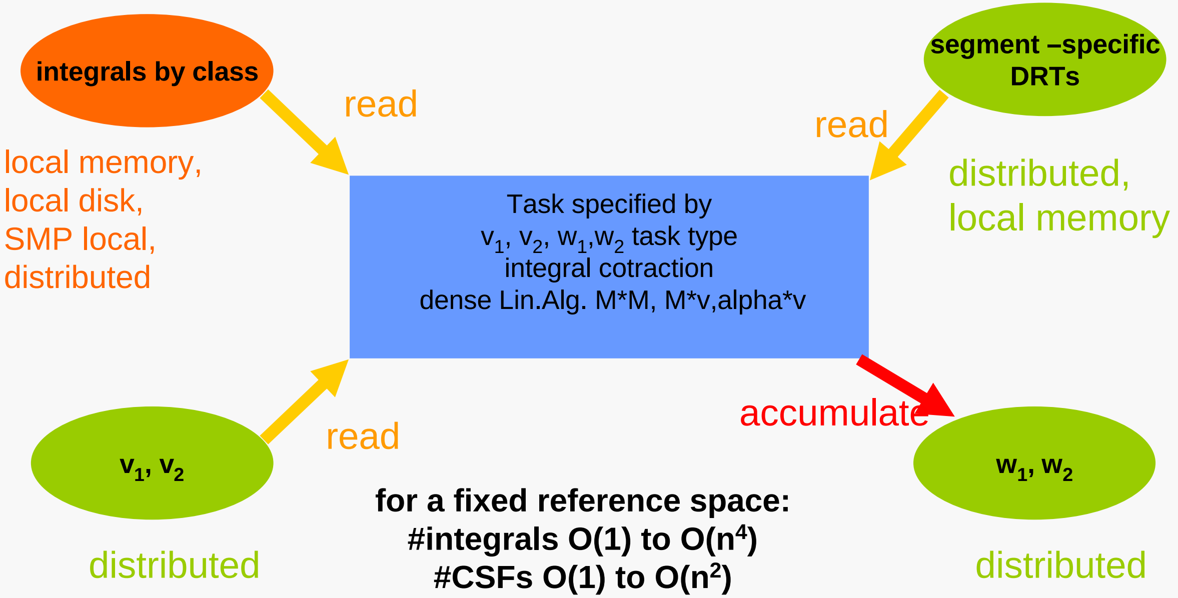


Software: COLUMBUS (Quantum Chemistry)
Ethylene, GVB-type (12,12) CAS, 898x10^6 CSFs
Execution times on BG/L (model), 400 MB memory

Communication & I/O Patterns



„subspace operations“ completely local, no I/O



local memory,
local disk,
SMP local,
distributed

Task specified by
v_1, v_2, w_1,w_2 task type
integral cotraction
dense Lin.Alg. M*M, M*v,alpha*v

distributed,
local memory

distributed

distributed

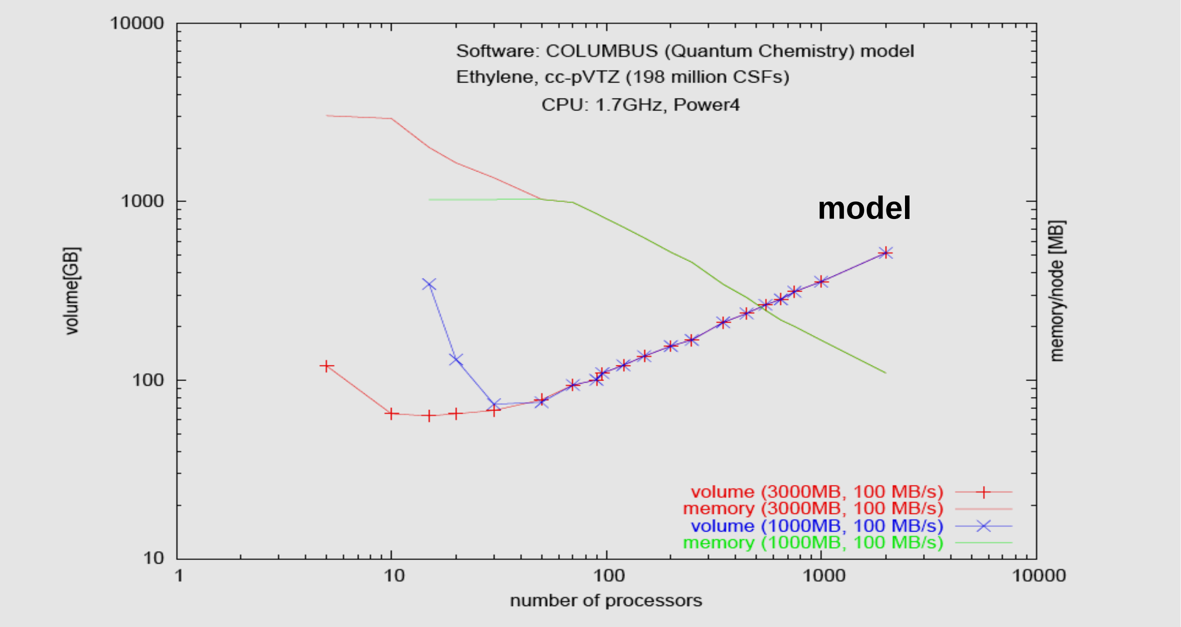
for a fixed reference space:
#integrals O(1) to O(n^4)
#CSFs O(1) to O(n^2)

formation of w vector:
non-local data transfer
total data volume ~ O(ncpu)

no intermediate I/O
(except for opt. local disk)

Communication and I/O Volume

- partitioning employing performance model: external constraints memory and effective bandwidth
- keeping integrals partially replicated & sparsity of H yields linearly increasing comm. volume



Input/Output Data

integrals (MO basis) O(n^4) of the order of GB (IN)

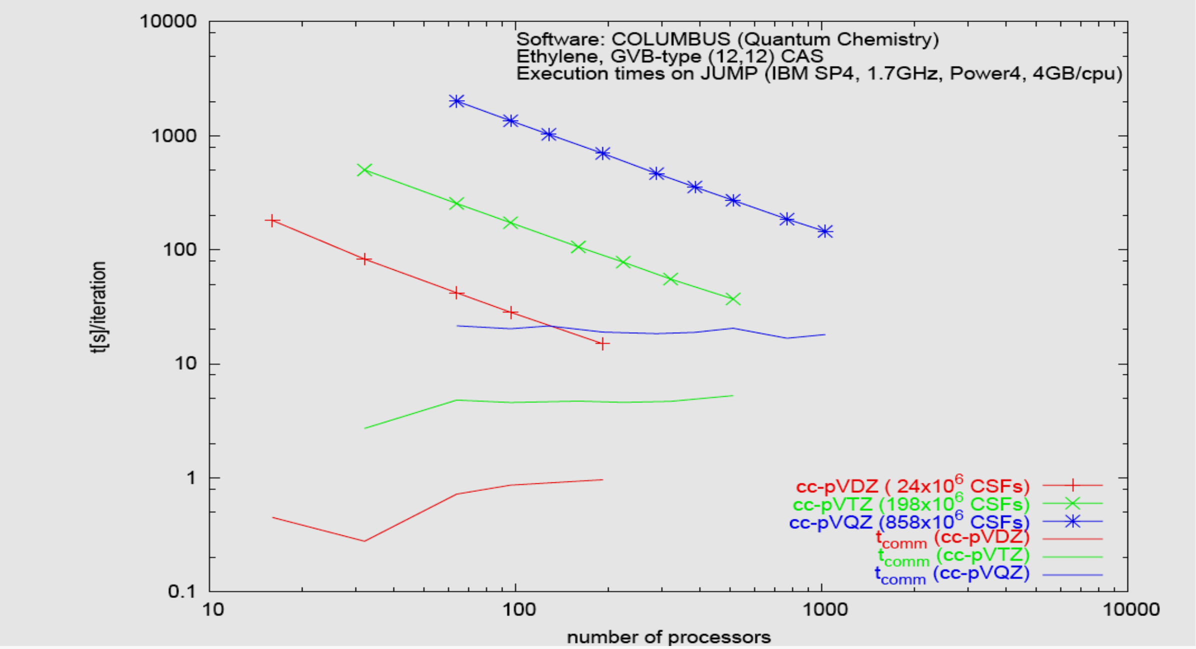
wavefunction expansion (CI vector) – N_CSF/2^{27} GB   typically 10 GB (OUT)

one-electron density matrix O(n^2)   negligible (OUT)

two-electron density matrix O(n^4)   of the order of GB   (OUT)

checkpointing disabled in favour of reduced I/O

Status and Scalability



- Porting and improved performance on BG/P, performance issues with IBMs ARMCI support
- Resolving potential message collision problems by different data distribution schemes possibly replacing dynamic by semi-static loadbalancing
- GA support
- Ineffictive vendor-specific one-sided communication

Performance Analysis Tools

Bottlenecks:

- variability of timings for identical tasks
- hot spots (message collisions)
- cache and code optimization issues

- Tracing at MPI level:
  - too large trace files
  - no mapping to higher-level programming
- Automated performance analysis tools (TAU, KOJAK, SCALASCA)
  - no support for one-sided access (ARMCI)
- simple task & cpu-specific perf. analysis via hardware counters

```
Ethylen.qz 1024 cpu, 145 secs wall clock, 6616 tasks
IBM eserver p690 (32 32-way SMP nodes)

-----------------------------------------------------
time           t_task         t_cont    t_comm(vw)
range         av     dev        av   dev     av      dev
-----------------------------------------------------
 0.062         0     100        52   279      2      408
 0.125         0     224        69   428     21      745
 0.250         3     470       117   669    136     1158
 0.500        19     817       189   849    500     1539
 1.000       325    2630       790  1907   4471     2007
 2.000       844    1205       537   921   1253      222
 4.000      1657     702      1520   608    185       34
 8.000      1317     316       990   304     46       10
 16.000      795      46       737    44      1        0
 32.000      947       0       941     0      0        0
 64.000      709       0       661     0      0        0
```

```
Butadien.tz+++, 128cpu, 38 sec wall clock, 913 tasks
IBM BlueGene/L

---------------------------------------------------
time        t_task        t_cont        t_comm (vw)
 range     av      dev     av      dev     av     dev
---------------------------------------------------
 0.062      1       39      8      193      8      71
 0.125      0       82     27      131     11      92
 0.250      2      156     67       47     59     153
 0.500     34      199     69       26    199     215
 1.000    297      305    267       74    584     270
 2.000    185       40    134       25     42      19
 4.000    123       18     80        9      3       4
 8.000    243        8    226        9      0       0
16.000     27        1     23        1      0       0
```

Debugging

- Parallel debugger totalview
  suitable for a modest number of processes (2 to 16)
  due to the huge amount of internally generated data parallel debuggers are of limited help
  (small problem size, tracebacks, analysing a problem in a small code section)

- Bugs associated with data corruption or inconsistency
  best to crudely trace back computing simple checksums on the fly wrt reference data
  applicable to (parallel) calculations of any size
  fast and allows to quickly draw conclusions on possible causes.

Roadmap

For dynamics, PES scanning or geom. optimization calculations may be carried out at optimized (reduced) problem-specific accuracy

- typical CSF spaces: 10^7 to 10^9
- typical basis set sizes < 500 basis functions
- point group symmetry frequently absent

- aim: reduce the total turn-around time for a single-point calculation to 1 to 30 minutes
  to make such tasks practicalv

For benchmarking calculations, calculations on difficult systems (transition metal compounds), spin-orbit CI etc.

- typical CSF spaces: 10^9 to 10^{10}
- typical basis set sizes < 1000 basis functions
- point group symmetry possibly exploited

- aim: make such calculations possible at all; however general MR-SDCI is most flexible
  but not necessarily the most efficient approach

The End.