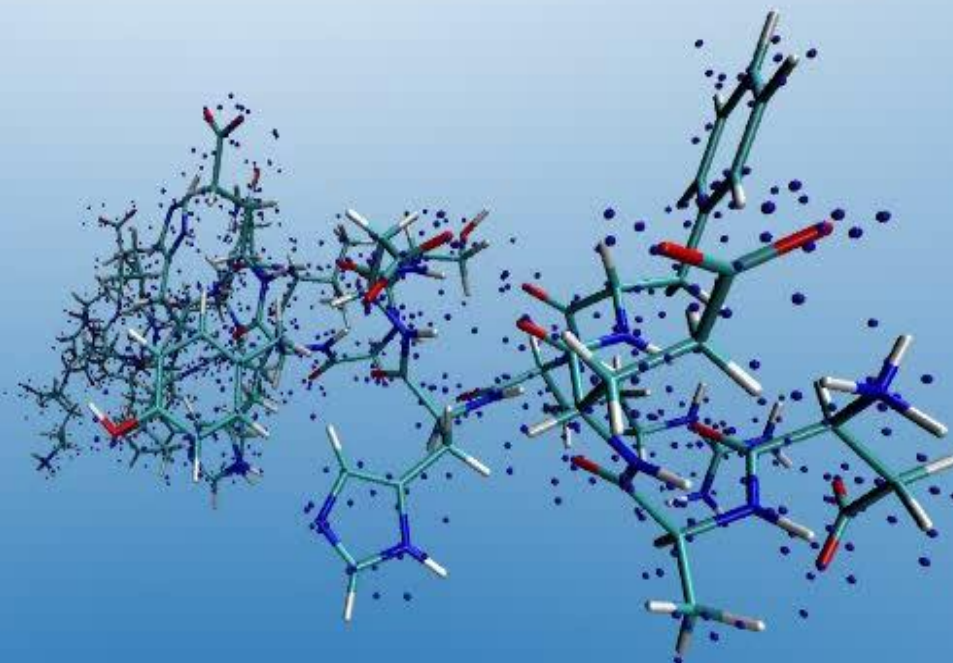


Quantum chemistry towards exascale with QMC=Chem



**A. Scemama,
M. Caffarel**

Laboratoire de Chimie et
Physique Quantiques
CNRS - IRSAMC
Université de Toulouse,
France

**E. Oseret,
W. Jalby**

Exascale Computing
Research Laboratory
GENCI-CEA-Intel
Université de Versailles St
Quentin, France

Ab initio Quantum chemistry

- Many chemical problems need highly accurate models, using an *ab initio* quantum mechanical description (Configuration Interaction or Coupled Cluster methods).



- Such methods need a large amount of memory and disk space.

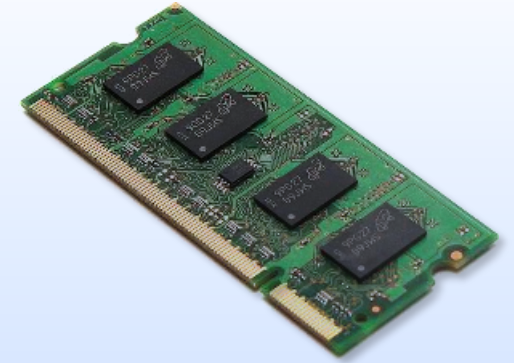
- Due to their iterative nature, synchronizations limit the parallel efficiency.
- These methods are not yet suited to massively parallel machines



Quantum Monte Carlo

Quantum Monte Carlo (QMC) are methods that :

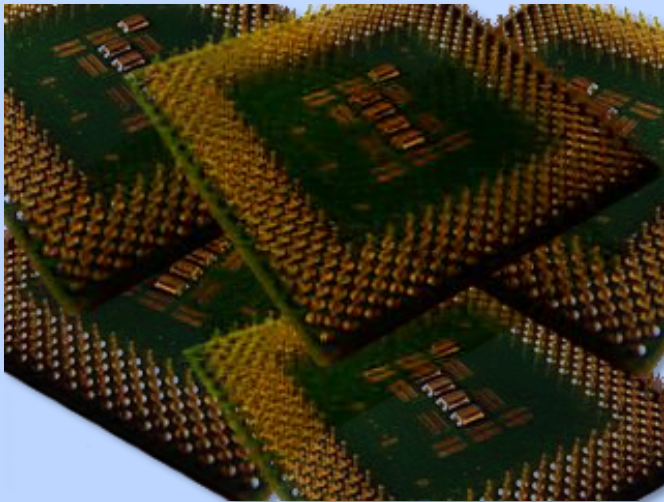
- require a small amount of memory (~100MB per core)
- make very few network communications
- have a much better scaling than standard methods with the size of the chemical system



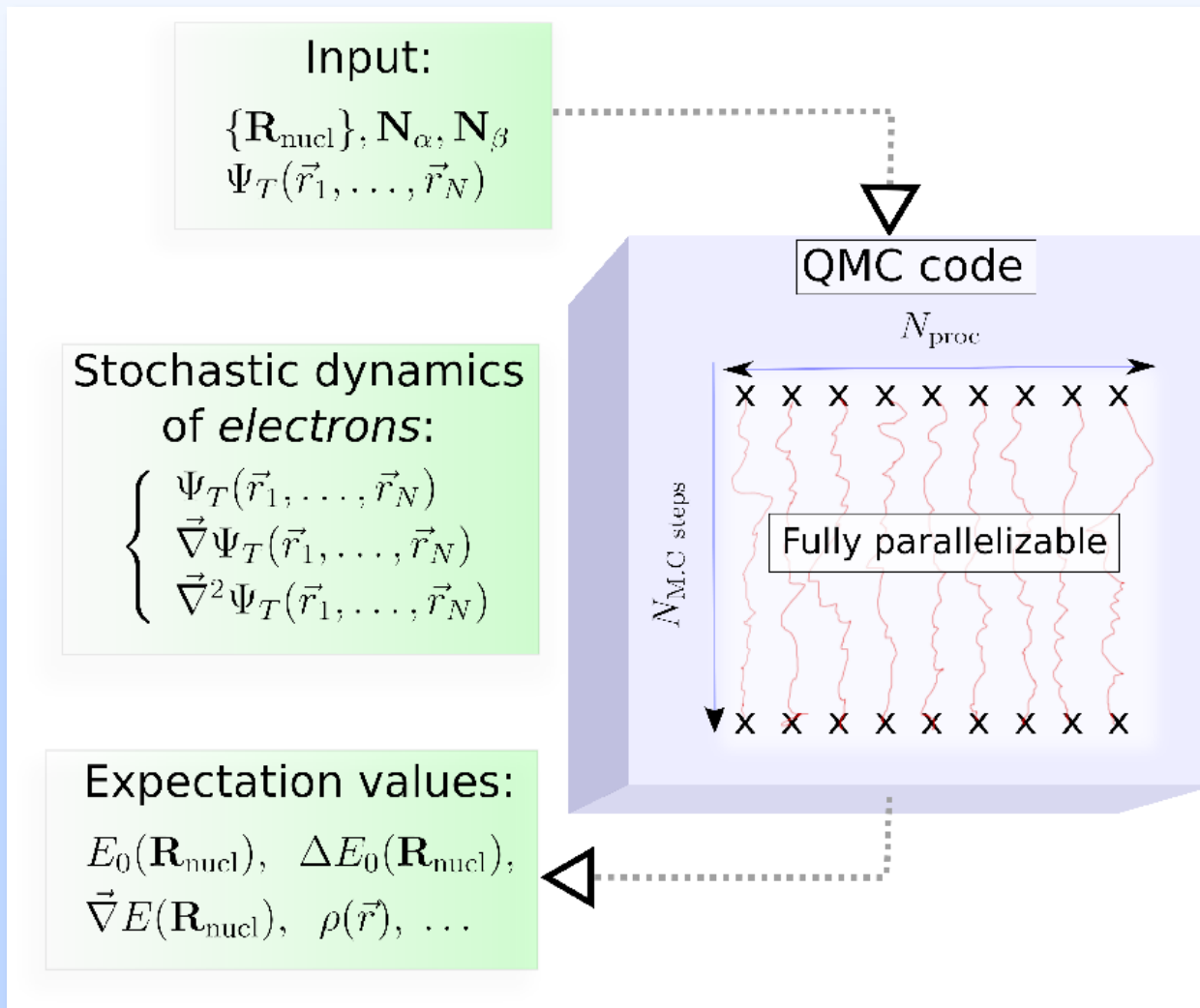
but:

require a large amount of CPU time

The evolution of massively parallel machines is very favorable to QMC methods



QMC methods are good candidates for exascale

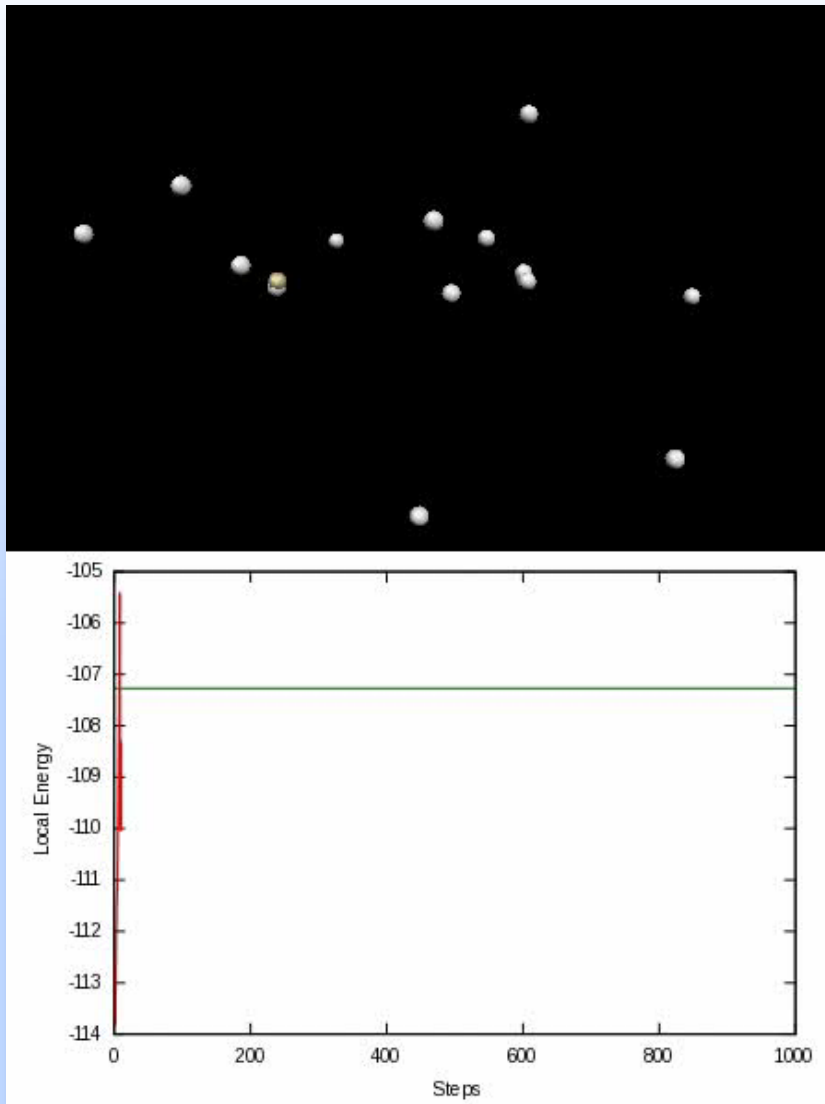


During the run, each process is completely autonomous (single core processes). The scaling is *ideal* !

Communications are mandatory *only* at the initialization and the finalization stages.

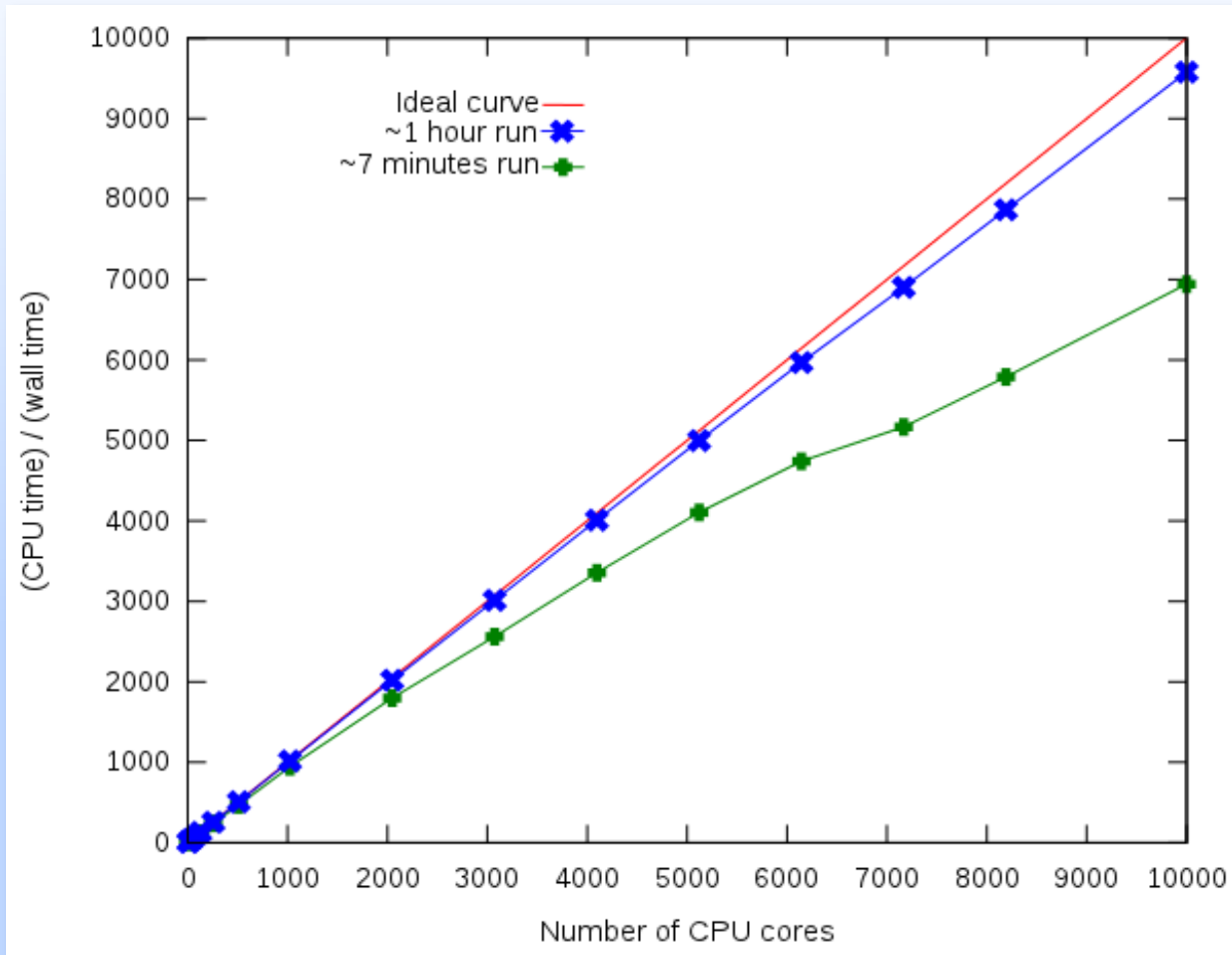
The initialization and finalization times don't depend on the length of the run.

QMC=Chem key points



- All the processes are completely independent
- Additional compute nodes can be added/removed dynamically to a running simulation
- Fault tolerance : any compute node can fail without killing the whole run
- I/O and network communications are fully asynchronous
- An almost ideal scaling with the number of cores is obtained
- Very good single core performance

Parallel efficiency of QMC=Chem



The performance of the application is determined by the efficiency of the *single-core* executable

Benchmarks performed on Curie (TGCC/CEA/GENCI, France) in April 2011.
The blue curve is estimated from the data collected on the green curve

Single-core performance

The scaling of one Monte Carlo step is limited by

- A matrix inversion, via the Intel MKL library ($O(N^3)$)
- Matrix-matrix products using a sparse-dense implementation ($O(N^2)$)

Two approaches to optimize the matrix product (ECR contribution)

- Static Analysis with MAQAO : Disassemble the binary and give information on the inner-most loops
- Decremental Analysis with DECAN : remove FP instructions or memory instructions from the binary and compare the timings with the real binary

Dense x Sparse Matrix multiplication

Static Analysis

```

!DIR$ VECTOR ALIGNED
do j=1,LDC
  C1(j,i)=C1(j,i)+(A(j,k_vec(1))*d11 &
    + A(j,k_vec(2))*d21 &
    + A(j,k_vec(3))*d31 &
    + A(j,k_vec(4))*d41)

  C2(j,i)=C2(j,i)+(A(j,k_vec(1))*d12 &
    + A(j,k_vec(2))*d22 &
    + A(j,k_vec(3))*d32 &
    + A(j,k_vec(4))*d42)
enddo
  
```

In green, SSE/AVX vectorized loops								
Nb used reg.			Performance in L1					
ID	XMM	YMM	Nb cycles	FLOP/cycle	IPC	Bytes loaded / cycle	Bytes stored / cycle	
1a	12	0	4	2	3	5	1	
1b	0	12	10	12.8	2.1	32	6.4	
2a	16	0	8	2	3.125	3	1	
2b	0	10	10	12.8	2.1	32	6.4	
Nb used reg.			Performance in L1					
ID	XMM	YMM	Nb cycles	FLOP/cycle	IPC	Bytes loaded / cycle	Bytes stored / cycle	
1	0	15	8	16	3.125	24	8	
2	0	15	8	16	3.125	24	8	

MAQAO Static analysis before (top) and after (bottom) optimization

- Examine the two hottest loops with MAQAO
- FLOP/cycle was not optimal : 12.8 but could be 16 (AVX, 32 bits elements, perfect ADD / MUL balance)
- Loop count (LDC) is always a multiple of 8. Replacing loop count with its hard coded value allows the compiler to factor loads
- We obtained a theoretically perfect efficiency for these loops

DECAN: results on QMC=Chem

Function	Loop weight	original	MI	FP	MAQAO asymptotic	Speed-up over original		
						MI	MI 1B	FPI
sparse_full_mm5_	20.0%	517	449	289	224	1.1	1.2	1.8
	14.0%	389	285	289	224	1.3	1.1	1.3
	6.7%	241	205	201	140	1.2	1.2	1.2
	1.1%	1353	1197	33	280	1.1	1.1	NA
bld_ao_oned_bloc_	2.9%	109	69	57	112	1.6	1.8	1.9
provide_elec_dist_	1.4%	1801	1775	221	324	1.0	1.0	8.1
provide_ao_value_block_	1.2%	1213	1217	277	248	1.0	1.0	4.4
Invert (MKL)	16%							



Fully vectorized loops

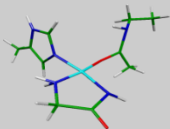
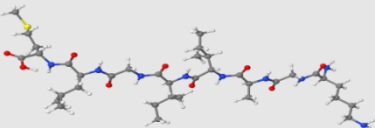
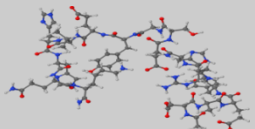
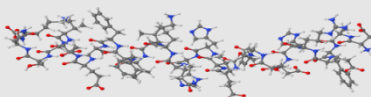
DECAN: results on QMC==Chem

- FP matching MI matching original (3 loops out of 7)
 - Code balanced, no clear bottleneck, no obvious optimization
- MI matching original (3 loops out of 7)
 - Loop dominated by memory accesses
- MAQAO Asymptotic match FP (4 loops out of 7)
 - No short vector issue
- Measurement accuracy challenged
 - 1 loop out of 7 is less than 100 cycles per instance
- Computational efficiency
 - MKL “invert” 54%
 - 3 hand written loops are 99%, 75% and 53%

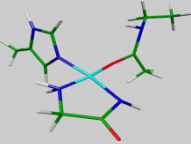
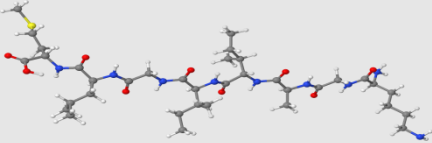
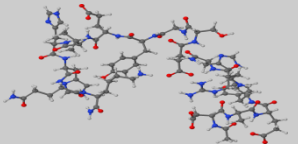
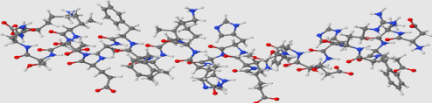
Single-core performance

The scaling of one Monte Carlo step is limited by

- A matrix inversion, via the Intel MKL library ($O(N^3)$)
- Matrix-matrix multiplications using an efficient sparse-dense implementation ($O(N^2)$)

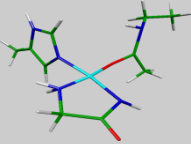
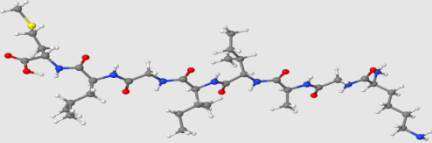
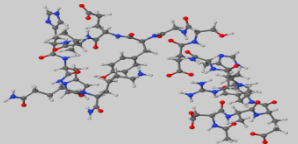
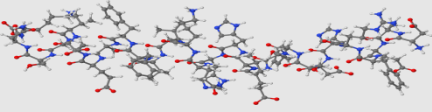
Molecular System	Number of electrons	% Peak performance of Multiplication	% Peak performance of Inversion	time(Inversion)/time(Multiplication)
	158	50%	24%	0.5
	434	64%	53%	0.6
	1056	58%	68%	2.0
	1731	53%	68%	2.6

Overall single-core performance

Molecular System	Number of electrons	RAM/core (MB)	CPU time / step on Core2 ¹ (s)	CPU time / step on Sandy Bridge ² (s)
	158	9.8	0.0073	0.0033
	434	65	0.0504	0.0186
	1056	133	0.3421	0.0980
	1731	313	1.2480	0.4226

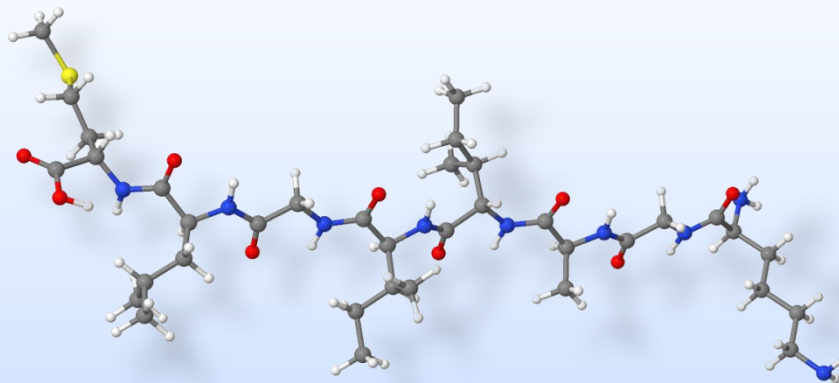
1. Intel Xeon 5140, Core 2 2.33GHz, Dual core, 4MB shared L2 cache
2. Intel Xeon E31240, Sandy Bridge 3.30GHz, Quad core, 256KB L2 cache/core, 8MB shared L3 cache

Overall single-core performance

Molecular System	Number of electrons	RAM/core (MB)	% CPU Peak performance on Core2 ²	% CPU Peak performance on Sandy Bridge ²
	158	9.8	25%	23%
	434	65	34%	38%
	1056	133	37%	49%
	1731	313	47%	55%

1. Intel Xeon 5140, Core 2 2.33GHz, Dual core, 4MB shared L2 cache
2. Intel Xeon E31240, Sandy Bridge 3.30GHz, Quad core, 256KB L2 cache/core, 8MB shared L3 cache

Latest performance results



We have modelled a peptide involved in Alzheimer's disease on a BullX supercomputer:

- Intel Sandy Bridge sockets, 2.7GHz, 8 cores, 20MB cache
- 190 dual socket nodes (3 040 cores)
- 64GB RAM/node

An average of 27.8 TFlops/s was obtained on a 7 minutes run.

For a 1 hour run we would obtain 31.2 TFlops/s.

We would be able to reach 1PFlops/s with 97 500 cores for 1 hour.