Some issues in large scale computing

Gérard MEURANT
CEA
France
(gerard.meurant@cea.fr)
Beijing, September 2004

September 1, 2004

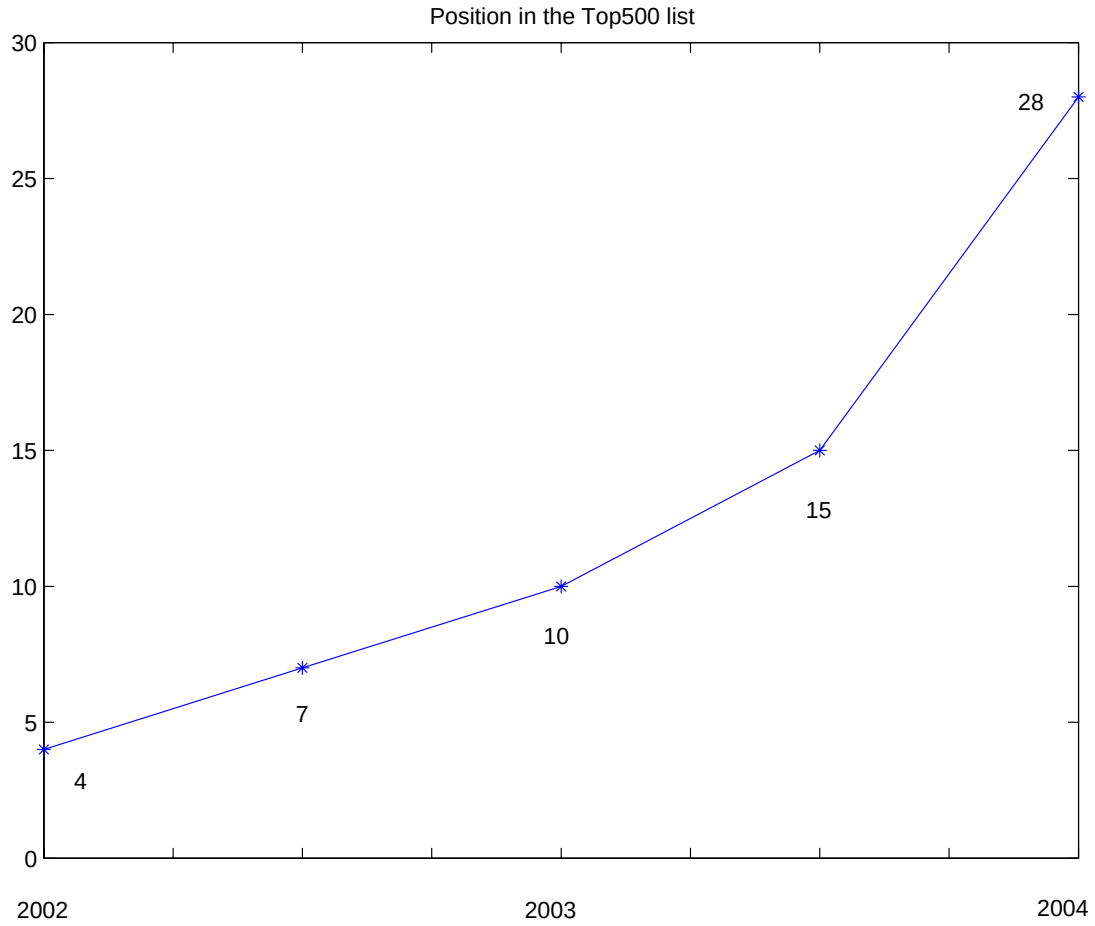
J.L. Lions has always been interested in High Performance Computing (HPC)

He was very influential in the creation of ORAP in 1994, www.irisa.fr/orap

- association between CEA, CNRS (National center for scientific Research) and INRIA
- aim: promote the use of parallelism for high performance scientific computing in France and Europe

- There are many Tflops peak parallel machines around the world each with hundreds or thousands of processors
- The 50th ranked machine in the last Top 500 list has a performance of 2.9 Tflops on Linpack
- The fastest one is still the Earth Simulator with a Linpack performance of 35.86 Tflops
- Although we can discuss the relevance of Linpack as a criterion, it means that there is a large potential of computing cycles available

Things are changing quite fast:



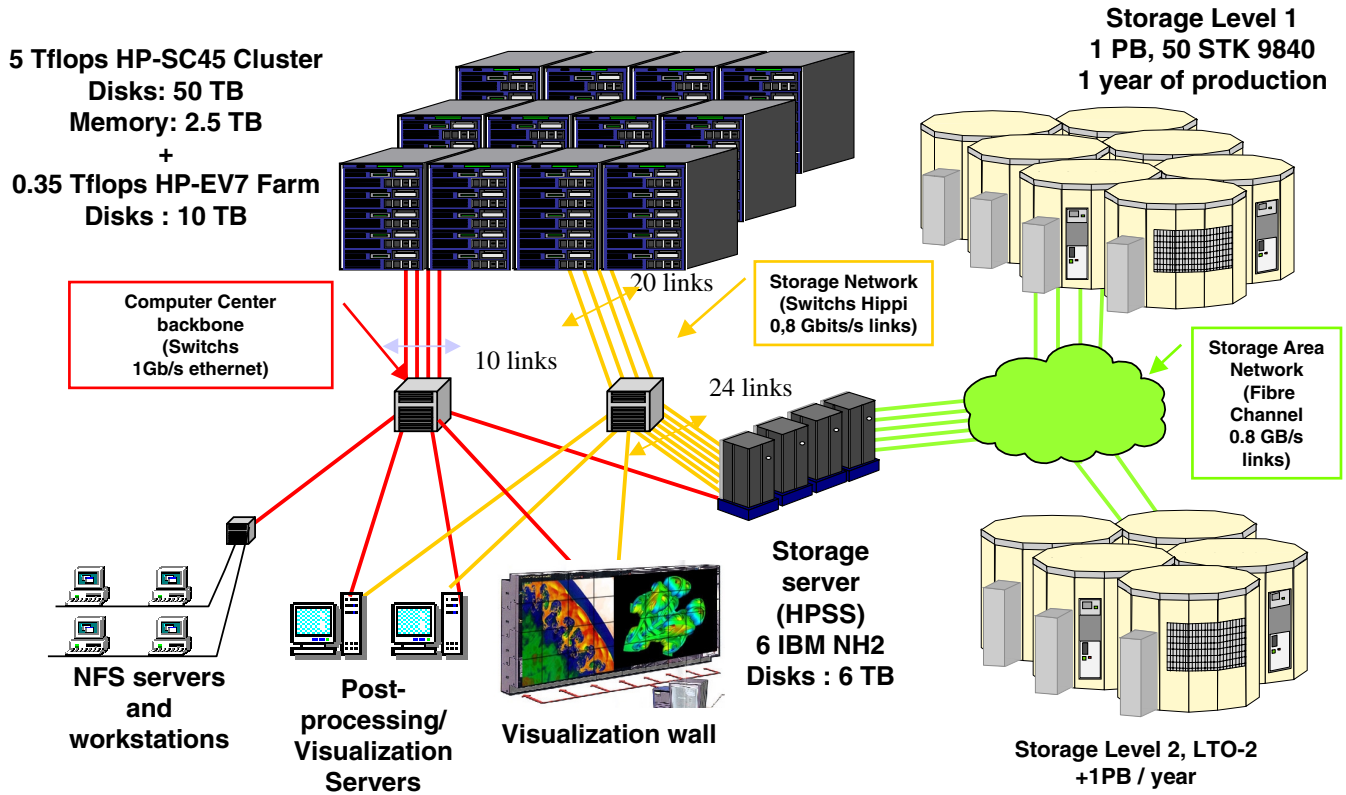
Top 500 rank of the largest CEA parallel computer

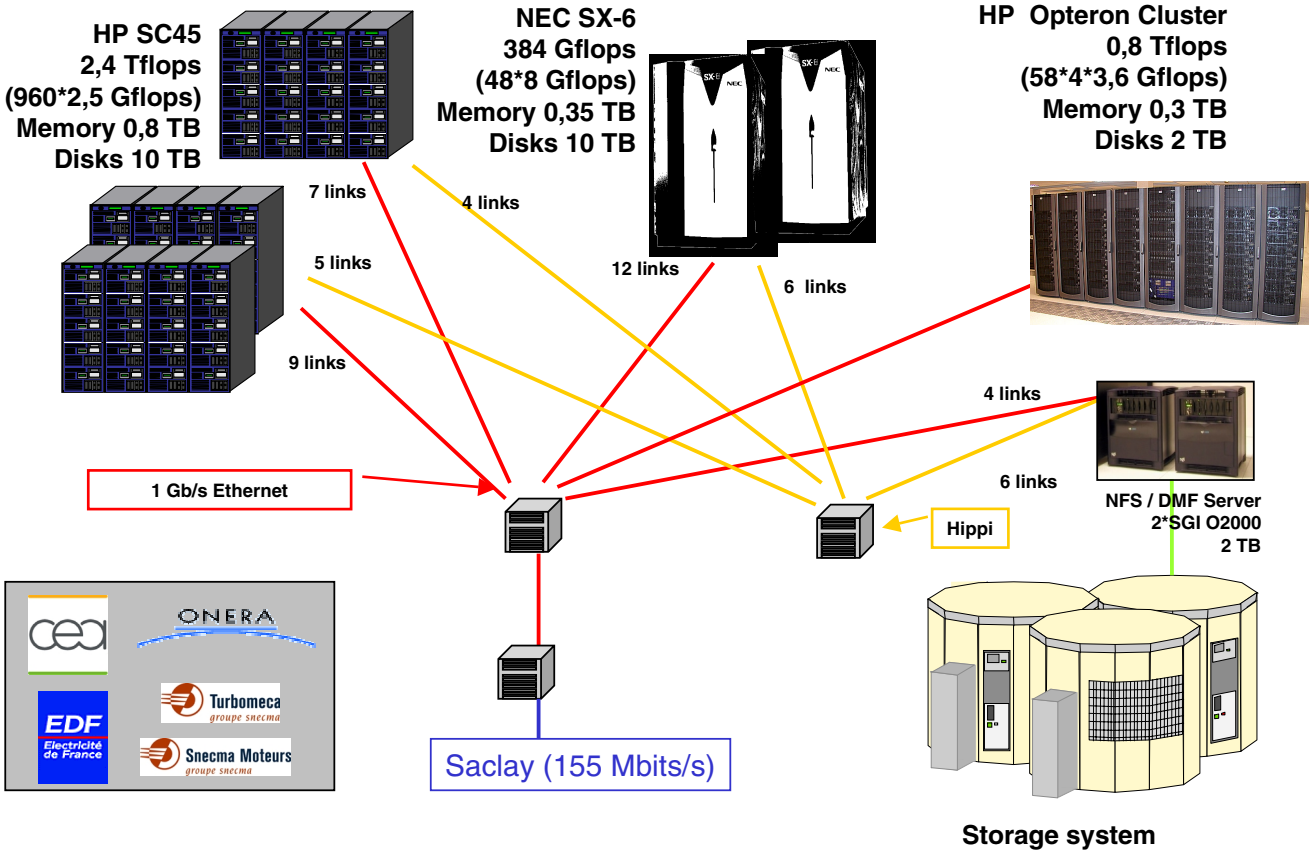
The questions we would like to discuss are:

- How difficult is it to obtain good performances on these machines?
- How efficiently could we use these parallel computers on large practical applications?
- What are the problems with I/O, visualization etc. . . ?

The CEA computing center

- 4 large machines
  - HP 2560 Alpha processors
  - HP 960 Alpha processors
  - HP 232 AMD Opteron processors
  - NEC SX-6 48 vector processors


5 Tflops HP-SC45 Cluster
Disks: 50 TB
Memory: 2.5 TB
+
0.35 Tflops HP-EV7 Farm
Disks : 10 TB
Storage Level 1
1 PB, 50 STK 9840
1 year of production
Computer Center backbone (Switchs 1Gb/s ethernet)
20 links
Storage Network (Switchs Hippi 0,8 Gbits/s links)
10 links
24 links
Storage Area Network (Fibre Channel 0.8 GB/s links)
Storage server (HPSS)
6 IBM NH2
Disks : 6 TB
NFS servers and workstations
Post-processing/ Visualization Servers
Visualization wall
Storage Level 2, LTO-2
+1PB / year


HP SC45
2,4 Tflops
(960*2,5 Gflops)
Memory 0,8 TB
Disks 10 TB
NEC SX-6
384 Gflops
(48*8 Gflops)
Memory 0,35 TB
Disks 10 TB
HP Opteron Cluster
0,8 Tflops
(58*4*3,6 Gflops)
Memory 0,3 TB
Disks 2 TB
7 links
4 links
5 links
12 links
6 links
9 links
4 links
6 links
1 Gb/s Ethernet
Hippi
NFS / DMF Server
2*SGI O2000
2 TB
ONERA
Turbomeca
groupe snecma
EDF
Electricité de France
Snecma Moteurs
groupe snecma
Saclay (155 Mbits/s)
Storage system

- Some of the machines are used by industrial partners
  - EDF (Power supply)
  - SNECMA (Airplane engines)
  - ONERA (National Lab for Aerospace research)
  - TURBOMECA

Largest CEA machine

HP COMPAQ SMP

- 640 nodes, 2560 processors Alpha EV68 1Ghz
- Peak performance: 5 Tflops (2 Gflops/proc)
- Memory 4 GB per node
- Quadrics interconnection network
- Needs a very large computer room: 1200 m^2 + 750 m^2 for the storage silos


COMPAQ

How difficult is it to obtain good performances on these machines?

- Even on one processor it is difficult to obtain a performance close to the peak:
  - Data flow between memory and caches is too slow to keep the processor busy
  - Compilers do not make the most efficient use of the hardware, particularly data cache reuse
  - This is why the Linpack benchmark makes use of optimized BLAS3 routines and is not representative of real applications

Communications

- Network latency is often quite small ?μs on the CEA machine. But:
  - Message Passing Interface (MPI) is not always efficiently implemented
  - Buffer problems
  - Traffic is perturbed by other jobs
  - ?

How efficiently could we use these parallel computers on large practical applications?

- Of course we need to have parallel algorithms
- An important issue is scalability
  - We would like the elapsed time to stay constant when we proportionally increase the size of the problem and the number of processors
  - This depends on the properties of the problem and the algorithms

Examples

- Time explicit schemes for CFD are easy to parallelize
  - Scalability depends on the relation between the time step and the spatial discretization
- Direct or iterative methods for solving linear systems are difficult to parallelize
  - Generally the complexity is $n^{\alpha}, \alpha > 1$
  - Of course, there are exceptions like multigrid or domain decomposition methods

What are the problems with I/O, visualization etc. . . ?

- CEA codes produce a lot of output data (3 TB per day) for check point/restart and visualization
- In a parallel application, each processor writes its own file
- Most parallel file systems are not very efficient
- A global reconstruction is often needed for visualization tools
- Very large viz files cannot be handled efficiently by commercial products

Examples of large scale computations

- Scalable linear systems solvers
- CFD Euler equations

Scalable linear systems solvers

$$Ax = b$$

A large sparse symmetric positive definite of order n

The iterative method must be (almost) scalable

We use Preconditioned Conjugate Gradient

The preconditioner must be such that:

- the number of iterations is (almost) constant, when the problem size is increased
- the complexity of applying the preconditioner is proportional to n
- easy to construct and use on a parallel computer

Two known possibilities:

- multilevel (algebraic multigrid–like) methods
- domain decomposition (DD)

Multilevel + DD techniques for parallelization

Multilevel preconditioners

◦ Algebraic methods (grid $\equiv$ (sub) set of unknowns without overlapping)

Starting from the zero vector:

0– if we are on the coarsest level, exact solve by Gaussian elimination or use diagonal CG, otherwise

1– do ν iterations of smoothing

2– restrict the residual r to $r_c = Rr$

3– recursively solve $A_c e_c = r_c$, $A_c = RAP$, $R = P^T$

4– interpolate e_c to $e = Pe_c$

5– add the correction e to the current iterate

6– do ν iterations of smoothing

Generally, we will use $\nu = 1$. We have to define:

- the smoother
- coarsening algorithm
- the interpolation

Many different choices!

Smoothers

◦ Symmetric Gauss–Seidel (not parallel)

parallelized by using Jacobi for the interface nodes (SGSJ)

◦ Incomplete Cholesky (IC not par. either) $M = LDL^T$

parallelized by ignoring dependencies between subdomains (ICp)

$$LD^{-1}L^T(x^{k+1} - x^k) = b - Ax^k$$

◦ Approximate inverse AINV from M. Benzi and al.
$M^{-1} = ZD^{-1}Z^T$ where Z is upper triangular and D is diagonal

Smoother: Richardson iteration defined as (matrix $\times$)

$$x^{k+1} = x^k + M^{-1}(b - Ax^k)$$

- Influence matrix

$$S_i^A = \{j \mid |a_{i,j}| > \tau \max_{k \neq i} |a_{i,k}|, \quad \tau < 1\}$$

- Coarsening algorithm

$$\mathcal{N} = F \cup C$$

  - Ruge-Stuben (Wagner) RS

parallized by subdomain plus a pass (partially sequential) on the interface nodes

  - Cleary, Falgout, Henson and Jones algorithms

- Standard interpolation algorithm using matrix entries, projection P, restriction $R = P^T$
- Coarse matrices

$$A_C = RAP$$

Numerical experiments on TERA

- 5 point finite differences, unit square, $m \times m$ mesh
- b random
- $x^0 = 0$
- stopping criterion $\|r^k\| \leq 10^{-10}\|r^0\|$
- Mesh decomposition into squares with m_p^2 unknowns per processor

• Partition the graph of A (mesh) into non overlapping subdomains but with ghost nodes

Nodes in relation to ghost nodes are interface nodes

• A is distributed by rows

Two problems

- Unit square
- Poisson equation
- Discontinuous diffusion problem

diffusion coeff=1 except 1000 in $[1/4\ 3/4]^2$

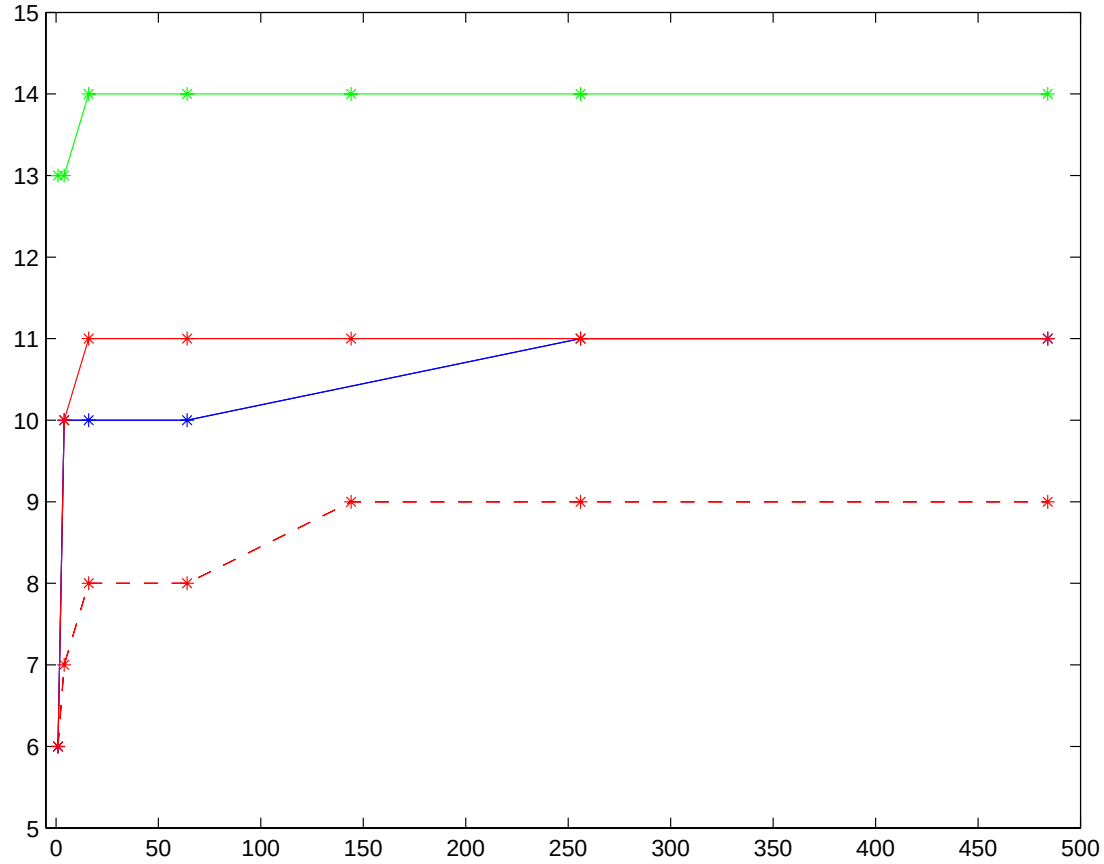
First experiment

- Poisson equation

$m_p = 250 \rightarrow 62500$ unknowns per processor,

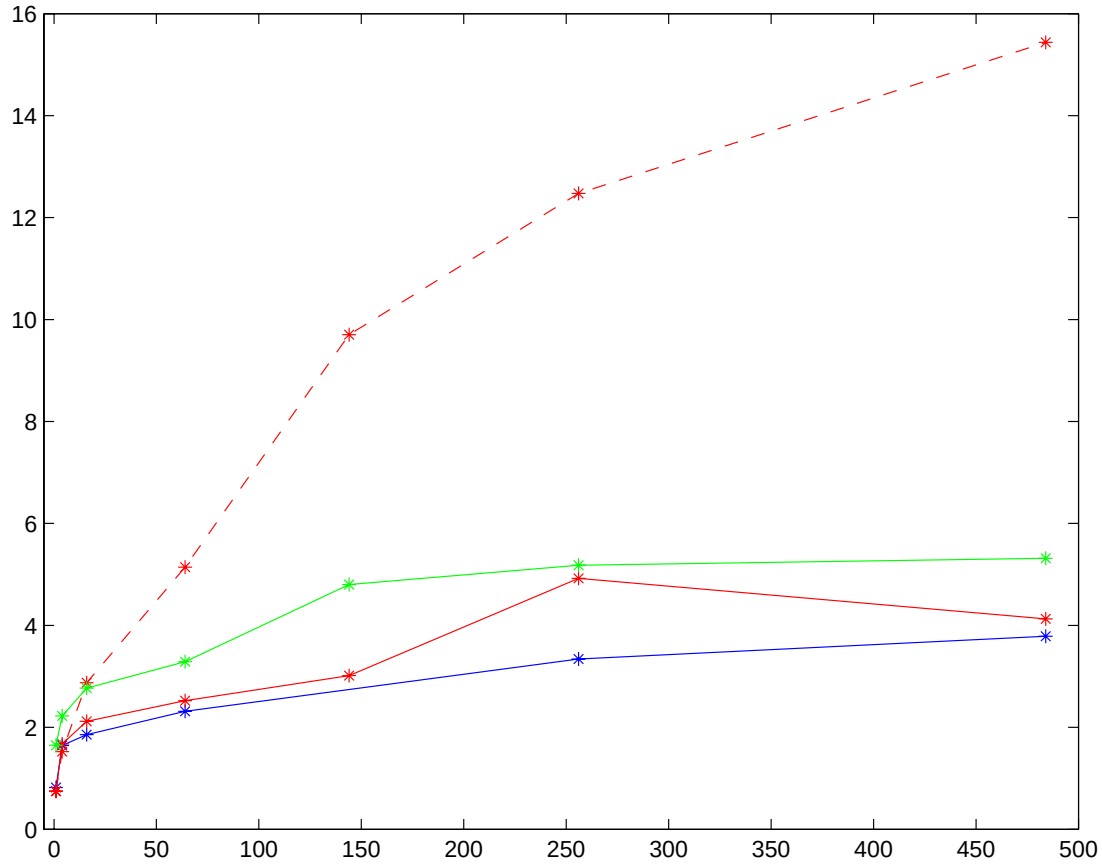
$p = 1, 4, 16, 64, 144, 256, 484$

- Largest problem is $\simeq 30\ 10^6$ unknowns



Nb of iterations for the Poisson equation as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp, 62500 unknowns per processor



Elapsed time (s) for the Poisson equation as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

Second experiment

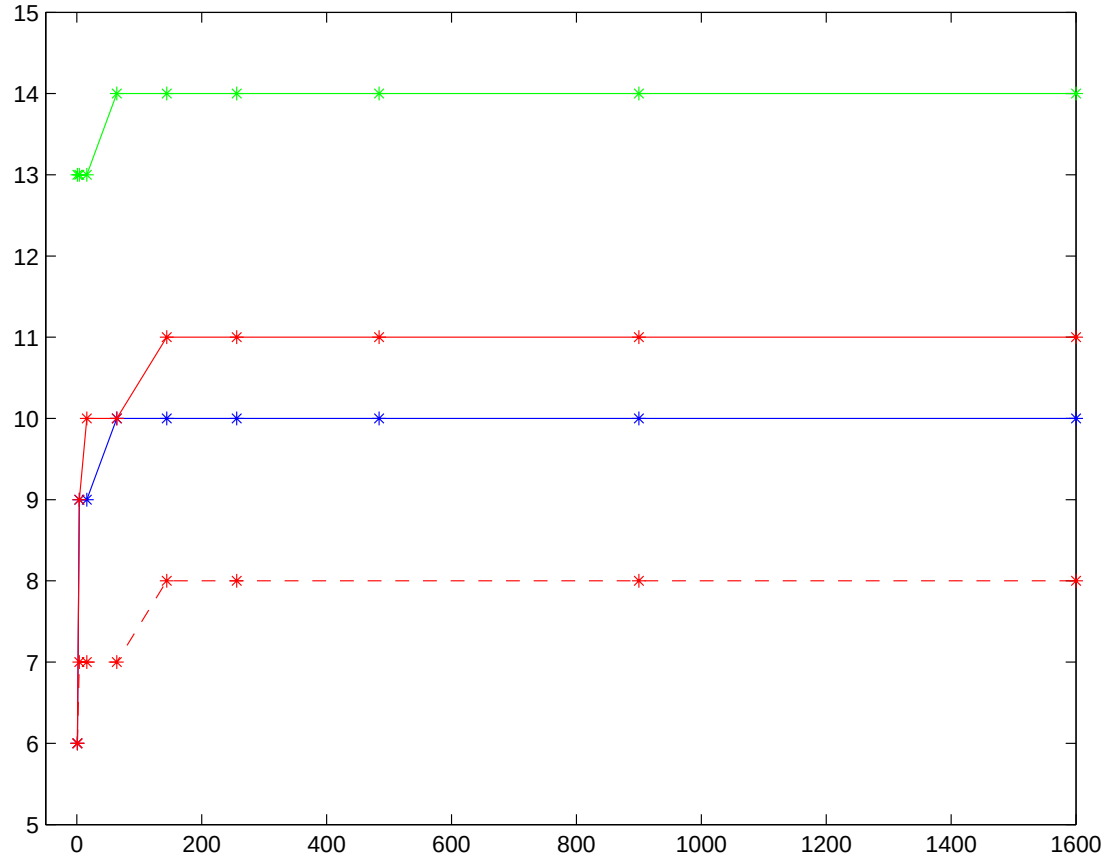
- Poisson equation

$m_p = 100 \rightarrow 10000$ unknowns per processor,

$p = 1, 4, 16, 64, 144, 256, 484, 900, 1600$

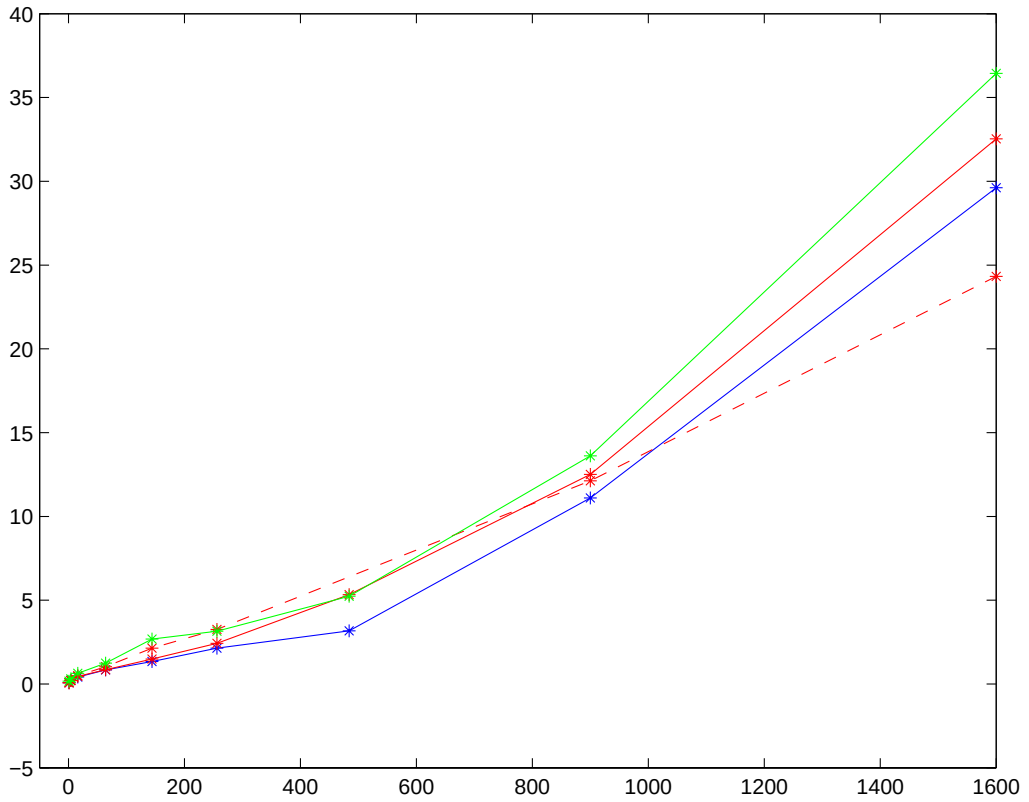
- Largest problem is $16\ 10^6$ unknowns

  - Note that the number of unknowns per processor is smaller than before



Nb of iterations for the Poisson equation as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp



Elapsed time (s) for the Poisson equation as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

This computation is dominated by communication

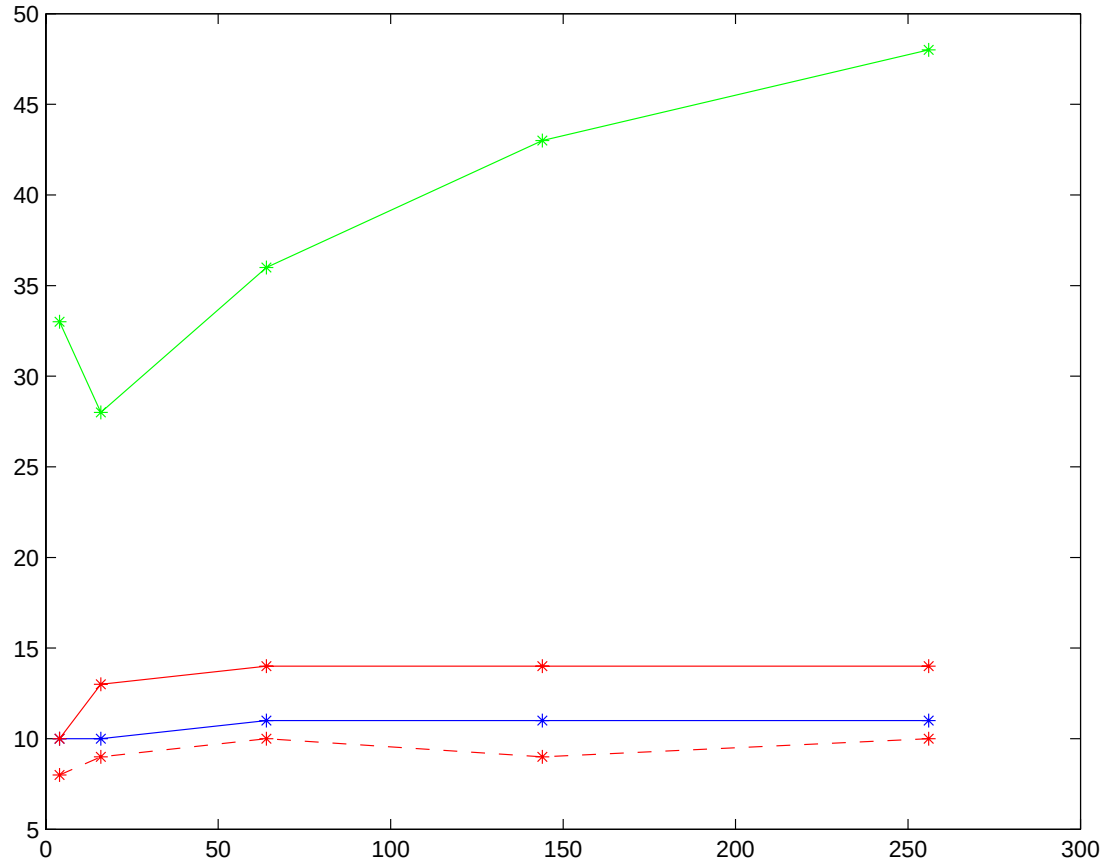
Third experiment

- Discontinuous problem

$m_p = 250 \rightarrow 62500$ unknowns per processor,

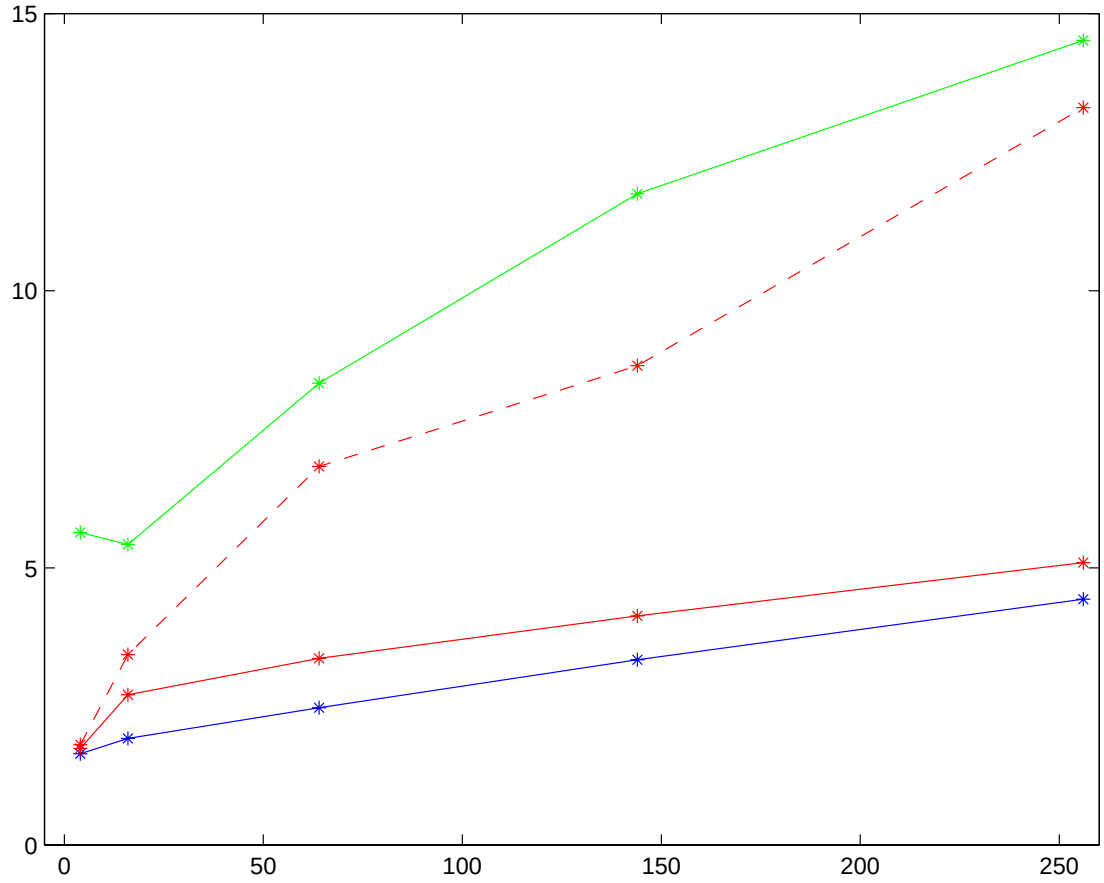
$p = 4, 16, 64, 144, 256$

- Largest problem is $16\ 10^6$ unknowns



Nb of iterations for the discontinuous problem as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp



Elapsed time (s) for the discontinuous problem as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

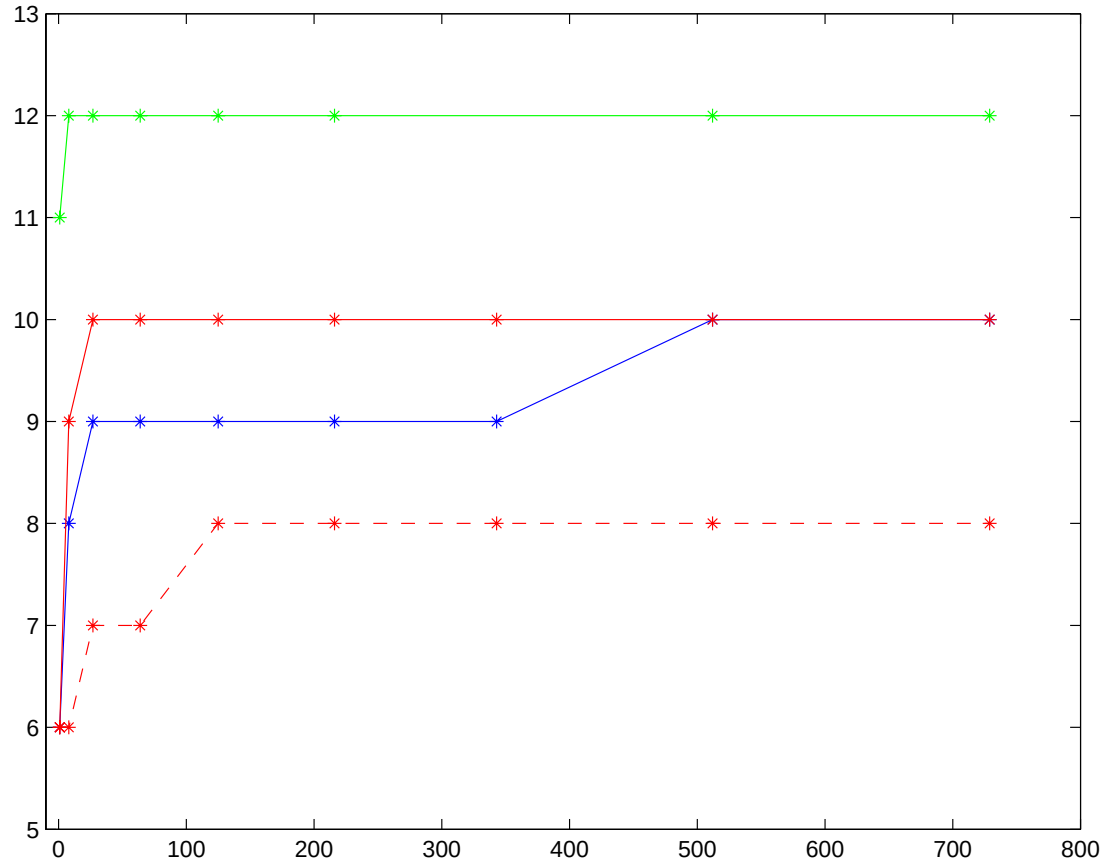
Last experiment

- 3D Poisson in $[0,1]^3$, 7 point finite differences

$m_p = 30 \rightarrow m_p^3 = 27000$ unknowns per processor,

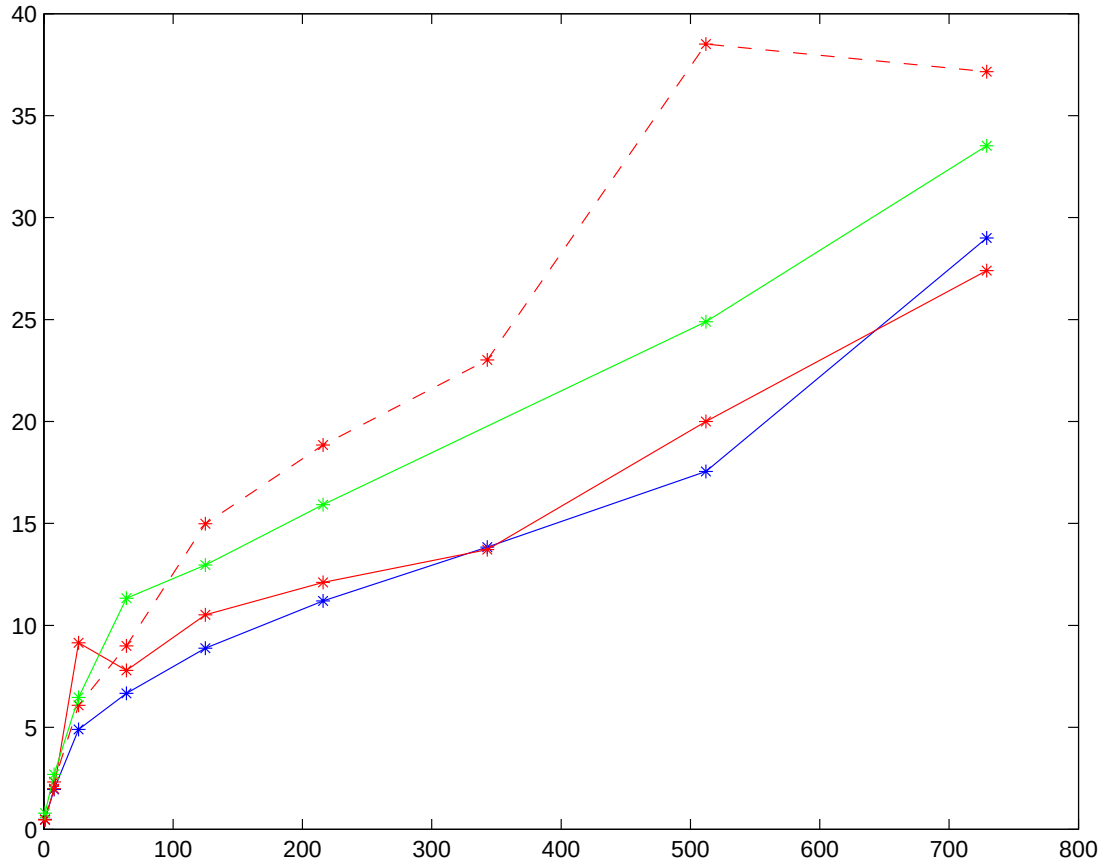
$p = 1, 8, 27, 64, 125, 216, 343, 512, 729$

- Largest problem is $\simeq 20\ 10^6$ unknowns



Nb of iterations for the 3D Poisson equation as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp



Elapsed time (s) for the 3D Poisson equation as a function of p

coarsening: LLNL, blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

- This library is still under development
- Algorithms are working fine. But:
  - Communications have to be optimized
  - Memory management has to be improved

The TERA Benchmark

- Specially designed Fortran 90 MPI code to check how efficient we could be on a real application using all the processors and all the memory
- 3D inviscid compressible Euler equations
- Alternating directions
- Transport + Projection time explicit Godunov scheme
- Approximate Riemann solver
- Order 3 projection (PPM–like)
- Parallelized by domain decomposition with ghost cells

Optimization

- On one processor:
  - 3 copies of arrays, one per spatial dimension to use cache efficiently
  - Avoid divisions and transcendental functions
  - Optimize the ratio memory accesses/floating point operations
- It is really difficult to obtain a very good performance on one processor

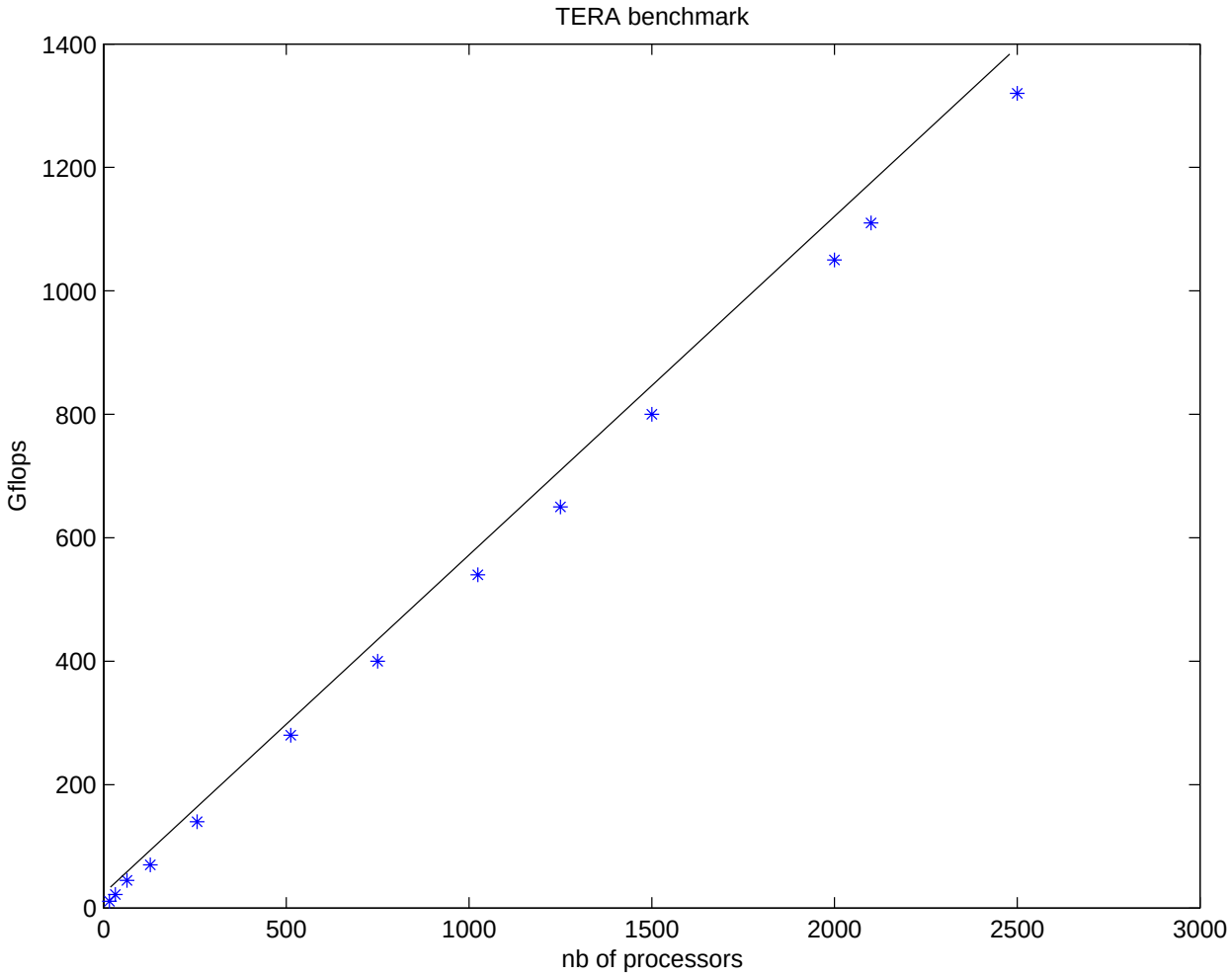
Communications

- One communication phase per time step
- Messages should not be too large (buffer problems) or too small (latency problems)
- Asynchronous non blocking messages of moderate size (64 KB)
- Communications 10 to 12% of total time

◦ We look at the scaled performance with a constant number of cells per processor, increasing the number of processors

◦ Available memory 650 MB per processor which leads to 4 to 5 million cells, subdomains: cubes of 160^3 cells

- Test case: shock tube
- Fastest computation: 10 billion cells, 1.32 Tflops,

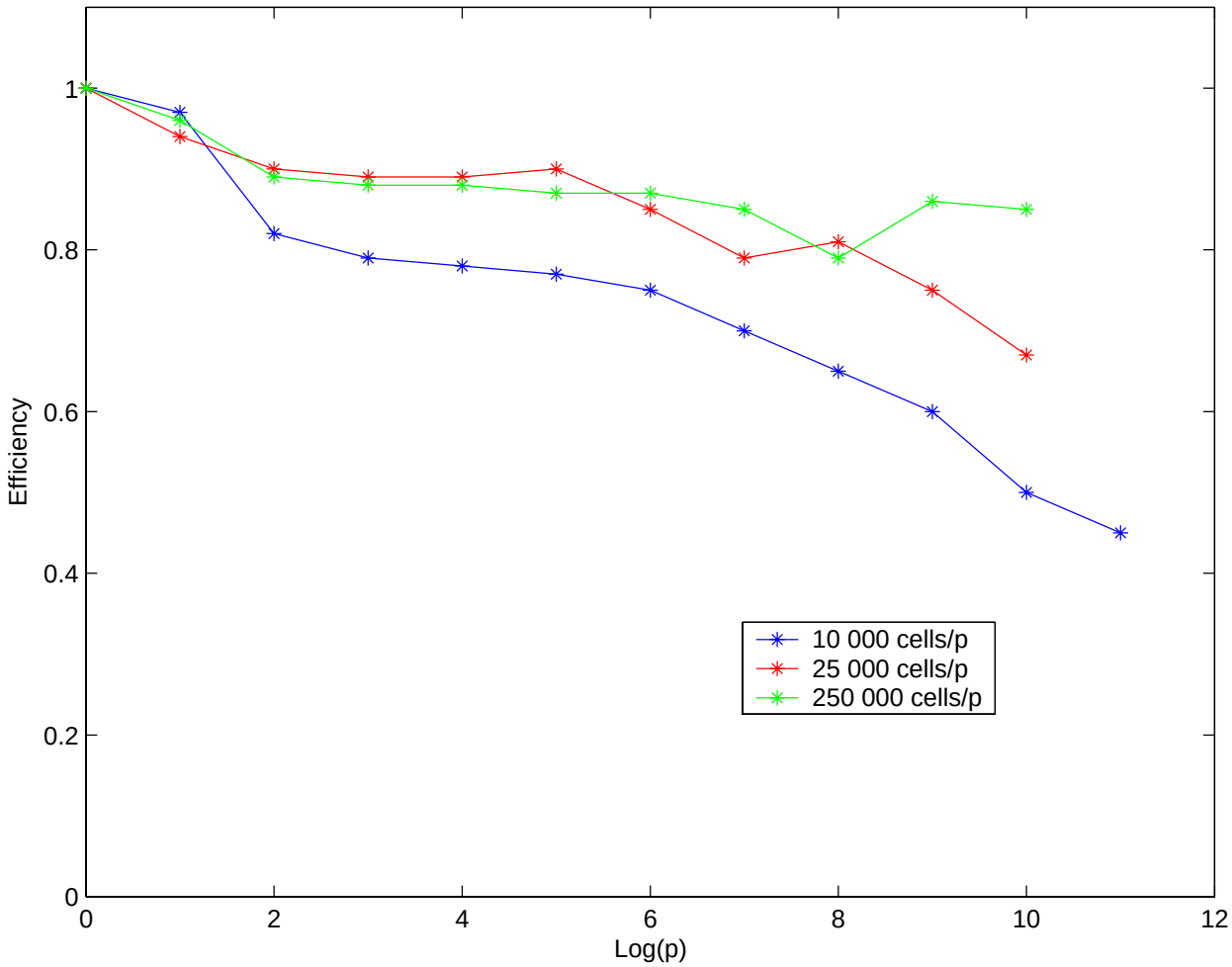


Performance as a function of processors

- Even though this code is carefully tuned and scales well, we only reach about 25% of the peak performance

3D Lagrangian code

- Inviscid compressible fluids
- Lagrangian coordinates with hexahedral or tetrahedral elements, time explicit
- Written in C++ with MPI
- Largest computation: 256 million cells
- Efficiency= ratio of computing time on one processor for m cells/elapsed time on n processors for nm cells
- If the number of cells per processor is large enough, the efficiency is good



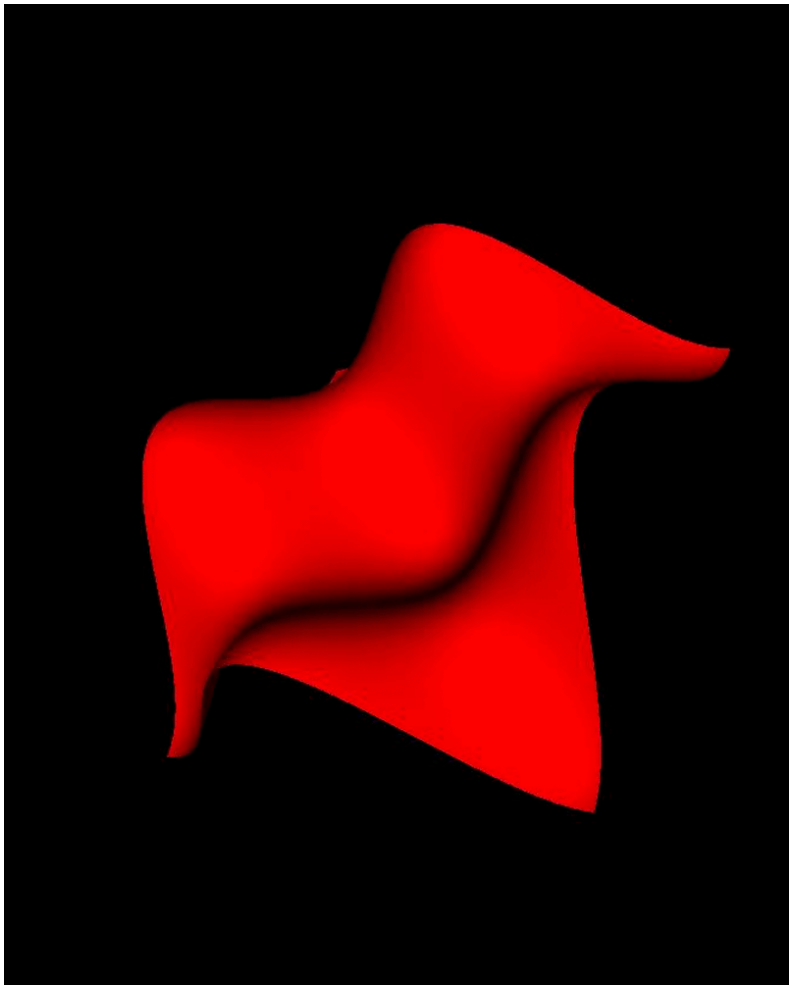
Efficiency as a function of processors

Interface instabilities computations

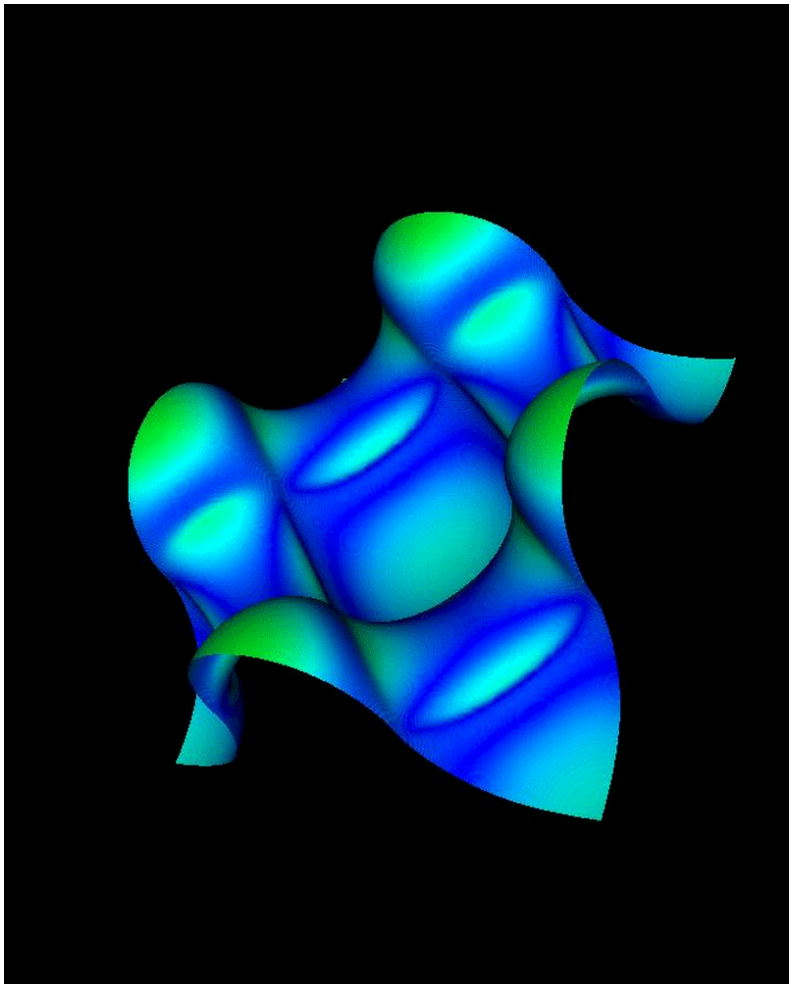
- Direct simulation of growth of perturbations or turbulence
- C++ 3D code for compressible Euler or Navier–Stokes
- Cartesian coordinates

◦ Shock tube: Xe/Kr, interface with monomode perturbation, 37 million cells

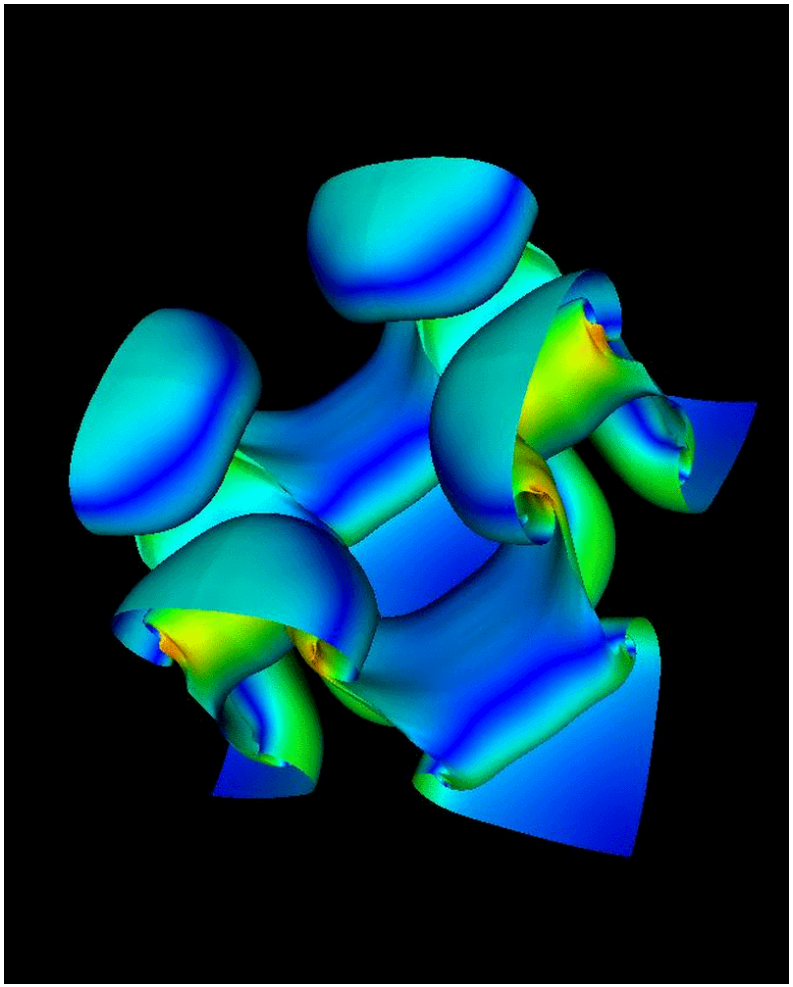
◦ Iso concentration surfaces colored with the module of velocity



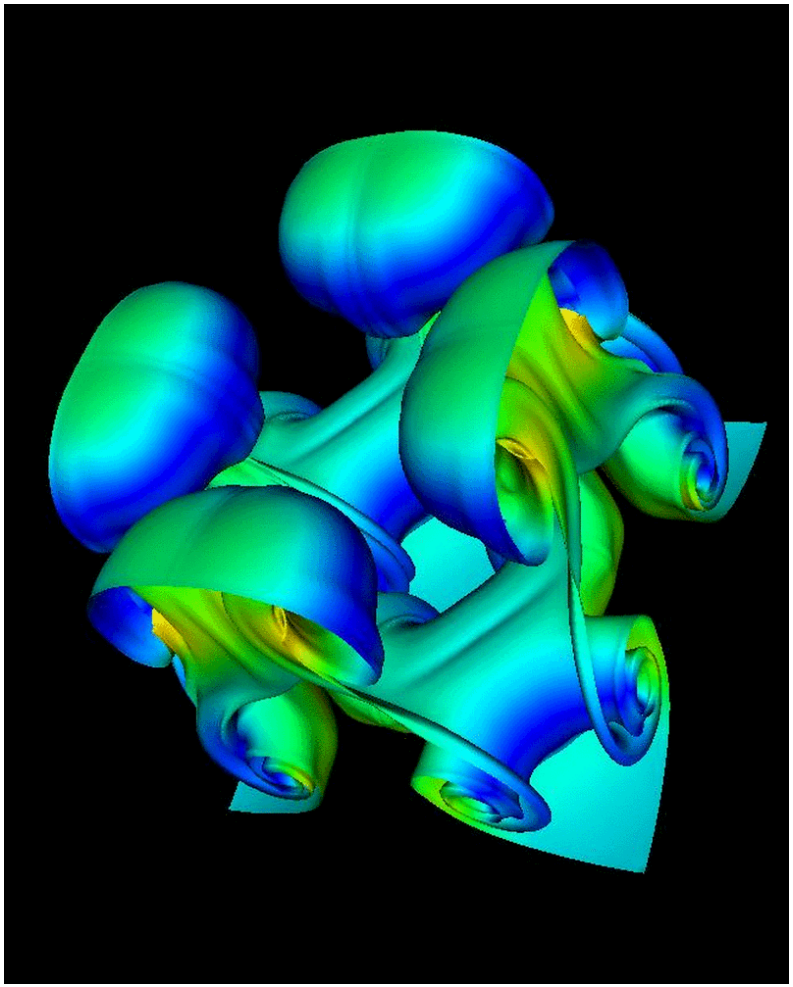
Initial state of the interface



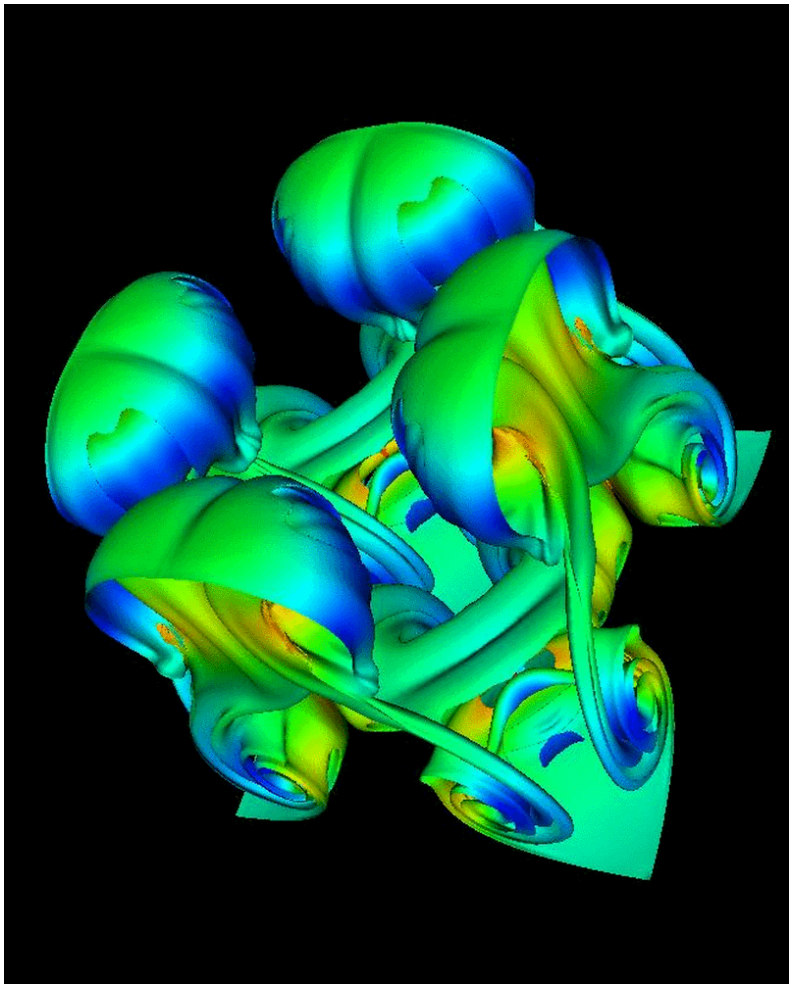
t=100 μs



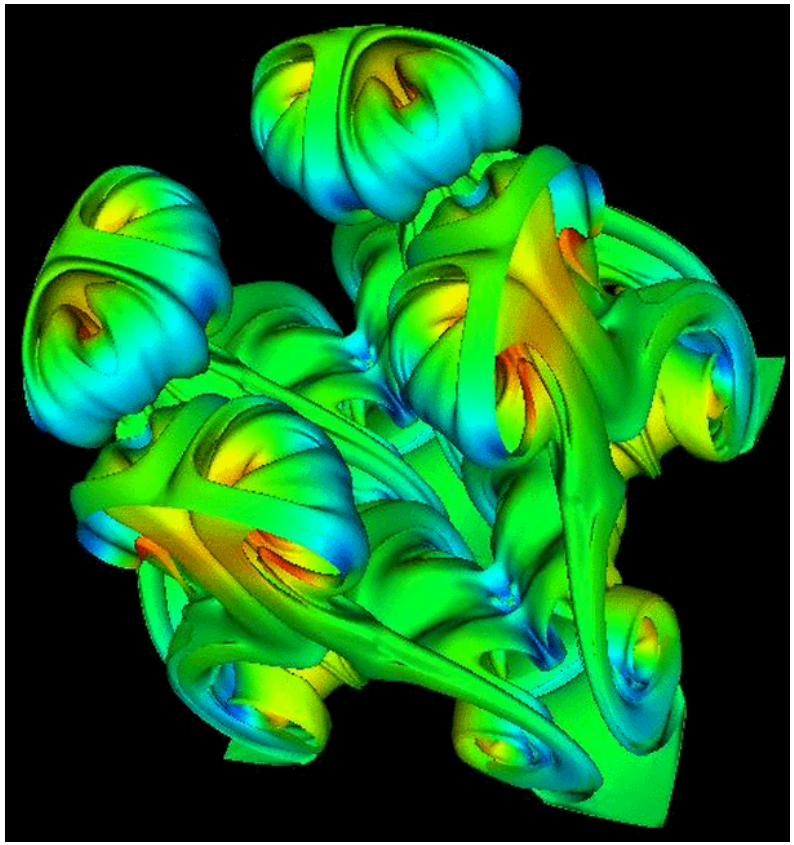
t=300 μs



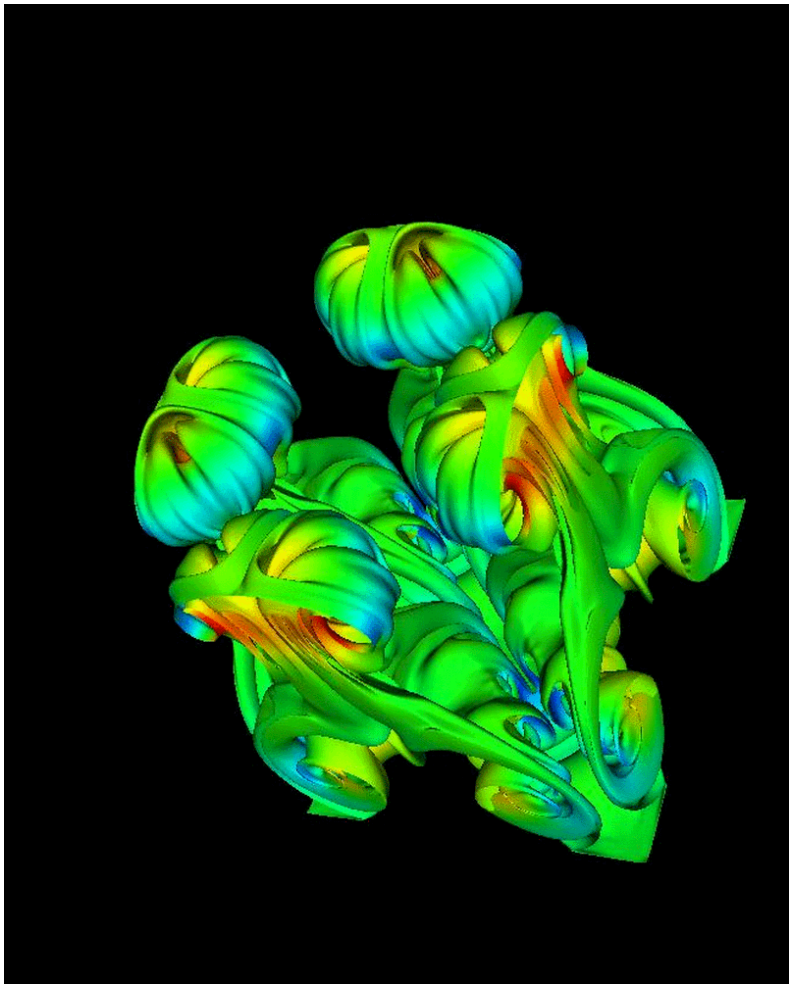
t=500 μs



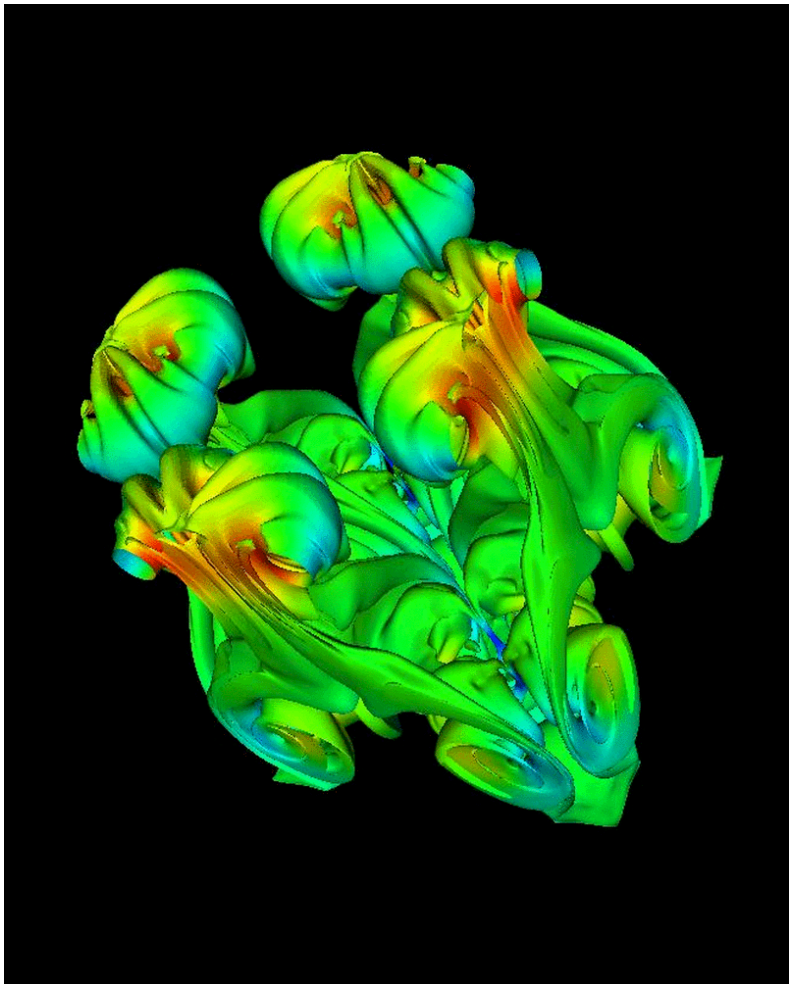
t=600 μs



t=800 μs



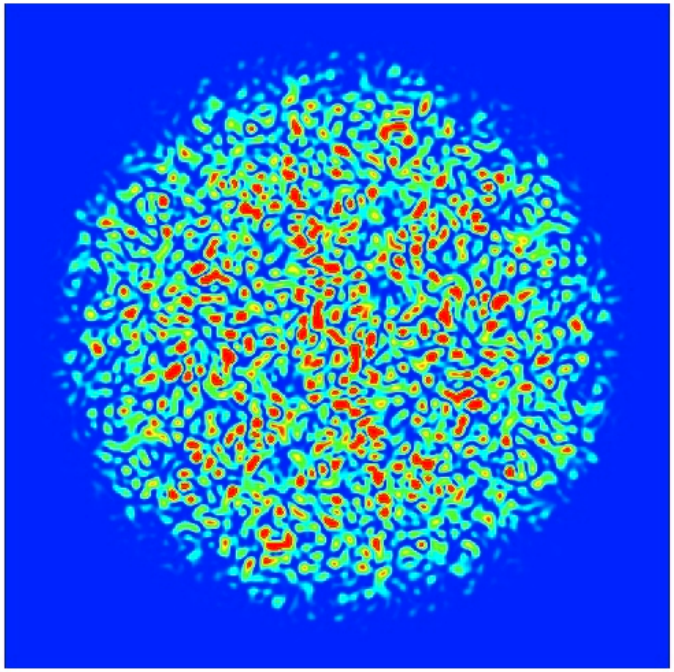
t=900 μs



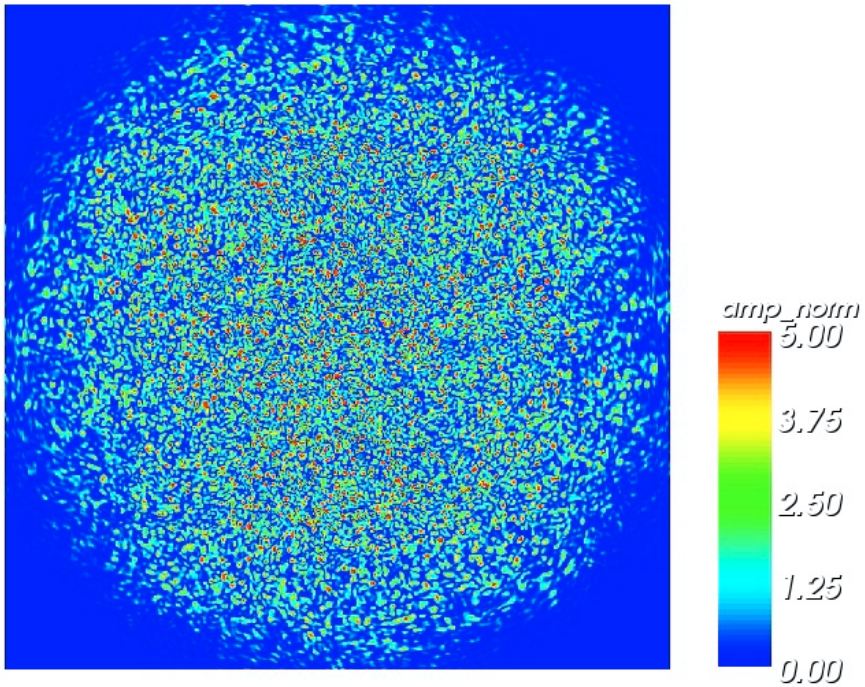
t=1000 μs

3D Laser–Plasma interaction

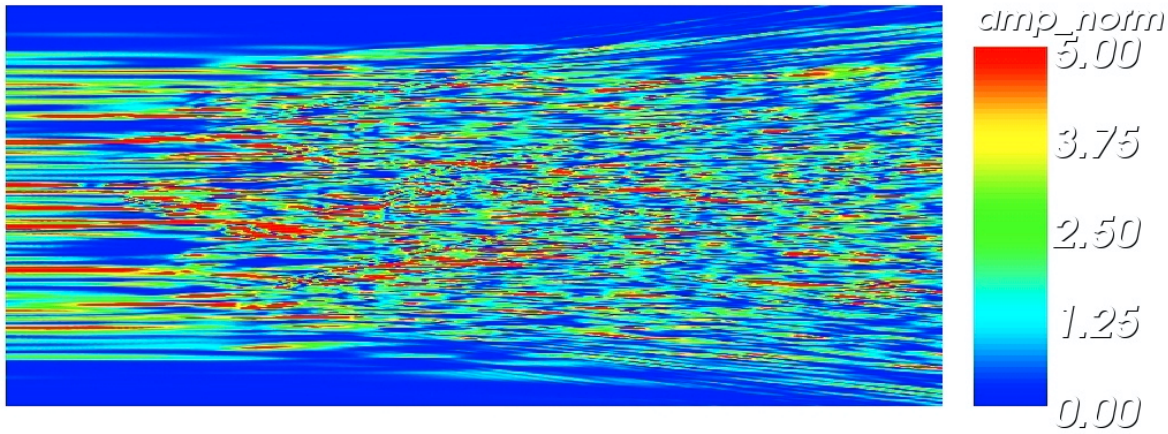
- Laser beam propagating through a plasma
- 3D inviscid compressible fluid + Schrödinger equation
- AMR (Adaptive Mesh Refinement) hydro
- Test case:
  - 505 million cells, 470 time steps
  - 900 processors, elapsed time: 10h 17 mn
  - one 3D picture 900 files = 90 GB



Inward boundary, t=58 ps



Outward boundary, t=58 ps



t=58 ps

- It was very difficult to obtain 3D pictures with 90 GB of data with usual viz packages
- We have to develop new parallel visualization tools using PC clusters
- Parallel read of the 900 files

Conclusions

- We have very fast parallel computers allowing interesting progresses in science and engineering
- However, it is still very difficult to get the most out of these machines
- Often, I/Os are a bottleneck
- Visualizing hundreds of million cells cannot be done with out of the shelf products