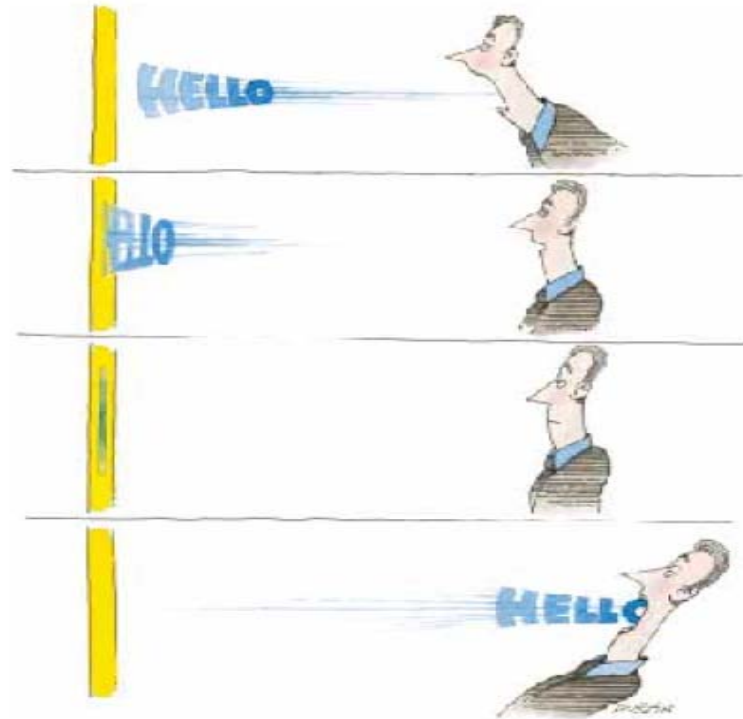

Time Reversal Cartoon.



From Scientific American, November 1999 (M. Fink).

GPUs for Seismic Depth Imaging

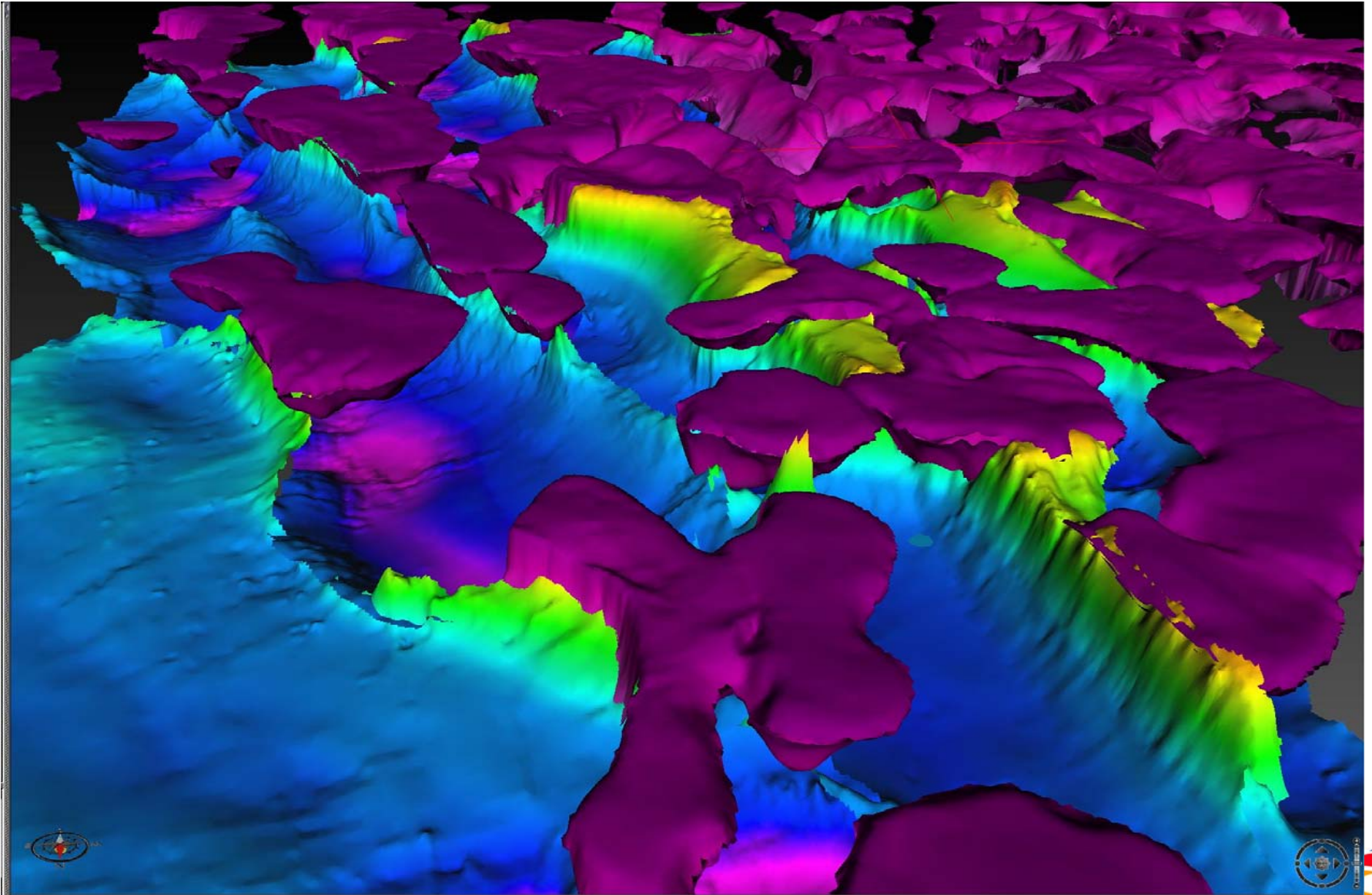
Henri Calandra, Rached Abdelkhalek, TOTAL

Georges-Emmanuel Moulard, CAPS



TOTAL

Seismic exploration challenges



- **Worldwide Context,**
- **Seismic in a nutshell**
- **The Reverse time migration**
- **Computing Power needs and HPC evolution**
- **Accelerating technology for seismic depth imaging:
status and perspectives in TOTAL**

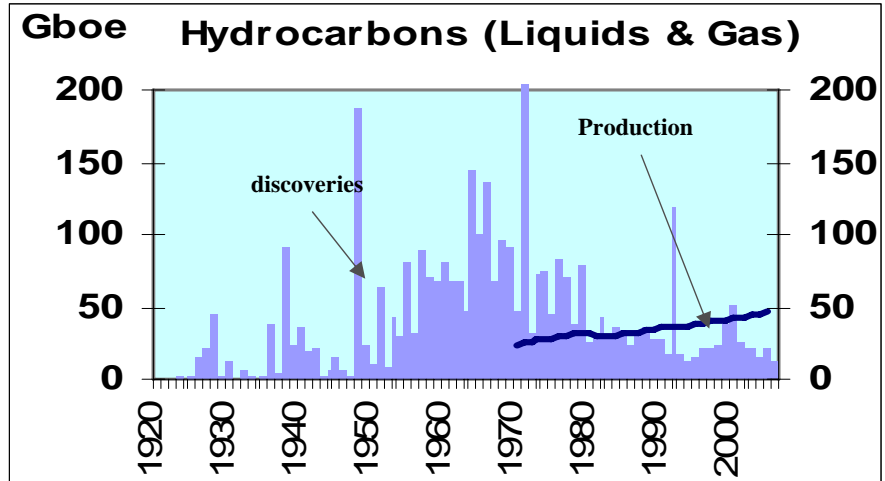
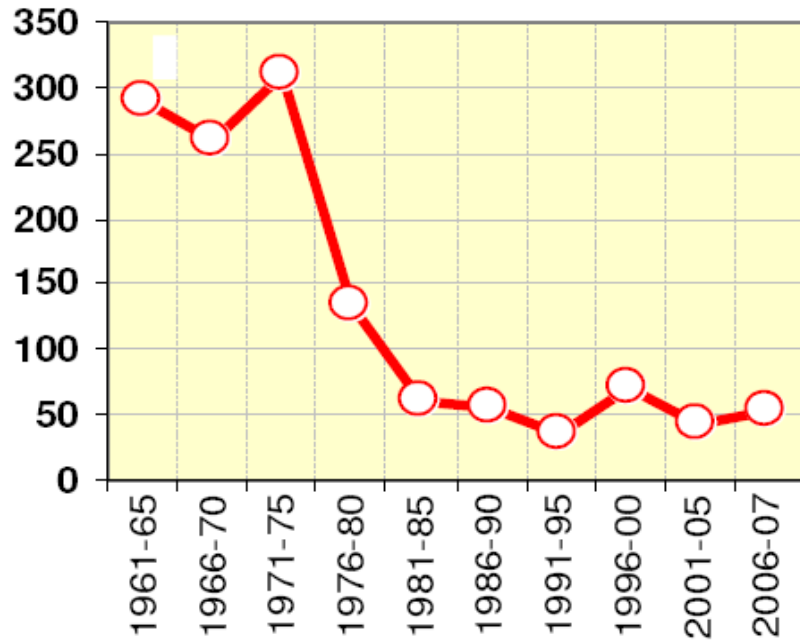
Worldwide context

Worldwide O&G consumption/year approx. 50 Gboe

Average size of new Oil field discovery
~ 50 Mboe since the last 25 years

Oil Discoveries vs. Oil Production

Mboep Average size of Oil discovery

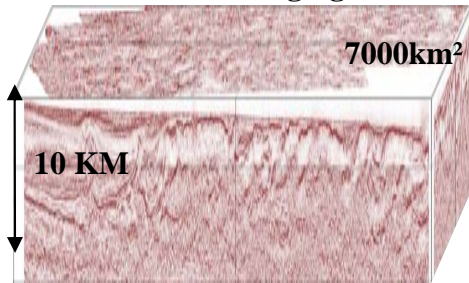


Seismic imaging in a nutshell

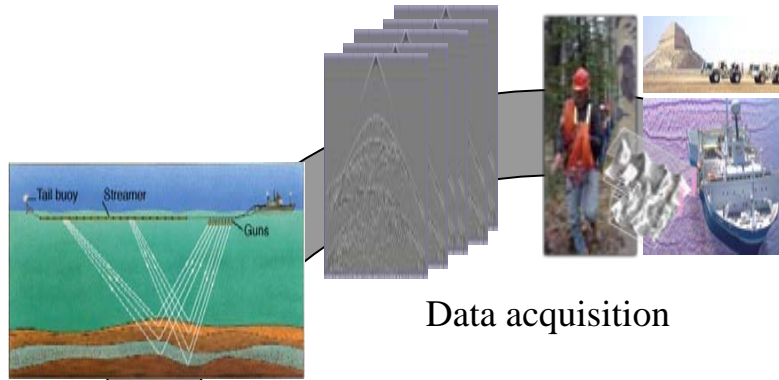
Medical imaging



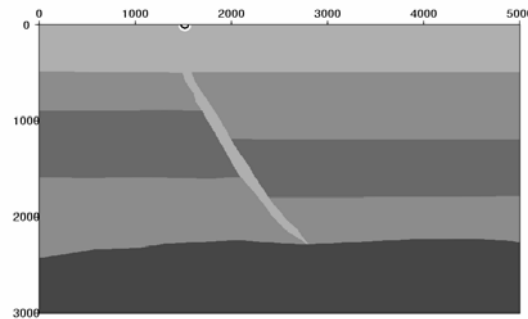
Seismic imaging



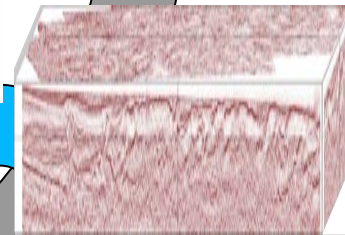
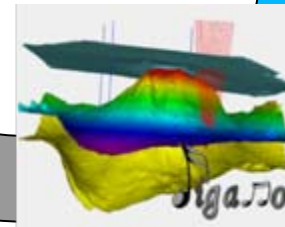
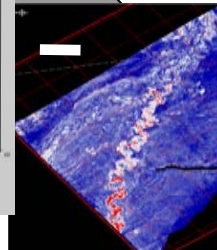
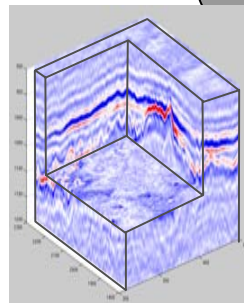
Principle: Ultrasound



Data acquisition

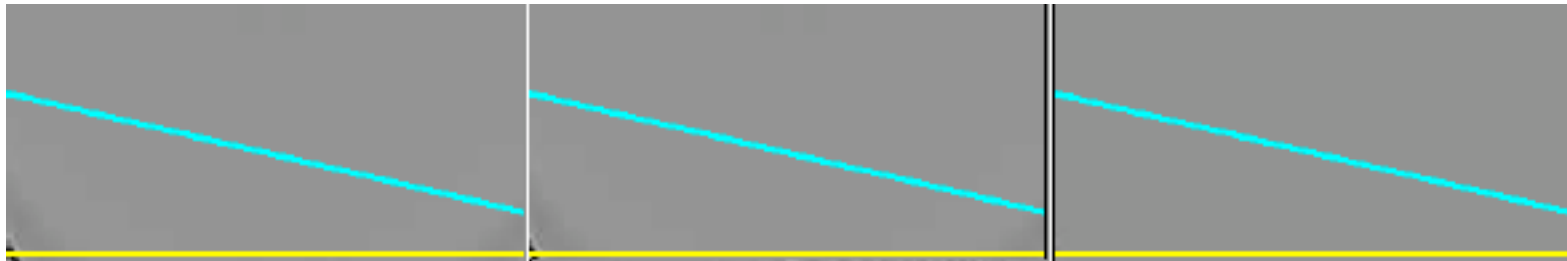
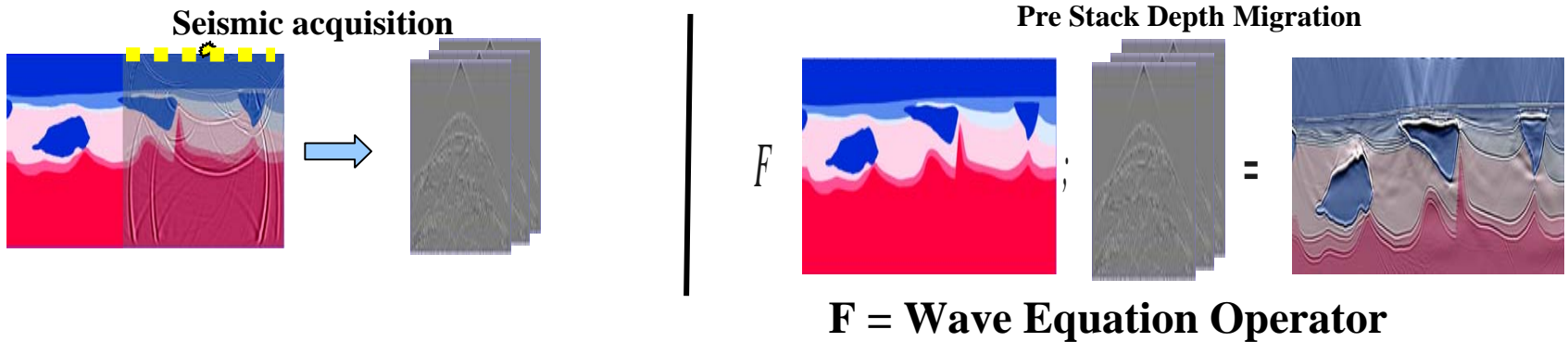


Interpretation



Depth imaging loop

Reverse Time Migration

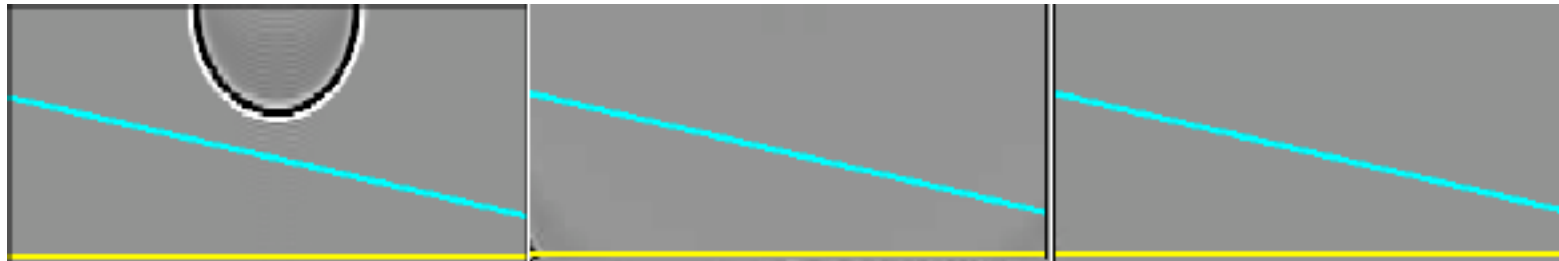
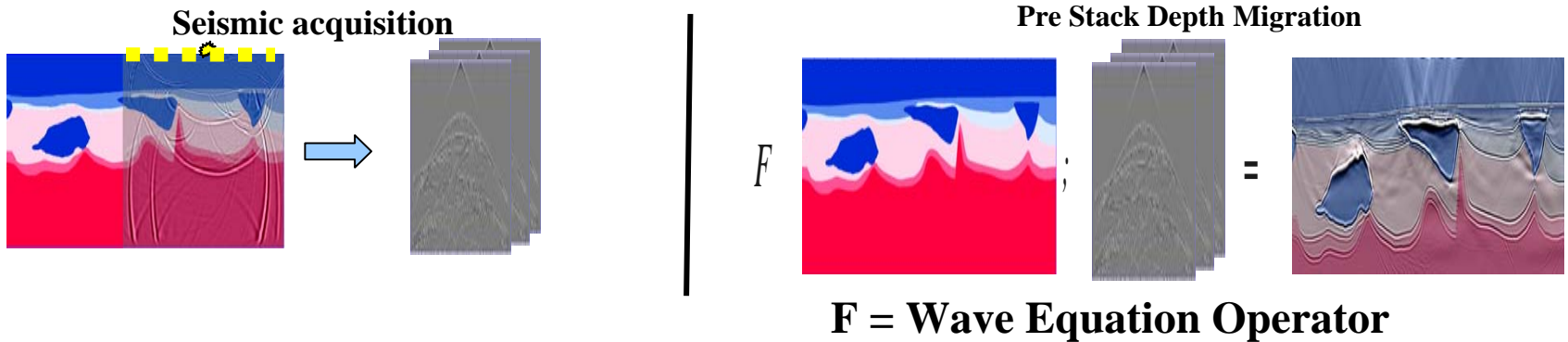


T=0

For each shot profile $\{S, (R_i)_{i=1,n}\}$

- a. **Forward Sweep (forward time)**
 - Propagate source, S , forward in time,
 - Store wavefield .
- b. **Backward Sweep (Reverse Time)**
 - Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
 - Read Source wavefield,
 - Imaging condition.

Reverse Time Migration



$T=0.30$

For each shot profile $\{S, (R_i)_{i=1,n}\}$

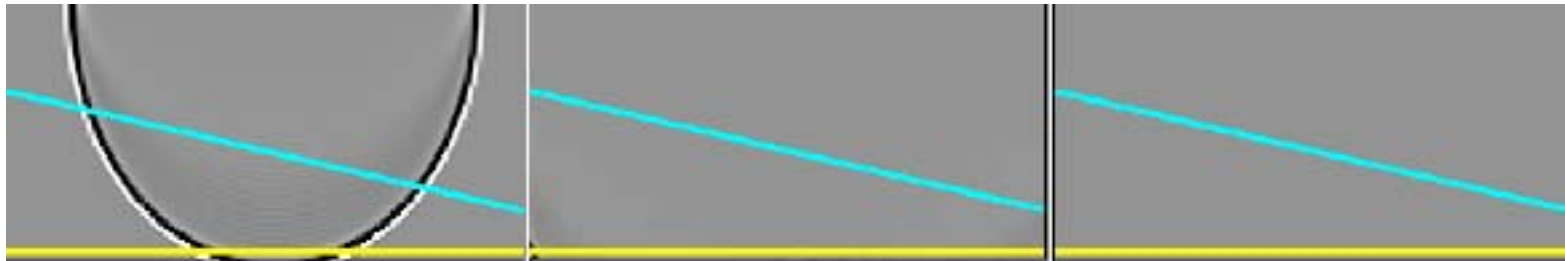
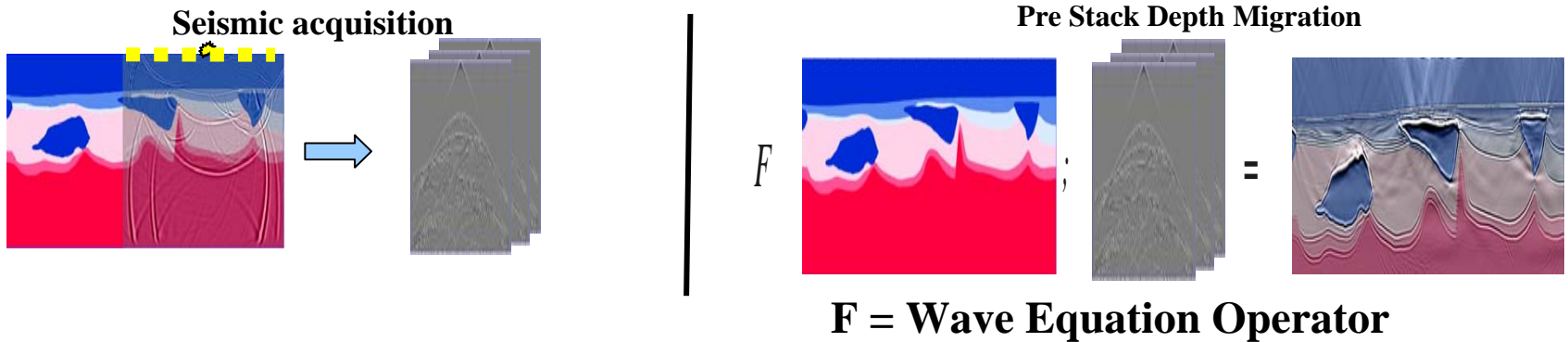
a. Forward Sweep (forward time)

- Propagate source, S , forward in time,
- Store wavefield .

b. Backward Sweep (Reverse Time)

- Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
- Read Source wavefield,
- Imaging condition.

Reverse Time Migration



$T=0.75$

For each shot profile $\{S, (R_i)_{i=1,n}\}$

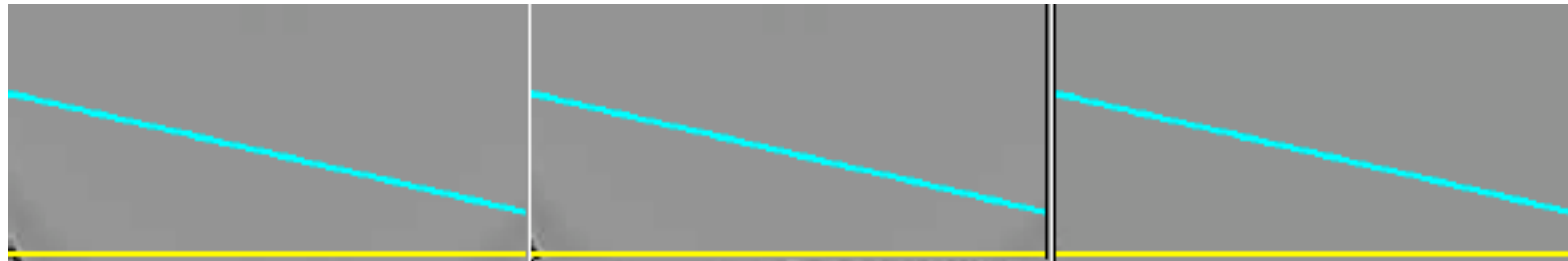
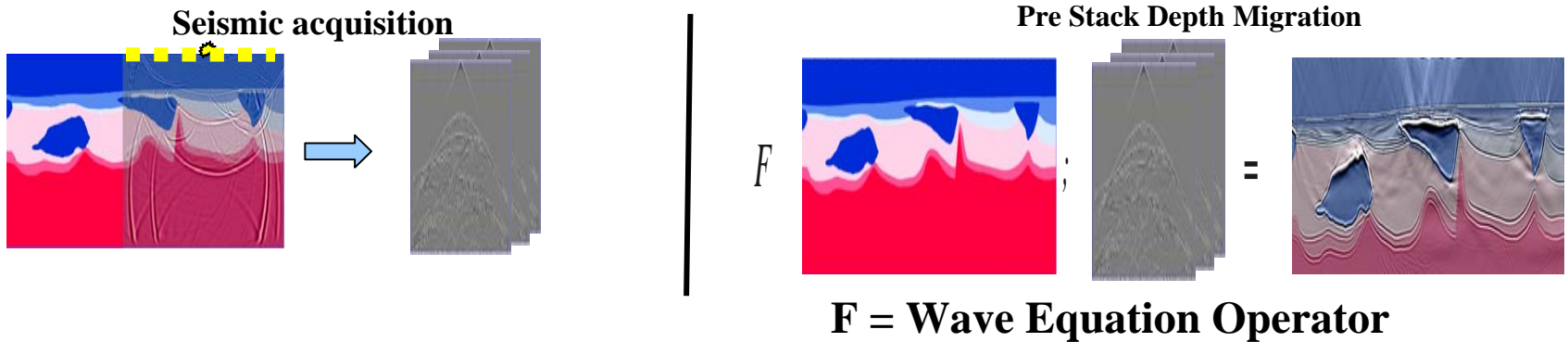
a. Forward Sweep (forward time)

- Propagate source, S , forward in time,
- Store wavefield .

b. Backward Sweep (Reverse Time)

- Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
- Read Source wavefield,
- Imaging condition.

Reverse Time Migration

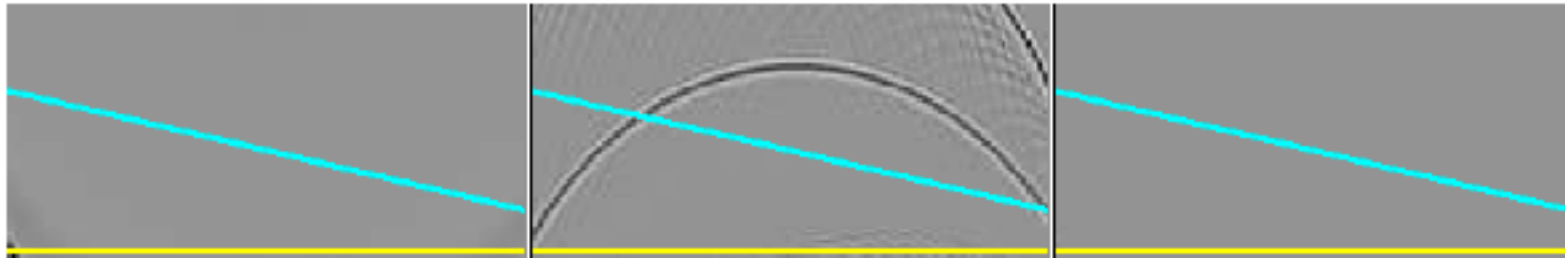
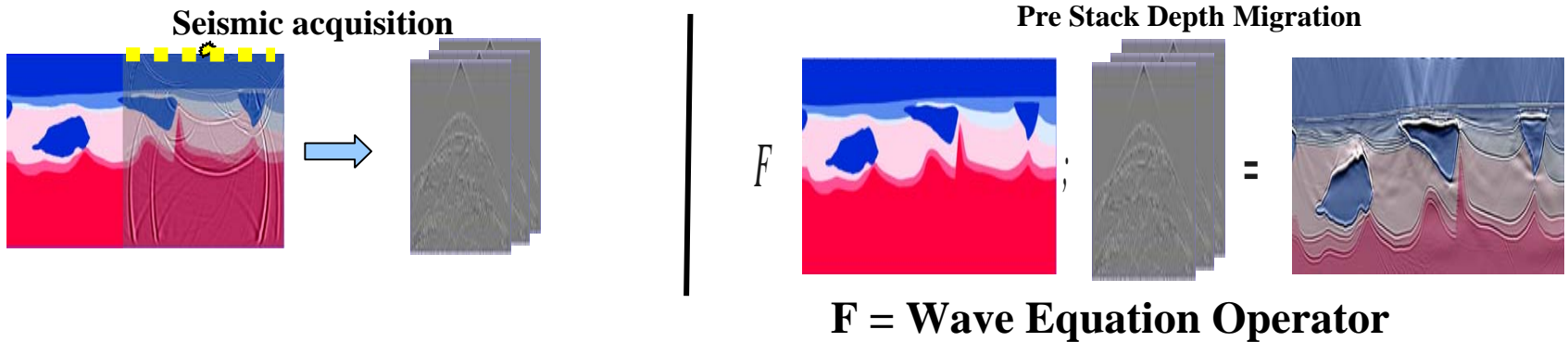


$T=1.20$

For each shot profile $\{S, (R_i)_{i=1,n}\}$

- a. **Forward Sweep (forward time)**
 - Propagate source, S , forward in time,
 - Store wavefield .
- b. **Backward Sweep (Reverse Time)**
 - Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
 - Read Source wavefield,
 - Imaging condition.

Reverse Time Migration

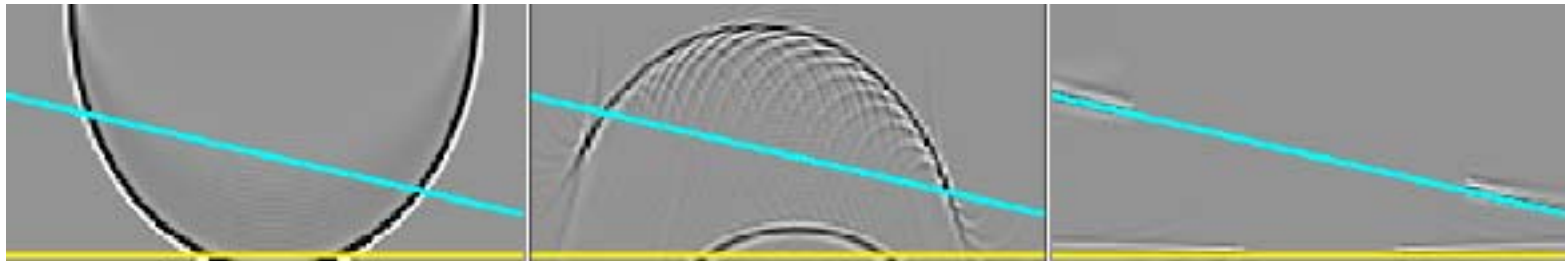
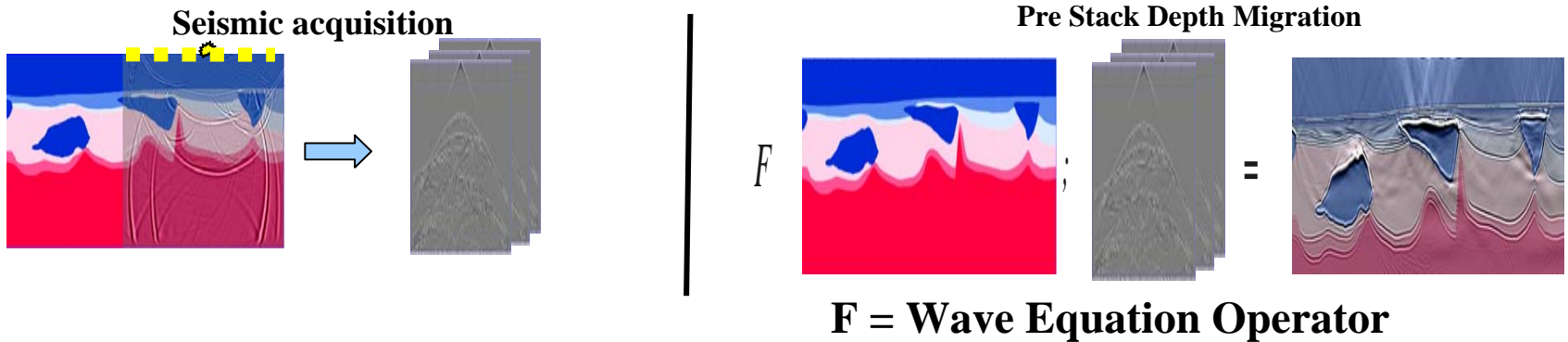


$T=1.20$

For each shot profile $\{S, (R_i)_{i=1,n}\}$

- a. **Forward Sweep (forward time)**
 - Propagate source, S , forward in time,
 - Store wavefield .
- b. **Backward Sweep (Reverse Time)**
 - Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
 - Read Source wavefield,
 - Imaging condition.

Reverse Time Migration

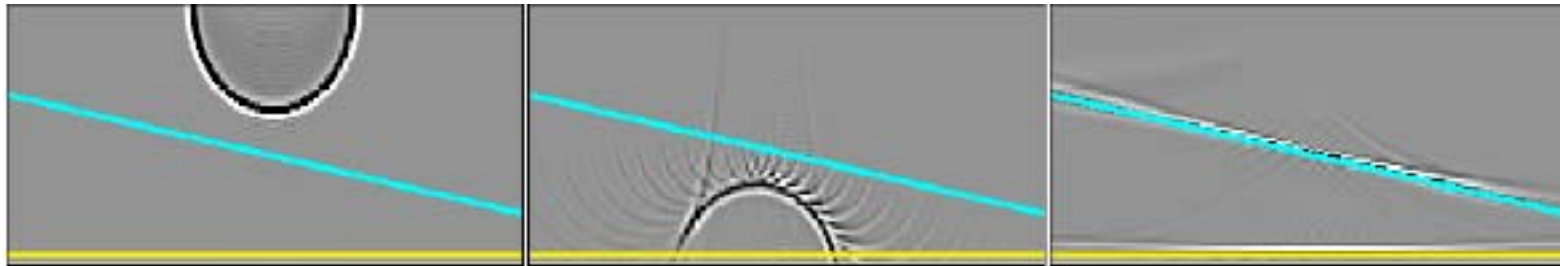
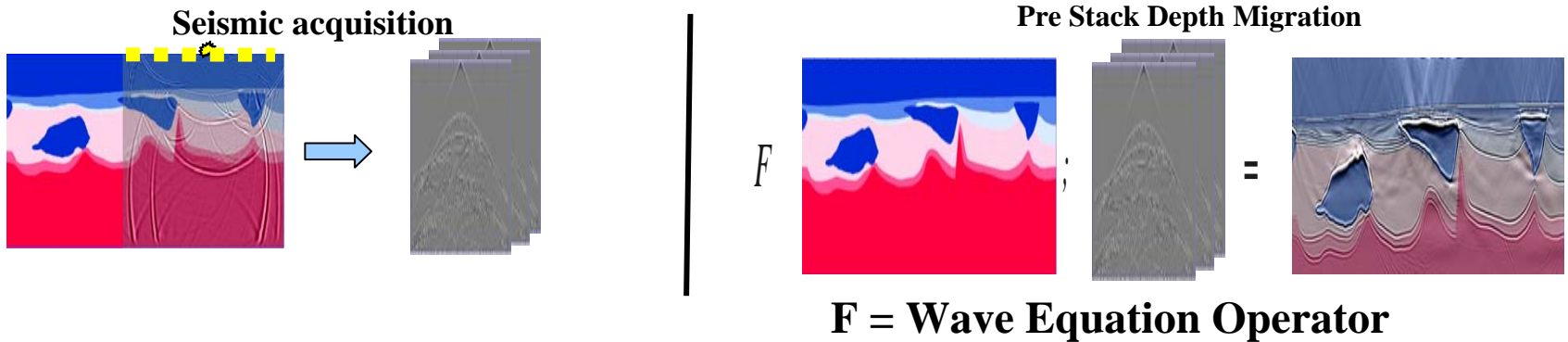


$T=0.75$

For each shot profile $\{S, (R_i)_{i=1,n}\}$

- a. **Forward Sweep (forward time)**
 - Propagate source, S , forward in time,
 - Store wavefield .
- b. **Backward Sweep (Reverse Time)**
 - Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
 - Read Source wavefield,
 - Imaging condition.

Reverse Time Migration

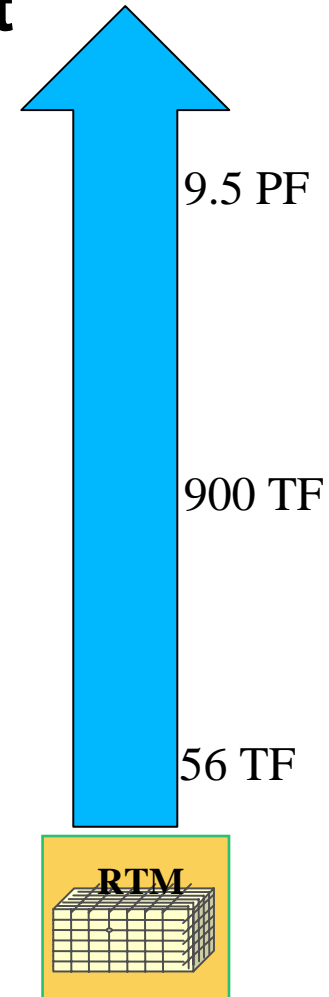
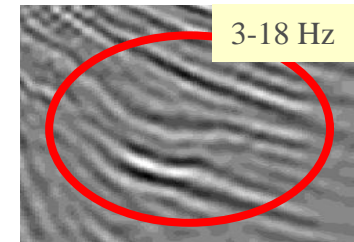
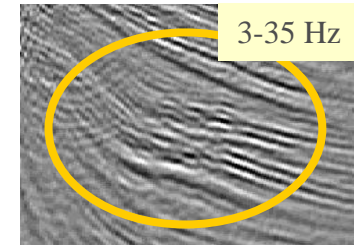
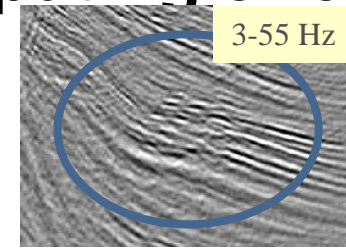
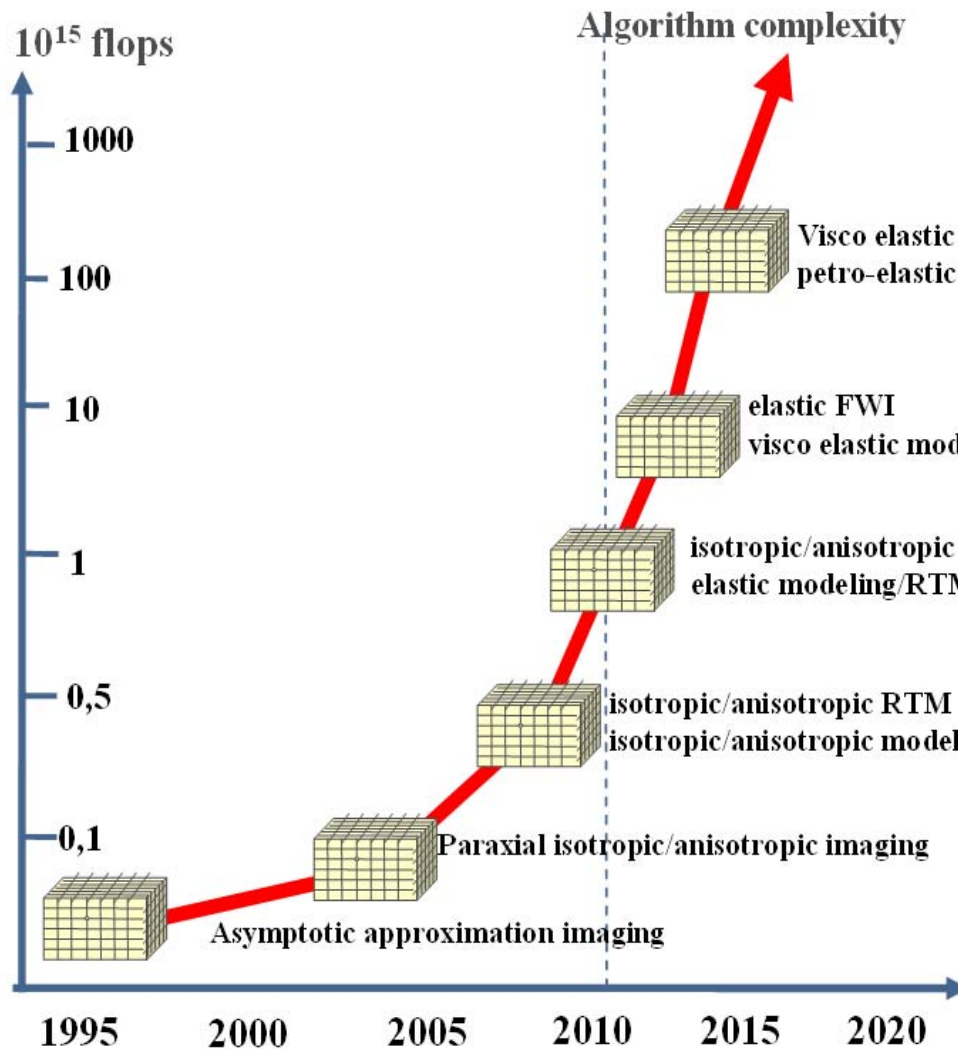


$T=0.30$

For each shot profile $\{S, (R_i)_{i=1,n}\}$

- a. **Forward Sweep (forward time)**
 - Propagate source, S , forward in time,
 - Store wavefield .
- b. **Backward Sweep (Reverse Time)**
 - Propagate receivers , $(R_i)_{i=1,n}$, backward in time,
 - Read Source wavefield,
 - Imaging condition.

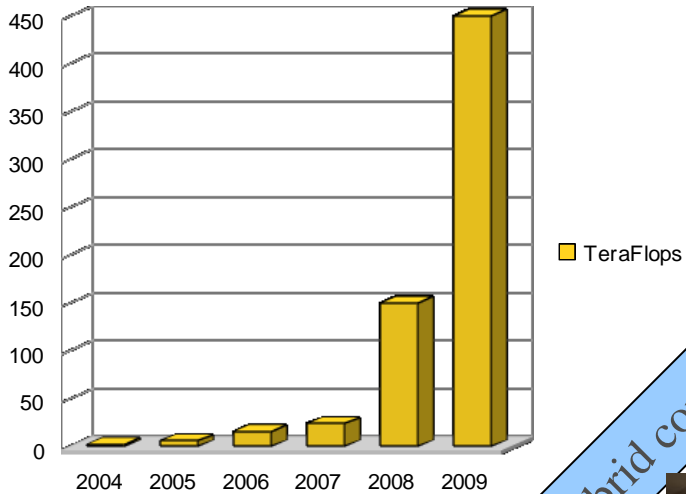
Codes complexity and computing effort



Computing effort function of the seismic frequency content

Algorithmic complexity and corresponding computing power

HPC Evolution in TOTAL



From SMP cluster to MPP and Hybrid computing



SGI ICE+ MPP (16384 Intel X86-64 + 256 GPUS)



SGI ICE+ MPP (10240 Intel X86-64)



IBM SMP cluster (2048 P5)



SGI Altix SMP cluster (1024 intel ia64)

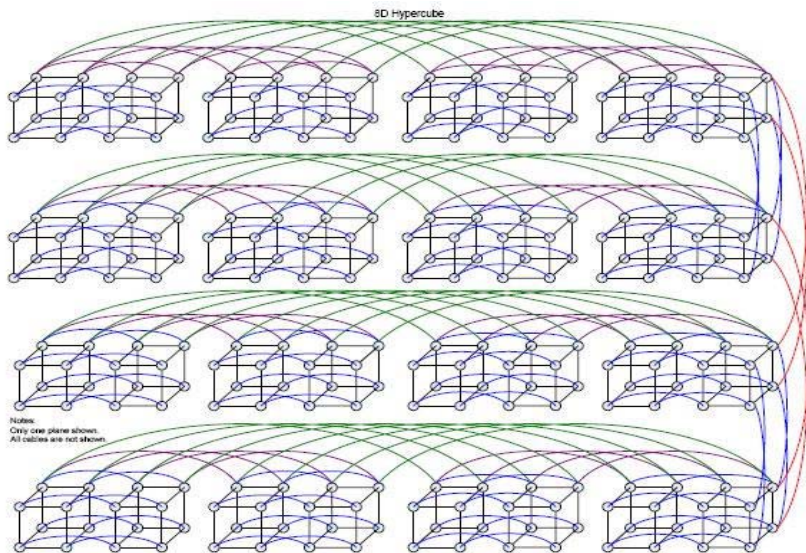
Why MPP ?

❑ Seismic depth is massively Parallel:

- Embarrassingly over the shots: ex 100000 shots can be solved in one iteration on a 100000 CPUs machine
- Explicit data distribution and data parallel model programming when processing one shot

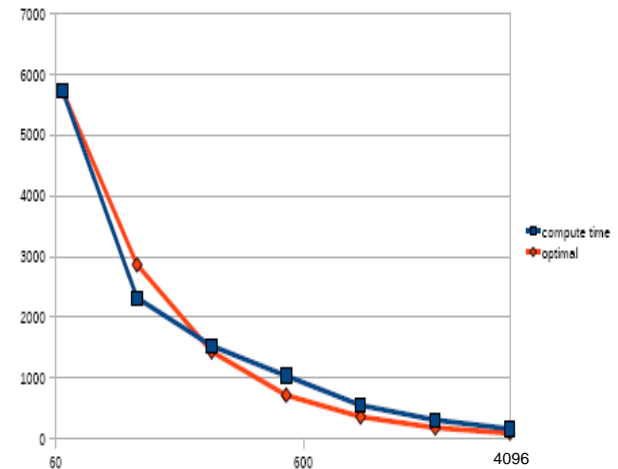
❑ Reliability and performances on very large configurations

❑ Impact on footprint and power dissipation



SGI ICE+ Hypercube topology interconnect:

- Scalable single system
- Easy to manage



Taking advantage of Scalable interconnect
Leads to very efficient algorithm implementation
(3D Wave Equation Finite difference solver)



GPU/many core technology for seismic depth imaging ?

► **Trends: multiply the number of cores**

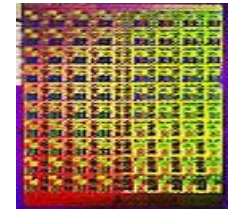
► **Our codes can take advantage of many core technology**

► **GPUs can be seen as large multicore processors and can offer tremendous speed up.**

► **GPUS first step to the massive multicore technology ?**



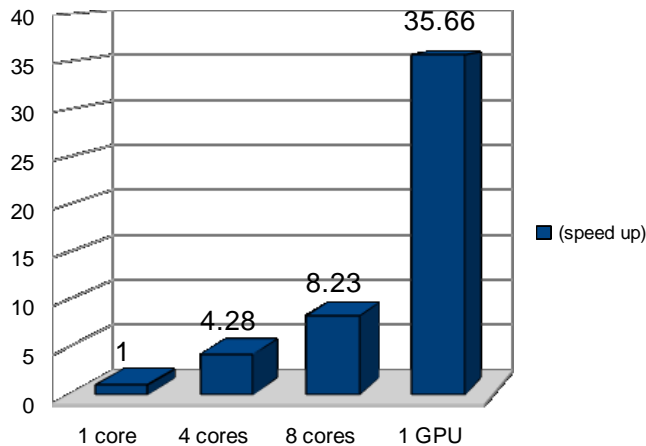
4 cores Intel processor



80 cores Intel processor (prototype)



Nvidia GPU 240 cores ~ 1TF single precision



3D Wave Equation solver speed up comparison

► **Having access to different technologies through the same interconnect can offer new perspectives in terms of model programming and application development**

► **To reach the performances needed for solving our problems, we have no choice but combining high scalable interconnect and massive multi-core technology.**

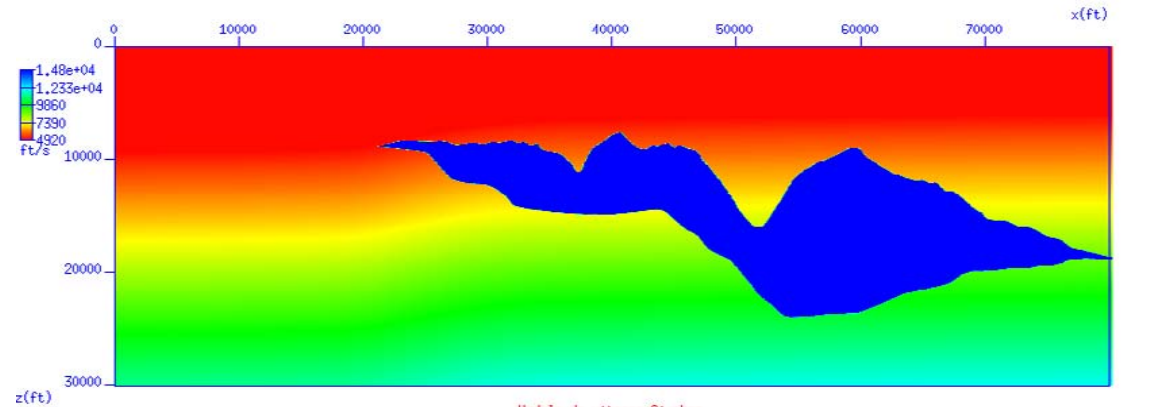
GPU/many core technology for seismic depth imaging ?

Starting in 2007 with the following questions

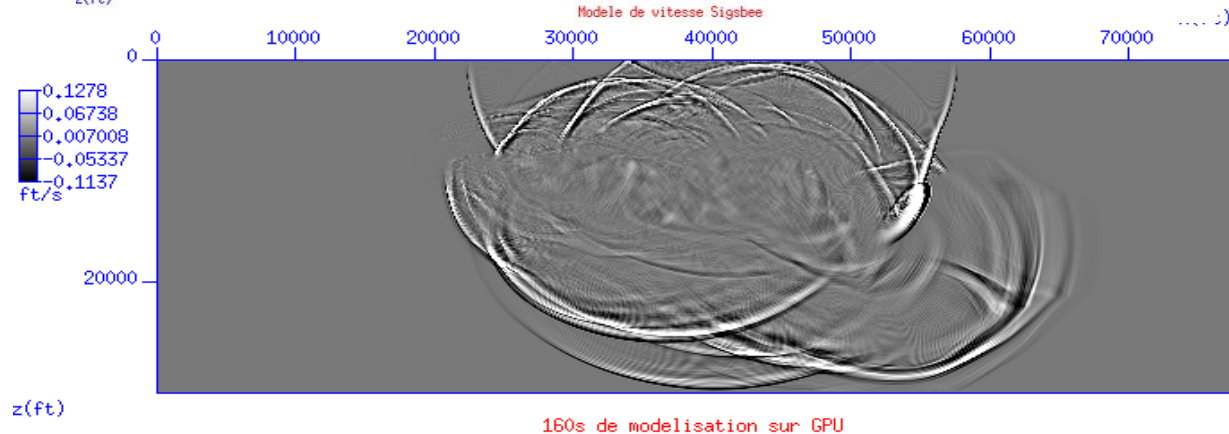
- **Is it possible to take advantage of GPU for seismic depth imaging**
- **What porting effort: coding, optimization constrains ?**
- **What programming tools ?**
- **What about building industrial applications (Modeling, RTM...) on a GPU cluster based system ?**

GPU/many core technology for seismic depth imaging ?

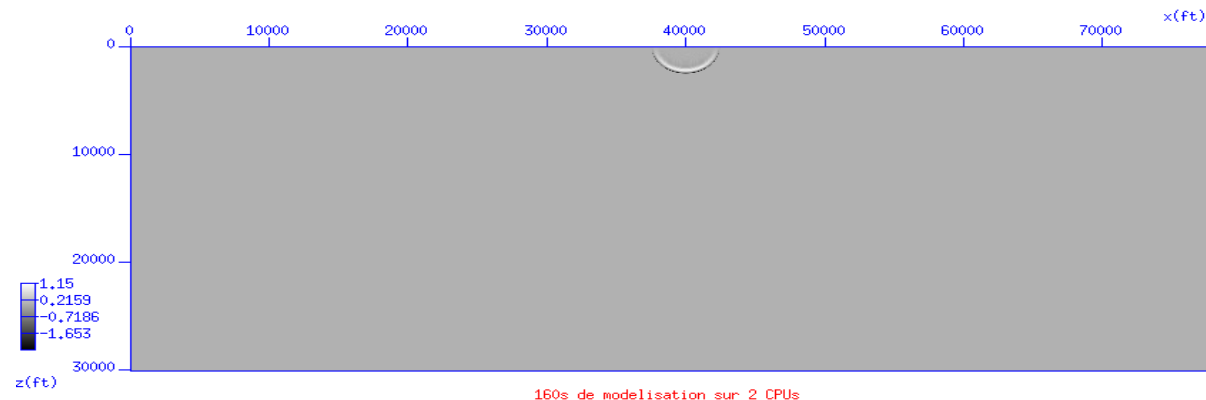
- Velocity model:
6400*1500 grid size



- GPU: 3.6s (12808 time steps)



- 2 CPUs: 0.5s
(1778 time steps).



Seismic application taking advantage of GPUS

❑ 2 main applications implemented on GPU cluster:

Seismic modeling

Reverse time Migration

❑ Seismic modeling

• motivation:

- **Accelerate survey acquisition modeling**
- **Wave Equation solver is the basic kernel for seismic depth imaging algorithm**
- **Explore different implementations on GPUS: (single GPU, multi GPUs)**
- **Explore model programming and programming language**

❑ RTM

•Motivation

- **Most accurate depth imaging method: the most expensive**
- **Take advantage of progress made on Seismic modeling (same kernel)**
- **Explore communication load balancing between host and device**



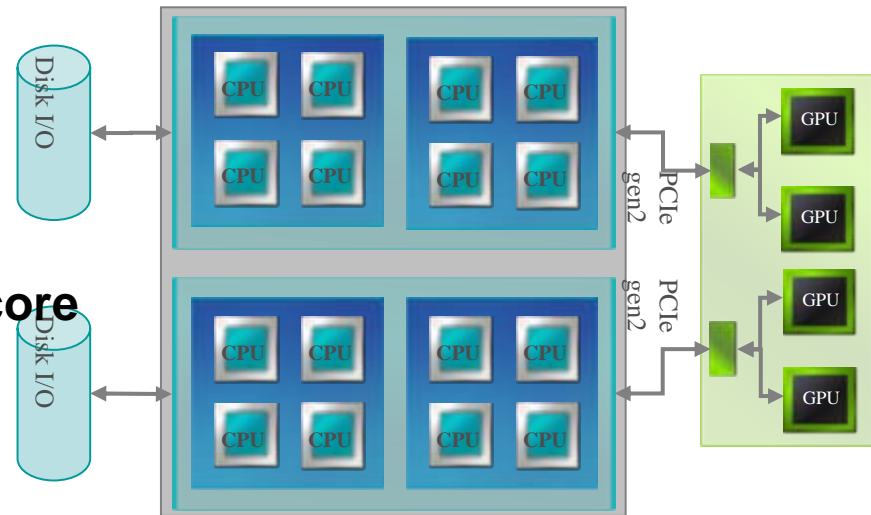
Implementation on GPU

Objectives:

- Take advantage of the data parallelism capabilities of GPU to accelerate the Finite difference kernels
- Optimize the communication between host and GPU to reduce the bottleneck of the PCI express interconnect
- Explore different configurations host-GPU

Machine configuration:

- Tesla S1070
 - 4 GB memory per GPU
 - Dual PCIe gen2 (2x6.4GB/s)
- Host server
 - Twin Xeon-based bi-socket quadcore
- 2x16GB memory
- 2.5GHz



Porting effort: Optimization

◆ Kernel Optimization

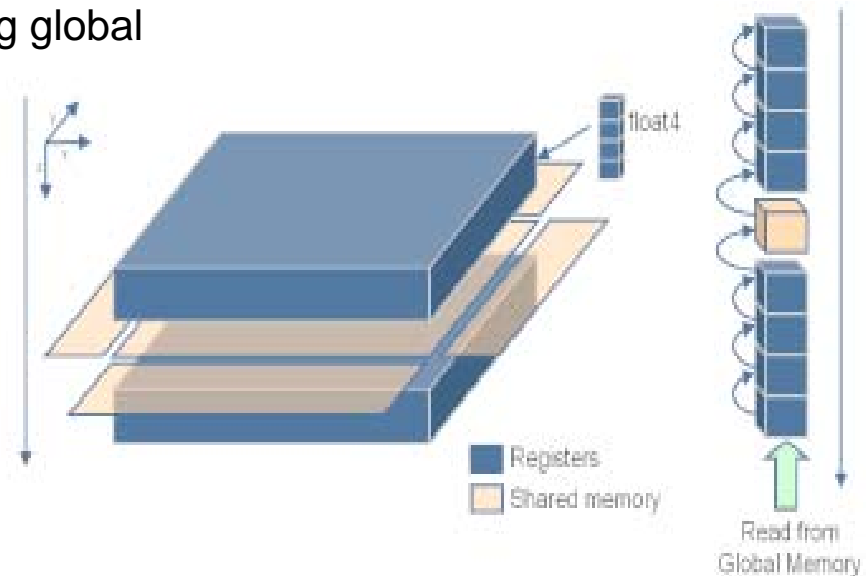
- Sliding 2D tile algorithm(P Micikevicius 2009)
- Shared memory and registers minimizing global memory access
- Loop over the z direction

◆ Global memory access pattern

- padding
- 16x16 threads blocks

◆ Host-GPU transfers

- partial transfers
- page locked host memory
- domain decomposition along z axis



Number of read accesses per data point

RTM 2D	4
RTM 3D with texture	29
RTM 3D with 3D shared memory accesses	7,5
RTM 3D with sliding 2D shared memory	4

Porting Effort: CPU-GPU Data transfer optimization

Take advantage of asynchronous communications to overlap data transfer

First implementation: **75%** of elapsed time spent on PCIe communications

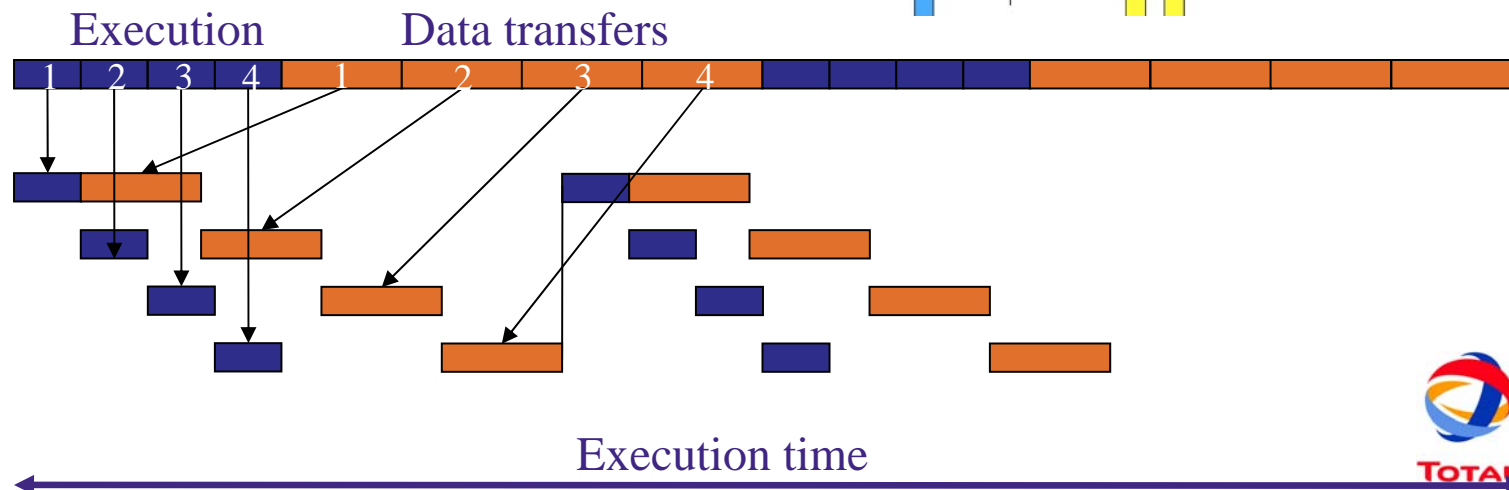
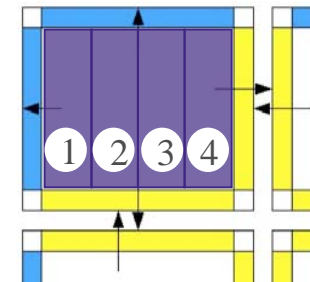
Take advantage of :

PCIe Gen2 bandwidth

DMA accessible “pinned memory”

Asynchronous communications

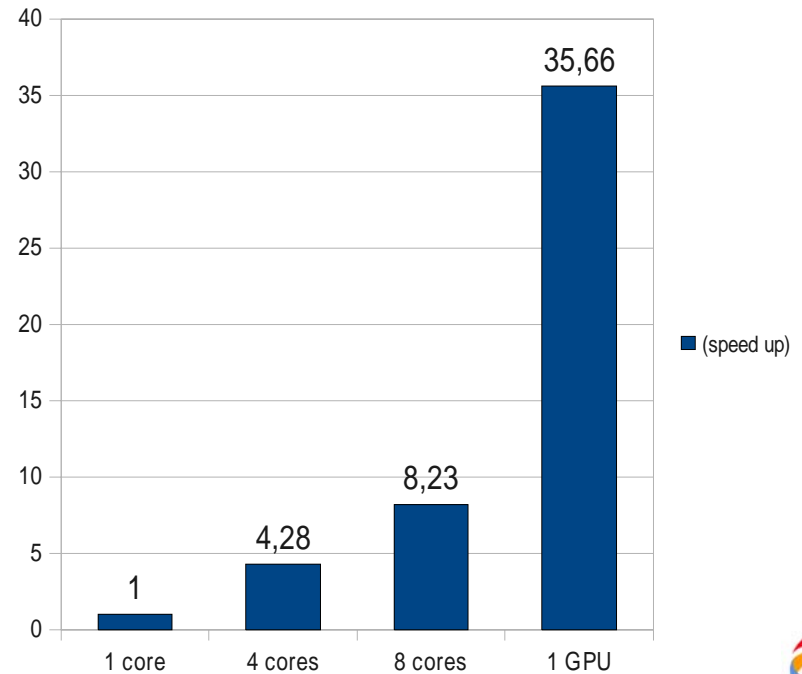
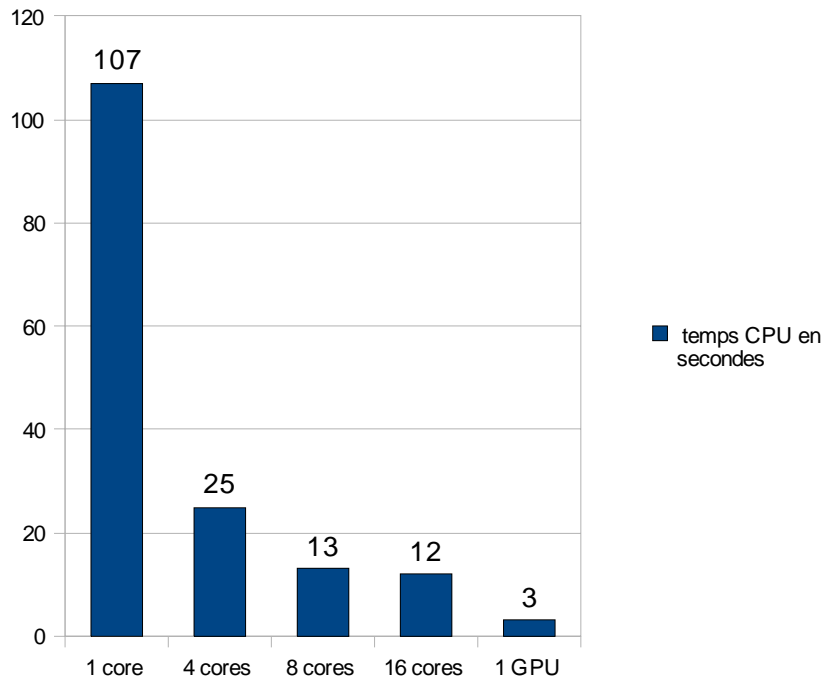
%	synch	asynch
FWD	32	15
BWD	41	25



Seismic modeling

Variable density modelling

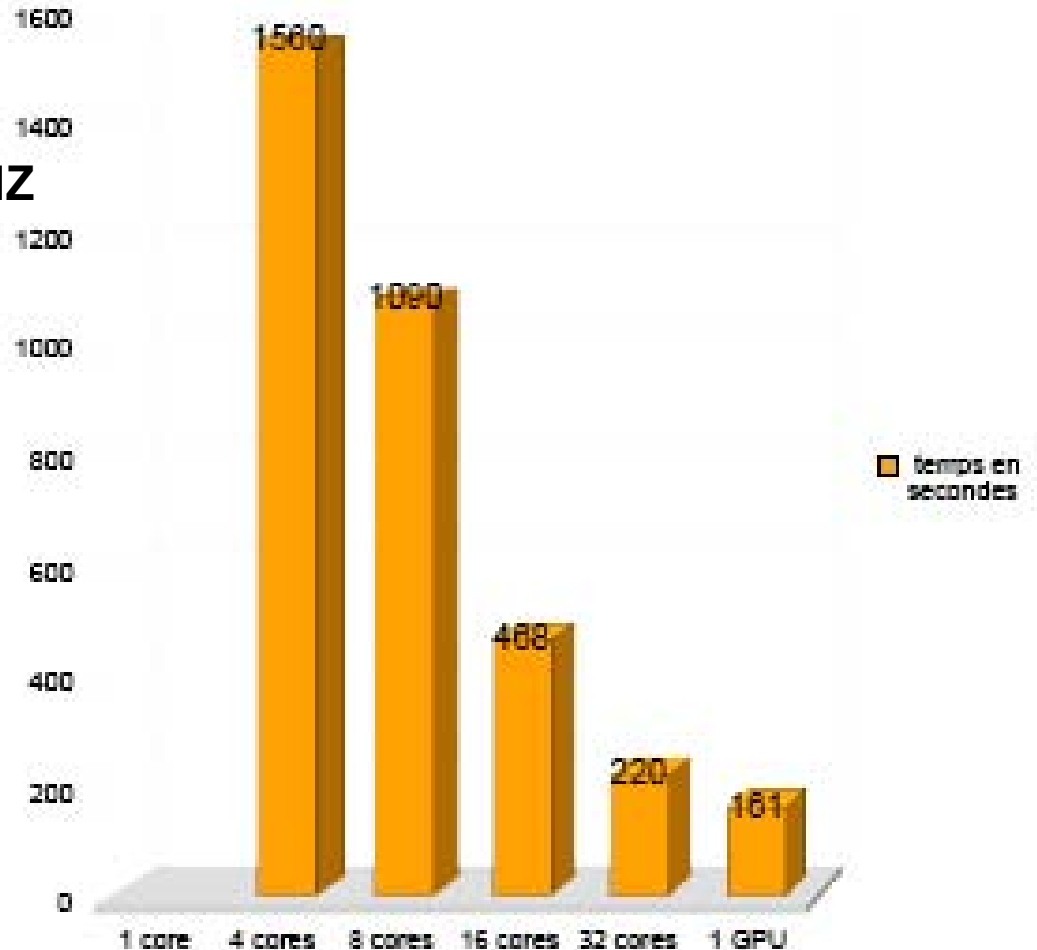
- Grid size: 544x529x465
- Hapertown 3.0GHZ



Seismic modeling

□ Example

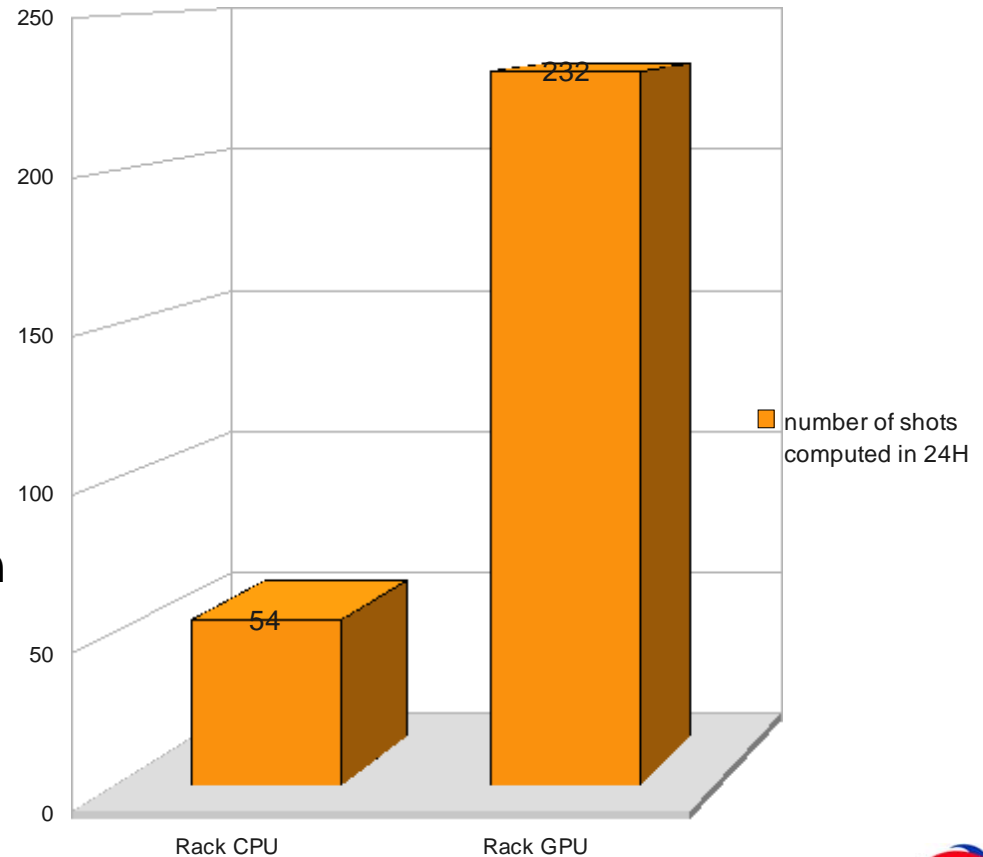
- Model size 512x512x512
- 490 time step
- SGI ICE+, Harpertown 3GHZ
- Test :
 - 1 GPU
 - 4 cores (4 MPI)
 - 8 cores (8 MPI)
 - 16 cores (16 MPI)
 - 32 cores (32 MPI)



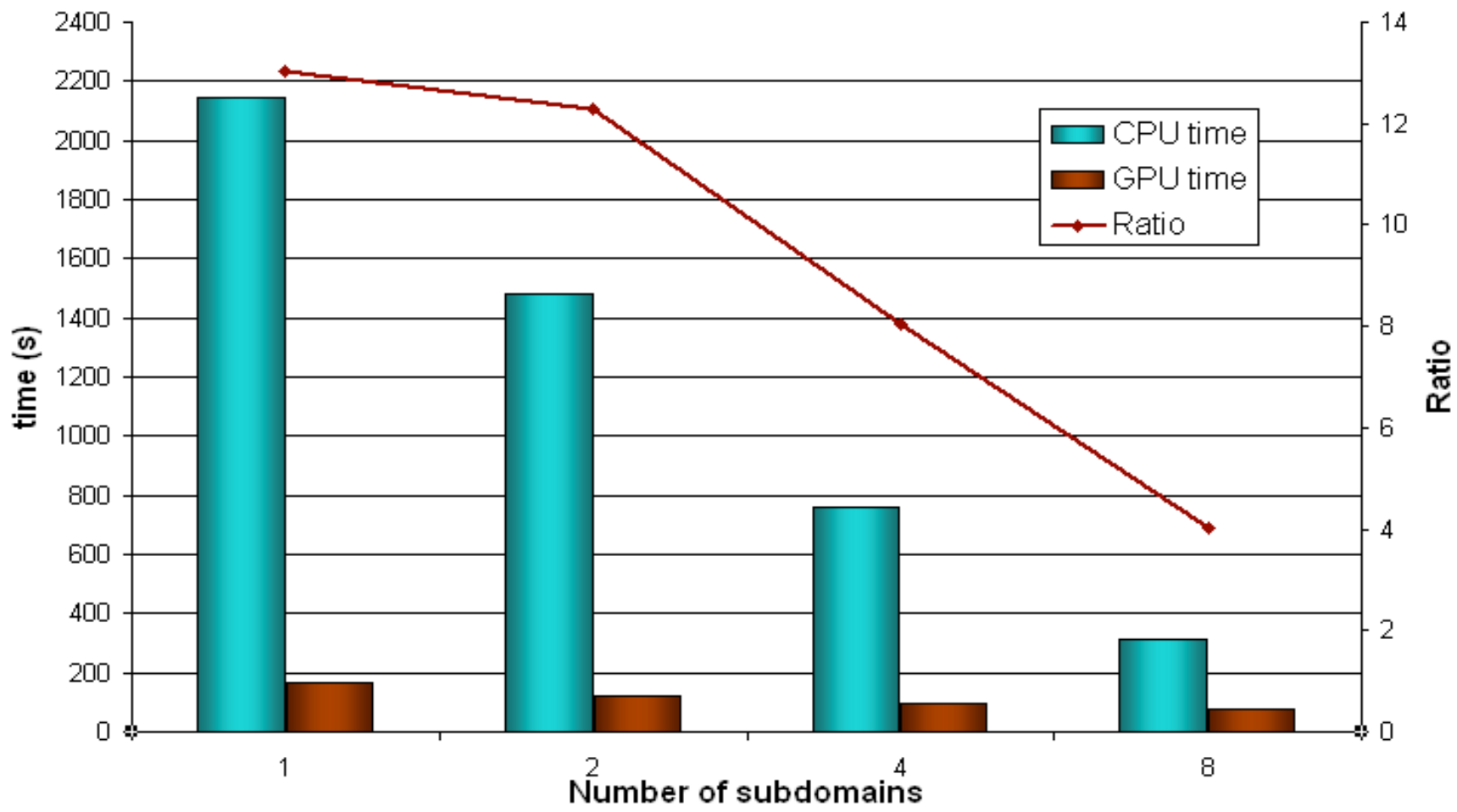
Seismic modeling

□ Example

- **Model size 560x557x905, 22760 time steps**
- **SGI ICE+, 512 harpertown 3GHZ**
- **SGI ICE+, 16 blades harpertown 3 GHZ, 8 Teslas**
- **8 cores per shot (64 shots in parallel)**
- **2 GPUs per shot (16 shots in parallel)**



Reverse Time Migration



Programming language

- ▶ **CUDA: progress rapidly, new features such as partial copy, asynchronous communications, pined memory data access... but no portability**
- ▶ **OpenCL: Standard ?(wait and see),**
- ▶ **HMPP: (www.caps-entreprise.com)**
 - ▶ **Express parallelism and communications in C, Fortran source code**
 - ▶ **offer a high level abstraction of GPU programming in scientific application**
 - ▶ **Direct integration through the use of directives “ à la OpenMP”**
 - ▶ **Ease application deployment in multi-GPUs systems**
 - ▶ **Ensure application interoperability**
 - ▶ **HMPP handles CPU-GPU communications and kernel launch.**
 - ▶ **HMPP manage automatically HW device.**

Porting effort: status

Solution 1: CUDA kernels written and optimized by hand

- Kernel integration with HMPP directives
- Optimization consisted in
 - Kernel execution
 - Speeding up data transfers
 - Hiding much as possible data transfers

Solution 2: CUDA kernels generator: Automatic code generation

Objectives: Hiding CUDA coding complexity

Take advantage of latest CUDA progress:

- Kernel execution
- Speeding up data transfers
- Hiding data transfers
- Zero memory copy...

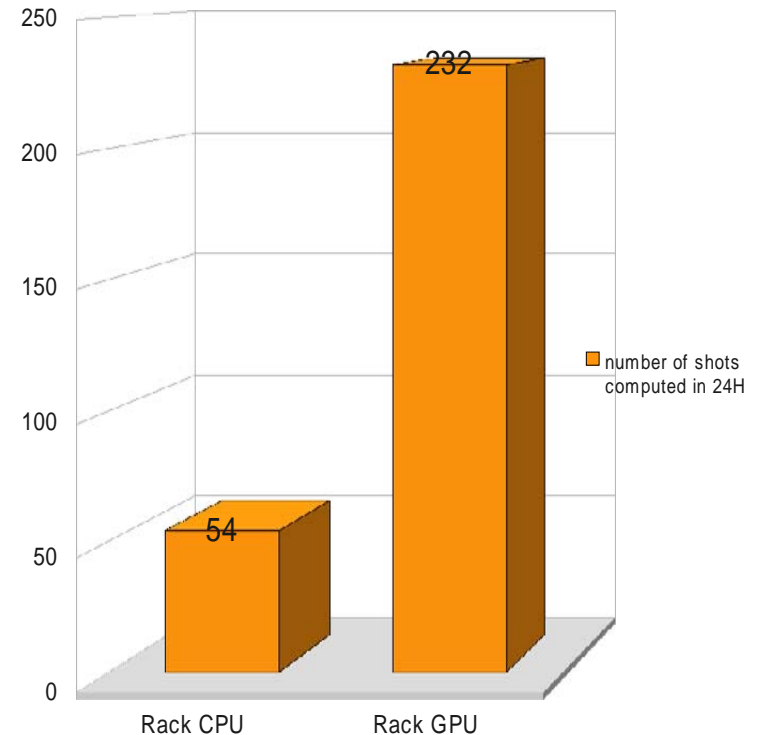
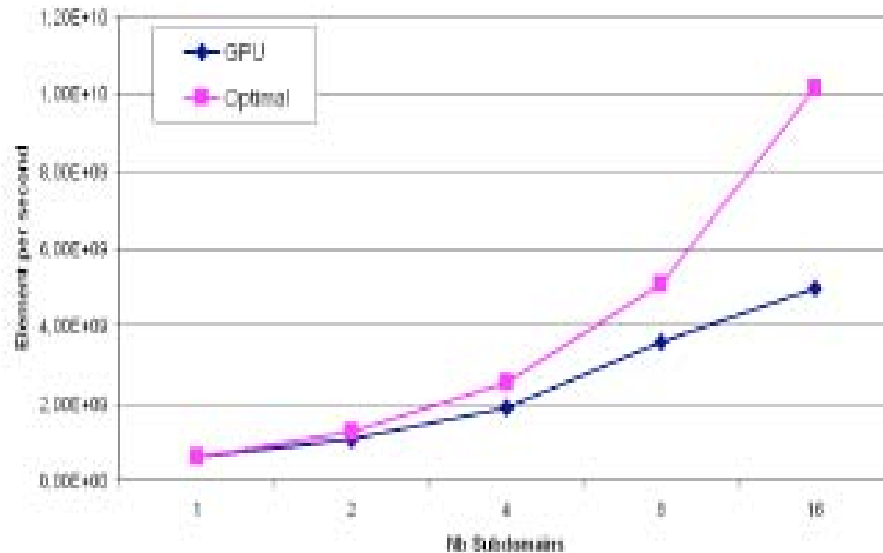
but still need to be improved: 50% slower than handed optimized code...

But progressing fast: parallel region, group of codelets, shared memory optimization...

Building industrial applications on a GPU cluster based system

Seismic modeling and RTM can take advantage of GPUS

Building a 256 GPUs system dedicated to seismic modeling and RTM



Still need some optimization work on Host-GPU communication and Load balancing

Multi GPU is possible but limited: needs more work

Conclusion

❑ Computation needs for seismic processing is nearly not bounded!

- Algorithm improvement taking advantage
 - Of high frequency seismic (impact the size of computational grid)
 - Seismic acquisition parameters (Azimuth width, distance between receivers, ...)
- Implement more complete solution of the imaging solver to solve new challenges: foothills, Heavy Oil....

❑ Need a pro-active R&D strategy

- Algorithms (Full Waveform Inversion, new Solvers, reservoir simulation, EM...)
- New Hardware architectures and programming models
- Impact of new technology on processing:
 - Several Teraflops in a WorkStation ?
 - HPC containers: On site processing, embedded processing

❑ Accelerating solution:

- good progress in understanding the use of GPUs
- we have a clear strategy : model programming and application development
- GPUs: first step before many core technology ?
- We do not exclude hybrid technology: (many core + GPUS)
- Still need to work in close relation with industrial partners to improve Hardware and tools.

What next ?

Looking for PetaFlop, ExaFlop: solutions ?

What if we target 10PF in 2012

1/ 4 cores technology (4 flops/cycle)

1 PF ~ 100000 cores ~ 160 racks (512 cores /rack) >3MW

10PF : multiply by 10

2/ 8 cores technology (8 flops/cycle)

1PF ~50000 cores ~ 50 racks (1024 cores/ rack) < 2MW

10PF: multiply by 10

→ **Solution many core technology (> 64cores)**

What next ?

→ Question: What technology ?

→ multi core “X86” like technology ?

→ GPU technology ?

→ **the most advanced to achieve such performances and constrains**

→ **but need to progress:**

→ integration host-device:

Bandwidth, memory access...

→ Model programming and compilers

→ efficiency: real Flops/ theoretical Flops

