

Portable task-based programming for Seismic Imaging

Lionel Boillot

▶ **To cite this version:**

Lionel Boillot. Portable task-based programming for Seismic Imaging. PRACEdays, May 2015, Dublin, Ireland. hal-01158967

HAL Id: hal-01158967
https://inria.hal.science/hal-01158967

Submitted on 2 Jun 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Portable task-based programming for Seismic Imaging

Lionel Boillot - Inria Magique3D, collaboration with ICL - University Of Tennessee

Depth Imaging Partnership (DIP), Inria-Total

Seismic Imaging

Several techniques exist to supply subsurface images. Among them, those based on seismic waves imposed, especially Reverse Time Migration (RTM) which focuses on the the interfaces detection, as illustrated on Fig. 1.

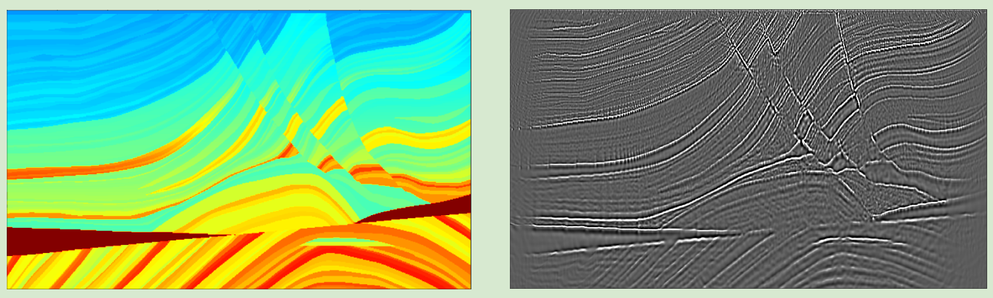
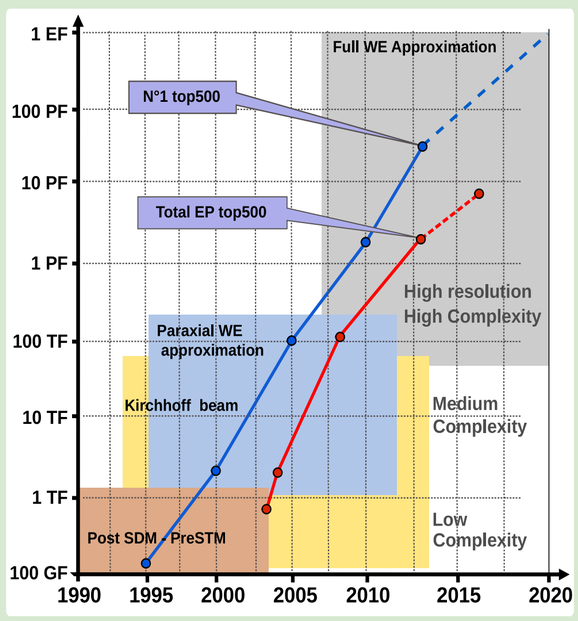


Fig. 1 - Marmousi model (left) and RTM result (right)



The accuracy level is directly related to the available computational power. Fig. 2 depicts this correlation where color blocks represent the different trends of seismic imaging techniques and the red curve is the power of the highest top500 supercomputer. RTM takes place on top right. In this strong HPC context, efficient and portable codes are mandatory.

Fig. 2 - Seismic Imaging (techniques) requirements in computational power and HPC evolution

From MPI to task-based

The space discretization is based on the Discontinuous Galerkin Method (DGM) coupled with a Leap-Frog time scheme, see e.g. [1]. This combining transforms the first-order elastic wave equation into an explicit scheme:

$$\begin{cases} \mathbf{v}^{n+1} &= \mathbf{v}^{n} - \Delta t M_{\mathbf{v}}^{-1} R_{\boldsymbol{\sigma}} \boldsymbol{\sigma}^{n+1/2} \\ \boldsymbol{\sigma}^{n+3/2} &= \boldsymbol{\sigma}^{n+1/2} - \Delta t M_{\boldsymbol{\sigma}}^{-1} R_{\mathbf{v}} \mathbf{v}^{n+1} \end{cases} \quad (1)$$

where the unknows are the velocity field $\mathbf{v}$ and the stress tensor $\boldsymbol{\sigma}$.

```
Input: [N_t, N_h]
Output: [v, σ]
N_{h_loc} ← DomainDecomposition(N_h)
[v^1, σ^{1/2}] ← Initialization(N_{h_loc})
For n in [1,N_t]:
  σ^{n+1/2} ← MPI_Communication(σ^{n+1/2})
  For k in [1,N_{h_loc}]:
    v_k^{n+1} ← UpdateVelocity(v^n, σ^{n+1/2})
  v^{n+1} ← MPI_Communication(v^{n+1})
  For k in [1,N_{h_loc}]:
    σ_k^{n+3/2} ← UpdateStress(σ^{n+1/2}, v^{n+1})
```

Algo. 1 - MPI-based parallel algorithm for (1), as implemented in the Total reference code

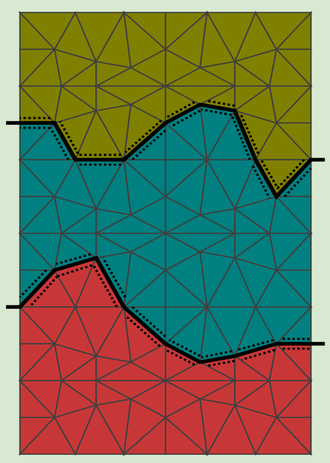


Fig. 3 - Domain decomposition on three subdomains

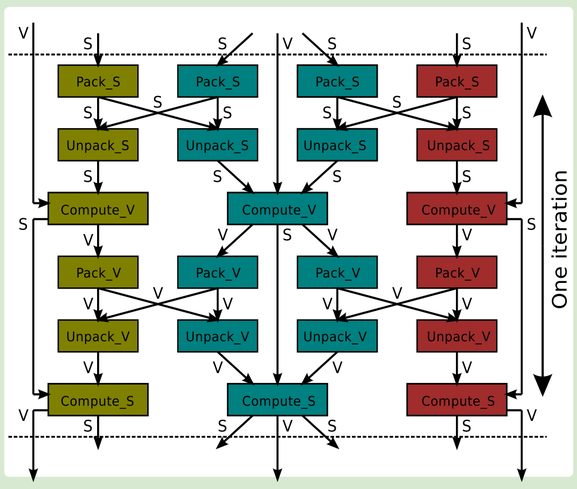


Fig. 4 - DAG of tasks for Algo. 1 on three subdomains (e.g. Fig. 3)

Task-based programming principle:

- describe the algorithm as a Direct Acyclic Graph (DAG) of tasks, without any reference to harware, as depicted Fig. 4;
- detail the dataflow, i.e. for each data -V and S- inside each task, what are the origin and the destination tasks.

The dataflow expression is made with the Parametrized Task Graph (PTG) abstraction specific language described in [2].

Performance Portabiliy

The runtime system P*a*RSEC [3] provides a generic framework for micro-task scheduling on distributed many-cores heterogeneous architectures, however, we first decided to address shared memory machines only.

Fig. 5 focuses on the single processor studying. This highlights the runtime system key features for task-based programming:

- coarse granularity does not allow any speedup;
- fine granularity is necessary for flexibility;
- work-stealing increases the load-balancing

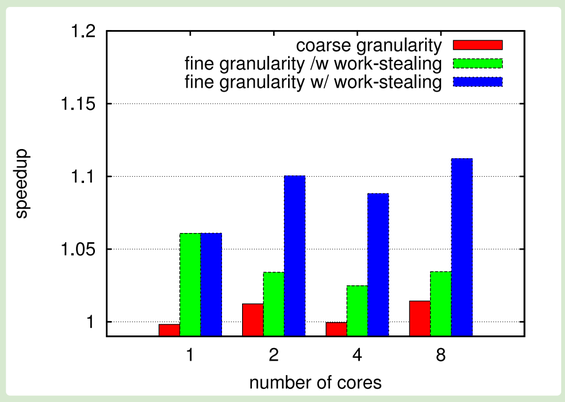


Fig. 5 - Speedup by the use of PaRSEC (tasks) over MPI-based code

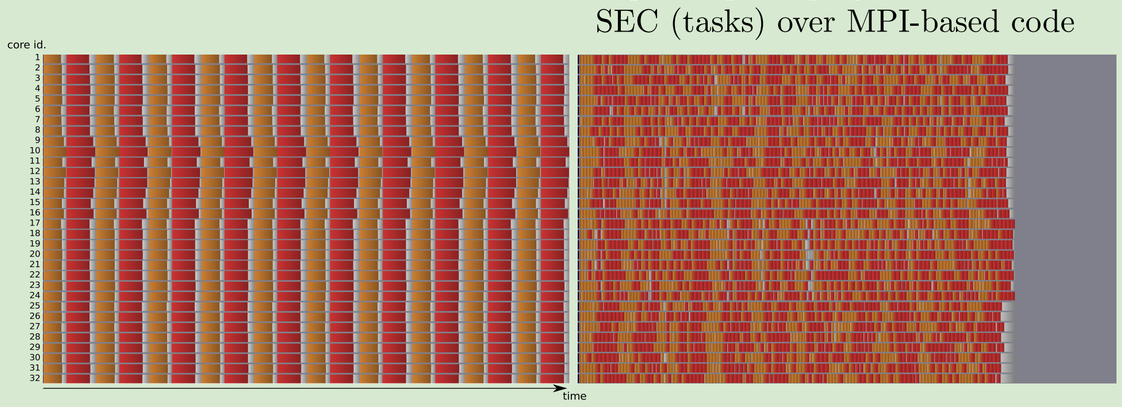


Fig. 6 - Trace execution on 32 cores for 10 timesteps, red and orange sections refer to computational kernels, light grey is the idle time, for MPI (left) and PaRSEC (right) codes, at the same time scale dark grey represents the gain

We extended single processor results on shared memory machines:

- a large cache-coherent Non-Uniform Memory Access (cc-NUMA) node, Fig. 6 is a selected trace comparison;
- an Intel Many Integrated Core (MIC) architecture accelerator, Fig. 7 shows the parallel efficiency in native mode.

The PaRSEC code remains unchanged, see further results in [4].

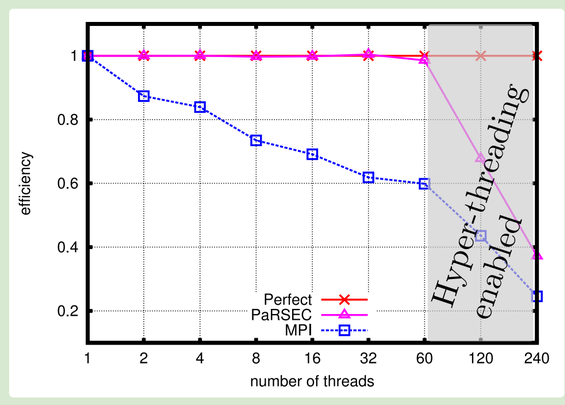


Fig. 7 - Parallel efficiency of PaRSEC and MPI-based codes on an Intel Xeon Phi co-processor

Perspectives

Preliminary results highlight the code and performance portability of our task-based programming model on different shared memory architectures. Ongoing work tackles distributed memory architectures.

The adaptability of the proposed solution to novel architectures will continue to be evaluated with a short-term goal targeting Intel Knight Landing many-core processors, especially inside hybrid nodes.

Bibliography

[1] Boillot L., *Contributions to the mathematical modeling and to the parallel algorithmic for the optimization of an elastic wave propagator in anisotropic media*, PhD thesis - Université de Pau, **2014**

[2] Danalis A., Bosilca G., Bouteiller A., Herault T., Dongarra J., *PTG: An Abstraction for Unhindered Parallelism*, Proceedings of the 4th International Workshop WOLFHPC, (10), 21-30, **2014**

[3] Bosilca G., Bouteiller A., Danalis A., Faverge M., Herault T., Dongarra J., *PaRSEC: Exploiting Heterogeneity to Enhance Scalability Computing*, Science Engineering, (15), 36-45, **2013**

[4] Boillot L., Bosilca G., Agullo E., Calandra H., *Task-based programming for Seismic Imaging: Preliminary Results*, Proceedings of the 16th IEEE International Conference HPCC, 1259-1266, **2014**