Guix et Org mode, deux amis du doctorant sur le chemin vers une thèse reproductible

Atelier reproductibilité des environnements logiciels
17 et 18 mai 2021

Marek Felšöci
marek.felsoci@inria.fr

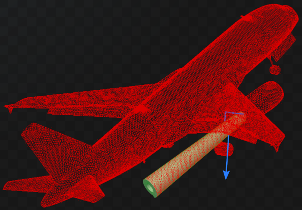
Solveurs rapides pour l'aéroacoustique haute-fréquence

AIRBUS

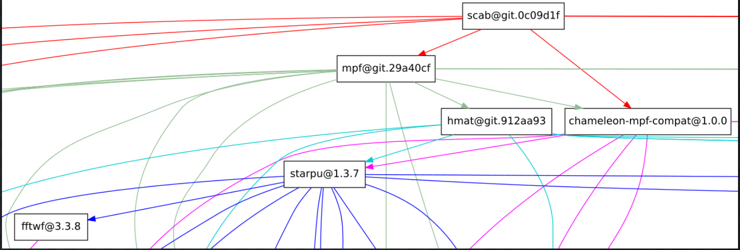
RÉGION Nouvelle-Aquitaine

- construction de modèles aéroacoustiques pour étudier la propagation des ondes sonores émises par un avion
- résolution de grands systèmes linéaires couplés
  - $> 10^9$ d'inconnues
  - composés de parties creuses et denses

Image de Sebaso [7]



Défis d'une thèse reproductible



1. environnement logiciel riche
   - beaucoup de dépendances
   - versions multiples
   - différentes plateformes de calcul
2. campagnes expérimentales
   - aisément extensibles
   - automatisées

Choix des outils

1. Gestionnaire de paquets Guix [3]
   - gestion efficace de plusieurs environnements
   - reproductibilité à travers différentes plateformes de calcul

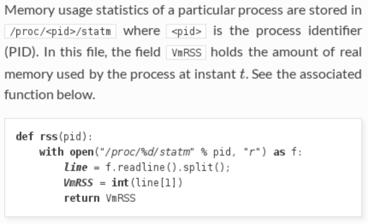
```
guix environment --pure --with-input=pastix-5=pastix-5-mkl \
     --with-input=mumps-scotch-openmpi=mumps-mkl-scotch-openmpi \
     --with-input=openblas=mkl --with-git-url=gcvb=$HOME/src/gcvb \
     --with-commit=gcvb=40d88ba241db4c71ac3e1fe8024fba4d906f45b1 \
     --preserve=^SLURM --ad-hoc bash coreutils inetutils findutils \
     grep sed bc openssh python python-psutil gcvb scab slurm@19 openmpi
```

Choix des outils

2. Org mode pour Emacs [4]
   - programmation littérale [6]
   - documentation détaillée de l'environnement et des expérimentations

```
#+PROPERTY: header-args :tangle rss.py ...
...
Memory usage statistics of a particular process are
stored in ~/proc/<pid>/statm~ where ~<pid>~ is the
process identifier (PID). In this file, the field
=VmRSS= holds the amount of real memory used by the
process at instant $t$. See the associated function
below.

#+BEGIN_SRC python
def rss(pid):
    with open("/proc/%d/statm" % pid, "r") as f:
        line = f.readline().split();
        VmRSS = int(line[1])
        return VmRSS
#+END_SRC
...
```


Memory usage statistics of a particular process are stored in `/proc/<pid>/statm` where `<pid>` is the process identifier (PID). In this file, the field `VmRSS` holds the amount of real memory used by the process at instant t. See the associated function below.

```python
def rss(pid):
    with open("/proc/%d/statm" % pid, "r") as f:
        line = f.readline().split();
        VmRSS = int(line[1])
        return VmRSS
```

```python
#!/usr/bin/env python3
...
def rss(pid):
    with open("/proc/%d/statm" % pid, "r") as f:
        line = f.readline().split();
        VmRSS = int(line[1])
        return VmRSS
...
```

Vers un environnement logiciel reproductible

Contraintes

- plusieurs environnements différents
  1. pré-traitement
  2. exécution des tests de performance
  3. extraction et collecte des résultats
  4. formattage des résultats
  5. post-traitement
- plusieurs versions de notre pile logicielle (de nouveaux algorithmes ou améliorations implémentés)
- plusieurs plateformes (avec ou sans Guix)
  1. PlaFRIM (Inria Bordeaux) [5]
  2. Curta (Nouvelle Aquitaine) [1]
  3. Occigen (GENCI) [2]

Canaux Guix

- répertoires Git définissant les paquets disponibles
- publics ou privés (Airbus)
- définis dans `$HOME/.config/guix/channels.scm` (niveau utilisateur courant)

```
(list (channel (name 'guix)
               (url "https://git.savannah.gnu.org/git/guix.git")
               (commit "1ac4959c6a94a89fc8d3a73239d107cfb1d96240"))
 (channel (name 'guix-hpc)
          (url "https://gitlab.inria.fr/guix-hpc/guix-hpc.git")
          (commit "9cc4593aaaaeaf17a602b620e9ab1974b5b82984"))
 (channel (name 'guix-hpc-airbus)
          (url "git@gitlab.inria.fr:mfelsoci/guix-hpc-airbus.git")
          (commit "bff0aa3f20140dbfda53d6882e25d3e9770e2911"))
...
```

Vers un environnement logiciel reproductible

Spécification et entrée en environnement

```
guix environment option1 option2 paquet1 paquet2 ...
```

Gestion simplifiée avec un script maison

- entrer plus facilement dans des environnements personnalisés
- créer des images Singularity correspondantes si besoin
- éviter de taper de longues commandes `guix environment` ou `guix pack` à la main

```
./setenv.sh -e "benchmarks"

./setenv.sh -e "gather"

./setenv.sh -e "post_process"

./setenv.sh -e "benchmarks" -s env-benchmarks.gz.squashfs
```

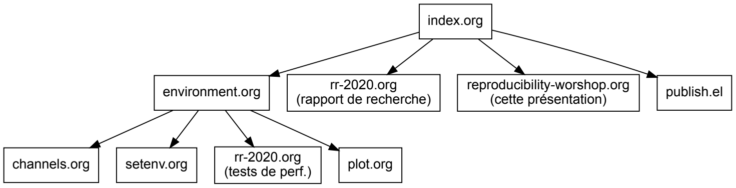
Vers une thèse reproductible

Un pas de plus...

- environnement logiciel reproductible
  - indépendamment de la plateforme (autant que possible)
- expérimentations reproductibles
  - présentation de cet environnement logiciel et de sa mise en place
  - description exhaustive des expérimentations
  - explication du traitement et visualisation des résultats
  - ...

Un dépôt central sur GitLab

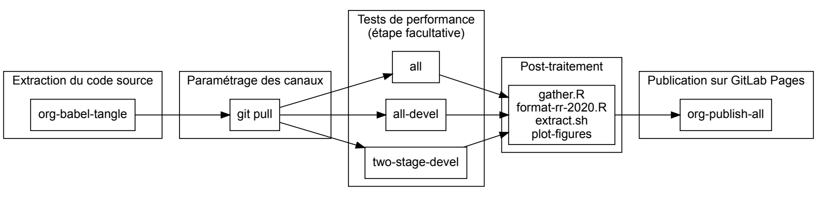
- le format Org privilégié pour tout document texte et code source
  - publiable sur un site Web accessible à tous (GitLab Pages)
- expérimentations et post-traitement automatisés
  - intégration continue (GitLab CI)



Vers une thèse reproductible

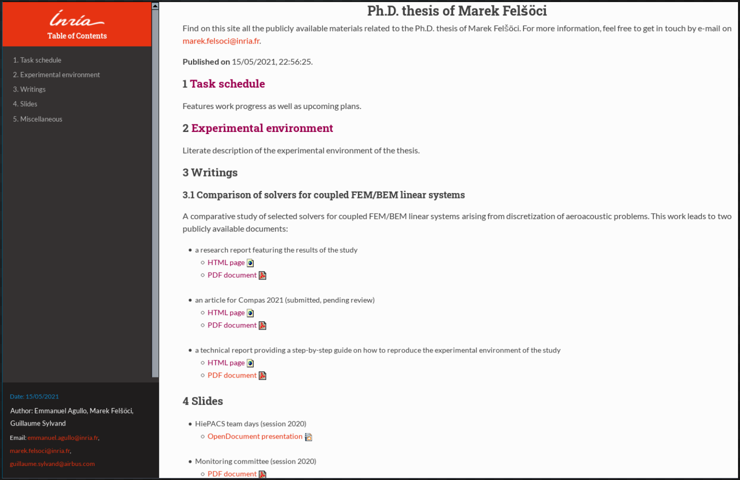
Intégration continue

- moyen d'automatisation et de test de non-régression
- séquence de tâches exécutée après chaque `git push`
- s'appuie sur `setenv.sh` et Singularity



Résultat final

`https://mfelsoci.gitlabpages.inria.fr/thesis/`



Inria
Table of Contents

Date: 15/05/2021

Author: Emmanuel Agullo, Marek Felšöci, Guillaume Sylvand

Email: emmanuel.agullo@inria.fr, marek.felsoci@inria.fr, guillaume.sylvand@airbus.com

Ph.D. thesis of Marek Felšöci

Find on this site all the publicly available materials related to the Ph.D. thesis of Marek Felšöci. For more information, feel free to get in touch by e-mail on marek.felsoci@inria.fr.

Published on 15/05/2021, 22:56:25.

1 Task schedule

Features work progress as well as upcoming plans.

2 Experimental environment

Literate description of the experimental environment of the thesis.

3 Writings

3.1 Comparison of solvers for coupled FEM/BEM linear systems

A comparative study of selected solvers for coupled FEM/BEM linear systems arising from discretization of aeroacoustic problems. This work leads to two publicly available documents:

- a research report featuring the results of the study
  - HTML page
  - PDF document
- an article for Compas 2021 (submitted, pending review)
  - HTML page
  - PDF document
- a technical report providing a step-by-step guide on how to reproduce the experimental environment of the study
  - HTML page
  - PDF document

4 Slides

- HiePACS team days (session 2020)
  - OpenDocument presentation
- Monitoring committee (session 2020)
  - PDF document

Résultat final

https://mfelsoci.gitlabpages.inria.fr/thesis/

Inria

Table of Contents

Author: Emmanuel Agullo, Marek Felšöci, Guillaume Sylvand

Email: emmanuel.agullo@inria.fr, marek.felsoci@inria.fr, guillaume.sylvand@airbus.com


```
Tests:
```

Firstly, we want to benchmark the SPIDO solver on dense BEM systems for various unknown counts. Under `template_instantiation` there are two array-like constructs later expanded by GCVB to generate multiple variants of the SPIDO benchmark, e. g. for various problem sizes. `slurm` holds the common job name prefix and the scheduling information used for the generation of the associated `sbatch` header file, here based on the template defined in Listing 1. The `nbpts` array defines the problem sizes to generate benchmarks for. Note that, `{slurm[prefix]}`, `{slurm[platform]}, ={nbpts}` and so on are the placeholders for the values defined in `template_instantiation`.

Given the current `template_instantiation` configuration, we generate $1 \times 3 = 3$ variants of the SPIDO benchmark grouped into a single job script with a time limit of 2 hours.

```
    -
      id: "spido-{nbpts}"
      template_files: "monobatch"
      template_instantiation:
        slurm:
          - { prefix: "spido", platform: "plafrim", node: "miriel", count: 1,
              tasks: 24, time: "0-02:00:00" }
        nbpts: [ 25000, 50000, 100000 ]
```

Follows the task corresponding to this benchmark. The launch command is read from the list of default values defined at the beginning of the file. We only override here the `nthreads` key to set the count of OpenMP and MKL threads to use for the computation. The values are propagated to the launch command through the `{@job_creation[options]}` placeholder.

```
      Tasks:
        -
          nthreads: "OMP_NUM_THREADS=24 MKL_NUM_THREADS=24"
          options: "--bem -withmpf -nbpts {nbpts}"
```

For the corresponding validation phase we need to specify an identifier as well as a launch command composed of the validation `executable` obtained here through the `{@job_creation[va_executable]}` placeholder, and some options specific to this benchmark such as the information on the solver used, the target platform as well as the variation of benchmark to make a difference between regular benchmarks based on parameter variation and scalability benchmarks and the target platform.

```
          Validations:
            -
              id: "validation-spido-{nbpts}"
              launch_command: "{@job_creation[va_executable]} -K solver=spido
              -K variation=parameters,platform={slurm[platform]}
```

Résultat final

https://mfelsoci.gitlabpages.inria.fr/thesis/

Inria
Table of Contents

Author: Emmanuel Agullo, Marek Felšöci, Guillaume Sylvand

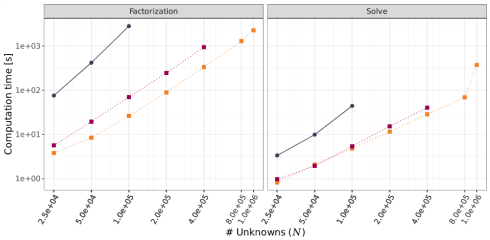
Email: emmanuel.agullo@inria.fr, marek.felsoci@inria.fr, guillaume.sylvand@airbus.com

7.5 Result analysis

7.5.1 Dense BEM systems

In the first place, we evaluate the performance of the direct solvers SPIDO and HMAT (see sections 4.2.1 and 4.2.2) on dense linear systems resulting from BEM discretization and counting from 25,000 up to 1,000,000 unknowns. We consider symmetric coefficient matrices and rely on LDL^T factorization in case of SPIDO and LL^T factorization in case of HMAT (see Section 4).

According to Figure 18, the computation times of SPIDO are significantly higher compared to HMAT. For illustration, the factorization time of SPIDO on a system with 100,000 unknowns is almost the same as the factorization time of HMAT on a system having 1,000,000 unknowns. Also, without out-of-core (see Section 7.3), SPIDO quickly approaches the 126 GiB memory limit of the miriel nodes (see Section 7.4.1) leaving it unable to process linear systems with 200,000 or more unknowns as shown in Figure 19. All of this witness the advantages of the hierarchical matrix structure implemented in HMAT (see Section 4.2.2) allowing the solver to not only speed up the computations but also to lower the memory footprint and process linear systems with up to 1,000,000 of unknowns when the solution accuracy threshold ϵ is set to 10^{-3}. Nevertheless, when we tighten up the threshold to 10^{-6}, the factorization time of HMAT increases more rapidly and the memory limit is reached sooner too, compared to runs with ϵ set to 10^{-3}. Consequently, runs on systems counting more than 400,000 (see Figures 18 and 19) fail due to the lack of memory. The difference is naturally less noticeable for the solve phase having a considerably lower complexity compared to factorization.



Résultat final

https://mfelsoci.gitlabpages.inria.fr/thesis/

Inria

A comparison of selected solvers for coupled FEM/BEM linear systems arising from discretization of aeroacoustic problems

Emmanuel Agullo*, Marek Felšöci†, Guillaume Sylvand‡

Project-Team PANIC

Research Report n° ??? — ??? — 50 pages

Abstract: When discretization of an aeroacoustic physical model is based on the application of both the Finite Elements Mehotd (FEM) and the Boundary Elements Method (BEM), this leads to coupled FEM/BEM linear systems combining sparse and dense parts. In this preliminary study, we compare a set of sparse and dense solvers applied on the solution of such type of linear systems with the aim to identify the best performing configurations of existing solvers.

Key-words: aeroacoustics, modelization, finite elements, boundary elements, solver comparison, coupled FEM/BEM linear systems, sparse and dense matrices

* Inria Bordeaux Sud-Ouest (emmanuel.agullo@inria.fr)
† Inria Bordeaux Sud-Ouest (marek.felsoci@inria.fr)
‡ Airbus Central R&T / Inria Bordeaux Sud-Ouest (guillaume.sylvand@airbus.com)

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

200 avenue de la Vieille Tour
33405 Talence Cedex

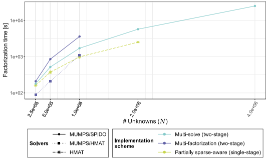


FIGURE 38: Best computation times of the two-stage **multi-solve** and the **multi-factorization** implementations (see Section 5.3) compared to the single-stage **partially sparse-aware** scheme (see Section 5.3) using HMAT on coupled FEM/BEM systems. Parallel runs using 1 MPI process, 24 OpenMP, MKL threads and StarPU workers on single miriel node with the precision parameter ϵ (see Section 6.2) set to 10^{-3} in case of MUMPS and HMAT. Out-of-core has been disabled.

8 Conclusion

In this report, we have presented our study on the performance of selected sparse and dense direct solvers committed to solve coupled FEM/BEM linear systems arising from the discretization of aeroacoustic problems relying on the Finite Elements Method (FEM) and Boundary Elements Method (BEM).

The first part of the experiments was dedicated to the evaluation of dense direct solvers SPIDO and HMAT on the solution of dense BEM systems. We have observed that the hierarchical matrix structure of HMAT provides the solver with an important performance advantage over SPIDO, both in terms of computation time and memory consumption. On the other hand, on single computation node, HMAT scales well only if it relies exclusively on StarPU worker parallelism. Its MPI parallelization does not seem to be optimized for usage on one node. SPIDO scales well in all parallel configurations.

The second part was aimed to evaluate sparse direct solvers MUMPS and HMAT on the solution of sparse FEM systems. The results showed a noticeably better performance of MUMPS compared to HMAT on sparse matrices. The implementation limitations, including the prototype of Nested Dissection, in HMAT which was primarily meant for dense matrices does not allow the solver to process as large linear systems as MUMPS and consumes more memory and computation time than the latter considering the same hardware. MUMPS scales well for all parallel configurations. The scalability of HMAT on sparse matrices presents the same tendencies as on dense matrices.

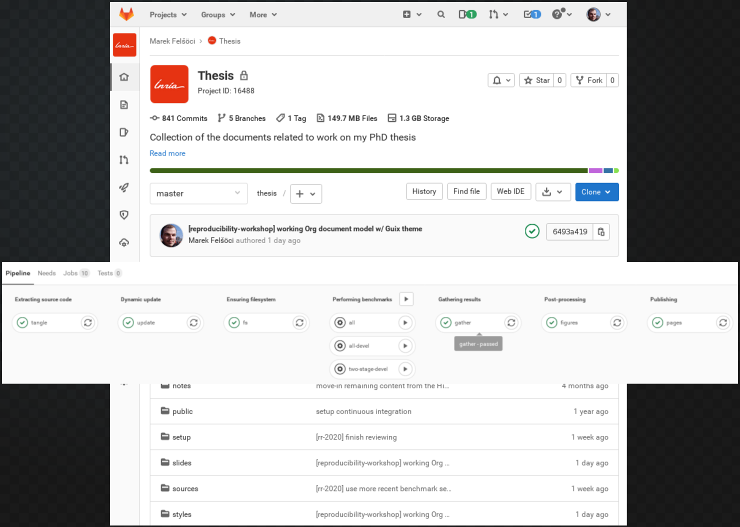
We have evaluated different implementation schemes involving various solvers for the solution of FEM/BEM linear systems in the third and last part of the experimental study. According to our findings, the worst performing two-stage implementation scheme was the multi-factorization

Guix

Inria

Résultat final

https://mfelsoci.gitlabpages.inria.fr/thesis/



Moi et Guix

- quelques expériences personnelles avec Guix et Org mode :
  `https://felsoci.sk/blog/posts.html`
- mon propre canal Guix (QtCreator IDE, utilitaire svgfix et quelques tentatives dans la branche `unstable`) :
  `https://gitlab.inria.fr/mfelsoci/guix-extra`
- traduction du système Guix en slovaque preložené 100% :
  `https://translate.fedoraproject.org/projects/guix/guix/sk/`

Avez-vous des questions ?

N'hésitez pas à me contacter même après le séminaire sur marek.felsoci@inria.fr !

- *Cluster Curta, Mésocentre de Calcul Intensif Aquitain.* `https://www.mcia.fr/projects/cluster-curta`.
- *GENCI: Grand équipement national de calcul intensif.* `http://www.genci.fr/fr/content/calculateurs-et-centres-de-calcul`.
- *GNU Guix software distribution and transactional package manager.* `https://guix.gnu.org`.
- *Org mode for Emacs.* `https://orgmode.org/`.
- *PlaFRIM: Plateforme fédérative pour la recherche en informatique et mathématiques.* `https://plafrim.fr/`.
- D. E. Knuth, *Literate programming*, Comput. J., 27 (1984), p. 97–111.
- Sebaso, *Jet engine airflow during take-off.* `https://commons.wikimedia.org/wiki/File:20140308-Jet_engine_airflow_during_take-off.jpg`.