Improving the execution of workflows for SAR image analysis

Matthieu Volat, Flavien Vernier, Marie-Pierre Doin, Cécile Lasserre, Emmanuel Trouvé, Erwan Pathier

Improving the execution of workflows for SAR image analysis

Conference Paper · November 2014

CITATIONS
0

READS
89

6 authors, including:

Matthieu Volat
Claude Bernard University Lyon 1
7 PUBLICATIONS **12** CITATIONS
SEE PROFILE

Flavien Vernier
Université Savoie Mont Blanc
38 PUBLICATIONS **231** CITATIONS
SEE PROFILE

Cecile Lasserre
Claude Bernard University Lyon 1
99 PUBLICATIONS **1,963** CITATIONS
SEE PROFILE

Emmanuel Trouvé
Université Savoie Mont Blanc
174 PUBLICATIONS **1,402** CITATIONS
SEE PROFILE

IMPROVING THE EXECUTION OF WORKFLOWS FOR SAR IMAGE ANALYSIS

Matthieu Volat, Flavien Vernier, Marie-Pierre Doin, Cecile Lasserre, Emmanuel Trouvé, Erwan Pathier

Matthieu Volat, Marie-Pierre Doin, Cecile Lasserre, Erwan Pathier
ISTerre CNRS - Université Joseph Fourier -
Maison des Géosciences -
1381 rue de la piscine - 38400 Saint Martin d'Hères, France
Flavien Vernier, Emmanuel Trouvé
LISTIC - Polytech Annecy-Chambéry, Université de Savoie -
BP 80439 - 74944 Annecy le Vieux Cedex, France

1. INTRODUCTION

While the resolution and repetitivity of SAR data has strongly increased in the last years, the execution of the processing workflows remained mostly the same. Many scientists continue to manage repetitive tasks over many files either with crude scripts or by hand. Moreover, the quantity of available SAR data increases more and more with the satellite generations and the "free and open access" data policy adopted by some space agencies.

In this paper, we propose a task-based framework oriented toward simpler and better workflow execution that deals transparently for the user with multi-core architectures and cluster or grid systems. The workflows are based on two sets of processing tools: the EFIDIR tools [1] [2] developed in collaboration between several French Earth sciences Image processing laboratories and the ISTerre's NSBAS chain [3], based on the JPL/Calltech ROI_PAC [4] project. These tools are the implementation ground for our framework. Our use-case is the processing of ascending and descending repeated orbits SAR image time series from raw data to surface displacement with those tools.

2. GENERAL PURPOSE

First, we investigated the qualities required by such a framework. Speed improvements through distributed computing is one of the main goals, but it is not the only one. Practical usage of both software showed that easiness of workflow construction, errors and interruption handling, resumption/recovery were equivalently important. This is especially true when distributed execution makes the process more complicated.

While the EFIDIR project focused on rigorously defined programs with single purpose, the NSBAS project is a loosely integrated collection of scripts calling either ROI_PAC or internal programs. NSBAS has a small, but active, community of developers and end-users that rely on the software to do their work. Implementing a complete workflow solution would have mean a total rewrite that is not acceptable by the team and the users. So we decided to use least intrusive techniques to improve the chain efficiency without changing too much code.

On the other hand, EFIDIR's workflow was to be written from scratch, which left us room for a clean implementation. We decided that a more rigorous approach would be used in EFIDIR, with an emphasis on both efficiency through distributed computing and execution control quality. A NSBAS rewrite is planned later on the basis of the EFIDIR workflow framework.

In both cases, we noticed that the atomic operation would be the execution of a program. We also observed that in both case, we had a strongly input/output oriented workflow were intermediate results had to be kept. This led us to think of our processing as a graph of program execution connected by their inputs and outputs. We did not assume anything about the average program duration, since we observed instances of a few seconds to a few hours. We also wanted it to be easily deployable on the target environments, which would be barebone computer clusters with no available display and little preinstalled libraries.

We analyzed a few existing frameworks proposed by various entities and communities, such as DAGMan [1], ipython task interface [2], and soma-workflows [3]. In each of those tools, we found blocking points, such as specific jobs scheduler dependencies and integration, reliance on frameworks not easily available in the computing environments. So we decided to implement our own solution. That would mean that we provide less features, but enough to cover our needs and that runs our solution in situation others would not.

[1] http://research.cs.wisc.edu/htcondor/dagman/dagman.html
[2] http://ipython.org/ipython-doc/rel-0.12.1/parallel/parallel_task.html
[3] http://brainvisa.info/soma/soma-workflow/

2.1. NSBAS

NSBAS is a processing chain that handle InSAR computation from raw data to time series analysis [3]. A large part of the chain is based on ROI_PAC programs, with original routines rearranged and combined with new routines to process in series and common radar geometry (figure 1 shows NSBAS output compared to ROI_PAC). NSBAS ground deformation measurment is based on the phase information by interferometry.

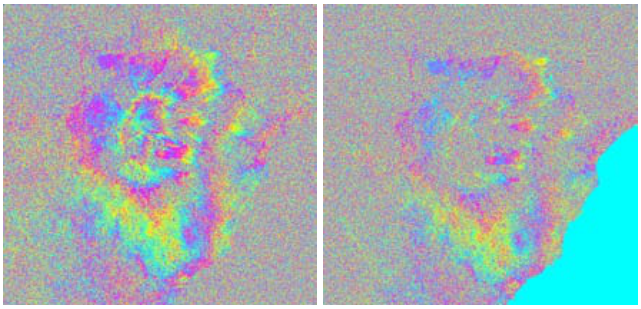

Fig. 1. Comparison of wrapped interferogram produced by NSBAS (left) and ROI_PAC (right) bewteen the 2010-06-09 and the 2010-09-22, using ERS images on Mount Etna. One colour cycle represent 2.8cm of phase signal.

As said previously, we operated with a set of constrains in this context. Moreover, modifications to the ROI_PAC files cannot not be accepted by the upstream team.

We observed that many tasks processed independently the input data set. Basic parallelism was already implemented to split the execution in multiple process. This system can scale up to the resources on one host. To bring this code to the next scale, we implemented a small task dispatcher that tries to mimic the previous one. This new dispatcher uses the MPI standard, so it can create and execute process over a pool of hosts. This allows us to scale the parallelism up to the size of the input size: for example, if we have 63 images to process, a single host would not have enough cores available to process all of them in a single pass, but 4 hosts with each 8 cores do. In regard of the task set scale (at most, a few thousands) a simple single master with multiple workers process model worked quite well.

2.2. EFIDIR

The EFIDIR tools package is a set of programs to process images (SAR and optic), and provides tools for many basic and complex operations such as large displacements [1], displacement fusion [2], look up tables and so on (figure 2 shows results produced by EFIDIR).

Taking into account the observations from 2.1, we noticed that a file input/output based process was very similar to the software building process. We decided to implement a system

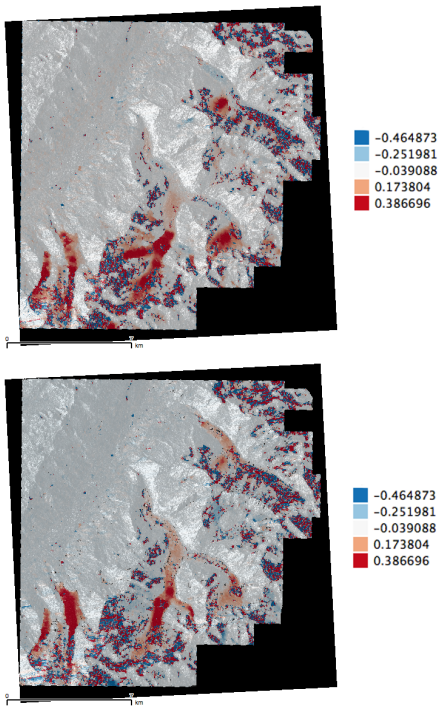

Fig. 2. 2D displacement field in cm (top is east displacement, bottom is north) of Bossons, Argentière & Taconnaz glaciers between the 2009-11-03 and 2011-09-25 using TerraSAR-X images.

based on the *make* build automation system, which would allow us to delegate tasks such as task dependency resolution, error handling and resumption.

This system was capable of scaling up to the resources of a single machine, but we wanted to be able to use more power if possible. Then, we paired it with a dispatcher to allow remote execution on a cluster of computers. Once again, we designed it taking into account the observations from NSBAS in 2.1: we also used a MPI-based system which relied on a single master processus due to the scale of the envisioned experiments. But, contrary to what was implemented in NSBAS, this dispatcher is a clean subsystem separated from the chain itself dedicated to the execution of subcommands.

Linking both systems was simple. Since we generate *make* input, we could generate commands that were called through a wrapper program that send the command execution towards the dispatcher. Since this system to work under *make*, the wrapper is synchronous and blocks until the dispatcher returns the task completion. In this context, *make* is started specifying a number of maximum concurrent jobs equals to the process pool size minus one (to account for the master).

The critical aspect of those systems is that they are meant to do a lot of their work in a invisible way to the user. We proposed a simple API which requires the user to specify, for each elementary task:

- which commands are to be run
- which files are needed (inputs)
- which files are produced (outputs)

Since the system is based on *make*, the dependency tree is created automatically from those. To ease this workflow description, we allowed the processing to be written as Python scripts. For example, to convert a serie of images to jpeg:

```
targets = []
for f in filelist:
  outputs = [ f+".jpeg" ]
  inputs = [ f ]
  commands = [
    "convert_efidir_to_jpeg"
      +" --input %s" % (inputs[0])
      +" --outputs %s" % (outputs[0]) ]
  t = target_create(commands,
                    inputs,
                    outputs)
  targets.append(t)

chain = chain_create("conversion", targets)
chain_run(chain)
```

This framework allows both complex and large scale automation and makes a lot of operations easier.

3. EXPERIMENTAL RESULTS

Our quantitative evaluation focused mainly on processing speeds. To that end, we have access to two platforms:

- A 12 Xeon cores host with 64GB of memory, hosted at the ISTerre laboratory. Dedicated to InSAR processing, this is our reference for state-of-the-art laboratory server.
- The other is the Froggy platform of the CIMENT infrastructure [4].

3.1. NSBAS

In order to evaluate the performance boost of our work in NSBAS, we processed a zone of roughly 50x50 km centered on Mount Etna, Sicily, which was used as illustration of figure 1. The input set is constitued of 63 images from the ERS satellite. Details about NSBAS processing are described in Doin & al 2011[3], but can be summarized as:

- Convert data from raw to Single Look Complex images
- Coregister a Digital Elevation Model to a "master" image
- Coregister all the images to this master
- Create an inteferogram list and compute them

Those steps are usually followed by more refinements, unwrapping and inversion, but as our work is not complete in those parts, they were left out from the benchmarking process.

Those steps were run on both our test platforms. The results of this experiment are presented on figure 3.

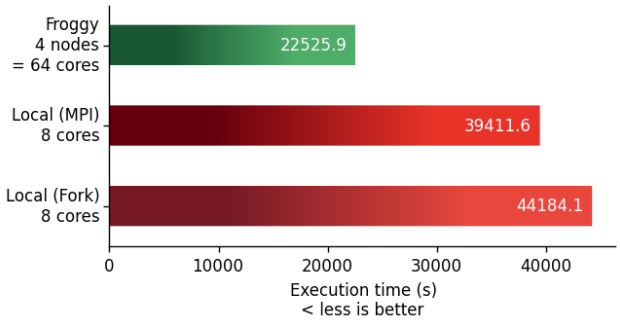

Fig. 3. NSBAS benchmark

In theory, when using a single host, the MPI version should be slower than the old fork() version, due to the usage of a process to coordinate tasks. In practice, we are seeing the MPI version begin a little faster, mainly due to better coding practices used during the rewrite. Moving to the cluster, which only the MPI implementation can do, results in a execution twice faster.

But, to achieve this, we used four time more computing power, which means our efficiency factor is only 0.5 on this scale. There is mainly two reasons for this result. The first is that while we scaled up pure computing power, we did not scale storage performance at the same rate, which lessen the speedup.

The second is the fact that task based distribution is bound to hit a limit when the ideal ressource size is attained. At this point, the execution of specific tasks is, once again, the most time consuming part of the processing.

With that point in mind, and our target being to improve execution speed, we find those results satisfying.

3.2. EFIDIR

The first EFIDIR chain developed is dedicated to the computation of 2D displacements from SAR Images by pixel offset tracking based on amplitude (whereas NSBAS use phase interfometry). The workflow of this chain is described by the following main steps:

- an initial co-registration is performed
- 2D displacements are computed for pairs of images

[4] https://ciment.ujf-grenoble.fr

- a range/azimuth offset correction is performed
- the corrected 2D displacements are ortho-rectified

This chain is executed on a set of 3 TerraSAR-X stripmap images of the Chamonix - Mont Blanc valley. The objective is to monitor the glacier flows of this valley. Figure 2 illustrates one of the results.

Using our test platforms, we measured the duration of the processing, starting with a scripted execution up to using the fully distributed workflow system. The results are presented on figure 4.

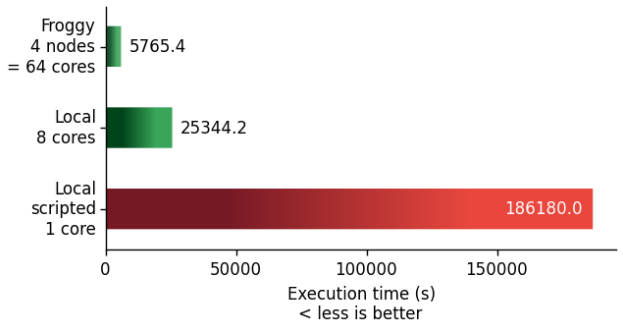

Fig. 4. EFIDIR benchmark

Compared to a scripted, single-core, execution, the benefits of the system are overwhelming. Despite the input set being only 3 images, execution speed profit greatly from much more resources as the most time-consuming task (correlation) was adapted to use the framework system to split and process images in regions.

4. FUTURE WORK

There are still perspectives to the present work. The first is the usage of chains as elements of others chains. Our first approach showed that *make* would not communicate properly the pool size in those cases and revert to linear execution. We are working to fix this and allow ressources to be splitted as needed between sub-workflows.

The other area is file storage systems. Experience with CIMENT showed us we could not expect data to be available in a single, shared, filesystem, but could be presented through various interfaces. We need to include this aspect in our framework to also automate data retrieval and submission.

We would also intend to deploy those tools on ESA's GPOD platform, and are currently collecting information about how to do so.

5. CONCLUSION

In this paper, we presented two scenario : one where we improve only the speed of execution with as little changes as possible, and the other where we implements a complete framework. Both aims to improve the workflow of non-realtime large optical and SAR image series processing using the power of distributed computing.

Our analysis showed that despite the common usage of those resources and existing tools, the best place to implement this framework if you operate under certain constrains is the software packages themselves.

We also provided a first speedup increment and, in the case of the EFIDIR tools, a more robust way of executing theses tasks.

6. ACKNOWLEDGEMENTS

This work was sponsored by the CNES (France) through the TOSCA/CESTENG project, and we received additional support through the Tera-SAR project (défi Mastodons) sponsored by the CNRS (France).

Part the computations presented in this paper were performed using the Froggy platform of the CIMENT infrastructure, which is supported by the Rhne-Alpes region (GRANT CPER07_13 CIRA), the OSUG@2020 labex (reference ANR10 LABX56) and the Equip@Meso project (reference ANR-10-EQPX-29-01) of the programme Investissements d'Avenir supervised by the Agence Nationale pour la Recherche.

7. REFERENCES

[1] F. Vernier, R. Fallourd, J. Friedt, Y. Yan, E. Trouvé, J.-M. Nicolas, and L. Moreau, "Fast correlation technique for glacier flow monitoring by digital camera and spaceborne SAR images," *EURASIP J. Image and Video Processing*, vol. 2011, pp. 11, 2011.

[2] Y. Yan, G. Mauris, E. Trouvé, and V. Pinel, "Fuzzy uncertainty representations of coseismic displacement measurements issued from SAR imagery," *IEEE Trans. Instrumentation and Measurement*, vol. 61, no. 5, pp. 1278–1286, 2012.

[3] M.-P. Doin, F. Lodge, S. Guillaso, R. Jolivet, C. Lasserre, G. Ducret, R. Grandin, E. Pathier, and V. Pinel, "Presentation of the small baseline NSBAS processing chain on a case example: the Etna deformation monitoring from 2003 to 2010 using ENVISAT data," in *Fringe 2011 Workshop*, Frascati, Italy, September 2012, , ESA SP-697.

[4] SM Buckley, PA Rossen, and Patricia Persaud, "Roi_pac documentation-repeat orbit interferometry package," *JET Propulsion Lab., Pasadena, CA*, 2000.