Grid-enabled workflows for data intensive medical applications

Tristan Glatard, Johan Montagnat, Xavier Pennec

Grid-enabled workflows for data intensive medical applications*

Tristan Glatard, Johan Montagnat, Xavier Pennec

Abstract

Data intensive medical image processing applications can easily benefit from grid capabilities. However, the setting up of complex medical experiments is not straight forward on current grid infrastructures. To ease such experiments we are developing a generic and grid-enabled workflow framework, relying on current standards. We show results on a concrete application to medical image registration assessment. We discuss the limitations induced by current standards and tools and how they were overcome for deploying the application.

1 Context and objectives

Computerized medical image analysis is now a well established area that provides assistance for diagnosis, modeling, and pathologies follow-up. With the growing inspection capabilities of imagers and the increase in medical data production, the need for large amounts of data storage and computing power increases. Grids have been identified as a tool suitable for dealing with medical data [9]. Successful example of grid application deployment for image databases analysis [10], optimization of medical image algorithms, simulation [2], etc, have been reported.

Thanks to emerging standards, grids are easing the usage of computerized medical image analysis tools for a large community of end users, not necessarily aware of computer technologies. These standards enable data distribution and sharing as well as access to algorithms needed both for assessing medical image processing algorithms, and for building large scale health-related experiments.

1.1 Medical imaging workflows

Medical image analysis procedures are often not based on a single image processing algorithm but rather assembled from a set of basic tools dedicated to process the data, model it, extract quantitative information, and analyze results. Given that interoperable algorithms packed in software components with a standardized interface enabling data exchanges are provided, it is possible to build complex workflows to represent such procedures for data analysis. High level tools for expressing and handling the computation flow are therefore expected to ease medical experiments development.

Workflow processing is a thoroughly researched area. Most workflow managers are based on a central director orchestrating a set of actors. In the Kepler system [1] the generic actors can wrap Web Services (WS) or OGSA based services. In the Taverna project [12] the actors can be local java processes, WS or home made services. Other workflow systems such as Triana [13] are decentralized and distribute several control units over different computing resources. Some

*This work is partially funded by the French research program "ACI-Masse de données", http://acimd.labri.fr/

workflow systems such as ICENI [6] implement different scheduling strategies. Systems such as the Virtual Data System (VDS) decouple the abstract description of the workflow (Chimera system [5]), the mapping of the workflow and data to available resources (Pegasus system [3]) and the execution orchestrator.

When dealing with medical experiments, the user often needs to process datasets made of *e.g.* hundreds of individual images. The workflow management is therefore data driven and the scheduler responsible for sharing the loads of computation should take into account the input dataset as well as the workflow graph topology. We are studying workflow processing of medical datasets on grids. We are mostly interested in workflow management systems that are compliant to recent grid standard (and especially WS which will largely be involved in the emerging WSRF specification), interoperable, and able to deal with large datasets. Our work is based on the Taverna workflow manager [12] that nicely decouples the different workflow components, provides an helpful GUI for workflows description, can orchestrate standard WS actors, and provides basic mechanisms to deal with sets of input data. Furthermore, Taverna is an open source project and it is possible to get deep insight and potentially modify the software. It proved to be simple to deploy on standard PCs.

1.2 Application to image registration assessment

Many data intensive and complex medical applications could benefit from grid enabled workflows. In this paper, we are studying how classical workflow managers can adapt to a grid infrastructure for a concrete compute intensive application which is the assessment of medical image registration algorithms.

In the absence of *ground truth* (which is usually the case in medical image processing), evaluation of the accuracy and robustness of image processing algorithms is very difficult [7]. A solution that has been recently proposed is to establish a *bronze standard* [11] by considering the "exact result" as an unknown variable that has to be estimated (along with its accuracy). This method is based on the registration of all possible pairs of images by many registration methods (different from the one to evaluate) in order to better exploit the redundancy of information. Given the cost of medical images registration (in the order of tens of minutes for each pair of 3D images), the bronze standard method is very compute intensive. In order to gridify it, the registration algorithms used are embedded in standard WS. The Taverna workflow manager is used to schedule the processes. Yet, we had to face many limitations induced by Taverna and its interaction with WS. Data parallelism is enabled through an ad-hoc technique enabling asynchronous execution of multiple instances of WS.

2 Bronze standard workflow gridification

We are using four different registration algorithms in our implementation of the bronze standard method: (1) **Baladin** and (2) **Yasmina** are intensity-based. The former uses a block matching strategy while the later optimizes a similarity measure on the complete images using the Powel algorithm. (3) **CrestMatch** is a prediction-verification method and (4) **PFRegister** is based on the ICP algorithm. Both CrestMatch and PFRegister register features (crest lines) extracted from the input images. These algorithms are further described in [11]. All produce a transformation and the bronze standard is computed from these results. Figure 1 illustrates the simplified application workflow. Each box in figure 1 represents an algorithm and arrows show computation dependencies. Lightweight computing tasks such as data format transformations have been omitted for clarification.

2.1 Prototyping the application

We prototyped our application on a local cluster of 4 machines, each of them hosting a particular registration algorithm among those described above. In order to mimic a grid architecture and to optimize data exchanges, we only used the Taverna workflow manager for the workflow execution control. Data exchanges were performed by the computing components themselves, through ssh tunnels. Thus, the information which was exchanged between the components and the workflow engine via SOAP messages was only containing *references* (*i.e.* file locations) to the input data of the components. No large data was included into those messages. Such an architecture is suitable for the gridification of data intensive application because it enables a rigorous centralized control of the execution without penalizing data transfers (data never goes via the central orchestrator).

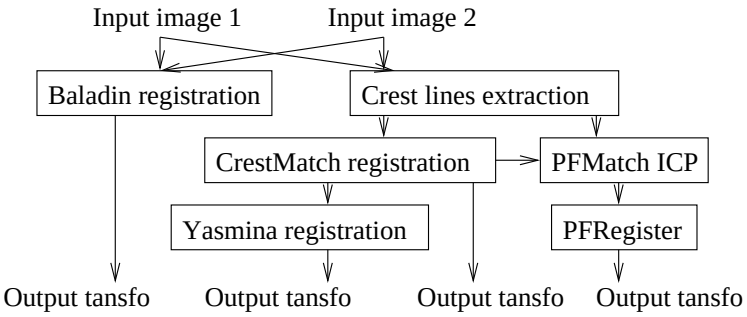

Figure 1. Bronze Standards application workflow

2.2 Grid enabling the workflow

Our application was deployed on the computer infrastructure provided by the EGEE European project [4]. The platform offered is a pool of thousands computing (standard PCs) and storage resources accessible through the LCG2 middleware [8]. The resources are assembled in computing centers, each of them running its internal batch scheduler. The EGEE middleware is mostly batch oriented and enables the submission of independent computing tasks on different resources through a command line interface.

Although today migrating to conform to emerging grid standards, the EGEE middleware is not yet compatible with the WS interface. Therefore, we have created dedicated WS able to start and monitor computing tasks on the grid infrastructure using the EGEE command line interface. The Taverna workflow manager can thus initiate these WS that in turn will launch grid computing tasks as illustrated in figure 2.

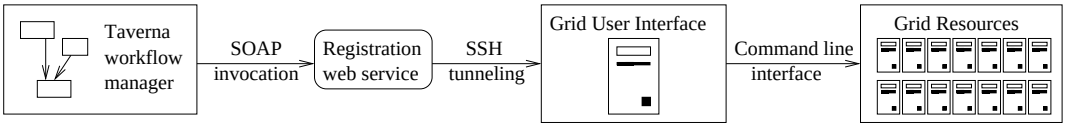

Figure 2. Interaction between Taverna and the EGEE infrastructure

There are three main limitations inherent to Taverna data management and its interaction with WS that made the gridification of the application workflow difficult. These are generic problems that will be encountered in any effort to gridify a data intensive application. The problems and the proposed solutions are described below.

2.3 Static nature of the interaction between Taverna and Web Services

Due to the blocking nature of its interaction with WS, Taverna only provides static task parallelism : in Taverna, a WS can only process one request at a time and will not respond to a second request until the first one is cleared and a result has been returned. Yet the latest version of Taverna is able to make a *statically fixed* number of query to a same WS to process as many identical tasks in parallel. This so called *thread mode* is however not satisfactory as (i) the number of parallel tasks is fixed at the workflow creation and will not adapt to the available resources on the grid nor to the amount of input data to be processed, (ii) Taverna still imposes a hard limitation to a maximum of 10 threads for WS, and (iii) it is up to each WS to adapt to the thread mode and to offer a service capable of dealing with multiple queries.

To overcome this limitation, we had to fake an asynchronous behavior not foreseen in the standard WS invocation framework of Taverna. The idea is to divide the service dedicated to grid job processing into two independent services: a submission service and a fetching service. The asynchronous nature of the grid middleware enables this possibility. When queried, the submission service sends a task to be processed on the grid, forks a monitoring process, and returns immediately although the computation is not terminated yet. When queried, the fetching service does return as soon as *any one of* the tasks initiated by the submission service terminates. In fact, the monitoring processes that have been forked for each submission do notify the fetching service as soon as the computing task they monitor is terminated. Figure 3 is a time diagram showing how the classical framework for job submission (left: Taverna is blocked until the query returns) is transformed into our fake asynchronous framework (right: several queries can be processed in parallel although the submission and fetching are standard blocking services. Note that this framework does not ensure that the first result returned corresponds to the first computation submitted.

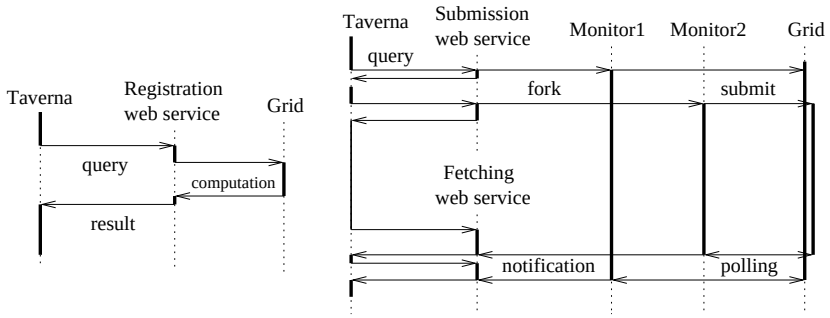

Figure 3. Classical (left) and asynchronous (right) framework based on WS

The stateless nature of WS was also a problem in implementing the asynchronous behavior: the fetching WS is invoked by Taverna for retrieving each result submitted and given the stateless nature of this service, it does not hold memory of the number of times it was invoked neither which results have already been returned. A file based state holding has been implemented to ensure proper operation. In the future, the WSRF specification is supposed to introduce stateful WS.

2.4 Data management in an asynchronous workflow

Another problem we faced was the ordered processing of data in workflows running in an asynchronous way. As we have shown above, a data is able to overtake another one during the execution

of the workflow. This raises a causality problem: consider the sub-workflow made by the CrestLine (CL), CrestMatch (CM), and PFMatchICP (PFM) in figure 1. Let D_0 and D_1 be two sets of input data and CL(D_0) be the result of the processing of the dataset D_0 by the CrestLine algorithm. If the order of computations in data is inverted between CL and CM, and in the absence of a data-aware flow control, the final output would be the computation of PFM(CM(CL(D_1)), CL(D_0)) and PFM(CM(CL(D_0)), CL(D_1)) while PFM(CM(CL(D_i)), CL(D_i)), $i \in \{0, 1\}$ was needed. To overcome this problem, we have linearized this sub-part of the workflow: the CrestLines are extracted first, then both the input data and the processing result are sent to the CrestMatch algorithm, and finally, the PFMatchICP algorithm is executed. This solves this particular problem but (i) this solution is clearly inefficient as this linearization of the workflow breaks any possible parallelism, and (ii) this is not a general solution that can adapt to any kind of workflow. This problem can only be properly tackled at the workflow level.

2.5 Data dependency

Another limitation induced by Taverna is its execution strategy : when multiple data are sent to a workflow, all data are processed by the first task before a second (dependent) task can start. Therefore, if data D_0 and D_1 are to be sent to task T_0 and if T_1 depends on T_0, the execution of T_1 will only start once both $T_0(D_0)$ and $T_0(D_1)$ have been processed although there is no dependency between data and one could perfectly compute $T_1(T_0(D_0))$ as soon as $T_0(D_0)$ is available, independently of the processing of $T_0(D_1)$. This limitation could not be overcome and it does alter the total computation time on a grid infrastructure. In particular, if ever a job is lost due to a system failure, the whole pipeline is blocked.

3 Experiments and results

We made experiments on a dataset of 5 CT-scan images of a phantom of the abdomen (Body Form ®, Limbs & Things Ltd, Bristol, UK). The images were acquired on a helical Siemens Somatom 4 plus CT-scan. They have between 501 and 571 slices of 512×512 pixels, with a spacing between slice of 1mm and a pixel size ranging from 0.78mm to 0.92mm. The phantom was acquired in 5 different positions, with different imaging parameters, in order to simulate the influence of most possible acquisition biases.

For each of the 25 pairs of images, we considered 2 different similarity measurements for Yasmina, 2 sets of different options for PFMatchICP and 3 sets of different options for Baladin. Table 1 summarizes the computing time for each of the algorithms locally and on the grid. There is a significant speed-up in grid execution. The computing time on a local machine for all registration exceeded 1 day (24 hours 25 min) whereas the same computation on the EGEE infrastructure only took 4h07, thus yielding to a 5.93 speed-up factor. These figures are dependent on the grid load and may slightly vary in different executions. However, the EGEE infrastructure is a production grid which is permanently under stress and therefore no drastic changes can be expected.

4 Conclusions

We presented a complete deployment of a data intensive application on the EGEE grid infrastructure. It encompasses image registration algorithms wrapped in standard Web-Services, a grid

Algorithm	Local execution	Grid execution	Speed-up
CrestLine (25 crest line extractions)	6h15	49 min	7.65
Baladin (75 registrations)	24h25	4h07	5.93
CrestMatch (25 registrations)	5h29	24min	13.7
PFMatchICP (50 registrations)	2h42	37 min	4.38
PFRegister (50 registrations)	7min	23min	0.3
Yasmina (50 registrations)	16h11	1h41	9.6
Total execution time	24h25	4h07	5.93

Table 1. Computation times, localy and on the EGEE infrastructure

enabled workflow manager and a grid middleware for performing the computations. The framework developed could easily be adapted to a wide variety of medical applications.

The results have shown the interest of such an approach in the case of a concrete data intensive medical application. However, we have shown that the setting up of this experiment was complicated by the static limitation of Taverna data management and the blocking nature of its interaction with Web-Services. It shows that there is still room for development of grid aware workflow managers. A most worrying concern is the limitation induced by the stateless nature of Web Services while this standard is plebiscited by the international grid community. The WSRF specification should enable stateful services though.

References

[1] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludaescher, and S. Mock. Kepler : Towards a grid-enabled system for scientific workflows. In *GGF10*, March 2004.

[2] H. Benoit-Catttin, F. Bellet, J. Montagnat, and C. Odet. Magnetic Resonance Imaging (MRI) Simulation on a Grid Computing Architecture. In *Biogrid (CCGrid 03)*, Tokyo, Japan, May 2003.

[3] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, and G. Mehta et al. Mapping abstract complex workflows onto grid environments. *Jnl of Grid Comp.*, 1(1):9 – 23, 2003.

[4] EGEE: Enabling Grids for E-sciencE, IST European project. http://www.eu-egee.org/.

[5] I. Foster, J. Voeckler, M. Wilde, and Y. Zhao. Chimera: A virtual data system for representing, querying and automating data derivation. In *Sc. and Stat. DB Managmt*, Edinburgh, Scotland, 2002.

[6] N. Furmento, A. Mayer, S. McGough, S. Newhouse, T. Field, and J. Darlington. Iceni : Optimisation of component applications within a grid environment. *Jnl of Parall. Comp.*, 28(12):1753 – 1772, 2002.

[7] P. Jannin, J.M. Fitzpatrick, D.J. Hawkes, X. Pennec, R. Shahidi, and M.W. Vannier. Validation of medical image processing in image-guided therapy. *IEEE Trans. on Med. Imaging*, 21(12):1445–1449, 2002.

[8] LCG middleware. http://lcg-web.cern.ch/.

[9] J. Montagnat, F. Bellet, H. Benoit-Cattin, V. Breton, L. Brunie, H. Duque, Y. Legré, I.E. Magnin, L. Maigne, S. Miguet, J.-M. Pierson, L. Seitz, and T. Tweed. Medical images simulation, storage, and processing on the european datagrid testbed. *to appear in Jnl of Grid Comp.*, February 2005.

[10] J. Montagnat, V. Breton, and I.E. Magnin. Partitionning medical image databases for content-based queries on a grid. *to appear in Methods of Information in Medicine*, 45, February 2005.

[11] S. Nicolau, X. Pennec, L. Soler, and N. Ayache. Evaluation of a new 3d/2d registration criterion for liver radio-frequencies guided by augmented reality. In *Intl. Symp. on Surgery Sim. and Soft Tissue Model.*, pages 270–283, Juan-les-Pins, France, 2003.

[12] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Greenwood, T. Carver, A. Wipat, and P. Li. Taverna : A tool for the composition and enactment of bioinformatics workflows. *Bioinformatics journal*, 2004. http://taverna.sourceforge.net/.

[13] I. Taylor, M. Shields, I. Wang, and R. Philp. Grid enabling applications using triana. In *Grid App. and Pro. Tools, GGF8*, Seattle, USA, June 2003.