Grid Computing Application for Brain Magnetic Resonance Image Processing

This article has been downloaded from IOPscience. Please scroll down to see the full text article.

2012 J. Phys.: Conf. Ser. 341 012011

(http://iopscience.iop.org/1742-6596/341/1/012011)

View the table of contents for this issue, or go to the journal homepage for more

Download details:
IP Address: 184.145.156.107
The article was downloaded on 09/02/2012 at 19:46

Please note that terms and conditions apply.
Grid Computing Application for Brain Magnetic Resonance Image Processing

F Valdivia 1, B Crépeault 1, and S Duchesne 1,2

1 Centre de Recherche de l’Université Laval Robert-Giffard, 2601 de la Canardière, Québec (QC) G1J 2G3, Canada
2 Département de radiologie, Université Laval, Québec (QC) G1K 7P4, Canada

E-mail: fernando.valdivia@crulrg.ulaval.ca, burt.crepeault@gmail.com, simon.duchesne@crulrg.ulaval.ca

Abstract: This work emphasizes the use of grid computing and web technology for automatic post-processing of brain magnetic resonance images (MRI) in the context of neuropsychiatric (Alzheimer’s disease) research. Post-acquisition image processing is achieved through the interconnection of several individual processes into pipelines. Each process has input and output data ports, options and execution parameters, and performs single tasks such as: a) extracting individual image attributes (e.g. dimensions, orientation, center of mass), b) performing image transformations (e.g. scaling, rotation, skewing, intensity standardization, linear and non-linear registration), c) performing image statistical analyses, and d) producing the necessary quality control images and/or files for user review. The pipelines are built to perform specific sequences of tasks on the alphanumerical data and MRIs contained in our database.

The web application is coded in PHP and allows the creation of scripts to create, store and execute pipelines and their instances either on our local cluster or on high-performance computing platforms. To run an instance on an external cluster, the web application opens a communication tunnel through which it copies the necessary files, submits the execution commands and collects the results.

We present result on system tests for the processing of a set of 821 brain MRIs from the Alzheimer’s Disease Neuroimaging Initiative study via a nonlinear registration pipeline composed of 10 processes. Our results show successful execution on both local and external clusters, and a 4-fold increase in performance if using the external cluster. However, the latter’s performance does not scale linearly as queue waiting times and execution overhead increase with the number of tasks to be executed.
1. Introduction
The primary objective of the Medical Data, Information and Knowledge (MEDICS) laboratory is to find early signs of neuropsychiatric diseases through novel techniques for medical image analysis. Early applications of the research lie in the establishment of biomarkers for Alzheimer’s disease (AD) based on population analyses of voxel-based features [1], high-dimensional morphometry characteristics [2, 3] and structural volumetry [4, 5], all extracted from magnetic resonance images (MRI) of human subjects.

To achieve the necessary statistical power in the conduct of this research, we collaborate with various research organizations to collect wide and comprehensive patient and image databases. Examples include research consortia (e.g. the Alzheimer’s Disease Neuroimaging Initiative (ADNI)[6]; European ADNI [7], International Consortium for Brain Mapping [8] or individual laboratories, either through open (e.g. Open Access Series of Imaging Studies [9]) or close-source access (e.g. Laboratory of Epidemiology, Neuroimaging, & Telemedicine [10]).

However, irrespective of the many problems related to processing multicentric images per se, this pooling of data from multiple sources presents a number of difficulties, hindering the timely conduct of research. Those relate to:

(1) formatting: patient and image data formats vary widely from one organization to another, making it difficult to perform similar analyses on different sets of data;
(2) searching: it is difficult, sometimes impossible, to comprehensively sift through large, incohesive data stacks based on specific patient criteria or image characteristics;
(3) cross-referencing: moreover, most dataset are decoupled, e.g. patient data tables are typically disconnected from the image files and must be reassembled; and
(4) processing: data (e.g. images) are typically not processed, and require dedicated application suites to extract relevant biomarker information.

2. Objective
In this article we report the design, development and use statistics of the integrated MEDICS database and processing platform, built to address these issues by complying with the following requirements:

(1) ability to accept different input formats, and to transpose to a common format for processing;
(2) ability to house massive amounts of data, in image form or otherwise in relational format;
(3) capacity for remote, web-access, search and processing;
(4) capacity for flexible data processing, with complete traceability; and
(5) ability to use grid computing to accelerate the processing of large-scale datasets.

3. Methods
3.1. Software Architecture
The MEDICS system is composed of three major software modules (Figure 1):

**Database - Module B.** The alphanumeric and image database is a relational database system, able to accept all prevalent medical image formats, starting with DICOM [11] and other reconstructed formats which are eventually all converted and stored as MINC\(^1\) images. Any number of scalar or text files can be accepted for alphanumeric data (e.g. neuropsychological testing). These files are accessed directly

\(^1\)Medical Image NetCDF format; http://www.bic.mni.mcgill.ca/ServicesSoftware/MINC.
through the User Interface or the Pipeline Processing modules (see below). The data has been organized in a semi-formal domain-specific ontology [12] assembled from various ontological sources (e.g. anatomical [13-15]); medical image data (DICOM reference guide 2001); data processing [16, 17]; pathological [18, 19] and controlled terminologies (e.g. quality control [20]; validation procedure and inter-relationships [21, 22]).

**Pipeline Post-Processing - Module EX.** The Post-Processing module allows the definition of processing pipelines (i.e. individual processes assembly into serial/parallel execution streams) and accepts user requests through the web-compatible User Interface module for their execution (EX). Module EX scripts manage concurrent pipeline executions, event logging and job submission to local or external computer clusters, using a queue management system as distributed resource management.

**User Interface – Module WX.** The Web-eXchange (WX) User Interface represents the interface between the user of the MEDICS system and the system itself. Its main functions are: user creation and authentication, data search, dataset creation, pipeline creation and execution, image quality control and rating [1].

---

**Figure 1.** MEDICS System Software Architecture [2].

### 3.2. Hardware Architecture

The MEDICS system rests on the following major hardware components (Figure 2):
**User Workstation**: Users can access the MEDICS system via any computer through an Internet connection and standard web browsers.

**WX server**: The server is a computer running web server software. The server will stand inside a demilitarized zone in order to be accessible through HTTPS.

**EX and B server**: The EX server is a computer running the MEDICS Pipeline Post-Processing EX module as well as grid control software when present; the B server houses the relational database management system. The B and EX servers stands behind a firewall in the secure zone. At present, we use the same physical server as both EX and B; they can be decoupled in the future to meet growing demands without loss of functionality.

**Local and External Grid**: both local MEDICS Laboratory computer grid or external high-performance clusters can be accessed to distribute jobs and resource management for the execution of processing pipelines.

**Figure 2. MEDICS System Hardware Architecture [1].**

### 3.2.1. Architectural Decisions

| Component                  | Products                                    |
|----------------------------|---------------------------------------------|
| Relational database        | PostgreSQL                                  |
| Image storage and files    | Network File System (NFS), format MINC image and text files |
3.3. Web User Interface
All instructions to the system are made through one of the WX functions (applications) that include [1]:

- User authentication, where a user enters credentials to access the system;
- Data search, where users can sift through data by entering ad-hoc queries to filter information from database;
- Dataset creation, where users can assemble search results and create datasets based on their search criteria;
- Pipeline creation, where users can assemble series of processes together that will perform a certain image-processing task;
- Pipeline execution, where users can instantiate pipelines with datasets, give values to certain process options (when applicable) and launch the execution; and
- Visual image quality control and rating, where users assign scores based on a scale to medical images produced by a processing pipeline.

3.4. Data processing

3.4.1. Processes
Processes are scriptable elements that perform concrete actions on data:

- Manipulate data arrays within the application;
- Perform data transformations on scalars and images;
- Perform statistical comparisons on scalars and images;
- Perform quality control of input data (e.g. scalars, files, images), of other processes, of whole pipelines; and
- Perform quality assurance of output data (e.g. scalars, files, images) from other processes, or whole pipelines.

Each process has distinct characteristics and options (mandatory and optional), recorded in the alphanumeric database. Processes can be modelled in a fashion represented in Figure 3, whereby each process has any number of imports and outputs of any type, as well as a number of options. Options can

---

2 The MEDICS system is compatible with any scriptable image processing library
3 FMRIB Software Library ; http://www.fmrib.ox.ac.uk/fsl/
4 Statistical Parametric Mapping ; http://www.fil.ion.ucl.ac.uk/spm/
be viewed as the arguments of a process, e.g. the various commands that can make a process execute in a certain way. The execution of a process will produce a return value that indicates if it completed successfully or not.

Ports can be defined as the entry or exit points for data entering or leaving processes (and by extension, pipelines). In our model so far we have defined entities that correspond to the port types depicted in Figure 3, along with their graphical notation.

---

**Figure 3.** Process Model and Port Types [1].

---

**Figure 4.** Pipeline Model [1].
3.4.2. Pipelines

Data processing is performed as a set of processes running within a pipeline [1] (Figure 4). Pipelines are collections of processes, each one performing a single task, that are assembled to perform a series of scalar and/or image processing steps that lead to the desired result(s).

In our model, we can observe that the pipeline has ports, representing the means by which data is fed into the pipeline for processing (inports) or produced as a result of the pipeline execution (outports). Inside the pipeline are chained processes that also have inports and outports that are linked to one another according to their type, graphically represented with different symbols. Outports of a process can only be linked to the inports of the following process if their types are the same [1].

The creation and execution of pipelines requires a set of parameters, stored in the database, which include: (i) mandatory and optional individual process options; (ii) mandatory and optional input and output individual process ports; (iii) individual process source/versions to execute; (iv) individual estimated execution time per process; (v) number of processes to execute in parallel per node; (vi) general execution cluster data (type local/remote, home path, user, max job in queue, commands, etc.).

3.4.3. Control scripts

A set of PHP (Hypertext Pre-processor) and BASh (Bourne Again Shell) scripts are responsible for monitoring execution of pipeline instances, build process execution commands, submit jobs to the queue, retrieve results, and copy files local to remote and remote to local, if necessary; all the while running from behind the web application. The three main scripts are: (i) **Wrapper**: Prepares the data and builds the basic commands with process execution options; (ii) **Ex-monitor**: Manages pipeline instances execution, builds final commands, opens communication tunnel to external/local cluster, copies files, submits jobs, retrieves results and creates log files; and (iii) **Launch command**: Interacts with Ex-monitor to invoke the execution of the processes in clusters.
4. Experiments and results

4.1. Databases

The alphanumeric database is relational and implemented in PostgreSQL. The database stores a number of scalar-type data on patients, such as demographics, family and past medical history; neurological, neuropsychiatric and neuropsychological test results; genotyping information; biomarkers; and results from 137 tables. Further, a number of original and processed MRI are stored and linked to database records.

| Table 2. Summary records of the database. |
|-------------------------------------------|
| **Origin** | **Quantity** | **Patients** | **Original Images** |
| ADNI  | 822 | 15 799 |
| E-ADNI | 3 | 79 |
| ICBM  | 150 | 300 |
| LENITEM | 1 038 | 1 698 |
| OASIS | 416 | 1 608 |
| MEDICS | 19 | 68 |
| **TOTAL** | **2 448** | **19 552** |

4.2. High-performance clusters

At present the MEDICS system can interface with two clusters: a local cluster (“MEDICS”), composed of all individual workstations and servers within the laboratory; and the external high-performance computer cluster Colossus, from the Consortium Laval UQAM McGill and Eastern Quebec⁵ (“CLUMEQ”). Basic features for each cluster are shown in Table 3.

| Table 3. MEDICS and CLUMEQ basic features |
|-------------------------------------------|
| **Description** | **MEDICS** | **CLUMEQ** |
| Processors | 32 (10 workstations (WS)) | 7664 |
| Core types | Dual core 2.8 Ghz – 3.06 Ghz | Core Intel Nehalem 2.8 Ghz. |
| Memory RAM | 2WS with 2GB, 6WS with 4GB and 2WS with 8 GB | 1/3 node 12GB and 2/3 nodes 24GB |
| Jobs in execution or in queue | Limited by hardware | 200 |
| Operational systems | Ubuntu release 9.10 | CentOS release 5.4 |
| Distributed resource management | Sun Grid Engine 6.2 | Sun Grid Engine |

⁵ http://www.clumeq.mcgill.ca/
### 4.3. Execution Time Analysis

For testing purposes we generated a single pipeline focused on spatial registration of image. The purpose of this pipeline was the computation of a nonlinear deformation grid exactly mapping one volume to another. At the core of this pipeline is the **MINC Tracc** process, which estimates the spatial transformation. Special attention is given to this process and this pipeline because it is one of the most expensive to run in terms of time and required resources.

Table 4 compares the execution times for different processes in the MEDICS and CLUMEQ clusters for a small dataset of 12 images. Table 5 compares the execution times for the whole pipeline on a set of 821 images taken from the ADNI database.

We tested every execution parameter on both clusters and observed that all specifications were maintained regardless of its nature [23, 24]. Regarding execution times, results show that **CLUMEQ** outperforms **MEDICS** in both experiments. This is unsurprising, given obvious hardware differences between the two clusters. First, while **MEDICS** normally executes one job for every workstation, we were able to assign eight jobs per node in **CLUMEQ**. Secondly, there existed an execution bottleneck in **MEDICS** within a single workstation with limited RAM (2GB), which severely limited its ability to handle large deformation volumes.

Interestingly however, we notice that the performance increase between the two clusters falls from a factor of 8.1 (small dataset) to 4.1 (large dataset). This diminution is due to the increase in queue waiting time at **CLUMEQ**, and the delays associated with overhead communications between our in-house EX server and the grid comptroller at **CLUMEQ**.

| Process          | Execution time by Cluster | Speed-Up |
|------------------|---------------------------|----------|
|                  | MEDICS (1 job per node)   | CLUMEQ (8 job per node) |         |
| **MINC Blur 8 mm** | 11 sec.                   | 6 sec.               | 14.66   |
| **MINC Blur 4 mm** | 15 sec.                   | 6 sec.               | 20      |
| **MINC Blur 2 mm** | 16 sec.                   | 7 sec.               | 18.24   |
| **MINC Tracc 8 mm** | 4 min.                    | 3 min.               | 10.64   |
| **MINC Tracc 4 mm** | 14 min. (8GB RAM)         | 13 min.              | 4.89    |
|                  | 16 min. (4GB)             |                     | 10.38   |
|                  | 17 min. (2GB)             |                     | 11.08   |
| **MINC Tracc 2 mm** | 16.5 min. (8GB)          | 14 min. ^b           | 2.35    |
|                  | 18 min. (4GB)             |                     | 5.16    |
|                  | 6 hours 6 min. (2GB)      |                     | 104.57  |
| **MINC Resample** | 4 min.                    | 4 min.               | 8       |
| **MINC Blob**    | 38 sec.                   | 21 sec.              | 14.47   |
| **MINC Resample** | 1 min.                    | 19 sec.              | 28.21   |
Table 5. Execution times for the Nonlinear Registration pipeline by cluster for a dataset of 821 images.

| Cluster   | Total execution time |
|-----------|----------------------|
| MEDICS    | 3d: 5h: 23m: 16s     |
| CLUMEQ    | 0d: 19h: 0m: 58s     |

\( \text{a} \) d=days, h=hours, m=minutes and s=seconds.

5. Discussion
We report our work towards the conceptualization, design and implementation of a web application that allows: a) construction of pipelines dynamically from a set of processes; b) interaction with an alphanumeric database and file system; and c) execution of high performance computing on different clusters. The parallel programming and parameterization of options in the execution of processes greatly improves application performance, but increases the complexity of the application.

Grid computing is shown to reduce significantly the execution time of post-processing of brain magnetic resonance images, even though the effect is mitigated by queue waiting times and the necessity for individual process control. Quite possibly, for applications such as ours, there exists a niche for a dedicated micro-cluster (e.g. 200 cores total capacity) that would result in optimal throughput, midway between low-cost ad hoc assemblies of workstations and high-cost high-performance supercomputers.

Our application was built to answer the specific needs of our laboratory, but compares in principle to the cluster-based CBRAIN system (with LORIS database)(see the AddNeuroMed study for an applicative example [25]) or the LONI pipelining and execution system [26]. Both of the latter systems however were designed to be either exported or support a large community of users.

6. Acknowledgments
We thank the Ministère du Développement Économique, de l’Innovation et de l’Exportation du Québec and the National Science and Engineering Research Council for financial support towards the creation of the processing system and the MEDICS hardware architecture ; the Fonds de Recherche en Santé du Québec (Réseau de Bio-Imagerie du Québec) for funding towards the MEDICS database ; Dr. G. B. Frisoni from the IRCCS Fatebenefratelli (Brescia, Italy) for access to LENITEM and E-ADNI data ; the ADNI, OASIS, and ICBM consortia for similar reasons ; and finally CLUMEQ for access to its Université Laval-based infrastructure.
7. References

1. Canu, E., et al., \textit{HOXA1 A218G polymorphism is associated with smaller cerebellar volume in healthy humans.} J Neuroimaging, 2009. 19(4): p. 353-8.
2. Duchesne, S., et al., \textit{Amnestic MCI future clinical status prediction using baseline MRI features.} Neurobiology of Aging, 2010. 31(9): p. 1606-17.
3. Duchesne, S., et al., \textit{MRI-Based Automated Computer Classification of Probable AD Versus Normal Controls.} IEEE Trans Med Imaging, 2008. 27(4): p. 509-20.
4. Mouiha, A. and S. Duchesne, \textit{Multi-decade hippocampal and amygdala volume analysis: Equal variability and limited age effect.} Neuroscience Letters, 2011. 499(2): p. 93-8.
5. Mouiha, A. and S. Duchesne, \textit{Hippocampal atrophy rates in Alzheimer's disease: automated segmentation variability analysis.} Neuroscience Letters, 2011. 495(1): p. 56-66.
6. Frisoni, G.B., et al., \textit{The pilot European Alzheimer's Disease Neuroimaging Initiative (ADNI).} Alzheimers Dement, 2005. 1(1): p. 1-10.
7. Temal, L., et al., \textit{Towards an ontology for sharing medical images and regions of interest in neuroimaging.} J Biomed Inform, 2009. 42(5): p. 632-40.
8. Aubry, F. and A. Todd-Pokropek, \textit{Mimos: a description framework for exchanging medical image processing results.} Stud Health Technol Inform, 2001. 84(Pt 2): p. 891-5.
9. Ciccarese, P., et al., \textit{The SWAN biomedical discourse ontology.} J Biomed Inform, 2008. 41(6): p. 862-71.
10. Clark, T. and J. Kinoshita, \textit{Alzforum and SWAN: the present and future of scientific web communities.} Brief Bioinform, 2007. 8(3): p. 163-71.
11. Ihalainen, T., O. Sipila, and S. Savolainen, \textit{MRI quality control: six imagers studied using eleven unified image quality parameters.} Eur Radiol, 2004. 14(10): p. 1859-65.
12. Jannin, P., et al., \textit{Validation of medical image processing in image-guided therapy.} IEEE Trans Med Imaging, 2002. 21(12): p. 1445-9.
13. Jannin, P., C. Grova, and C.R. Maurer, \textit{Model for designing and reporting reference based validation procedures in medical image processing.} IJCARS, 2006. 1(2): p. 1001-1115.
14. Crepeault, B., \textit{ADNI+ICBM – ABC Pipeline Verification Procedure, 2009, Centre de recherche Université Laval - Robert Giffard: Québec City, Canada.} p. 1-105.
24. Crepeault, B., *MEDICS Image processing system - Verification procedure*, 2009, Centre de recherche Université Laval - Robert Giffard: Quebec City, Canada. p. 1-40.
25. Simmons, A., et al., *The AddNeuroMed framework for multi-centre MRI assessment of Alzheimer's disease: experience from the first 24 months*. Int J Geriatr Psychiatry. 26(1): p. 75-82.
26. Dinov, I.D., et al., *Efficient, Distributed and Interactive Neuroimaging Data Analysis Using the LONI Pipeline*. Frontiers in neuroinformatics, 2009. 3: p. 22.