INFORMATICS FOR HIGH-THROUGHPUT AND DISTRIBUTED ANALYSIS OF MEDICAL IMAGES

By

Kelsie Jade Covington

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Computer Science

May, 2012

Nashville, Tennessee

Approved:

Professor Bennett A. Landman

Professor Aniruddha S. Gokhale

ACKNOWLEDGEMENTS

This work would not have been possible without the financial support of grants 1R03EB012461 and N01-AG-4-0012.

I am grateful to all of those with whom I have had the pleasure to work during this and other related projects. Those members of the Vanderbilt EECS faculty with whom I have had to the pleasure of learning have been critical to the development of my research interests and goals, whether it be through lectures, projects, or personal guidance. I would especially like to thank my advisor, Dr. Bennett Landman, without whose guidance I would not be publishing this thesis. He has directed me to fascinating and innovative work, and has set an example for the potential for what a good scientist and leader can be.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Analysis of MR data typically involves interleaving numerous toolsets into a processing workflow. However, these toolsets may not be compatible, may not support all desirable file formats, or may be limited in their ability to interface with multi-processing computing environments or distributed resource management (DRM) systems. These limitations impede the ability of researchers to automatically process large data sets using state-of-the-art, computationally complex algorithms and associated workflows, thus inhibiting both the standardization of these techniques as well as the advancement of our understanding of MR data that will lead to improved processing techniques. The Java Image Science Toolkit (JIST)[1], released in 2010, addresses these issues by ensuring interoperability[2], generating graphical user interfaces[1], providing advanced batch processing tools, and, most importantly, requiring minimal additional programming or computational overhead.

Since its release, we have gained appreciation for the limitations of JIST and of the current state of MR analysis toolsets as a whole. In response, we recognize several opportunities to make JIST a more efficient, interoperable, and intuitive framework for large scale MR analysis. Integrating algorithms from alternate imaging frameworks fosters collaboration and innovation amongst researchers[2]; generalized grid access and efficient resource requirement prediction realizes large scale batch operations that would otherwise be infeasible. Our implementation of these goals has

[1] B. Lucas, J. Bogovic, A. Carass et al., “The Java Image Science Toolkit (JIST) for rapid prototyping and publishing of neuroimaging software,” Neuroinformatics, 8(1) 5-17 (2010).
[2] K. Covington, E. S. McCreedy, M. Chen et al., "Interfaces and Integration of Medical Image Analysis Frameworks: Challenges and Opportunities," Biomedical Science and Engineering Conference. Oak Ridge, TN, 1-4 (2010)

significantly improved JIST as a viable tool for large scale image processing. Figure 1 describes the JIST architecture and our contributions.

In this thesis we describe two major extensions to the JIST functionality within Pipeline Layout Tool and the Process Manager. The Pipeline Layout Tool (Figure 2) provides a graphical interface to search for, arrange, and customize the input and output parameters for JIST pipeline modules. The resulting layout can then be loaded into the Process Manager (Figure 3), which allows real-time management of and feedback from JIST jobs, execution of dependent jobs in the correct order, and execution of batch processes in parallel. In our enhancements to the JIST framework, we improve upon JIST's strengths, and work towards alleviating or eliminating many of limitations in the large scale image processing domain, allowing researchers to focus on large scale analysis and innovation rather than on volatile, low-level software interactions. Chapter II illustrates the challenges and opportunities for grid computing in large-scale MR analysis, describes the limitations of generalized grid interfaces and our solution to this constraint. Chapter III introduces a novel system for distributed estimation of run-time and memory resource constraints for individual jobs in a JIST workflow. Chapter IV describes a proof-of-concept implementation that adapts our grid access to apply it to the Phillips MRI scanner. Chapter V discusses these technologies and addresses avenues of future research.

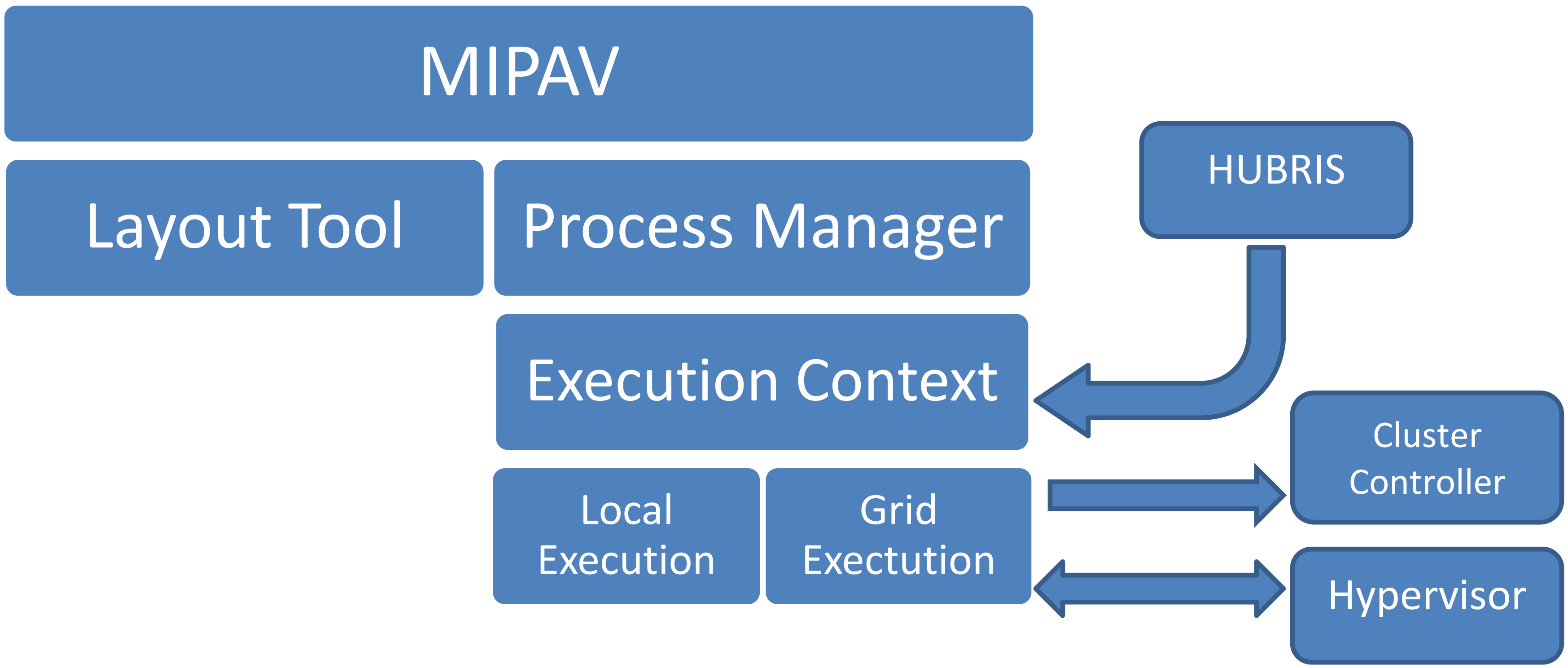

Figure 1: The JIST architecture and its interactions with our three contributions to the JIST workflow.

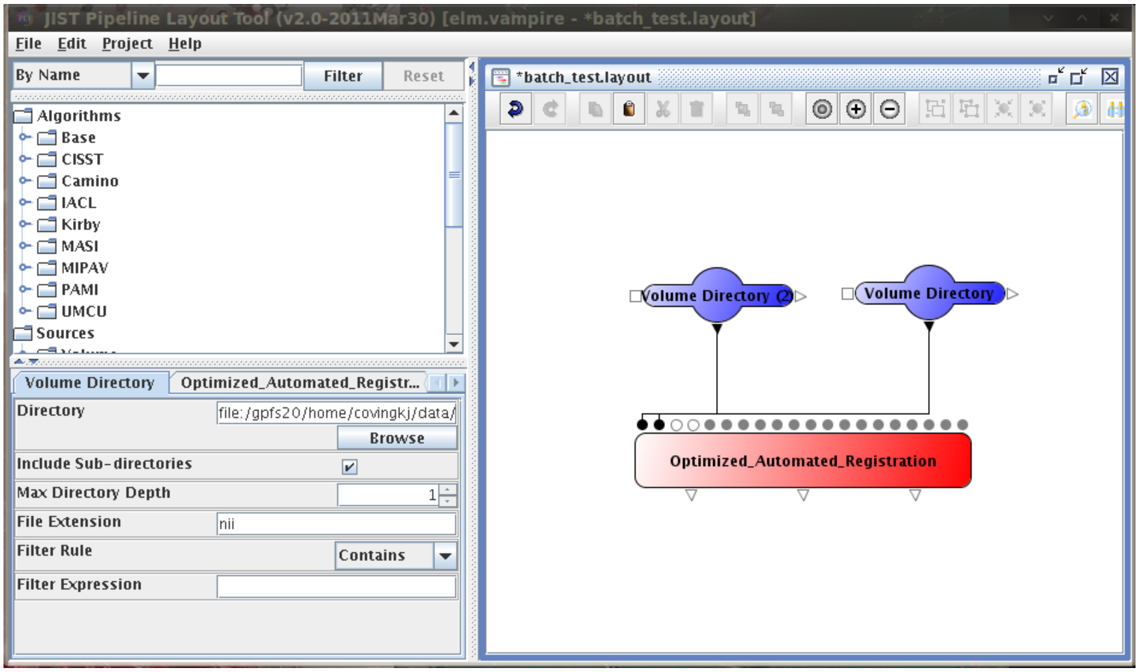

Figure 1: The Pipeline Layout Tool. Algorithms from the menu can be dragged and dropped into the layout. Input and output parameters for each selected algorithm are populated here.

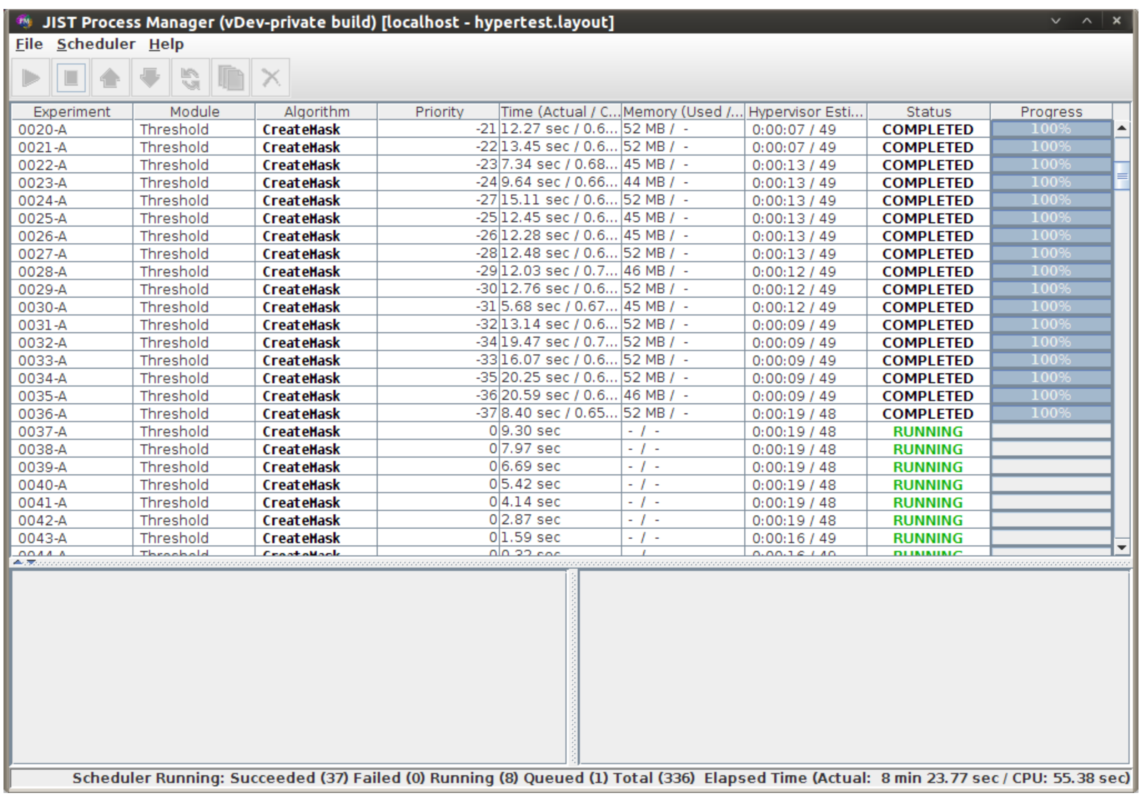

Figure 2: The JIST Process Manager. A dependency tree tracks which processes are ready to be executed. Processes that are ready are executed when they are next in queue, and when there are fewer jobs running than the maximum number of jobs if specified by the user.

CHAPTER II

CUSTOMIZABLE CLUSTER PROCESS CONTROL

Process pipelines created in the JIST Layout Tool often represent hundreds or thousands of individual jobs, each of which may take several hours to compute. While the Process Manager eliminates user overhead during execution by automatically scheduling processes in batch, it is often inappropriate to execute these computation-intensive processes locally. When first released, the JIST Process Manager relied on the Distributed Resource Management Application API (DRMAA)[3] for task scheduling and resource assignment through the Process Manager, allowing complex processes to be executed through supercomputing resources when available. This feature significantly increases computation throughput by allowing more processes to execute simultaneously, and removes the resource burden from the experimenter's machine which may be far less powerful than a processor on a supercomputer. However, while the Java DRMAA package is designed to handle Java bindings to grid process controllers, it assumes that the accessed grid architectures are relatively homogeneous. In the last several years modern DRM implementations have emerged, including the Oracle Grid Engine and the Torque Resource Manager, each prioritizing a highly customizable interface. Configuring DRMAA to manage execution of JIST processes on any users' implementation of these DRMs is a daunting task that has proved unsuccessful. In the absence of a generalizable DRMAA implementation, grid processing has been limited to individual job execution through command line modules which require user knowledge of complicated Java classpaths. This solution does not support

[3] Peter Tröger, Hrabri Rajic, Andreas Haas, and Piotr Domagalski. Standardization of an API for Distributed Resource Management Systems. In Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007), page 619-626, Rio de Janeiro, Brazil, May 2007.

batch operations, nor does it support those algorithms not classified by JIST as "Processing Algorithms," which includes algorithms derived from other frameworks (see Section I), limiting both the scale and scope of processing operations.

Individual JIST users process data in radically different environments; even those who access supercomputing resources through the same DRM may encounter different configurations of that DRM which require different access parameters. However, these users do not require extensive network connections or access to multiple schedulers; instead they need to connect to a single grid computing interface, and are not required to have an extensive knowledge of grid access protocols. To facilitate a user base that may use different DRM configurations but have minimal knowledge about the specifics of their supercomputing resources, we require a light-weight, user-friendly protocol, that is abstract so that most grid configurations are accessible, but does not come preloaded with many different configurations that the user must choose from. These requirements are not facilitated by existing middleware – most of these interfaces bind only to a subset of common DRM implementations, or bind to many systems but require extensive user-side setup and significant user resources to maintain. For instance, the highly successful Globus Toolkit may provide the functionality required for JIST, but it also is capable of supplying data about operating system, hardware, and network load among many other features. The overhead of maintaining a JIST interface to Globus[4] or another toolkit, as well as the added installation of a comprehensive solution, is beyond the scope of JIST's grid access requirements.

As an alternative to a DRMAA implementation, and in lieu of a sufficient alternative middleware, we have designed a new cluster process controller that provides abstract access to

[4] I. Foster, C. Kesselman. "Globus: A Metacomputing Infrastructure Toolkit," International Journal of Supercomputer Applications, 11(2), 115-128 (1997).

grid computing resources from JIST. Our implementation assumes (1) the user is executing JIST on a processor with access to a cluster and (2) this cluster can execute "qsub" and "qdel" commands. Typically users who have access to cluster resources are given disk space to read and write data to on a cluster executable processor. The former assumption requires the user to install JIST in this cluster-executable location, which may not be the default for the user, but eliminates the need for complex ssh logic. We have determined that this tradeoff significantly simplifies initial client-side system setup, and allows for more generalizable and maintainable grid access overall. More importantly, this assumption also improves data security, which is a primary concern when transporting medical imaging data. Portable Batch Script (PBS), a POSIX-compliant suite of commands, is supported by several popular meta schedulers and is compatible with "qsub and "qdel" commands. Due to wide support for PBS, we believe the latter assumption affords a maintainable solution to grid process control without sacrificing portability.

Implementation and Applications

In order to generalize PBS job submissions, we have introduced a PBS template file (Figure 4) which is initialized with a PBS file that submits JIST jobs to the grid to be executed.

```
#!/bin/bash
#PBS -M covingkj@gmail.com
#PBS -m bae
#PBS -l nodes=1:ppn=1:x86
#PBS -l walltime=%%WALLTIME%%
#PBS -l mem=%%MEMORY%%MB
#PBS -o %%OUTPUTLOC%%report.out
#PBS -e %%OUTPUTLOC%%report.err
#PBS -j oe

xvfb-run -f %%OUTPUTLOC%%xvfb-run -e %%OUTPUTLOC%%null -a --
server-args="-screen 0 1600x1280x24 -ac -extension GLX" %%
COMMAND%%
```

Figure 4: Template file which executes JIST jobs on the Vanderbilt ACCRE grid.

As visualized by Table 1, this template allows the user to specify grid-specific parameters, as well as additional commands specific to the user, while special variables (see Table 1) distinguished by "%%" characters are dynamically instantiated by JIST's Process Manager for each process, providing seamless integration with the user's PBS configuration. These variables represent known parameters that may depend on the experiment and process to be run.

TABLE I
JIST Template Interface

Variable	**Description**
EMAIL	The email address that the grid should send notifications to.
MEMORY	The estimated memory requirements for the process.
WALLTIME	The estimated wall time requirements for the process.
OUTPUTLOC	The output location for process results.
COMMAND	The Java call specified by the JIST classpath and process command.

When a process begins, the Process Manager reads the template file specified by the user in the Layout Preferences Panel (Figure 5). JIST then creates a pbs file in the output directory for the process, replacing all variables with the correct values. Finally it submits the job specified by the new pbs file to the available cluster via the "qsub" command. If the user clicks the Stop or Stop All commands in the Process Manager, JIST submits a request to cancel the job via the "qdel" command.

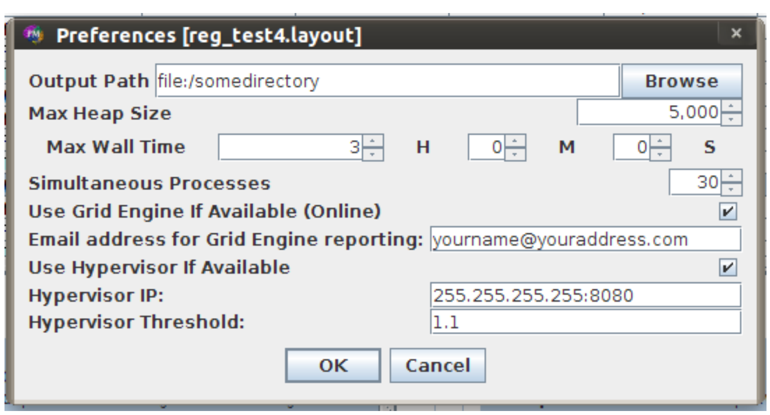

Figure 5: The Layout Preference Panel. The user can specify the output location, heap and wall time estimates in lieu of Hypervisor prediction, as well as grid engine and hypervisor parameters (see Chapter III).

Often a user will need to track the progress of a job during execution, or determine why a job has failed. This can be done both through JIST and through log files found in the process's output directory. Should the user wish to track the job's status directly from the grid engine, the Job ID provided by the grid can be accessed through the JIST command line output, or via a text file in the process's output directory. This Job ID can then be used to request information from the grid interface directly. The user can also track output and error data through the Process Manager interface, or through the log files in the process's output directory. For example, the PBS file created by the template file in Figure 4 would log this data to the report.out and report.err files respectively. Access to this data both inside and outside JIST simplifies the debugging of template files and complicated workflows, as well as the diagnosis of potential grid interface issues.

Our PBS-based process control system has proven immediately useful to researchers, allowing large scale batch operations through JIST where previously only those workflows that could feasibly be executed locally were possible. This system also permits advanced quality control measures for JIST and JIST-supported algorithms by facilitating the efficient execution of batch unit tests on the ACCRE cluster. Finally, it provides an opportunity for us to enhance JIST's resource scheduling and allocation mechanisms, which we describe in the following section.

CHAPTER III

CENTRALIZED PROCESS SUPERVISOR

Many of the DRM implementations considered in Section II require an estimate of the resource requirements (memory and wall time) that a process will consume. Users creating a grid engine work flow must make an estimate, which might be incorrect, or which may not be accurate for all jobs in the workflow. This inaccuracy can greatly impact efficiency when submitting to a grid engine scheduler; overestimating resource requirements could place an individual process lower in the cluster queue, while underestimating them loses time when the job has to be resubmitted. In large process trees both of these options could significantly increase the runtime of the tree as a whole.

In addition to cluster process controller described in Chapter II we have implemented a centralized process supervisor, called the Hypervisor, which provides memory and wall time estimates for each process individually with minimal user interaction. In the following sections, we describe the existing technologies that compose this service, the implementation and methods used to build the service and calculate accurate estimates, and provide an analysis of the convergence of the estimator on several workflows with several confidence thresholds.

Existing Technologies

An Apache Axis 2 webserver facilitates communication between JIST and the database server. Axis2 allows development of both client- and server-side code in Java, using Plain Old Java Objects (POJOs) which are converted automatically to Web Service Definition Language

(WSDL) messages[5]. These features allows rapid integration into JIST's Java codebase and ensures future interoperability by placing client-side code directly into JIST.

A MySQL database stores usage statistics on a secure Ubuntu 64-bit virtual machine. MySQL is accessible from our server code through the java.sql library[6], and is scalable to handle a reasonable estimate of our required throughput[7].

VirtualBox[8] houses our Ubuntu 64-bit virtual machine. This machine is secured by opening only the port required by Axis2 for communication.

Implementation

Message passing

Client-side code exists in several places throughout the JIST source code, allowing new execution contexts to request resource estimates, and finishing contexts to update the database with the inputs and resource usage of the completed process. Both types of messages are only sent if the user enables resource estimation and provides the IP address of the Axis2 server (Figure 5).

The Axis2 server-code receives these client messages and executes the appropriate routine. These routines call a MySQL database whose location is specified in the preferences of the Axis2 server-side implementation. An update request simply returns a Boolean value to the client confirming that the update was successful. An estimation request returns time and memory

[5] The Apache Software Foundation. (2004). *Welcome to Apache Axis2/Java*. Retrieved April 12, 2011, from http://axis.apache.org/axis2/java/core/

[6] Oracle Corporation. (2011). *Package java.sql*. Retrieved November 8, 2011, from http://download.oracle.com/javase/6/docs/api/java/sql/package-summary.html

[7] Oracle Corportation. (2011). *Guide to Scaling Web Databases with MySQL Cluster*. Retrieved November 8, 2011, from http://mysql.com/why-mysql/white-papers/mysql_wp_scaling_web_databases.php

[8] Oracle Corporation. (2011). *Welcome to VirtualBox.org!* Retrieved November 8, 2011, from https://www.virtualbox.org/.

estimates as an integer array of size two. The client extracts these values and formats them to be compatible with the current JIST implementation.

Data Storage

The MySQL database creates a table for each algorithm it encounters named with the MD5 hash of the algorithm's name. Estimation requests simply hash the algorithm name and request the data under this pseudonym. This ensures that table names will not exceed the 64-character limit imposed by MySQL.

Within an algorithm table, each column represents either an input or a resource requirement. Since the number of inputs varies based on the algorithm, while the number of resource requirement outputs provided by JIST is not, each column that represents an input is named as "inputX", where X is a number, while resource requirement variables have specific names. The total number of inputs and outputs is limited by MySQL to 4096, and cannot exceed 65,535 bytes of storage. These constraints have not been found to limit the number of inputs logged for JIST algorithms.

Each row of the table represents a specific instance of the algorithm's execution, and stores each of the execution's inputs, and the observed resource requirements.

This storage scheme allows past executions to be rapidly compared with a proposed execution via the server's resource estimation routine.

Resource Estimation

Run-time and memory usage predictions are still an active area of research, which in a general case can be inaccurate and error-prone. In light of the current literature, several different techniques for estimating run-time resource requirements were tested:

- Mean resource usage
- Exponential smoothing
- K-Nearest Neighbors

Mean Resource Usage and Exponential Smoothing

The mean of resource usage in previous runs of the same module as well exponential smoothing[9] of this data were implemented, in the hope that a computationally simple solution would suffice for our performance requirements. Exponential smoothing is meant to overcome the disadvantages of taking a simple average, or of taking the longest observed time: while the simple average is less likely to greatly overestimate, taking the longest observed time greatly decreases the chances that a job will fail because the resource requirements were underestimated. This technique was chosen based on the assumption that users will tend to execute a module many times within shorter bursts, and these users will likely be processing somewhat homogeneous data sets. Thus, weighting the most recent data (theirs) strongly was assumed to be a better method than weighting previous jobs from different users or on different data sets. Taking a weighted average based on recent events is meant to overestimate less than taking the longest observed time while protecting more jobs from failure.

Unfortunately, this assumption provided only slight performance gains over the mean and random guess methods. Furthermore, contrary to our initial belief that recent data could serve as

[9] O. Sonmez, N. Yigitbasi, A. Iosup, and D. Epema. "Trace-based evaluation of job runtime and queue wait time predictions in grids," in HPDC, 2009, pp. 111–120.

an indicator for resource requirements in the near future, our exponential smoothing method proved unable to learn over time and converge towards satisfactory performance.

K-Nearest Neighbors

Beyond these simple techniques JIST has a distinct advantage over many systems discussed in the literature regarding run-time predictions[9,10,11]. The inputs to a JIST module are defined by the module-writer such that JIST knows the number of inputs as well as their individual labels and the data types. When a process is about to be scheduled by the Process Manager, the label, data-type and value of the inputs of the module that the process will execute are easily accessible. After execution, the inputs are still accessible in addition to information about the resources that the process required to run. Because the inputs and outputs for a given process are well-defined, and each process that executes the same module has the same number of inputs with the same labels and data-types, it is much simpler to aggregate usage statistics and to interpret those statistics before a process is scheduled than it is for many other systems. We have combined this key advantage with a K-Nearest Neighbors (KNN) implementation to derive the relationship between a process's inputs and its resource requirements.

[10] H. Li, D. Groep, L. Wolters. "An Evaluation of Learning and Heuristic Techniques for Application Run Time Predictions," in Proceedings of 11th Annual Conference of the Advance School for Computing and Imaging (ASCI), 2005.

[11] O. Sonmez, N. Yigitbasi, S. Abrishani, A. Iosup, and D. Epema. "Performance Analysis of Dynamic Workflow Scheduling in Multicluster Grids," in HPDC, 2010, pp. 49-60.

Methods

Definition of Estimation Success

For a resource estimation technique to be considered successful in our usage case, it must be able to perform moderately well with few training samples, and it must converge over the term of a typical JIST processing workflow. A user who engages the estimation module and sees failures or unusually long queue times will not be willing to continue using it; on the other hand it is reasonable to suggest that the estimator should be more accurate over time, and since a user may not process a similar workflow the next time they engage the module, some convergence should be evident during the lifetime of the particular workflow.

The initial accuracy of an estimator could be analyzed simply by considering the percentage of jobs that completed successfully. However, quantifying convergence towards a successful estimation, and an estimator's success compared to an algorithm that is similarly accurate upon initialization, is more complex. If the resource requirements for a process are overestimated, the process may wait longer in queue before it can be executed. If underestimated, the process may be terminated by the scheduler before it can finish, and will have to be run again with a higher estimate of the resource requirements.

To provide a numerical comparison of different estimation techniques, we have constructed a unique cost function that penalizes both overestimation and underestimation, where the loss for a particular estimate is modeled as a function of the estimate. While it would be more precise to model overestimation as the time spent in queue with due to the actual estimate, versus how long it would have spent in queue given the true resource requirements, the latter cannot be determined accurately. For each resource requirement, wall time and memory, we model overestimating the resource requirements of a process as the difference between what is estimated and what the true resource usage is,

$$L(\theta) = \theta - t \tag{1}$$

where θ is the estimate and t is the true resource requirement. Both of these values are known after execution. Underestimating the resource requirements of a process is more complicated. The grid engine will allow the process to execute until the estimated resource consumption, $\theta,$ is reached, at which time the process will be terminated. In the case of an underestimated process, the Hypervisor does not attempt to re-estimate the resource requirements, because it will not necessarily have more reliable information about the process on a second pass. It is up to the user to determine the resource estimation for subsequent attempts to execute the process. Thus the loss function is represented by the resources used by the process before it is terminated on the first execution and the loss incurred by whatever function the user creates for estimating resource requirements successive executions, or:

$$L(\theta) = \theta + L(f(\theta)) \tag{2}$$

where f(θ) is a user-defined function. Since this function depends on the user, we simplify the loss function for estimation in our analysis:

$$L(\theta) = \theta \tag{3}$$

Equations (1) and (3) model our belief that a user will prefer a slightly longer queue time to having to rerun processes, so that the goal for the estimation algorithm is to approximate the resource requirements as closely as possible without underestimating.

KNN Implementation

The inputs to our KNN implementation are the inputs of the module to be run, and whose outputs are the expected execution time and memory consumption of the module given those inputs. Since image data must be obscured to ensure subject privacy, only the dimensions of image inputs are sent to the Hypervisor. We assume that images that vary greatly in size may also have different resource requirements when processing them. While comparing the images themselves could provide more information on how the algorithm performs, we do not consider our assumption to be a major loss in data. A general distance metric for any two images would need to consider the algorithm that is being run – different JIST algorithms, and different image inputs within algorithms, may need different distance metrics. While this would increase overhead when increasing the library of JIST algorithms, simply comparing the image dimensions does not. We have made similar assumptions about configuration files, matrices, and non-numeric options – for these inputs we only compare whether or not they are identical. For numerical inputs, we use a Euclidean distance metric. These assumptions decrease complexity and improve maintainability, while still allowing the Hypervisor to report adequate estimations.

The ideal number of neighbors to choose may vary greatly from algorithm to algorithm; the number of training samples, the number of inputs to the algorithm and how much those inputs vary in user experiments cannot be reasonably determined *a priori*. Thus in order to maintain flexibility in our implementation, we have chosen to take the $\sqrt{n}$ nearest neighbors which we assume will retrieve a sufficient coverage of a neighborhood without overfitting. Estimation is suppressed until there are at least ten training samples for a particular algorithm.

Finally, we have determined that for different workflows and experiments the user may require a different level of confidence that an algorithm will succeed based on the resulting resource estimates. To provide this flexibility, we have added a threshold to the Layout Preferences Panel (Figure 5). For threshold values greater than or equal to one, the Hypervisor returns

$$f(\bar{x}) = \max(x) * \alpha \tag{4}$$

for each resource requirement, where x is the array of the resource requirement for each neighbor, and α is the user-selected threshold. For threshold values less than one, equidistant neighbors form finite Gaussian curves, which are then weighted by their inverse distance and summed with all other curves. The returned resource estimate represents the point on the resulting curve at which α*100 percent of the chosen set of neighbors would have succeeded.

Typical Workflows for Analysis of Estimation Modules

In order to cover a range of common experiments that would benefit from resource estimation, we tested two different workflows. The first (Figure 6) simply creates a mask over many images that have the same dimensions and thus appear to be identical to our estimator. This provides a homogenous process tree with several hundred instances to train the estimators on (Figure 6a), demonstrating the ability of our estimator to converge quickly to an estimation that succeeds for processes that will require similar resources. The second (Figure 6b) registers a large set of images to itself. Those processes that register an image to itself will take significantly less time and memory than others of its class, without providing the estimator with a clear numeric reason for this fluctuation. This workflow tests the ability of our estimator to manage a large number of inputs, which unlike the first workflow will vary for each process, as well as its performance under extreme fluctuations in resource requirements that will not have an obvious numerical difference associated with it.

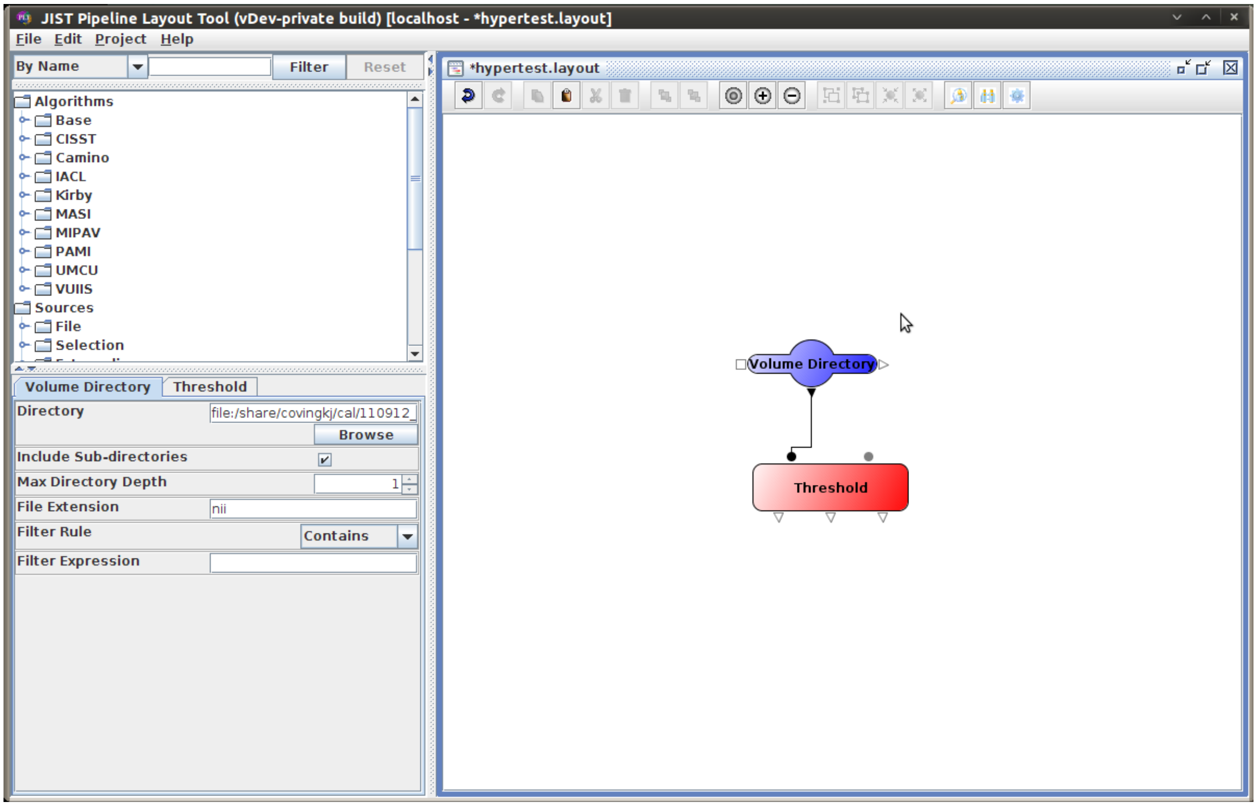

Figure 6a: Layout for the Create Mask algorithm.

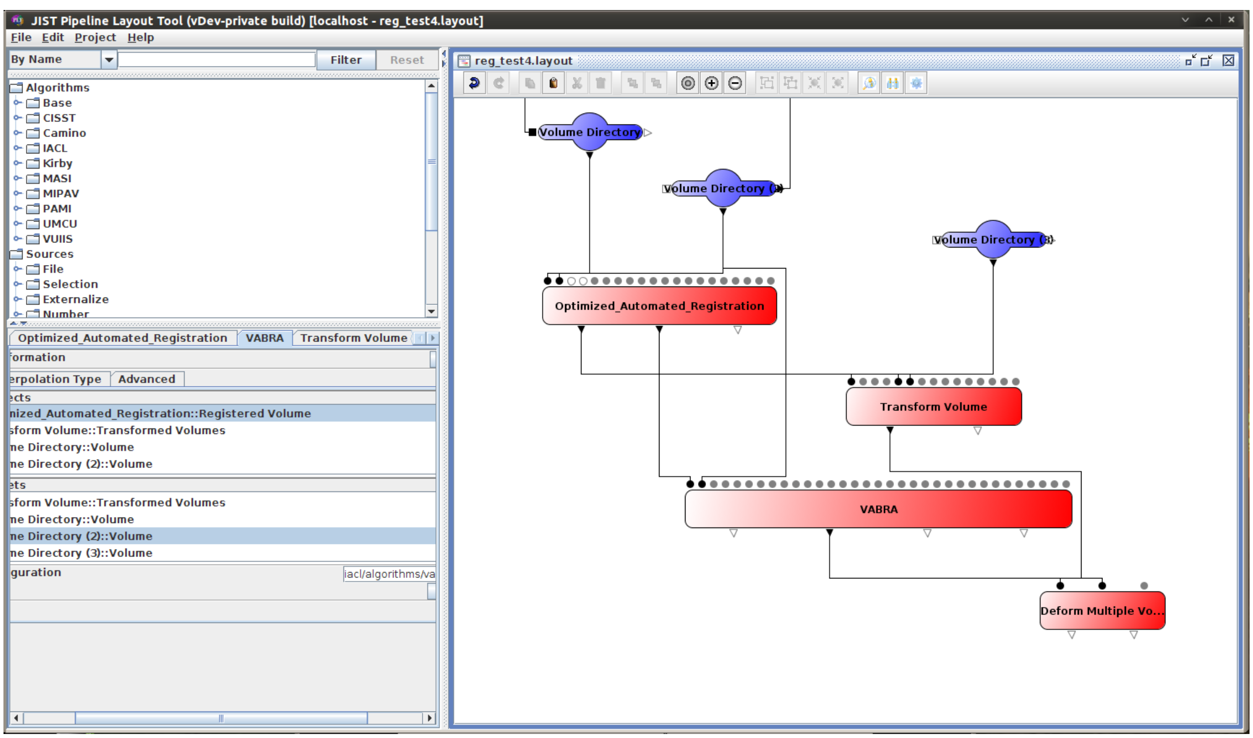

Figure 6b: Layout that registers a specified dataset to itself.

Results and Discussion

In order to test both the convergence of the Hypervisor on multiple workflows and the proper threshold for different types of processes, the Create Mask and Registration layouts were tested with a threshold of 1.2, 1.0, and 0.8. Since we found that the resource consumption of the processes is deterministic, experiments were only processed once per threshold.

The Flirt and Vabra, and the Transform and Deform Volume algorithms within the Registration Layout were found to have similar properties; as such our analysis focuses only on the Vabra and Deform Volume processes within this layout. The first ten processes for each algorithm were executed independently of the others, to ensure that the jobs would have sufficient training data to receive estimations. After the first ten processes, the "Simultaneous Processes" value in the Layout Preferences Panel was set to 10 for the Create Mask Layout, and to 30 for the Registration Layout, to maintain the number of sets of processes allowed to converge to a reasonable estimation given that there are three times as many processes in the Registration Layout. The estimator converged rapidly however, so only the first 300 processes in the Registration Layout are analyzed here. Table 2 presents the results of these experiments, omitting the 0.8 threshold for Vabra and Deform Volume because processes were failing the majority of the time.

These results demonstrate two important messages: 1. with a well-chosen threshold, our implementation provides highly stable results with only a few training samples; 2. this threshold varies depending on the process type and the user's preference for safety vs. lowered queue times. 1. Given at least ten training samples, a threshold of 1.2 allowed most processes to succeed. Note that since the major varying input for each of these process types is the image input, the neighborhood is relatively simple. While more complicated experiments might require more training data, or a higher threshold, most workflows will not vary all of the parameters that they are given. Because of this tendency, our observations are relevant to most JIST users. 2. Workflows that require more resources, and those whose varying parameters more heavily in impact resource usage, require a higher threshold for optimal results. For homogeneous processes like those in the Create Mask Layout, a threshold of 1.0 results is dramatically lower loss than those above or below 1.0. For more heterogeneous workflows, a higher threshold such as 1.2 is more appropriate. A threshold below 1.0 does not perform as well, and may be most useful in situations where it is less important that a process succeeds, and more important that is start at a better position in the grid engine queue.

TABLE II
HYPERVISOR EXPERIMENTAL RESULTS

Process Type	Threshold	Number of Processes Executed	Number Estimated by Hypervisor	Number not Underestimated (Number Completed)	Average Memory Loss (MB)	Average Wall Time Loss (ms)
Create Mask	1.2	321	321	318	28.2229	13577.1
Create Mask	1	321	321	288	7.84472	8455.57
Create Mask	0.8	321	316	97	27.8474	17450.3
Deform Volume	1.2	300	300	284	147.85	270571
Deform Volume	1	300	300	44	851.273	196197
Vabra	1.2	300	300	266	406.587	112
Vabra	1	300	300	22	1660.41	960149

CHAPTER IV

HIGH-THROUGHPUT BUNDLED RESOURCE IMAGING SYSTEM

Picture Archiving and Communication Systems (PACS) enables systematic and verifiable tracking, access, visualization, and data providence for medical imaging data throughout the healthcare information management system from acquisition to interpretation, especially for three- and higher- dimensional imaging techniques, such as magnetic resonance imaging[12]. These pervasive informatics platforms serve as ideal targets in which to integrate and evaluate new analyses and techniques for eventual inclusion in patient care[13]. Advanced integration of PACS system throughout the entire chain of care is an active area of informatics research and engineering development[14].

A variety of imaging frameworks have emerged to fit the needs of research groups both in terms of functionality and visualization capabilities[1,15,16,17], while a multitude of specialized image analysis techniques remain largely in the hands of their creators and collaborators. Integrating these techniques, and bridging the gap between these emerging image analysis technologies and those technologies ready for clinical research and patient trials, is a

---

[12] L. H. Schwartz, P. Kijewski, K. Lynch *et al.*, “Review and interpretation of MR imaging studies with PACS: creating uniform series descriptors for radiologists and referring physicians,” AJR Am J Roentgenol 179(3), 575-7 (2002).

[13] T. Yoshinobu, K. Abe, Y. Sasaki *et al.*, “Data Management Solution for Large-Volume Computed Tomography in an Existing Picture Archiving and Communication System (PACS),” J Digit Imaging, (2009).

[14] T. Fillicelli, “Future of PACS: advanced integration with RIS and workflow management,” Radiol Manage, 23(1) 12-3 (2001).

[15] M. McAuliffe, F. Lalonde, D. McGarry, W. Gandler, K. Csaky, B. Trus. "Medical image processing, analysis and visualization in clinical research Proc IEEE Intl Symp on Computer-Based Medical Systems CBMS, 381-386 (2001).

[16] B. Lucas, B. Landman, J. Prince, D. Pham. "MAPS: A Free Medical Image Processing Pipeline," Organization for Human Brain Mapping, Melbourne, Australia, (2008).

[17] S. Peiper, M. Halle, R. Kikinis. “3D Slicer,” Proc IEEE Intl Symp on Biomedical Imaging ISBI 1, 632-5 (2004).

monumental effort[18,19,20,21]. Here, we explore a set of technologies which could allow for earlier feedback of clinical collaborators on emerging technologies and facilitate collaborations between imaging researchers, who typically develop methods using custom software platforms, and clinical researchers, who typically prefer established visualization and analysis approaches afforded by PACS-centric systems.

Custom image post-processing in the clinical as well as the research medical imaging setting is often limited by the separation of the processing from the image acquisition hardware platform that connects to the institution's PACS. The separation decreases the chances that the post-processed images will ever be entered into the PACS and creates an obstacle for routine usage and acceptance of the available post-processing algorithms. For example, novel image reconstruction techniques often require substantive changes to the software pipeline within the image reconstruction environment. PACS systems are typically not configured to provide seamless acceptable, transmission, and access the raw data (prior to reconstruction), so it is often not possible to use a PACS system to archive the requisite non-imaging data and meta-data necessary for reconstruction. Hence, an alternative data processing workflow must be constructed, which can involve error-prone human-machine interaction (e.g., numerous clicks). Finally, the alternative workflow must be configured as a DICOM send node and must retain institution specific accession syntax (which requires additional meta-data and programming) and must be authorized to transmit data to the central PACS server (which may conflict with institutional security policies).

---

[18] W. Wolfson, "caBIG: seeking cancer cures by bits and bytes," Chem Biol, 15(6) 521-2 (2008).
[19] A. C. von Eschenbach, and K. Buetow, "Cancer Informatics Vision: caBIG," Cancer Inform 2, 22-4 (2007).
[20] K. K. Kakazu, L. W. Cheung, and W. Lynne, "The Cancer Biomedical Informatics Grid (caBIG): pioneering an expansive network of information and tools for collaborative cancer research," Hawaii Med J 63(9), 273-5 (2004).
[21] S. Pieper, B. Lorensen, W. Schroeder, R. Kikinis. "The NA-MIC Kit: ITK, VTK, Pipelines, Grids and 3D Slicer as An Open Platform for the Medical Image Computing Community Proc IEEE Intl Symp on Biomedical Imaging ISBI, 698-701 (2006).

To address this problem, we developed a high-throughput bundled resource imaging system (HUBRIS) for integrated medical imaging analysis which directly interfaces with the scanner and injects custom image analysis results into the PACS pipeline at the point of acquisition. HUBRIS extends the Philips Research Imaging Development Environment (PRIDE, Philips Healthcare, Best, The Netherlands). Using HUBRIS, three distinct systems (the image acquisition equipment, image post-processing algorithm workflow server and algorithm execution server) seamlessly communicate to provide advanced image post-processing directly from the medical imaging equipment. The images generated by this framework can be sent to the institution's PACS in the same manner as other acquired images.

Methods

In this proof-of-concept case, the medical imaging equipment is a magnetic resonance imaging scanner (Philips Healthcare, Best, The Netherlands) running release 2.6.3.4 software or newer. The scanner is equipped with the inline PRIDE functionality that allows any user-defined executable or script to be invoked automatically upon the completion of an image acquisition. The automatically called process is passed a unique identifier that enables the process to extract the associated image volume from the scanner's database. Additional parameters can be passed using the inline PRIDE process invocation. Upon completion of operation, the invoked inline PRIDE process can write resulting images to a pre-defined folder location. Images found in that location will be read automatically by the scanner's software and imported into the scanner's database to enable the new images to be easily sent to the PACS. Typically, inline PRIDE processes run on the scanner console computer and are restricted in memory and CPU usage. Furthermore, the available post-processing library and tools are limited to those that have been compiled and provided for the scanner console computer's operating system (Windows XP embedded 64-bit).

To increase the power and flexibility of the image post-processing workflow available directly from the scanner, we have integrated the Java Image Science Toolkit (JIST)[1] server with the scanner's inline PRIDE capability to provide access to a large library of algorithms using JIST's open application program interface (API)[2]. JIST is an open-source, platform-independent framework to rapidly develop image analysis tools and distribute them to the scientific community. JIST provides a high level of interoperability, advanced batch processing tools, and requires minimal additional programming or computational overhead. Finally, to address the demands for memory and processing power by many of the algorithms, the JIST server is able to sends jobs to a supercomputer grid.

In this case, the processing power is provided by Vanderbilt's Advanced Computing Center for Research and Education (ACCRE, http://www.accre.vanderbilt.edu/). ACCRE is an education/research grid High Performance Computing facility with a theoretical peak performance of 20 TeraFLOPS using approximately 3,000 interlinked x86-based core processors running 64-bit Linux. Compute nodes all have a disk drive and dual copper gigabit Ethernet ports, but hardware visualization capabilities are disabled. Resource management, scheduling of jobs, and usage tracking are handled by an integrated scheduling system by Moab/Torque.

Results

Figure 7 illustrates the data flow in the HUBRIS system. Running the "Start Hubris" tool in Inline PRIDE opens StartHubris.xml. These tools are defined with a simple, human readable eXtensible Markup Language (XML) definition file as shown in Figure 8. This file configures

PRIDE to call StartHubris.bat, which first calls the PRIDE-Leacher tool to pull the image data (or raw data) from the internal PRIDE database and copy it into to the "tempinputseries" folder. While access to raw data is not a natural part of the in-line PRIDE feature, the unique identifier passed to the inline PRIDE invocation is enough information to query the scanner's database and to extract raw data associated with a given MR imaging series. As long as the called script is able to go all the way from raw data to XML/REC image results, the inline PRIDE tool accepts the results even if the XMLRECleacher program is never used. However, in the cases where raw data is saved and there are some scanner produced images, those XML/REC files are a convenient template for stuffing in the new reconstruction results. Different configuration files with different default arguments can be configured to automatically run for differing modalities. While the scanner's inline PRIDE feature is limited to serial execution, and as such inline PRIDE jobs cannot run simultaneously, multiple images and raw data acquisitions can be simultaneously processed by sending these to the "Stage Hubris" script.

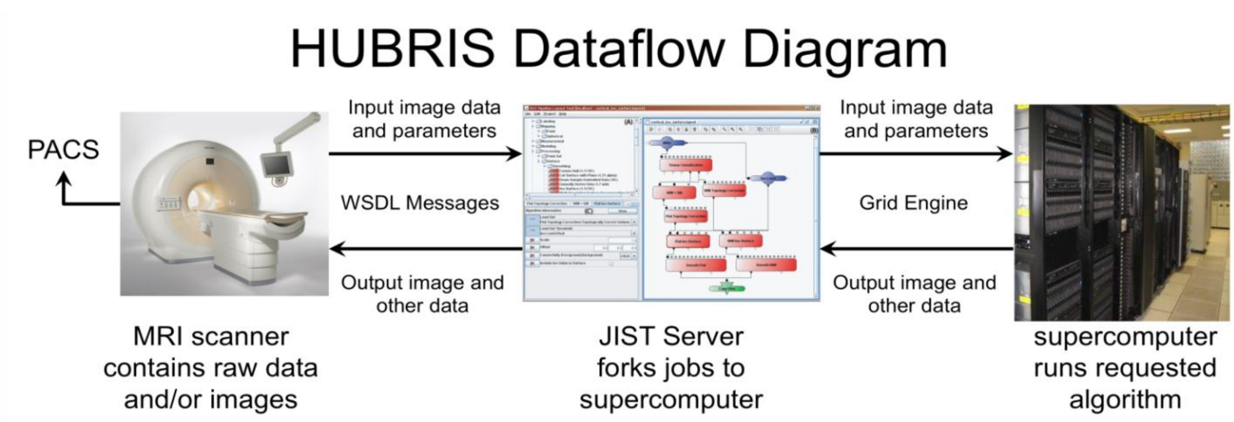

Figure 7: HUBRIS dataflow diagram. The image acquisition device (left) automatically triggers a processing script with each scan event of a specific type. This script communicates with a JIST server (center) through a standard WSDL connection. In JIST stages the data and manages job submission to a parallel computing facility using the Sun Grid Engine (right).

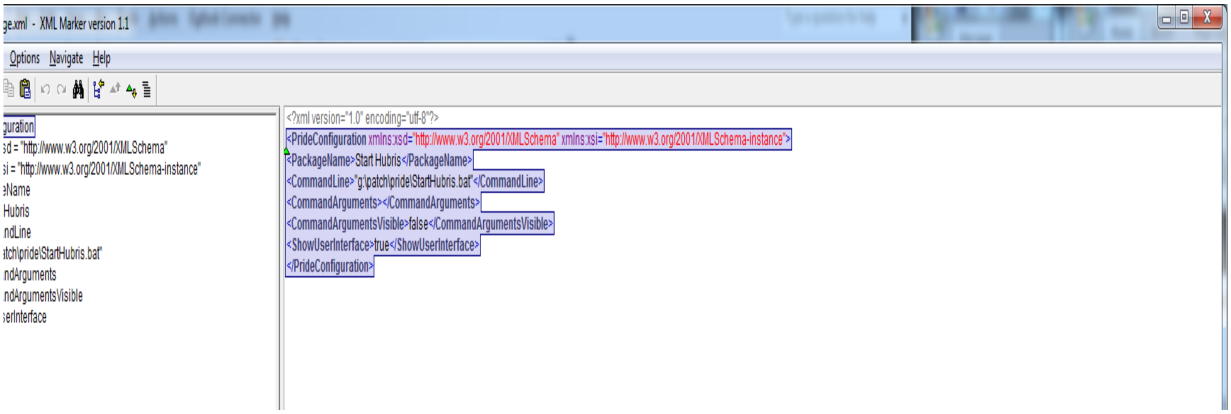

Figure 8: Inline PRIDE interface configuration. Standard XML scripts define process events and allow for default parameters to be configured.

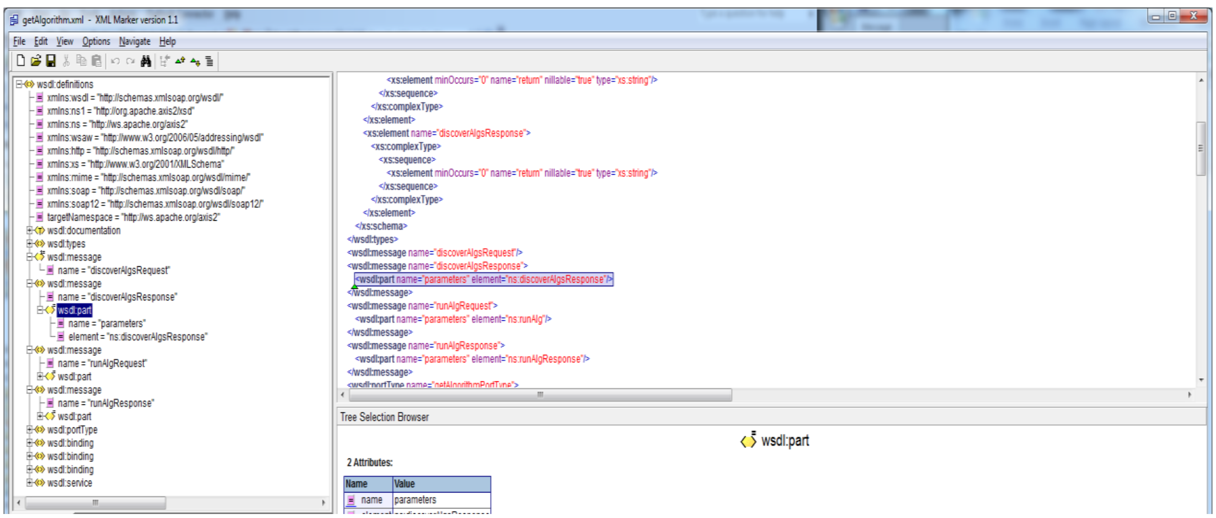

Figure 9: The JIST WSDL server provides XML encapsulation of algorithm discovery, self-documentation, algorithm submission, and response.

A (WSDL) Axis 2 server provides access to the JIST image processing server routines (Figure 9), and facilitates data transfer via client- and server-side Java methods that use Plain Old Java Objects (POJOs). The Axis2 server automatically converts these POJOs to WSDL XML, which clients use to access server data, and which the server stores on the Internet to be accessed by clients.

Once the data are extracted, a Java class HubrisClient, with WSDL support via Axis 2 libraries, is initialized and loads the data from the tempinputseries folder. HubrisClient is designed to either automatically pass a predefined script along with the tempinputseries data to the server, or in the absence of a script or server location information to load a graphical user interface (GUI).

The GUI simplifies debugging and configuration of the client-server architecture. The GUI allows the user to specify the IP address and port number for the Axis 2 server. Once HubrisClient has successfully contacted the server, it uses Axis 2 WSDL messages to call Jist's discover() method from a Java class HubrisServer, which is called by the Axis 2 server. HubrisServer collects the results from discover(), and builds a list of available algorithms along with their inputs and requirements. This list is returned to HubrisClient, which populates the GUI with the possible algorithms (Figure 10a). Note that HUBRIS/JIST infrastructure detects any custom modules installed on the server and exposes these functionalities to the client — even if the client does not know *a priori* that such modules exist.

When the user clicks on a desired algorithm, a new window is created which lists all available input arguments, notes which arguments are required, and allows the user to set the desired input arguments (Figure 10b). The user can then choose to run the algorithm with the appropriate input arguments, or generate the corresponding script for use in automatic run configurations.

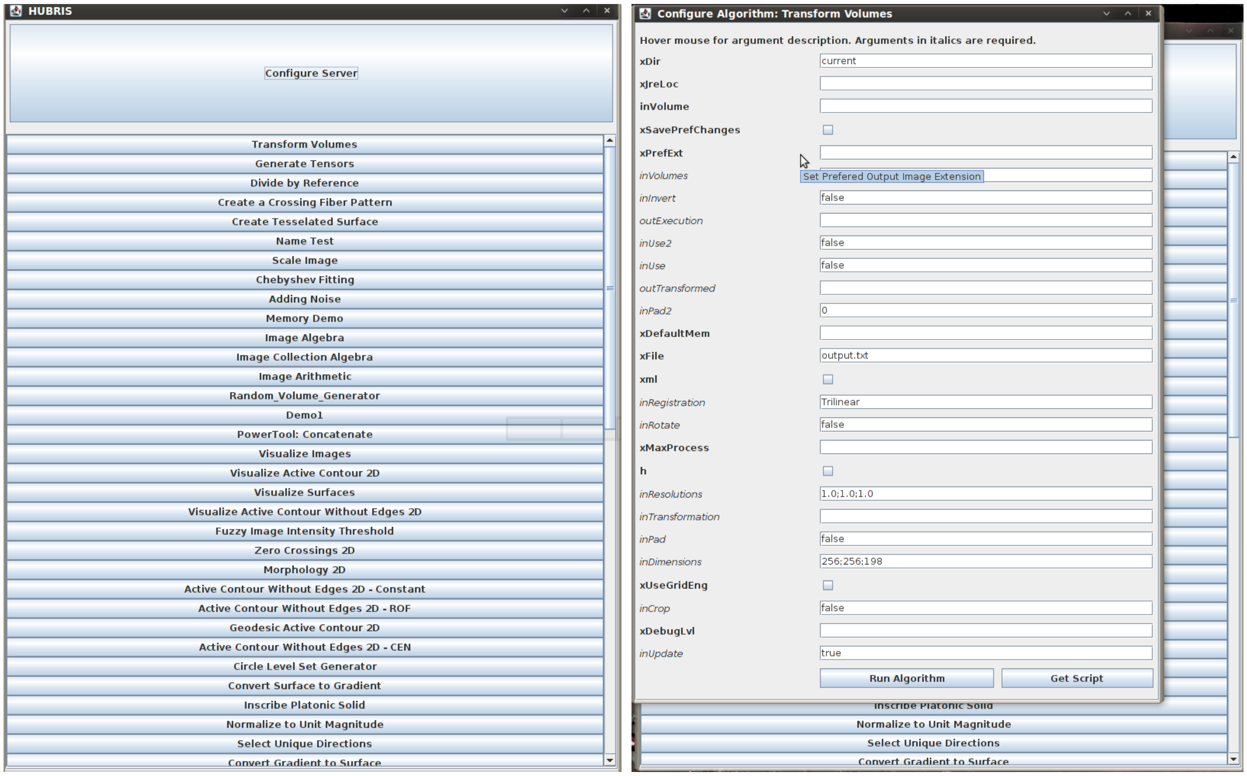

Figure 10. Client-side Hubris GUI. Server configuration and a clickable listing of algorithms available for processing on the server are made available to the user (left). Arguments to individual algorithm are customized by the user and then either submitted for processing or the corresponding script is saved for future use (right).

If the user elects to run the algorithm as defined, HubrisClient uses Axis 2 generated WSDL messages to call the RunAlgorithm method in the HubrisServer class on the server, to send the image data, and to provide any other necessary data as arguments as illustrated in Figure 11. RunAlgorithm parses the arguments and sends the request and all necessary data to JIST's run() method on the cluster, which calls the specified algorithm and returns the output to RunAlgorithm. The server sends the output back to HubrisClient, which writes the output data in an appropriate place. Finally PRIDEXMLRECleacher removes the original image data from tempinputseries to prevent a buildup of data on the server. For grid processing, HUBRIS invokes executables through the X virtual framebuffer (Xvfb, David P. Wiggins, The Open Group, Inc.) framework so that a software X session is available for algorithms which require access to visual

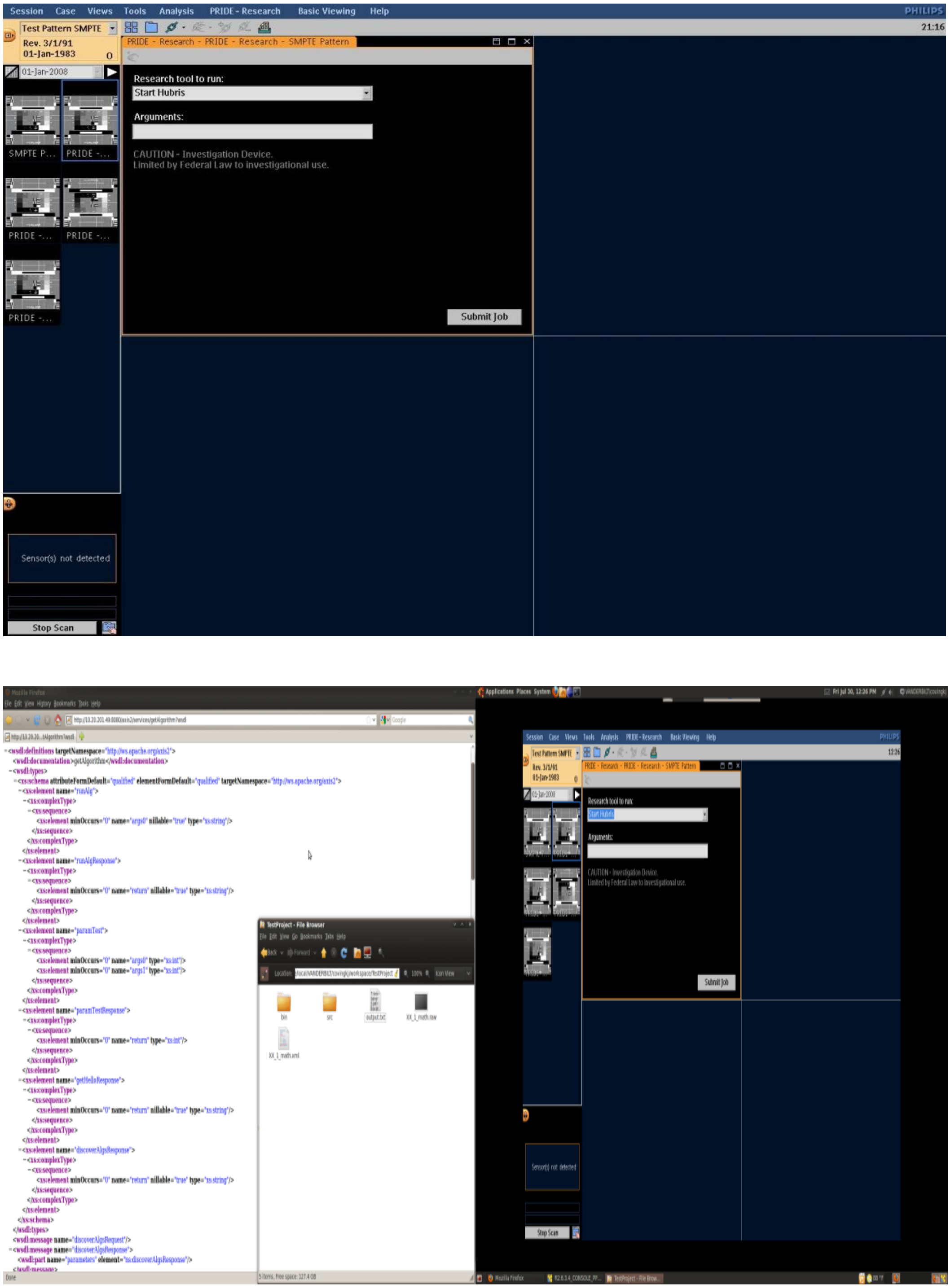

Figure 11: Inline PRIDE interface on the Philips MRI scanner (top) communicates with a JIST server running on a different computer (bottom) to send, processes, and retrieve imaging data.

elements (e.g., font handles or rendering libraries) even if the elements are not displayed on screen.

Image arithmetic is a simple example of HUBRIS functionality. In order to add two reconstructed images from the MRI console, the user sends the first image to a temporary location via StageHubris, The user can then send the second image via StartHubris, either by sending the image with a predefined HUBRIS script that sends the two images to the server and requests that the two images be added, or by running StartHubris with no commands. The latter will cause the HUBRIS GUI to be initialized, at which point the user can select the ImageArithmetic algorithm, leave the operation argument at its default (add), and input the locations of the images in the “inFirst” and “inSecond” arguments respectively. Finally the user selects “Run Algorithm,” at which point the image data and arguments are sent to the server, the server performs the operation, and the output data is returned to the tempoutputseries folder, which the user can then access. Figure 12 demonstrates an example of the potential use of HUBRIS for testing a novel off-line image reconstruction method for diffusion-weighted (DW) sensitivity encoding (SENSE) MR images. It compares fractional anisotropy (FA) map acquired using conventional single-shot (ssh) (Figure 12a) and new multi-shot (msh) EPI (Figure 12b) technique. Usually the evaluation of a new imaging method has been limited to specific users and systems due to relatively large volume of raw data and/or intensive user interactions and computations required for off-line image reconstruction. However, the use of HUBRIS system could provide more convenient environment for imaging scientists and clinicians by incorporating any conventional and emerging technologies within existing PACS system.

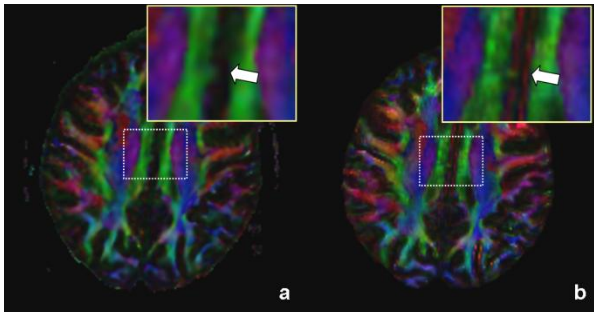

Figure 12. Example potential application of HUBRIS for the reconstruction of improved diffusion tensor imaging (DTI) fractional anisotropy (FA) maps using a multi-shot acquisition. The presented color maps from single-shot (a) and multi-shot (b) echo-planar imaging (EPI) data show multi-shot EPI exhibits reduced blurring and geometric distortions. The generation of the improved images requires a custom reconstruction algorithm that can be accessed by HUBRIS.

CHAPTER V

CONCLUSIONS AND FUTURE WORK

Given layouts that represent workflows that we might expect a user to process through JIST, the Hypervisor converges so rapidly to its estimate that we can rely on it for resource estimations after as little as ten samples processes. This is due to two key advantages that the JIST processing pipeline has compared to other systems that require resource estimates. 1. The inputs to an algorithm are known *a priori*, and their types are fixed to predetermined JIST classes such as 3D integer points, matrices, configuration files, and image inputs. These input constraints are ideal for a nearest neighbor model. 2. In many user-designed layouts, only one or two out of as many as several dozen inputs will vary throughout an experiment. As a result, the dimensionality of the neighborhood for the experiment may remain low, and be populated quickly by processes that executed within the layout, yielding more accurate results early in the experiment. The addition of a user-selected confidence threshold provides flexibility between efficient queue times and safe resource predictions. While this threshold varies depending on the type of layout to be processed, a sufficient threshold can be determined by considering the properties discussed in our results.

While the results of this estimation technique are highly successful, it does not consider all variables that could influence resource consumption. Our current model aggregates all usage statistics for a given algorithm independent of the user who submitted it, which grid to which it was submitted to or if it was executed locally, and the hardware that executed the process.

Furthermore, the assumptions taken by our distance metric may not be ideal. All of these considerations are promising areas of improvement for our estimator.

In addition to cluster control and resource estimation within JIST, we have presented an advanced proof-of-concept workflow to enable seamless medical image post-processing invoked from a clinical magnetic resonance imaging scanner. The generated images can then easily be transferred to the institution's PACS like any other typically acquired medical image. The system integrates three distinct systems: (1) the medical imaging acquisition hardware, (2) a JIST server to provide access to advanced open-source post-processing algorithms, and (3) a high-powered supercomputer to execute the requested algorithms. While obstacles still exist that prevent its release, the current implementation promises significant advances in the MRI processing workflow if deployed.

Appendix A

Create Mask Results, Threshold 1.2

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	107	-1	25663	-1
1	107	-1	24685	-1
2	114	-1	22176	-1
3	117	-1	29515	-1
4	114	-1	35328	-1
5	114	-1	40243	-1
6	111	-1	36235	-1
7	114	-1	32221	-1
8	111	-1	40235	-1
9	114	-1	34447	-1
10	127	136	18187	48000
11	107	136	21237	48000
12	114	136	29247	48000
13	114	136	36750	48000
14	111	136	36592	48000
15	114	136	41224	48000
16	114	136	34394	48000
17	103	136	33306	48000
18	112	136	32245	48000
19	114	136	41534	48000
20	114	136	24387	44000
21	103	152	29684	35000
22	111	152	37188	41000
23	103	136	28722	44000
24	103	152	42016	41000
25	112	136	31661	49000
26	103	136	33555	49000
27	107	136	31978	49000
28	103	136	32067	49000
29	103	136	38790	49000
30	103	136	25104	44000
31	103	136	29705	49000

32	103	136	25683	44000
33	103	136	29587	44000
34	103	134	25540	50000
35	107	134	26614	50000
36	111	134	27232	46000
37	103	136	38803	50000
38	114	134	31613	50000
39	103	134	28902	46000
40	107	128	29348	46000
41	103	123	28390	46000
42	103	123	29572	46000
43	103	128	33006	46000
44	107	133	27849	35000
45	114	136	24516	46000
46	103	136	29375	46000
47	103	128	36581	35000
48	103	128	38433	35000
49	103	136	32565	46000
50	107	136	21208	39000
51	112	136	19172	39000
52	111	136	24473	39000
53	103	136	25306	39000
54	103	136	27978	39000
55	111	136	23192	46000
56	107	136	26361	43000
57	107	136	24679	46000
58	111	136	27614	43000
59	111	128	33652	39000
60	112	134	25216	46000
61	103	136	36169	46000
62	111	136	37283	46000
63	107	134	24249	33000
64	107	133	24435	40000
65	103	133	23411	40000
66	103	133	26523	40000
67	103	133	25300	40000
68	103	134	34726	46000
69	107	133	27404	40000
70	103	134	26098	40000
71	107	134	30270	43000
72	103	134	25775	44000

73	103	134	28805	44000
74	107	133	25561	44000
75	103	134	34443	44000
76	107	133	24086	44000
77	103	134	28038	44000
78	107	133	27695	44000
79	114	128	26308	41000
80	103	128	25424	41000
81	111	128	37177	41000
82	107	128	21254	41000
83	107	128	25644	41000
84	112	128	27213	41000
85	103	128	29393	41000
86	107	128	23637	41000
87	103	128	27582	41000
88	103	128	36309	41000
89	103	136	27729	41000
90	103	136	29577	44000
91	107	136	30810	44000
92	114	134	22348	44000
93	112	136	30769	44000
94	107	134	28531	44000
95	103	136	34743	44000
96	103	134	29989	44000
97	103	134	28830	44000
98	107	136	35767	44000
99	107	134	23549	43000
100	103	134	27318	44000
101	103	134	30355	44000
102	107	136	28328	43000
103	103	136	24187	43000
104	107	136	19411	42000
105	112	136	22408	42000
106	107	136	24207	42000
107	103	136	27166	43000
108	103	136	24272	42000
109	103	136	28371	42000
110	103	136	37301	42000
111	107	136	32621	42000
112	103	128	34292	42000
113	112	128	33974	42000

114	107	134	31115	42000
115	107	134	29984	42000
116	107	134	35954	42000
117	103	134	31211	42000
118	107	134	30836	36000
119	111	134	30368	36000
120	103	134	27708	44000
121	103	134	27621	44000
122	107	134	16172	44000
123	112	134	23248	44000
124	103	134	21265	44000
125	107	134	20551	44000
126	114	134	27000	44000
127	103	134	30444	44000
128	107	134	22467	43000
129	103	134	28415	44000
130	103	134	25814	44000
131	103	134	30975	43000
132	103	134	30001	43000
133	107	136	27517	43000
134	112	134	32573	43000
135	103	134	35406	43000
136	107	136	28816	37000
137	103	134	35407	43000
138	107	136	25287	36000
139	103	136	30327	36000
140	103	136	35946	36000
141	103	136	28326	37000
142	112	136	22424	42000
143	103	136	25304	42000
144	103	134	24792	42000
145	107	134	23618	42000
146	107	134	20905	42000
147	107	134	27900	42000
148	112	134	21266	43000
149	103	134	31809	42000
150	103	134	24281	43000
151	103	134	32820	42000
152	107	134	24216	43000
153	103	134	32869	43000
154	112	134	31317	43000

155	114	134	27720	43000
156	107	134	24218	43000
157	103	134	34293	43000
158	103	134	26201	43000
159	107	134	25475	43000
160	103	134	30330	43000
161	103	134	35898	43000
162	107	136	22466	41000
163	112	136	28300	39000
164	103	136	30444	39000
165	107	136	27313	39000
166	103	136	26191	39000
167	103	134	34603	39000
168	103	136	28275	41000
169	103	136	30334	41000
170	107	136	26484	43000
171	107	136	32373	41000
172	107	136	30266	43000
173	103	134	21160	43000
174	103	136	29567	43000
175	112	134	25296	43000
176	103	136	28437	43000
177	103	136	27637	43000
178	107	134	21302	43000
179	112	134	30373	43000
180	103	134	29265	43000
181	107	134	27376	43000
182	103	134	35661	43000
183	103	134	27218	41000
184	107	134	26268	41000
185	111	134	33459	43000
186	103	134	30543	41000
187	107	134	32579	41000
188	107	134	24742	42000
189	112	134	30351	38000
190	103	134	29783	38000
191	103	134	34741	41000
192	103	134	28265	42000
193	107	134	26193	42000
194	107	134	29946	42000
195	103	134	34123	42000

196	103	134	32901	42000
197	107	134	27391	42000
198	112	134	23375	42000
199	103	134	24333	42000
200	103	134	28377	42000
201	103	134	24286	42000
202	107	134	27570	42000
203	112	134	20444	41000
204	107	134	22285	41000
205	103	134	27459	42000
206	103	134	29662	42000
207	103	134	28755	41000
208	107	134	30314	41000
209	103	134	29647	41000
210	112	134	34579	41000
211	107	134	26408	41000
212	103	134	33315	41000
213	107	134	24280	40000
214	103	134	29419	41000
215	103	134	28821	40000
216	103	134	28464	40000
217	111	134	39329	41000
218	112	134	25219	41000
219	103	134	25568	41000
220	107	134	28245	41000
221	107	134	30948	41000
222	103	134	33819	40000
223	103	134	28310	41000
224	103	134	30356	41000
225	103	134	29343	41000
226	107	134	24335	47000
227	107	134	31329	41000
228	103	134	25629	47000
229	103	134	25334	47000
230	112	134	24311	47000
231	103	134	30540	47000
232	107	134	29227	47000
233	103	134	30272	47000
234	103	134	28338	47000
235	107	134	33470	47000
236	107	134	31773	47000

237	112	134	36257	47000
238	103	134	30801	47000
239	107	134	28481	40000
240	107	134	31509	40000
241	103	134	34939	47000
242	107	134	29512	40000
243	107	134	29278	40000
244	112	134	28972	40000
245	107	134	25560	43000
246	103	134	26681	43000
247	103	134	29426	43000
248	107	134	28815	43000
249	103	134	30198	43000
250	107	134	26345	43000
251	107	134	25252	43000
252	103	134	30595	43000
253	112	134	28343	43000
254	107	134	25264	43000
255	103	134	27414	43000
256	103	134	25349	43000
257	103	134	29826	43000
258	112	134	24200	41000
259	107	134	23203	41000
260	107	134	27385	43000
261	107	134	28791	43000
262	103	134	27574	43000
263	107	134	25297	36000
264	103	134	29992	36000
265	103	134	31360	41000
266	103	134	33721	41000
267	107	134	31396	36000
268	112	134	23501	36000
269	107	134	22340	36000
270	114	134	28280	36000
271	107	134	28331	36000
272	103	134	31147	36000
273	103	134	23209	40000
274	107	134	29247	36000
275	103	134	25278	40000
276	107	134	27202	40000
277	103	134	31029	40000

278	112	136	30469	40000
279	103	134	33404	40000
280	107	136	32373	40000
281	107	136	28674	40000
282	107	136	32337	40000
283	103	136	23681	40000
284	103	136	28385	40000
285	107	136	27855	40000
286	103	136	24224	40000
287	103	136	30330	40000
288	112	136	25253	40000
289	107	136	29328	40000
290	107	136	29201	40000
291	103	136	30794	40000
292	103	136	33040	40000
293	107	134	28510	40000
294	103	136	32367	40000
295	107	136	32304	40000
296	103	134	31345	40000
297	107	134	30329	40000
298	103	134	30771	40000
299	107	134	25417	40000
300	112	134	27651	40000
301	107	134	30620	40000
302	103	134	29590	40000
303	103	134	26775	40000
304	107	134	24478	39000
305	107	134	28105	39000
306	103	134	31925	40000
307	107	134	30797	39000
308	112	134	25702	39000
309	103	134	25182	39000
310	103	134	26922	39000
311	103	134	25369	39000
312	107	134	30491	39000
313	112	134	28572	39000
314	107	134	32766	39000
315	107	134	29739	39000
316	107	134	28770	39000
317	103	134	33740	39000
318	103	134	29294	39000

319	103	134	26876	38000
320	107	134	24560	38000
321	103	134	27021	38000
322	107	134	30829	38000
323	103	134	27401	38000
324	112	134	25318	39000
325	107	134	24426	40000
326	107	134	25476	40000
327	103	134	29036	39000
328	103	134	30313	40000
329	103	134	27339	40000
330	103	134	31418	40000

APPENDIX B

CREATE MASK RESULTS, THRESHOLD 1.0

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	114	-1	24432	-1
1	107	-1	26777	-1
2	107	-1	23319	-1
3	117	-1	32452	-1
4	107	-1	23212	-1
5	107	-1	31250	-1
6	107	-1	29221	-1
7	114	-1	26320	-1
8	114	-1	36396	-1
9	114	-1	27399	-1
10	103	114	20457	36000
11	103	114	24476	36000
12	103	114	26351	36000
13	103	114	29547	36000
14	122	114	24232	36000
15	107	114	24036	36000
16	103	114	29415	36000
17	107	114	31219	36000
18	103	114	30375	36000
19	107	114	32618	36000
20	122	122	21220	29000
21	103	122	19319	29000
22	103	103	28865	26000
23	103	122	26210	29000
24	103	122	28962	29000
25	103	122	31684	29000
26	112	122	28388	31000
27	103	107	26016	32000
28	103	107	25498	32000
29	103	107	30665	31000
30	112	122	28865	32000
31	103	122	25310	28000

32	103	103	24422	31000
33	103	122	28545	28000
34	103	122	32856	32000
35	103	103	27288	31000
36	111	112	27204	31000
37	111	112	25306	31000
38	103	112	33742	31000
39	103	112	25200	31000
40	103	112	24356	32000
41	103	112	31427	30000
42	103	112	25300	32000
43	103	112	31448	30000
44	103	112	33430	32000
45	112	112	30874	32000
46	103	111	29457	32000
47	103	111	26851	33000
48	103	111	32392	32000
49	103	111	29724	33000
50	103	111	25956	33000
51	103	112	20553	33000
52	103	112	20492	33000
53	103	112	24909	33000
54	103	112	29562	33000
55	103	112	23445	33000
56	103	112	18429	33000
57	103	112	29270	33000
58	103	112	32280	33000
59	112	112	27234	33000
60	103	103	23376	32000
61	103	112	31216	32000
62	103	103	27493	32000
63	103	103	35470	32000
64	103	103	31740	32000
65	103	103	33283	29000
66	103	103	32385	29000
67	112	112	31292	32000
68	103	112	30689	32000
69	103	103	36280	29000
70	103	112	29327	32000
71	103	112	24273	35000
72	103	112	26039	35000

73	103	112	31277	32000
74	103	112	21355	36000
75	103	112	26906	35000
76	103	112	25630	35000
77	112	112	24198	35000
78	103	112	22291	36000
79	103	112	34396	35000
80	112	112	25586	36000
81	103	112	30224	36000
82	103	112	25371	36000
83	103	112	27908	36000
84	103	112	22411	34000
85	103	103	28449	36000
86	107	112	25359	34000
87	103	112	28366	31000
88	112	112	28410	31000
89	103	103	31692	36000
90	103	112	31355	34000
91	112	112	28587	34000
92	103	112	31682	34000
93	103	112	37622	34000
94	103	112	29307	34000
95	103	112	26237	31000
96	107	112	35619	34000
97	111	112	31974	31000
98	103	112	35183	34000
99	103	112	31439	31000
100	103	112	27200	31000
101	112	112	23237	37000
102	103	112	26220	31000
103	103	112	30560	31000
104	103	112	22226	37000
105	103	112	32758	37000
106	103	112	26874	37000
107	103	112	33865	37000
108	103	112	29573	37000
109	103	112	32423	37000
110	103	112	26794	37000
111	112	112	29383	37000
112	103	112	27654	35000
113	103	112	35937	37000

114	103	112	29649	35000
115	103	112	22204	33000
116	103	112	21357	33000
117	103	112	23189	33000
118	103	112	30231	35000
119	103	112	28690	33000
120	103	112	32759	33000
121	112	112	28732	33000
122	103	112	24500	35000
123	103	112	23372	35000
124	103	112	26525	35000
125	103	112	20331	35000
126	103	112	24317	35000
127	103	112	25382	35000
128	103	112	30572	35000
129	111	112	35607	35000
130	112	112	27656	32000
131	103	112	30382	32000
132	103	112	26401	32000
133	103	112	28439	32000
134	103	112	33790	35000
135	103	112	24843	32000
136	103	112	28278	32000
137	103	112	23188	35000
138	103	112	33764	32000
139	103	112	30280	32000
140	103	112	21282	35000
141	103	112	22343	35000
142	112	112	25322	35000
143	103	112	23267	35000
144	103	112	30265	35000
145	103	112	31351	35000
146	103	112	29212	35000
147	103	112	26195	35000
148	103	112	29262	35000
149	112	112	32432	35000
150	103	112	29240	33000
151	103	112	28249	33000
152	103	112	27768	33000
153	103	103	32363	33000
154	103	112	36916	33000

155	112	112	27546	33000
156	103	112	24393	33000
157	103	112	30709	33000
158	107	112	27721	33000
159	103	112	31607	33000
160	112	112	28522	32000
161	103	112	24408	36000
162	103	112	31534	32000
163	103	112	26378	36000
164	103	112	32255	36000
165	103	112	29371	36000
166	103	112	28387	36000
167	103	112	27506	36000
168	103	112	32780	36000
169	103	112	31454	36000
170	103	112	29538	36000
171	103	112	30876	36000
172	112	112	37838	36000
173	103	112	32324	36000
174	112	112	21354	32000
175	103	112	28273	36000
176	103	112	28271	36000
177	103	112	28245	32000
178	103	112	29461	32000
179	103	112	21349	37000
180	112	112	27277	32000
181	103	112	22408	37000
182	103	112	24203	37000
183	103	112	29268	37000
184	103	112	30289	37000
185	103	112	26333	37000
186	103	112	29286	37000
187	103	112	25413	37000
188	103	112	34269	37000
189	103	112	27487	37000
190	112	112	32324	37000
191	103	112	30306	37000
192	103	112	29297	37000
193	103	112	31157	37000
194	103	112	29264	32000
195	103	112	25094	34000

196	112	112	27590	34000
197	103	112	34649	37000
198	103	112	31289	30000
199	103	112	35480	30000
200	103	112	25366	34000
201	103	112	27857	34000
202	103	112	33971	34000
203	103	112	32372	34000
204	103	112	24471	34000
205	112	112	30284	34000
206	103	112	23268	35000
207	103	112	25456	35000
208	103	112	29349	34000
209	103	112	32415	34000
210	103	112	25321	35000
211	112	112	22174	35000
212	103	112	25442	35000
213	103	112	28039	35000
214	103	112	29191	35000
215	103	112	32206	35000
216	103	112	27241	35000
217	103	112	26211	35000
218	112	112	29474	35000
219	103	112	33427	35000
220	103	112	24485	33000
221	103	112	33209	35000
222	103	112	31367	35000
223	103	112	30243	35000
224	103	112	28299	33000
225	112	112	23257	33000
226	103	112	26285	33000
227	103	112	25231	33000
228	103	112	29687	33000
229	103	112	33365	32000
230	103	112	29468	33000
231	103	112	32357	33000
232	103	112	32510	33000
233	103	112	31658	33000
234	103	112	28224	33000
235	103	112	28294	33000
236	112	112	31304	33000

237	107	112	25431	33000
238	103	112	28330	33000
239	103	112	29077	33000
240	103	112	29375	33000
241	103	112	26723	33000
242	103	112	25313	33000
243	103	112	32046	33000
244	112	112	26348	33000
245	103	112	25486	33000
246	103	112	32460	33000
247	103	112	27615	33000
248	103	112	30392	33000
249	107	112	30395	33000
250	103	112	22201	33000
251	112	112	25361	33000
252	103	112	29369	33000
253	103	112	29413	33000
254	103	112	31259	32000
255	103	112	22508	32000
256	103	112	29238	32000
257	112	112	30290	32000
258	103	112	27201	32000
259	103	112	30345	32000
260	103	112	29389	32000
261	103	112	31175	32000
262	111	112	28992	32000
263	103	112	34269	32000
264	103	112	26472	32000
265	103	112	25387	32000
266	112	112	33610	32000
267	103	112	29984	32000
268	103	112	34312	32000
269	103	112	31538	32000
270	103	112	27655	32000
271	103	112	27805	31000
272	111	112	26412	34000
273	111	112	27311	34000
274	103	112	29543	34000
275	112	112	28382	34000
276	103	112	27409	34000
277	103	112	27391	34000

278	103	112	26592	34000
279	103	112	28352	34000
280	103	112	27709	34000
281	103	112	30603	34000
282	103	112	34611	34000
283	103	112	34517	34000
284	103	112	25703	34000
285	112	112	31514	34000
286	103	112	28252	34000
287	103	112	29508	34000
288	103	112	26324	34000
289	103	112	27189	34000
290	103	112	25310	34000
291	112	112	28266	34000
292	103	112	27494	34000
293	103	112	26425	34000
294	103	112	25333	34000
295	112	112	27251	34000
296	103	112	26225	34000
297	103	112	32207	34000
298	103	112	27187	34000
299	103	112	32272	34000
300	103	112	25267	34000
301	103	112	29205	34000
302	112	112	31225	34000
303	103	112	28343	34000
304	103	112	26248	34000
305	103	112	27276	34000
306	103	112	32235	34000
307	103	112	27288	34000
308	103	112	35508	34000
309	103	112	31468	34000
310	103	112	31311	32000
311	112	112	27333	32000
312	103	112	29295	32000
313	103	112	31394	32000
314	103	112	29203	32000
315	103	112	31487	32000
316	103	112	28260	32000
317	103	112	26196	35000
318	103	112	28498	35000

319	103	112	33206	32000
320	103	112	27225	35000
321	103	112	31276	35000
322	112	112	28369	35000
323	103	112	34281	35000
324	103	112	25489	35000
325	103	112	32326	35000
326	103	112	27341	35000
327	103	112	31305	35000
328	103	112	28230	35000
329	103	112	29179	35000
330	112	112	28179	35000

APPENDIX C

CREATE MASK RESULTS, THRESHOLD 0.8

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	117	-1	25526	-1
1	107	-1	18609	-1
2	103	-1	22637	-1
3	111	-1	24246	-1
4	103	-1	20223	-1
5	103	-1	27223	-1
6	103	-1	30610	-1
7	107	-1	20172	-1
8	103	-1	26366	-1
9	112	-1	31459	-1
10	107	109	18521	26000
11	107	109	19161	26000
12	103	109	30226	26000
13	107	109	23571	26000
14	107	109	22268	26000
15	107	109	29324	26000
16	107	109	25283	26000
17	103	109	34175	26000
18	117	109	28407	25000
19	103	110	23223	23000
20	103	110	23491	23000
21	107	110	31544	23000
22	117	-1	28441	-1
23	103	106	24279	27000
24	103	-1	31361	-1
25	103	106	34379	22000
26	107	-1	30212	-1
27	103	106	30279	27000
28	117	110	29399	29000
29	107	109	28231	27000
30	103	109	27426	26000
31	103	108	30776	27000

32	103	109	25479	26000
33	117	109	28762	26000
34	103	104	21163	30000
35	107	108	26402	30000
36	103	104	26464	30000
37	107	108	29248	29000
38	103	109	28450	29000
39	107	109	30278	30000
40	103	108	26247	28000
41	117	109	33231	30000
42	103	108	29186	28000
43	103	108	24179	26000
44	117	107	32253	26000
45	107	107	35452	26000
46	107	107	33322	26000
47	103	108	21448	29000
48	103	106	29342	27000
49	103	108	29511	29000
50	103	108	29486	29000
51	103	108	27450	29000
52	103	108	25439	29000
53	103	110	23167	31000
54	103	110	28424	29000
55	117	107	32494	28000
56	107	110	27238	30000
57	103	104	23234	28000
58	117	107	32564	29000
59	103	104	28364	28000
60	107	107	31367	29000
61	103	107	33305	29000
62	103	107	31292	30000
63	103	-1	27263	-1
64	117	107	22260	27000
65	107	-1	29737	-1
66	114	106	27719	28000
67	103	107	31066	27000
68	107	107	25580	29000
69	112	107	24466	29000
70	103	107	23345	29000
71	103	109	28375	27000
72	103	109	29709	29000

73	112	108	26521	28000
74	103	108	29259	28000
75	103	108	26432	28000
76	103	108	25181	29000
77	107	108	32186	28000
78	112	109	26310	26000
79	103	107	26188	27000
80	103	109	31295	26000
81	103	107	28498	27000
82	103	107	29442	27000
83	103	106	27364	26000
84	103	107	30295	26000
85	103	105	24443	27000
86	112	107	30197	26000
87	107	107	33461	27000
88	112	105	23371	28000
89	103	106	26613	28000
90	107	107	32611	28000
91	103	107	32493	27000
92	103	106	30249	28000
93	103	105	25172	29000
94	107	105	27175	29000
95	103	105	30187	28000
96	112	105	31237	28000
97	112	107	22594	29000
98	103	106	28424	28000
99	103	107	24424	29000
100	103	107	28288	28000
101	107	107	29192	29000
102	103	106	35358	28000
103	103	107	29406	29000
104	103	107	28259	29000
105	103	107	28633	28000
106	114	107	33346	28000
107	112	106	24333	28000
108	103	106	27358	28000
109	107	106	26251	27000
110	103	106	30257	28000
111	107	106	25198	28000
112	103	106	34366	28000
113	103	106	30244	28000

114	103	106	28221	28000
115	107	106	27212	28000
116	107	106	34493	28000
117	112	107	26238	28000
118	103	107	28390	28000
119	103	107	24410	28000
120	103	107	32439	29000
121	103	107	31282	29000
122	107	107	30660	28000
123	107	107	28285	28000
124	112	106	25435	28000
125	107	106	28366	28000
126	103	107	31557	28000
127	103	106	23400	29000
128	112	106	29475	29000
129	107	106	32334	29000
130	103	106	25420	29000
131	103	106	33637	28000
132	103	106	31231	29000
133	107	107	27200	28000
134	107	107	24626	29000
135	103	106	30453	29000
136	103	107	30428	28000
137	107	107	24270	29000
138	107	107	26280	29000
139	103	107	25260	29000
140	103	107	30205	28000
141	103	107	31621	28000
142	103	107	36324	28000
143	107	106	31333	28000
144	112	106	28551	29000
145	103	107	34172	28000
146	107	106	35635	29000
147	111	105	23330	28000
148	103	105	30633	28000
149	103	105	27298	28000
150	103	105	26332	28000
151	111	106	34652	28000
152	103	106	35264	28000
153	112	106	24247	29000
154	107	106	23302	30000

155	103	106	25515	29000
156	107	105	30489	29000
157	103	107	31246	29000
158	112	107	28416	30000
159	103	106	29413	30000
160	103	107	25438	31000
161	103	107	29236	30000
162	103	107	28039	31000
163	107	107	26194	29000
164	107	107	31302	30000
165	103	107	30022	29000
166	107	107	24257	29000
167	103	107	18410	28000
168	107	107	24213	28000
169	103	107	26356	28000
170	112	107	24202	28000
171	103	107	21349	28000
172	103	107	24176	28000
173	112	106	26463	28000
174	103	106	25289	28000
175	103	106	32352	28000
176	107	107	31199	27000
177	103	106	24227	25000
178	103	106	26507	25000
179	107	106	31301	27000
180	103	106	31201	26000
181	112	106	30203	26000
182	103	106	37296	27000
183	107	106	24374	26000
184	103	106	32341	25000
185	103	106	32473	25000
186	103	106	30258	26000
187	103	106	31840	25000
188	112	107	30393	27000
189	103	107	27829	28000
190	107	106	34391	25000
191	107	107	30139	27000
192	103	106	35518	25000
193	103	106	30397	29000
194	103	105	29362	29000
195	103	105	30880	29000

196	103	106	36868	28000
197	103	106	23534	31000
198	103	106	23052	31000
199	107	106	27930	31000
200	107	106	25776	31000
201	112	106	24331	31000
202	103	106	22046	31000
203	103	105	29434	31000
204	103	105	26432	31000
205	103	105	26148	31000
206	103	105	28880	31000
207	103	105	22228	28000
208	112	105	25218	28000
209	107	105	29063	28000
210	111	105	32287	31000
211	107	105	31147	28000
212	111	105	28314	28000
213	103	105	23587	27000
214	103	105	22174	27000
215	103	105	22342	27000
216	103	105	28242	28000
217	103	106	27180	26000
218	103	107	25269	27000
219	103	106	29332	26000
220	107	107	27482	26000
221	103	106	31201	26000
222	103	105	36228	26000
223	103	106	27176	26000
224	107	106	30190	26000
225	103	107	34234	26000
226	117	106	34194	26000
227	107	106	24279	26000
228	103	106	22160	28000
229	103	106	31175	26000
230	103	106	27332	26000
231	103	106	26187	26000
232	103	106	25182	28000
233	112	105	27490	28000
234	103	106	26531	28000
235	103	106	31193	28000
236	103	105	36017	28000

237	103	106	26345	29000
238	103	106	29456	28000
239	103	106	31277	28000
240	103	106	26329	28000
241	107	106	30280	28000
242	103	106	31854	28000
243	103	106	22304	29000
244	112	106	29205	29000
245	103	106	28347	29000
246	103	106	32190	28000
247	112	106	29304	28000
248	103	105	28370	28000
249	103	106	30195	28000
250	103	106	35252	28000
251	103	106	34214	28000
252	103	104	29224	28000
253	103	105	25437	28000
254	112	105	27197	28000
255	103	104	33246	28000
256	107	105	28764	28000
257	103	105	22923	29000
258	103	105	26640	29000
259	103	105	32777	29000
260	103	105	31724	29000
261	103	106	35978	29000
262	103	105	31288	29000
263	112	106	27210	29000
264	103	106	33516	29000
265	107	106	33777	29000
266	103	106	38705	29000
267	111	106	31385	29000
268	103	105	20287	30000
269	103	105	20294	30000
270	103	105	24168	30000
271	103	105	27452	30000
272	103	106	32067	28000
273	103	105	27689	30000
274	112	106	24334	30000
275	103	106	23330	30000
276	103	105	27591	30000
277	103	105	34385	30000

278	103	105	27273	29000
279	103	106	29556	29000
280	103	106	31261	30000
281	107	105	27649	29000
282	107	105	31480	29000
283	103	106	22224	28000
284	103	106	28187	28000
285	103	106	27410	28000
286	103	106	26288	28000
287	103	105	30264	29000
288	112	105	25219	27000
289	103	105	29229	28000
290	103	105	28338	27000
291	103	105	32366	28000
292	103	105	32298	28000
293	103	105	28220	28000
294	103	105	31318	27000
295	107	105	29514	28000
296	103	105	35368	26000
297	107	105	30342	28000
298	112	105	25238	28000
299	103	105	22228	28000
300	103	105	28218	27000
301	103	105	25642	28000
302	103	105	27237	28000
303	103	105	25172	29000
304	103	105	30212	28000
305	103	105	26292	29000
306	103	105	31258	28000
307	112	105	31366	28000
308	103	105	30224	28000
309	103	105	32214	28000
310	103	105	34219	29000
311	107	105	30226	28000
312	103	105	16221	28000
313	103	105	28248	28000
314	103	105	27362	28000
315	103	105	29267	28000
316	112	105	32372	28000
317	103	105	28378	28000
318	107	105	27340	29000

319	103	105	31365	28000
320	103	105	33332	28000
321	103	105	32382	29000
322	107	105	25508	28000
323	112	105	30231	28000
324	103	105	30333	28000
325	103	105	28450	28000
326	103	105	32256	27000
327	103	105	36504	27000
328	112	105	23295	29000
329	103	105	27248	29000
330	103	105	32395	28000

APPENDIX D

DEFORM VOLUME RESULTS, THRESHOLD 1.2

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	957	-1	53345	-1
1	960	-1	315727	-1
2	957	-1	246518	-1
3	960	-1	254250	-1
4	963	-1	275801	-1
5	958	-1	224290	-1
6	959	-1	239276	-1
7	960	-1	170404	-1
8	958	-1	65085	-1
9	958	-1	61926	-1
10	1019	1152	67669	378000
11	1017	1152	71951	378000
12	958	1152	129203	378000
13	1017	1152	142909	378000
14	1017	1152	163286	378000
15	958	1152	94399	378000
16	1018	1152	97520	378000
17	1015	1152	105945	378000
18	1020	1152	104981	378000
19	1016	1152	161974	378000
20	961	1152	124347	378000
21	961	1152	110392	378000
22	958	1152	96513	378000
23	1026	1152	129248	378000
24	961	1152	104760	378000
25	1018	1152	153635	378000
26	1016	1152	152120	378000
27	966	1155	91022	378000
28	958	1155	103533	378000
29	962	1155	113464	378000
30	963	1155	128688	378000
31	1019	1155	126993	378000
32	1017	1155	79439	378000

33	958	1155	88834	378000
34	961	1155	92282	378000
35	960	1155	98411	378000
36	959	1155	71681	378000
37	1148	1155	70400	378000
38	962	1155	102937	378000
39	962	1155	69309	378000
40	1015	1155	68388	378000
41	1027	1155	71972	378000
42	1012	1155	69969	378000
43	1018	1155	92837	378000
44	1011	1155	69289	378000
45	1018	1155	96919	378000
46	1024	1155	78298	378000
47	1025	1155	81055	378000
48	1017	1155	88911	378000
49	1020	1155	73132	378000
50	1017	1155	81172	378000
51	1016	1155	68989	378000
52	1016	1155	74171	378000
53	1157	1155	78557	378000
54	1158	1155	118342	378000
55	1150	1155	103815	378000
56	1018	1155	129881	378000
57	1152	1155	95791	378000
58	1020	1155	167471	378000
59	1150	1155	127029	378000
60	1159	1155	135234	378000
61	1017	1155	89246	378000
62	1151	1155	103228	378000
63	1159	1155	117915	378000
64	1160	1155	134565	378000
65	1138	1155	73832	378000
66	1159	1155	106558	378000
67	1155	1155	81942	378000
68	1149	1155	90579	378000
69	1159	1155	72974	378000
70	1160	1155	76281	378000
71	1159	1155	86196	378000
72	1149	1155	108572	378000
73	1150	1155	113877	378000

74	1149	1155	96410	378000
75	1150	1155	77642	378000
76	1150	1155	79388	378000
77	1151	1155	88214	378000
78	1159	1155	285875	378000
79	1150	1155	181985	378000
80	1149	1155	261268	378000
81	1150	1155	153836	378000
82	1135	1155	135793	378000
83	1159	1155	90317	378000
84	1137	1155	97213	378000
85	1127	1155	150217	378000
86	1137	1155	109788	378000
87	1128	1155	133652	378000
88	1139	1155	110384	378000
89	1139	1155	99358	378000
90	1138	1155	104766	378000
91	1135	1155	107973	378000
92	1130	1155	122310	378000
93	1140	1155	90100	378000
94	1040	1155	102073	378000
95	1137	1155	83868	378000
96	1130	1155	129857	378000
97	1132	1155	78068	378000
98	1129	1155	122245	378000
99	1123	1155	119680	378000
100	1132	1155	117423	378000
101	1124	1155	120817	378000
102	1148	1155	80057	378000
103	1128	1155	76806	378000
104	1128	1155	86617	378000
105	1138	1155	84366	378000
106	1137	1155	84167	378000
107	1129	1155	81398	378000
108	1137	1155	96488	378000
109	1133	1155	105312	378000
110	1037	1155	135686	378000
111	1042	1155	123917	378000
112	1038	1155	111919	378000
113	1043	1155	136062	378000
114	1127	1155	136252	378000

115	1047	1155	152842	378000
116	1046	1155	150258	378000
117	1043	1155	194452	378000
118	1034	1155	184302	378000
119	1039	1155	90024	378000
120	1047	1155	121336	378000
121	1046	1155	267237	378000
122	1044	1155	126272	378000
123	1043	1155	72524	378000
124	1045	1155	111812	378000
125	1039	1222	84861	378000
126	1166	1222	90387	378000
127	1043	1222	81836	378000
128	1045	1222	88025	378000
129	1045	1222	94078	378000
130	1045	1222	93885	378000
131	1045	1222	71561	378000
132	1043	1222	67799	378000
133	1045	1222	74765	378000
134	1044	1222	77462	378000
135	1037	1222	360091	378000
136	1041	1222	178287	378000
137	1044	1222	533433	378000
138	1039	1222	575343	378000
139	1167	1222	196417	378000
140	1174	1222	161838	378000
141	1043	1222	174886	378000
142	1164	1222	117879	378000
143	1174	1222	160827	378000
144	1164	1222	135807	378000
145	1171	1222	150596	378000
146	1163	1222	104723	378000
147	1168	1222	126998	378000
148	1159	1222	195623	378000
149	1165	1222	193186	378000
150	1173	1222	198573	378000
151	1172	1222	119657	378000
152	1159	1222	140313	378000
153	1173	1222	134073	378000
154	1172	1222	86475	378000
155	1006	1222	62230	378000

156	1164	1222	105446	378000
157	1161	1222	90478	378000
158	1163	1222	84249	378000
159	1174	1222	78978	378000
160	1174	1222	87319	378000
161	1159	1222	96533	378000
162	1164	1222	81628	378000
163	1174	1222	97431	378000
164	1163	1222	70274	378000
165	1009	1222	107961	378000
166	1164	1222	693764	378000
167	1005	1222	176517	378000
168	1165	1222	162089	378000
169	1005	1222	98690	378000
170	1007	1222	60579	378000
171	1005	1222	119910	378000
172	1006	1222	87427	378000
173	1011	1222	110689	378000
174	1173	1222	94451	378000
175	1008	1222	102677	378000
176	1006	1222	93325	378000
177	1010	1222	105818	378000
178	1008	1222	84259	378000
179	1007	1222	80684	378000
180	1008	1222	111911	378000
181	1006	1222	93252	378000
182	1009	1222	110204	378000
183	1004	1222	72835	378000
184	1075	1222	70526	378000
185	1004	1222	67067	378000
186	1006	1222	70787	378000
187	1007	1222	80518	378000
188	1007	1222	99910	378000
189	1004	1222	76797	378000
190	1013	1222	64440	378000
191	1005	1222	74486	378000
192	1004	1222	74349	378000
193	1016	1222	69136	378000
194	1004	1222	73207	378000
195	1008	1222	97689	378000
196	1008	1222	113198	378000

197	1080	1222	87169	378000
198	1086	1222	76133	378000
199	1081	1222	116235	378000
200	1087	1222	104853	378000
201	1074	1222	105464	378000
202	1081	1222	79514	378000
203	1083	1222	99770	378000
204	1082	1222	73856	378000
205	1080	1222	78654	378000
206	1084	1222	95494	378000
207	1081	1222	91473	378000
208	1079	1222	90612	378000
209	1111	1222	68422	378000
210	1079	1222	85584	378000
211	1079	1222	76986	378000
212	1084	1222	76100	378000
213	1074	1222	83296	378000
214	1084	1222	106201	378000
215	1074	1222	96408	378000
216	1071	1222	115134	378000
217	1081	1222	81918	378000
218	1080	1222	106184	378000
219	1075	1222	73556	378000
220	1084	1222	83369	378000
221	1080	1222	91417	378000
222	1079	1222	81279	378000
223	1080	1222	78385	378000
224	1079	1222	66305	378000
225	1076	1222	103367	378000
226	1112	1222	104377	378000
227	1110	1222	75112	378000
228	1112	1222	76945	378000
229	1099	1222	106776	378000
230	1111	1222	292375	378000
231	1100	1222	345913	378000
232	1100	1222	459323	378000
233	1104	1222	297679	378000
234	1098	1222	177576	378000
235	1098	1222	100907	378000
236	1098	1222	203539	378000
237	1105	1222	137787	378000

238	1111	1222	134194	378000
239	1100	1222	162839	378000
240	1099	1222	236310	378000
241	1127	1222	112439	378000
242	1100	1222	120041	378000
243	1100	1222	147831	378000
244	1106	1222	112038	378000
245	1112	1222	84940	378000
246	1100	1222	116248	378000
247	1102	1222	92077	378000
248	1098	1222	85237	378000
249	1100	1222	112987	378000
250	1099	1222	142564	378000
251	1105	1222	101767	378000
252	1112	1222	75829	378000
253	1100	1222	76349	378000
254	1099	1222	71020	378000
255	1128	1222	73586	378000
256	1128	1222	68966	378000
257	1128	1222	75192	378000
258	1121	1222	76461	378000
259	1118	1222	69092	378000
260	1128	1222	75497	378000
261	1124	1222	80336	378000
262	1124	1222	69187	378000
263	1117	1222	79062	378000
264	1128	1222	110633	378000
265	1127	1222	146926	378000
266	1128	1222	185502	378000
267	1118	1222	127592	378000
268	1116	1222	132722	378000
269	1117	1222	135397	378000
270	1118	1222	127183	378000
271	1169	1222	66414	378000
272	1116	1222	78826	378000
273	1128	1222	99519	378000
274	1128	1222	70142	378000
275	1123	1222	115106	378000
276	1116	1222	138903	378000
277	1128	1222	73328	378000
278	1116	1222	75652	378000

279	1118	1222	144980	378000
280	1127	1222	77584	378000
281	1113	1222	83665	378000
282	1166	1222	68478	378000
283	1123	1222	77651	378000
284	1118	1222	78373	378000
285	1160	1222	82265	378000
286	1161	1222	71117	378000
287	1162	1222	112223	378000
288	1170	1222	99729	378000
289	1169	1222	81288	378000
290	1161	1222	103175	378000
291	1160	1222	642942	378000
292	1161	1222	121198	378000
293	1166	1222	117100	378000
294	1165	1222	96798	378000
295	1160	1222	135072	378000
296	1168	1222	89610	378000
297	1160	1222	134009	378000
298	1161	1222	129721	378000
299	1162	1222	128911	378000

APPENDIX E

DEFORM VOLUME RESULTS, THRESHOLD 1.0

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	957	-1	53345	-1
1	960	-1	315727	-1
2	957	-1	246518	-1
3	960	-1	254250	-1
4	963	-1	275801	-1
5	958	-1	224290	-1
6	959	-1	239276	-1
7	960	-1	170404	-1
8	958	-1	65085	-1
9	958	-1	61926	-1
10	1013	960	68474	315000
11	1016	960	68183	315000
12	1018	960	70612	315000
13	1017	960	81257	315000
14	1016	960	91235	315000
15	1018	960	95855	315000
16	1011	960	98414	315000
17	1016	960	83811	315000
18	1017	960	97148	315000
19	959	960	79314	315000
20	963	960	85054	315000
21	1018	960	131661	315000
22	961	960	93810	315000
23	960	960	83179	315000
24	1017	960	92381	315000
25	960	960	101389	315000
26	960	960	88935	315000
27	959	963	87810	315000
28	961	963	114536	315000
29	1021	963	117744	315000
30	1020	963	125707	315000
31	957	963	77305	315000

32	960	963	96115	315000
33	961	963	99986	315000
34	963	963	76539	315000
35	961	963	199775	315000
36	958	963	134021	315000
37	962	963	108861	315000
38	960	963	118352	315000
39	960	963	103192	315000
40	966	963	77391	315000
41	1149	963	71424	315000
42	1016	963	101230	315000
43	1017	963	286932	315000
44	1012	963	248578	315000
45	1019	963	252262	315000
46	1023	963	175488	315000
47	1013	963	222028	315000
48	1006	963	139251	315000
49	1012	963	226883	315000
50	1013	963	179395	315000
51	1019	963	130339	315000
52	1020	963	90027	315000
53	1019	963	150033	315000
54	1022	963	208500	315000
55	1160	963	108006	315000
56	1151	963	362010	315000
57	1017	963	122736	315000
58	1145	963	123310	315000
59	1149	963	100213	315000
60	1159	963	78931	315000
61	1149	963	77242	315000
62	1151	963	111918	315000
63	1150	963	74176	315000
64	1158	963	82970	315000
65	1130	963	88355	315000
66	1144	963	96327	315000
67	1146	963	83776	315000
68	1151	963	89800	315000
69	1156	963	75318	315000
70	1150	963	88495	315000
71	1155	963	82729	315000
72	1151	963	73253	315000

73	1159	963	92707	315000
74	1148	963	124787	315000
75	1158	963	196438	315000
76	1160	963	294719	315000
77	1150	963	351073	315000
78	1158	963	302409	315000
79	1160	963	369115	315000
80	1150	963	308182	315000
81	1151	963	251189	315000
82	1151	963	216395	315000
83	1145	963	145583	315000
84	1151	963	153638	315000
85	1128	963	123683	315000
86	1138	963	124524	315000
87	1125	963	96741	315000
88	1128	963	75760	315000
89	1129	963	93961	315000
90	1127	963	96262	315000
91	1128	963	87746	315000
92	1139	963	73903	315000
93	1039	963	73746	315000
94	1138	963	87435	315000
95	1124	963	109887	315000
96	1129	963	108726	315000
97	1129	963	107584	315000
98	1139	963	90734	315000
99	1136	963	88035	315000
100	1129	963	92594	315000
101	1138	963	92290	315000
102	1128	963	95768	315000
103	1134	963	96939	315000
104	1138	963	99589	315000
105	1129	963	100866	315000
106	1138	963	102030	315000
107	1137	963	102021	315000
108	1038	963	72603	315000
109	1139	963	273237	315000
110	1129	963	216304	315000
111	1124	963	239080	315000
112	1129	963	199767	315000
113	1044	963	179251	315000

114	1138	963	130473	315000
115	1038	963	183732	315000
116	1043	963	80968	315000
117	1036	963	95701	315000
118	1042	963	107820	315000
119	1046	963	108037	315000
120	1044	963	115126	315000
121	1046	963	121005	315000
122	1037	963	73705	315000
123	1128	1013	132570	315000
124	1044	1013	76347	315000
125	1173	1013	74217	315000
126	1038	1013	86704	315000
127	1044	1013	102562	315000
128	1046	1013	92540	315000
129	1036	1013	69036	315000
130	1043	1013	109431	315000
131	1045	1013	83622	315000
132	1037	1013	83167	315000
133	1043	1013	84248	315000
134	1046	1013	85144	315000
135	1046	1013	112509	315000
136	1046	1013	275133	315000
137	1043	1013	896690	315000
138	1044	1013	1044313	315000
139	1043	1013	958074	315000
140	1045	1013	842795	315000
141	1173	1013	228679	315000
142	1172	1013	238293	315000
143	1164	1013	167699	315000
144	1159	1013	158797	315000
145	1165	1013	133950	315000
146	1170	1013	157527	315000
147	1038	1013	107460	315000
148	1164	1013	160155	315000
149	1174	1013	100028	315000
150	1170	1013	102466	315000
151	1168	1016	68450	315000
152	1173	1016	103560	315000
153	1169	1016	115329	315000
154	1008	1016	86453	315000

155	1045	1016	98406	315000
156	1174	1016	115435	315000
157	1159	1016	86422	315000
158	1163	1016	74630	315000
159	1170	1016	107261	315000
160	1165	1016	105911	315000
161	1173	1016	151007	315000
162	1164	1016	172058	315000
163	1167	1016	203677	315000
164	1164	1016	193580	315000
165	1174	1016	191734	315000
166	1163	1016	256451	315000
167	1007	1016	133425	315000
168	1008	1016	122187	315000
169	1006	1016	197514	315000
170	1006	1018	174508	315000
171	1006	1018	196767	315000
172	1008	1018	151087	315000
173	1007	1018	167110	315000
174	1009	1018	140070	315000
175	1174	1018	160435	315000
176	1006	1018	76400	315000
177	1003	1018	84769	315000
178	1006	1018	105077	315000
179	1006	1018	131263	315000
180	1005	1018	134341	315000
181	1007	1018	108621	315000
182	1165	1018	143671	315000
183	1165	1018	118451	315000
184	1006	1018	127336	315000
185	1007	1018	114270	315000
186	1079	1018	193470	315000
187	1006	1018	88332	315000
188	1005	1018	126313	315000
189	996	1018	118598	315000
190	1009	1018	102503	315000
191	1007	1018	85116	315000
192	1008	1018	137624	315000
193	1007	1018	105347	315000
194	1006	1018	70562	315000
195	1005	1018	73397	315000

196	1006	1018	182886	315000
197	1004	1018	135384	315000
198	1003	1018	193491	315000
199	1083	1018	152474	315000
200	1087	1018	155712	315000
201	1084	1018	142784	315000
202	1079	1018	107640	315000
203	1079	1018	185499	315000
204	1082	1018	123107	315000
205	1077	1018	168238	315000
206	1079	1018	152083	315000
207	1081	1018	84743	315000
208	1080	1018	92439	315000
209	1083	1018	125709	315000
210	1082	1018	106285	315000
211	1081	1018	83317	315000
212	1111	1018	109712	315000
213	1080	1018	74494	315000
214	1079	1018	168819	315000
215	1076	1018	188512	315000
216	1082	1018	90383	315000
217	1074	1018	85011	315000
218	1081	1018	89122	315000
219	1074	1018	102221	315000
220	1085	1018	74346	315000
221	1080	1018	73327	315000
222	1081	1018	71737	315000
223	1081	1018	100166	315000
224	1071	1018	380959	315000
225	1081	1018	112123	315000
226	1101	1018	99983	315000
227	1105	1018	91269	315000
228	1101	1018	72437	315000
229	1099	1018	139355	315000
230	1103	1018	173188	315000
231	1100	1018	133690	315000
232	1111	1018	88653	315000
233	1099	1018	111518	315000
234	1111	1018	129105	315000
235	1100	1018	110380	315000
236	1111	1018	100978	315000

237	1107	1018	103725	315000
238	1101	1018	101795	315000
239	1099	1018	105934	315000
240	1105	1018	129695	315000
241	1116	1018	72042	315000
242	1107	1018	76942	315000
243	1102	1018	261625	315000
244	1102	1018	104080	315000
245	1096	1018	265565	315000
246	1110	1018	116521	315000
247	1101	1018	107425	315000
248	1110	1018	85173	315000
249	1103	1018	81290	315000
250	1112	1018	77171	315000
251	1097	1018	123235	315000
252	1104	1018	115215	315000
253	1111	1018	112522	315000
254	1100	1018	79325	315000
255	1128	1018	103757	315000
256	1126	1018	130758	315000
257	1114	1018	123556	315000
258	1123	1018	98454	315000
259	1121	1018	67093	315000
260	1117	1018	366022	315000
261	1117	1018	147366	315000
262	1128	1018	91443	315000
263	1128	1018	77062	315000
264	1128	1018	79509	315000
265	1121	1018	134860	315000
266	1117	1018	124566	315000
267	1117	1018	101953	315000
268	1117	1018	93619	315000
269	1164	1018	97422	315000
270	1128	1018	93955	315000
271	1127	1018	100451	315000
272	1118	1018	79689	315000
273	1114	1018	134328	315000
274	1128	1018	74345	315000
275	1119	1018	80633	315000
276	1129	1018	88098	315000
277	1128	1018	77276	315000

278	1129	1018	95100	315000
279	1116	1018	91195	315000
280	1117	1018	89254	315000
281	1128	1018	67663	315000
282	1129	1018	102172	315000
283	1160	1018	137174	315000
284	1119	1018	219574	315000
285	1159	1018	265803	315000
286	1155	1018	582092	315000
287	1160	1018	266088	315000
288	1155	1018	233675	315000
289	1156	1018	149347	315000
290	1157	1018	135489	315000
291	1162	1018	84543	315000
292	1171	1018	81546	315000
293	1170	1018	84687	315000
294	1161	1018	107533	315000
295	1168	1018	113449	315000
296	1161	1018	98821	315000
297	1156	1018	111738	315000
298	1225	1018	66154	315000
299	1160	1018	111197	315000

APPENDIX F

VABRA RESULTS, THRESHOLD 1.2

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	742	-1	90898	-1
1	1698	-1	3015518	-1
2	1718	-1	3038205	-1
3	1698	-1	2961156	-1
4	1713	-1	2974774	-1
5	1726	-1	3061159	-1
6	1721	-1	3149141	-1
7	1722	-1	3117045	-1
8	1713	-1	3401391	-1
9	1698	-1	3444166	-1
10	1361	2061	93013	3645000
11	1799	2061	2828759	3645000
12	1724	2061	2996176	3645000
13	1799	2061	2754938	3645000
14	1828	2061	2724308	3645000
15	1725	2061	3048015	3645000
16	1826	2061	2857711	3645000
17	1826	2061	3098704	3645000
18	1819	2061	2890244	3645000
19	1799	2061	2888102	3645000
20	1726	2061	3200971	3645000
21	1698	2061	3158477	3645000
22	1827	2061	3191348	3645000
23	1821	2061	3089065	3645000
24	1724	2061	3147243	3645000
25	1721	2061	3188205	3645000
26	1821	2061	3183042	3645000
27	1725	2061	3590547	3645000
28	1799	2061	3225536	3645000
29	1720	2061	3529108	3645000
30	1725	2061	3214378	3645000
31	1725	2061	3387681	3645000

32	1820	2061	2704315	3645000
33	1723	2061	3561741	3645000
34	1722	2061	3404454	3645000
35	1726	2061	3532389	3645000
36	1717	2061	3561711	3645000
37	821	2071	97568	3673000
38	1726	2061	3547859	3645000
39	1721	2061	3731385	3645000
40	1824	2061	3266773	3645000
41	1724	2061	5817889	3645000
42	1799	2061	2657782	3645000
43	1818	2071	2710516	3673000
44	1825	2071	2749649	3673000
45	1822	2071	2671039	3673000
46	1822	2061	3223980	3645000
47	1799	2071	2680509	3673000
48	1827	2071	2666103	3673000
49	1825	2071	2648703	3673000
50	1827	2071	2828751	3673000
51	1821	2071	3131585	3673000
52	1824	2071	3155168	3673000
53	1825	2071	2914700	3673000
54	2070	2071	2875463	3673000
55	2065	2071	2863435	3673000
56	2074	2071	2994445	3673000
57	1826	2071	3233553	3673000
58	1827	2071	3239492	3673000
59	2069	2071	2839398	3673000
60	2072	2071	2869689	3673000
61	2067	2071	2970052	3673000
62	1799	2071	3272478	3673000
63	2072	2071	2991704	3673000
64	2064	2071	2909518	3673000
65	2069	2071	3018064	3673000
66	822	2071	137558	3778000
67	2079	2071	2961021	3673000
68	2072	2071	2828390	3673000
69	2047	2071	2904176	3673000
70	2068	2071	3000639	3673000
71	2068	2071	3357823	3673000
72	2068	2071	3135646	3673000

73	2069	2071	2866989	3778000
74	2069	2071	2752811	3778000
75	2073	2071	2914676	3778000
76	2067	2071	2999071	3778000
77	2078	2071	2997615	3778000
78	2068	2071	3033800	3778000
79	2068	2071	3038956	3778000
80	2072	2071	3331402	3778000
81	2066	2071	3271252	3778000
82	2069	2071	2946690	3778000
83	2033	2071	3134277	3778000
84	2066	2071	3285592	3778000
85	2034	2071	2894572	3778000
86	2033	2071	3049249	3778000
87	2031	2071	3058051	3778000
88	2009	2071	3056016	3778000
89	2035	2071	2999520	3778000
90	2031	2071	3065217	3778000
91	2030	2071	3169835	3778000
92	2029	2071	3683392	3778000
93	2034	2071	3159987	3778000
94	2030	2071	3398458	3778000
95	1417	2071	113693	4081000
96	2009	2071	3298298	3778000
97	2032	2071	3012127	3778000
98	2028	2071	3241200	3778000
99	2035	2071	3520915	3778000
100	2034	2071	3549843	3778000
101	2034	2071	3290440	3778000
102	2033	2071	3395175	3778000
103	2031	2071	3720879	3778000
104	2036	2071	4212157	3778000
105	2034	2071	3089181	4081000
106	2009	2071	3022853	4081000
107	2026	2071	2820552	4081000
108	2028	2071	2957800	4081000
109	2029	2071	3085216	4081000
110	2035	2071	3113826	4081000
111	2033	2071	3071719	4132000
112	1870	2071	3005942	4132000
113	1868	2071	3113359	4132000

114	1870	2071	3077783	4132000
115	1871	2071	3110044	4132000
116	2009	2071	3227104	4132000
117	1871	2071	3134597	4132000
118	1870	2071	3137018	4132000
119	1866	2071	3179812	4132000
120	1868	2071	3276035	4132000
121	1870	2071	3406716	4132000
122	1869	2071	3465092	4132000
123	1844	2071	3551085	4132000
124	1872	2071	3544603	4132000
125	1872	2071	3047882	4132000
126	1868	2071	3680902	4132000
127	832	2071	189372	4132000
128	1844	2071	3177107	4132000
129	1871	2071	3034189	4132000
130	1844	2071	3074702	4132000
131	1871	2071	3076212	4132000
132	1869	2071	3010638	4132000
133	1866	2071	3149513	4132000
134	1863	2071	3138699	4132000
135	1864	2071	3199471	4132000
136	1870	2071	3009731	4132000
137	1844	2071	3109829	4132000
138	1844	2071	3111135	4132000
139	1870	2071	3171441	4132000
140	1866	2071	3620288	4132000
141	2104	2071	2974403	4132000
142	2097	2071	2910391	4132000
143	1868	2071	3492464	4132000
144	2072	2071	2942547	4132000
145	2095	2071	2980293	4132000
146	2096	2071	2906300	4132000
147	2094	2071	2947323	4132000
148	2090	2071	2942869	4132000
149	2072	2071	2891133	4132000
150	2098	2071	2894171	4132000
151	2094	2071	2864887	4132000
152	2072	2071	2878140	4132000
153	2104	2071	2955641	4132000
154	2072	2071	2954840	4132000

155	2091	2071	2911393	4132000
156	2089	2071	3008633	4132000
157	1243	2158	116248	4132000
158	2095	2071	3345546	4132000
159	2072	2071	3351816	4132000
160	2096	2071	2933836	4132000
161	2096	2071	3359167	4132000
162	2091	2071	2857991	4132000
163	2098	2071	2843289	4132000
164	2092	2071	3430412	4132000
165	2089	2071	3534368	4132000
166	2072	2071	3053523	4132000
167	2072	2071	3341652	4132000
168	2104	2071	3429637	4132000
169	1807	2158	2617157	4132000
170	1806	2158	2620456	4132000
171	2072	2158	3346645	4132000
172	1804	2158	2912272	4132000
173	1804	2158	2674203	4132000
174	1805	2158	2785747	4132000
175	1780	2158	3020157	4132000
176	1780	2158	2718255	4132000
177	2088	2158	3242760	4132000
178	1807	2158	2695922	4132000
179	1804	2158	2624177	4132000
180	1808	2158	2655397	4132000
181	1807	2158	2650868	4132000
182	1780	2158	3047003	4132000
183	1800	2158	2988313	4132000
184	1808	2158	2665828	4132000
185	1807	2158	2950123	4132000
186	1780	2158	2732617	4132000
187	1537	2158	132990	4132000
188	1808	2158	2738876	4132000
189	1808	2158	3147233	4132000
190	1799	2158	3021025	4132000
191	1801	2158	3071299	4132000
192	1801	2158	2823632	4132000
193	1806	2158	3075220	4132000
194	1809	2158	3007338	4132000
195	1807	2158	3260150	4132000

196	1801	2158	2652459	4132000
197	1807	2158	2966947	4132000
198	1802	2158	2667254	4132000
199	1798	2158	2646851	4132000
200	1936	2158	3001611	4132000
201	1939	2158	2747282	4132000
202	1941	2158	2699575	4132000
203	1913	2158	2768859	4132000
204	1939	2158	3043345	4132000
205	1913	2158	2634398	4132000
206	1941	2158	2689771	4132000
207	1913	2158	2665196	4132000
208	1941	2158	2682389	4132000
209	1941	2158	3181069	4132000
210	1937	2158	2689535	4132000
211	1937	2158	2788015	4132000
212	1027	2158	103094	4132000
213	1929	2158	2718362	4132000
214	1913	2158	2698430	4132000
215	1940	2158	2837598	4132000
216	1935	2158	2896647	4132000
217	1933	2158	2717511	4132000
218	1941	2158	2749781	4132000
219	1937	2158	2851333	4132000
220	1941	2158	2725834	4132000
221	1930	2158	2701017	4132000
222	1941	2158	3048406	4132000
223	1935	2158	3215092	4132000
224	1941	2158	2724692	4132000
225	1941	2158	3167362	4132000
226	1937	2158	2676385	4132000
227	1934	2158	2760520	4132000
228	1935	2158	2668287	4132000
229	1977	2158	2913980	4132000
230	1982	2158	2839839	4132000
231	1958	2158	2784128	4132000
232	1979	2158	2930369	4132000
233	1985	2158	2895530	4132000
234	1958	2158	2846112	4132000
235	1979	2193	2833741	4132000
236	1958	2158	3042156	4132000

237	1958	2193	2862990	4132000
238	1958	2193	2832163	4132000
239	1981	2193	2863798	4132000
240	1984	2193	2920252	4132000
241	1985	2193	2790986	4132000
242	1980	2193	2884618	4132000
243	1978	2193	2920891	4132000
244	1773	2193	181396	4132000
245	1980	2193	3182359	4132000
246	1958	2193	3224888	4132000
247	1980	2193	3306893	4132000
248	1958	2193	2896044	4132000
249	1984	2193	2870733	4132000
250	1981	2193	3336393	4132000
251	1983	2193	2889541	4132000
252	1978	2193	3385138	4132000
253	1983	2193	2987203	4132000
254	1985	2193	3547232	4132000
255	1973	2193	2923908	4132000
256	1980	2193	3306027	4132000
257	1985	2193	3037357	4132000
258	1990	2193	2748955	4132000
259	2010	2193	3079170	4132000
260	1990	2193	2823698	4132000
261	2012	2193	3120684	4132000
262	2015	2193	3216428	4132000
263	1990	2193	2735872	4132000
264	2017	2193	2792541	4132000
265	1990	2193	2598329	4132000
266	2010	2193	2649689	4132000
267	2010	2193	2736010	4132000
268	2017	2193	2710809	4132000
269	2008	2193	3137200	4132000
270	2012	2193	2567733	4132000
271	2014	2193	2750744	4132000
272	2015	2193	3123173	4132000
273	2017	2193	3166862	4132000
274	2013	2193	2608565	4132000
275	2015	2193	2702226	4132000
276	845	2193	237985	4132000
277	1990	2193	3024403	4132000

278	2014	2193	2680621	4132000
279	2010	2193	3085789	4132000
280	2016	2193	3215437	4132000
281	2010	2193	3163134	4132000
282	2009	2193	2872249	4132000
283	2014	2193	3162832	4132000
284	2014	2193	2744338	4132000
285	2091	2193	2945854	4132000
286	2013	2193	3236355	4132000
287	2013	2193	3094235	4132000
288	2066	2193	3000793	4132000
289	2092	2193	2959487	4132000
290	2066	2193	3408776	4132000
291	2066	2193	2986240	4132000
292	2090	2193	2971973	4132000
293	2093	2193	3020483	4132000
294	2090	2193	3006424	4132000
295	2088	2193	2886055	4132000
296	2087	2193	3027515	4132000
297	2090	2193	2985048	4132000
298	2066	2193	3493107	4132000
299	2092	2193	3045062	4132000

APPENDIX G

VABRA RESULTS, THRESHOLD 1.0

Experiment Number	Used Memory (MB)	Allocated Memory (MB)	Used Time (ms)	Allocated Time (ms)
0	742	-1	90898	-1
1	1698	-1	3015518	-1
2	1718	-1	3038205	-1
3	1698	-1	2961156	-1
4	1713	-1	2974774	-1
5	1726	-1	3061159	-1
6	1721	-1	3149141	-1
7	1722	-1	3117045	-1
8	1713	-1	3401391	-1
9	1698	-1	3444166	-1
10	1329	1718	106700	3038000
11	1799	1718	2688658	3038000
12	1815	1718	2744966	3038000
13	1821	1718	2732932	3038000
14	1823	1718	2811857	3038000
15	1826	1718	2709477	3038000
16	1827	1718	2765340	3038000
17	1824	1718	2721788	3038000
18	1826	1718	2716696	3038000
19	1725	1718	3102207	3038000
20	1826	1718	2833199	3038000
21	1717	1718	2986858	3038000
22	1724	1718	2997478	3038000
23	1725	1718	2993165	3038000
24	1820	1718	2837989	3038000
25	1724	1718	3261136	3038000
26	1724	1718	3104621	3038000
27	1823	1718	2944317	3038000
28	1726	1718	3171528	3038000
29	1827	1718	2956969	3038000
30	1725	1718	3145917	3038000
31	1725	1718	3120293	3038000

32	1724	1718	3120433	3038000
33	1726	1718	3172976	3038000
34	1725	1718	3122791	3038000
35	1725	1718	3502132	3038000
36	1719	1718	3585348	3038000
37	1726	1718	3530341	3038000
38	1725	1718	3591954	3038000
39	1725	1718	3622778	3038000
40	857	1726	124390	3061000
41	1724	1718	3823703	3038000
42	1816	1718	3100543	3038000
43	1799	1718	2826582	3038000
44	1826	1718	2763647	3038000
45	1799	1718	2732614	3038000
46	1826	1726	2750041	3061000
47	1799	1726	2706149	3061000
48	1821	1726	2743239	3061000
49	1820	1726	2855477	3061000
50	1799	1726	2728182	3061000
51	1825	1726	2727645	3061000
52	1827	1726	2686784	3061000
53	1822	1726	2854718	3061000
54	2066	1726	2820775	3061000
55	1818	1726	3170888	3061000
56	2070	1726	2890747	3061000
57	2074	1726	2881919	3061000
58	1826	1726	3286751	3061000
59	2069	1726	2917403	3061000
60	2047	1726	3327724	3061000
61	2073	1726	3305718	3061000
62	2065	1726	2937609	3061000
63	2047	1726	3310720	3061000
64	2074	1726	2877359	3061000
65	824	1726	102199	3149000
66	2067	1726	2920937	3061000
67	2072	1726	2876679	3061000
68	2066	1726	2902542	3061000
69	2047	1726	3203050	3061000
70	2073	1726	2930067	3061000
71	2069	1726	3348803	3061000
72	2072	1726	2935017	3061000

73	2074	1726	3200152	3061000
74	2065	1726	2875008	3149000
75	2047	1726	2893225	3149000
76	2069	1726	2883904	3149000
77	2074	1726	2880108	3149000
78	2072	1726	3004290	3149000
79	2047	1726	2855071	3149000
80	2073	1726	2959095	3149000
81	2066	1726	3319641	3149000
82	2071	1726	2977684	3149000
83	2070	1726	3003062	3149000
84	2067	1726	3248675	3149000
85	2035	1726	3065015	3149000
86	2034	1726	3072381	3149000
87	2026	1726	3260393	3149000
88	2032	1726	2887635	3149000
89	2032	1726	3128849	3149000
90	2030	1726	2969523	3149000
91	2032	1726	2907643	3149000
92	2033	1726	3172079	3149000
93	1424	1726	106395	3401000
94	2009	1726	3165118	3149000
95	2035	1726	3023415	3149000
96	2027	1726	3175510	3149000
97	2036	1726	3147003	3149000
98	2033	1726	3112911	3149000
99	2009	1726	3165348	3149000
100	2033	1726	3387553	3149000
101	2028	1726	3595452	3149000
102	2009	1726	3348116	3149000
103	2009	1726	2989058	3149000
104	2027	1726	3599416	3149000
105	2009	1726	3160591	3401000
106	2028	1726	3308187	3401000
107	1868	1726	3402510	3401000
108	2031	1726	3334030	3401000
109	2031	1726	2893589	3444000
110	2027	1726	2952134	3444000
111	2030	1726	2981691	3444000
112	2030	1726	3006021	3444000
113	1869	1726	3003363	3444000

114	1872	1726	3030713	3444000
115	2032	1726	3150577	3444000
116	1872	1726	3084717	3444000
117	1870	1726	3109260	3444000
118	1844	1726	3126959	3444000
119	1844	1726	3140171	3444000
120	1866	1726	3188078	3444000
121	1866	1726	3114174	3444000
122	1870	1726	3260621	3444000
123	2009	1726	3456111	3444000
124	1869	1726	3537465	3444000
125	833	1726	123386	3444000
126	1869	1726	2983373	3444000
127	1869	1726	3022994	3444000
128	1871	1726	3102244	3444000
129	1867	1726	3143992	3444000
130	1870	1726	3074530	3444000
131	1866	1726	3150188	3444000
132	1868	1726	3102013	3444000
133	1871	1726	3732076	3444000
134	1863	1726	3372983	3444000
135	1866	1726	3376105	3444000
136	1870	1726	4836173	3444000
137	1870	1726	4577861	3444000
138	1862	1726	3021676	3444000
139	1844	1726	3211726	3444000
140	1868	1726	3132927	3444000
141	2092	1726	2960309	3444000
142	2103	1726	3031290	3444000
143	2095	1726	2929928	3444000
144	2092	1726	2939297	3444000
145	2096	1726	2957796	3444000
146	2089	1726	3007131	3444000
147	2095	1726	3094145	3444000
148	1867	1726	3139276	3444000
149	2088	1726	2924874	3444000
150	2088	1726	3237545	3444000
151	2091	1726	3004100	3444000
152	2104	1726	3349802	3444000
153	2093	1726	3240225	3444000
154	746	1799	121356	3444000

155	1870	1726	3468629	3444000
156	2095	1726	2843824	3444000
157	2096	1726	2935173	3444000
158	2096	1726	3008080	3444000
159	2097	1726	3326922	3444000
160	2091	1726	3388897	3444000
161	2094	1726	3315896	3444000
162	2099	1726	3331938	3444000
163	2096	1726	3442600	3444000
164	2096	1726	2996186	3444000
165	2089	1726	4146953	3444000
166	2097	1726	3676721	3444000
167	2087	1726	2961351	3444000
168	1808	1799	2601435	3444000
169	1808	1799	2600990	3444000
170	1807	1799	2566629	3444000
171	1806	1799	2562250	3444000
172	1807	1799	2656744	3444000
173	1808	1799	2713016	3444000
174	1808	1799	2733007	3444000
175	1808	1799	2688126	3444000
176	2092	1799	2910249	3444000
177	1808	1799	2672374	3444000
178	1801	1799	2664334	3444000
179	1780	1799	2866147	3444000
180	1808	1799	2839881	3444000
181	1808	1799	2678247	3444000
182	2096	1799	3045374	3444000
183	1808	1799	2805148	3444000
184	2072	1799	3318017	3444000
185	2095	1799	3275139	3444000
186	1806	1799	2892374	3444000
187	1599	1815	149044	3444000
188	1807	1799	3053378	3444000
189	1780	1799	2819261	3444000
190	1807	1799	3038024	3444000
191	1795	1799	2986521	3444000
192	1799	1799	2909589	3444000
193	1802	1799	2935559	3444000
194	1808	1815	2678002	3444000
195	1800	1815	2853191	3444000

196	1798	1815	2805720	3444000
197	1805	1815	2683385	3444000
198	1803	1815	3046112	3444000
199	1808	1815	2742170	3444000
200	1780	1815	2634618	3444000
201	1913	1815	2757640	3444000
202	1939	1815	2712534	3444000
203	1939	1815	2714420	3444000
204	1941	1815	2748237	3444000
205	1936	1815	2739747	3444000
206	1939	1815	2714609	3444000
207	1941	1815	2714257	3444000
208	1940	1815	2831008	3444000
209	1931	1815	2845670	3444000
210	1937	1815	2758599	3444000
211	1941	1815	3043399	3444000
212	1936	1815	2775626	3444000
213	1744	1821	118848	3444000
214	1936	1815	2911892	3444000
215	1913	1815	2738791	3444000
216	1932	1815	2699788	3444000
217	1937	1815	2898418	3444000
218	1936	1815	2737630	3444000
219	1913	1815	2729686	3444000
220	1937	1815	2674793	3444000
221	1937	1821	2667453	3444000
222	1939	1821	2661042	3444000
223	1939	1815	3060117	3444000
224	1940	1821	3134730	3444000
225	1939	1821	2734607	3444000
226	1940	1821	2809926	3444000
227	1937	1821	2785449	3444000
228	1937	1821	3193257	3444000
229	1982	1821	3220963	3444000
230	1958	1821	2823044	3444000
231	1982	1821	2816307	3444000
232	1981	1821	2843600	3444000
233	1984	1821	2866698	3444000
234	1979	1821	3266054	3444000
235	1979	1821	3318927	3444000
236	1985	1821	2872066	3444000

237	1982	1821	2860596	3444000
238	1982	1821	2922510	3444000
239	1985	1821	2858214	3444000
240	1985	1821	2954178	3444000
241	1985	1821	2987482	3444000
242	1986	1821	3143990	3444000
243	1984	1821	2873217	3444000
244	1113	1823	113960	3444000
245	1978	1821	2926788	3444000
246	1979	1821	2903399	3444000
247	1985	1821	2921638	3444000
248	1984	1821	3222629	3444000
249	1983	1821	3392724	3444000
250	1984	1821	3452025	3444000
251	1985	1823	2884126	3444000
252	1984	1823	2890037	3444000
253	1981	1823	2865229	3444000
254	1977	1823	3117929	3444000
255	1984	1823	2907165	3444000
256	1982	1823	2924813	3444000
257	1978	1823	2931268	3444000
258	2014	1823	2711697	3444000
259	2011	1823	3065130	3444000
260	2015	1823	2770248	3444000
261	2017	1823	3170539	3444000
262	2016	1823	2722667	3444000
263	2011	1823	2606860	3444000
264	2009	1823	2705964	3444000
265	1990	1823	2703367	3444000
266	2010	1823	2737740	3444000
267	2017	1823	2701911	3444000
268	1990	1823	2829451	3444000
269	2017	1823	2721528	3444000
270	2013	1823	2735060	3444000
271	2015	1823	2738651	3444000
272	1266	1826	119861	3444000
273	2015	1823	3252311	3444000
274	2016	1823	2833194	3444000
275	1990	1823	2750166	3444000
276	2014	1823	3291466	3444000
277	2012	1823	3098238	3444000

278	2016	1823	2954495	3444000
279	1990	1823	3098685	3444000
280	1990	1823	2906765	3444000
281	2012	1823	3075541	3444000
282	2015	1826	2796204	3444000
283	2010	1823	3139239	3444000
284	2015	1826	2847824	3444000
285	2016	1826	3086851	3444000
286	2097	1826	3048038	3444000
287	2013	1826	3092942	3444000
288	2090	1826	2996082	3444000
289	2092	1826	3021274	3444000
290	2090	1826	2964232	3444000
291	2084	1826	3059836	3444000
292	2086	1826	3035787	3444000
293	2084	1826	2986384	3444000
294	2088	1826	2957369	3444000
295	2086	1826	3060217	3444000
296	2087	1826	3002115	3444000
297	2092	1826	3052100	3444000
298	2083	1826	3230704	3444000
299	2091	1826	3013917	3444000

REFERENCES

ACCRE, http://www.accre.vanderbilt.edu/

A. C. von Eschenbach, and K. Buetow, “Cancer Informatics Vision: caBIG,” Cancer Inform 2, 22-4 (2007).

The Apache Software Foundation. (2004). Welcome to Apache Axis2/Java. Retrieved April 12, 2011, from http://axis.apache.org/axis2/java/core/

B. Lucas, B. Landman, J. Prince, D. Pham. "MAPS: A Free Medical Image Processing Pipeline," Organization for Human Brain Mapping, Melbourne, Australia, (2008).

B. Lucas, J. Bogovic, A. Carass et al., “The Java Image Science Toolkit (JIST) for rapid prototyping and publishing of neuroimaging software,” Neuroinformatics, 8(1) 5-17 (2010).

I. Foster, C. Kesselman. “Globus: A Metacomputing Infrastructure Toolkit,” International Journal of Supercomputer Applications, 11(2), 115-128 (1997).

H. Li, D. Groep, L. Wolters. “An Evaluation of Learning and Heuristic Techniques for Application Run Time Predictions,” in Proceedings of 11th Annual Conference of the Advance School for Computing and Imaging (ASCI), 2005.

K. Covington, E. S. McCreedy, M. Chen et al., "Interfaces and Integration of Medical Image Analysis Frameworks: Challenges and Opportunities," Biomedical Science and Engineering Conference. Oak Ridge, TN, 1-4 (2010)I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

K. K. Kakazu, L. W. Cheung, and W. Lynne, “The Cancer Biomedical Informatics Grid (caBIG): pioneering an expansive network of information and tools for collaborative cancer research,” Hawaii Med J 63(9), 273-5 (2004).

L. H. Schwartz, P. Kijewski, K. Lynch *et al.*, “Review and interpretation of MR imaging studies with PACS: creating uniform series descriptors for radiologists and referring physicians,” AJR Am J Roentgenol 179(3), 575-7 (2002).

M. McAuliffe, F. Lalonde, D. McGarry, W. Gandler, K. Csaky, B. Trus. "Medical image processing, analysis and visualization in clinical research Proc IEEE Intl Symp on Computer-Based Medical Systems CBMS, 381-386 (2001).

Oracle Corportation. (2011). Guide to Scaling Web Databases with MySQL Cluster. Retrieved November 8, 2011, from http://mysql.com/why-mysql/white-papers/mysql_wp_scaling_web_databases.php

Oracle Corporation. (2011). Package java.sql. Retrieved November 8, 2011, from http://download.oracle.com/javase/6/docs/api/java/sql/package-summary.html

Oracle Corporation. (2011). Welcome to VirtualBox.org! Retrieved November 8, 2011, from https://www.virtualbox.org/

O. Sonmez, N. Yigitbasi, A. Iosup, and D. Epema. “Trace-based evaluation of job runtime and queue wait time predictions in grids,” in HPDC, 2009, pp. 111–120.

O. Sonmez, N. Yigitbasi, S. Abrishani, A. Iosup, and D. Epema. “Performance Analysis of Dynamic Workflow Scheduling in Multicluster Grids,” in HPDC, 2010, pp. 49-60.

Peter Tröger, Hrabri Rajic, Andreas Haas, and Piotr Domagalski. Standardization of an API for Distributed Resource Management Systems. In Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007), page 619-626, Rio de Janeiro, Brazil, May 2007.

S. Pieper, B. Lorensen, W. Schroeder, R. Kikinis. “The NA-MIC Kit: ITK, VTK, Pipelines, Grids and 3D Slicer as An Open Platform for the Medical Image Computing Community Proc IEEE Intl Symp on Biomedical Imaging ISBI, 698-701 (2006).

S. Peiper, M. Halle, R. Kikinis. “3D Slicer,” Proc IEEE Intl Symp on Biomedical Imaging ISBI 1, 632-5 (2004).

T. Fillicelli, “Future of PACS: advanced integration with RIS and workflow management,” Radiol Manage, 23(1) 12-3 (2001).

T. Yoshinobu, K. Abe, Y. Sasaki *et al.*, “Data Management Solution for Large-Volume Computed Tomography in an Existing Picture Archiving and Communication System (PACS),” J Digit Imaging, (2009).

W. Wolfson, “caBIG: seeking cancer cures by bits and bytes,” Chem Biol, 15(6) 521-2 (2008).