IJDC | *General Article*

Leveraging High Performance Computing for Managing Large and Evolving Data Collections

Ritu Arora
Texas Advanced Computing Center
University of Texas at Austin

Maria Esteva
Texas Advanced Computing Center
University of Texas at Austin

Jessica Trelogan
Institute of Classical Archaeology
University of Texas at Austin

Abstract

The process of developing a digital collection in the context of a research project often involves a pipeline pattern during which data growth, data types, and data authenticity need to be assessed iteratively in relation to the different research steps and in the interest of archiving. Throughout a project's lifecycle curators organize newly generated data while cleaning and integrating legacy data when it exists, and deciding what data will be preserved for the long term. Although these actions should be part of a well-oiled data management workflow, there are practical challenges in doing so if the collection is very large and heterogeneous, or is accessed by several researchers contemporaneously. There is a need for data management solutions that can help curators with efficient and on-demand analyses of their collection so that they remain well-informed about its evolving characteristics. In this paper, we describe our efforts towards developing a workflow to leverage open science High Performance Computing (HPC) resources for routinely and efficiently conducting data management tasks on large collections. We demonstrate that HPC resources and techniques can significantly reduce the time for accomplishing critical data management tasks, and enable a dynamic archiving throughout the research process. We use a large archaeological data collection with a long and complex formation history as our test case. We share our experiences in adopting open science HPC resources for large-scale data management, which entails understanding usage of the open source HPC environment and training users. These experiences can be generalized to meet the needs of other data curators working with large collections.

Received 13 January 2014 | *Accepted* 26 February 2014

Correspondence should be addressed to Ritu Arora, Texas Advanced Computing Center, 10100, Burnet Road, J.J. Pickle Research Campus, Austin, Texas 78758. Email: rauta@tacc.utexas.edu

An earlier version of this paper was presented at the 9th International Digital Curation Conference.

The *International Journal of Digital Curation* is an international journal committed to scholarly excellence and dedicated to the advancement of digital curation across a wide range of sectors. The IJDC is published by the University of Edinburgh on behalf of the Digital Curation Centre. ISSN: 1746-8256. URL: http://www.ijdc.net/

http://dx.doi.org/10.2218/ijdc.v9i2.331
DOI: 10.2218/ijdc.v9i2.331

Introduction

The curators at the Institute of Classical Archaeology (ICA) at the University of Texas at Austin needed resources for managing an evolving data collection (~4.3 TB in size at the time of writing) efficiently and frequently. Besides being large in size, this data collection is in a semi-disorganized state, has a deeply-nested structure, and is constantly changing, such that the data continuously transitions from active research to publication and archiving phases. Only in the past four years, with a centralized infrastructure and a practical plan for data curation, has it been possible to manage new incoming data consistently (Esteva et al., 2010), still leaving a very large portion (~3 TB) of legacy data in need of organization and integration. Routine data management tasks such as finding records, identifying data types and production dates, sorting through multiple copies, culling corrupted and redundant files, and reorganizing data have been conducted manually, placing a significant burden on research staff.

To conduct these tasks more efficiently, the team started experimenting with powerful data analysis methods that exploit collection-level metadata related to file system, file formats, file sizes, and checksums. As a result of the analyses, the team can dynamically decide what files to send to the archive instance of the collection as they go about and curating the data. The metadata were extracted via the open source application DROID[1] and custom-written scripts, but it became clear that with this constantly evolving collection, metadata needed to be extracted regularly to capture the changes and to iterate through the data curation pipeline. The metadata extraction step for such a large collection proved to be a serious bottleneck, taking a minimum of two days for DROID to run on the server where the collection is stored. In order to support more frequent metadata extraction with a short turn-around time, the idea of using the open science HPC resources at the Texas Advanced Computing Center (TACC) looked attractive. However, the curators at ICA did not have prior experience in using HPC resources.

The ICA staff applied for, and obtained, an ECSS-supported allocation through XSEDE (Charge No. TG-HUM130001), and with the help of HPC experts at TACC, developed familiarity with the HPC environment. Together they developed a metadata extraction workflow that can be used by data curators on HPC resources with minimal Linux training. The workflow is interactive so that the data curators can conduct and verify the different steps when needed. The development of the workflow involved:

1. Porting the software to an open science HPC platform;
2. Transferring and synchronizing terabytes of data over the network to HPC resources at TACC for analyses and long term storage; and
3. Developing scripts for simultaneously running multiple copies of the software on different portions of the collection, also known as parallelization.

In this paper, we present an overview of the test collection, provide an introduction to HPC, and discuss the methods used in parallelizing the metadata extraction.

[1] DROID v.6.1.3: http://www.nationalarchives.gov.uk/information-management/projects-and-work/droid.htm

A Complex Archaeological Data Collection

The ICA test collection consists of approximately one million files representing 40 years of research activities in Italy and Ukraine. Typical of a large archaeological archive touched by many generations of specialists, it contains everything from scanned photographs, drawings and field notes, to GIS datasets, 3D visualizations, and complex databases. Due to the multidisciplinary nature and long history of ICA's research, these data have accumulated throughout many generations of scholars and students from various institutions and disciplines, each with its own research needs, technologies, and record keeping systems. As such, it has a rich history and complex layers of accumulation, a concept that resonates with the archaeologists dealing with its curation. Over time, efforts to consolidate, organize and document the digital collection had been piecemeal and dictated by specific research questions and technological limitations. The result was a disorganized collection without overarching consistency in the conventions of file naming, metadata, or organization.

Data Management Efforts

As ICA's focus has recently shifted away from new fieldwork to publication, the state of the data was making it difficult for researchers to efficiently retrieve and assimilate digital resources for study, archiving, and dissemination. As a first step, data previously distributed over several hard drives and detached storage devices were consolidated on a shared server administered by the Liberal Arts Information Technology Services (LAITS)[2]. The ICA team can now access the full collection and work more collaboratively than before with new incoming data, albeit with some inevitable trade-offs in access speed and loss of full control of the dataset.

Together with the team from TACC, a thorough assessment of the collection was performed, focusing on data from two archaeological excavations; one mostly born-digital excavation from Chersonesos in Ukraine, the other digitized more recently from paper archives of an early excavation at Pantanello in southern Italy. These two projects resulted in a semi-automated record keeping and metadata system for data archiving, but a large portion (~3 TB) of the disorganized legacy collection remains to be inspected, documented and archived. To approach this work, a visual analytics tool was used to assess the collection's structure and data types, and to prioritize areas for immediate attention (Esteva et al., 2013). Also, an Entity Resolution (ER) algorithm was developed to help identify and reduce data redundancy (Xu et al., 2013). Both tools use file format, file system and checksums of the entire collection to render results that aid making decisions about what data to reorganize, delete, archive, and make public.

Using these analysis tools, the team was able to detect large portions of corrupted, redundant and useful data, and considerably improved the state of the collection. However, to continue cleansing the data, the curators needed to improve the efficiency of the metadata extraction step. Because of the current flurry of research and data-consolidation activities at ICA, once a metadata and checksum snapshot is complete, it quickly becomes obsolete. Just extracting a snapshot of the current collection was taking too long, placing too much load on the shared server, and becoming a bottleneck in the curation process. We decided to implement a semi-automated metadata extraction workflow using the HPC resources at TACC such that it could be conveniently run by curators with basic training on Linux and on HPC user environments.

[2] Liberal Arts Information Technology Services: http://www.utexas.edu/cola/laits

HPC for Data Management

During a research project's lifecycle, functions such as data analysis, curation, archiving and access may be carried out simultaneously. These functions involve diverse technologies and require computational power relative to the size and complexity of the data collection. For large, complex and evolving data collections like ICA's, data management tasks such as extracting metadata or calculating checksums quickly become effort- and resource-intensive. Thus, repeating these tasks at the desired frequency and speed becomes challenging in a non-scalable desktop computing environment. The quest for bigger computing and storage resources leads to HPC platforms and solutions. However, using these platforms, which are mostly Linux-based, might be initially challenging for data curators without prior experience, as was the case with the curators at ICA. Through their collaboration with TACC, the ICA staff received HPC user environment training and consultancy in the development of a practical workflow for ongoing curation. They were first trained to install the required software, run scripts for data transfers, and run the scripts for metadata extraction on an HPC platform. During the workflow development phase, the curators were involved in testing when needed. The collaboration prepared them to work independently in an HPC environment, and the experiences gained in the process may be valuable for other projects with similar data management needs.

High Performance Computing: A High Level Overview

At a very coarse-grain level, HPC can be defined as computing carried out on a cluster of computational resources in order to reduce the overall time-to-results while solving large computational and analytical problems. The main principle involved in HPC is parallel processing, in which any given task and/or portion of data is divided among multiple compute-nodes that then work simultaneously to arrive at a given solution, hence reducing the overall time-to-results (Wilkinson and Allen, 2005). Each computer works on a piece of the big problem instead of one computer solving the entire problem.

Each computer in a cluster has processors, memory, storage and operating system associated with it. The individual computers forming a cluster are connected with each other through a high speed network and a software layer is required to make the individual computers in a cluster communicate with each other. The communication between the computers in a cluster is needed so that they can exchange their local results over the network while working on different pieces of a big problem. The local results can then be coalesced to accomplish a global result. Through this divide-and-conquer approach, known as parallelization, the overall time-to-results can be reduced for most computational or analytical problems. It should be noted that there are some processes or computational problems that are inherently serial and hence might not be amenable for parallelization. However, even such processes can benefit from the large memory and high-end processing elements available on top class HPC platforms.

Extreme Science and Engineering Discovery Environment (XSEDE)[3] and Partnership for Advanced Computing in Europe (PRACE)[4] are major research infrastructures providing access to high-end HPC resources and services through a peer-reviewed process in the United States and Europe respectively. Data curators in the United States and Europe can leverage these resources for their HPC needs without any

[3] Extreme Science and Engineering Discovery Environment: https://www.xsede.org

[4] Partnership for Advanced Computing in Europe (PRACE): http://www.prace-ri.eu/

direct cost to them. In addition, a large number of academic and government institutions have been making investments to establish their own private HPC infrastructure that is locally managed and shared within their campuses.

Most of the shared HPC resources run on the Linux Operating System and can be accessed remotely through a secure shell client. Therefore, if data curators are not familiar with Linux, they would first need basic training in this area. The open science HPC resources are simultaneously shared by multiple users and have a batch-processing environment with which the data curators would also need to gain familiarity.

It should be noted that due to the shared nature of the open science HPC resources, each user of the resource has a limit on the number of files and amount of the storage space on a given resource. Therefore, they might have to move their data between storage and computational resources as needed to comply with their account's quota before and after their computational job is completed.

The open science HPC platforms have a life span that is mainly determined by the amount of funding available to cover the cost of setting up the infrastructure (hardware resources and physical space), system operation, and maintenance. Besides the budgetary constraints, advancement in the area of computer architecture and end user needs for higher computational power often drive the replacement of old HPC platforms with new ones. Therefore, when working in an open science HPC environment, the data curators would also need to be prepared to participate in data transfer and account migration to new HPC platforms when needed, including updates of any custom-installed software. One also needs to be mindful of hardware and software compatibility.

These large HPC platforms undergo maintenance periodically. During the maintenance period, the HPC platforms are not accessible to their end users. Hence, data curators would need to plan for their data management activities taking the maintenance period into consideration. In addition to planned maintenance periods, there could be sporadic episodes during which an HPC platform could be unavailable. Hence, for critical tasks needing perpetual availability of data, curators should consider maintaining a duplicate copy of the dataset on a different (secondary) platform to increase the likelihood of data availability in the event the primary HPC platform is unavailable.

Solution Strategy

The goal of the work presented in this paper was to reduce the overall time spent in the metadata extraction for the test collection using parallel processing on the Stampede[5] supercomputer, an HPC resource at TACC. Stampede has more than 6400 compute-nodes, each outfitted with two Intel Xeon E5 (Sandy Bridge) processors and an Intel Xeon Phi Coprocessor (MIC Architecture). Each Sandy Bridge processor on a compute-node has eight cores, and hence in total 16 cores are available on each compute node. The frequency of the core is 2.7 GHz and has a theoretical peak performance of 21.6 GFLOPS/core or 346 GFLOPS/compute-node. Each compute-node contains at least 32GB of memory (2GB/core).

Setting up the semi-automated workflow for data curators at ICA to run on Stampede involved:

1. Data transfer from the LAITS server to TACC's computational and storage resources;

[5] Stampede: http://www.tacc.utexas.edu/stampede/

2. Installing DROID on Stampede;
3. Writing Linux scripts for running DROID in parallel, checksum calculation, load-balancing, etc.

In order to do parallel processing of the ICA collection, multiple instances of DROID and scripts for checksum calculation were run simultaneously on multiple compute nodes. It should be noted that only one installation of DROID was required for this purpose. We did not make any changes to the DROID code in order to achieve the goal of parallelization. Instead, we submitted a batch of DROID commands all at once to a set of compute nodes and provided different subdirectories as the parameters to these DROID commands. The results of all DROID runs were coalesced in the end to form a combined output file with the desired metadata. The process of combining the individual files into the final output file was carried out using existing DROID commands. In order to make the order of the files and directories the same in the output as both the serial and parallel runs, we sorted the output files in lexicographic order.

We checked the output of the serial run of DROID with the parallel run and did not find any loss of precision in metadata extraction. The process of running DROID in parallel is shown diagrammatically in Figure 1.

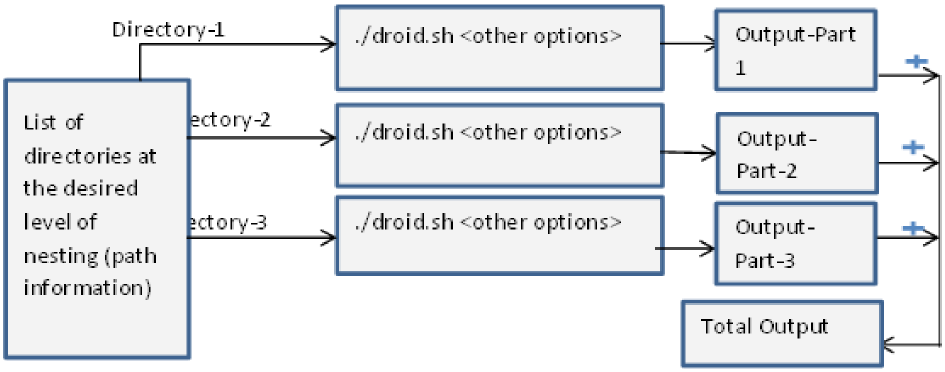
Directory-1
List of directories at the desired level of nesting (path information)
ectory-2
ectory-3
./droid.sh <other options>
./droid.sh <other options>
./droid.sh <other options>
Output-Part 1
Output-Part-2
Output-Part-3
+
+
+
Total Output

Figure 1. Parallel execution of DROID.

Because this collection has a deeply nested hierarchical structure, doing data-partitioning for multiple parallel runs of DROID and the checksum script was not straightforward. We conducted experiments to determine a strategy for optimally distributing the portions of the data collection to multiple compute nodes so that one compute node does not have larger workload to manage with respect to other nodes. This is known as load-balancing. We selected a very coarse-grain data distribution scheme in which a list of subdirectories up to the second level of nesting was selected. Each DROID instance got a subdirectory to work with recursively. A high-level view of part of the directory tree is shown in Figure 2.

For fine-grained load-balancing, all the information related to the directory tree for the data collection, including the contents of the directory and the sizes of individual files, is required. This can be done using Linux commands like "`du -sh`". However, running this command takes about the same amount of time as running DROID serially. Hence, the purpose of parallelization is defeated if too much time goes into analysing the directory tree for load-balancing purposes. For the entire ICA data collection, which has now grown to 4.3 TB, it took approximately six hours to extract the metadata using a single instance of DROID – or in running DROID serially – on Stampede.

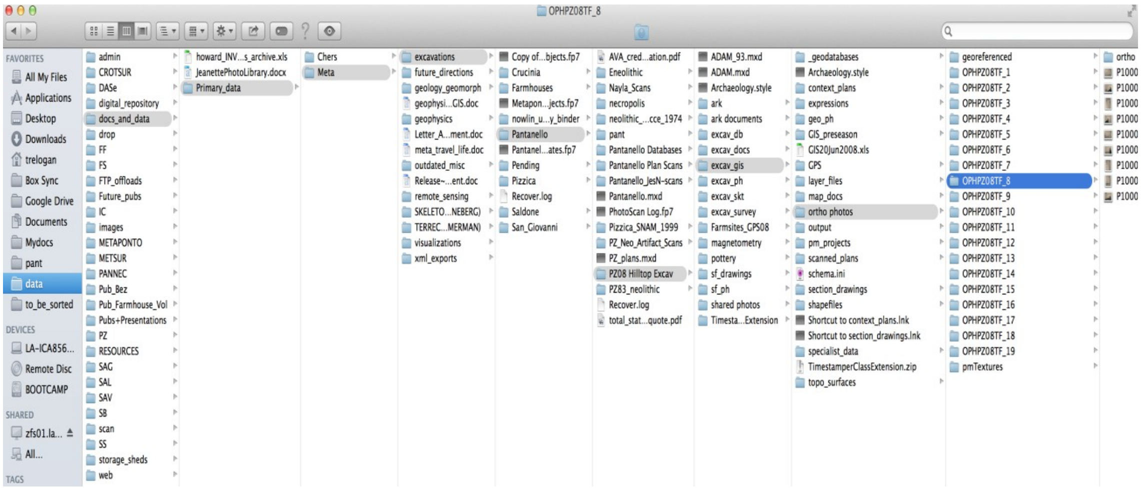

Figure 2. Partial directory tree showing multiple levels of nesting.

The parallel run with two instances of DROID, each working on one subdirectory of the parent directory of the data collection, took about four hours to complete. Figure 3 shows the comparison of the time involved in running multiple DROID instances with different levels and numbers of subdirectories. It should be noted here that the level of nesting of the directory hierarchy, along with the number of files in each directory, has an impact on the overall runtime of the DROID jobs.

It is clear (from Figure 3) that in our test case, running 31 instances of DROID (as there are 31 directories at the second level of nesting) led to the shortest time-to-result with the optimal load-balancing scheme – one hour and fifteen minutes in total. The rate-determining step, the one that took longest to complete, was related to one of the subdirectories of a directory at level-1 of nesting-hierarchy. Since each compute-node that was involved in the parallel runs of DROID had 32 GB of memory only, we ran only one instance of DROID on this compute-node. It is possible to run up to 16 instances of DROID on each compute node of Stampede, as there are 16 cores on each node. Each instance of DROID could be running on one core of the compute-node. However, the size of the directory associated with each DROID instance determines the total number of instances of DROID that can be launched on a node. This is because all the instances of the DROID software running on a node will have to use the shared memory available on the compute-node, which is 32 GB in case of Stampede. Therefore, to prevent the computational job from crashing due to memory-starvation on a node, it is important to distribute the parallel runs over multiple compute nodes such that each DROID instance has enough memory available to it.

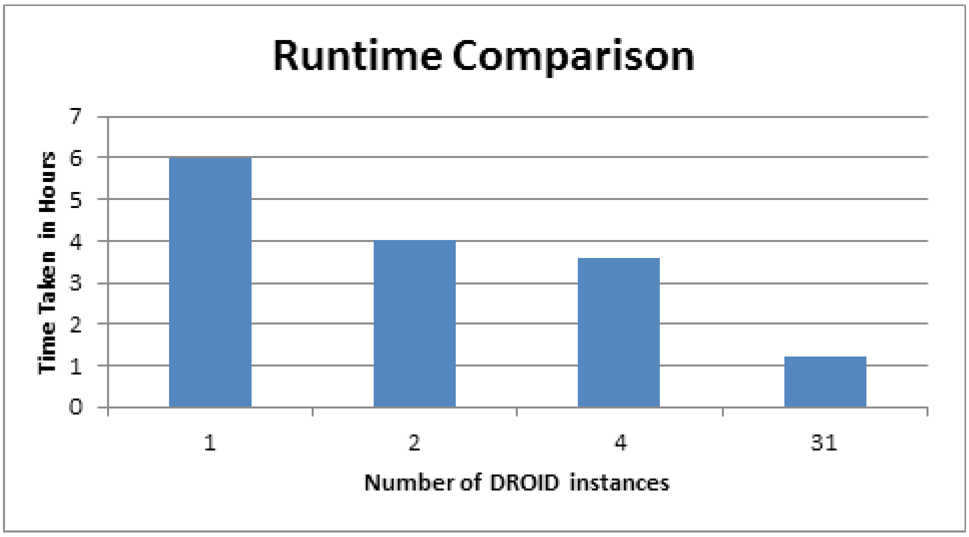

Figure 3. Comparison of time taken to complete metadata extraction with different number of DROID instances.

Challenges in Adopting HPC

The data transfer from the LAITS server to Stampede was completed in multiple attempts due to issues related to network connectivity, security and system down-time during maintenance. The initial transfer took approximately 28 hours using the `rsync` utility that provides fast incremental data transfers and the subsequent syncing of only changed files took insignificant amount of time.

We also ran into some challenges while using DROID in the Linux environment on Stampede. We had to make sure that no two instances of DROID are working on directories that share a parent-child relationship. We also observed that DROID uses the system time in milliseconds for generating a unique identification number. This unique identification number (or timestamp) is used in creating database profiles. However, when running multiple DROID instances simultaneously, it was common to see that more than one DROID instances were trying to create profiles with the same identification number and eventually crashing due to the collision with each other. As a workaround for this problem, we pipelined the DROID runs so that no two DROID instance start their processing at the same time.

Normally, the checksum calculation is part of the metadata extraction process, but the command-line version of DROID that we used did not support the same. Hence, we had to write our own Linux script for the checksum calculation and run it separately from DROID for extracting rest of the metadata.

The performance of the DROID tool on Stampede seems to depend upon the overall load on the file system (which manages the files on the storage hardware) that houses the data collection and the rate of the network traffic. If an `rsync` process is running simultaneous to a DROID process, then in such scenarios we observed that the performance of DROID is negatively impacted. Moreover DROID is written in Java programming language and does not support distributed memory and shared memory parallelization inherently.

Generalizing the Test Case

For ICA's data collection, the workflow implemented on TACC's HPC resources consisted of:

1. Transferring all the data from remote storage to compute resources;
2. Extracting metadata, calculating checksums, and outputting the results as a *.csv file;
3. Syncing/updating the collection so that it reflects how it has changed as data are added, reorganized and cleaned; and
4. Repeating the process from Step 2.

To generalize this test case, a user will have to consider issues related to their particular collection, such as size, structure and location in relation to the available HPC infrastructure and the networking speed between computational and data storage resources to assure efficient large data transfers.

Supercomputing centres have high speed connections between the computational and the data storage resources that they maintain, as well as with the outside world. In addition, open science national HPC resources are connected through powerful academic networks and have multiple protocols and technologies for efficient data transfer. However, for remote users working outside of powerful networks or without access to efficient data transfer technologies, transferring large amounts of data can become a significant bottleneck in their projects. In our model, the collection is stored in a remote location, albeit within the University of Texas at Austin network.

The advent of data intensive research is directly impacting the architecture and configuration of HPC platforms to avoid time spent in transferring data from storage to compute nodes and to speed up computational processing times. The latest HPC platforms are designed and deployed to include tiered storage that enables fast reading and writing of data, and executes millions of input/output operations per second (TACC, 2013). Thus, they combine data management and storage with computational analysis functions. Therefore, curators would need to consider the features of the available hardware infrastructure and their software for selecting a parallelization strategy.

Curators Training and HPC Adoption

In order to make the metadata extraction workflow an integral part of the test collection's data management activities, the ICA team needed to learn how to perform all its steps independently. As the curators were mostly unfamiliar with working in an HPC environment, initial training on the resources at TACC along with basic Linux commands for dealing with data transfers, file permissions, and running of scripts was required. Although the learning curve was somewhat steep for those with no prior Linux experience, it required about two days of practice to become proficient at copying and syncing the data collection, basic trouble-shooting and the running of batch scripts. After an initial one-on-one training session, and some self-paced online Linux tutorials, the curators were ready to transfer the data collection from a remote server to the TACC system. Some further instructions and correspondent scripts were given for running DROID serially and in parallel, and both tasks were accomplished successfully.

Most of the instructions from the trial runs of these methods came in the form of cookbook style recipes. These are easy enough to follow and can be deconstructed to understand what each parameter entails, but if errors are generated during any steps of the routines, it is difficult for novice users to troubleshoot. A ticket system is in place for questions, but further training would be needed to give the users enough confidence to ask for help in a meaningful way. Further training to obtain a deeper understanding of the systems architecture and the use of advanced Linux commands and batch scripting would go a long way toward ensuring the adoption of these resources by non-traditional HPC users.

Conclusions and Future Work

As data collections grow in size (e.g., 4 TB and above), routine data management tasks, such as extracting metadata, calculating checksums and allowing dynamic archiving, are difficult to perform in a desktop computing environments. However, with the massive growth in the size of data, data management is a data-intensive computing problem now. Implementation of data management workflows over HPC resources is one solution for the fast data processing required for efficiently managing large and evolving collections. Through this scalable solution, variable loads of data can be managed at the desired frequency. Tasks like metadata extraction, file-format conversions and checksum calculation can be made a part of the workflows that can be run as batch processes on HPC resources, thus freeing the local resources at the data curators' end for other tasks.

Transferring large data from remote storage locations to the compute nodes for completing tasks can produce noteworthy bottlenecks. These can be mitigated by access to high speed networks, or by storing data in close proximity to the computing resources. Depending on the computing resources available, there is more than one way or technology to implement parallelization of metadata extraction tools (Schlarb, 2013). Given that DROID was not designed for parallel processing, the possibility to modify DROID for supercomputing environments also exists. Future work will involve implementing DROID in a Hadoop-configured environment and comparing its performance with our current results.

Our current work helped achieve the goal of obtaining updated collection snapshots and has also demonstrated that it is possible for data curation workflows to be moved to HPC resources. The lessons learnt from this project indicate that by lowering the adoption barriers to HPC through trainings and workshops, the likelihood of performing data management tasks frequently and efficiently will increase. Thus, the access to and reuse of the ever expanding quantities of digital data will increase.

References

Esteva, M., Trelogan, J., Rabinowitz, A.,Walling, D. & Pipkin, S. (2010). From the site to long-term preservation: A reflexive system to manage and archive digital archaeological data. In *Archiving 2010 Final Program and Proceedings* (pp. 1–6). Retrieved from http://ist.publisher.ingentaconnect.com/content/ist/ac/2010/00002010/00000001/art00001

Esteva, M., Trelogan J., Xu, W. & Solis, A. (2013, June). *Lost in the data: Aerial views of an archaeological collection.* Short paper presented at the 2013 Digital Humanities Conference. Lincoln, NE. Abstract retrieved from http://dh2013.unl.edu/abstracts/ab-371.html

Schlarb, S. (2013). Droid file format identification using Hadoop [Web log post]. Retrieved from http://www.openplanetsfoundation.org/blogs/2013-05-24-droid-file-format-identification-using-hadoop

TACC. (2013). TACC receives NSF grant to deploy innovative new data resource [Press release]. Retrieved from https://www.tacc.utexas.edu/news/press-releases/2013/wrangler-nsf-grant

Wilkinson, B. & Allen, M. (2005). *Parallel programming: Techniques and applications using networked workstations and parallel computers*. Prentice-Hall, 2nd edition.

Xu, W., Esteva, M., Trelogan, J. & Swinson, T. (2013). A case study on entity resolution for distant processing of big humanities data. In *Proceedings of the 2013 IEEE International Conference on Big Data* (pp. 113–120). doi:10.1109/BigData.2013.6691678