



11th International Conference on Open Repositories
Trinity College, Dublin, Ireland
13th - 16th June, 2016

Are Repositories Impeding Big Data Reuse?

Zhiwu Xie¹, Andrej Galad², Yinlin Chen^{1,2}, and Edward Fox²

¹University Libraries and ²Department of Computer Science, Virginia Tech
[zhiwuxie, agalad, ylchen, fo]@vt.edu

Session Type (select one)

- Panel
- Presentation

Abstract

In this intentionally provocative presentation, we question the scalability of popular digital repositories and whether they are suitable for big data reuse. Are the layers of API these repositories have painted over file system primitives necessary? How essential is it for the repository to insist on being the sole manager of the content, and arranging files in ways to prevent access other than from their own APIs? We explore these questions from the perspective of big data reuse, and describe controlled reuse experiments against Fedora 4 to evaluate the cost of these practices.

Conference Themes

Select the conference theme(s) your proposal best addresses:

- Supporting Open Scholarship, Open Data, and Open Science
- Repositories and Cultural Heritage
- Repositories of high volume and/or complex data and collections
- Managing Research Data, Software, and Workflows
- Integrating with the Wider Web and External Systems
- Exploring Metrics, Assessment, and Impact
- Managing Rights
- Developing and Training Staff

Keywords

Institutional repository, data management, big data, scalability, throughput

Audience

Repository managers, developers, data producers, librarians, data users.



Background

This presentation focuses on reuse issues facing repositories with high volumes of research data. Many big data reuse workloads, especially those employed in the exploratory stages of research, that skim high volumes of data to generate panoramic views, are “embarrassingly parallelizable”. Many digital curation workloads also fall into this category. For example, when checking fixity on large numbers of files, calculating a digest on one file does not depend on the calculation of the others, therefore can be done in parallel on different computing resources. On the other hand, modern computing hardware and systems have become remarkably efficient and scalable on such workloads. We therefore anticipate the overall data reuse performance to be a close approximation to linear scalability. That is, if we want to complete the job faster, we only need to allocate proportionally more computing resources.

This assumption, however, is only partially realized in our experiments on data reuse. Figure 1 shows a fixity checking experiment we ran on the Amazon cloud against a Fedora 4 repository holding sensor data. The linear scalability quickly hits a bottleneck when we increase the number of computing nodes to only 3 (using m4.large instances) or 4 (using t2.medium instances), most likely throttled by the single server bandwidth limit. For scalability bottlenecks like this, is the repository system usually the culprit? If so, what can we do? We embark on a project to further investigate the root causes of these bottlenecks and to determine how best to eliminate it.

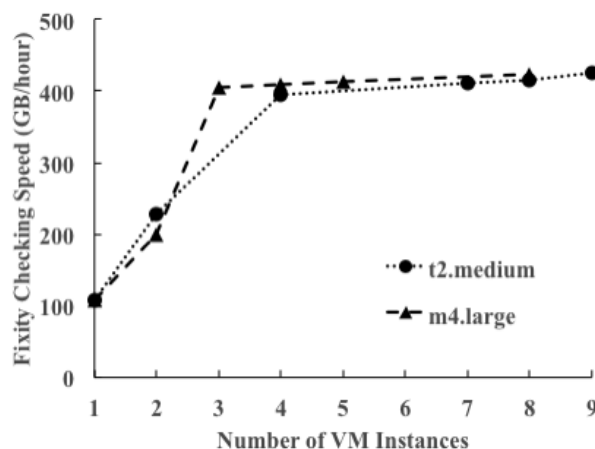


Figure 1. Linear Scalability and Bottleneck



11th International Conference on Open Repositories
Trinity College, Dublin, Ireland
13th - 16th June, 2016

Presentation content

Since the computing is highly parallelized, the bottleneck either should be with the repository, the data movement, or somewhere in-between. We measure the scalability of moving data around at each stage of the reuse scenario:

- Moving data out of the repository when data are managed directly by the repository, which may be further broken down into 3 different scenarios, moving: 1) to the local file system; 2) to a network attached file storage system like S3; and 3) directly to the processing node. The repository API calls, the underlying storage system API calls (e.g., Modeshape and InfiniSpan), the local file system bandwidth, and the network bandwidth between the repository and external systems, may potentially be the limiting factors. More specifically, we are testing how well the repository API deals with moving high volumes of data, e.g., when the API response message contains a large payload.
- Moving data through the repository when data are only “registered” with the repository. In this case, we can move data via channels different from the ones provided by the repository. We thus are only testing how well the repository API scales with tiny message payloads.
- We also experiment with read-dominated Fedora clustering configurations and see how well it scales for messages with large and small payloads.
- Moving data into the processing node, which may be further broken down into: 1) moving from a remote file system hosting the repository; 2) moving from a remote storage system like S3.

We will present the results of these experiments, then explore how to identify the bottleneck encountered in the prior experiments, and discuss how to eliminate it.

Conclusion

When the data volume grows higher, repository users increasingly need efficient utilities and infrastructure to extract knowledge from the data. While many such utilities and infrastructures are under development, we take one step back and review the reuse issue from the perspective of the repository architecture, and ask whether the current design constitutes a reuse bottleneck.