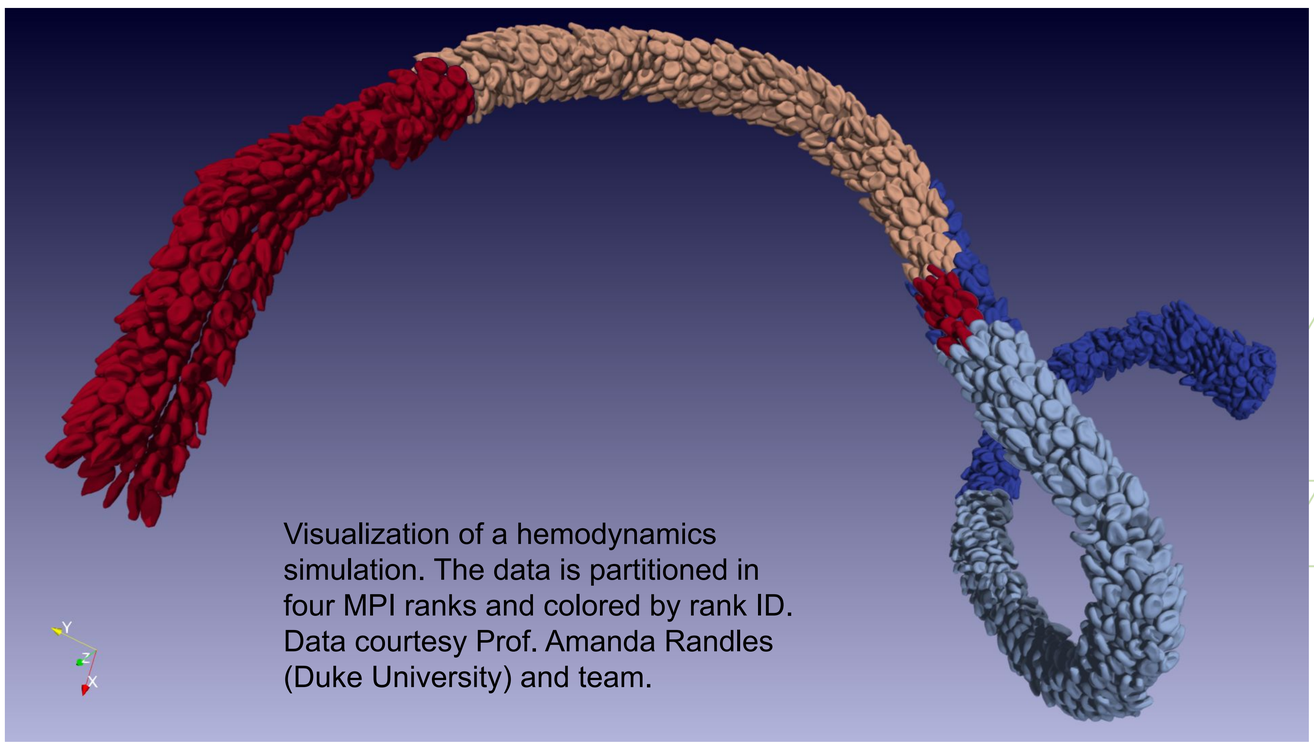
Evaluating Performance of Rendering with Containerized Visualization Tools

Renee Waverly Sonntag[¶] – Joseph Insley[*¶] – Janet Knowles[*] – Victor Mateevitsi[*] – Michael E. Papka[*¶] – Silvio Rizzi[*]

[¶]Northern Illinois University [*]Argonne National Laboratory

Abstract

Docker and Singularity are well known containerization platforms, becoming increasingly popular in high performance computing. They allow for complex installation processes to be performed once; the resulting environment can be executed later on other systems. Scientific visualization tools, such as ParaView, depend on a significant number of libraries that may be challenging to build and deploy on new systems. Moving ParaView and its libraries into a container would simplify the deployment of the installation, handling dependencies independently of the system in use. Through this work, we explore the process for containerizing ParaView and evaluate its performance on a number of distributed parallel computers against their native installations.



Visualization of a hemodynamics simulation. The data is partitioned in four MPI ranks and colored by rank ID. Data courtesy Prof. Amanda Randles (Duke University) and team.

Introduction

- Striking a balance between user control and platform security, containers have found use in high performance computing.
- As visualization technology grows, the ability to deploy a complete visualization solution on any machine grows in importance.
- Virtualization is known to have an overhead and performance impact.
- This work seeks to provide a framework for producing a container with ParaView, and to evaluate its performance against a native installation.

Related Work

- In situ visualization and analysis is a very active field of research in preparation for exascale systems
- Shudler, et al. found that the run time for calculations were approximately equivalent between the containerized and native installations.
- Our work follows a similar approach to demonstrate a scientific visualization software stack based on ParaView.

Docker and Singularity

- Docker is the most common containerization tool in use today, but is generally considered insecure.
- Singularity was created to address those concerns, and is installed in many high performance computing facilities.
- Singularity is able to import Docker images, making them interchangeable.
- Writing a Dockerfile, therefore, would allow the image to be run in any facility regardless of the specific containerization flavor in use.

ParaView, OSPRay

- ParaView supports modules which can extend it to support new hardware.
- OSPRay supports raytracing with Intel CPUs.
- ParaView also supports MPI, which makes it ideal for use with supercomputing resources.
- Combining these modules together yields a

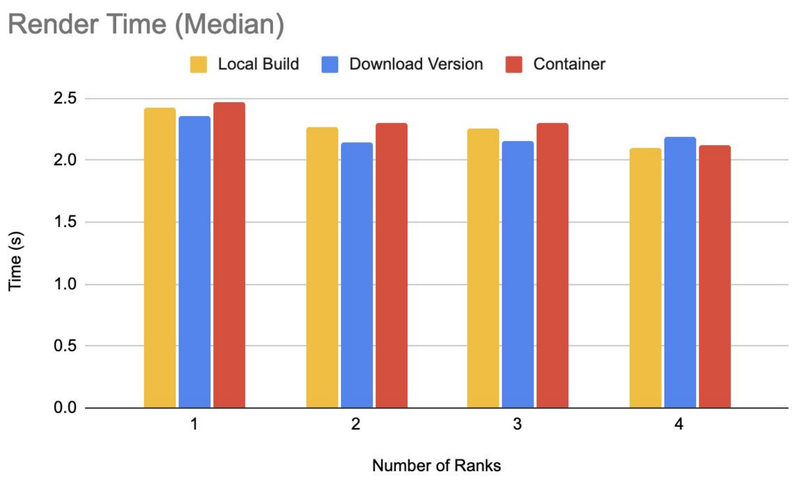


Figure 2: Median Render time on Cooley.

Experiment

- Medium scale machine:
  - Cooley, a visualization cluster at Argonne Leadership Computing Facility.
- Small scale machines:
  - Hopcroft, a small cluster at Northern Illinois University.
- Developed a container which provides ParaView, OSPRay, and all of the dependencies required for those modules.
- The source for the image is a Dockerfile, which is built and imported into Singularity.
- The image relies on dynamic linking to leverage the native MPI on the host and take advantage of network optimizations, such as Infiniband.

Results

- 60 Renders total, 36 on Cooley and 24 on Hopcroft.
- 12 native, 12 container, an additional 12 on Cooley since it had a prebuilt ParaView 5.9 version available.
- In Figure 2 we see that the container is closely comparable to the local build, trending a bit slower no matter how many nodes are used.
- In Figure 3 we find that the container is closely comparable to the downloaded version.

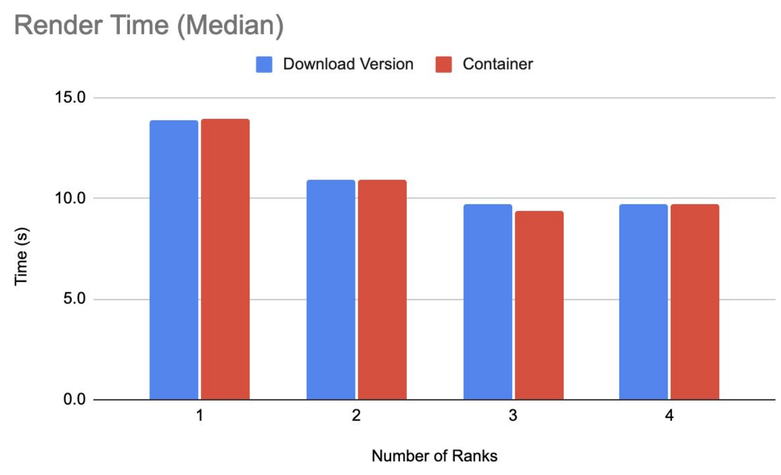


Figure 3: Median Render time on Hopcroft.

Conclusion

- We could reasonably conclude that the overhead of containers has a minimal impact on the performance of the render as a whole.
- Especially when considering the benefit that containerization of ParaView and other in situ frameworks would bring in portability, containerization holds promise.
- In the future we will expand this work to include in situ visualization and analysis frameworks that use ParaView with OSPRay as a rendering backend. We will also increase the scale of these experiments in preparation for the arrival of the first exascale supercomputers.

This research used resources of the Argonne Leadership Computing Facility, which is a U.S. Department of Energy Office of Science User Facility supported under Contract DE-AC02- 06CH11357.