Transcript of 5-minute ePoster presentation
Viral-NGS: Cloud Compute for Viral Genomics Using GA4GH Standards
Presenter: D. J. Park

Text:

Thanks for checking out this ePoster! My name is Danny Park, from the Broad Institute's Viral Genomics group. I'm presenting our computational tool suite called "viral-ngs", on behalf of the engineers, scientists, users, students, postdocs, and collaborators, that have all contributed to it over the past several years. *[Zoom to upper left panel]* **There are two key ideas I want to convey in this poster: how we maximize the portability of our pipelines across different compute environments, and relatedly, how we increase its accessibility to a wide range of researchers.** And I believe that the technical lessons we've learned in addressing these issues over the past few years are important enough to make generalized recommendations for anyone building similar "bioinformatic pipeline" styled analysis tools.

[Zoom to lower left panel]

I'll start by briefly describing what "viral-ngs" does -- there are other tool sets like it, but this one is ours. It is a series of analysis pipelines that start with raw Illumina data from viral sequencing runs, and provide the ability to perform several kinds of analyses on the short reads. This includes sequencing QC, metagenomic classification, assembly of viral genomes, phylogenetics, and related visualization tools. This is typically provided in a secure, FedRAMP-compliant cloud compute platform that provides shared "workspaces" that manage shared data and compute jobs. But the key thing is that "viral-ngs" is currently routinely used on multiple different compute platforms. We heavily utilize the Terra platform, which is an academic computational genomics platform that utilizes Google Cloud resources, we often also make use of the DNAnexus platform, a commercial vendor that utilizes AWS and Azure resources, and also utilize execution engines like miniWDL and Cromwell to run these very same analyses on our laptops, shared machines, or clusters. In fact, we regularly test and use "viral-ngs" in all of these different engines.

[Zoom to lower center panel]

We do this because we found that some of these compute environments are better suited for different users. "Viral-ngs" was originally created to provide all the needed analytic capability for both the viral group at the Broad, as well as the new (as of 2013) viral sequencing labs based at our H3Africa consortial partners in West Africa. We learned quite quickly that our cloud-based compute platforms (Terra and DNAnexus) provided the most effective, secure, reliable, and accessible compute environment for everyone in our consortium. Over the years, it has enabled a lot of locally driven analysis on pathogen genomic data generated by our research partners, including numerous outbreak investigations and surveillance studies.

But we also found that US state public health labs, who were onboarding NGS capabilities around the same time, were eager to adopt the same analysis capabilities, but were often not approved for cloud use outside of a training context, and preferred the on premises / command line versions of “viral-ngs” (provided by miniWDL and Cromwell).

[Zoom to right panel]

Ultimately, achieving cross platform portability required intentional design and implementation decisions from the beginning. For starters, it involves recognizing that this is a very well trod path, especially in computational genomics. Many standards and APIs have formed over the past several years, especially out of the Global Alliance for Genomics and Health (GA4GH). Although our “viral-ngs” toolset predated many of these standards, we aggressively adopted them as soon as they became mature enough for our needs.

In addition to fully containerizing and modularizing your pipelines, the next step requires expressing the “plumbing” code of your pipeline in a GA4GH-compatible workflow language. We chose to utilize “WDL” (the workflow description language), as it is one of the more formal and portable options, and quite popular in human genomics, but many also use “CWL”, and increasingly, “Nextflow” has been engaging the GA4GH standards more as well. Here, I am showing the depiction of our assembly pipeline as a flowchart on the left, and in WDL code on the right, to show its simplicity at describing the “plumbing” of different steps.

Once you have your code in a good workflow language, it’s not quite enough to say “here it is on my github, go get it and try it yourself”. Going the extra step to distribute your pipelines in a Tool Registry Service, such as Dockstore, allows for a more robust validation and assertion of its portability, and more seamless integration with a number of cloud compute platforms (this picture shows 1-click launch buttons for an assembly workflow on five different platforms, including both Terra and DNAnexus). It also allows independent entities or users to curate collections of related tools written by various tool authors.

Long story short, this type of interoperability exists in computational genomics today, and any toolmaker would be remiss not to take advantage of it. I described what this looks like for portability of tools and pipelines, but this level of federation also exists for enabling data portability and shared compute resources. I’d be happy to connect on our experiences with any of this for those who are interested!