COVID Information Commons (CIC) Research Lightning Talk

Transcript of a Presentation by Niema Moshiri (University of California, San Diego), April 24, 2023

Title: Massively scalable reference-guided Multiple Sequence Alignment of viral genomes

Niema Moshiri CIC Database Profile

NSF Award #: 2028040

YouTube Recording with Slides

Spring 2023 CIC Webinar Information

Transcript Editor: Lauren Close

Transcript

Slide 1

Niema Moshiri:

Awesome, yeah, thank you for the introduction. Hopefully, folks can see my screen. Yeah, so hey everyone. As mentioned, my name is Niema Moshiri. I'm an Assistant Teaching Professor in the Computer Science and Engineering department at UC San Diego. My talk is going to be focused on some methods that my lab developed using the NSF funding for speeding up viral genomic analysis. Specifically, today's talk is just going to focus on how we've enabled massively scalable reference guided multiple sequence alignment of complete viral genomes. We actually have done multiple other accelerations as well that I just didn't have time to talk about today. I'll end with a link to my website if folks are curious about how else you could speed up other aspects of this type of analysis.

Slide 2

So let's get started. Just to give a little bit of context - here's a framework for a standard viral phylogenetics workflow. And, you know, before I even talk about this, viral phylogenetics is very important to be able to study how the virus is mutating over time. How is it, kind of, branching off? How are the different samples that we collect all over the world related? There's a ton of uses in the world of viral molecular epidemiology that are out of the scope of this talk, but generally, having a phylogeny inferred from viral genomes is very useful to have. Typically the workflow starts like this where you start with a bunch of unaligned viral genome sequences, which I'm showing over here. The first step is usually multiple sequence alignment where you

try to kind of place these gaps in the different kind of positions of each of the sequences to get them to line up better. This gives you some notion of sequence homology after you do this. Then, given the multiple sequence alignment, we can then perform phylogenetic inference to try to infer an unrooted evolutionary relationship between these sequences. Then, typically after that, we do what's called rooting to determine what is the most likely common ancestor of all of the sequences. That kind of then tells us what was the forward in time evolutionary history of these sequences. Then, maybe you'll do some additional downstream analyzes. Maybe you do transmission clustering. There's there's a lot of other analyzes that you can do on the phylogeny and on the sequences. But this is kind of the the building blocks of how you do all these other analyzes. So typically these steps over here are the key computational bottlenecks. The multiple sequence alignment and then the phylogenetic inference. In today's talk, I won't be talking about phylogenetic inference I'm just going to be zooming in on multiple sequence alignment.

Slide 3

So, some context - multiple sequence alignment this is what's called an NP-Complete computational problem. What that means - there's a very technical computer science term - but basically what this means is there's no polynomial time exact solution. Basically it gave me a bunch of sequences and asked me to come up with the optimal multiple sequence alignment. There is no way to do this in polynomial time. It's very very slow. Heuristics have been developed to provide optim- to provide approximate solutions. For example, you might have heard of ClustalOmega, MUSCLE, and MAFFT. These are some kind of standard tools that are used in the space. However, even these heuristics - they generally scale quadratically with respect to the number of sequences. For context, the GISAID database, which is the database where most folks are storing their full SARS-CoV-2 genomes, this database is growing extremely rapidly and as of today we have over 15 million SARS-CoV-2 sequences available from all over the world. The next epidemic is going to be more and kind of sequencing genomes in real time. This is going to be a tool that hopefully we continue to use in in the viral epidemics to come. We can expect this to be even more significant of a big data problem. Currently with these tools like ClustalOmega, MAFFT, and MUSCLE, we're looking at runtimes of decades to centuries, which, you know, for obvious reasons, if we're trying to do real-time molecular analysis, decades or centuries is just a little bit too slow. So how can we speed this up? Well, it turns out that the problem is actually a little bit easier than what we're trying to solve. Multiple sequence alignment, in general, is kind of assuming no homology of the sequences whatsoever. This is the time it takes to align completely arbitrary sequences. But SARS-CoV-2 and with viruses in general we have a much simpler problem, right? We have a lot of sequence homology. Even if the virus is mutating, you know, significantly across the world, every single viral sequence that we obtain is going to be almost identical to the reference gene. It's not going to be exactly identical, but it's going to be almost identical. So we actually are facing a much simpler computational problem which is multiple sequence alignment of highly similar sequences. So how can we use that feature to speed up this analysis?

Slide 4

We can do what's called an align-to-reference approach. So instead of just trying to align everything with each other all at once, what we could do is individual pair-wise alignments against a reference unit. So in this figure, the thicker green bar at the top represents the reference to our scope 2 genome, and each of these other colored genomes represents a sequence that I collect from the real world. I want to align each of these to the reference genome. What I could do is just one by one by one by one I can independently align each of these genome sequences against the reference genome, which I could do each of these fairly quickly and I can do massive parallelization because each of these pairwise alignments to the reference can be done completely independently. I can parallelize however many cores my computer has, I can throw that many at this problem. Then, once I've competed all those pairwise lines to the reference, I could use the wrap wrench genome - I can kind of use its anchors, its positions as anchors to create the columns of my multiple sequence line. For example, maybe I'll start with the first position of the reference genome and I'll see, okay, well, in the red sequence this is the letter that aligned to that position. In the orange sequence, this is the letter. In the pink sequence, this is the letter. In the blue sequence, this is the letter. And I can kind of merge all of those letters into one column of my multiple sequence alignment. And I could do the same thing for the second position of my reference genome, the same thing for the third position, fourth position, all the way across. And kind of position by position by position I can build my multiple sequence alignment. This idea, this is really good because it is massively parallelizable and it scales linearly with the number of sequences rather than quadratically. So it has much better scalability as well. Do we have to implement this approach from scratch? We actually don't.

Slide 5

It turns out that if you kind of step back and think about this problem, this is really equivalent, in a sense, to the long read mapping problem. Let's just kind of step back and rethink what is the problem that we're tackling. Our input is a reference genome and a bunch of long sequences that are very similar to the reference genome. Our output is an alignment of each of those sequences against the reference genome. This is exactly the same computational problem as mapping long reads. Instead of having to reinvent the wheel, we could just kind of build off of all of these really advanced techniques that folks have built to solve the long read mapping problem and just kind of apply it to this context.

Slide 6

To that aim, I developed a tool called ViralMSA and what it does is it just wraps around existing long read mappers to perform this reference-guided multiple sequence alignment. It kind of treats each of those genomes that I've collected as long reads and it treats the reference genome as a reference genome. It just calls that read mapper - I wrap against a few different read mappers just to demonstrate flexibility - but I mainly suggest people use Minimap2 for both speed and accuracy. Then, given those read mapping results, I can then - or given the mapping results - I can then just kind of compile them into a single multiple sequence line.

So all you have to do to run ViralMSA is you just give ViralMSA a reference genome and a bunch of sequences to align. It'll automatically handle indexing the reference genome, maybe if you give it an accessory number it'll handle downloading and indexing the reference genome. It'll handle all of the pre-processing and all the downstream stuff and it'll just output - it'll call the read mapper, it'll merge the results into the multiple SQL alignment, and it'll just output a single standard file that is your multiple sequence alignment.

Slide 7

How does it do against existing tools? We we did a benchmark experiment where we compared the runtime of ViralMSA wrapping around Minimap 2 compared against Virulign, which is an existing align to reference approach but that just kind of implements its own from scratch aligned to reference. We also compared against MAFFT which is typically considered one of the most commonly used multiple sequence segment tools. In this plot, on the horizontal axis I have number of sequences. On the vertical axis I have total execution time in seconds. This was done on complete SARS-CoV-2 genome sequences, so genome length roughly 29,000. As we can see, the blue line, which is ViralMSA, is orders of magnitude faster than the existing tools. Compared to VIRULIGN, which is also scaling linearly, we're getting - and by the way, this this plot is a log scale plot - so compared to VIRULIGN, we're roughly like a thousand times faster-ish. And with MAFFT, we're not quite as much faster, but you can see that because MAFFT scales quadratically, our speed up with respect to MAFFT is actually increasing as time progresses. Even at just a thousand sequences. we hit roughly a thousand times faster and that gap kind of increases.

Slide 8

Now, you might be wondering, okay, well, fast is good but what's the point if it doesn't give me good alignments? We also compared the accuracy. What we did was we took the multiple sequence alignment computed by MAFFT, took the multiple sequence alignment computed by ViralMSA on a bunch of hand-curated alignments from HIV, Ebola, and I'm blanking on the third virus, but basically we took viruses that we had curated alignments from the Los Alamo- oh actually, no - this plot is from just HIV-1. From the Los Alamos National Lab we took their curated multiple sequence alignments and we use that as ground truth. Then we saw how does mapped in ViralMSA compare against the ground truth curated multiple sequence alignment. If we compute pairwise distances of the sequences that we get in our alignment, and then we do a mantle test for accuracy - we find the correlation between our pairwise distances, and the pairwise is calculated directly from the true multiple sequence alignment, we see that the correlation is negligibly different. Here, you know, we get a correlation coefficient like 0.994 for viral MSA compared to 0.997 for pairwise distance calculation. Actually, when we calculated phylogenies, the phylogenies inferred using the ViralMSA multiple sequence elements are actually slightly higher topological accuracy than those estimated from MAFFT. So, negligibly so, but still what we're showing is that these are kind of essentially equivalent in terms of accuracy for all intents and purposes.

Slide 9

The conclusion - ViralMSA is a tool that enables rapid multiple sequence alignment of ultra large viral datasets. It's open source, you can find it on GitHub, and you know, please consider using it in your viral analyzes.

Slide 10

And acknowledgments - I want to thank Heng Li, he's the developer of Minimap2 and it's really his expertise in developing Minimap2 that enables ViralMSA's speed and performance. I want to thank the NSF for the grant that supports this project. And the the research was also supported using Google Cloud platform research credits.

Slide 11

So I'll save time for any questions or I'm happy to end it here.