blastEasy - A Scalable Approach to SequenceServer's BLAST+ Using WorkQueue, Atmosphere Virtual Machines, and Docker Containers

Team Members:

Sateesh Peri: sateeshp@email.arizona.edu

Tanner Campbell: tcampb@email.arizona.edu

Naomi Yescas: yescasa@email.arizona.edu

Mohammad Moghaddam: moghaddam@email.arizona.edu

Sahil Brahmankar: sbrahmankar@email.arizona.edu

blastEasy

Summary

blastEasy is a framework of virtual machine images for the purpose of improving genomics training and research for students and professors part of the Genomics Education Alliance. blastEasy provides an easy and scalable way of running protein and nucleotide searches by modifying and sequenceserver, an existing tool with launches a server to allow users to use the Basic Local Alignment Search Tool (BLAST). SequenceServer provides a user-friendly interface to allow those with limited command line or computer science experience to conduct protein and nucleotide database searches using BLAST. This service is limited in the number of users it can support without significant loss in search speed. blastEasy provides a solution by intercepting each BLAST search and using the Cooperative Computing Lab's WorkQueue, distributes work loads across multiple machines. This allows for scalability and improved search times for classrooms of multiple students conducting searches at the same time.

Description and Technical Objectives

blastEasy harnesses the power and flexibility of Cyverse's Atmosphere cloud computing platform by packaging sequenceserver, cctools, and BLAST databases into virtual machine images. It requires a level of familiarity with launching virtual machines (VMs) on Atmosphere as well as a Cyverse account. One simply creates a Master-VM using the TeamBLASTEasy SeqServer 1.0.12 image and one or more Worker-VMs using the CCTools_7.0.19 image. Then, the instructor can launch sequencserver on the Master-VM and connect as many Worker-VMs as needed. The students can conduct BLAST searches using sequenceserver as they normally would and the blastEasy framework will handle distributing searches across all available workers.

blastEasy is Scalable:

- blastEasy uses WorkQueue to distribute tasks consisting of genomic sequence searches over a custom database using Atmosphere virtual machines
- blastEasy containers makes it easy to connect any machine to the Master-VM running sequenceserver

This means blastEasy's framework is not limited to Atmosphere VM's. It's possible for any machine with access to the blastEasy containers to supply computational power.

blastEasy is Customizable:

- Commonly used databases or Course-Specific databases can be preloaded into a blastEasy Atmosphere image upon request.

blastEasy is Easy:

- It's as easy as launching a virtual machine on Atmosphere

<u>GOAL:</u> Create GEA BLAST service to support genomics training and research for undergraduate students

+ Develop a system that divides and distributes BLAST searches across multiple nodes and processors to obtain results faster. [critical]

+ Host BLAST implementations that support multiple classes of at least 100 students. [critical]

+ Assurance that the ~100 jobs will finish at approximately the same time. [critical]

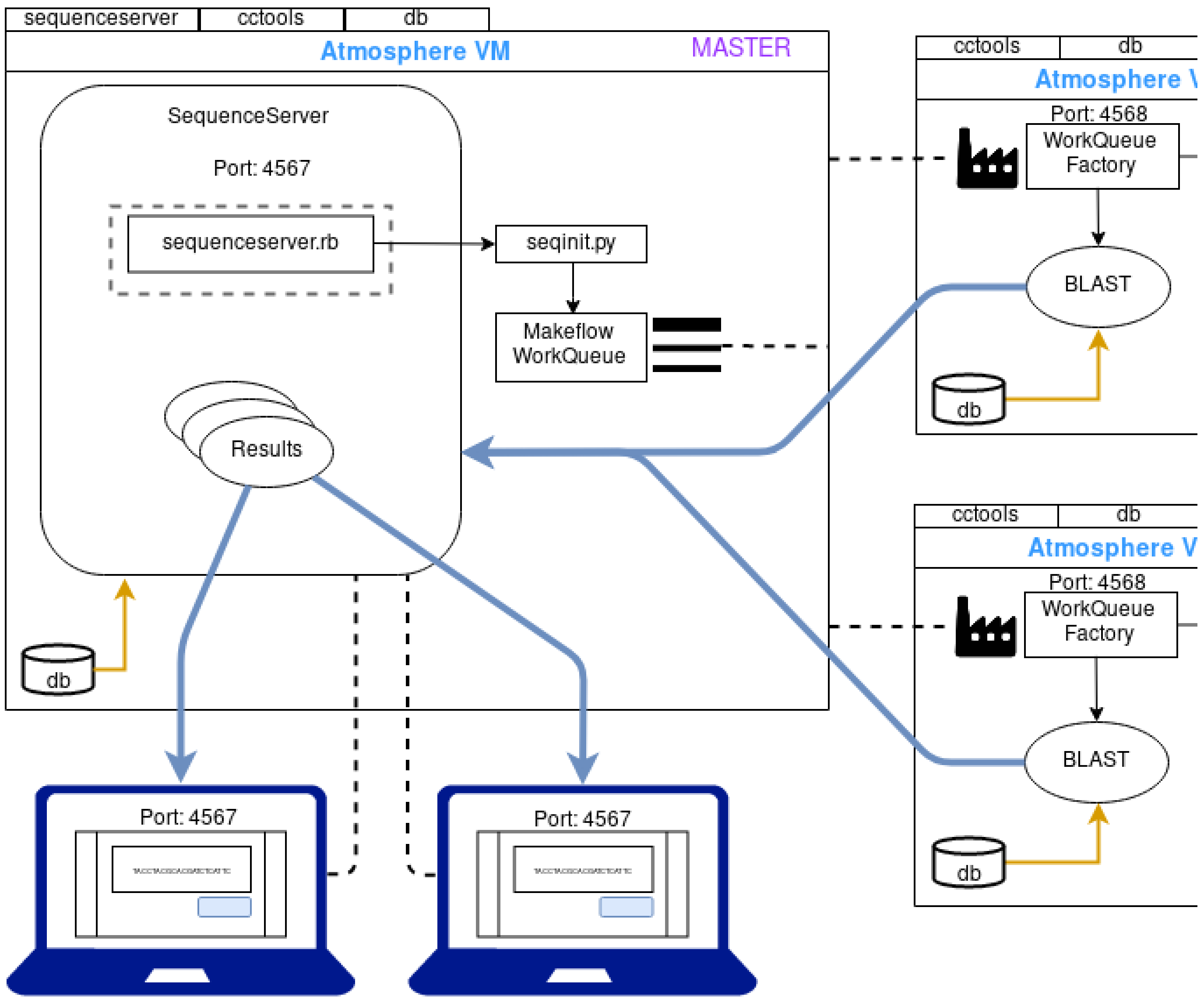
+ Require support for faculty to create custom BLAST databases and adjust BLAST search parameters [nice-to-have]

+ Require support for caching BLAST results [nice-to-have]

+ Provide authentication for security [nice-to-have]

+ video tutorial demo of the end-product [nice-to-have]

blastEasy Concept Map



blasteasy source code:

GitHub: https://github.com/raptorslab/blastEasy.git

Atmosphere VMs:

Master Image: TeamBLASTEasy SeqServer 1.0.12 https://atmo.cyverse.org/application/images/1756

Worker Image: CCTools_7.0.19 https://atmo.cyverse.org/application/images/1748

blastEasy Setup Instructions

Instructions to Instructor:

Note: Setup time takes around half an hour prior to class

- To deploy blastEasy setup on CyVerse Atmosphere cloud, you will need access to Atmosphere.
- You will need to launch a Master instance that will host sequenceServer and one or more `Work_Queue_Factory` instances as needed to distribute the blast jobs.

Blast Databases

- **For this setup to work, blast databases should be placed in the same location on both Master and Worker VM's.**
- This sequenceServer image has three test protein databases (mouse.1, mouse.2, zebrafish.1) in `/Data` that can be used for testing.
- Several NCBI public databases are also hosted in CyVerse Data Commons. Access them using icommands as follows:
    - List available databases
        - `ils /iplant/home/shared/iplantcollaborative/example_data/blast_dbs`

    You should see a listing of the blast_dbs folder like this:

```
/iplant/home/shared/iplantcollaborative/example_data/blast_dbs:
C- /iplant/home/shared/iplantcollaborative/example_data/blast_dbs/16SMicrobial
C- /iplant/home/shared/iplantcollaborative/example_data/blast_dbs/human_genomic
C- /iplant/home/shared/iplantcollaborative/example_data/blast_dbs/pdbaa
C- /iplant/home/shared/iplantcollaborative/example_data/blast_dbs/pdbnt
C- /iplant/home/shared/iplantcollaborative/example_data/blast_dbs/refseq_protein
C- /iplant/home/shared/iplantcollaborative/example_data/blast_dbs/refseqgene
```

- Download a database from CyVerse data commons to the `/scratch` folder on your VM as follows:
    - `irsync -r i:/iplant/home/shared/iplantcollaborative/example_data/blast_dbs/pdbaa /scratch/`
    - `irsync -r i:/iplant/home/shared/iplantcollaborative/example_data/blast_dbs/pdbnt /scratch/`
- To use CUSTOM databases, we recommend uploading the sequences to CyVerse data store and use DE apps to make blast databases that can be downloaded to Master and Worker VMs using iRODS. Read more [here] for more detailed instructions

Steps:

1. Launch a Master (small) instance which will broadcast as a Master using **this** image.
2. Launch a Worker (medium to large) instance with this image with **this** cctools image.
3. On the Master VM, launch sequenceServer as follows: `sequenceserver -d /path_to_databases`

Note: Take a note of the Master VM's IP_ADDRESS and the port on which sequenceServer is listening for the next steps.

1. Now you or your students can open a web-browser and go to `IP_ADDRESS_of_Master_VM:PORT` to access sequence server front-end.
2. Connect Work_Queue_Factory to Master VM before submitting blast jobs by `work_queue_factory IP PORT -T local -w Min_NUM_OF_Workers`

NOTE: The PORT for connecting work_queue_factory above would be the (Sequence_Server_PORT_NUM + 1)

Note: One can connect as many `Work_Queue_Factory's` as needed as above but, make sure to have the blast databases in the same path as Master and other workers.

1. Once worker factory is connected, blast queries can be submitted and results can be accessed using front end while the time to blast query is printed on the Master VM backend terminal for benchmarking.

Team Members

Sateesh Peri: sateeshp@email.arizona.edu

Role: Team lead, backend-design

Expertise: Bioinformatics, Genetics, Cyverse & Cloud Computing

Tanner Campbell: tcampb@email.arizona.edu

Role: sequenceserver-reverse-engineering, code, backend-design

Expertise: Celestial Mechanics/Spacecraft GNC/ Machine Learning

Mohammad Moghaddam: moghaddam@email.arizona.edu

Role: Benchmarking, Testing

Expertise: Hydrology/MIS, Machine Learning, Statistics

Sahil Brahmankar: sbrahmankar@email.arizona.edu

Role: Benchmarking, Testing

Expertise: Information Science

Naomi Yescas: yescasa@email.arizona.edu

Role: Documentation, Concept Map

Expertise: Information Science, Machine Learning

Project Timeline:

09/19/19	Identify Stakeholders, Preliminary Planning and Concept Map
10/03/19	Use and test sequenceserver docker container
10/08/19	Benchmark sequenceserver on 1, 2, 4, 8, and 16 CPU-virtual machines
10/06/19	Implement single BLAST queue parallelization using Makeflow and Workqueue
10/15/19	Create sequenceserver Atmosphere Image modified to wait for workers
10/30/19	Launch sequenceserver with workers across multiple VMs
01/05/19	Deliverables: • blastEasy GitHub source code • blastEasy DockerHub container • Master Atmosphere Image • Worker Atmosphere Image

Benchmarking Part-1: Initial testing

Query_seq	CPU#1	CPU#2	CPU#4	CPU#8	CPU#16	Database
	Mo	Naomi	Mo	Sateesh	Sateesh	
dna_1000	10m 25s	9m50s	8m34s	2m27s	18s	nucleotide (nt) (73GB)
dna_2000	9m 35s	8m27s	5m49s	1m17s	23s	
dna_5000	15m 20s	10m12s	6m 25s	1m13s	43s	
dna_10k	19m 59S	14m27s	7m7s	2m1s	1m12s	
dna_50k	47m 47s	39m02s	15m 28s	6m5s	3m59s	
dna_100k	1hr 48min	47m23s	30m 32s	38m9s	15m20s	
dna_500k	2h12m08s	1h58m29s	1h15m06s	57m42s	33m41s	
prot_1	9m12s	6m 4s	6m 58s	1m8s	39s	refseq_protein (88GB)
prot_5	25m34s	17m 57s	9m 12 s	3m31s	1m59s	
prot_10	42m45s	30m 25s	13m39s	5m59s	3m23s	
prot_50	59m55s	1h 48m 50s	40m34s	21m29s	12m7s	
prot_100	1h39m21s	1h02m12s	57m10s	41m36	23m1s	
prot_500	3h12m23s	2h23m58s	1h24m45s	1h14m32s	2h1m26s	

Table-1: Benchmark results; multiple Atmosphere virtual machines with 1, 2, 4, 8, and 16 CPU cores

The benchmarking was one by launching virtual machines with sequenceserver images with access to 1, 2, 4, 8, and 16 CPUs. The ncbi-blast nt and refseq_protein databases were downloaded and random dna and protein sequences were generated. Nucleotide sequences of 1000, 2000, 5000, 10000, 50000, 100000, 500000 were tested across each virtual machine. The protein queries were tested using multiple sequences: 1, 5, 10, 50, 100, and 500. The time was calculated using sequenceserver's debug mode, which displays the time each search begins and when it ends and starts a new process. This was done using sequenceserver's debug command (-D): sequenceserver -n 14 -D -d blast_dbs/ and checking the output for the displayed time:

Blast Begins:

[Date&Time] DEBUG Executing: blastn -db …

Blast Ends:

[Date&Time] DEBUG Executing: blast_formatter …

Benchmarking Part-2: Prototype Testing

...

Additional Links and Resources:

Cyverse:

Atmosphere: https://atmo.cyverse.org

More info and Instructions: https://wiki.cyverse.org/wiki/display/atmman/Atmosphere+Manual+Table+of+Contents

SequenceServer

Github: https://github.com/wurmlab/sequenceserver.git

More info and Instructions: http://sequenceserver.com

NCBI-BLAST

Blast: https://blast.ncbi.nlm.nih.gov/Blast.cgi

Databases: ftp.ncbi.nlm.nih.gov/../../blast/db

Presentation Slides:

https://docs.google.com/presentation/d/1JEvYqTRk9SJbwhIcsrwjVSi1XVOJyad2nsBxN3jne4A/edit?usp=sharing

Midterm Demo:

Human GAPDH sequence:

```
>NR_152150.2 Homo sapiens glyceraldehyde-3-phosphate dehydrogenase (GAPDH), transcript variant 6,
non-coding RNA
GCTCTCTGCTCCTCCTGTTCGACAGTCAGCCGCATCTTCTTTTGCGTCGCCAGCCGAGCCACATCGCTCA
GACACCATGGGGAAGGTGAAGGTCGGAGTCAACGGATTTGGTCGTATTGGGCGCCTGGTCACCAGGGCTG
CTTTTAACTCTGGTAAAGTGGATATTGTTGCCATCAATGACCCCTTCATTGACCTCAACTACATGGTTTA
CATGTTCCAATATGATTCCACCCATGGCAAATTCCATGGCACCGTCAAGGCTGAGAACGGGAAGCTTGTC
ATCAATGGAAATCCCATCACCATCTTCCAGGAGCGAGATCCCTCCAAAATCAAGTGGGGCGATGCTGGCG
CTGAGTACGTCGTGGAGTCCACTGGCGTCTTCACCACCATGGAGAAGGCTGGGGCTCATTTGCAGGGGGG
AGCCAAAAGGGTCATCATCTCTGCCCCCTCTGCTGATGCCCCCATGTTCGTCATGGGTGTGAACCATGAG
AAGTATGACAACGAATTTGGCTACAGCAACAGGGTGGTGGACCTCATGGCCCACATGGCCTCCAAGGAGT
AAGACCCCTGGACCACCAGCCCCAGCAAGAGCACAAGAGGAAGAGAGAGACCCTCACTGCTGGGGAGTCC
CTGCCACACTCAGTCCCCCACCACACTGAATCTCCCCTCCTCACAGTTGCCATGTAGACCCCTTGAAGAG
GGGAGGGGCCTAGGGAGCCGCACCTTGTCATGTACCATCAATAAAGTACCCTGTGCTCAACCA
```

Random DNA generator:

https://faculty.ucr.edu/~mmaduro/random.htm

Post-Mortem Analysis

What worked well:

- As this project was multi-faceted requiring skills ranging from understanding of biology to re-purposing software source code, each individual's talents became particularly useful in driving this project to a completion.
- Following AGILE principles in organizing our project
- Team communication via slack was pivotal in keeping this team together

What didn't work well:

- The team was quite innovative with their solution but, not investing more time in a secondary backup plan in-case there were issues with plan-A.

What could have been differently:

- Extensive benchmarking involving larger BLAST databases would be nice to have
- Project timelines and deadlines to be communicated well in advance so as to not rush the results in the end.