# Rapid RNA sequencing data analysis using serverless computing

Ling-Hong Hung[1], Dimitar Kumanov[1], Xingzhi Niu[1], Wes Lloyd[1], Ka Yee Yeung[1,*]
[1]School of Engineering and Technology, University of Washington, Tacoma, WA 98402, USA
[*]Corresponding author

## Abstract

We have used serverless AWS Lambda functions to align 640 million reads in less than 3 minutes, a speed-up of 500x over the single-threaded implementation. Using a hybrid cloud architecture and software modified to optimize disk transfers, an entire RNA sequencing workflow transforming multiplexed reads to transcript counts that originally took 29 hours can be completed in 18 minutes. This is a 100x improvement over the original single threaded implementation and 12x faster than an optimized cloud server-based implementation using 16 threads. The total cost of the analyses is $2.82 for 96 wells or 3 cents per multiplexed sample. This approach can be used for human datasets that are generated for single experiments and does not rely on processing large numbers of samples to achieve the performance gains. The workflow is publicly available under a M.I.T. license (https://github.com/BioDepot/RNA-seq-lambda).

# Introduction

Cloud computing has become an essential resource in the analyses of big biomedical data by offering massive scalable computing and storage, secure access to protected data, and on-demand access to resources and applications [1, 2]. A major barrier to widespread adoption of the cloud for large scale parallel bioinformatics jobs is the need to provision and configure virtual servers before computation can proceed. Setting up the infrastructure requires expertise and can consume more time than the actual computation. In contrast, code snippets can be easily deployed to the cloud in the form of microservices using serverless function-as-a-service (FaaS) platforms where resource provisioning, monitoring, scaling, 24/7 availability, and fault tolerance are provided automatically [3, 4]. The serverless function-as-a-service paradigm has emerged as a simplified programming model that can be used to create scalable cloud applications with reduced configuration and management overhead [5]. In addition to the ease of use, these microservices, are inexpensive, can be launched in under a second and require no provisioning or specification of instances. This is a true on-demand model: there are no compute resources reserved and no cost to the user when a user's function code is not being executed. Most major public cloud vendors provide FaaS computing platforms including AWS Lambda [6], Google Cloud Functions [7], Microsoft Azure Functions [8] and IBM Cloud Functions [9].

Our proof of concept case study is an RNA sequencing (RNA-seq) pipeline used by the Drug Toxicity Signature Generation center (DToxS) at Mount Sinai [10] to quantify gene expression. Xiong *et al.* developed a 3'-end directed library approach that uses unique molecular identifiers (UMI) to quantify gene expression of cell lines treated with drugs for 48 hours on 96 well plates [10]. Their goal is to identify gene signatures for drug-induced cardiotoxicity by performing three types of drug treatments: protein kinase inhibitors that can cause cardiotoxicity, non-cardiotoxic drugs and vehicle control. This pipeline consists of 3 computational steps. The first step is a de-multiplexing step that separates the reads from the 96 originating wells. These reads are then aligned to the human transcriptome using the Burrows-Wheeler Aligner (BWA) [11]. The resulting alignments are merged and de-duped using Unique Molecular Identifiers (UMIs) to compute the observed counts for each transcript. The de-multiplexing and merge steps of the original pipeline were written in Python and took 29 hours on an AWS EC2 instance m4.4xlarge to process a dataset consisting of 600 million reads. We have converted and optimized the Python steps using C++ executables to reduce the total time of pipeline execution to 3.5 hours when using 16 threads [12]. By using AWS Lambda functions, we reduced the alignment time to 2 minutes, and the entire processing time to approximately 18 minutes.

## Results

To take full advantage of the serverless computing approach (see **Figure 1**) we modified the RNA-seq pipeline to function within the limitations of the Lambda functions (512MB of disk space, 15 minutes maximum runtime, 3GB RAM, and 2 vCPUs) to minimize disk transfers to and from the AWS Simple Storage Service (S3). The major set of files that need to be copied to each Lambda function, other than the input fastq files, are the human reference files. BWA does not actually use the sequence after the indices are generated but only requires the sequence file name to infer the names of index files. To save space, we provided a dummy sequence file for BWA. Since Lambda functions can retain their files after termination, we checked before downloading whether the files already existed to save bandwidth. The Lambda handler script, executables, and support files require 250 MB leaving 250 MB for the input and output files. The pipeline was modified to produce smaller input and output files for the alignment step. The demultiplex and split step was modified to produce input fastq files with a user-specified maximum size. To save execution time transfer of these files to S3 proceeds as soon as the file is created rather than waiting for all files to be generated. Instead of saving large SAM or BAM files after alignment, we wrote a small executable to hash the alignment and UMI barcode of each read into an 8-byte value. The resulting file takes up 50x less space than a SAM file. Not only does this approach conserve disk space, but it also reduces the size of files transferred to and from S3 before the final merge step. Before these modifications, execution required almost 3 hours, largely due to the time required for disk transfers to and from S3. The final optimized pipeline can be executed in 18 minutes.  See **Table 1** for the detailed run time for each of the steps shown in Figure 1 using an AWS EC2 instance (m4.4xlarge with 16 threads).

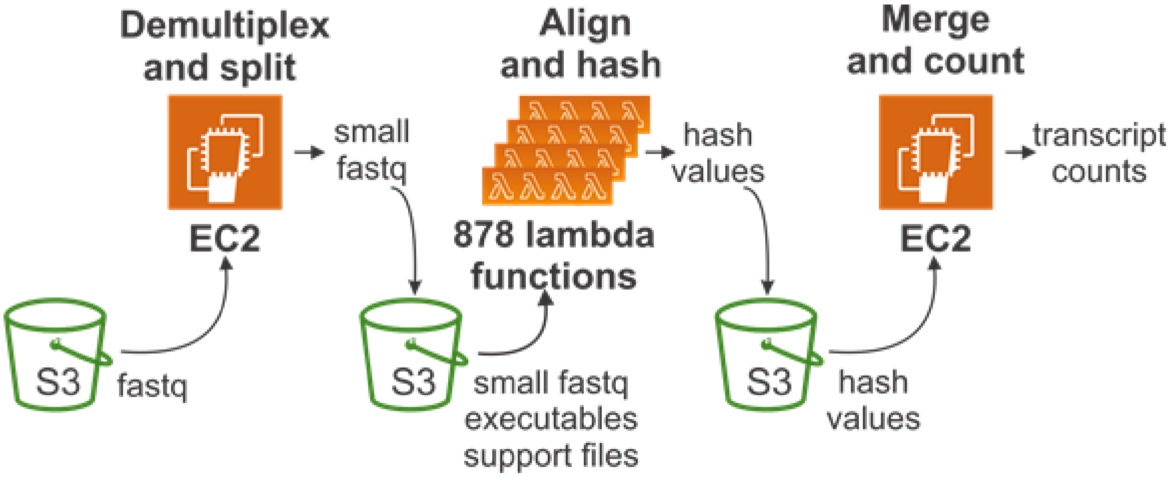

**Figure 1. RNA-seq pipeline using serverless computing.**

The computational and data transfer steps involved in executing the RNA-seq pipeline are shown. The pipeline begins with fastq files comprising 640 million reads (approximately 46 GB total in size) in S3. The Python Lambda handler script, executables, and support files were also on S3. The fastq files are copied to the local disk of an EC2 instance. The files are demultiplexed and split into 150 MB chunks and copied back to S3. 878 Lambda functions, one per fastq chunk were then launched simultaneously. Each Lambda function executes the Python Lambda handler script which copies one of the split fastq files, as well as the executables and support files to the Lambda local disk. The *bwa aln* executable is then called to align the reads. The alignments are piped to a small C++ executable which extracts the transcript ID, the mapping position, and the UMI barcode for each read and produces an 8-byte hash value. These values are written to the local disk of the Lambda function when alignment is performed, producing a file roughly 50x smaller than a SAM file. When the alignment completes the hash file is copied to S3. The final merge step takes place on the EC2 instance. The hash files are copied from S3 to the EC2 instance where the hash values are combined into a unique set to compile the final transcript counts. The cost of the serverless component is $2.82, and the cost of the entire analyses is $3.13 to $3.83 depending on the type of EC2 spot instance used. This translates to a cost of 3 to 4 cents per multiplexed sample.

| Run | Upload fastq data from S3 to EC2 | Split (EC2) | Data transfer from EC2 to S3 | Align (Lambda) | Data transfer SAF files from S3 to EC2 | Merge (EC2) | Total |
|---|---|---|---|---|---|---|---|
| 1 | 0:03:24 | 0:06:50 | 0:02:20 | 0:02:38 | 0:00:15 | 0:01:49 | 0:17:16 |
| 2 | 0:03:24 | 0:07:34 | 0:02:27 | 0:02:43 | 0:00:15 | 0:02:08 | 0:18:31 |
| 3 | 0:03:23 | 0:07:07 | 0:03:37 | 0:02:47 | 0:00:16 | 0:01:49 | 0:18:59 |
| **Median** | 0:03:24 | 0:07:07 | 0:02:27 | 0:02:43 | 0:00:15 | 0:01:49 | 0:17:45 |

**Table 1**. Runtime of each of the steps in our hybrid cloud architecture. The split and merge steps were performed on an AWS m4.4xlarge EC2 instance with a fast i01 SSD while the align step was performed on AWS Lambda.  Compared to our single threaded results using the same optimized implementation on a m4.4xlarge from a slower General Purpose SSD (gp2) reported in Hung et al [12],  split time was reduced from  3 hours to 7 minutes, align times from 17 hours to 2.5 minutes, merge times 50 minutes to 2 minutes. The total pipeline time was reduced from more than 21 hours to 18 minutes, a more than 70x speedup. The speedup vs the reported execution times on 16 threads was still 12x.

## Discussion

Our case study demonstrates how a computationally intensive bioinformatics workflow can leverage a hybrid cloud architecture that incorporates serverless cloud computing to dramatically speed up execution time at a low cost. Bioinformatics workflows typically consist of multiple components. Compute bound tasks can leverage the highly scalable and performance efficient Function-as-a-service cloud serverless platforms whereas the disk bound tasks can benefit from instances with faster disks. In our case study, we used a hybrid cloud architecture where the disk bound tasks (split and merge) are deployed on an AWS EC2 instance with a fast SSD, while the parallelizable compute bound task (alignment using BWA) is deployed on AWS Lambda. The pipeline that we have described is freely available (https://github.com/BioDepot/RNA-seq-lambda) and applicable to any set of UMI RNA-seq data. The RAM and memory limitations of Lambda functions are not impediments to some computational bioinformatics applications, such as alignments to smaller references, or all-vs-all calculations of Smith-Waterman mapping distance in a set of proteins. In general, tasks can be split into smaller sub-tasks and distributed for execution (scattered) on Lambda functions and results assembled (gathered) and merged (reduced) to obtain the final result. Fortunately, many CPU-bound applications also have a GPU implementation, which involves a large task divided into sub-tasks that are processed by small groups of GPU compute units with very limited resources. Sequence alignment, protein-folding, and deep-learning are computationally intensive bioinformatics tasks that can be potentially adapted to leverage a serverless computing approach. With the capability of serverless cloud computing to quickly leverage hundreds of CPU cores, the computational power that was once the exclusive domain of supercomputers is now easily accessible, available on demand, and at low cost, to enable solving many resource intensive bioinformatics problems.

## Software and Data Availability

The code and workflow are publicly available under a M.I.T. license (https://github.com/BioDepot/RNA-seq-lambda). The processed UMI RNA-seq data are publicly available at https://martip03.u.hpc.mssm.edu/data.php and the raw fastq files are available upon request.

## FUNDING

L.H.H., N.X., W.L. and K.Y.Y. are supported by NIH grant R01GM126019. W.L. is supported by the NSF grant OAC-1849970. We would like to acknowledge support from the AWS Cloud Credits for Research (to Hung, Lloyd and Yeung) for providing cloud computing resources.

## ACKNOWLEDGEMENT

We thank Drs. Eric Sobie and Yuguang Xiong at Ichahn School of Medicine at Mount Sinai for providing the DToxS RNA sequencing data.

## AUTHOR CONTRIBUTIONS

L.H.H., D.K. and X.N. performed the empirical experiments and collected results from the empirical studies. L.H.H. designed the case study, wrote and optimized the workflow and helped to write the manuscript. W.L. contributed key ideas on optimizing the workflow using serverless computing and helped to write the manuscript. K.Y.Y. drafted the manuscript, helped with analyzing results from the empirical studies and directed the study. L.H.H., W.L. and K.Y.Y. all worked closely with D.K. and X.N.

## References

1. Langmead, B. & Nellore, A. Cloud computing for genomic data analysis and collaboration. *Nat Rev Genet* **19**, 325 (2018).

2. Calabrese, B. & Cannataro, M. Cloud Computing in Bioinformatics: current solutions and challenges. *PeerJ Preprints* **4**, e2261v2261 (2016).

3. Baldini, I. et al. in Research Advances in Cloud Computing 1-20 (Springer, Singapore, 2017).

4. Lloyd, W., Ramesh, S., Chinthalapati, S., Ly, L. & Pallickara, S. in IEEE International Conference on Cloud Engineering (IC2E) 159-169 (IEEE, 2018).

5. Lynn, T., Rosati, P., Lejeune, A. & Emeakaroha, V. in 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (IEEE, Hong Kong, China; 2017).

6. AWS Lambda. https://aws.amazon.com/lambda/

7. Google Cloud Functions. https://cloud.google.com/functions/

8. Microsoft Azure Functions: Build apps faster with a serverless architecture. https://azure.microsoft.com/en-us/services/functions/

9. IBM Cloud Functions. https://www.ibm.com/cloud/functions

10. Xiong, Y. et al. A Comparison of mRNA Sequencing with Random Primed and 3'-Directed Libraries. *Sci Rep* **7**, 14626 (2017).

11. Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **25**, 1754-1760 (2009).

12. Hung, L.H. et al. Holistic optimization of RNA-seq workflow for multi-threaded environments. *Bioinformatics*, btz169 (2019).