An Investigation on Public Cloud Performance Variation for an RNA Sequencing Workflow

David Perez
School of Engineering and Technology
University of Washington
Tacoma, WA
daperez@uw.edu

Ling-Hong Hung
School of Engineering and Technology
University of Washington
Tacoma, WA
sxu253@uw.edu

Sonia Xu
School of Engineering and Technology
University of Washington
Tacoma, WA
kayee@uw.edu

Ka Yee Yeung
School of Engineering and Technology
University of Washington
Tacoma, WA
kayee@uw.edu

Wes Lloyd
School of Engineering and Technology
University of Washington
Tacoma, WA
wlloyd@uw.edu

ABSTRACT

Public Infrastructure-as-a-Service (IaaS) clouds abstract various details regarding the implementation of resources provided to users. For example, users are not informed about the exact physical location of their virtual machines (VMs), the specific hardware used, the number of co-resident VMs they reside with, or the workloads that co-resident VMs are running. Detecting when VMs underperform can help identify resource contention from co-resident VMs to spur their replacement. Resource utilization metrics can be used to help classify performance of runs for use in VM performance model datasets to sample the distribution of performance outcomes in the cloud. VM performance models are key to predicting the cost of bioinformatics analyses in the public cloud. This paper investigates the performance variations of running a RNA sequencing workflow in the public cloud. We examine causes of performance variations including VM provisioning, CPU heterogeneity, and resource contention. We leverage Amazon Elastic Compute Cloud (EC2) placement groups, a feature designed to help influence VM placement to help examine how VM placement impacts performance variations. As a use case, we investigate the performance of a multi-stage bioinformatics RNA sequencing (RNA-seq) analytical workflow consisting of four distinct phases, executing in 90 minutes on average using 8-core public cloud VMs. In addition, we investigate whether Linux resource utilization metrics collected by profiling workflow runs can help identify performance implications.

ACM-BCB '20, September 21–24, 2020, Atlanta, GA, USA

ACM ISBN 978-1-4503-6666-3/19/09.
https://doi.org/10.1145/3307339.3343465

CCS CONCEPTS

• **Computer systems organization** → *Cloud computing*;

KEYWORDS

Cloud computing, bioinformatics, Infrastructure-as-a-Service

ACM Reference Format:
David Perez, Ling-Hong Hung, Sonia Xu, Ka Yee Yeung, and Wes Lloyd. 2020. An Investigation on Public Cloud Performance Variation for an RNA Sequencing Workflow. In *11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (ACM-BCB '20), September 21–24, 2020, Atlanta, GA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3307339.3343465

1 INTRODUCTION

In public clouds, provisioning variation refers to the random nature of VM placement across physical servers that occurs when cloud providers load balance VM launch requests. Where VMs are hosted on public clouds is abstracted, and is considered a challenge to infer in real-time [3, 7, 9, 13, 15, 17, 19]. Public clouds support features including availability zones, virtual private networks, and placement groups to help consolidate VMs. These features can help influence VM placement relative to other user VMs for application hosting to help improve performance. Though the placement of user VMs can be influenced, shared physical hosts in the cloud can still host many other co-resident VMs that consume an unusual share of CPU, memory, disk, or network resources. In particular, resource contention has been shown to degrade performance of scientific applications hosted in public clouds [11, 14, 16].

CPU heterogeneity occurs when public cloud providers implement the same VM type using physical servers implemented with more than one CPU type. Farley et al. initially noted CPU heterogeneity on Amazon Web Services (AWS) Elastic Compute Cloud (EC2) VMs of the same instance type in [6]. Farley's work focused on the m1.small instance type. In particular, they demonstrated cost savings by discarding VMs with lower performing CPUs of the same instance type. Ou et al. identified heterogeneous VM implementations on multiple public clouds and observed at least four

different Intel Xeon CPUs used to implement the m1.large EC2 instance type producing performance variation of 20% for operating system benchmarks [12]. Lloyd et al. tested 12 different EC2 VM types and found that 25% were implemented with more than one CPU [11]. Lloyd et al. then developed a "trial-and-better" approach where the CPU type of VMs is checked upon launch, and those with lower performing CPUs are terminated and replaced. By leveraging the trial-and-better approach, Lloyd demonstrated potential for up to 14% performance improvement for RESTful environmental modeling web service workloads.

In this paper, we investigate the implications of VM provisioning variation and CPU heterogeneity on the performance of a multi-stage RNA sequencing (RNA-seq) workflow. An RNA-seq workflow consists of multiple computational tasks, each of these tasks could potentially exhibit different performance characteristics. We investigate the performance of running concurrent instances of tasks from this workflow across c5.2xlarge (Intel Xeon) and new c5a.2xlarge (AMD EPYC) EC2 instances equipped with 8 virtual CPUs. Running the RNA-seq workflow concurrently on the cloud is a common scenario for exploratory investigations over genomics data. We leverage EC2 placement groups to control VM placement as much as possible and study runtime implications. Our empirical experiments show that Intel Xeon-based c5 instances, considered the current generation of compute optimized VMs in us-east-2 (Ohio), exhibit CPU heterogeneity. Nearly half of the instances were implemented with the Intel Xeon Platinum 8124M CPU, while the other half used the Intel Xeon Platinum 8275CL CPU. This CPU heterogeneity produced a difference between min/max performance of 20.9% for RNA-seq spanning from a minimum of 55m 33s (8275CL) to a maximum of 67m 11s (8124M). As tasks are deployed thousands of times, this performance variation translates to performance losses and cost increases for big data analyses.

To optimize the performance of genomics workflows on the public cloud, we are interested in developing techniques to automatically identify under-performing VMs in real-time so they can be replaced. Additionally, when profiling resource utilization of tasks to train VM performance models, there is a desire to adequately sample the entire input space that represents the range of possible performance outcomes for a task in the public cloud (e.g. 19.5% for c5.2xlarge). To capture the full spread of possible runtimes, we aim to develop techniques that suggest where a profiling sample lies across the distribution before knowing the distribution. We investigate Linux profiling metric relationships with runtimes of individual tasks to identify relationships to spur this effort.

1.1 Research Questions

This paper investigates the following research questions:

RQ-1: What is the performance variation of running genomics data analytical tasks on the public cloud? How much do factors such as provisioning variation, CPU heterogeneity, and resource contention contribute to performance variation? How does performance compare to analyses on isolated hosts?

RQ-2: Over a 24-hour period, how does performance of individual cloud VMs vary for repeated runs of analytical tasks? What relationships exist between Linux resource utilization metrics (CPU, memory, disk, and network) and task runtimes? Which metrics correlate with runtimes? Can these relationships help infer where a task's runtime lies along the distribution of runtimes for a particular VM?

2 BACKGROUND

2.1 CPU Heterogeneity

Public cloud providers largely have chosen to offer distinct types of VMs to cloud users to simplify the task of resource allocation to users. By fixing VM resources to have distinct quantities of virtual CPUs (vCPUs), memory, storage capacity, and network bandwidth, cloud providers can focus on optimizing hardware to deliver these resources in a highly available and scalable manner. For example, the Amazon, Microsoft, and Google public clouds presently offer more than 265, 204, and 35 fixed VM types each with predefined hardware specifications for the number of vCPUs, RAM size, storage type and capacity, and network bandwidth. As cloud hardware ages, however, cloud providers are forced to replace aging hardware to implement existing VM types with new CPUs. This CPU heterogeneity has been shown to produce performance variation for a variety of application workloads [6, 11, 12].

In this paper we focus on performance analysis of the alignment step for an RNA-seq workflow on c5.2xlarge and c5a.2xlarge AWS EC2 instances. These VMs are equipped with 8 virtual CPUs (vCPUs), 16 GB RAM, EBS storage, and up to 10 Gigabit network throughput. For c5.2xlarge testing, we created 30 VMs, where 16 were randomly implemented with the Intel Xeon Platinum 8124M CPU, and 14 with the Intel Xeon Platinum 8275CL CPU. For c5a.2xlarge testing, we created 30 VMs where all were implemented with the AMD EPYC 7R32 processor. This is to be expected as the c5a instance type was brand new at the time of the study having been released in June 2020 [1]. We provide an overview of the CPUs backing c5.2xlarge and c5a.2xlarge instances, as well as their prevalence statistics for different VM placement strategies in Table 1.

2.2 VM Placement Groups

AWS EC2 offers VM placement groups to help influence the placement of VMs across public cloud physical servers [2, 4]. Options include spread, cluster, and partition placement groups. With spread placement, AWS places instances on distinct servers located on different server racks, where each rack has its own network and power source to maximize dispersion and improve fault tolerance on resource failure. Spread placement is limited to 7 VMs per availability zone for standard user accounts forcing us to launch instances across two availability zones to obtain 10 distinct VM placements in the us-east-2 Ohio region for our experiments. Spread placement guarantees that no two VMs will be co-located with each other. Given that genomics workflows are both CPU and I/O intensive, co-locating all concurrent runs on the same hardware will result in resource contention. Spread placement guarantees user VM's won't interfere with each other, but it does not guarantee resource isolation from other user's VMs. Partition placement is similar to spread placement but allows for more than one VM to exist in each partition allowing for distinct destinations for VMs. Users are limited to 7 partitions per availability zone. We do not investigate

	Intel(R) Xeon(R) Platinum 8124M CPU @ 3.00 GHZ	Intel(R) Xeon(R) Platinum 8275CL CPU @ 3.00 GHZ	AMD EPYC 7R32 CPU @ 2.80 GHz GHZ
EC2 Instance Type	c5.2xlarge	c5.2xlarge	c5a.2xlarge
Family/microns/yr	Skylake/14nm/2017	Cascade Lake/14nm/2019	Rome/7& 14nm/2019
vCPUs/host	72	96	96
Physical CPU cores/host	36	48	48
Burst clock MHz (single/all)	3400/3500	3600/3900	3300/3300
L1 cache (per core)	1.125 MiB (1/2 data, 1/2 instruction)	1.75 MiB (1/2 data, 1/2 instruction)	64K (1/2 data, 1/2 instruction)
L2 cache (per core)	18 MiB	24 MiB	512K
L3 cache (per core)	24.75 MiB	35.75 MiB	16384K
Watts	205	240	280
Total Freq.	53%	47%	100%
Standard Freq.	13%	20%	100%
Cluster Freq.	13%	20%	100%
Spread Freq.	27%	7%	100%

Table 1: EC2 Fifth Generation Compute Optimized Processor Comparison: c5.2xlarge instances (Intel Xeon Platinum 8124M and 8275CL), and c5a.2xlarge instances (AMD EYPC /7R32)

partition placement groups here because RNA-seq workflows run standalone on individual VMs and we do not want to co-schedule concurrent runs on VMs running on the same physical server in a partition.

Cluster placement packs instances close together inside an Availability Zone, to ensure the lowest possible network latency and the highest possible network throughput (up to 10 Gbps) for TCP/IP traffic. Instances in a cluster placement group are placed on the same rack, or on racks close to one another in the cloud data center. Cluster placement for concurrent jobs of the same type may increase resource contention and reduce performance when VMs share the same physical server to run identical tasks. Cluster placement, however, does not guarantee that VMs will share physical servers. VM placement is still subject to provisioning variation and the actual placement of VMs across servers is not disclosed.

3 METHODOLOGY

3.1 UMI RNA-seq Workflow

As a use case we study the performance of the multi-stage unique molecular identifiers (UMI) RNA-seq workflow [18]. To reduce computation time and cost we performed 330 (240 from RQ-1, 90 from RQ-2) workflow runs, we used a partial dataset generated by excluding all but the first million reads from the original FASTQ files. The workflow consisted of 4 distinct phases each requiring different computational resources to execute. Figures 2 and 3 depict the resource utilization of our RNA-seq workflow as measured on the Intel Xeon 8124M and AMD EPYC 7R32 processors. The figures show Linux CPU time accounting metrics to describe the state of the CPU while executing the workflow. These figures enable a comparison of how two different x86-64 processors perform an identical workflow from the perspective of Linux CPU time accounting. In Figure 1 we see the first phase is a download phase where the workflow downloads input data (8 GB of FASTQ files) from the Amazon Simple Storage Service (S3). The second phase is a split step where data demultiplexing is performed. Data is sorted

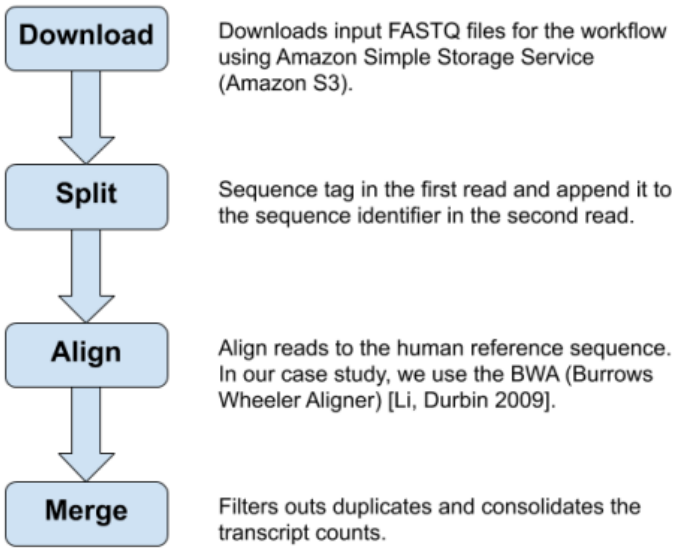

Figure 1: Flow diagram of the UMI RNA-seq workflow [18].

using a sequence barcode to identify the originating sample. The third phase aligns the reads to a human reference sequence using the Burrows Wheeler Aligner (BWA) [10]. The final stage is the "merge" phase which counts all the aligned reads to compute the number of transcripts produced by each gene. Please refer to our previous work [8] for implementation details of these steps. In this paper, we focus on the alignment step as this has been shown to be CPU-intensive [8] and the most time-consuming phase in the entire workflow. Since the alignment to a reference genome is the rate-limiting step in many genomic workflows, we expect that an analyses of factors affecting alignment performance will be of broad interest.

These tasks were deployed on EC2 instances using a Docker container with Ubuntu 16.04 LTS as the base operating system. A VM image was created which included Docker and all required software dependencies for the VMs.

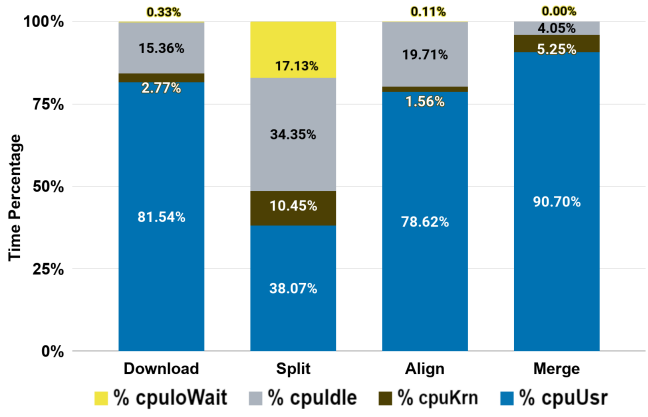

Figure 2: Intel(R) Xeon(R) Platinum 8124M CPU utilization graph for the four phases (download, split, align, and merge) of the RNA-seq workflow (c5.2xlarge). The graph depicts % CPU time in each CPU mode: cpuUsr (CPU user mode), cpuKrn (CPU kernel mode), cpuIdle (CPU idle time), cpuIOWait, and cpuSftIntSrvc (CPU soft interrupt service mode)[5].

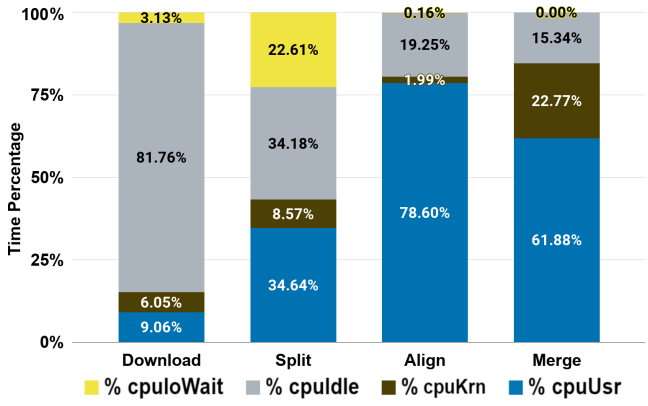

Figure 3: AMD EPYC 7R32 CPU utilization graph for the four phases (download, split, align, and merge) of the RNA-seq workflow (c5a.2xlarge). The graph depicts % CPU time in each CPU mode: cpuUsr (CPU user mode), cpuKrn (CPU kernel mode), cpuIdle (CPU idle time), cpuIOWait, and cpuSftIntSrvc (CPU soft interrupt service mode)[5].

3.2 Container Profiler

To profile resource utilization of the tasks within the UMI RNA-seq workflow in our experiments, the *Container Profiler* tool was used [5]. The *Container Profiler* measures and records resource utilization of any containerized task capturing over 50 individual metrics to characterize CPU, memory, disk, and network utilization at the VM, container, and process levels. The *Container Profiler* can produce resource utilization snapshots at multiple time points, allowing for continuous monitoring of the resources consumed by a container workflow. All experimental data were obtained using the *Container Profiler* including the runtime of each task.

3.3 Cloud Infrastructure for Experiments

To investigate (**RQ-1**), we profiled the performance of tasks within the UMI RNA-seq workflow using 30 x c5.2xlarge and 30 x c5a.2xlarge instances using three different AWS EC2 placement groups to test for performance variations. We launched 10 VMs using each placement group: standard placement (i.e. no strategy, standard VM launch), spread, and cluster. We leveraged 30 instances to run each task in the UMI RNA-seq workflow 3 times each on c5.2xlarge for a total of 90 runs, and 4 times each on c5a.2xlarge for a total of 120 runs. For c5.2xlarge instances we received 16 with the Intel Xeon 8124M processor, and 14 with the Intel Xeon 8275CL processor. All c5a.2xlarge instances were backed with the AMD EPYC 7R32 processor. Processor breakdowns by placement group are shown in Table 1.

To investigate (**RQ-2**), we test for relationships between Linux resource utilization metrics and task runtimes, we launched 16 x c5.2xlarge instances. 9 instances were created using standard placement, and 7 instances were created with cluster placement. For standard placement we received 77.7% Xeon 8124M CPUs, and 22.3% 8275CL CPUs. For cluster placement we received 71.4% Xeon 8124M CPUs, and 28.6% Xeon 8275CL CPUs for a total of 12 x 8124M CPUs and 4 x 8275CL CPUs. Each instance ran the tasks in the UMI RNA-seq workflow 15 times over a 24-hour period for a total of 240 runs. By running 15 consecutive iterations of the tasks on each VM, we sought to observe if task performance was constant or variable over a 24-hour period. To ensure that experimental costs were tractable we leveraged spot instances and restricted the duration to approximately 1-day. This enabled us to monitor performance of VMs that ran RNA-seq repeatedly for a total of 15 runs. During our experiment 4 spot instances were terminated due to capacity demand. Persistently slow VMs, once identified, could then be replaced to improve throughput and runtime to lower cloud computing costs.

4 RESULTS AND DISCUSSION

4.1 RNA-seq Public Cloud Performance Variations

To determine performance variations for each task in the UMI RNA-seq workflow **(RQ-1)**, we profiled 30 runs on c5.2xlarge, and 40 runs on c5a.2xlarge using each VM placement strategy (e.g. standard, spread, and cluster). To assess performance on a dedicated server with no other users, we profiled the performance of 3 runs on a c5.2xlarge EC2 dedicated host, a private isolated cloud server not shared by other users to measure runtime performance when there is no resource contention. At the time when we performed our tests, AWS did not offer a dedicated host to support this test for c5a (AMD EPYC) instances.

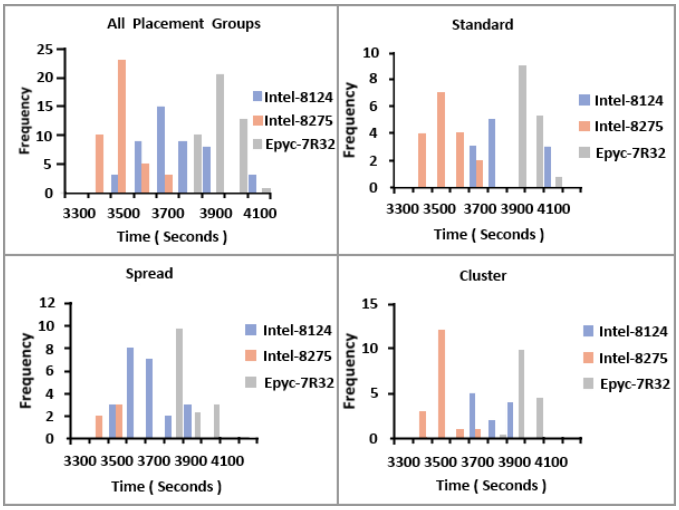

Figure 4: Runtime distribution graphs of the alignment step for c5.2xlarge and c5a.2xlarge instances by VM placement group and processor type. The graphs depict the runtime distribution for the alignment task of the workflow. To normalize the number of samples in the plot, data for c5a.2xlarge AMD EPYC instances is scaled by $\frac{3}{8}$.

Processor	Total	Standard	Cluster	Spread
Intel Xeon 8124M	48	12	24	12
Intel Xeon 8275CL	42	18	6	18
AMD EPYC 7R32	119	40	40	39

Table 2: Number of RNA-seq workflow runs by processor and VM placement strategy

	Standard	Cluster	Spread	Dedicated Host
Max runtime (s)	4031	3890	3828	3463
Min runtime (s)	3668	3643	3590	3461
Average (s)	3788	3749	3628	3462
(%) variation	9.58%	6.59%	6.56%	0.06%
CV	3.70%	2.47%	2.78%	0.03%

Table 3: c5.2xlarge (Intel Xeon 8124M) Runtime performance of the alignment step based on VM placement strategy.

	Standard	Cluster	Spread
Max runtime (sec)	3674	3611	3413
Min runtime (sec)	3358	3367	3333
Average (sec)	3470	3467	3383
(%) Runtime Variation	9.11%	7.04%	2.37%
Coefficient of Variation	2.62%	1.88%	1.12%

Table 4: c5.2xlarge (Intel Xeon 8275CL) Runtime performance of the alignment step based on VM placement strategy.

	Standard	Cluster	Spread
Max runtime (sec)	3957	3912	3923
Min runtime (sec)	3752	3732	3713
Average (sec)	3829	3825	3771
(%) Runtime Variation	5.35%	4.71%	5.57%
Coefficient of Variation	1.57%	1.21%	1.88%

Table 5: C5a.2xlarge (AMD EPYC 7R32) Runtime performance of the alignment step based on VM placement strategy.

Figure 4 depicts the runtime distributions of RNA-seq for each placement group and processor for the alignment step. Most runs on the 8275CL outperform the 8124M and EPYC 7R32 processors. Figure 4 depicts the challenge of capturing training data for VM performance models. CPU heterogeneity increases the sample space of the distribution. Statistically, instance types with heterogeneous CPUs are best handled by separating data for each CPU into separate distribution curves. Generalized cloud performance models should consider the probability of obtaining a particular CPU. Figure 4 also illustrates the distribution of processors for each VM placement strategy. For example, using spread placement we obtained significantly more 8124M processors than 8275CL processors when launching VMs.

Table 2 enumerates all profiling runs conducted to evaluate alignment step performance variation to investigate **RQ-1**. Table 3 and 4 detail runtime statistics of the alignment step on the Intel Xeon 8124M and 8275CL CPUs. Table 5 provides runtime statistics of the alignment step on the AMD EPYC 7R32 CPU. All tables include the percent runtime variation which capture the differences between the minimum and maximum. The tables also include the coefficient of variation (CV), which provides as a percentage, a standardized measure of dispersion describing the extent of variation of the runtimes equal to the standard deviation over the mean. The tables also include the minimum, maximum, and average runtime of the alignment step for each respective configuration. We also measured the alignment performance in isolation using an EC2 dedicated host. EC2 dedicated hosts can be rented to provide access to an entire private isolated server in the public cloud. This allows benchmarking performance of workloads where resource contention from other cloud users is not present. We rented a C5 dedicated host based by the Intel Xeon 8124M processor and detail our performance measurements in Table 3. Using an isolated host reduced runtime compared to standard VMs with an Intel Xeon 8124M processor by 8.6% on average, and by 14.1% in the extreme case.

For Intel Xeon processors, the runtime distribution was the greatest when creating an instance with standard VM placement (9.58% 8124M) in the public cloud, and smallest with spread placement (2.37% 8275CL). CV was also greatest for standard cloud placement (3.7%). Spread placement provided the lowest average runtime for both Intel Xeon CPUs. These results demonstrate up to a 20.9% performance variation for RNA-seq on the c5.2xlarge EC2 instance type with differences explained by CPU heterogeneity (8124M vs.

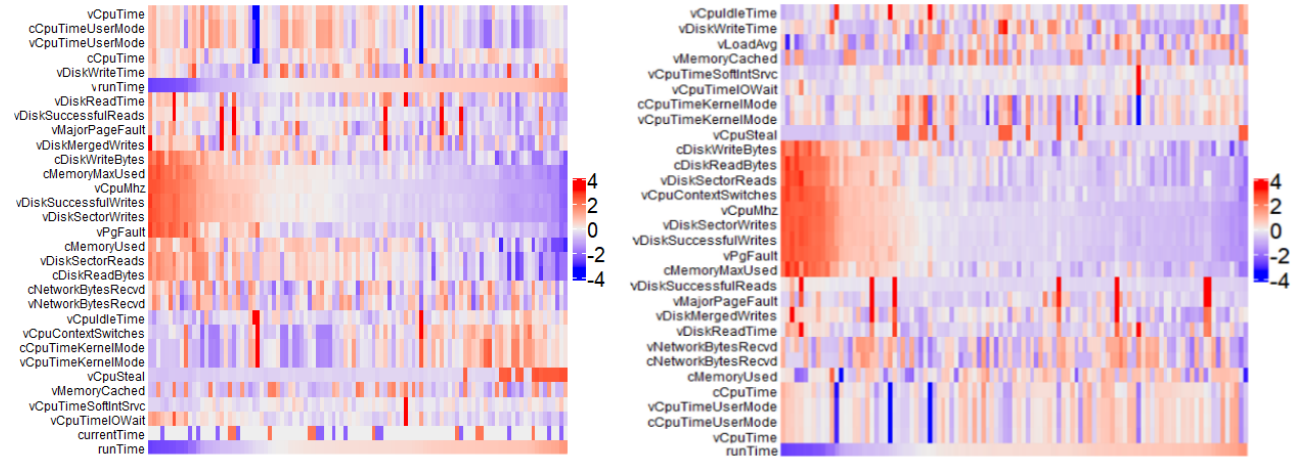

Figure 5: Resource utilization heatmaps (8124M CPU) for the full RNA-seq workflow (left) and only the alignment step (right) with clustered (ordered) rows. Negative correlations between profiling metrics and runtime can be seen. Columns depict 180 individual workflow runs from left to right sorted by increasing runtime.

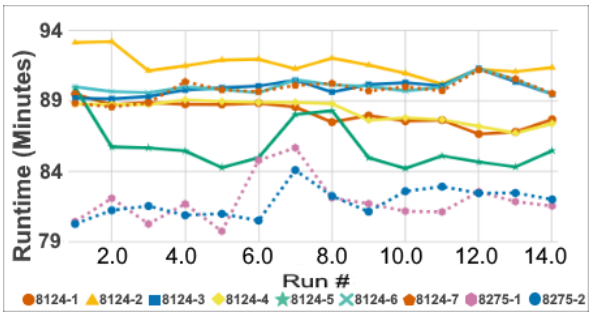

Figure 6: 24-hour RNA-seq runtime performance change on c5.2xlarge instances created with no placement strategy with the Intel Xeon 8124M and 8275CL processors. The X-axis is of runs of the workflow, and the Y-axis is the combined workflow runtime for all four tasks (download, split, align, and merge) in minutes. Each VM executed the RNA-seq workflow 14 times in secession over a 24 hour period.

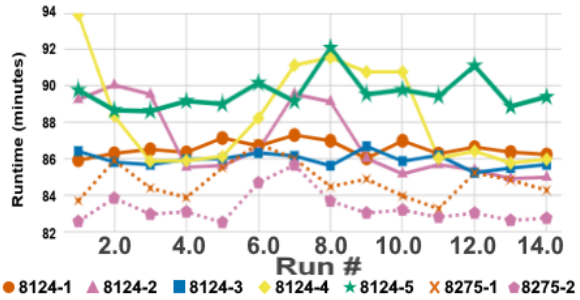

Figure 7: 24-hour RNA-seq runtime performance change on c5.2xlarge instances created with the cluster placement strategy with Intel Xeon 8124M and 8275CL processors. The X-axis is the of runs of the workflow, and the Y-axis is the workflow runtime in minutes. Each VM executed the RNA-seq workflow 14 times in succession over a 24 hour period

8275CL), resource contention (standard vs. dedicated host), and VM placement (standard vs. spread). For c5a.2xlarge instances backed by the AMD EPYC 7R32 processor, we measured approximately half of the performance variation as compared to Intel backed instances. We suspect that at least a portion of the reduction in performance variation can be attributed to fewer users creating resource contention with c5a instances. The c5a instances debuted on June 4, 2020, and our experiments were run on June 23, 2020, a period of less than 3 weeks. In general it may take several months for users to evaluate new VM types and migrate existing workloads once determining a potential for performance improvement or cost savings.

4.2 RNA-seq Performance over 24 hours

To investigate **RQ-2** we profiled how performance of the entire workflow varied over a 24-hour period across 16 x c5.2xlarge EC2 instances. We launched 9 VMs with standard placement, and 7 VMs with cluster placement, receiving 12 x Xeon 8124M processors and 4 x Xeon 8275CL processors. Each VM ran the entire workflow 15 times in succession. Our objective was to investigate whether a VM that produced fast or slow runs initially would sustain the same performance behavior for an entire day. Figure 6 and 7 illustrate our observations of performance variation over 24-hours on c5.2xlarge EC2 instances created using no placement strategy and with the cluster placement strategy respectively. Each line depicts the performance of a specific VM. From the graphs we can see that no VM had perfectly consistent runtime, but we observed that, in general, fast VMs were remained generally fast, and that slow VMs

remained generally slow over 24 hour period. From the figures we do see where a few VMs experienced positive or negative spikes in runtime over the 24 hour period. For these spikes, it is notable that runtimes across multiple VMs trended in the same direction (+/-) at the same time indicating a potentially higher load on the local hardware backing these instances during these times. Spikes from fast to slow(er) were observed on approximately 6 of the 16 EC2 instances. VMs without performance spikes tended to exhibit overall lower runtime performance over 24 hours.

4.3 Resource Utilization Relationships with Workflow Runtime

We next investigated relationships between Linux resource utilization metrics collected by the *Container Profiler* with runtime **(RQ-2)** on c5.2xlarge EC2 instances by analyzing data from 240 runs. 75% of the runs ran on the 8124M CPU, while 25% ran using the 8275CL. We normalized metrics using per minute averages to investigate correlations with runtime. Several metrics had statistically significant negative correlations with runtime ($p<.01$). These correlations include: *(VM metrics)* disk sector reads, CPU context switches, disk sector writes, of successful disk writes, and page faults; *(container metrics)* disk read bytes, and max memory used.

Figure 5 provides two heatmaps to visualize relationships with Linux resource utilization metrics for the entire workflow runtime (left) and the alignment step runtime (right). A cluster of inverse relationships with runtime is seen including *(container metrics)*: cDiskWriteBytes, cDiskReadBytes, cMemoryMaxUsed, and *(VM metrics)*: vDiskSectorReads, vCpuMhz, vCpuContextSwitches, vDiskSectorWrites, vDiskSuccessfulWrites, and vPgFaults. As future work, we will investigate the use of machine learning approaches to characterize VM performance using resource utilization metrics as features.

5 CONCLUSIONS

In this paper, we investigated two core research questions **(RQ-1)** how much performance variation occurs when running genomics data analytical tasks on the public cloud, and **(RQ-2)** what relationships exist between Linux resource utilization metrics and runtime. We leverage AWS placement groups to investigate how changes to VM placement impact the performance of genomics analysis. We also identified heterogeneous CPUs used to implement cloud VMs, and detail the performance implications.

Our research findings include: **(RQ-1)** The performance variation of long running compute-bound tasks on the public cloud were found to be as high as 20.9% using the same instance type (c5.2xlarge) which occurred as a result of CPU heterogeneity, VM placement, and resource contention. The minimum to maximum performance variation for the alignment step spanned from a low of 55 minutes 33 seconds using spread placement (8275CL) to 67 minutes 11 seconds using standard VM placement (8124M). Average performance for the fastest VM placement type provided a 12% performance improvement over average performance of the slowest VM placement type for c5.xlarge, and a 13.1% improvement over the average performance for c5a.2xlarge having standard placement.

For **(RQ-2)**, our results demonstrate performance variation of runs across VMs, while runs across individual VMs appear to have more consistent performance over a 24 hour period. We have also identified a subset of metrics gathered by the *Container Profiler* that were shown to exhibit a strong inverse relationship with runtime.

ACKNOWLEDGMENTS

This research is supported by the US National Science Foundation Advanced Cyberinfrastructure Research Program (OAC-1849970), NIH grants R01GM126019, and R01GM126019-02S2.

REFERENCES

[1] New – Amazon EC2 C5a Instances Powered By 2nd Gen AMD EPYC™ Processors. https://aws.amazon.com/blogs/aws/new-amazon-ec2-c5a-instances-powered-by-2nd-gen-amd-epyc-processors/
[2] AWS. 2020. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html#placement-groups-spread. Last accessed July, 2020.
[3] Adam Bates, Benjamin Mood, Joe Pletcher, Hannah Pruse, Masoud Valafar, and Kevin Butler. 2014. On detecting co-resident cloud instances using network flow watermarking techniques. *International Journal of Information Security* 13, 2 (2014), 171–189.
[4] Brian Posey. 2020. https://awsinsider.net/articles/2017/06/12/ec2-placement-groups.aspx. Last accessed July, 2020.
[5] Huazeng Deng, Ling-Hong Hung, Raymond Schooley, David Perez, Niharika Arumilli, Ka Yee Yeung, and Wes Lloyd. 2020. Profiling Resource Utilization of Bioinformatics Workflows. *arXiv preprint arXiv:2005.11491* (2020).
[6] Benjamin Farley, Ari Juels, Venkatanathan Varadarajan, Thomas Ristenpart, Kevin D Bowers, and Michael M Swift. 2012. More for your money: exploiting performance heterogeneity in public clouds. In *Proceedings of the Third ACM Symposium on Cloud Computing*. 1–14.
[7] Xinlei Han, Raymond Schooley, Delvin Mackenzie, Olaf David, and Wes J Lloyd. 2020. Characterizing Public Cloud Resource Contention to Support Virtual Machine Co-residency Prediction. In *2020 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 162–172.
[8] Ling-Hong Hung, Wes Lloyd, Radhika Agumbe Sridhar, Saranya Devi Athmalingam Ravishankar, Yuguang Xiong, Eric Sobie, and Ka Yee Yeung. 2019. Holistic optimization of an RNA-seq workflow for multi-threaded environments. *Bioinformatics* 35, 20 (2019), 4173–4175.
[9] Mehmet Sinan Inci, Berk Gulmezoglu, Thomas Eisenbarth, and Berk Sunar. 2016. Co-location detection on the cloud. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer, 19–34.
[10] Heng Li and Richard Durbin. 2009. Fast and accurate short read alignment with Burrows–Wheeler transform. *bioinformatics* 25, 14 (2009), 1754–1760.
[11] Wes Lloyd, Shrideep Pallickara, Olaf David, Mazdak Arabi, and Ken Rojas. 2017. Mitigating resource contention and heterogeneity in public clouds for scientific modeling services. In *2017 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 159–166.
[12] Zhonghong Ou, Hao Zhuang, Andrey Lukyanenko, Jukka K Nurminen, Pan Hui, Vladimir Mazalov, and Antti Ylä-Jääski. 2013. Is the same instance type created equal? exploiting heterogeneity of public clouds. *IEEE transactions on cloud computing* 1, 2 (2013), 201–214.
[13] John O'Loughlin and Lee Gillam. 2016. Sibling virtual machine co-location confirmation and avoidance tactics for Public Infrastructure Clouds. *The Journal of Supercomputing* 72, 3 (2016), 961–984.
[14] M Suhail Rehman and Majd F Sakr. 2010. Initial findings for provisioning variation in cloud computing. In *2010 IEEE Second International Conference on Cloud Computing Technology and Science*. IEEE, 473–479.
[15] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. 2009. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM conference on Computer and communications security*. 199–212.
[16] Jörg Schad, Jens Dittrich, and Jorge-Arnulfo Quiané-Ruiz. 2010. Runtime measurements in the cloud: observing, analyzing, and reducing variance. *Proceedings of the VLDB Endowment* 3, 1-2 (2010), 460–471.
[17] Venkatanathan Varadarajan, Thawan Kooburat, Benjamin Farley, Thomas Ristenpart, and Michael M Swift. 2012. Resource-freeing attacks: improve your cloud performance (at your neighbor's expense). In *Proceedings of the 2012 ACM conference on Computer and communications security*. 281–292.
[18] Yuguang Xiong, Magali Soumillon, Jie Wu, Jens Hansen, Bin Hu, Johan Gc Van Hasselt, Gomathi Jayaraman, Ryan Lim, Mehdi Bouhaddou, Loren Ornelas, et al. 2017. A comparison of mRNA sequencing with random primed and 3-directed libraries. *Scientific reports* 7, 1 (2017), 1–12.
[19] Ziye Yang, Haifeng Fang, Yingjun Wu, Chungi Li, Bin Zhao, and H Howie Huang. 2012. Understanding the effects of hypervisor i/o scheduling for virtual machine performance interference. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*. IEEE, 34–41.