Characterizing Performance Variation of Genomic Data Analysis Workflows on the Public Cloud

David Perez, Ling-Hung Hong, Sonia Xu,
Ka Yee Yeung, Wes Lloyd
daperez@uw.edu, wlloyd@uw.edu

August 17-24, 2020

School of Engineering and Technology
University Of Washington, Tacoma

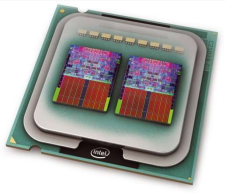
CBDCOM 2020: IEEE International Conference on Cloud and Big Data Computing

1

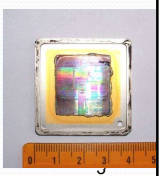
Outline

- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

CPU Heterogeneity

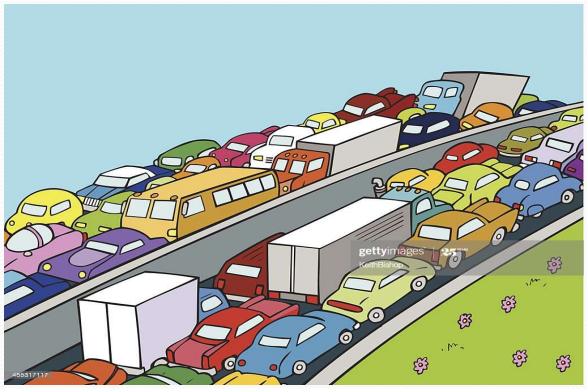


- Public cloud providers offer distinct **VM types** to simplify resource allocation to users
- **VM types:**
  - Have distinct configurations: (e.g. # of virtual CPUs (vCPUs), memory/storage capacity, and network bandwidth)



Resource Contention

- Resource Contention is when there is a competition over shared resources on a shared server



Provisioning Variation

- Provisioning variation is the random nature of VM placement across physical servers that occurs when cloud providers load balance VM launch requests.
- Where these VMs are hosted on public clouds is abstracted and not easily inferable in real time.

Outline

- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

Research Questions

RQ1: What is the performance variation of running genomics data analytical tasks on the public cloud?

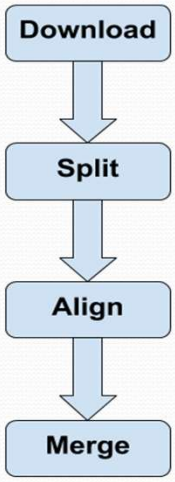
How much do factors such as provisioning variation, CPU heterogeneity, and resource contention contribute to performance variation?

RQ2: What relationships exist between Linux resource utilization metrics (CPU, memory, disk, and network) and workflow runtime?

Outline

- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

Use Case: UMI RNA Sequencing Workflow (Xiong, Yuguang, et al)



Downloads input FASTQ files for the workflow using Amazon Simple Storage Service (Amazon S3).

Sequence tag in the first read and append it to the sequence identifier in the second read.

Align reads to the human reference sequence. In our case study, we use the BWA (Burrows Wheeler Aligner) [Li, Durbin 2009].

Filters outs duplicates and consolidates the transcript counts.

https://www.nature.com/articles/s41598-017-14892-x.pdf

Outline

- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

Container Profiler

The Container Profiler measures and records resource utilization of any containerized task capturing over 50+ Linux system metrics to characterize CPU, memory, disk, and network utilization at the VM, container, and process levels.

These metrics are important as they can help identify what system resources your workflow is consuming the most.

Controlling provisioning variation with AWS EC2 Placement Groups

- **Standard Placement:** No strategy – standard VM launch
- **Spread Placement:** Instances placed on distinct servers located on different server racks.
- **Cluster Placement: I**nstances placed packed together inside an Availability Zone

AWS. 2020. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html Last accessed July, 2020.

Experimental Setup

Using AWS EC2, we provisioned 30 x ec2 c5.2xlarge instances, 10 VMs for each placement strategy:

	Total VMS	Standard	Cluster	Spread
8124M	16	4	4	8
8275CL	14	6	6	2

c5.2xlarge Heterogeneous CPU comparison

	Intel Xeon(R) Platinum 8124M CPU @ 3.00 GHZ	Intel Xeon(R) Platinum 8275CL @ 3.00 GHZ
EC2 Instance Type	c5.2xlarge	c5.2xlarge
Family/microns/yr	Skylake/14nm/2017	Cascade Lake/14nm/2019
Virtual CPU cores/host	72	96
Physical CPU cores/host	36	48
Burst clock MHz (Single/all)	3400/3500	3600/3900
L1 Cache (Per core)	64K (½ data, ½ instruction)	64k (½ data, ½ instruction)
L2 Cache (Per core)	1024K	1024K
L3 Cache (Per core)	1375K	1525K
Total Occurrences:	53%	47%
Standard Placement	13%	20%
Cluster Placement	13%	20%
Spread Placement	27%	7%

https://en.wikipedia.org/wiki/List_of_Intel_Skylake-based_Xeon_microprocessors#Xeon_Platinum_8124M
https://en.wikipedia.org/wiki/List_of_Intel_Cascade_Lake-based_Xeon_microprocessors#Xeon_Platinum_8275CL

Outline

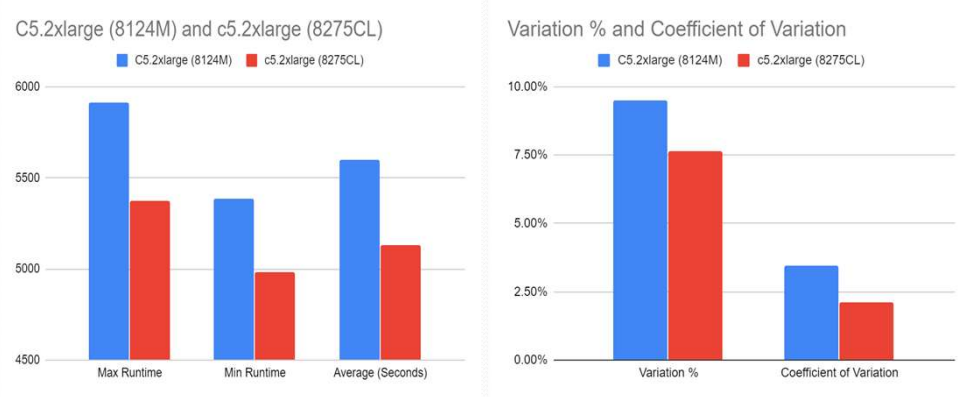
- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

RQ-1: Performance Variation

- What is the performance variation of running genomics data analytical tasks on the public cloud?

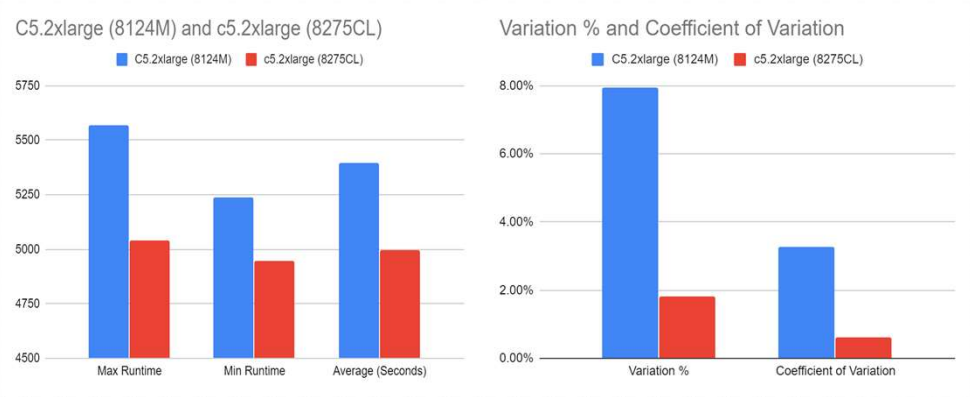
Performance Variation: Standard Placement

CPU runtime variation - c5.2xlarge, Standard placement:



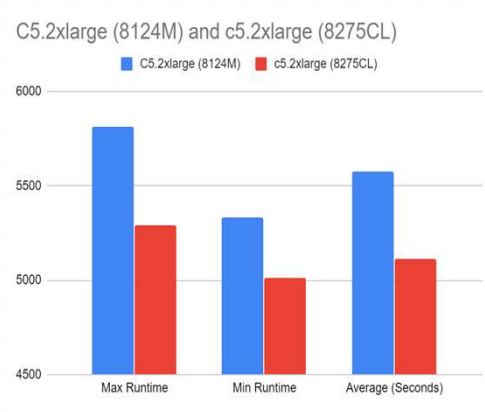
Performance Variation: Spread Placement

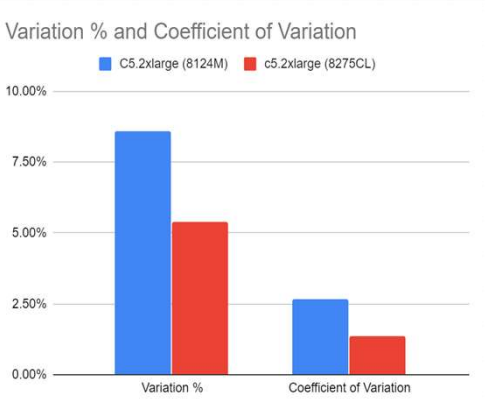
CPU runtime variation - c5.2xlarge, Spread placement:



Performance Variation: Cluster Placement

CPU runtime variation - c5.2xlarge, Cluster placement:



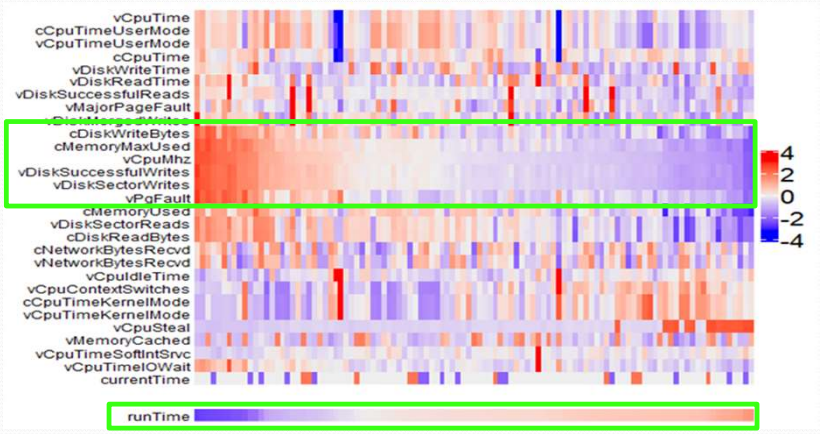


RQ-2: Inferring performance from resource utilization metrics

What relationships exist between Linux resource utilization metrics (CPU, memory, disk, and network) and workflow runtime?

RQ-2: Inferring performance from resource utilization metrics

Resource utilization heatmap using collected data from the Container Profiler with clustered rows.



Outline

- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

Summary

- RQ-1 Performance variation:

  Performance variance of long running compute-bound tasks on were found to be as high as 18.9% and as low as 12.5% using the same instance type (c5.2xlarge).

- RQ-2 Metric relationships with performance:

  A subset of metrics gathered by the Container profiler have been shown to exhibit a strong inverse relationship with runtime.

Outline

- Background
- Research Questions
- Use Case
- Experimental Implementation
- Experimental Results
- Summary
- Conclusions

Conclusions

From RQ-1 we determined when running our genomics data analysis workflow that:

- Spread is fastest and most consistent, with the fastest possible runtime.
- Standard is the slowest, least consistent, with the worst possible runtimes.
- Cluster is middle of the pack.

From RQ-2 we determined when running our genomics data analysis workflow that:

- cDiskWriteBytes, cMemoryMaxUsed, vCpuMhz, vDiskSuccessfulWrites, vDiskSectorWrites, vPgFaults have an inverse relationship to runtime.
- For future work we can use these metrics as candidates for categorizing whether a VM is slow, typical or fast.

THANK YOU FOR WATCHING

- Questions or Comments?
- Please Email:
- daperez@uw.edu or wlloyd@uw.edu