Leveraging Serverless Computing to Improve Performance for Sequence Comparison

Xingzhi Niu, Dimitar Kumanov, Ling-Hong Hung, Wes Lloyd, Ka Yee Yeung
School of Engineering and Technology,
University of Washington Tacoma

Sept 7, 2019

Outline

- Introduction on serverless computing
- Protein sequence alignment as an example
- Serverless pipeline architecture
- Experiment and results
- Cloud provider comparison: AWS vs. Google
- Summary

Why Serverless Computing

Advantages

- No Setup
- High Availability
- Pay only for actual run time
- Scalability

Limitations

- Memory
- Running Time
- Disk
- Languages
- Code Size

FaaS Platforms

- AWS Lambda
- Azure Functions
- IBM Cloud Functions
- Google Cloud Functions

Commercial

Open Source

- Apache OpenWhisk
- Fn (Oracle)

Smith–Waterman (Dynamic Programming)

- gap penalty $W_k = uk + v$ with u=10, v=1
- u gap extension penalty
- v gap opening penalty
- scoring matrix (BLOSUM50) by default

$O(n^3)$

From: https://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm *Identification of Common Molecular Subsequences*, Smith and Waterman, 1981

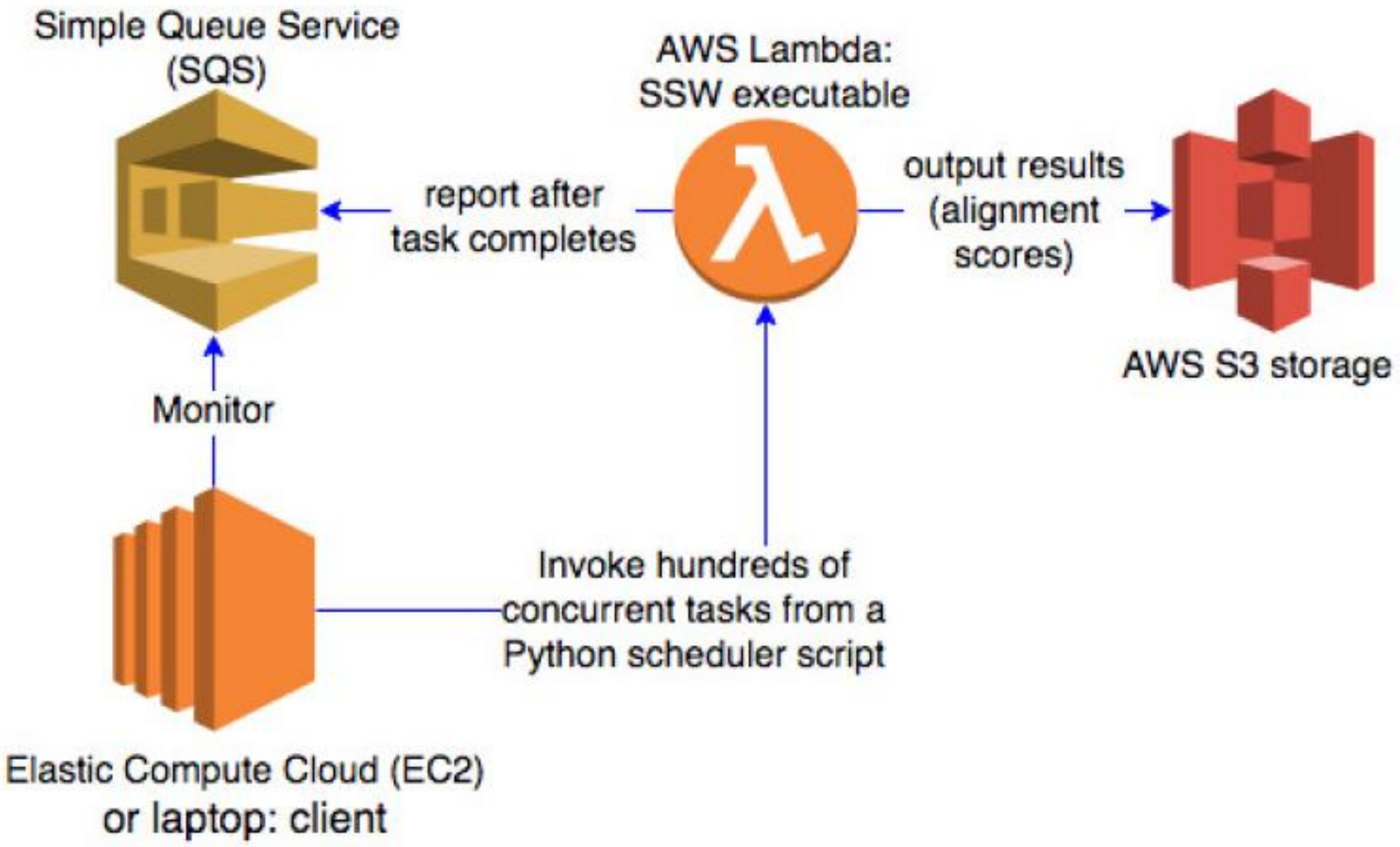
Protein Sequence Alignment

- Smith-Waterman Algorithm
  - 2007 Farrar Striped Algorithm (SSW library):

    https://github.com/mengyao/complete-striped-smith-waterman-library

    Striped Smith-Waterman speeds database searches six times over other SIMD implementations, Farrar, 2007
- Partition 20,336 unique human protein sequences into 41 subsets
- 861 pairwise comparison tasks (41 * 40 / 2 + 41)

Our Serverless Architecture


Simple Queue Service (SQS)
AWS Lambda: SSW executable
report after task completes
output results (alignment scores)
AWS S3 storage
Monitor
Invoke hundreds of concurrent tasks from a Python scheduler script
Elastic Compute Cloud (EC2) or laptop: client

Serverless Pipeline Components

- Corresponding components across cloud providers

Provider	amazon webservices™	Google Cloud
Function	Amazon Lambda	Google Cloud Functions
Notification	amazon SQS	Google Cloud Pub/Sub
VM	Amazon EC2	Google Compute Engine
Storage	amazon S3	Google Cloud Storage

Experiment: Benchmarking Different Clients

Client	OS	Memory
Local Laptop	Ubuntu 16.04	4GB
AWS m5.2xlarge	Ubuntu 18.04	32GB
Google n1-standard-8	Ubuntu 18.04	30GB
FaaS Platform	**Memory**	**Timeout**
AWS Lambda	2GB	540s
Google Cloud Functions	2GB	540s

Configurations Tested (one client thread)

- **Local Laptop client w/&w/o both FaaS providers**
- **AWS VM client w/&w/o AWS Lambda**
- **Google VM client w/&w/o Google Cloud Functions**

Execution Time (Speedup)

Client Type	AWS	Google
Laptop w/o serverless	8h:42m:0s (1x)	
Cloud VM w/o serverless	4h:17m:16s (2.0x)	5h:2m:25s (1.7x)
Laptop + serverless	0h:2m:32s (206.1x)	0h:11m:49s (44.2x)
Cloud VM + serverless	0h:1m:17s (406.8x)	0h:11m:6s (47.0x)

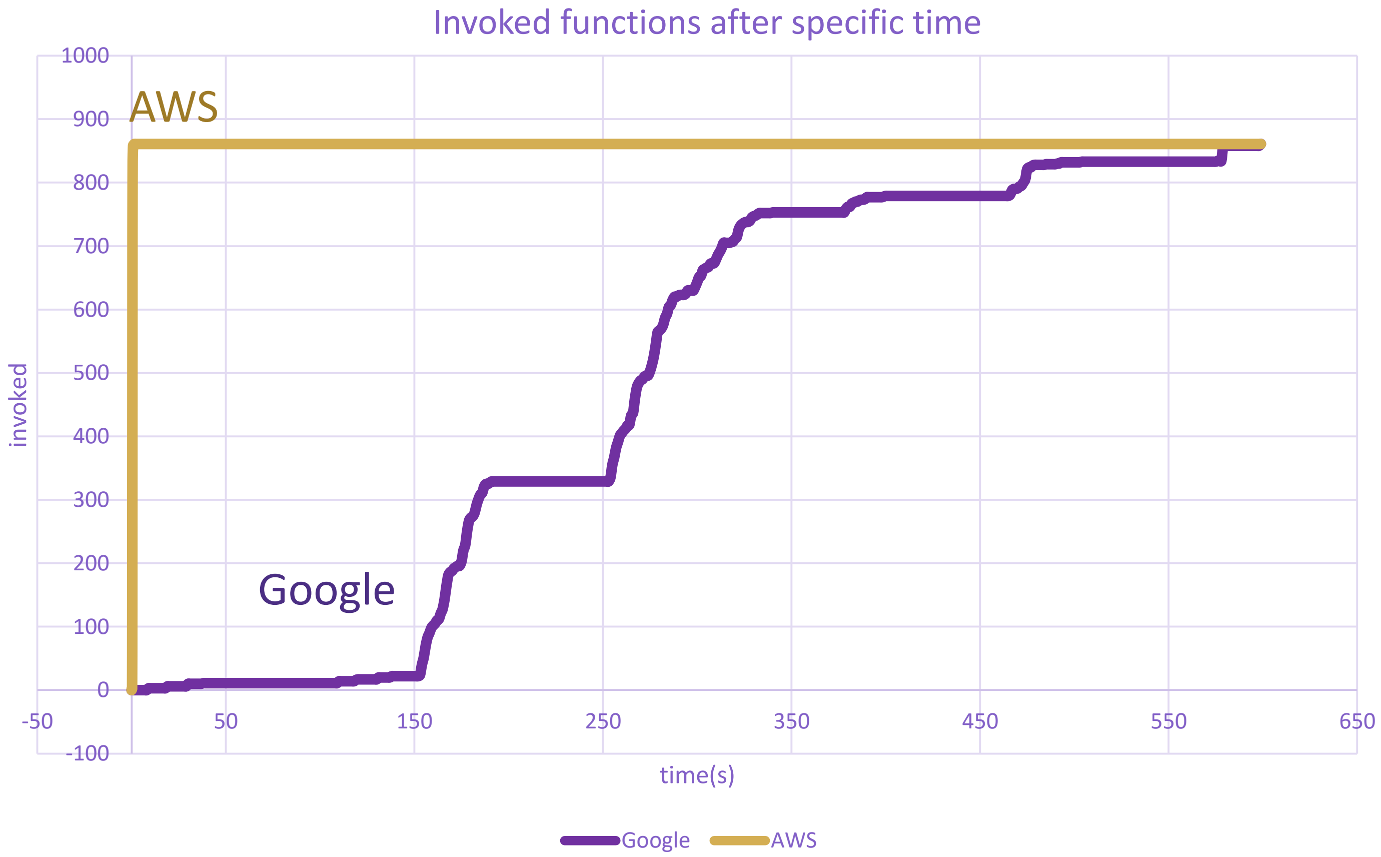
Price Comparison

Client Type	AWS	Google
Laptop w/o serverless	N/A	
Cloud VM w/o serverless	$1.65	$1.82
Laptop + serverless	$0.85	$0.76
Cloud VM + serverless	$0.89	$0.79

AWS vs. Google Comparison

Benchmarking metrics	AWS	Google
Average Function Run Time	76s	67s
Total Invocation Time	1.4s	599s
Maximum Deployment size	50MB	500MB, <100MB each file
Deployment time	<2s	~2min

Details on Invocation Rate



Summary

- Leveraging serverless computing to improve sequence comparison workflows
- Experiments on Smith-Waterman algorithm with AWS and Google platform
- The advantage on both speed and price of serverless computing
- Comparison between two serverless providers: AWS and Google

Questions

Thank you

Supported by NIH R01GM126019.

UNIVERSITY *of* WASHINGTON