CS 744: PYWREN

Shivaram Venkataraman
Fall 2020
Project checkins due Nov 20th
In-class project presentations
  Dec 8th and Dec 10th

Project grade breakdown
  Intro: 5%
  Mid-semester checkin: 5%
  Presentation: 10%
  Final Report: 10%
NEW HARDWARE MODELS
Serverless Computing

Compute Accelerators

Infiniband Networks

Non-Volatile Memory
SERVERLESS COMPUTING
## MOTIVATION: USABILITY

**EC2 Instances**

**Region/Loc: East (Virginia)**

<table>
<thead>
<tr>
<th>Filter: Min Instance (GB)</th>
<th>Compute Units</th>
<th>Storage</th>
<th>Memory</th>
<th>CPU</th>
<th>Price</th>
<th>Price Tier</th>
<th>MAC</th>
<th>Network Performance</th>
<th>EBS Optimized</th>
<th>Max Bandwidth</th>
<th>VPC Only</th>
<th>Linux On-Demand Cost</th>
<th>Linux Reserved Cost</th>
<th>Windows On-Demand Cost</th>
<th>Windows Reserved Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>m1.large</td>
<td>16 GB</td>
<td>16 GB</td>
<td>0.068</td>
<td>1</td>
<td>Low</td>
<td>30 GB</td>
<td>No</td>
<td>No</td>
<td>$0.028/hour</td>
<td>$0.060/hour</td>
<td>$0.096/hour</td>
<td>$0.100/hour</td>
<td>$0.196/hour</td>
<td>$0.410/hour</td>
<td>$1.190/hour</td>
</tr>
<tr>
<td>m1.xlarge</td>
<td>32 GB</td>
<td>32 GB</td>
<td>0.118</td>
<td>2</td>
<td>Low</td>
<td>60 GB</td>
<td>No</td>
<td>Low</td>
<td>$0.056/hour</td>
<td>$0.110/hour</td>
<td>$0.180/hour</td>
<td>$0.186/hour</td>
<td>$0.360/hour</td>
<td>$0.690/hour</td>
<td>$1.930/hour</td>
</tr>
<tr>
<td>m3.large</td>
<td>48 GB</td>
<td>48 GB</td>
<td>0.250</td>
<td>4</td>
<td>Low</td>
<td>120 GB</td>
<td>No</td>
<td>No</td>
<td>$0.080/hour</td>
<td>$0.152/hour</td>
<td>$0.260/hour</td>
<td>$0.268/hour</td>
<td>$0.492/hour</td>
<td>$0.970/hour</td>
<td>$2.910/hour</td>
</tr>
<tr>
<td>m4.large</td>
<td>64 GB</td>
<td>64 GB</td>
<td>0.322</td>
<td>8</td>
<td>Low</td>
<td>240 GB</td>
<td>No</td>
<td>No</td>
<td>$0.128/hour</td>
<td>$0.225/hour</td>
<td>$0.400/hour</td>
<td>$0.410/hour</td>
<td>$0.780/hour</td>
<td>$1.530/hour</td>
<td>$4.560/hour</td>
</tr>
<tr>
<td>m5.large</td>
<td>128 GB</td>
<td>128 GB</td>
<td>0.645</td>
<td>16</td>
<td>Low</td>
<td>480 GB</td>
<td>No</td>
<td>No</td>
<td>$0.256/hour</td>
<td>$0.450/hour</td>
<td>$0.800/hour</td>
<td>$0.820/hour</td>
<td>$1.590/hour</td>
<td>$3.100/hour</td>
<td>$9.100/hour</td>
</tr>
<tr>
<td>m5.xlarge</td>
<td>256 GB</td>
<td>256 GB</td>
<td>1.285</td>
<td>32</td>
<td>Low</td>
<td>960 GB</td>
<td>No</td>
<td>No</td>
<td>$0.512/hour</td>
<td>$0.900/hour</td>
<td>$2.000/hour</td>
<td>$2.020/hour</td>
<td>$3.980/hour</td>
<td>$7.700/hour</td>
<td>$22.700/hour</td>
</tr>
<tr>
<td>m5a.large</td>
<td>256 GB</td>
<td>256 GB</td>
<td>1.285</td>
<td>32</td>
<td>Low</td>
<td>960 GB</td>
<td>No</td>
<td>No</td>
<td>$0.512/hour</td>
<td>$0.900/hour</td>
<td>$2.000/hour</td>
<td>$2.020/hour</td>
<td>$3.980/hour</td>
<td>$7.700/hour</td>
<td>$22.700/hour</td>
</tr>
<tr>
<td>m5ad.large</td>
<td>512 GB</td>
<td>512 GB</td>
<td>2.525</td>
<td>64</td>
<td>Low</td>
<td>1920 GB</td>
<td>No</td>
<td>No</td>
<td>$1.024/hour</td>
<td>$1.800/hour</td>
<td>$4.000/hour</td>
<td>$4.040/hour</td>
<td>$7.960/hour</td>
<td>$15.400/hour</td>
<td>$45.400/hour</td>
</tr>
<tr>
<td>m5a.ad_large</td>
<td>512 GB</td>
<td>512 GB</td>
<td>2.525</td>
<td>64</td>
<td>Low</td>
<td>1920 GB</td>
<td>No</td>
<td>No</td>
<td>$1.024/hour</td>
<td>$1.800/hour</td>
<td>$4.000/hour</td>
<td>$4.040/hour</td>
<td>$7.960/hour</td>
<td>$15.400/hour</td>
<td>$45.400/hour</td>
</tr>
<tr>
<td>m6.large</td>
<td>128 GB</td>
<td>128 GB</td>
<td>0.322</td>
<td>8</td>
<td>Low</td>
<td>240 GB</td>
<td>No</td>
<td>No</td>
<td>$0.256/hour</td>
<td>$0.450/hour</td>
<td>$0.800/hour</td>
<td>$0.820/hour</td>
<td>$1.590/hour</td>
<td>$3.100/hour</td>
<td>$9.100/hour</td>
</tr>
<tr>
<td>m6a.large</td>
<td>256 GB</td>
<td>256 GB</td>
<td>0.644</td>
<td>16</td>
<td>Low</td>
<td>480 GB</td>
<td>No</td>
<td>No</td>
<td>$0.512/hour</td>
<td>$0.900/hour</td>
<td>$2.000/hour</td>
<td>$2.020/hour</td>
<td>$3.980/hour</td>
<td>$7.700/hour</td>
<td>$22.700/hour</td>
</tr>
<tr>
<td>m6a.ad_large</td>
<td>512 GB</td>
<td>512 GB</td>
<td>0.644</td>
<td>16</td>
<td>Low</td>
<td>480 GB</td>
<td>No</td>
<td>No</td>
<td>$1.024/hour</td>
<td>$1.800/hour</td>
<td>$4.000/hour</td>
<td>$4.040/hour</td>
<td>$7.960/hour</td>
<td>$15.400/hour</td>
<td>$45.400/hour</td>
</tr>
<tr>
<td>m6i.large</td>
<td>128 GB</td>
<td>128 GB</td>
<td>0.322</td>
<td>8</td>
<td>Low</td>
<td>240 GB</td>
<td>No</td>
<td>No</td>
<td>$0.256/hour</td>
<td>$0.450/hour</td>
<td>$0.800/hour</td>
<td>$0.820/hour</td>
<td>$1.590/hour</td>
<td>$3.100/hour</td>
<td>$9.100/hour</td>
</tr>
<tr>
<td>m6i.ad_large</td>
<td>256 GB</td>
<td>256 GB</td>
<td>0.644</td>
<td>16</td>
<td>Low</td>
<td>480 GB</td>
<td>No</td>
<td>No</td>
<td>$1.024/hour</td>
<td>$1.800/hour</td>
<td>$4.000/hour</td>
<td>$4.040/hour</td>
<td>$7.960/hour</td>
<td>$15.400/hour</td>
<td>$45.400/hour</td>
</tr>
<tr>
<td>m6i.ad_large</td>
<td>512 GB</td>
<td>512 GB</td>
<td>0.644</td>
<td>16</td>
<td>Low</td>
<td>480 GB</td>
<td>No</td>
<td>No</td>
<td>$1.024/hour</td>
<td>$1.800/hour</td>
<td>$4.000/hour</td>
<td>$4.040/hour</td>
<td>$7.960/hour</td>
<td>$15.400/hour</td>
<td>$45.400/hour</td>
</tr>
</tbody>
</table>

### What instance type?

**What base image?**

**How many to spin up?**

**What price? Spot?**
Why is there no "cloud button"?

When to use the Cloud?

Data
- Large amounts of data. Can’t store locally
- Shared data across users
- Long term storage

Compute
- Need lots of CPUs for short jobs
- Varying compute requirements (Yes)
- No admin overhead (Yes)
APPLICATION LEVEL?

- Application
  - Compute Framework
    - Hardware
  - Logistic Regression
    - Spark
      - Amazon EC2
      - CloudLab
      - Private Cluster
      - ...
  - Application
    - Compute Framework
STATELESS DATA PROCESSING

Function Scheduler

Key Value Store (Low Latency)

Blob Store (High Bandwidth)

Containers
“SERVERLESS” COMPUTING

300-900 seconds single-core

512 MB in /tmp

3GB RAM

Python, Java, node.js
import pywren
import numpy as np

def addone(x):
    return x + 1

wrenexec = pywren.default_executor()
xlist = np.arange(10)
futures = wrenexec.map(addone, xlist)

print([f.result() for f in futures])

The output is as expected:

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
PYWREN: HOW IT WORKS

future = runner.map(fn, data)

future.result()

your laptop               the cloud
future = runner.map(fn, data)

Serialize func and data
Put on S3
Invoke Lambda

pull job from s3
download anaconda runtime
python to run code
pickle result
stick in S3

result

poll S3
unpickle and return

your laptop

the cloud
**STATELESS FUNCTIONS: WHY NOW?**

What are the trade-offs?

<table>
<thead>
<tr>
<th>Storage Medium</th>
<th>Write Speed (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD on <code>c3.8xlarge</code></td>
<td>208.73</td>
</tr>
<tr>
<td>SSD on <code>i2.8xlarge</code></td>
<td>460.36</td>
</tr>
<tr>
<td>4 SSDs on <code>i2.8xlarge</code></td>
<td>1768.04</td>
</tr>
<tr>
<td>S3</td>
<td>501.13</td>
</tr>
</tbody>
</table>
PARAMETER SERVERS

Use lambdas to run "workers"

Parameter server as a service?
WHEN SHOULD WE USE SERVERLESS?

Yes!

Maybe not?
Summary

Motivation: Usability of big data analytics
Approach: Language-integrated cloud computing

Features
- Breakdown computation into stateless functions
- Schedule on serverless containers
- Use external storage for state management

Open question on scheduling, overheads
DISCUSSION

https://forms.gle/PAMDKmwHepmPWDrBA
Consider you are a cloud provider (e.g., AWS) implementing support for serverless. What could be some of the new challenges in scheduling these workloads? How would you go about addressing them?
OPEN QUESTIONS

- Scalable scheduling: Low latency with large number of functions?

- Debugging: Correlate events across functions?

- Launch overheads: Fraction of time spent in setup (OpenLambda)

- Resource limits: 15 minute AWS Lambda (Oct 2018)