Overview

• Setup your instance for the demo
  • Information in the handout

• Administrative Issues
  • TA hours, guidelines on Piazza posts

• Last Week’s Reflection
  • Project 1.1, OLI Unit 1, Quiz1

• This Week’s Schedule
  • Deadlines for OLI Unit 2, Module 3 and 4, Project 1.2

• Demo

• Questions
Administrative

• TA office hours are posted on Piazza and Google calendar.

• Suggestions for using Piazza
  • Discussion forum, contribute questions and answers
  • Read the Piazza Post Guidelines (@20) before asking
  • Read Piazza questions & answers carefully to avoid duplicate ones
  • Don’t ask a public question about a quiz question
  • Try to ask a public question if possible
Platforms

• Open Learning Initiative (OLI)
  • Access through Blackboard
  • Contains Units and Quizzes

• Amazon Web Services (AWS) Account
  • Create AWS account (@8)
  • Complete Account Linking Form
  • Receive email request and click link to confirm!

• http://theproject.zone
  • Project write up, submissions and scoreboard
  • Registration Link in Email

• Piazza
  • Discussion forum

• If you do not have access to all of these platforms, please contact us immediately!
Last Week Reflection

• Reading:
  • Unit 1: Introduction to Cloud Computing
    • Module 2: Cloud Building Blocks
  • Quiz 1: Introduction to Cloud Computing

• Project:
  • Project 1.1:
    • Wikipedia Dataset
    • Filtering one hour’s worth of data
FAQ this week, 1

• Q: Service level agreement is for example what a SAAS like Netflix signs with AWS, right? (@165)
• A: An SLA is the entire agreement that specifies
  • what service is to be provided, how it is supported,
  • times, locations, costs, performance,
  • and responsibilities of the parties involved.
• You accepted the an SLA when you created the AWS Account.
• Q: I still don't understand what a service level objective (SLO) is. (@165)
• A: SLOs are specific measurable characteristics of the SLA such as availability, throughput, frequency, response time, or quality.
  • Eg: An SLO might be acceptable downtime of 10 minutes per month
  • This translates into 99.xxxx% uptime requirement
  • AWS guarantees is 99.95% uptime for EC2 instances
FAQ this week, 2

• Q: Does AWS actively monitor and remove malware hosted by it’s users? Are they legally allowed to look inside AMIs? (@165)

• A: It Depends!
  • Amazon can monitor traffic in and out of your resources to detect anomalies (DDoS, Spamming etc.)
  • Your use is governed by Amazon Terms of Service
    • They will contact you in case of a Government Order or Subpoena
    • They could investigate further or delete files if you don’t respond
    • Eg: USA PATRIOT ACT and DMCA

• Join the Discussion on Piazza.
Quiz 1 Clarifications, 1

• Docker and it’s relation to cloud services. (@162)
• The question is specifically asks what kind of service is used to run a Docker app.
• Check out AWS’s Docker Container service: https://aws.amazon.com/ecs/
Quiz 1 Clarifications, 2

• Private and Public Clouds (@170)
• How can private clouds help save money?
  • They enable sharing of resources within the organization.
    • Eg: Operations, Finance, Billing, Inventory can share the same pool of computers if they use a private cloud.
• AWS started as a private cloud
Looking back at Project 1.1

• Loading all the data to memory to filter and process is a bad idea!
  • Recurring theme in the course projects
  • You will see this in 15619 Team Project (ETL)

• Better Approach: Work from disk, build a processing pipeline
  • Write programs that process line by line
  • Unix pipes typically great to achieve this
  • `cat | grep | awk`
Time to compare!

• Java v.s. Python v.s. Bash v.s. others
• Dataset size: ~70 MB
• Filtering
  • Filtering Complexity : O(N)
• Sort
  • Time and space complexities vary
  • Unix sort: MergeSort
  • Java collections sort: Merge/QuickSort
  • Python default sort: TimSort
• Other Aspects
  • Effect of the Garbage Collectors
How did the various approaches do?

Runtime through t1.micro instance

Category 1

Java | Python | Bash
Join the Discussion on Piazza

• Big Question: What is the optimal set of scripts to answer the questions in P1.1? (@91)

• Share your approach and thoughts.

• The insight may help you considerably for the rest of the Projects in the course
This Week’s Schedule

• Complete Unit 2 (Modules 3 & 4)
  • Complete activities on each page
    • In-module activities are not graded
  • If you encounter a bug in the OLI write-up
    • Provide feedback at the end of each OLI page

• Complete Project 1.2 (Using Elastic MapReduce)
  • Submission Deadline, Sunday, Feb 1, 11:59pm ET
TPZ Scoreboard

1.2 Using Amazon's Elastic MapReduce

<table>
<thead>
<tr>
<th>NickName</th>
<th>Q0(20)</th>
<th>Q1(25)</th>
<th>Q2(5)</th>
<th>Q3(15)</th>
<th>Q4(15)</th>
<th>Q5(20)</th>
<th>Q6(20)</th>
<th>Q7(20)</th>
<th>Q8(20)</th>
<th>Q9(20)</th>
<th>Q10(20)</th>
<th>Total</th>
<th>Attempts</th>
<th>Time Stamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>JeevesSobs</td>
<td>0</td>
<td>25</td>
<td>5</td>
<td>15</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>60</td>
<td>2</td>
<td>2015-01-27 06:28:15.203</td>
</tr>
<tr>
<td>FlinkyPiggy</td>
<td>0</td>
<td>25</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>96</td>
<td>2</td>
<td>2015-01-27 07:25:36.716</td>
</tr>
</tbody>
</table>

Showing 1 to 2 of 2 entries
Why Study Data Centers?

• Data centers are your new computers!
• The cloud is the data center!
• Make sure to read and understand the content of Unit 2
  • Equipment in a data center
  • Power, cooling, networking
  • How to design data centers
  • What could break
    • All software layers are on top of physical resources
Module 3: Data Center Trends

• Definition & Origins
  • Infrastructure dedicated to housing computer and networking equipment, including power, cooling, and networking.

• Growth
  • Size (No. of racks and cabinets)
  • Density

• Efficiency
  • Servers
  • Server Components
  • Power
  • Cooling

Facebook data center
Module 4: Data Center Components

• IT Equipment
  • Anything that is mounted in a stack
  • Servers: rack-mounted
    • Motherboard
    • Expansion cards

• Type of Storage
  • Direct attached storage (DAS)
  • Storage area network (SAN)
  • Network attached storage (NAS)

• Networking
  • Ethernet, protocols, etc.

• Facilities
  • Server room
  • Power (distribution)
  • Cooling
  • Safety

Source: http://www.google.com/about/datacenters/efficiency/internal/
Project 1.2

• In Project 1.1, we processed 1 hour of data
  • Slow, took time to process
  • Many minutes to run on a single file
  • How do you filter and sort the data for one month?
    • 720 files total (~ 70 GB compressed, 250 GB uncompressed)

• Parallel & Distributed Processing
  • How about Pthreads/MPI/...?
  • How simple are these frameworks?
  • Need to design many elements from scratch:
    • File Handling
    • Task Management
    • Orchestration
  • Painful. Take 15440/15618 for a taste 😊
Processing Large Files

• When the input is 200MB
  • In memory data structures can be used to process on a single machine
  • HashMaps, Trees, ArrayLists etc.

• When the file size is 200 GB or TB
  • Large-scale data processing
  • Out of memory
  • Slow
  • How would you deal with it?
    • Partition the input?
    • Distribute the work?
    • Coordinate the effort?
    • Aggregate the results?
Introduction to MapReduce

- **Definition**: Programming model for processing large data sets with a parallel, distributed algorithm on a cluster
- **Map**: Extract something you care about
- **Group by key**: Sort and Shuffle
- **Reduce**: Aggregate, summarize, filter or transform
- **Output** the result
MapReduce Example

How many times does the word “apple” appear in all books in Hunt Library?

I heard 6 “Apple”s!
MapReduce Example

What if we want to count the number of times all fruits appeared in these books?

You can have multiple aggregators, each one working on a distinct set of “fruits”.

Orange, 1
Blueberry, 1
Apple, 1

Apple, 1
Apple, 1
Apple, 1
Orange, 1

Apple, 1
Apple, 1
Orange, 1
Blueberry, 1

Apple ?
Orange ?
Blueberry ?
MapReduce Example

Map
- Orange, 1
- Blueberry, 1
- Apple, 1

Shuffle
- Orange, 1
- Orange, 1
- Orange, 1
- Apple, 1
- Apple, 1
- Apple, 1
- Blueberry, 1
- Blueberry, 1
- Blueberry, 1

Reduce
- Apple, 6
- Orange, 3
- Blueberry, 3
MapReduce Example

Map Output / Reduce Input
(K',V')

Output
(K'',V'')

Input (K,V)

Map
Orange,1
Blueberry,1
Blueberry,1
Apple,1

Shuffle
Orange,1
Orange,1
Apple,1
Apple,1
Apple,1
Apple,1
Apple,1
Apple,1

Reduce
Blueberry,1
Blueberry,1
Blueberry,1

Apple 6
Orange 3
Blueberry 3
Steps of MapReduce

• Map
• Shuffle
• Reduce
• Produce final output
Steps of MapReduce

• Map
  • Prepare input for mappers
    • Split input into parts and assign them to mappers
  • Map Tasks
    • Each mapper will work on its portion of the data
    • Output: **key-value pairs**
      • Keys are used in Shuffling and Merge to find the Reducer that handles it
      • Values are messages sent from mapper to reducer
      • e.g. (Apple, 1)
Steps of MapReduce

• Shuffle
  • Group by key: sort the output of mapper by key
    • Split keys and assign them to reducers (based on hashing)
    • Each key will be assigned to exactly one reducer

• Reduce
  • Each reducer will work on one or more keys
  • Input: mapper’s output (key-value pairs)
  • Output: the result needed
    • Different aggregation logic may apply

• Produce final output
  • Collect all output from reducers
  • Sort them by key
Mapreduce: Framework

- MapReduce framework takes care of:
  - Partitioning the input data
  - Scheduling the program’s execution across a set of machines
  - Perform the Group by key (sort & shuffle) step
    - In practice, this is the bottleneck
  - Handling machine failures
  - Manage required inter-machine communication
Parallelism in MapReduce

• Mappers run in parallel, creating different intermediate values from input data
• Reducers also run in parallel, each working on different keys
• However, reducers cannot start until all mappers finish
Example: Friend/Product Suggestion

- Facebook gathers information on your profile and timeline
  - e.g. contact list, messages, direct comments made, page visits, common friends, workplace/residence nearness.
  - This info is dumped into a log or a database
- Analyze this information
  - weighted matrix analysis
  - connections which are above a threshold value are chosen to be shown to the user.
Real Example: Friend/Product Suggestion

Generate key-value pairs:
- Key: Friends pair
- Value: Friends statistics (e.g. common friends)
  - e.g. (Tom, Sara) statistics

Aggregate the statistics value for the same key and output the friends pair if it’s above the threshold
  - e.g. (Tom, Sara)
Project 1.2 Elastic MapReduce

• Processing sequentially can be limiting, we must:
  • aggregate the view counts and
  • generate a daily timeline of page views for each article we are interested in

• Process a large dataset (~70 GB compressed)
• Setup a Streaming Elastic MapReduce job Flow in AWS
• Write simple Mapper and Reducer in the language of your choice

• You will understand some of the key aspects of Elastic MapReduce and run an Elastic MapReduce job flow

• **Note:** For this checkpoint, assign the tag with
  • Key: Project and Value: 1.2 for all resources
How to write the Mappers and Reducers?

• We are working with **Streaming MapReduce in EMR**
• Write individual mapper and reducer programs in the language of your choice
  • The programs must read input files through **stdin**
  • They have to write output through **stdout**
• Example Job Flow: Wordcount provided in Writeup
• Test your program on a local machine before launching a cluster!
  
  `cat input | mapper | sort | reducer > output`

• Mapper, reducer and input data should be in S3
• Launch a cluster to process the data
P1.2 Grading

• P1.2 is 9% of your grade for the course!
• P1.2 code submissions are auto-graded
• Scores will be made available on [http://theproject.zone](http://theproject.zone) after submission.
• We will grade all the code (both auto and manually)
  • Be sure to make your code readable
  • If your code is not well documented and is not readable, we will deduct points
Project 1.2 – Budgets and Penalties

• Configure EMR cluster instances in US-East-1 (N. Virginia)

• Tag all resources with Key: Project and Value: 1.2
  • Before Launching!
  • No tags → 10% grade penalty

• Budget
  • For P1.2, each student’s budget is $15
  • Exceeding $15 → 10% project penalty
  • Exceeding $30 → 100% project penalty

• Be very careful with EMR, very easy to burn through the budget!

• Plagiarism → the lowest penalty is 200% & potential dismissal
How to Work on a Budget

• P1.2 Budget → $15
• You will need to create an EMR cluster
  • EMR has additional hourly cost per instance.
  • Example: 10 x m1.large = 10 x (0.175 + 0.044) = $2.19 per hour!
  • Total time you have: ~ 6.84 hours in this configuration
  • Use any configuration you like, all we want is the answer.
• 25 x m3.2xlarge = 25 x (0.560 + 0.140) = $17.5 per hour!
• **Spot Instances are your friend:**
  • Same cluster @ spot pricing = 10 x (0.0161 + 0.044) = $0.601 per hour!
• Test and Debug first!
  • Local Machine using Unix Pipes first
  • Then use Small Clusters with a part of the data set
• Other Costs to consider:
  • EBS is $0.1 per GB/month
  • Dataset is in S3, S3→EC2 transfers are free.
Demo

• Quick Tour of AWS
  • EMR
  • On-Demand and Spot Instances
  • Billing and Monitoring Costs

• Demo: Wordcount on EMR

• Auto-grader for P1.2
  • How to make a submission
Questions?

• Reminder: Office hours are posted on Piazza.
Upcoming Deadlines

• Quiz 2: Data Centers
  • Quiz Window Opens 02/06/2015 12:01 AM ET
  • Due 02/06/15 11:59 PM ET

• Project 1.2: Introduction to Big Data Analysis
  • Using Elastic MapReduce
  • Due 01/25/15 11:59 PM ET