Principles of Computer Science II

Computational Thinking

Ioannis Chatzigiannakis

Sapienza University of Rome

Lecture 1

Computational Thinking

Wing, J. M. 2006 Computational thinking. CACM 49, 33–35

Computational thinking is taking an approach to solving problems, designing systems and understanding human behaviour that draws on concepts fundamental to computing.

Wing, J. M. 2006 Computational thinking. CACM 49, 33–35

Computational thinking represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.

Wing, J. M. 2006 Computational thinking. CACM 49, 33–35

Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction.

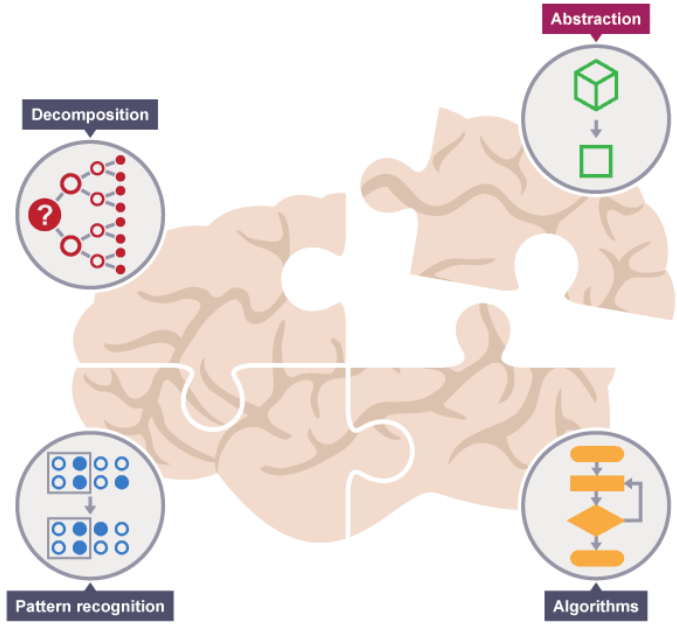
The riddle of machine intelligence

Computational thinking confronts the riddle of machine intelligence:

- What can humans do better than computers?
- What can computers do better than humans?
- What is computable?

Computational Thinking

- Computers are here to help us.
- What do we need from computers?
- What is our problem?
- Computational Thinking allows us to understand what needs to be solved.
- Four key techniques (cornerstones) to computational thinking:
  1. Decomposition – breaking down a complex problem or system into smaller, more manageable parts
  2. Pattern Recognition – looking for similarities among and within problems
  3. Abstraction – focusing on the important information only, ignoring irrelevant detail
  4. Algorithms – developing a step-by-step solution to the problem, or the rules to follow to solve the problem



Computational Thinking vs Programming

Thinking computationally is not programming.

- . . . not even thinking as a computer.
- Programming tells computer what to do / how to do it.
- Computational thinking enables us to understand what we need to tell to computers.
- . . . what to program.

Examples:

- Explain to a friend how to drive to your house
- Organize a party at the park
- Prepare your luggage
- Teach a kid addition/subtraction
- . . .

Decomposition

Turn a complex problem into one we can easily understand.

- . . . probably you already do every day.
- The smaller parts are easier to solve.
- . . . we already know/have the solutions.

Examples:

- Brushing our teeth
  Which brush? How long? How hard? What toothpaste?
- Solving a crime
  What crime? When? Where? Evidence? Witnesses? Recent similar crimes?
- . . .

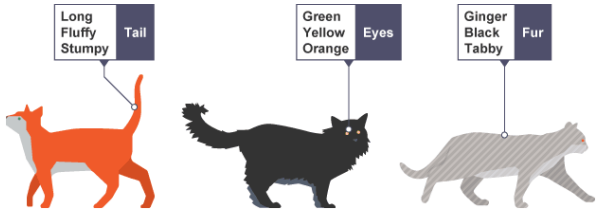
Pattern Recognition

We often find patterns among the smaller problems we examine.

- The patterns are similarities or characteristics that some of the problems share.

Example: Cats

- All cats share common characteristics.
  they all have eyes, tails and fur.
- Once we know how to describe one cat we can describe others, simply by following this pattern.



Abstraction

Hiding irrelevant details to focus on the essential features needed to understand and use a thing

- A compression process – multiple different pieces of constituent data to a single piece of abstract data.
  e.g., "cat"
- Ambiguity – multiple different references.
  e.g., "happiness", "architecture"
- Simplification – no loss of generality
  e.g., "red" - many different things can be red

Thought process wherein ideas are distanced from objects

Abstraction Example: Car vs Car Breaks

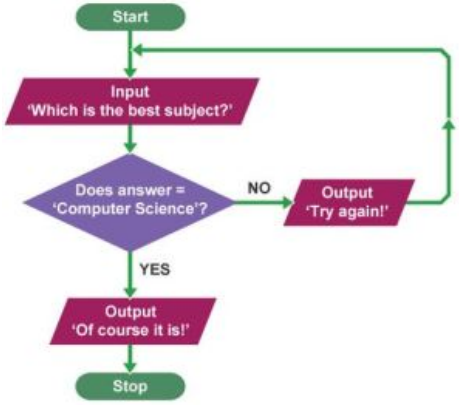
- Do we know how car breaks work?
- Do we know how to use them?

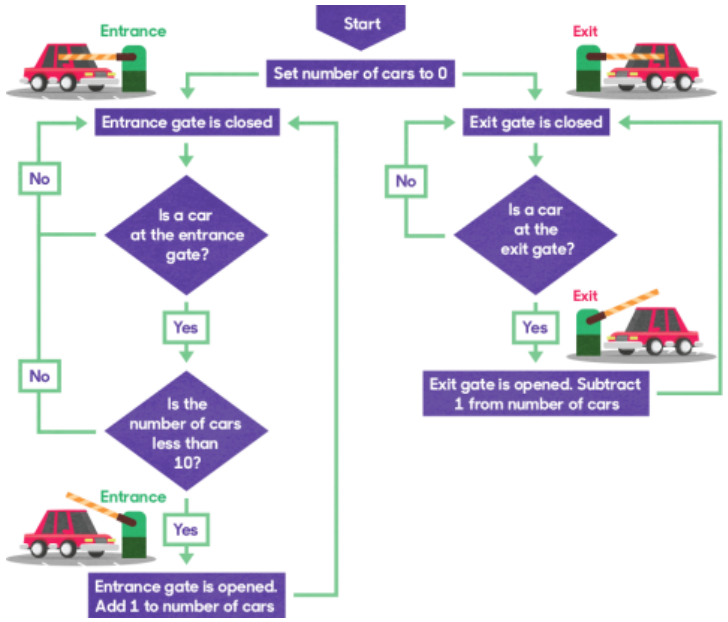
Filter out (ignore) the characteristics that we don't need in order to concentrate on those that we do.

Algorithms

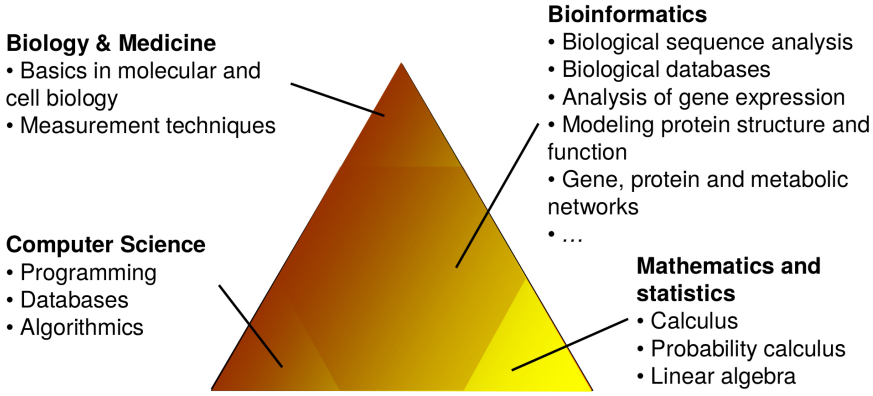
A plan, a set of step-by-step instructions to solve a problem.

- In an algorithm, each instruction is identified and the order in which they should be carried out is planned.





Bioinformatician's skill set



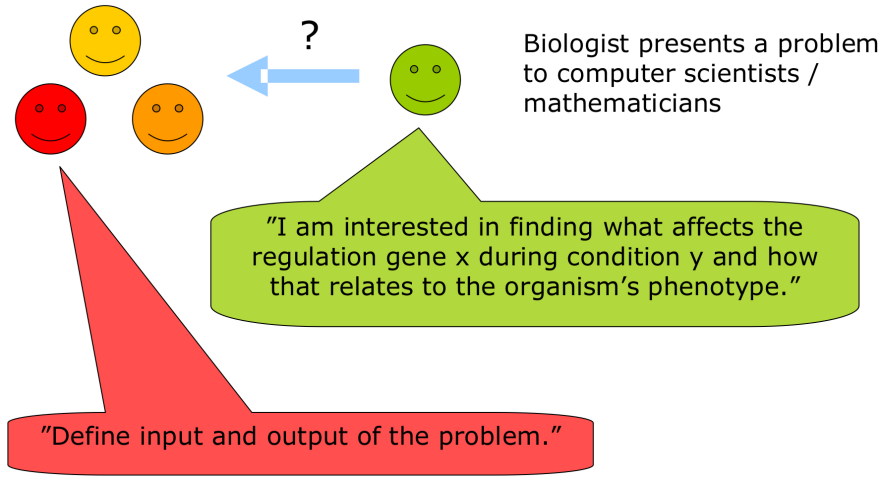
Where would you be in this triangle?

Prof. Juho Rousu, 2006

Bioinformatician's skill set

- Statistics, data analysis methods
  - Lots of data
  - High noise levels, missing values
  - #attributes $\gg$ #data points
- Programming languages
  - Scripting languages: Python, Perl, Ruby, ...
  - Extensive use of text file formats: need parsers
  - Integration of both data and tools
- Data structures, databases
  - New measurement techniques produce huge quantities of biological data.
- Scientific computation packages
  - R, Matlab/Octave, ...

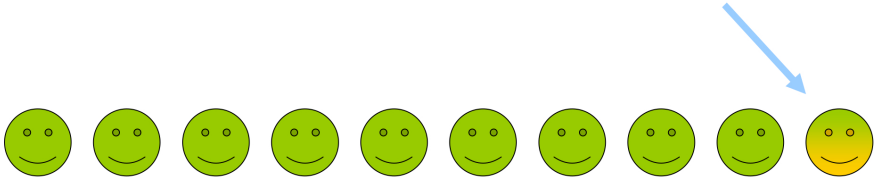
Bioinformatician's Competences



Prof. Esa Pitkänen, 2008

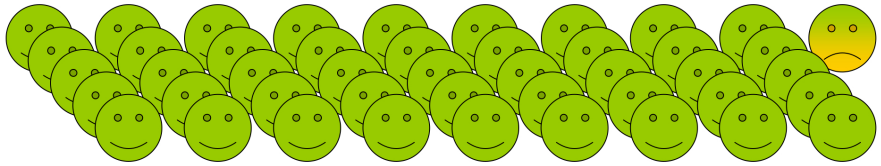
Bioinformatician's Competences

Bioinformatician is a part of a group that consists mostly of biologists.



Prof. Esa Pitkänen, 2008

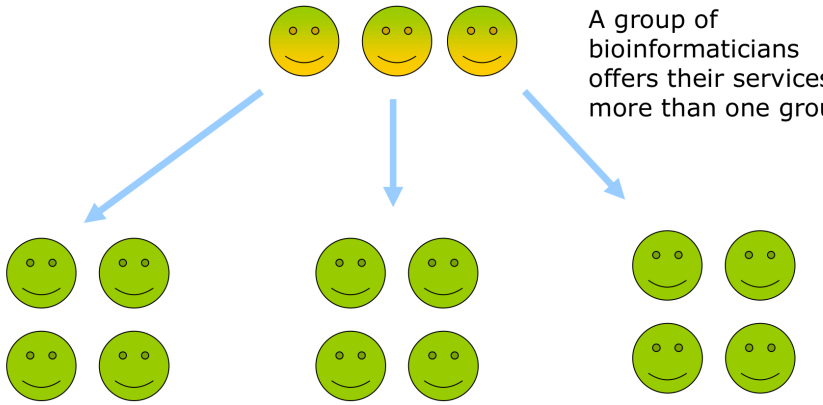
Bioinformatician's Competences



...biologist/bioinformatician ratio is important!

Prof. Esa Pitkänen, 2008

Bioinformatician's Competences



A group of bioinformaticians offers their services to more than one group

Prof. Esa Pitkänen, 2008

Axis 1: Python

1. Integrated Development Environment
2. Data Structures
3. Data Sets
4. Data Formats
5. Data Storage
6. Visualization

Axis 2: Algorithms

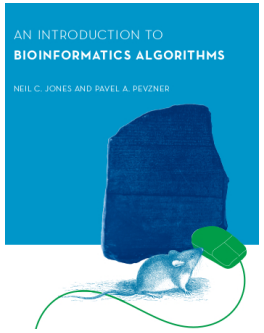
1. Complexity Analysis
2. Sorting
3. Exhaustive Search
4. Branch-and-Bound Algorithms
5. Greedy Algorithms
6. Divide-and-Conquer Algorithms
7. Data Mining Algorithms

Axis 3: Cloud Computing

1. Cloud Storage
2. Databases
3. Elastic Compute
4. Handling Large Data Sets

Literature

Jones, Pevzner: An Introduction to Bioinformatics Algorithms. MIT Press, 2004



JOHN M. ZELLE: Python Programming: An Introduction to Computer Science (Third Edition)

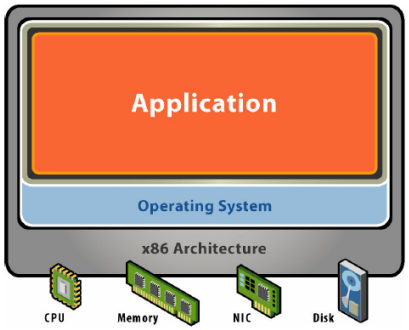
Jeff Chang, Brad Chapman, Iddo Friedberg, Thomas Hamelryck, Michiel de Hoon, Peter Cock, Tiago Antao, Eric Talevich, Bartek Wilczyński: Biopython Tutorial and Cookbook

Virtualization

- Virtualization deals with "extending or replacing an existing interface so as to mimic the behavior of another system"
- Virtual system examples:
  - virtual private network,
  - virtual memory,
  - virtual machine,
  - ...

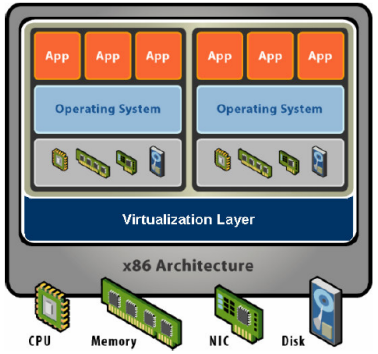
Starting Point: Physical System

- Physical Hardware
  - Processors, Memory, I/O devices, ...
  - Physical resources often underutilized
  - Periods that are over-utilized
- Software:
  - Tightly coupled to Hardware,
  - Single active OS,
  - OS controls Hardware



What is a Virtual Machine?

- Hardware-level Abstraction
  - Virtual Hardware: Processors, Memory, I/O devices, ...
  - Encapsulates all OS and application state.
- Virtualization Software:
  - Extra level of indirection decouples hardware and OS,
  - Multiplexes physical hardware across multiple "guest" VMs,
  - Strong isolation between VMs,
  - Manages physical resources, improves utilization.

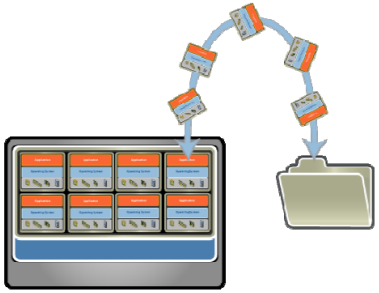


Virtual Machine Isolation

- Secure Multiplexing:
  - Run multiple VMs on single physical host,
  - Processor hardware isolates VMs.
- Strong Guarantees:
  - Software bugs, crashes, viruses within one VM cannot affect other VMs
- Performance Isolation:
  - Partition system resources,
  - Example: VirtualBox controls for reservation, limit, shares.

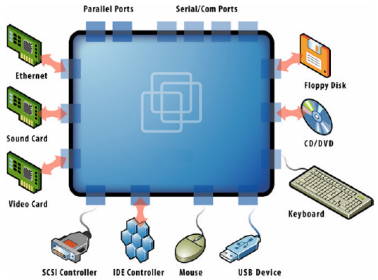
Virtual Machine Encapsulation

- Entire VM in a file:
  - OS, applications, data;
  - Memory and device state.
- Snapshots and Clones:
  - Capture VM state on the fly and restore to point-in-time,
  - Rapid system provisioning, backup, remote mirroring.
- Easy Content Distribution:
  - Pre-configured apps, demos.
  - Virtual Appliances.



Virtual Machine Compatibility

- Hardware Independent:
  - Physical hardware hidden by virtualization layer,
  - Standard virtual hardware exposed to VM.
- Create Once, Run Anywhere:
  - No configuration issues,
  - Migrate VMs between hosts.
- Legacy Virtual Machines:
  - Run legacy OS on new platform.



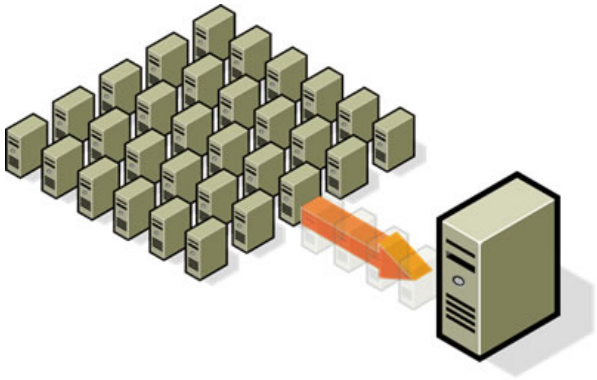
Common Uses

- Test and Development
  - Rapidly provision test and development servers.
  - Store libraries of pre-configured test machines.
- Business Contunuity
  - Reduce cost and complexity by encapsulating entire systems into single files
  - Replicated and restored on demand into any target system.
- Enterprise Desktop
  - Secure unmanaged PCs without compromising end-user autonomy by layering a security policy in software around desktop virtual machines.

Common Uses

- Run legacy software on non-legacy hardware
- Run multiple operating systems on the same hardware
- Create a manageable upgrade path
- Manage outages (expected and unexpected) dynamically

Virtualized Data Centers



Reduce costs by consolidating services onto the fewest number of physical machines

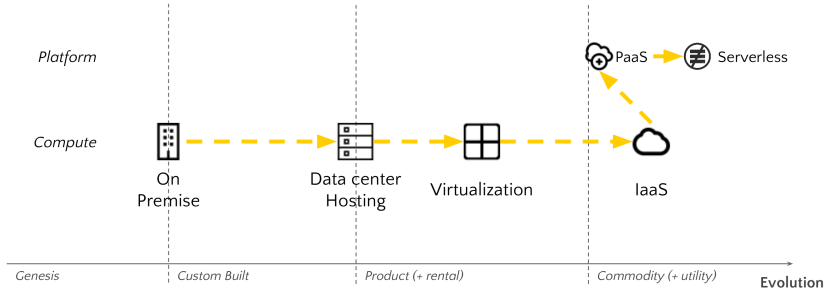
Non-virtualized Data Centers

- Too many servers for too little work
- High costs and infrastructure needs
  - Maintenance
  - Networking
  - Floor space
  - Cooling
  - Power
  - Disaster Recovery

Dynamic Data Centers

- Virtualization helps us break the "one service per server" model
- Consolidate many services into a fewer number of machines when workload is low, reducing costs
- Conversely, as demand for a particular service increases, we can shift more virtual machines to run that service
- We can build a data center with fewer total resources, since resources are used as needed instead of being dedicated to single services

Towards Serverless Computing



Function as a Service

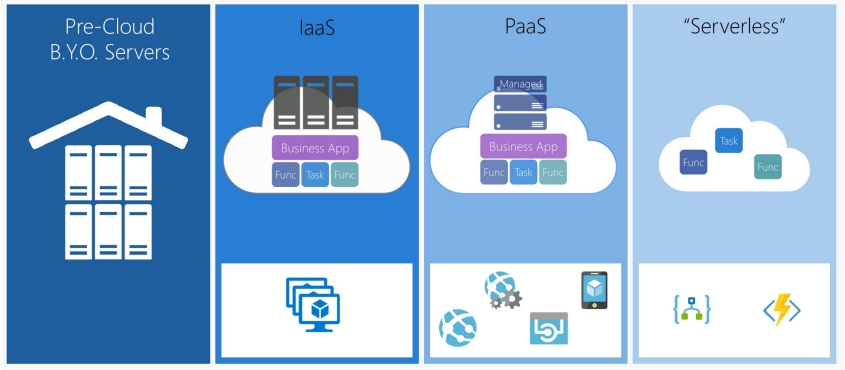
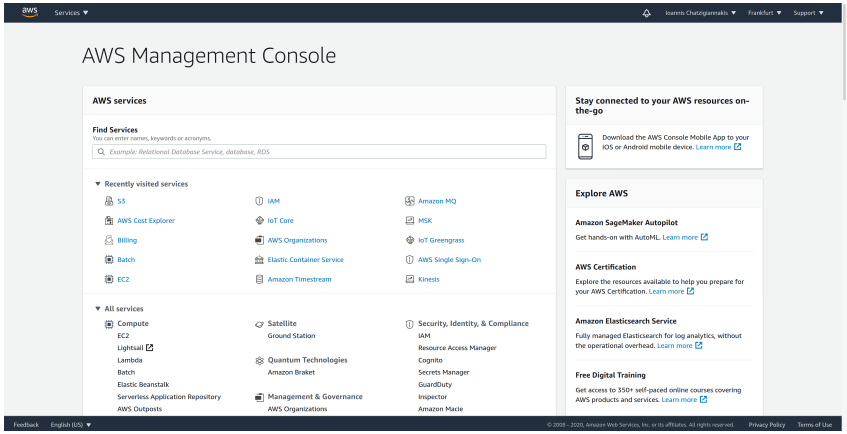
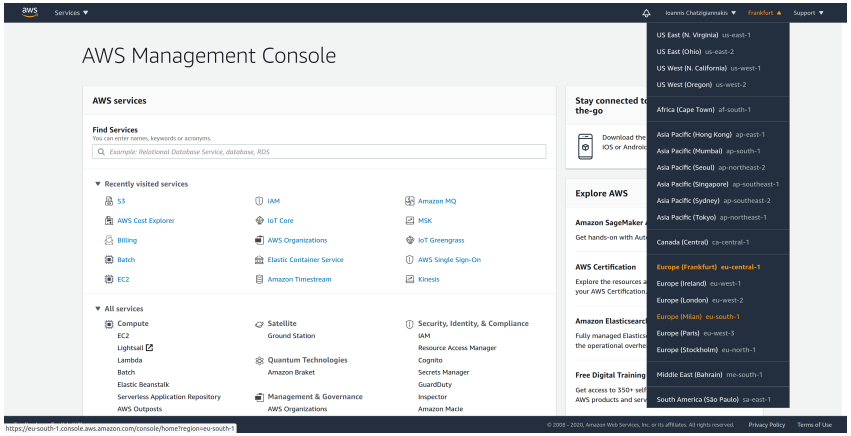
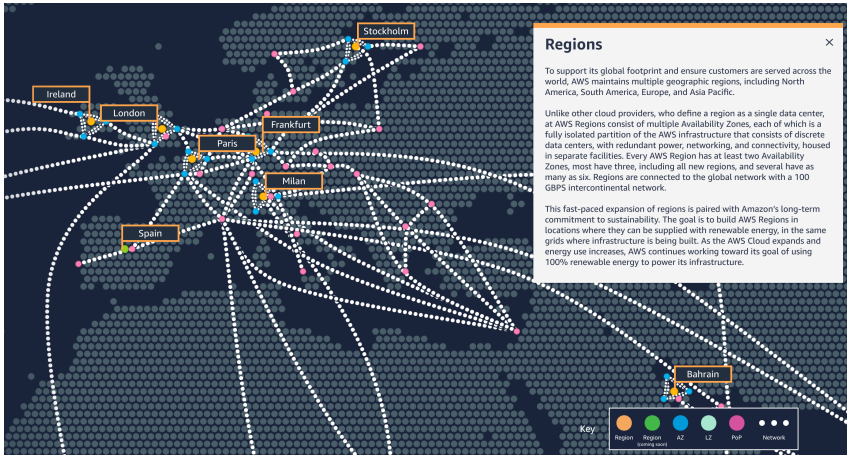


Figure 1. Magic Quadrant for Cloud Infrastructure and Platform Services





AWS Infrastructure



Regions

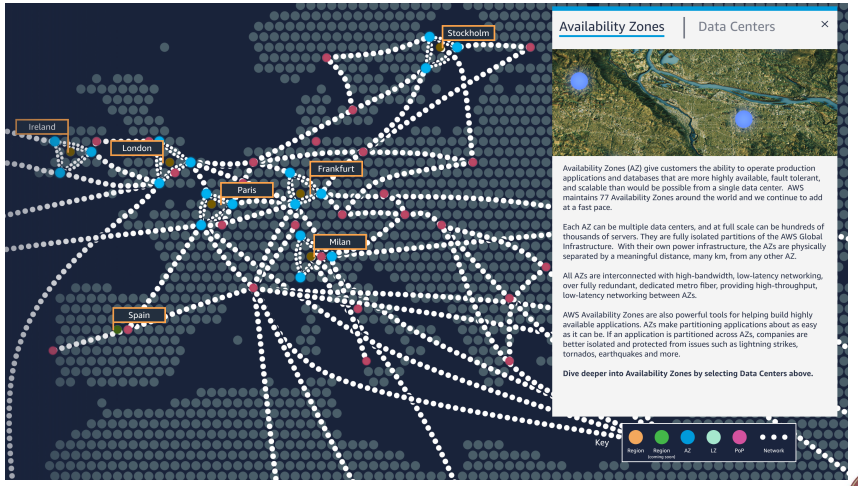
To support its global footprint and ensure customers are served across the world, AWS maintains multiple geographic regions, including North America, South America, Europe, and Asia Pacific.

Unlike other cloud providers, who define a region as a single data center, at AWS Regions consist of multiple Availability Zones, each of which is a fully isolated partition of the AWS infrastructure that consists of discrete data centers, with redundant power, networking, and connectivity, housed in separate facilities. Every AWS Region has at least two Availability Zones, most have three, including all new regions, and several have as many as six. Regions are connected to the global network with a 100 GBPS intercontinental network.

This fast-paced expansion of regions is paired with Amazon's long-term commitment to sustainability. The goal is to build AWS Regions in locations where they can be supplied with renewable energy, in the same grids where infrastructure is being built. As the AWS Cloud expands and energy use increases, AWS continues working toward its goal of using 100% renewable energy to power its infrastructure.

Computational Thinking | Course Topics | Virtualization | Amazon Web Services
Introduction

AWS Infrastructure



Availability Zones | Data Centers

Availability Zones (AZ) give customers the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center. AWS maintains 77 Availability Zones around the world and we continue to add at a fast pace.

Each AZ can be multiple data centers, and at full scale can be hundreds of thousands of servers. They are fully isolated partitions of the AWS Global Infrastructure. With their own power infrastructure, the AZs are physically separated by a meaningful distance, many km, from any other AZ.

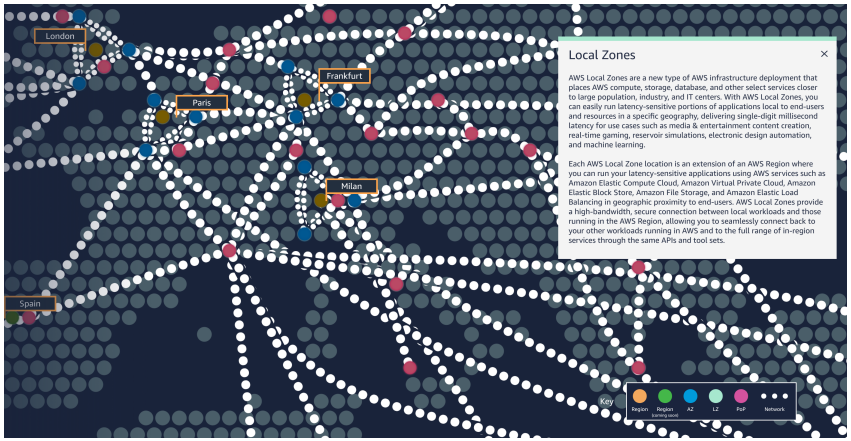
All AZs are interconnected with high-bandwidth, low-latency networking, over fully redundant, dedicated metro fiber, providing high-throughput, low-latency networking between AZs.

AWS Availability Zones are also powerful tools for helping build highly available applications. AZs make partitioning applications about as easy as it can be. If an application is partitioned across AZs, companies are better isolated and protected from issues such as lightning strikes, tornados, earthquakes and more.

Dive deeper into Availability Zones by selecting Data Centers above.

Computational Thinking | Course Topics | Virtualization | Amazon Web Services
Introduction

AWS Infrastructure



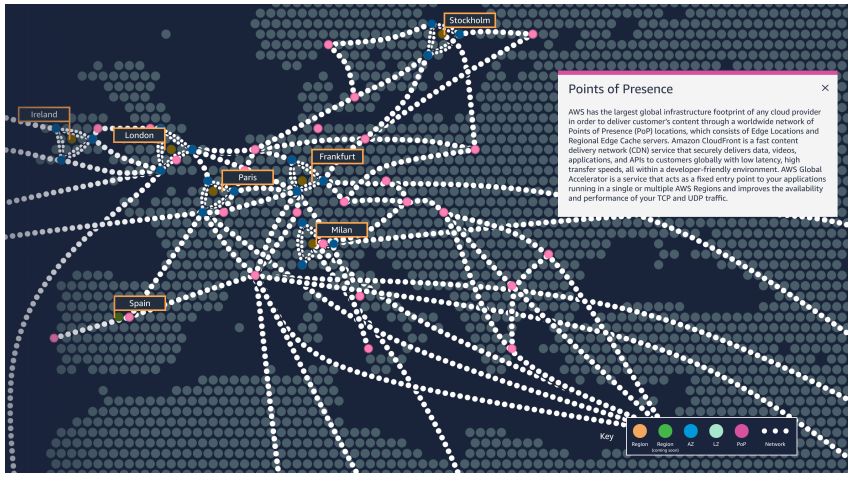
Local Zones

AWS Local Zones are a new type of AWS infrastructure deployment that places AWS compute, storage, database, and other select services closer to large population, industry, and IT centers. With AWS Local Zones, you can easily run latency-sensitive portions of applications local to end-users and resources in a specific geography, delivering single-digit millisecond latency for use cases such as media & entertainment content creation, real-time gaming, reservoir simulations, electronic design automation, and machine learning.

Each AWS Local Zone location is an extension of an AWS Region where you can run your latency-sensitive applications using AWS services such as Amazon Elastic Compute Cloud, Amazon Virtual Private Cloud, Amazon Elastic Block Store, Amazon File Storage, and Amazon Elastic Load Balancing in geographic proximity to end-users. AWS Local Zones provide a high-bandwidth, secure connection between local workloads and those running in the AWS Region, allowing you to seamlessly connect back to your other workloads running in AWS and to the full range of in-region services through the same APIs and tool sets.

Computational Thinking | Course Topics | Virtualization | Amazon Web Services
Introduction

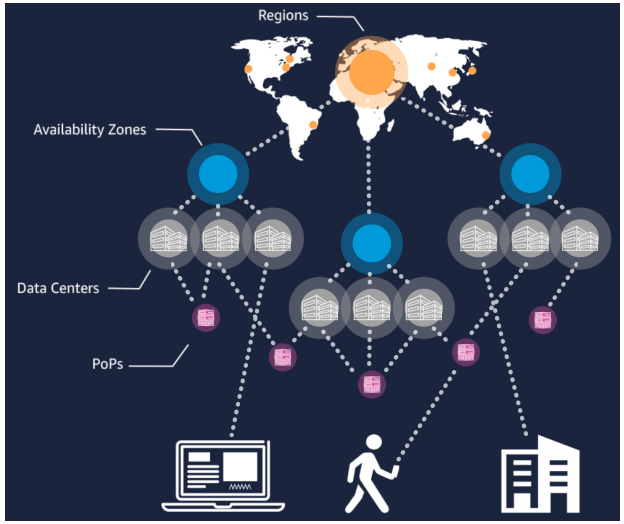
AWS Infrastructure



Points of Presence

AWS has the largest global infrastructure footprint of any cloud provider in order to deliver customer's content through a worldwide network of Points of Presence (PoP) locations, which consists of Edge Locations and Regional Edge Cache servers. Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment. AWS Global Accelerator is a service that acts as a fixed entry point to your applications running in a single or multiple AWS Regions and improves the availability and performance of your TCP and UDP traffic.

AWS Infrastructure



Introduction to AWS S3

- S3 = Simple Storage Service
  - From 0 bytes to 5 Tbytes.
- Provides a secure, durable, highly-scalable storage space.
  - AWS secures content with encryption, ACL and bucket policies.
  - AWS guarantees 99.999999999% durability (11 $\times$ 9s).
  - AWS guarantees 99.99% availability.
- We can access items stored:
  - Using the web.
  - Using the Web Console.
  - Using the Smartphone App.
  - From the Command line AWS tool.
  - Programmatically through the AWS S3 API.

S3 Basics

- Object-based storage.
  - Files = Objects.
  - Not suitable to install an operating system or host a database.
- Files/Objects are organized in Buckets.
- Bucket names must be unique – S3 is a universal namespace.
  - `http://sapienza2020adm.s3.amazonaws.com/`
  - When you create a new S3 bucket, AWS creates a new web address.
- Objects (Files) have the following properties:
  - Key: the name of the object.
  - Value: the actual contents.
  - Version ID: used by the versioning system.
  - Metadata: tags that we can attach to objects.
  - ACL: who can access the object.

S3 Storage Classes

- Free Tier – new AWS accounts
  - 5GB of S3 storage.
  - 20,000 GET – 2,000 PUT/COPY/POST/LIST
  - 15GB of Data Transfer Out each month for one year
- S3 Standard
  - \$0.0245 per GB
  - \$0.0054 per 1000 PUT/COPY/POST/LIST
  - \$0.00043 per 1000 GET/SELECT/all other requests.
- S3-IA Infrequent Access
  - \$0.0135 per GB – a minimum storage duration of 30 days.
  - \$0.01 per 1000 PUT/COPY/POST/LIST
  - \$0.001 per 1000 GET/SELECT/all other requests.
- S3 Glacier
  - \$0.0045 per GB – a minimum storage duration of 90 days.
  - \$0.06 per 1000 PUT/COPY/POST/LIST
  - \$0.00043 per 1000 GET/SELECT/all other requests.

1st Assignment – Part 1

https://www.hackerrank.com/

- Complete a total of 25 Python challenges from the following subdomains:
  - Python: Basic Data Types (any 5), Strings (any 4), Sets (any 4), Math (any 4), IterTools (any 4), Collections (any 4)
- You can cooperate, You can search on the Internet, ...
- You need to write **your own code**
- Email `ichatz@diag.uniroma1.it`
  Subject: [PCS2] Homework 1
  Your GitHub repository with your solutions, for all challenges.
  Also send your hackerrank user account link:
  https://www.hackerrank.com/{username}

1st Assignment – Part 2

https://awsacademy.instructure.com/

- You must complete the knowledge check of the modules listed below:
  - Module 1 (Global Infrastructure)
  - Module 2 (Structures of the Cloud)
  - Module 3 (AWS Console)
  - Module 4 (Virtual Servers)
  - Module 7 (Security)
  - Module 13 (Emerging Technologies in the Cloud)
  - Module 14 (Billing and Support)

Deadline: 24 October 2022