MY472 - Week 11
Cloud Computing

Friedrich Geiecke

Outline

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS
3. Customising an EC2 instance and running it via the command line and a file transfer client

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS
3. Customising an EC2 instance and running it via the command line and a file transfer client

A definition

“Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models.” (From *NIST Definition of Cloud Computing*)

You are already using cloud application services

These are the examples of Software as a Service (more on this later)

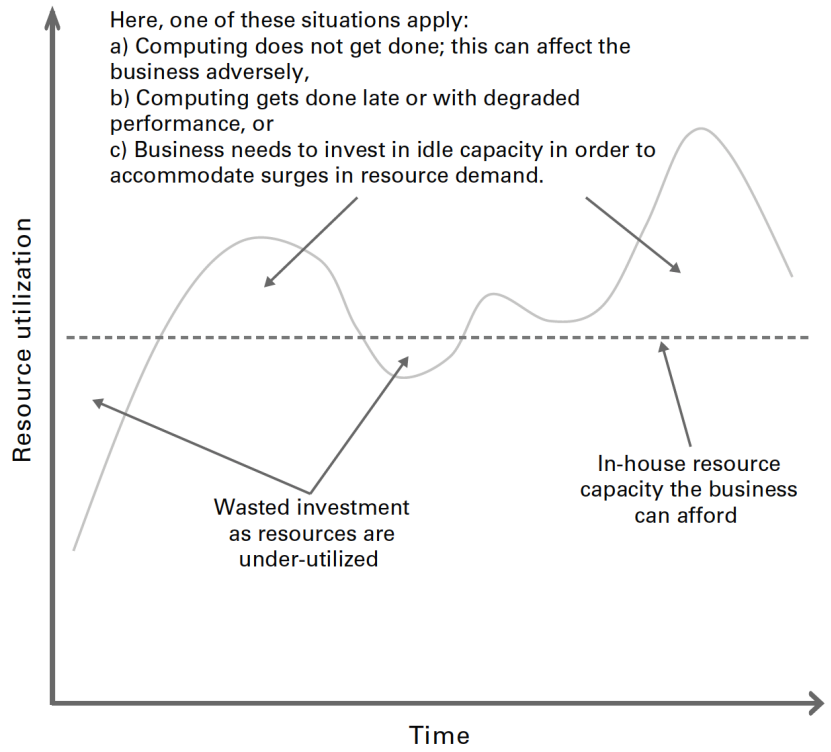
- Email
  - Gmail
  - Exchange mail
- Storage
  - Google drive
  - Dropbox
- Software
  - Google docs
  - Adobe

Possible use cases in data science

- Continuous scraping or API requests
- Large scale natural processing
- Hosting and querying (very large) databases
- Training machine learning models, e.g. in deep learning or reinforcement learning
- Hosting web applications
- ...

Let’s buy a powerful computer?

Stylised issues with fixed environments and fluctuating demand



How do cloud computers look like? Virtualisation

In cloud computing, computers are virtualised

Similar idea to virtual machine on your computer but readily scalable in a short time

In data centers connected to the internet, hardware hosts a number of virtual machines

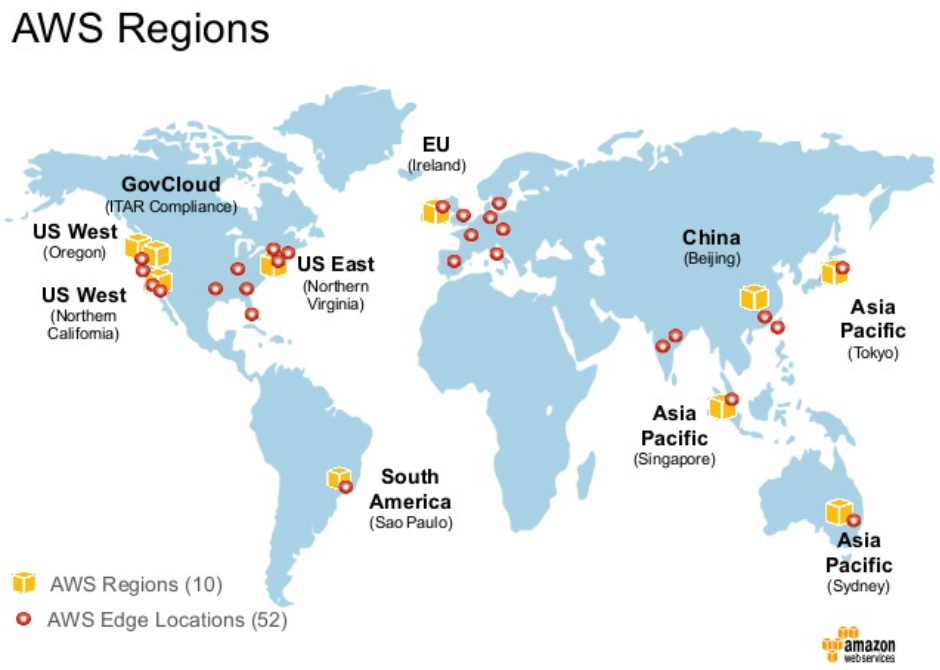
Data centers

For major providers of cloud service, there are data centers everywhere and you can deploy resources in any location

Reasons for specific locations could be: Latency, data protection laws, costs

Exact locations of data centers are usually undisclosed for security reasons

Example AWS



Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS
3. Customising an EC2 instance and running it via the command line and a file transfer client

“As a service” models of cloud computing

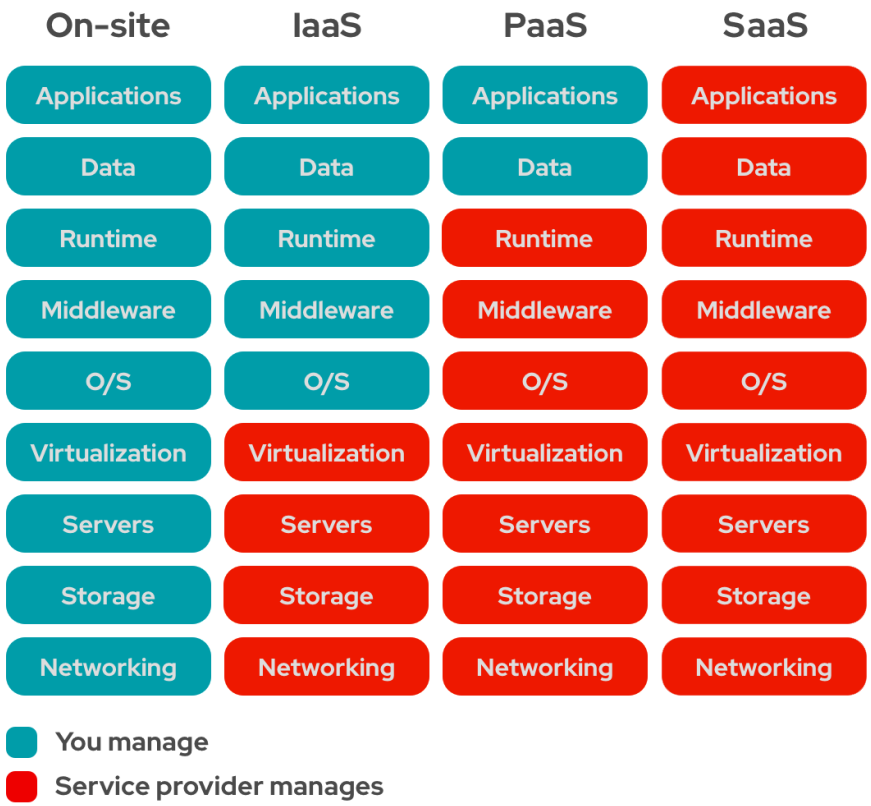
To differentiate cloud computing services, it can be useful to think of them in the following broad categories

Infrastructure as a service (IaaS): A cloud provider hosts the infrastructure components traditionally present in an on-premises data center, including servers, storage and networking hardware. Includes a variety of associated services such as backup, log maintenance, updates, security, etc. For example, AWS EC2, Microsoft Azure, Google Compute Engine

Platform as a service (PaaS): Additional delivery of hardware and software tools – often those needed for application development. For example, Google App Engine, Heroku, AWS Elastic Beanstalk

Software as a service (SaaS): Software distribution model in which applications are remotely hosted and served to users via the Internet, usually through a standard web browser. For example, Microsoft 365

Stylised figure



Source

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS
3. Customising an EC2 instance and running it via the command line and a file transfer client

Serverless computing

There is a server, but cloud provider runs and manages it

Developer can focus on code

One area: Function as a Service (FaaS)

Example: User enters input into website, back end runs some computation and returns it (e.g. cheapest flights on a day)

Approach one (traditional): Rent/buy fixed sever equipment (on premise or in cloud) and run it 24/7 which might leave it mostly idle

Approach two (serverless): Once user enters input, send function call to cloud provider who computes output and returns it

For example AWS Lambda, Azure “Functions”, Google Cloud “Functions”

Serverless cloud computing vs traditional solutions

- Advantages
  - Often cheaper - only pay when code is running
  - No/little maintenance
  - Scalable (instantly)
- Disadvantages
  - Not really suited for tasks that takes long time to process (e.g. 15min limit for AWS Lambda)
  - Not for memory intensive tasks
  - Can have higher delays for first requests
  - Costs less predictable than with fixed capacity

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS
3. Customising an EC2 instance and running it via the command line and a file transfer client

Advantages

- Availability and durability
  - Services can be down, but arguably less frequently than many on-premise solution
  - Similarly, loss of data can happen but arguably less frequently than in other solutions
- Scalability
  - Inherently scalable with regard to computational power, data, etc. Deploy the same app on bigger instances or scale inherently with serverless approaches
- Costs
  - Pay computational resources depending on demand
  - Three costs of ownership: Capital expenses (hardware, software), operating expenses (service, maintenance), indirect costs (downtime, time to market)
- Security
  - For example, up to date computer systems, automated backups, physical security of infrastructure

Disadvantages

- Costs
  - Can be expensive relative to own hardware and software, e.g. for frequent computations at a stable scale
  - Lock-in: Potentially large costs in terms of developer wages to change to a different solution if the current one becomes expensive
- Security
  - Connection to the cloud and all computation/data through the internet requires careful setups and can be vulnerable
  - User responsible for connection between client and cloud, but also data encryption, application security, etc.
- Legal compliance
  - For example, GDPR requirement of encryption, data security, data physical location

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud

RStudio server via posit cloud

Register and log in via https://posit.cloud/plans/free

Very easy to set up: Create RStudio server in the cloud by clicking on “New Project”

Free version allows to use 1 CPU and 1 GB RAM per project

Restricted to 25 project hours per month

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS

Using other cloud platforms to host RStudio servers

We can also set up an RStudio server ourselves, e.g. via AWS https://aws.amazon.com/

Registration requires a credit card, but free tier options available during first 12 months

Because AWS is arguably the most popular cloud computing provider as of now, we use it as the example in this lecture. Yet, there are many other service e.g. Google Cloud, Microsoft Azure, Alibaba Cloud, IBM Cloud, Salesforce, …

On AWS, EC2 (Elastic Compute Cloud) allows to create and launch virtual machines

Many other services such as S3 (storage) or Lambda (serverless)

Setting up the RStudio server via AWS

Rather than installing software from scratch, AMI (Amazon Machine Images) can make it more convenient to set up virtual machines

We will use Louis Aslett's RStudio Server AMIs
https://www.louisaslett.com/RStudio_AMI/

If logged into AWS, clicking on one of the AMI links on his website leads directly to creating the associated EC2 instance

Pick a free tier option

In the "security group" option, choose HTTP to make the server accessible via the browser later

After creating the instance, the public DNS is the URL, and the RStudio log-in details are "rstudio" (account name) and the instance id (password)

Change this with `passwd` in terminal within RStudio after logging in

Introduction

Service models

Trends: Going serverless

Advantages and disadvantages of cloud computing

Guided coding session

1. Running an RStudio server via posit cloud
2. Running an RStudio server via AWS
3. Customising an EC2 instance and running it via the command line and a file transfer client

Customising and using instances via the command line

In this example, we will use an AMI with less pre-installed software and customise our own instance by installing R etc.

Common approach (for scientific computing): Connect to the remote machine via SSH (secure shell), run scripts via the command line and use file transfer client to add/download files such as a scripts, data, outcomes, etc. to/from the instance

Particularly helpful for large computations in the cloud such as machine learning or textual analysis, or code in production

Does not require user input in a GUI such as selecting code, submitting to console, etc

Could still use an AMI with pre-installed R such as the ones by Louis Aslett also for command line access to the EC2, but will create our own to study these topics

Step 1: Launching the instance

In the EC2 dashboard click “Launch instance”

Choose Amazon Linux 2 AMI (many other options available which would require slightly different steps and commands)

Choose free tier option

When launching instance, create key pair for SSH access (or choose existing one)

Good to keep in mind: User name for Amazon Linux 2 AMI is “ec2-user” (see documentation for other AMIs)

Step 2: Connecting to the instance, setting swap memory

Connect to EC2 instance via command line (more info)

Mac/linux: `chmod 400 yourkeyname.pem` and `ssh -i "path/to/key/yourkeyname.pem" ec2-user@public-instance-dns`

Windows: Use PuTTY - tutorial as pdf on course page

Linux console via the browser (simpler): Click on instance -> Connect -> EC2 Instance Connect -> Connect

First, for small free tier instances with only 1GB of ram, create swap memory (reference) to install some larger R packages such as the tidyverse without maxing out memory (not necessary for larger instances with more ram)

```
sudo /bin/dd if=/dev/zero of=/var/swap.1 bs=1M count=2048
sudo /sbin/mkswap /var/swap.1
sudo /sbin/swapon /var/swap.1
sudo sh -c 'echo "/var/swap.1 swap swap defaults 0 0 " >> /etc/fstab'
```

Step 3: Installing Linux libraries and R

Before installing some of the R packages later, we need to install additional Linux libraries `sudo yum install libcurl-devel openssl-devel libxml2-devel`

Install R with `sudo amazon-linux-extras install R4`

Afterwards you can open R by typing `R` or run scripts with `Rscript myscript.R` (more later)

Step 4: Copying files to and from the EC2 instance

We can copy files from and top the EC2 via the command line (`scp` command)

Can be more convenient to use a file transfer programme, for an overview of supported programmes see this link

One option is Cyberduck, can be downloaded from cyberduck.io or Mac App Store / Windows Store

Choose “Open connection”, SFTP (secure file transfer protocol), enter the public DNS for server, the user name “ec2-user” and supply your .pem file for the SSH private key

Side note: We can also access the RStudio Server from Example 2 via the file transfer client: For this, choose the user name “rstudio” and the instance number as password when connecting

Step 5: Running scripts and installing R packages

`ls` shows all folders and files in the current directory

`cd path/to/some/folder` goes to folder

`Rscript myscript.R` runs R script in current folder

`Rscript myscript.R &` runs R script in the background while the shell is open

Install the R packages that we need with the script `install_packages.R` (this takes a while)

Run a single iteration of the scraping script `scraping_example_to_schedule.R` to see the output it produces

Step 6: Scheduling scripts with cron (1/2)

With this setup, we can run scripts once e.g. for large computations in textual analysis and machine learning, and then download the outcomes with our file transfer client

Yet, for repeated tasks such as scraping or API calls, it can be helpful to run scripts on a schedule

This can e.g. be done via `crontab`

Set schedule with `crontab -e` (in vim editor)

vim editor basics: `i` for inserting/writing mode, `esc` to exit writing mode which allows commands such as `:w` for saving, `:q!` for quitting without saving, and `:wq` for saving and exiting

Step 6: Scheduling scripts with cron (2/2)

Cron syntax: * * * * * (minute, hour, day, month, day of week), e.g. 15 9 * * * runs every day at 9.15am

Important: Check time zone on machine by typing in `date` (often UTC)

`30 8 * * * sudo reboot`: Reboots at 8.30am

`0 9 * * * Rscript /home/ec2-user/code/scraping_example_to_schedule.R >> /home/ec2-user/code/cron_log.txt 2>&1`: Executes the scraping script every day at 9am and stores the output into a log file called cron_log.txt

Alternative solutions

Serverless approaches such as e.g. AWS Lambda (Google Functions, Azure Functions, etc.) can also be helpful for the tasks discussed here. For example:

- Put scraping code into a small Lambda Function (unfortunately need to call R via Python in these functions as of now) which is e.g. executed on a schedule without creating an EC2 instance
- Usually creating EC2 instances is required for machine learning, scientific computing etc.
- For more cost efficiency in recurring tasks that require an EC2 instance, e.g. start and stop an EC2 instance via lambda function

More information

The end

What we have covered today

- General introduction to cloud computing
- Setup of an RStudio server via RStudio for doing some simple computations
- Setup of RStudio sever via AWS, free tier allows longer use in first year, example of continuous scraping within R
- Launch of own instance, installing software, and accessing it via command line and file transfer client. Very flexible setup for computation which also allows continuous scraping
- Connected topics from R programming, IDE vs command line, cloud computing, databases, web scraping
- Use free tier instances for the topics discussed, **stop/terminate them after use**, and make sure to keep an eye on costs via e.g. `Services` -> `Cost Explorer` and `Services` -> `Billing`