Big Data

RABY Nicolas, VILLAVICENCIO Carlos

January 2019

1 Introduction

Avec les avancées et l'utilisation de plus en plus massive dans le domaine de l'informatique, la gestion de données massives (Big Data) est devenue un domaine primordiale, notamment pour les géants du web tels que Yahoo, Google etc. Au cours de notre cursus, nous avons appris à maitriser les outils utilisés pour ce type de gestion de données, que ce soit par le biais de différents frameworks tel que HDFS ou Spark, ou encore via des systèmes de gestion de base de données tel que HBase. Afin de finaliser notre apprentissage, nous avons du mettre en place un système de visualisation de données via une interface web pour afficher une carte représentant les altitudes en fonction de données GPS. Ces données représentant plusieurs Gigas, il nous faudra utiliser la grande puissance de calcul de notre infrastructure. Nous présenterons ici dans un premier temps les différents frameworks ou outils que nous avons utilisés pour la mise en place de ce projet. Nous détaillerons ensuite le fonctionnement de notre infrastructure ainsi que de notre interface web. Enfin nous parlerons des difficultés rencontrées et des possibles améliorations de notre projet.

2 Outils utilisés

Nous listons ici les différents outils, que ce soit des frameworks, des bibliothèques etc. en appuyant notre choix sur les avantages que ces derniers proposent.

2.1 Gestion des données (HDFS)

Pour ce projet nous avons décidé d'utiliser Spark pour le framework de calcul distribué. Notre choix s'est porté vers Spark plutôt que Map Reduce grâce à sa plus grande rapidité et sa facilité d'utilisation. En effet Spark gère la plupart de ses opérations en mémoire (RAM) tandis que MapReduce lit les données depuis le disque dur, ce qui le rend plus lent. En ce qui concerne le stockage des données, nous utilisons HBase. En effet HBase profitant du système de fichier distribué de HDFS, cela nous a parru un choix évident.

2.2 Interface Web

En ce qui concerne l'affichage et la représentation de l'analyse des données, nous utilisons un serveur d'application sous NodeJS en utilisant le framework express.js, utilisé pour les applications web. Express permet de gerer plus facilement les routes (Url) de notre application. Les routes ne sont pas dans un premier temps forcément nécessaires pour l'affichage de la carte, mais nous pourrons utiliser un système de routes plus tard afin de récupérer des coordonnées dans l'url pour afficher la localisation correspondante. En ce qui concerne l'affichage et la gestion de la carte intéractive nous utilisons la bibliothèque javascript leaflet, bibliothèque utilisée pour la cartographie en ligne et notament par OpenStreetMap. Elle permet l'utilisation de "tuiles", que nous utiliserons aussi afin de représenter les différentes régions de la carte.

Afin de visualiser le résultat de notre programme nous utilisons express.js. Une fois nos données inscrites dans HBase nous allons pouvoir établir un lien entre notre application et notre base de données. Ici nous utilisons le port 8000 pour le déploiement et pour vérifier le bon fonctionnement de notre application nous récupérons les cartes associées à des coordonnées via l'url. Ainsi il faudra indiquer ces valeurs.
Exemple :

http://localhost:8000/:z/:x/:y ou z est le zoom, x la lattitude et y la longitude

3 Fonctionnement

3.1 Structure

Notre programme se divise en trois fichiers principaux :

- PngGenerator.java
- MapProjection.java
- ProjetMaps.java

PngGenerator.java : Classe implémentant Serializable. Ce fichier contient les différentes fonctions permettant de génerer nos images et de les écrire. Deux fonctions permettent de générer les images.

- void generateEmptyImageWithColor(Color color) permet la génération d'image sans les gradients mais avec une couleur unie.
- void generateWithImageGradient(String imagePath) génère les images en utilisant le gradient pour représenter l'altitude.

Une autre fonction byte[] getBytes() elle, sert à traduire une image au format PNG en Array de bytes.

MapProjection.java :
Ce fichier contient les fonctions permettant de faire les conversions nécessaires au sein de notre projet. De plus une classe Point y est intégrée. On retrouve dans ce fichier :

- Point fromLatLngToPoint(double lat, double lng, int zoom) permet en fonction d'une latitude et d'une longitude donnée de retourner un Point.
- Point degToTilePosition(double lat, double lon, int zoom) qui à partir d'une latitude et d'une longitude retourne un Point représentant la position d'une tuile.

ProjetMaps.java : Contient, dans un premier temps, les fonctions relatives à HBase.

- void createOrOverwrite(Admin admin, HTableDescriptor table) permet de réinitialiser les tables de HBase.
- void createTable(Connection connect) sert de son coté à créer une table dans HBase pour nos données.
- ArrayList$< String >$ getAllCoordinates() créer et renvoie un tableau contenant des coordonnées au format de nos fichier .hgt

Enfin ce fichier contient aussi notre fonction main et fait de ce fichier notre fichier principal contenant notre programme et nos configurations Spark.

Pour la partie visualisation, nous sommes resté dans la simplicité avec un déploiement express.js. Nous avons dans BigDataMaps/node le fichier index.js. Ce fichier permet le déploiement et le routing ainsi que la récupération de nos tuiles sur HBase en se servant des paramètres passés dans l'url.

3.2 Exemples

Nous allons présenter ici des exemples de cartes s'affichant grâce à notre infrastructure.

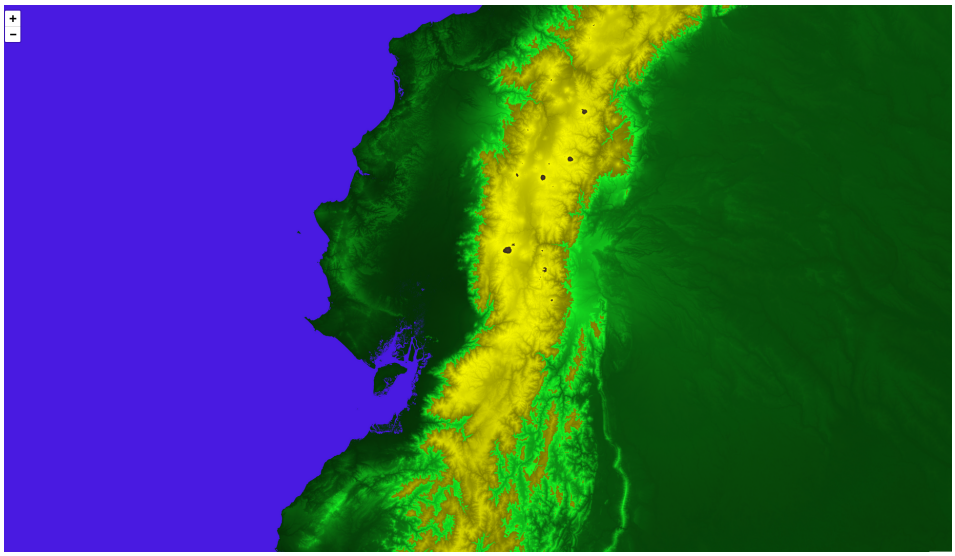


Figure 1: Example de visualisation (1)

L'exemple de la Figure 1 nous montre donc une tuile récupérée depuis HBase. Celle-ci grâce à notre serveur express utilisant leaflet à pu être récupérée et ainsi affichée dans notre navigateur. L'url utilisée est la suivante : http://jordan:8000/8/101/90.png. Notre URL se trouve sous la forme http://localhost:8000/:z/:x/:y tel que :

- z = 8 représente le zoom appliqué
- x = 101 représente donc les coordonnées en lattitude
- y = 90 représente les coordonnées en longitude

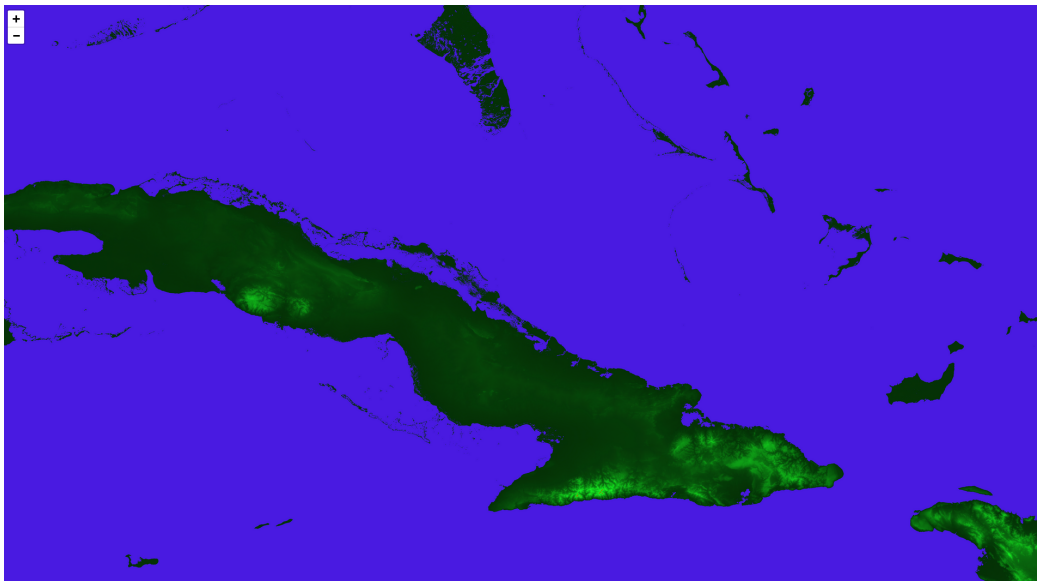


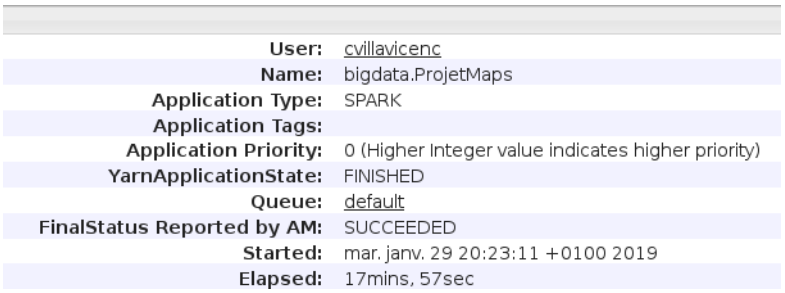
Figure 2: Example de visualisation (2)

Enfin un deuxième ci-dessus (Figure 2) récupéré grâce à cette URL : http://jordan:8000/8/101/67.png.

- z = 8 représente le zoom appliqué
- x = 101 représente donc les coordonnées en lattitude
- y = 67 représente les coordonnées en longitude

3.3 Performances

Concernant les performances de notre architecture, le résultat est visible sur la Figure 3. Ces performances correspondent à la totalité des fichiers hgt analysés en utilisant 50% du cluster. On peut voir que notre job se réalise en approximativement 18 minutes. On remarquera que l'état est bien à "FINISHED" et que le statut final est à "SUCCEDED".



User:	cvillavicenc
Name:	bigdata.ProjetMaps
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	mar. janv. 29 20:23:11 +0100 2019
Elapsed:	17mins, 57sec

Figure 3: Performances Spark

4 Difficultés rencontrées

- La mise en place de l'affichage de toutes les tuiles n'a pas été implémenté. Nous nous sommes concentré d'avantage sur le principal, à savoir l'utilisation de Spark et de HBase. Nous avons tout de même mis un place un système pour récupérer une image dans HBase afin de vérifier le bon fonctionnement de notre infrastructure.
- L'utilisation du cluster intensive est un frein à l'avancement du projet. Il à donc fallut tester les performances sur des échantillons et non sur la totalité des données, donnant des performances suceptibles d'être éloigné de la réalité dans un premier temps. Nous avons quand même reussi à établir la totalité des jobs avec une utilisation de 50% du cluster.
- L'implémentation du zoom s'est révélée être fastidieuse de pars toutes les opérations à appliquer aux images.
- La projection de Mercator n'a pas aboutie car nous n'avons pas réussi à l'utiliser avec Spark.
- L'insertion dans HBase prends plus de temps que prévu, nous pensons que cela pourrait se justifier à cause de la génération des images.

5 Conclusion

La mise en place de ce projet afin d'achever notre cursus nous a beaucoup apporté tant au point de vue technique qu'a notre compréhension. La gestion de donnée massive est un problème grandissant et détient d'énormes enjeux dans le futur. L'utilisation d'outils multiples, reconnus et utilisés dans plein d'entreprises actuelles sera un atout majeur pour notre entrée dans le monde de l'entreprise.