Dan & Briggs

Outil de recherche d’experts

Présentation

- La recherche d’experts à recruter peut s’avérée très difficile dans certains domaines.
- La capacité à analyser automatiquement des volumes de données importants traitant de domaines divers peut être très utile pour la recherche d’experts.
- Afin de trouver des experts dans des domaines incongrus, nous allons sélectionner les trois utilisateurs qui contribuent le plus sur un sujet et ses voisins dans le Wikipédia en français.
- Création d’un Data Lake contenant un master Dataset composé des révisions des pages et des liens entre pages du Wikipédia en français.
- Nous analyseront l’extraction du 01/05/2018.

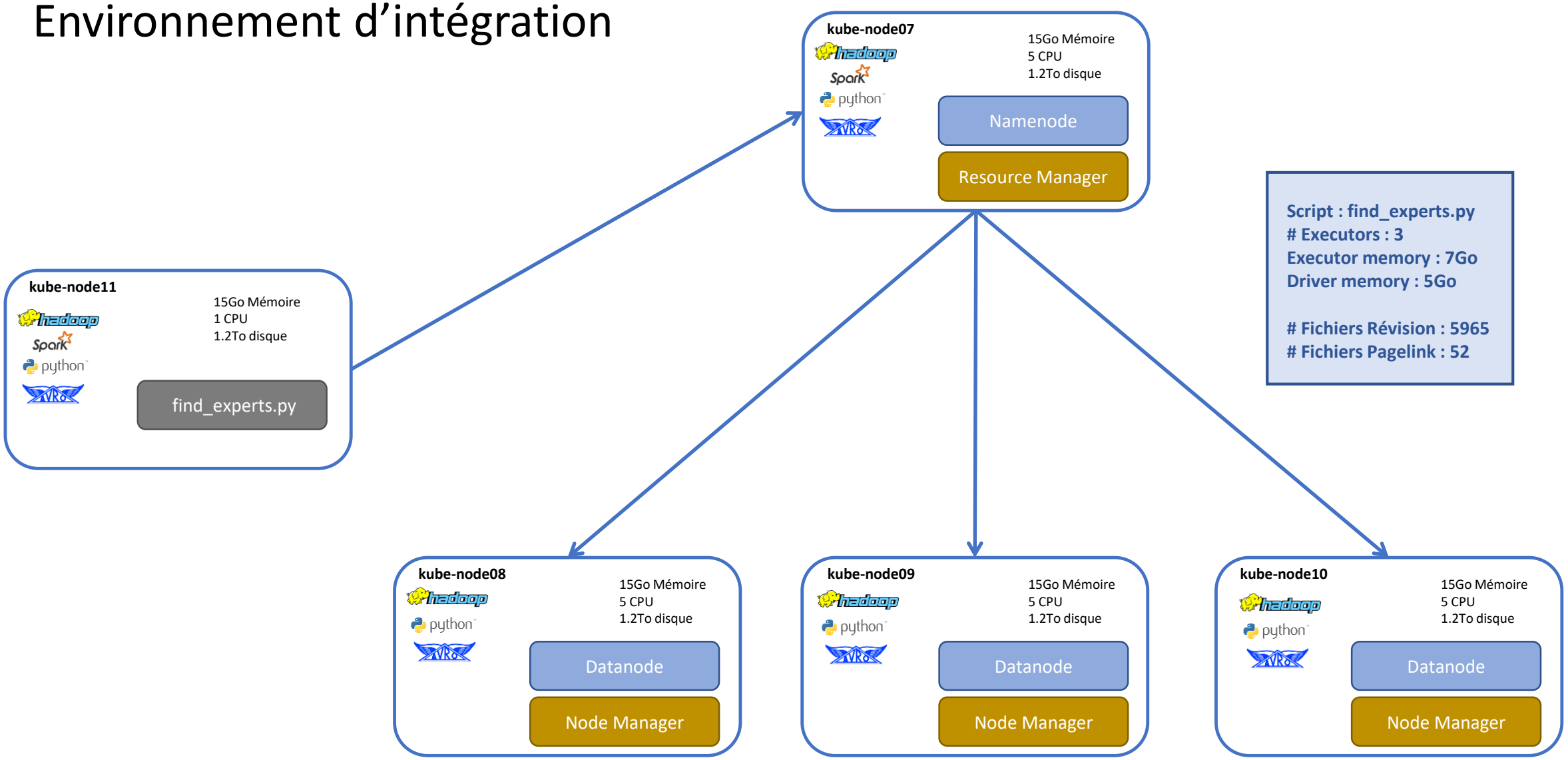
Choix des composants logiciels

- Apache Hadoop 2.8.4
  - Cluster HDFS hébergeant le Data Lake pour le stockage distribué des Dataset.
  - Apache Yarn pour gestion distribuée des ressources et l'exécution distribuée des applications.
- Apache Spark 2.3.1
  - Framework de calcul distribué utilisé pour l'analyse des Dataset stockés dans le Data Lake.
- Python 3.5.2
  - Langage de programmation pour le développement de l'application
- fastavro 0.21.5
  - Package Python pour la sérialisation Avro

Environnement d'Intégration

- Un environnement composé de 5 VMs :
  - Client :
    - kube-node11 : VM cliente utilisée pour soumettre les jobs Spark
  - Server :
    - kube-node07 : VM Namenode et Resource Manager
    - kube-node08 : VM Datanode et Node Manager
    - kube-node09 : VM Datanode et Node Manager
    - kube-node10 : VM Datanode et Node manager
  - Configuration et sources : https://github.com/plawson/oc-project4
- En production :
  - mettre une configuration HA avec un Namenode supplémentaire en standby en utilisant trois Quorum Journal Manager ou NFS. Ajouter également un standby pour le Resource Manager avec un cluster Zookeeper pour la coordination.

Environnement d'intégration



Import de données : pagelinks

- L’extraction des pagelinks est effectuée via dump MySQL. Environ 9Go.
- Il y a plusieurs milliers de pagelinks par ligne d’INSERT dans le fichier dump.
- Le script process_pagelinks.py lit directement le fichier dump.
- Un nouveau fichier Avro sur est généré sur le système de fichiers HDFS toutes les 190 lignes d’INSERT (paramétrable).
- A chaque pagelink correspond un enregistrement dans le fichier Avro.
- L’import sur HDFS a créé 52 fichiers Avro.

Import des données : révisions

- Les révisions sont extraites dans un fichier au format XML d'environ 60Go.
- Le script process_history.py traite le fichier de révision en utilisant du streaming XML.
- Pour ne pas saturer la mémoire, chaque fois qu'un fichier Avro est généré sur le système de fichiers HDFS, on libère toutes les ressources occupées par les éléments corresponds en mémoire.
- Chaque élément page contient l'ensemble de révisions concernant cette page.
- Un ficher Avro est généré toutes les 1580 pages (paramétrable).

Import de données : Arborescence

- Description : https://github.com/plawson/oc-project4/blob/master/documents/Data-Lake-Arborescence_v1.0.pdf

Sérialisation

- La recherche d'experts ne nécessite qu'un sous ensemble des propriétés présentes dans l'extraction du Wikipédia en français.
- Le Data Lake contient le master Dataset. Ainsi d'autres applications peuvent être amenées à utiliser ces données.
- Le choix est de sérialiser l'ensemble des propriétés extraites afin de ne pas limiter l'utilisation de ce Dataset à la recherche d'experts.
- Table pagelinks : https://www.mediawiki.org/wiki/Manual:Pagelinks_table
- Schéma Avro : https://github.com/plawson/oc-project4/blob/master/datalake/pagelinks.avsc
- Table revision : https://www.mediawiki.org/wiki/Manual:Revision_table
- Schéma Avro : https://github.com/plawson/oc-project4/blob/master/datalake/history.avsc

Requêtes Spark SQL et résultats

- Description : https://github.com/plawson/oc-project4/blob/master/documents/Data-Lake-Recherche-dExperts_v1.0.pdf

Temps d’exécution

Script : find_experts.py
 # Executors : 3
Executor memory : 7Go
Driver memory : 5Go

 # Fichiers Révision : 5965
 # Fichiers Pagelink : 52

Spark 2.3.1 **History Server**

Event log directory: hdfs://kube-node07:9000/spark-logs

Last updated: 2018-10-11 01:56:22

Client local time zone: Europe/Paris

App ID	App Name	Started	Completed	Duration	Spark User
application_1537890560442_0031	Dennō_Senshi_Porigon	2018-10-11 01:06:43	2018-10-11 01:51:32	45 min	kubernetes
application_1537890560442_0030	Anthropologie_marxiste	2018-10-11 00:17:27	2018-10-11 01:00:20	43 min	kubernetes
application_1537890560442_0028	Cinéma_surréaliste	2018-10-10 21:09:05	2018-10-10 21:54:56	46 min	kubernetes