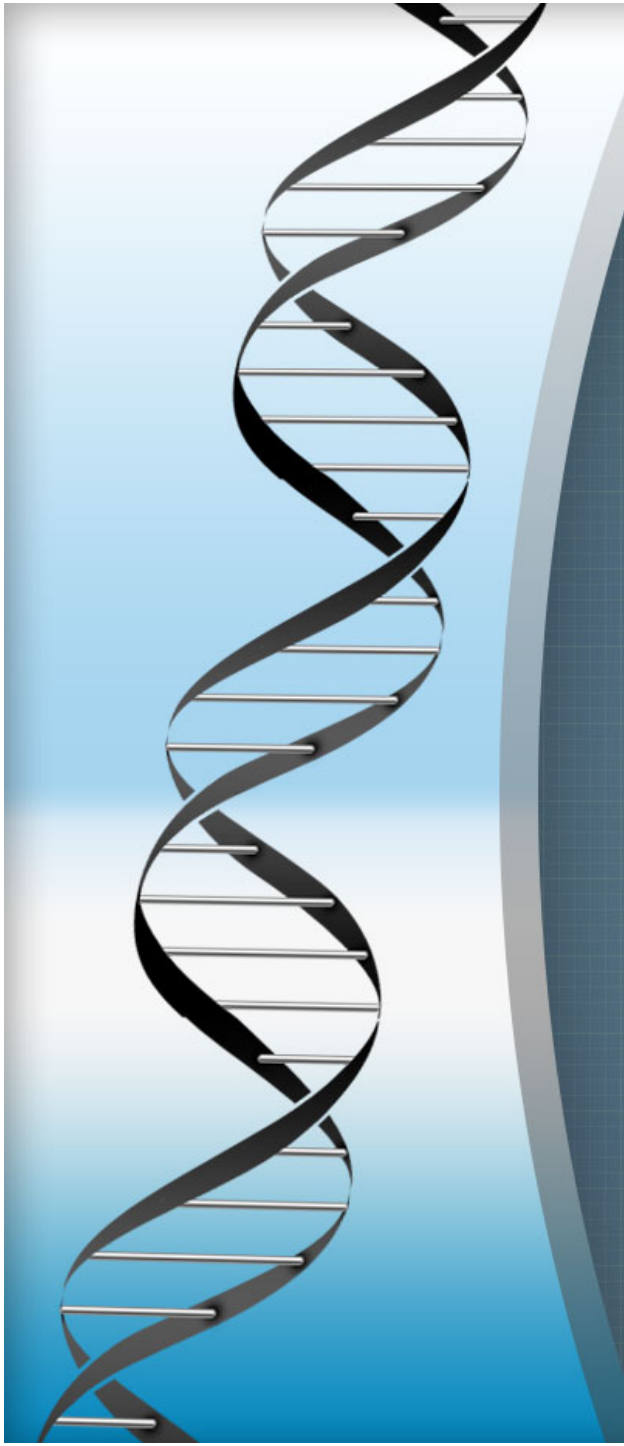


ÉTS

Le génie pour l'industrie

Base de données distribuée appliquée à la génétique dans le cadre de l'analyse du séquençage génomique

J.F.Hamelin, ÉTS

Plan de la présentation

1. Contexte, problématique & objectifs
2. Revue de littérature
    I. Hadoop
    II. MapReduce
    III. HBase
    IV. Sqoop
3. Stratégie de requête du CRCHUM
4. Schéma de données HBase
5. Migration des données vers HBase
6. Stratégie de requête HBase
7. Démonstration de la preuve de concept
8. Optimisation des performances HBase
9. Comparatif des performances
10. Recommandations
11. Retour sur le projet & conclusion
12. Période de questions

Contexte, problématique & objectifs

Contexte

- Parties prenantes: Laboratoire Rouleau & ÉTS
- Besoins du laboratoire Rouleau vs. Besoins prof. April

Problématique

- Solution SQL avec plusieurs millions d'enregistrements
- Saturation imminente
- En quête d'une solution innovatrice viable à long terme

Objectifs du projet

- Comprendre les requêtes problématiques
- Définir un environnement de développement pseudo-distribué
- Créer un environnement Hadoop/HBase en grappe
- Migrer les données vers le schéma HBase
- Reproduire les requêtes sur HBase
- Analyser les performances & émettre des recommandations

Revue de littérature

- Framework open source destiné au développement d'applications distribuées
- Facilite le développement d'application sur un grand nombre de machines contenant jusqu'a des pétaoctets de données
- Inspiré des projets MapReduce et GoogleFS

Plusieurs sous-projets :

- Hbase – Base de données distribué
- Hive – Entrepôt de données
- Mahout – Librairie d'apprentissage machine
- Avro – Système de sérialisation de données
- Plusieurs autres…

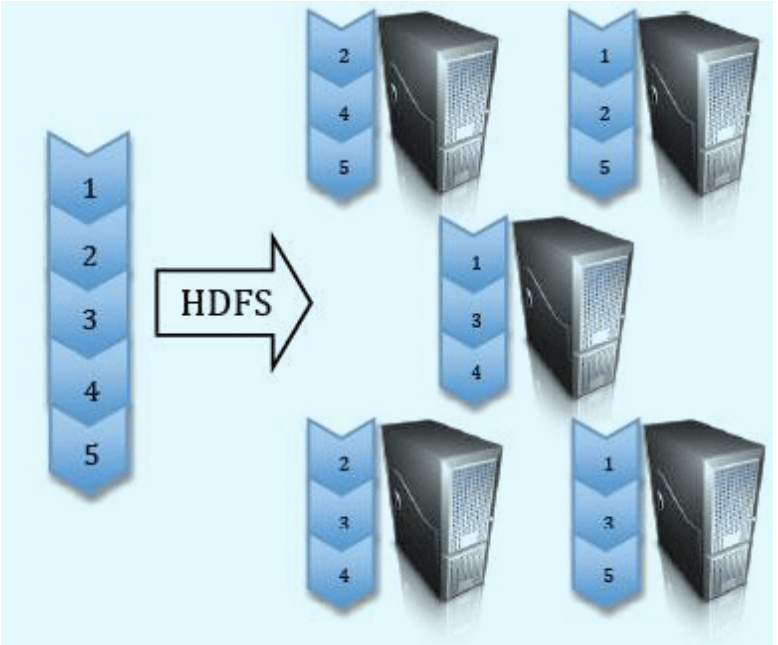
Revue de littérature

HDFS (Hadoop Distributed File System)

•Les fichiers ont enregistré dans des blocks de données (typiquement 64MB ou 128MB)

•Les blocks de données sont distribué sur les nœuds du cluster

•Architecture maitre/esclaves



Revue de littérature

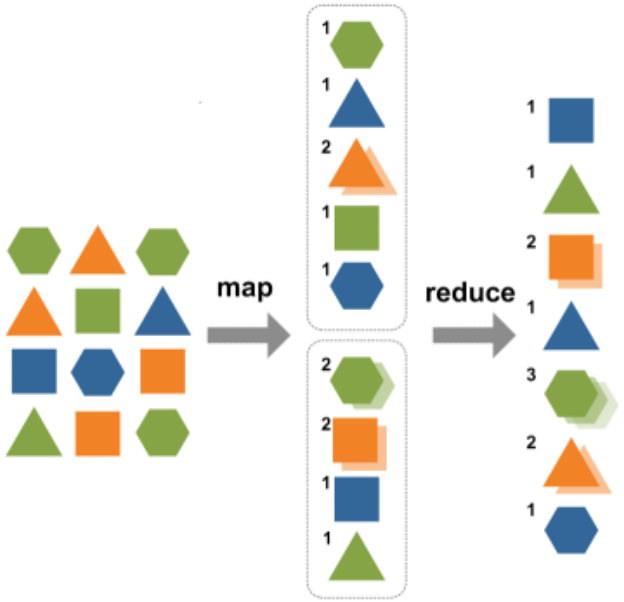
MapReduce

•Implémentation sur Hadoop de la technique développé par Google

•Permet de faire des opérations en parallèle sur les machines du cluster

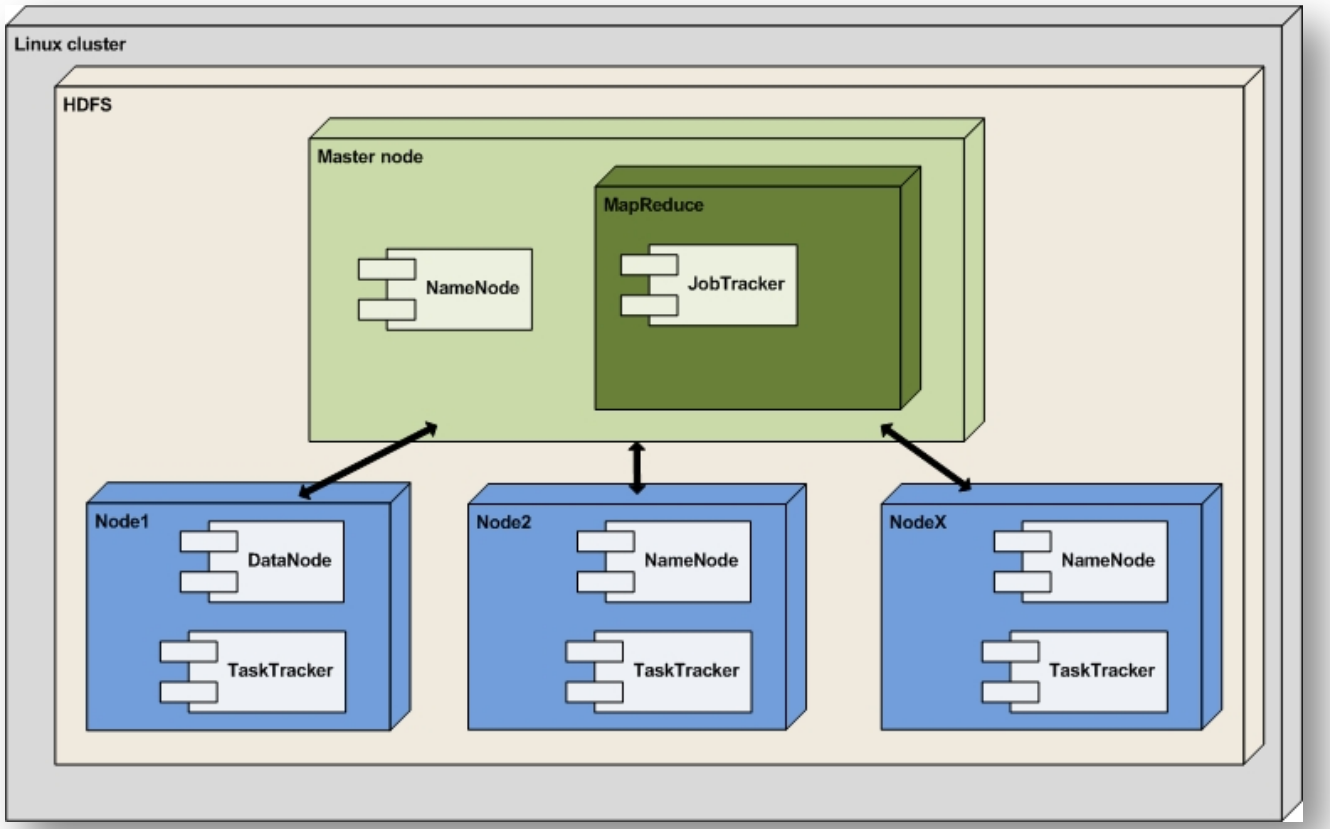
•Traitement effectué en 2 étapes :

- Map
- Reduce



Revue de littérature

Architecture



Revue de littérature

APACHE HBASE

- Inspiré du projet BigTable de Google
- Entrepôt de données très volumineuses extensible et distribué
- Base de données de type “NoSQL”
  - N’utilise pas SQL comme moyen d’interrogation
  - Ne peut garantir les propriétés transactionnelles ACID
  - Repose sur une architecture distribuée et tolérante aux fautes

« Le support pour une conception soutenant de grandes tables clairsemées et orientées colonne élimine souvent la nécessité de normaliser les données et, dans le processus, les coûteuses opérations de jointure nécessaires pour agréger les données au moment de la requête. » - Lars Georges [1]

Revue de littérature

APACHE HBASE

Architecture

- Librairie client
- Serveur maître
- Serveurs de région

ZooKeeper

« Logiciel de gestion de configuration pour systèmes distribués, basé sur le logiciel Chubby développé par Google » [2]

HBase utilise ZooKeeper pour s'assurer que seulement un serveur maître fonctionne, pour stocker l'emplacement du « bootstrap » utilisé lors de la découverte des régions, comme un registre contenant les serveurs de région, ainsi que pour d'autres fins.

Revue de littérature

APACHE HBASE

Modes d'exécution

- Autonome
  - Mode par défaut
  - N'utilise pas HDFS, mais le système de fichiers local
  - Tous les processus liés à HBase et ZooKeeper s'exécutent sur une seule JVM.
  - Très peu performant. Utile pour tester.
- Pseudo-distribué
  - Utilise HDFS
  - Une seule machine compose la grappe
  - Utile pour des tests et du prototypage
- Pleinement distribué
  - Répartis les processus et requêtes sur plusieurs machines de la grappe
  - Nécessite au moins deux machines
  - Mode pour tester les performances et s'exécuter en environnement de production

Revue de littérature

Apache Sqoop

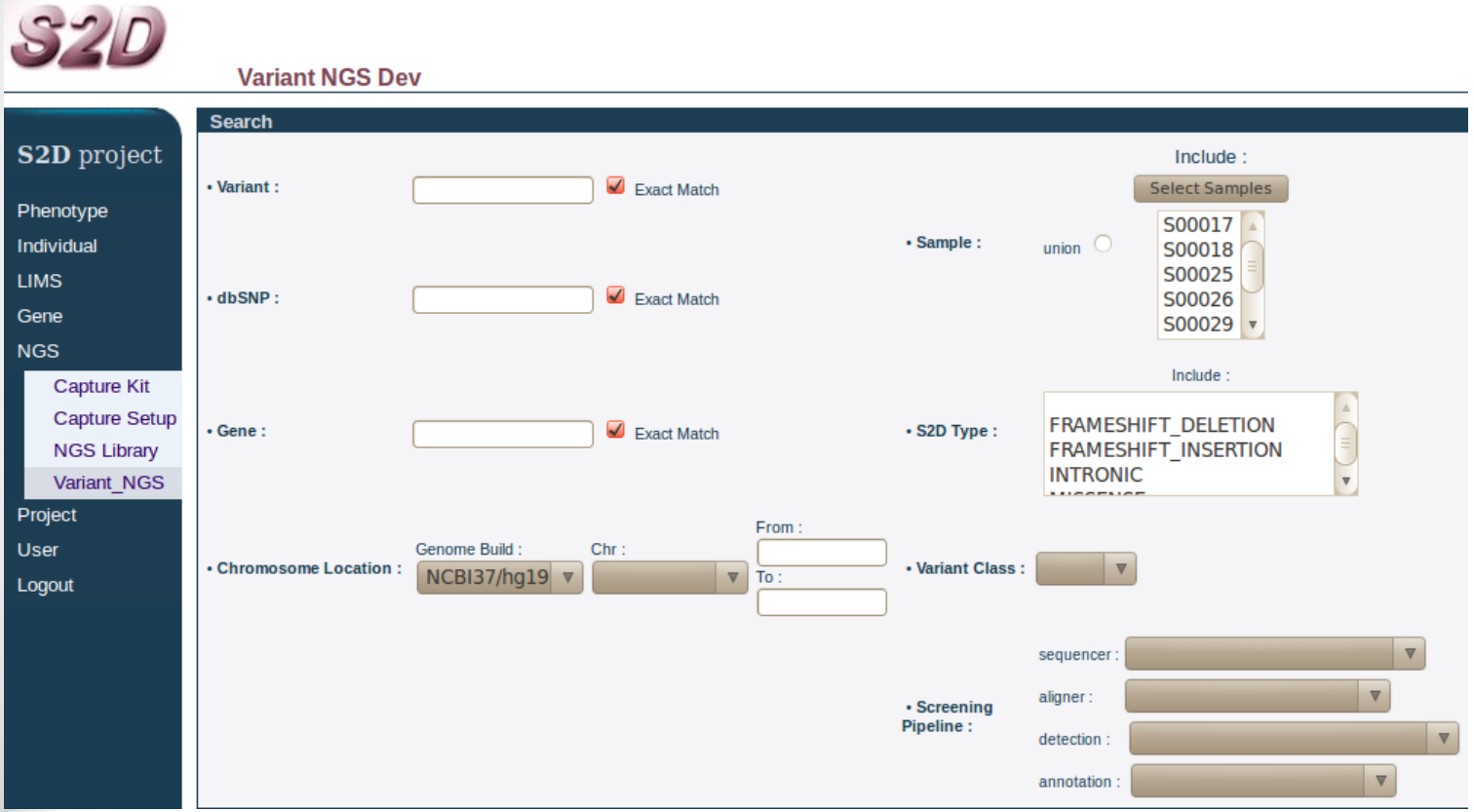
« *Outil conçu pour transférer efficacement les données en vrac entre Apache Hadoop et des bases de données plus conventionnelles telles que les bases de données relationnelles.* »
- The Apache Software Foundation [3]

•Fonctionne dans les deux sens
•Supporte les bases de données suivante:
- HSQLDB version 1.8.0 +
- MySQL version 5.0 +
- Oracle version 10.2.0 +
- PostgreSQL version 8.3+

•Nécessite les pilotes JDBC de chacune pour fonctionner
•Utilise MapReduce

Système S2D du laboratoire Rouleau

- Application client développé en Perl
- Base de données MySQL avec l'engin Infobright



Base de données

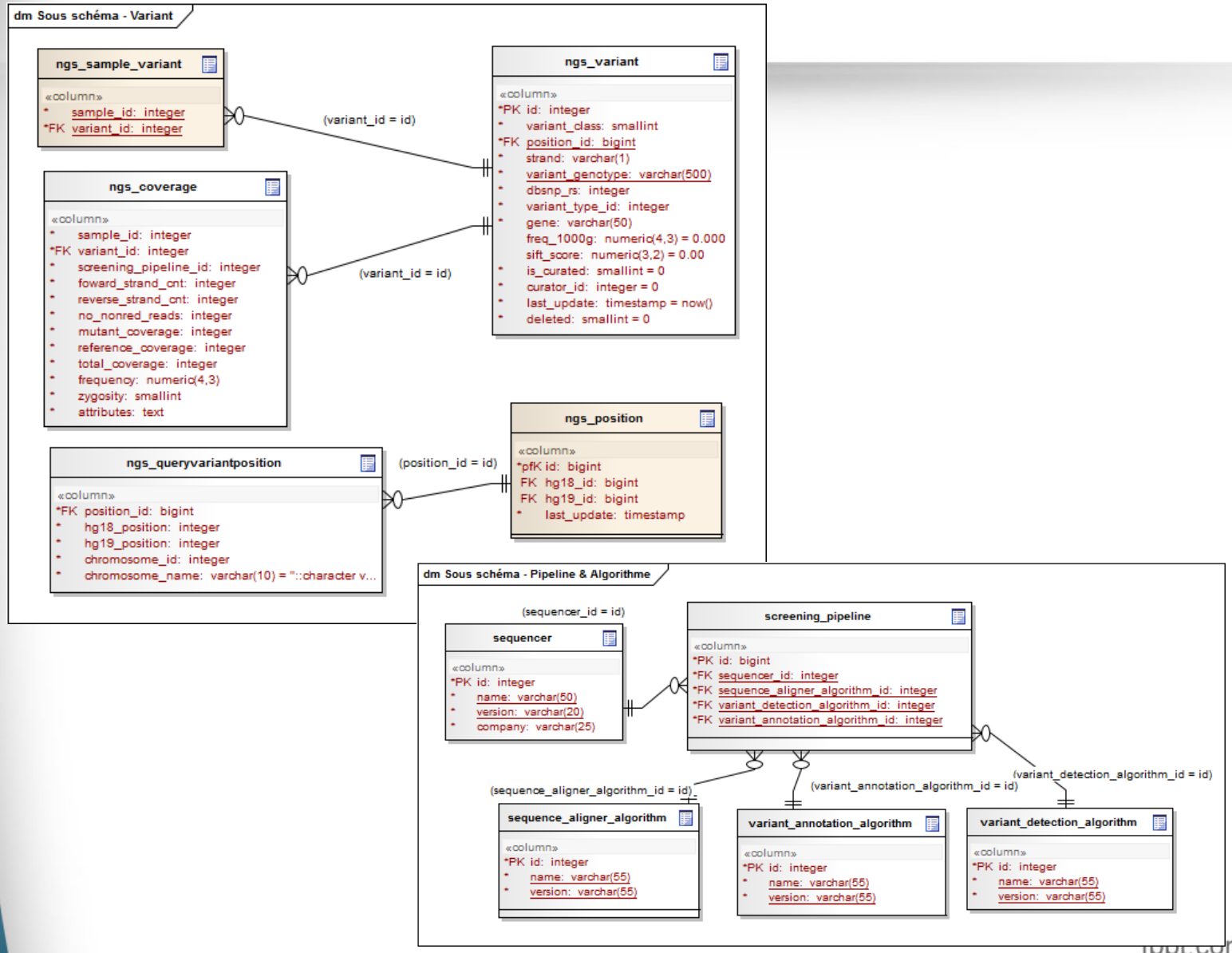


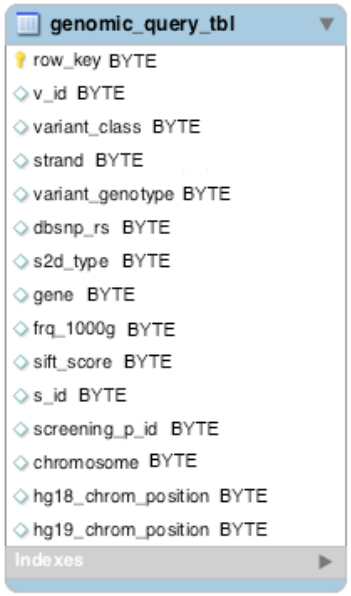
Schéma de données HBase

- Une grande table pour la requête finale
- Clé de rangée intelligente:

```
# Format de la clé de chaque rangée
variant_type_id|sample_id|pipeline_id|variant_id
```

- Colonnes = extrants, parties de la clé = paramètres variables

Structure de la table:



Migration des données vers HBase

1. Création d'une vue SQL dans PostgreSQL qui modélise la structure de la table HBase
   - La vue contient toutes les informations à afficher à l'utilisateur en guise de résultat
   - La vue est l'endroit où le format de la clé HBase est défini

```
(
        to_char(v.variant_type_id, '000000') ||
        '|' ||
        to_char(c.sample_id, '000000') ||
        '|' ||
        to_char(c.screening_pipeline_id, '000000') ||
        '|' ||
        to_char(v.id, '000000000000000')
) AS row_key
```

2. Transfert des données via Sqoop

Migration des données vers HBase

```
$ sqoop import
--connect jdbc:postgresql://localhost:5432/postgres
--username _postgres
--query "select row_key, s2d_type, v_id, variant_class,
strand, variant_genotype, dbsnp_rs, gene, freq_1000g,
sift_score, chromosome, hg18_chrom_position,
hg19_chrom_position, s_id from \"NGS_query_view\" where \
$CONDITIONS"
--hbase-table genomic_query_tbl
--columns
row_key,s2d_type,v_id,variant_class,strand,variant_genotyp
e,dbsnp_rs,gene,freq_1000g,sift_score,chromosome,hg18_chro
m_position,hg19_chrom_position,s_id
--column-family cf_v_c_qvp
--hbase-row-key row_key
--hbase-create-table
--split-by row_key
```

Transfert de 8 158 083 millions d'enregistrements en 15 minutes (représente ~25% des données)

Stratégie de requête sur HBase

Get :

- Requête permettant de récupérer une rangé dans la base de données
- On doit connaitre la clé de la rangé que l'on veut récupérer

Scan :

- Permet d'itérer sur un ensemble de rangés
- Permet de limiter la recherche de données en spécifiant une rangé de départ et de fin
- Permet de filtrer les données balayées à l'aide de filtres personnalisés
- L'itérateur est exécuté sur le RegionServer

Stratégie de requête sur HBase

Utilisation du scan avec des filtres

000001 | 001122 | 000020 | 000000000012230

Variant type id **Sample Id** **Pipeline Id** **Variant id**

Génération d'expressions régulières en fonction des choix de l'utilisateur

(000003) \| (013127)|(013141) \| [0-9]{6}\| [0-9]{15}

Variant type id **Sample Id** **Pipeline Id** **Variant id**

Performance :

- Un seul scan
- Plusieurs scans dans un seul thread
- Plusieurs scans exécuté en parallèle

Stratégie de requête sur HBase

Utilisation multiple scans qui itère sur des intervalles paramétrables :

- Les scan sont généré en fonction des paramêtres choisies
- Les expressions régulières sont généré en fonction des intervales de chacun des scans
- Les résultats des scans sont stocké dans un hashmap dont la clé est la clé de la rangé

Exemple de paramètres :

1	50	25	
000001	001122	000020	000000000012230
Variant type id	Sample Id	Pipeline Id	Variant id

Stratégie de requête sur HBase

1 50 25

####### | ####### | ####### | ##################

Variant type id | Sample Id | Pipeline Id | Variant id

Si l'utilisateur choisie :

- Variant id : 000001, 000004
- Sample id : 000001, 000040, 001140, 001170
- Pipeline : 000001 à 000024

Il y aurait 4 scans pour itérer sur les intervalles suivants :

000001 | 000001 | 000000 | 000000000000000 - 000001 | 000040 | 000024 | ##################

000001 | 001140 | 000000 | 000000000000000 - 000001 | 001170 | 000024 | ##################

000004 | 000001 | 000000 | 000000000000000 - 000001 | 000040 | 000024 | ##################

000004 | 001140 | 000000 | 000000000000000 - 000001 | 001170 | 000024 | ##################

Stratégie de requête sur HBase

Gestion des exclusions :

- Générer des expressions régulières qui permettent d'exclure des variants
- Exclusions des variants durant le balayage des données en utilisant un Hashmap référençant les ids à exclures

Gestion des exclusions sur MapReduce :

- Exclusions des variants dans l'étape Reduce en utilisant un Hashmap référençant les ids à exclures

Démonstration

Application Cliente :

- « Single Page Interface » (SPI)
- ExtJS 4.1

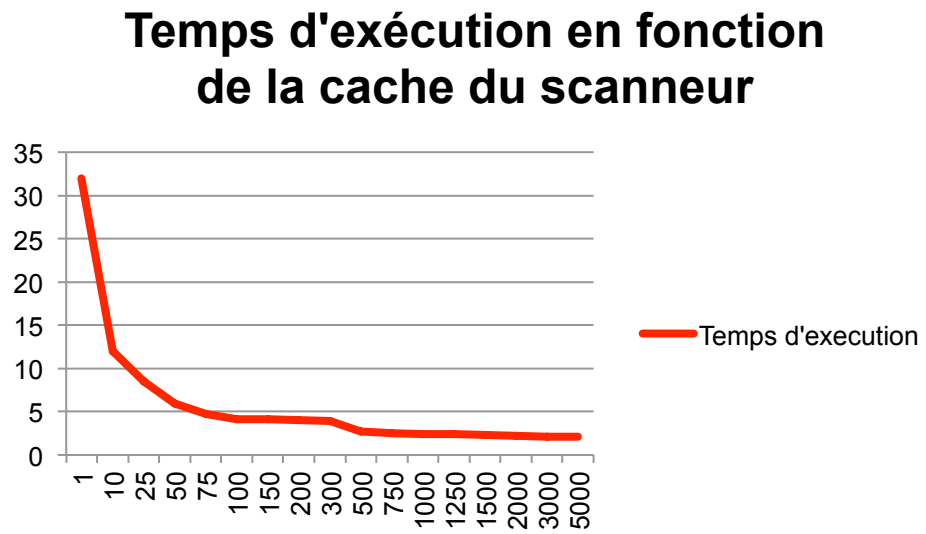
Application Serveur :

- Tomcat 7.2
- Spring MVC
- Services Web « Restful »
- Hadoop 1.03
- HBase 0.92.1

Optimisation des Performances HBase

Scanner caching :

- Permet de gérer le nombre de rangés qui sont traité avant d'envoyer les résultats soient envoyés au client



Optimisation des Performances HBase

Utilisation de la compression :

- Augmente presqu'à coup sûr les performances du système
- L'utilisation du CPU pour décompresser les fichiers est moins couteuse que la lecture des données sur le disque
- Google Snappy – Librairie de compression et décompression Utilisé dans BigTable et maintenant disponible pour HBase

Bloom Filter :

- Structure de données probabiliste permettant de savoir si un élément se retrouve dans un ensemble.
- Au niveau de Hbase, il permet de vérifier si une clé se retrouve dans un block de données sans avoir à le charger en mémoire
- L'index du filtre est stocké en mémoire

Comparatif des performances

HBase vs PostgreSQL :

- Difficile de faire la comparaison puisque la logique pour faire les requêtes n'est pas la même
- Sur notre environnement local, les requêtes sur HBase sont entre 2 et 4 fois plus rapide que sur PostgreSQL

Recommendations

- Utiliser la stratégie de Scan développée dans un contexte MapReduce optimisé. Utiliser une job Map par objet Scan et lancer le tout en parallèle dans un environnement distribué plus grand que trois noeuds. Le Reducer serait utiliser pour effectuer les exclusions de variants en fonction des échantillons.

- Revoir le schéma de données du laboratoire Guy Rouleau dans son ensemble. Nous avons uniquement eu un ensemble de données limité (et qui date de l'an dernier) sur lequel travailler. Le schéma SQL du CRCHUM a changé depuis. Revoir entièrement le schéma HBase sur réception du schéma SQL complet du laboratoire Rouleau.

- Revoir la composition de la clé de rangée selon l'occurrence des paramètres de recherche. La composition de la clé est très importante, puisqu'elle définit l'ordonnancement effectué par HBase.

Recommendations

- Aller chercher une rétroaction du laboratoire Rouleau face au travail accomplit. Itérer sur leur commentaires.
- Tester d'autres solutions s'appuyant sur des bases de données distribuées, telles que HyperTable et Cassandra. Comparer les performances en fonction des besoins du laboratoire Rouleau. Comparer la complexité de mise en place en fonction des besoins du professeur April.
- Évaluer la qualité et la performance du système SeqWare, un projet libre complexe utilisant Hadoop et HBase, qui se spécialise dans la gestion, la recherche et les « workflow » de données génomiques
- Il n'y a de limites que celles que l'on s'impose lorsqu'il est question d'amélioration logicielle. Les possibilités d'amélioration d'un tel système sont infinies.

Conclusion

- ✓ Compréhension des requêtes problématiques de S2D
- ✓ Création d'un environnement Hadoop/HBase pseudo-distribué sous Mac OS X Lion et Ubuntu (Linux)
- ✓ Création d'un environnement Hadoop/HBase pleinement distribué sous Ubuntu (Linux) composé de trois serveurs
- ✓ Migration des données SQL vers HBase
- ✓ Transposition des requêtes PostgreSQL sur HBase à l'aide de chaîne de filtres à expression régulières
- ✓ Analyse et comparaison des performances en fonction des différentes configurations et versus le système PostgreSQL
- ✓ Implémentation d'une application Web permettant de tester les requêtes et de modifier les paramètres Hadoop/HBase

Pour en savoir plus concernant cette présentation vous pouvez consulter le rapport de projet de fin d’études des étudiants