Bases de données documentaires et distribuées, http://b3d.bdpedia.fr

Introduction Ãă Spark

Qu'est-ce que Spark ?

Un **moteur d'exécution** basé sur des opérateurs de haut niveau, comme Pig.

Comprend des opérateurs Map/Reduce, et d'autres opérateurs de second ordre.

Introduit un concept de collection résidente en mémoire (RDD) qui améliore considérablement certains traitements, dont ceux basés sur des itérations.

De nombreuses librairies pour la fouille de données (MLib), le traitement des graphes, le traitement de flux (*streaming*).

Les *Resilient Distributed Datasets* (RDD)

C'est le concept central : **Un RDD est une collection (pour en rester à notre vocabulaire) calculée à partir d'une source de données** (MongoDB, un flux, un autre RDD) .

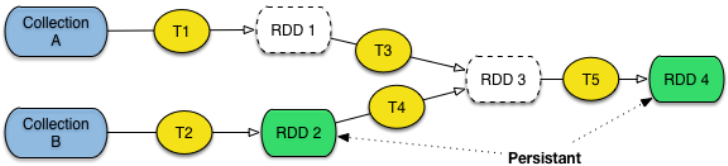
Un RDD peut être marqué comme **persistant** : il est alors placé en mémoire RAM et conservé par Spark.

Spark conserve **l'historique des opérations** qui a permis de constituer un RDD, et la reprise sur panne s'appuie sur cet historique afin de reconstituer le RDD en cas de panne.

Un RDD est un "bloc" **non modifiable**. Si nécessaire il est **entièrement recaculé**.

Un *workflow* avec RDD dans Spark

Des **transformations** (opérateurs comme dans Pig) créent des RDD à partir d'une ou deux sources de données.



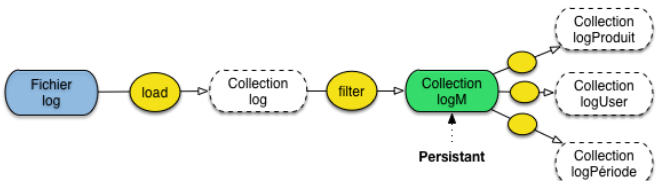
Les RDD persistants sont en préservés en mémoire RAM, et peuvent être réutilisés par plusieurs traitements.

Exemple : analyse de fichiers log

On veut analyser le fichier journal (*log*) d'une application dont un des modules (M) est suspect.

On construit un programme qui charge le *log*, ne conserve que les messsages produits par le module *M* et les analyse.

On peut analyser par produit, par utilisateur, par période, etc.



Spécification avec Spark

Première phase pour construire logM

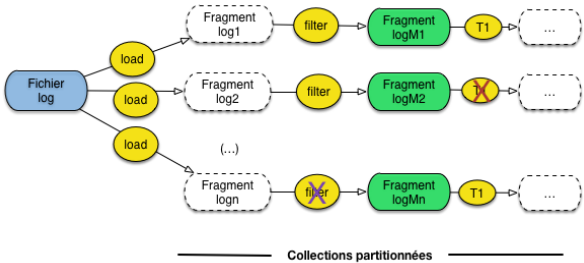
```
// Chargement de la collection
log = load ("app.log") as (...)
// Filtrage des messages du module M
logM = filter log with log.message.contains ("M")
// On rend logM persistant !
logM.persist();
```

Analyse à partir de logM

```
// Filtrage par produit
logProduit = filter logM
   with log.message.contains ("product P")
// .. analyse du contenu de logProduit
```

Reprise sur panne dans Spark

Un RDD est une collection **partitionnée**.



Panne au niveau d'un nœud implique un recalcul basé sur le fragment F persistant qui précède ce nœud dans le workflow.

Dataframes et *Datasets*

Un RDD, du point de vue du programmeur, c'est un conteneur d'objets java.

Le type précis de ces objets n'est pas connu par Spark. Du coup :

- Tout ce que Spark peut faire, c'est appliquer la sérialisation/désérialisation java
- Aucun accès aux objets grâce à un langage déclaratif n'est possible.
- Et donc pas d'optimisation, et la nécessité de tout écrire sous forme de fonctions java.

Depuis la version 1.6 : on dispose de RDD améliorés : les *Datasets*. On peut les traiter comme des tables relationnelles.

Finalement, le schéma c'est utile, et le relationnel, c'est bien !