MapReduce – présentation synthétique

Module Cloud Computing & Big Data

Catarina FERREIRA DA SILVA & Mahmoud BARHAMGI

Catarina.Ferreira@Univ-Lyon1.fr

2016

MapReduce

Introduction

- Introduit par Google
- C'est un modèle de programmation permettant de résoudre un problème plus rapidement en le divisant en plusieurs sous problèmes qui peuvent être résolus en parallèle (par un ensemble de processeurs ou nœuds)
- Il est surtout utilisé pour traiter et analyser de très grands volumes de données en distribuant les taches à plusieurs nœuds (i.e., serveurs)
  - L'ensemble de nœuds est appelé un *Cluster*
  - Le déplacement de données est limité aux maximum pour minimiser le coût des transferts réseau (i.e., les sources de données sont traitées par les nœuds qui leur sont proches)

Principe de MapReduce (1/2)

Le Framework de MapReduce est composé de trois opérations

- **Map**: applique une fonction à une collection de données
  - La fonction émet (comme sortie) des couples clé-valeur
  - Un nœud qui exécute une partie de Map est appelé *Mapper*
- **Shuffle**: les sorties de Map sont organisées (par le système) en fonction de leurs clés et envoyées à des nœuds appellés *Reducers*
- **Reduce**: applique une fonction d'agrégation sur les données intermédiaires pour produire les sorties finales
  - Un nœud qui exécute une partie de Reduce est appelé *Reducer*

Principe de MapReduce (2/2)

Exemple (simple): *Nous souhaitons compter les nombres d'apparition de tous les mots dans un ensemble de documents*

- **Input**: un ensemble de documents
- **Map**: pour chaque apparition d'un mot "*w*" la fonction Map émet un couple <*w*, 1>
  - C'est le système qui désigne les *Mappers*
  - L'affectation des documents aux *Mappers* est réalisée (par le system) en respectant le principe de localité
- **Shuffle**: les couples <*w*, 1> de tous les *Mappers* sont organisés en fonction de leurs clés *w*, et envoyés aux *Reducers*. Les couples ayant la même clé sont traités par le même *Reducer*
- **Reduce**: chaque *Reducer* additionne la valeur 1 de tous les couples qu'il reçoive. Le système ensuite collecte les sorties de tous les *Reducers* pour construire la sortie finale

MapReduce en MongoDB

MapReduce est utilisé en MongoDB pour réaliser des requêtes d'agrégation sur des collections de documents

```
Collection
   ↓
db.orders.mapReduce(
        map      ──►  function() { emit( this.cust_id, this.amount ); },
        reduce   ──►  function(key, values) { return Array.sum( values ) },
                      {
        query    ──►    query: { status: "A" },
        output   ──►    out: "order_totals"
                      }
                   )
```

Quelle est la somme totale de toutes les commandes avec un statut "A" par client?