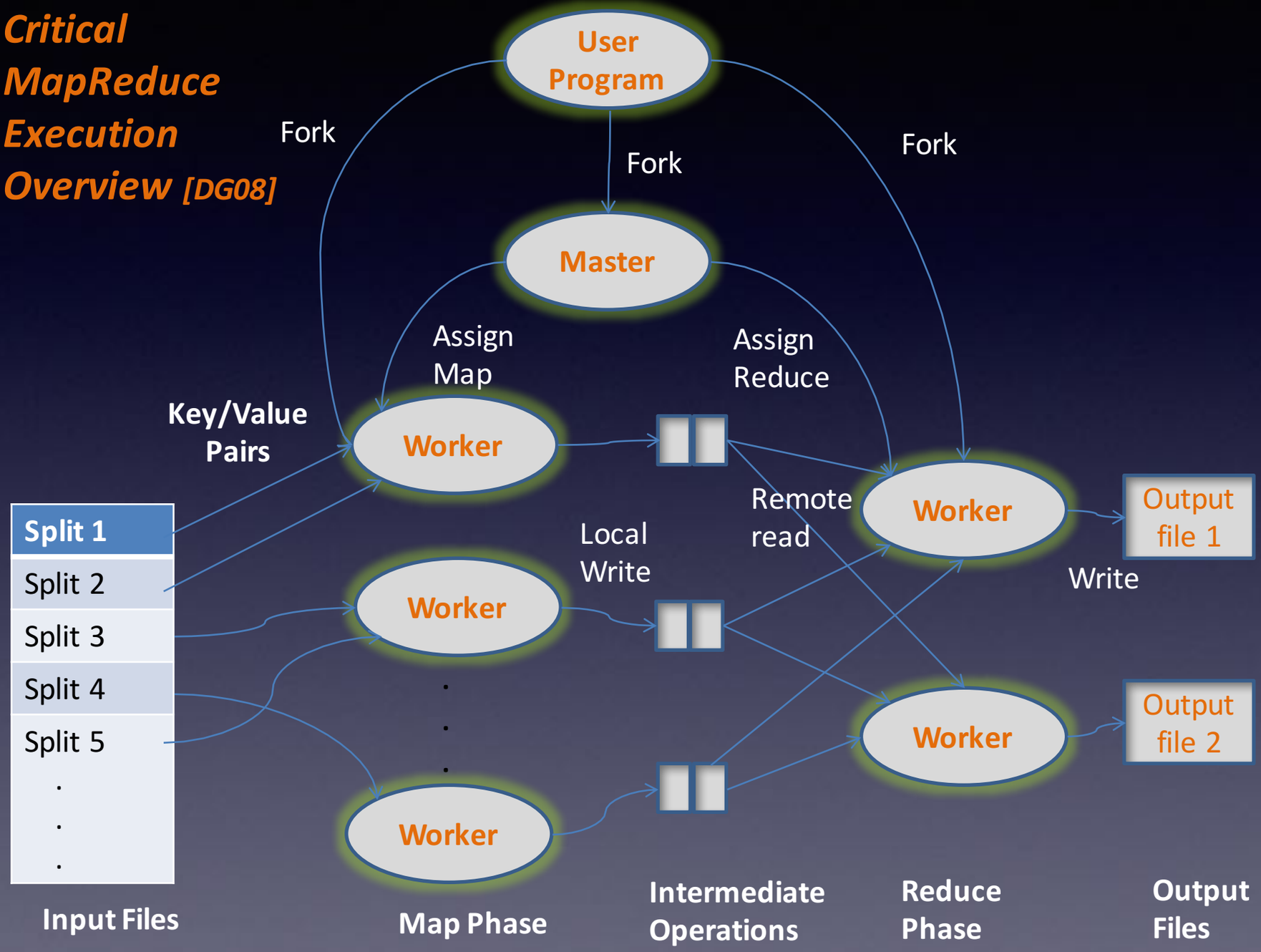
NoSQL 4-5

MapReduce, Exercices

Objectifs (aujourd'hui)

- Principes MapReduce
- Exemple
- MapReduce et mongoDB
- Exemple
- TP

Critical MapReduce Execution Overview [DG08]



Le paradigme MapReduce

- MapReduce a été introduit par Google en 2004
- MapReduce est :
  - Un modèle de programmation,
  - avec un schéma très contraint,
  - qui permet :
    - parallélisation automatique,
    - de l'equilibrage de charge,
    - des optimisations sur les transferts disques et réseaux,
    - de la tolérance aux pannes

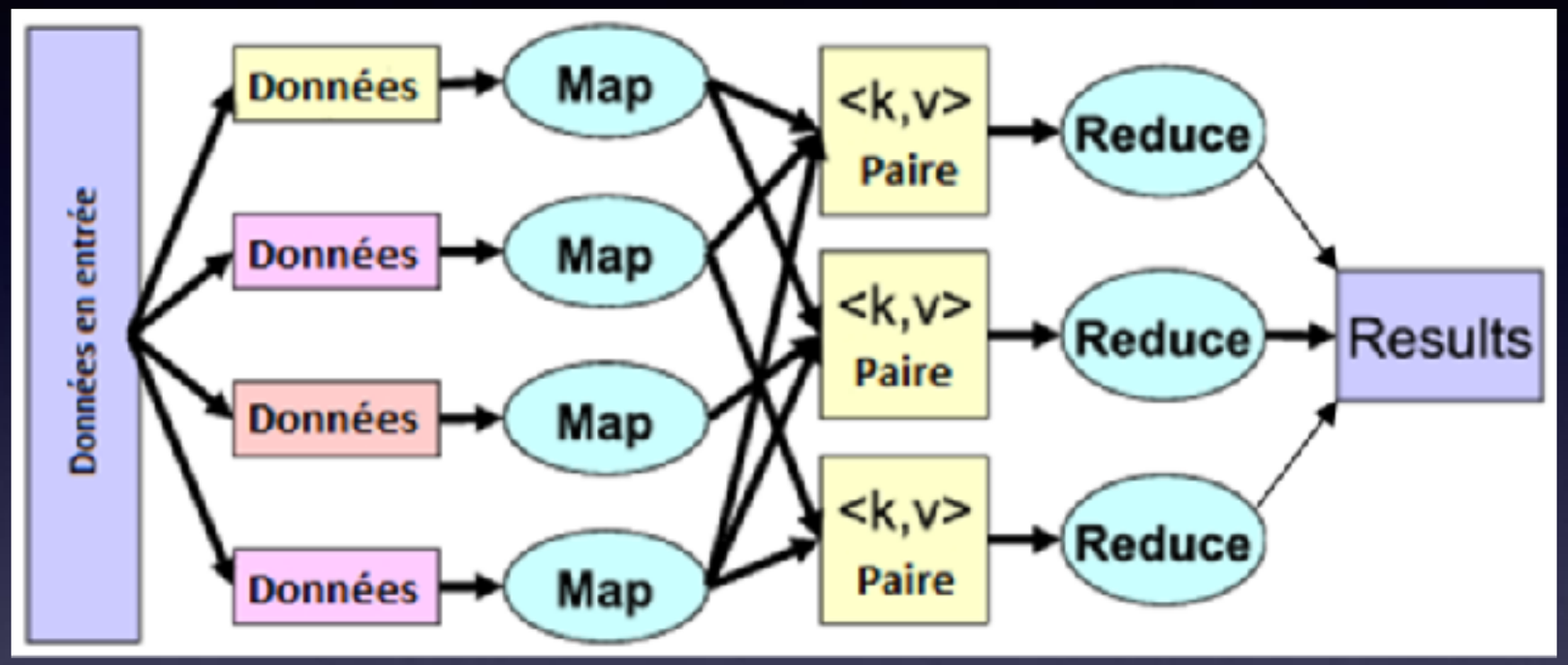
Qui ?

- Yahoo! Search: Webmap (application Hadoop sur un cluster Linux avec plus de 10.000 coeurs)
- Google: la taille d'une seule phase de computation de l'index a baissé de 3.800 lignes de code (C++) à 700 lignes de code avec MapReduce
- Facebook: plusieurs clusters Hadoop, le plus gros 2.500 coeurs cpu
- Hadoop utilisé par Twitter, Amazon, Rackspace, LinkedIn, IBM, Microsoft, etc.

MapReduce

- Resolution d'un problème de manière distribuée
  - Découpage en sous-problèmes
  - Execution des sous-problèmes sur les différentes machines du cluster
  - Stratégie algorithmique dite du *Divide et Impera*
- MapReduce
  - Paradigme de programmation parallèle visant à généraliser les approches existantes pour produire une approche unique applicable à tous les problèmes
  - Origine du nom: languages fonctionnels
  - Calcul distribué: "MapReduce: Simplified Data Processing on Large Clusters" [Google, 2004]

Étapes



- Deux étapes principales :
  - MAP : Emission de paires <clé, valeur> pour chaque donnée d'entrée lue
  - Reduce : Regroupement des valeurs de clé identique et application d'un traitement sur ces valeurs de clé commune

Étapes

- Écrire un programme Map Reduce :

1. Choisir une manière de découper les données afin que Map soit parallélisable
2. Choisir la clé à utiliser pour notre problème
3. Écrire la fonction pour l'opération Map
4. Écrire la fonction pour l'opération Reduce

Comptage de mots

- Exemple classique : le comptage de mots
  - Fichiers d'entrée textuels
  - On veut connaître le nombre d'occurrences de chacun des mots dans ces fichiers
- Il faut décider :
  - De la manière dont on découpe les textes
  - Des couples <clé, valeur> à émettre lors du Map appliqué à chaque morceau de texte
  - Du traitement à opérer lors du regroupement des clés communes (Reduce)

Fichier d'entrée

Celui qui croyait au ciel
Celui qui n'y croyait pas
[…]
Fou qui fait le délicat
Fou qui songe à ses querelles

(Louis Aragon, *La rose et le Réséda*, 1943, fragment)

- Pour simplifier, on retire tout symbole de ponctuation et caractères spéciaux. On passe l'intégralité du texte en minuscules

Fichier d'entrée

celui qui croyait au ciel

celui qui ny croyait pas

fou qui fait le délicat

fou qui songe a ses querelles

Découpage des données d'entrée : par exemple par ligne

- Ici, 4 unités de traitement après découpage

Map

- Opération Map :
  - Séparation de l'unité en mots (selon les espaces)
  - Emission d'une paire <mot,1> pour chaque mot

Map

(celui;1) (celui; 1)

(qui;1) (qui;1) (qui;1) (qui;1)

(croyait;1) (croyait;1)

(au;1) (ny;1) (ciel;1) (pas;1)

(fou;1) (fou;1)

(fait;1) (le;1) (delicate;1)

(a;1) (ses;1)

(querelles;1)

- Après le Map : regroupement (shuffle) des clés communes
  - Effectué par un tri distribué

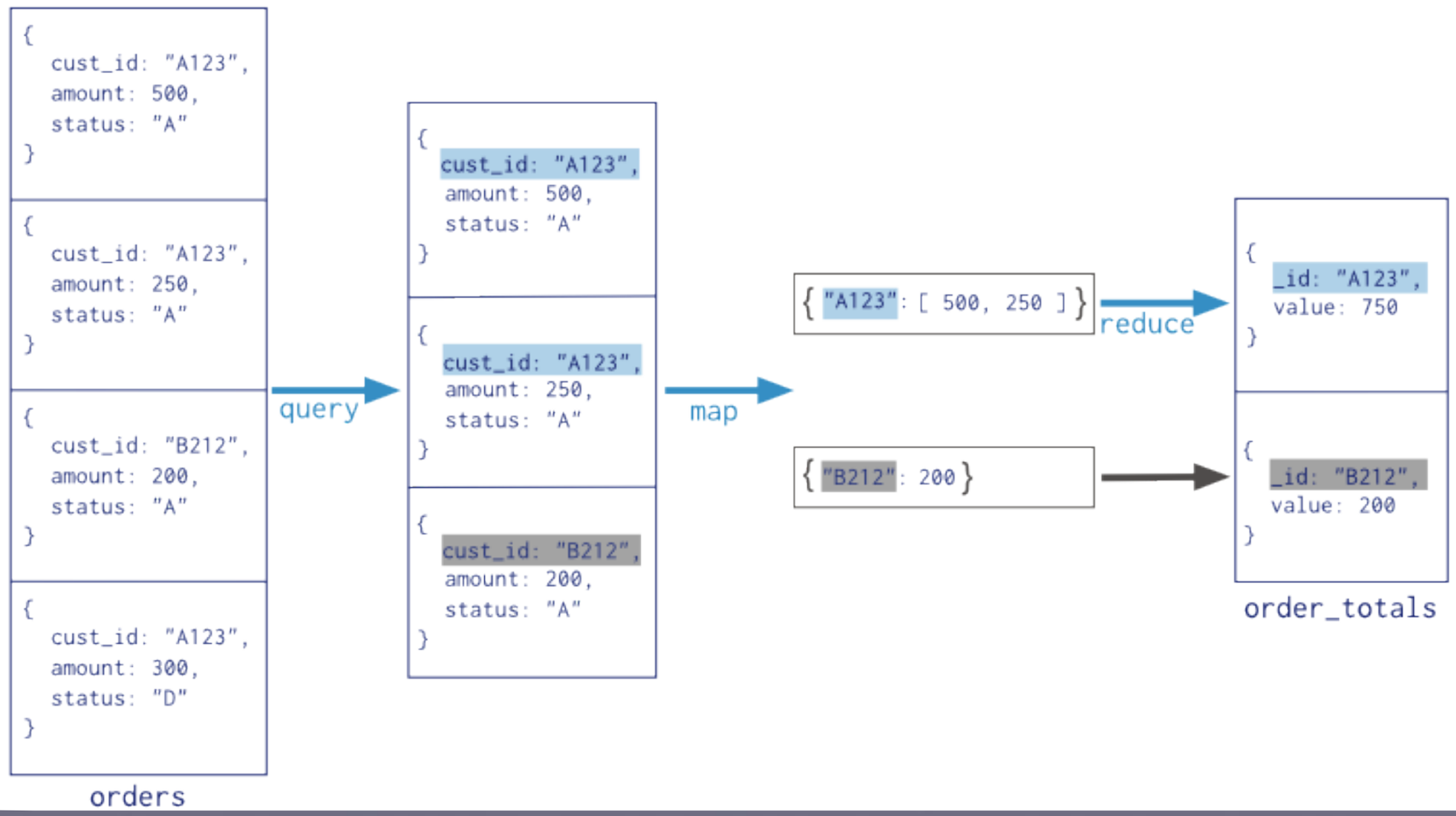
Reduce

- Opération Reduce :
  - Sommation des valeurs de toutes les paires de clé commune
  - Ecriture dans (ou des) fichier(s) résultats

qui: 4
celui: 2
croyait: 2
fou: 2
au: 1
ciel: 1
ny: 1
pas: 1
fait: 1
[…]

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"
                        }
                      )
```



Exemple

Prix total par client

```
{
    _id: ObjectId("50a8240b927d5d8b5891743c"),
    cust_id: "abc123",
    ord_date: new Date("Oct 04, 2012"),
    status: 'A',
    price: 25,
    items: [ { sku: "mmm", qty: 5, price: 2.5 },
             { sku: "nnn", qty: 5, price: 2.5 } ]
}
```

```
var mapFunction1 = function() {
                       emit(this.cust_id, this.price);
                   };
var reduceFunction1 = function(keyCustId, valuesPrices) {
                          return Array.sum(valuesPrices);
                      };
```

Exemple

Prix total par client

```
db.orders.mapReduce(
                        mapFunction1,
                        reduceFunction1,
                        { out: "map_reduce_example" }
                    )
```

Exemple

Calculer la commande et la quantité totale avec la quantité moyenne par article

```
var mapFunction2 = function() {
                        for (var idx = 0; idx < this.items.length; idx++) {
                            var key = this.items[idx].sku;
                            var value = {count: 1, qty: this.items[idx].qty};
                            emit(key, value);}};
```

```
var reduceFunction2 = function(keySKU, countObjVals) {
                        reducedVal = { count: 0, qty: 0 };
                        for (var idx = 0; idx < countObjVals.length; idx++) {
                            reducedVal.count += countObjVals[idx].count;
                            reducedVal.qty += countObjVals[idx].qty;
                        }
                        return reducedVal;
                    };
```

Exemple

Calculer la commande et la quantité totale avec la quantité moyenne par article

```
var finalizeFunction2 = function (key, reducedVal) {

                        reducedVal.avg = reducedVal.qty/reducedVal.count;

                        return reducedVal;

                    };

db.orders.mapReduce( mapFunction2,
                     reduceFunction2,
                     {
                       out: { merge: "map_reduce_example" },
                       query: { ord_date:
                                   { $gt: new Date('01/01/2012') }
                              },
                       finalize: finalizeFunction2
                     }
                   )
```

TP2: MapReduce DBLP

- Téléchargez le sujet depuis: http://www.lsis.org/chifua/#teachings (onglet NOSQL - mongoDB → TP)