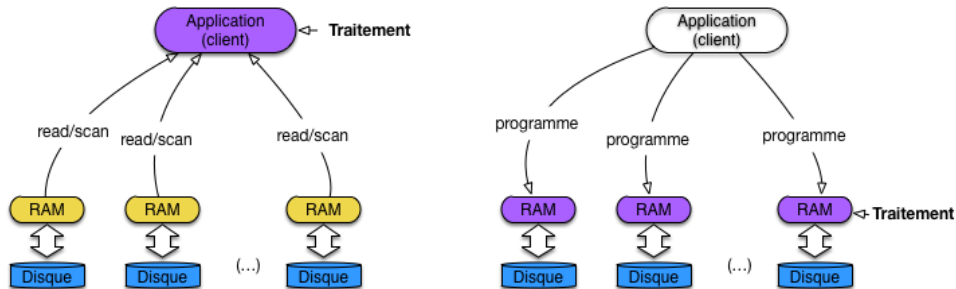


Bases de données  
documentaires et distribuées,  
<http://b3d.bdpedia.fr>

Map Reduce

# Les programmes vers les données, pas l'inverse !

Client serveur pour traiter des TOs de données ? **Ne marche pas.**



Il faut placer les traitements au plus près des données.

# Pourquoi un *framework*

Un framework **pilote** un traitement, conduit selon un processus générique (notion d'inversion de contrôle).

L'application "injecte" des fonctions dans le framework.

Le framework applique les fonctions pendant l'exécution du processus.

- Une fonction *map()* à appliquer pendant la phase de Map
- Une fonction *reduce()* à appliquer pendant la phase de Reduce.

## Rôle d'un *framework* MapReduce

Prendre en charge la distribution, et gérer les reprises sur panne.

# Comptons les mots : la fonction de Map

Phase de Map : on extrait les termes, on les compte (localement), on transmet

```
function mapTF($id, $contenu)
{
  // $id: identifiant du document
  // $contenu: contenu textuel du document

  // On boucle sur tous les termes du contenu
  foreach ($t in $contenu) {
    // Comptons le nb d'occurrences de $t dans $contenu
    $count = nbOcc ($t, $contenu);
    // Emission du terme et de son nombre d'occurrences
    emit ($t, $count);
  }
}
```

NB : rien n'indique le contexte de distribution.

# Comptons les mots : la fonction de Reduce

Phase de Reduce : on reçoit, pour chaque terme, les compteurs, on les additionne.

```
function reduceTF($t, $compteurs)
{
  // $t: un terme
  // $compteurs: les nombres d'occurrences, un pour chaque doc.
  $total = 0;
  // Boucles sur les compteurs et calcul du total
  foreach ($c in $compteurs) {
    $total = $total + $c;
  }
  // Et on produit le total
  return $total;
}
```

Même remarque : rien n'indique le contexte de distribution.



# L'exécution par le *framework*

URL	Document
$u_1$	the jaguar is a new world mammal of the felidae family.
$u_2$	for jaguar, atari was keen to use a 68k family device.
$u_3$	mac os x jaguar is available at a price of us \$199 for apple's new "family pack".
$u_4$	one such ruling family to incorporate the jaguar into their name is jaguar paw.
$u_5$	it is a big cat.

# Déroulons le processus

term	count	
jaguar	1	
mammal	1	
family	1	
jaguar	1	
available	1	Sortie
jaguar	1	
family	1	
family	1	
jaguar	2	
...		

du *map*

Entrée du *shuffle*

# Déroulons le processus

term	count
jaguar	1
mammal	1
family	1
jaguar	1
available	1
jaguar	1
family	1
family	1
jaguar	2
...	

du *map*

Entrée du *shuffle*

Sortie

term	count
jaguar	1,1,1,2
mammal	1
family	1,1,1
available	1
...	

du *shuffle*

Entrée du *reduce*

Sortie



# Déroulons le processus

term	count
jaguar	1
mammal	1
family	1
jaguar	1
available	1
jaguar	1
family	1
family	1
jaguar	2
...	

du *map*  
Entrée du *shuffle*

Sortie

term	count
jaguar	1,1,1,2
mammal	1
family	1,1,1
available	1
...	

du *shuffle*  
Entrée du *reduce*

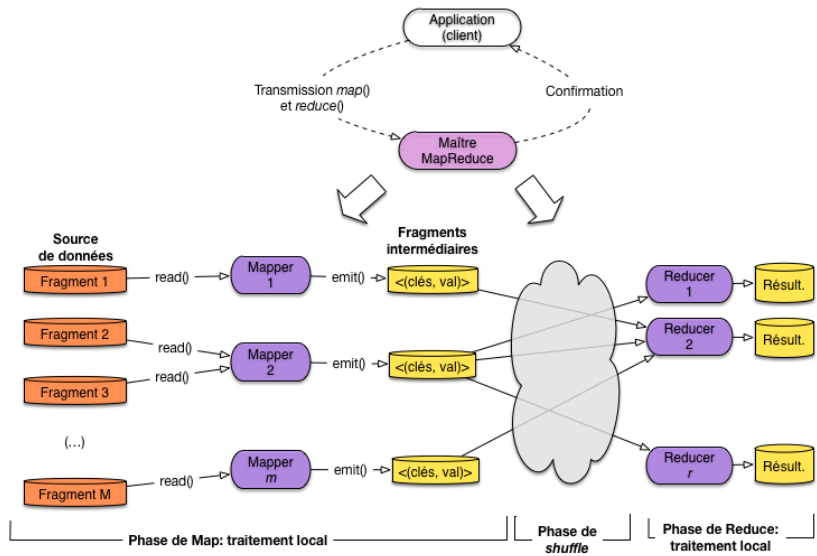
Sortie

term	count
jaguar	5
mammal	1
family	3
available	1
...	

final

Résultat

# La grande vision



# Quelques calculs

Supposons une collection avec 1 million de documents ; chaque document contient 100 termes en moyenne, et sa taille est (en moyenne) de 1 KO.

Le partitionnement découpe en fragments de 64 MO : chaque fragment contient 64 000 documents. Il y a à peu près  $M = \lceil 1,000,000/64,000 \rceil \approx 16,000$  fragments.

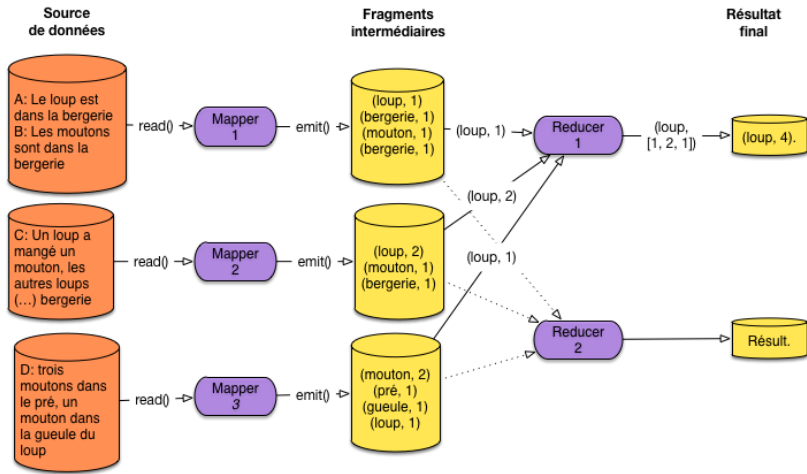
Si j'ai 16 machines, chacune traitera 1 000 fragments par 1 000 tâches.

Chaque tâche (Mapper) produit 6 400 000 paires (terme, compteur)

Si j'ai 10 machines pour la phase de Reduce : chaque Reducer  $R_i$  reçoit 640 000 paires de chaque Mapper, pour tous les termes  $t$  tels que  $hash(t) = i$  ( $1 \leq i \leq 10$ )

# Un tout petit exemple

Quatre documents qui parlent de loups, de moutons, de bergerie.



# Gestion des pannes

Un traitement MapReduce peut durer des jours, sur des centaines de machines : la probabilité de panne est très grande.

Ré-exécuter à chaque panne  $\Rightarrow$  : on ne finira jamais

Dans Hadoop, le Maître (*JobTracker*) surveille l'ensemble du processus.

- Panne d'un Reducer : on reprend le résultat des Mappers, **stocké sur disque**.
- Panne d'un Mapper : on recommence la tâche sur le fragment (**ou sur un réplica**)
- Si le Maître tombe en panne : on recommence tout, (probabilité très faible).

## Essentiel

Tout repose sur une **sérialisation** (écriture sur disque) aux différentes étapes. **Robuste mais très lent**.