



L'ÉCOLE NATIONALE SUPÉRIEURE
D'ÉLECTROTECHNIQUE, D'ÉLECTRONIQUE,
D'INFORMATIQUE, D'HYDRAULIQUE ET DES
TÉLÉCOMMUNICATIONS

SYSTÈMES CONCURRENTS ET COMMUNICANTS

Projet Hadoop 2: Rapport d'avancement 1

Soumis par les étudiants de 2A SN

HPC :

Wilfried L. Bounsi

Rachid Elmontassir

Charles Meyer-Hilfiger

Obeida Zakzak

Année académique 2019/2020

Table des matières

1	Corrections et modifications	2
1.1	Introduction du déploiement centralisé	2
1.2	Unité de données pour les jeux de tests	2
2	Mode d'emploi	3
2.1	Installation	3
2.2	Déploiement	3
2.3	Utilisation	3
2.3.1	Lancement & arrêt des démons	3
2.3.2	Interactions Hdfs	3
2.3.3	Exécution MapReduce	3
2.3.4	Monitoring	4
2.3.5	Evaluations des performances	4
3	Démonstration	5
3.1	Monitoring	5
3.1.1	Affichage des configurations courantes	5
3.1.2	Affichage de l'état des workers	6
3.2	Interactions Hdfs	8
3.2.1	Application de HdfsWrite	8
3.2.2	Application de HdfsRead	8
3.2.3	Application de HdfsDelete	8
3.3	Lancement d'un Job MapReduce	9
3.4	Évaluation des performances	10
4	Etude de la scalabilité	11
4.1	Procédure	11
4.2	Résultats	11
4.3	Interprétation	11
4.3.1	Courbe HdfsWrite	12
4.3.2	Comparatif Worcount Itératif VS MapRed	12
5	Conclusion	14

Chapitre 1

Corrections et modifications

1.1 Introduction du déploiement centralisé

Dans la version 1 du projet, nous fournissions déjà un utilitaire facilitant le déploiement, l'utilisation et l'évaluation de la plateforme.

Toutefois, nous devions, pour chaque machine du cluster, lancer l'utilitaire sur cette machine.

Nous avons donc amélioré notre solution en ce que maintenant, il suffit de lancer l'utilitaire sur le noeud **master** et automatiquement, à l'aide de connections ssh, l'utilitaire lancera les daemons appropriés sur tous les autres noeuds du cluster.

1.2 Unité de données pour les jeux de tests

Dans la version 1, nous utilisions un fichier de base de taille 1Mo pour générer des jeux de données de grandes tailles. Pour cette version, nous avons pris en compte les nouvelles limitations imposées et utilisons maintenant un fichier de base de 10Ko à la place.

Chapitre 2

Mode d'emploi

2.1 Installation

Sur chaque machine du cluster :

1. Copiez le dossier du projet dans l'emplacement de votre choix.
2. Editez à votre convenance les fichiers de configurations qui se trouvent dans le dossier `config`.
3. Ajoutez les lignes suivantes dans votre profil bash puis recharger le.

```
1 # Adding Hidoop
2 HIDOOP_HOME=/path/to/hidoop
3 export HIDOOP_HOME
4 PATH=$PATH:$HIDOOP_HOME/bin
5 export PATH
```

Listing 2.1 – Variables d'environnement

2.2 Déploiement

Sur le master :

1. Ouvrez un terminal.
2. Tapez la commande `hidoop start`.

2.3 Utilisation

Les interactions avec la plateforme se font par l'intermédiaire de la commande `hidoop` qui peut être lancée depuis n'importe quel répertoire quand l'étape 3 de l'installation s'est bien déroulé.

2.3.1 Lancement & arrêt des démons

- `hidoop start`
Lance les bons démons sur tous les noeuds.
- `hidoop stop`
Stop les démons sur tous les noeuds.
- `hidoop restart`
Arrête et relance les démons sur tous les noeuds.

2.3.2 Interactions Hdfs

- `hidoop write line|kv <input>`
- `hidoop read <hdfs_file> <output_file>`
- `hidoop delete <hdfs_file>`
- `hidoop ls [<hdfs_file>]`

2.3.3 Exécution MapReduce

- `hidoop run <app_name> <hdfs_file>`

2.3.4 Monitoring

- `hidoop report configs`
Affiche juste les configurations en cours
- `hidoop report master`
Affiche juste les infos sur le master
- `hidoop report workers`
Affiche juste les infos sur les workers
- `hidoop report worker <hostname>`
Affiche les infos sur un worker
- `hidoop report`
Affiche un rapport global

2.3.5 Evaluations des performances

- `hidoop bench mb|gb <data_size>`
Evalue les performances du cluster sur un fichier de taille `datasize` méga octets (opt. mb) ou giga octets (opt. gb). Ce fichier est généré automatiquement à l'aide d'un script `data-gen` disponible dans le dossier bin du projet.

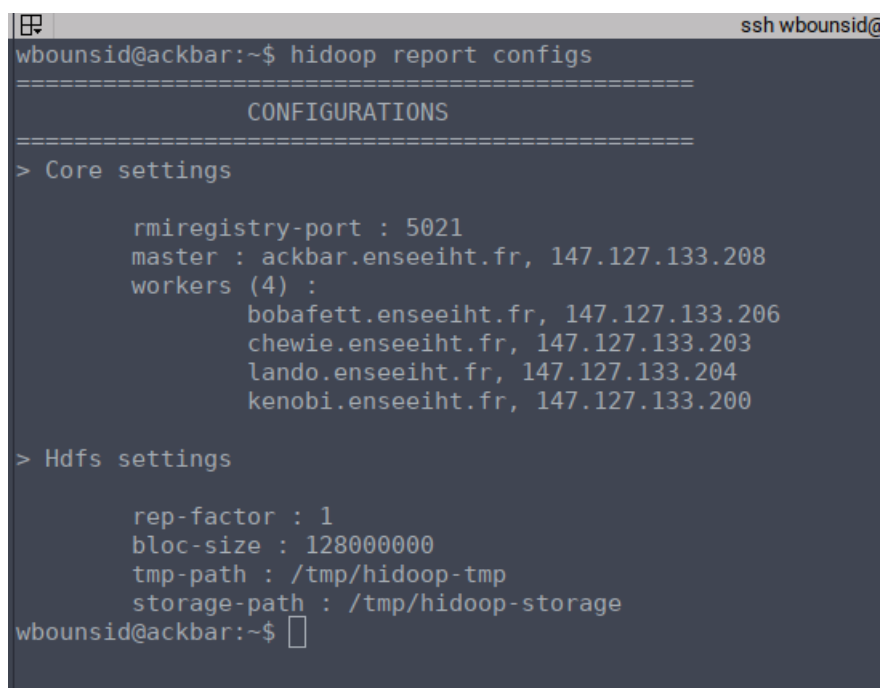
Chapitre 3

Démonstration

3.1 Monitoring

Nous ne présentons ici que quelques captures d'écran illustrant le fonctionnement des commandes shell disponibles.

3.1.1 Affichage des configurations courantes



```
ssh wbounsid@ackbar:~$ hadoop report configs
=====
CONFIGURATIONS
=====
> Core settings

rmiregistry-port : 5021
master : ackbar.enseeiht.fr, 147.127.133.208
workers (4) :
    bobafett.enseeiht.fr, 147.127.133.206
    chewie.enseeiht.fr, 147.127.133.203
    lando.enseeiht.fr, 147.127.133.204
    kenobi.enseeiht.fr, 147.127.133.200

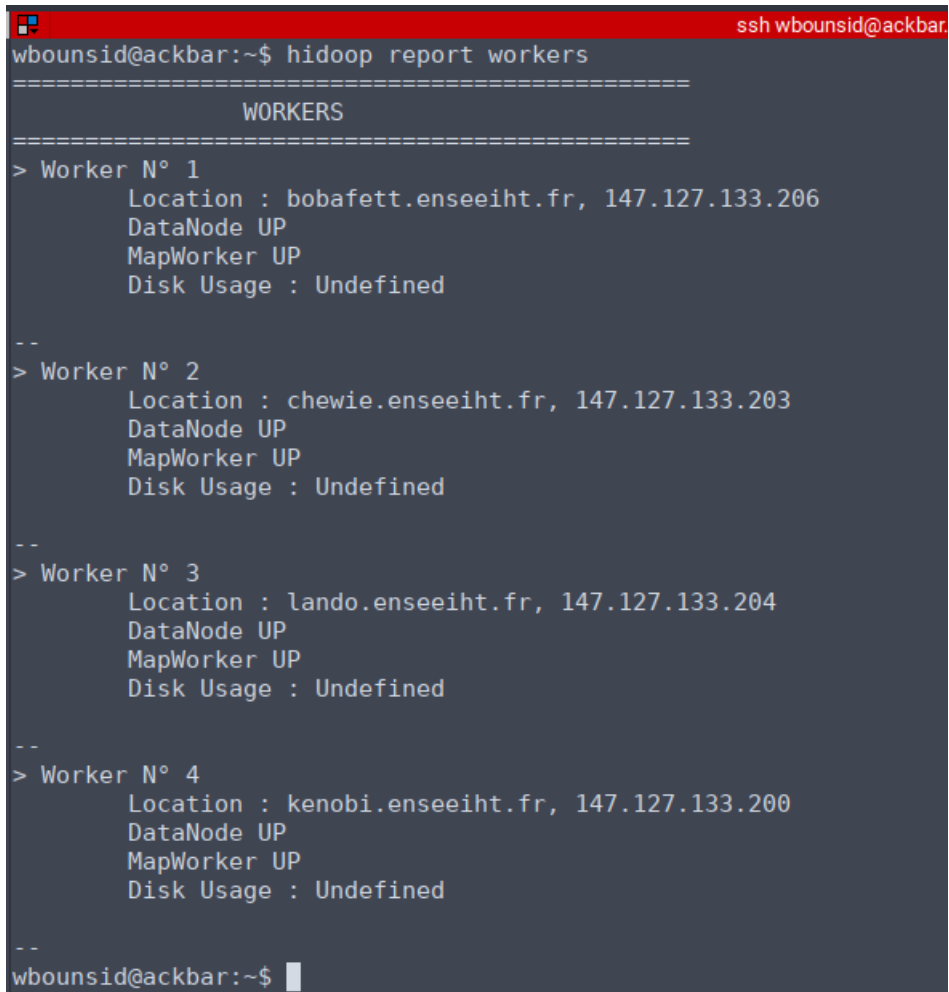
> Hdfs settings

rep-factor : 1
bloc-size : 128000000
tmp-path : /tmp/hadoop-tmp
storage-path : /tmp/hadoop-storage
wbounsid@ackbar:~$
```

FIGURE 3.1 – Affichage des configurations

3.1.2 Affichage de l'état des workers

Tous les workers sont actifs



```
ssh wbounsid@ackbar
wbounsid@ackbar:~$ hadoop report workers
=====
                        WORKERS
=====
> Worker N° 1
  Location : bobafett.enseeiht.fr, 147.127.133.206
  DataNode UP
  MapWorker UP
  Disk Usage : Undefined
--
> Worker N° 2
  Location : chewie.enseeiht.fr, 147.127.133.203
  DataNode UP
  MapWorker UP
  Disk Usage : Undefined
--
> Worker N° 3
  Location : lando.enseeiht.fr, 147.127.133.204
  DataNode UP
  MapWorker UP
  Disk Usage : Undefined
--
> Worker N° 4
  Location : kenobi.enseeiht.fr, 147.127.133.200
  DataNode UP
  MapWorker UP
  Disk Usage : Undefined
--
wbounsid@ackbar:~$
```

FIGURE 3.2 – Workers tous actifs

Un Worker est inactif

```
ssh wbounsid@bobafett.enseeiht.fr
wbounsid@bobafett:~$ hidoop stop
Stopping DataNode and MapWorker...
wbounsid@bobafett:~$
```

FIGURE 3.3 – Arret d'un worker

```
ssh wbounsid@ackbar.enseeiht.fr
wbounsid@ackbar:~$ hidoop report workers
=====
                        WORKERS
=====
> Worker N° 1
    Location : bobafett.enseeiht.fr, 147.127.133.206
    DataNode DOWN
    MapWorker DOWN
    Disk Usage : Undefined
--
> Worker N° 2
    Location : chewie.enseeiht.fr, 147.127.133.203
    DataNode UP
    MapWorker UP
    Disk Usage : Undefined
--
> Worker N° 3
    Location : lando.enseeiht.fr, 147.127.133.204
    DataNode UP
    MapWorker UP
    Disk Usage : Undefined
--
> Worker N° 4
    Location : kenobi.enseeiht.fr, 147.127.133.200
    DataNode UP
    MapWorker UP
    Disk Usage : Undefined
--
```

FIGURE 3.4 – Workers tous actifs sauf un

3.2 Interactions Hdfs

3.2.1 Application de HdfsWrite

```
wbounsid@ackbar:/tmp$ hidoop write line 1mb.txt
1mb.txt
Fragmentation du fichier: 1mb.txt ... OK
  1 fragments créés
Suppression du fragment /tmp/hidoop-tmp/1mb.txt.frag.0 ...OK
wbounsid@ackbar:/tmp$ hidoop ls
1mb.txt => [
            (0, bobafett.enseeiht.fr);
          ]
```

3.2.2 Application de HdfsRead

```
wbounsid@ackbar:/tmp$ hidoop ls
1mb.txt => [
            (0, bobafett.enseeiht.fr);
          ]

wbounsid@ackbar:/tmp$ hidoop read 1mb.txt out.txt
--Reception du fichier 0 ...OK
--Concatenation des fichiers reçu ...
--Fichier out.txt crée
wbounsid@ackbar:/tmp$ ls -lh out.txt
-rw-rw-r-- 1 wbounsid wbounsid 1023K janv. 15 19:32 out.txt
wbounsid@ackbar:/tmp$ diff 1mb.txt out.txt
wbounsid@ackbar:/tmp$ █
```

3.2.3 Application de HdfsDelete

```
wbounsid@ackbar:/tmp$ hidoop ls
1mb.txt => [
            (0, bobafett.enseeiht.fr);
          ]

wbounsid@ackbar:/tmp$ hidoop delete 1mb.txt
wbounsid@ackbar:/tmp$ hidoop ls

wbounsid@ackbar:/tmp$ █
```

3.3 Lancement d'un Job MapReduce

```
ssh wboundsid@ackbar.enseeiht.fr
wboundsid@ackbar:~/nosave/hadoop/bin$ hadoop ls
512mb.txt => [
  (0, bobafett.enseeiht.fr);
  (1, chewie.enseeiht.fr);
  (2, lando.enseeiht.fr);
  (3, kenobi.enseeiht.fr);
  (4, bobafett.enseeiht.fr);
]

wboundsid@ackbar:~/nosave/hadoop/bin$ hadoop run application.MyMapReduce 512m
Job : Lancement des maps...
Job : Lancement d'un map sur le noeud bobafett.enseeiht.fr
Job : Map lancé sur le noeud bobafett.enseeiht.fr
Job : Lancement d'un map sur le noeud chewie.enseeiht.fr
Job : Map lancé sur le noeud chewie.enseeiht.fr
Job : Lancement d'un map sur le noeud lando.enseeiht.fr
Job : Map lancé sur le noeud lando.enseeiht.fr
Job : Lancement d'un map sur le noeud kenobi.enseeiht.fr
Job : Map lancé sur le noeud kenobi.enseeiht.fr
Job : Lancement d'un map sur le noeud bobafett.enseeiht.fr
Job : Map lancé sur le noeud bobafett.enseeiht.fr
Job : Tous les maps sont lancés
Job : un map terminé 4 restant(s)...
Job : un map terminé 3 restant(s)...
Job : un map terminé 2 restant(s)...
Job : un map terminé 1 restant(s)...
Job : un map terminé 0 restant(s)...
Job : Tous les maps sont terminés
Job : Fusion des resultats des maps...
--Reception du fichier 0 ...0K
--Reception du fichier 1 ...0K
--Reception du fichier 2 ...0K
--Reception du fichier 3 ...0K
--Reception du fichier 4 ...0K
--Concatenation des fichiers reçu ...
--Fichier 512mb.txt-map créé
Job : Succes
Job : Lancement du reduce...
Job : Succes
Job : Terminé, fichier output -> 512mb.txt-reduce
time in ms =4225
wboundsid@ackbar:~/nosave/hadoop/bin$ head -n 10 512mb.txt-reduce
Conflict.--As<->1024
Reserve<->512
happiness."<->1024
confirmed--Napoleon<->1024
costing<->512
```

FIGURE 3.5 – Exécution job mapreduce

3.4 Évaluation des performances

```
wbounsid@ackbar:~/nosave/hadoop/bin$  
wbounsid@ackbar:~/nosave/hadoop/bin$ hidoop bench mb 512  
Taille fichier de test généré => 511,482MiB  
Temps d'exécution hdfs write : 14427ms  
Temps d'exécution wordcount itératif : 9128ms  
Temps d'exécution wordcount mapred : 4034ms  
Différences entre les sorties : 0  
wbounsid@ackbar:~/nosave/hadoop/bin$
```

FIGURE 3.6 – Démo evaluation des performances

NB La dernière ligne affichée ici signifie qu'il n'existe pas de différences entre les lignes des deux fichiers résultats, l'un issue de l'exécution du wordcount itératif et l'autre de l'exécution du wordcount MapReduce

Chapitre 4

Etude de la scalabilité

Pour les configurations suivantes :

- Taille bloc : 128Mo
- Facteur de réplication : 1
- 4 Workers¹

4.1 Procédure

Nous écrivons le script suivant

```
1 touch evals
2 hidoop bench mb 64 &>> evals
3 hidoop bench mb 128 &>> evals
4 hidoop bench mb 256 &>> evals
5 hidoop bench mb 512 &>> evals
6 hidoop bench gb 1 &>> evals
7 hidoop bench gb 2 &>> evals
8 hidoop bench gb 5 &>> evals
9 hidoop bench gb 10 &>> evals
```

Listing 4.1 – Script d'évaluation

4.2 Résultats

L'exécution du script précédent nous génère un fichier `eval_perfs`² contenant les résultats que nous synthétisons dans le tableau suivant.

Taille Fichier	Temps HdfsWrite (ms)	Temps WordCount Itératif (ms)	Temps WordCount MapRed (ms)
64M	2 309	1 395	2 073
128M	4 564	2 909	3 254
256M	8 432	5 323	3 511
512M	16 679	9 920	3 758
1G	31 954	18 130	4 753
2G	63 674	37 661	6 050
5G	162 346	93 194	13 947
10G	281 866	185 142	36 871

4.3 Interprétation

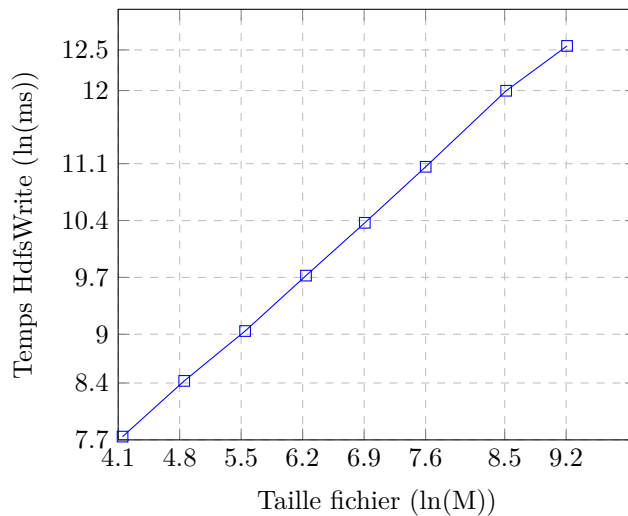
On peut étudier ces résultats sous forme graphique avec une double échelle logarithmique pour faciliter leurs représentation.

1. Caractéristiques : 16Gb RAM, Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz ×4, Il en va de même pour le noeud Master

2. Ce fichier se trouve dans le dossier doc du projet

4.3.1 Courbe HdfsWrite

Comparaison du temps HdfsWrite en fonction de la taille des fichiers



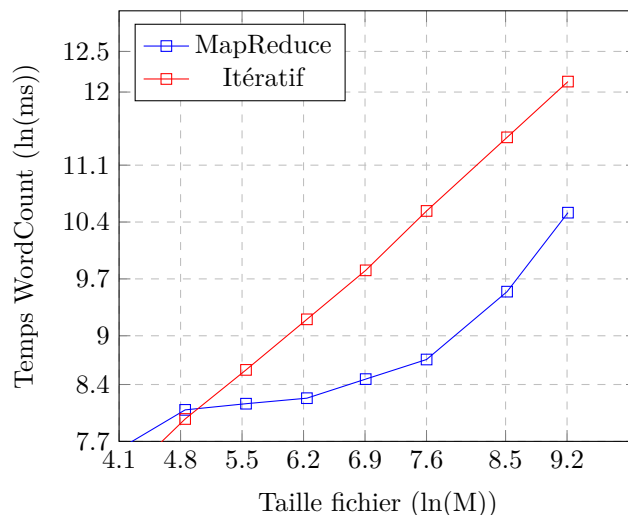
Comme on le voit, le temps du HdfsWrite évolue linéairement par rapport à la taille du fichier. On se retrouve donc, dans les cas où les fichiers sont volumineux (supérieur à 100 Méga et beaucoup plus), avec HdfsWrite qui prends le plus de temps. Le WordCount ne représentant alors qu'une petite partie du temps de calcul total. Il est en revanche bien attendu que le temps du HdfsWrite évolue linéairement par rapport à la taille du fichier car les envois des fragments sont successifs (même avec des threads).

4.3.2 Comparatif Wordcount Itératif VS MapRed

On compare maintenant le temps du WordCount en itératif et avec MapReduce. Le temps WordCount MapReduce prend en compte seulement le temps d'exécution des procédures Maps (le plus long), de la récupération des données avec HdfsRead et de la procédure Reduce.

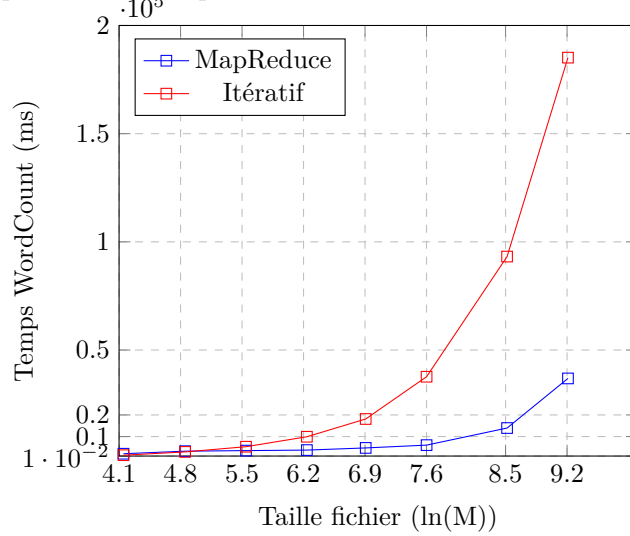
Double échelle logarithmique

Comparaison des temps de WordCount en fonction de la taille des fichiers



Echelle logarithmique sur la taille des fichiers

Comparaison des temps de WordCount en fonction de la taille des fichiers



On remarque que le WordCount en MapReduce est bien plus efficace que celui en Itératif. Cela est dû en partie au fait que le WordCount est une procédure qui est facilement partageable et que la charge du Reducer n'est pas trop importante. Par une procédure d'optimisation classique (Gradient Conjugué Tronqué) de fonction est bien plus difficilement au MapReduce de part sont coté "itératif obligatoire". Il faut donc être critique et ne pas tout transformer en procédure MapReduce..

Chapitre 5

Conclusion

L'étude de la scalabilité réalisée s'est faite en faisant varier la taille du fichier d'entrée, à taille de bloc fixé et à nombre de noeuds fixé.

Nous comptons également faire deux études supplémentaires

- Faire varier la taille des blocs, à taille de fichier d'entrée fixée et à nombre de noeuds fixée.
- Faire varier le nombre de noeuds à taille de fichier d'entrée fixée et à taille de blocs fixée.

Ces dernières études n'ont pu être menées en raison de difficultés rencontrées par la personne en charge. Nous comptons donc l'assister et fournir dans les plus brefs délais les résultats de l'étude.

Nous concédons aussi accuser un certain retard au regard de la répartition des tâches initiales. Ceci est principalement dû à notre surcharge sur le plan académique. Toutefois nous comptons réajuster notre planification et rattraper notre retard.