

Rapport de Travail d'Etude et de Recherche

Utilisation de Spark pour Map-Reduce sur des objets combinatoires

Thomas Foltête Adrien Pavão

M1 Informatique
Université de Paris-Sud

Avril 2017

Sommaire

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

1 Introduction

2 Notions préliminaires

- Recursively Enumerated Set
- Monoïdes numériques
- Map-Reduce

3 Spark

- Premiers pas avec Spark
- Resilient Distributed Datasets
- Implémentations utilisant des RDDs

4 Résultats et optimisations

- Résultats des approches testées
- Optimisations des performances

5 Conclusion

Introduction

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Framework Spark :

Haut niveau.

Jusqu'à 100 (10) fois plus rapide qu'Hadoop en mémoire (sur disque).

Utilisation classique sur des données stockées.

Combinatoire :

Étudie les collections d'objets, leurs combinaisons et leur dénombrement.

L'informatique est adapté à ce genre de calcul.

Notions préliminaires

Parallélisation

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Distribution du calcul pour qu'il soit exécuté simultanément (en parallèle) par plusieurs agents.

- Multi threads (mono machine)
- Multi processeurs (mono machine)
- Multi machines

Notions préliminaires

Recursively Enumerated Set

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

**Recursively
Enumerated
Set**

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Ensemble récursivement énumérable :

- Une racine (élément initial).
- Une fonction successeur.

Ensemble infini si pas de condition (donc semi-décidable).

Topologie possible en arbre.

Parcours distribué possible.

Notions préliminaires

Monoïdes numériques

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

**Monoïdes
numériques**
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Numerical semi-group :

- Sous-ensemble de \mathbb{N} contenant 0
- Stable par addition
- Complémentaire fini

Propriétés d'un semi-group :

- Multiplicité
- Genus
- Frobenius
- Conducteur

Notions préliminaires

Exemple d'un monoïde numérique

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

**Monoïdes
numériques**
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

$$S = \{0, 3, 6, 7, 9, 10\} \cup [12, +\infty[= \langle 3, 7 \rangle$$

Le sous-ensemble $\langle 3, 7 \rangle$ est l'ensemble générateur.

Son ensemble complémentaire est $\{1, 2, 4, 5, 8, 11\}$

Notions préliminaires

Map-Reduce

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Très utilisée en calcul distribué
Se base sur deux fonctions :

- **Map** : Application d'une fonction à tous les éléments d'une collection.
- **Reduce** : Réduction de tous les éléments à l'aide d'un opérateur associatif.

Notions préliminaires

Map-Reduce

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques

Map-Reduce

Spark

Premiers pas
avec Spark

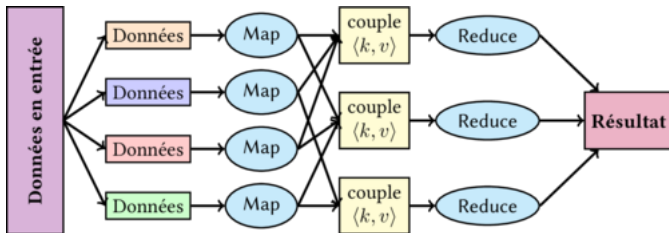
Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Figure – Schéma explicatif du MapReduce



Premiers pas avec Spark

Évaluation et récursivité

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

**Premiers pas
avec Spark**

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Avantage de Spark :

- **Évaluation paresseuse** : L'évaluation réelle n'est faite que si elle est nécessaire. Résultats non préservés par défaut.
- **Aplatissement en mémoire** : Fixe les données en mémoire en les évaluant.

Spark se base sur les RDDs. Ils sont tolérants à la panne et ont une structure parallèle. Ils permettent à l'utilisateur de :

- Garder les données intermédiaires explicitement en mémoire
- Contrôler leur partitionnement afin d'optimiser l'emplacement des données
- Manipuler les données en utilisant un ensemble important d'opérateurs

Création d'un RDD à partir d'une liste Python :

```
rdd = sc.parallelize([1,2,3,4,5])
```

Il y a deux différents types de méthodes qu'on peut appliquer aux RDDs :

- Les **transformations** : Union, Map, etc.
- Les **actions** : First, Reduce, etc.

Spark

Différentes implémentations utilisant des RDDs

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

**Implémentations
utilisant des
RDDs**

Résultats et
optimisations

Résultats des
approches

Ensembles des mots d'un alphabet : Produit cartésien d'un alphabet A avec lui-même. Nous nous baserons sur cet exemple pour la suite. Implémentés en Python plutôt qu'en Sage.

Ensemble des arbres binaires à n nœuds :

Cet ensemble se modélise ainsi :

$$\begin{cases} T_0 = Leaf \\ T_n = \cup_i T_i \times Node \times T_{n-i-1} \end{cases} \quad (1)$$

On peut ensuite par exemple compter les arbres ou compter les arbres par profondeur. Ce modèle peut également permettre de manipuler des polynômes.

Spark

Différentes approches possibles

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

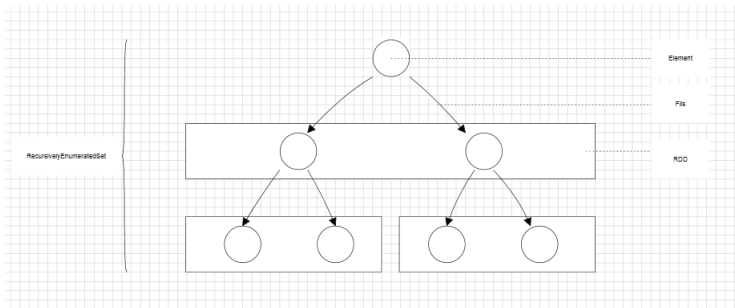
**Implémentation
utilisant des
RDDs**

Résultats et
optimisations

Résultats des
approches

Approche A

Figure – Schéma explicatif de l'approche A



Spark

Différentes approches possibles

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

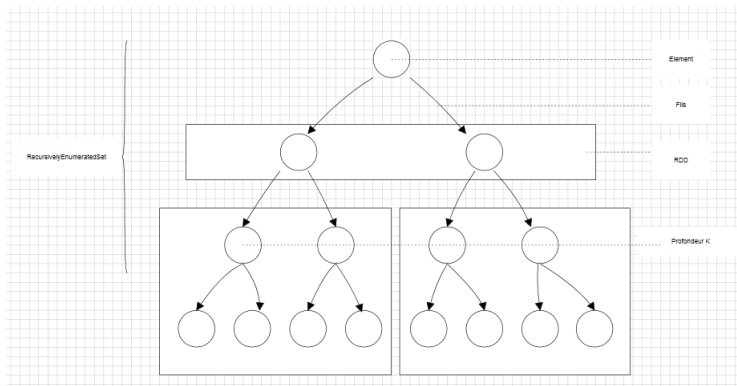
**Implémentation
utilisant des
RDDs**

Résultats et
optimisations

Résultats des
approches

Approche B

Figure – Schéma explicatif de l'approche B



Spark

Différentes approches possibles

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

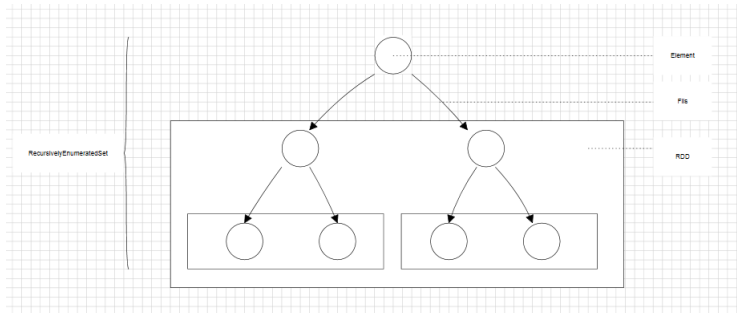
**Implémentation
utilisant des
RDDs**

Résultats et
optimisations

Résultats des
approches

Approche C

Figure – Schéma explicatif de l'approche C



Spark

Différentes approches possibles

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

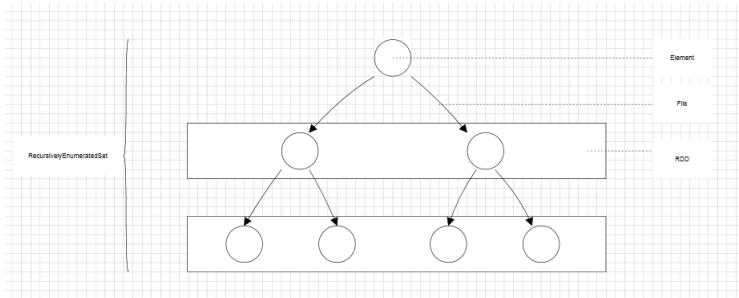
Implémentation
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Approche D

Figure – Schéma explicatif de l'approche D



Résultats et optimisations

Critères d'évaluation

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

On observe :

- Le temps de calcul : On souhaite le minimiser
- L'utilisation mémoire : Savoir si tout est développé en mémoire
- La distribution du calcul et le partitionnement des RDDs

Résultats et optimisations

Protocole de test

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

- Dénombrement des mots de taille n sur l'alphabet $A = \{a, b, c, d\}$
- Machine : 2 processeurs à 2.40GHz et 8 Go de RAM
- Count avec un Map-Reduce
- Mesure du temps dans le code Python avec *time.time()*
- Observation des processus et de la mémoire avec *top*
- Nombre de partitions du RDD en temps réel avec la méthode *getNumPartitions()*

Résultats et optimisations

Résultats des approches testées

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

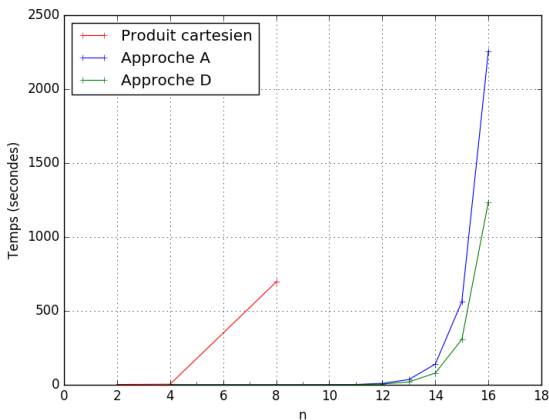
Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Figure – Temps d'exécution en fonction de n



Résultats et optimisations

Analyse des premiers résultats

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Conjecture pour ces résultats décevants :

- Mauvaise parallélisation due à notre implémentation Python, ou aux Strings.
- Certains RDDs sont réévalués plusieurs fois.
- Création récursive de petits RDDs qui prend beaucoup de temps.

Résultats et optimisations

Analyse des premiers résultats

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Idées de solution :

- Faire du profiling.
- Aplatir le RDD en mémoire à un certain niveau.
L'aplatissement semble permettre un grand grain de performance.
- Faire un algorithme hybride.

Optimisations de performances

Aplatissement naïf

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

On essaye de collecter le RDD dans sa totalité.
Saturation de la mémoire pour $n > 12$.

$n = 12$

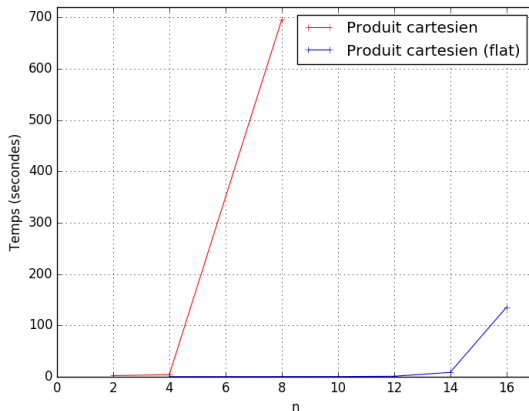
count = 16777216

Temps d'exécution : 4.88712310791 secondes.

Optimisations des performances

Produit cartésien avec aplatissement mémoire

Figure – Schéma comparatif du produit cartésien avec son homologue avec aplatissement mémoire



Optimisations des performances

Produit cartésien avec aplatissement mémoire

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Commande top :

PID	USER	[..]	CPU	MEM	TIME+	COMMAND
29700	ubuntu	[..]	99.7	0.3	1:07.20	python
29701	ubuntu	[..]	99.7	0.3	1:06.79	python
29705	ubuntu	[..]	99.7	0.3	1:06.62	python
29709	ubuntu	[..]	99.7	0.3	1:06.78	python

Il s'agit pour le moment du meilleur temps que nous ayons obtenu :

$n = 16$

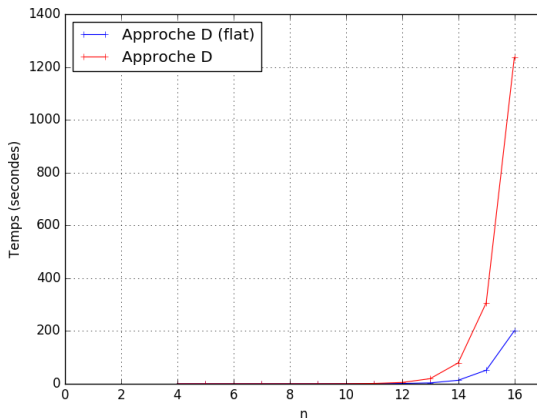
count = 4294967296

Temps d'exécution: 129.296506166 secondes.

Optimisations des performances

Approche D améliorée

Figure – Schéma comparatif de l'approche D et de sa version améliorée avec aplatissement en briques de taille 1024



Optimisations de performances

Applatissage dynamique

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

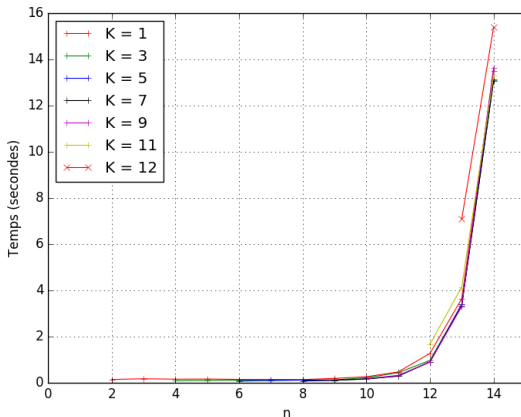
Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Figure – Temps d'exécution en fonction de n pour différents K



Résultats et optimisations

Vol de tâches

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Un arbre déséquilibré en largeur :

```
rdd = sc.parallelize(range(1,5))  
  
for i in range(n):  
    rdd = rdd.flatMap( \\  
        lambda x : range(1,4*x+1))
```

Les processus s'arrêtent chacun leur tour.

Résultats et optimisations

Utilisation mémoire

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

Exemple d'utilisation mémoire pendant un calcul avec Spark :

PID	USER	[...]	CPU	MEM	TIME+	COMMAND
8214	ubuntu	[...]	71.1	0.3	0:02.14	python
18122	ubuntu	[...]	43.2	3.9	0:12.74	java
18207	ubuntu	[...]	12.6	0.3	0:00.38	python
18210	ubuntu	[...]	4.3	0.3	0:00.13	python
18211	ubuntu	[...]	4.3	0.3	0:00.13	python
18217	ubuntu	[...]	3.3	0.3	0:00.10	python

Conclusion

Rapport de
Travail
d'Etude et
de Recherche

Thomas
Foltête,
Adrien Pavão

Introduction

Notions
préliminaires

Recursively
Enumerated
Set

Monoïdes
numériques
Map-Reduce

Spark

Premiers pas
avec Spark

Resilient
Distributed
Datasets

Implémentations
utilisant des
RDDs

Résultats et
optimisations

Résultats des
approches

- Possibilité de calcul combinatoire
- Un peu lent mais plus rapide en Scala
- Peut-être pas de vol de tâches prévu par Spark
- Les structures de données influent grandement les performances et la distribution du calcul

Notre étude de Spark ne répond pas forcément à toutes les questions qui l'entourent, cependant ce projet nous aura été très pédagogique.