

Juvénal Chokogoue

Maîtrisez l'utilisation des technologies

HADOOP

Initiation à l'écosystème Hadoop



EYROLLES

Le manuel d'apprentissage de référence

Cet ouvrage est un manuel d'apprentissage technique qui a été rédigé pour toute personne souhaitant développer des compétences sur une ou plusieurs technologie(s) de l'écosystème Hadoop. Il permet d'utiliser de façon professionnelle 18 technologies clés de l'écosystème Hadoop : Spark, Hive, Pig, Impala, ElasticSearch, HBase, Lucene, HAWQ, MapReduce, Mahout, HAMA, Tez, Phoenix, YARN, ZooKeeper, Storm, Oozie et Sqoop.

L'ouvrage permet d'initier les débutants pour les emmener vers une utilisation professionnelle de ces technologies. Pour faciliter la compréhension de l'ouvrage, chaque chapitre s'achève par un rappel des points clés et un guide d'étude qui permettent au lecteur de consolider ses acquis. Des compléments web sont également disponibles en téléchargement sur le site www.editions-eyrolles.com/dl/0067478.

Au fil de la lecture de cet ouvrage, vous allez comprendre les approches conceptuelles de chacune de ces technologies pour rendre vos compétences indépendantes de l'évolution d'Hadoop. Vous serez finalement capable d'identifier les portées fonctionnelle, stratégique et managériale de chacune de ces technologies.

À qui cet ouvrage s'adresse-t-il ?

- Aux consultants BI/big data, data scientists, chargés d'études et chefs de projets data
- Aux étudiants désireux de s'orienter vers le big data
- Plus généralement, à tout professionnel souhaitant prendre le virage du big data ou souhaitant valoriser les données de son entreprise

Au sommaire

Les modèles de calcul de l'écosystème Hadoop. Les modèles de calcul batch • Les modèles de calcul interactifs • **Les abstractions des modèles de calcul d'Hadoop.** Les langages d'abstraction d'Hadoop • Le SQL sur Hadoop • **Le stockage de données en Hadoop.** Généralités sur le stockage des données • HBase • L'indexation de contenu • Apache Lucene • ElasticSearch • **La gestion du cluster Hadoop.** YARN • Apache ZooKeeper • **Le streaming en temps réel dans Hadoop.** Apache Storm • Les outils annexes de l'écosystème Hadoop. Oozie et Sqoop • Hue et Ambari • **Adoption à grande échelle d'Hadoop.** Distributions d'Hadoop • Solutions Hadoop embarquées • Hadoop dans le Cloud • Le big data

Consultant, auteur, expert et conférencier, **Juvéнал Chokogue** est spécialisé depuis 2011 dans les problématiques de valorisation à large échelle des données. Il possède une forte expertise, tant sur l'ingénierie que sur l'analyse des données. Il est titulaire de 7 certifications, dont 4 en data analytics, 1 en big data et 1 en business intelligence. Juvéнал est également certifié *John Maxwell Leadership*. Il travaille actuellement comme consultant en big data et data analytics. Vous pouvez le contacter sur le site web dédié à l'ouvrage : <http://www.data-transitionnumerique.com>.

Maîtriser l'utilisation des technologies Hadoop

SUR LE MÊME THÈME

- R. BRUCHEZ. – **Les bases de données NoSQL et le Big Data.**
N° 14155, 2^e édition, 2015, 320 pages.
- J. GRUS. – **Data science par la pratique.**
N° 11868, 2017, 308 pages.
- E. BIERNAT, M. LUTZ. – **Data science : fondamentaux et études de cas.**
N° 14243, 2015, 312 pages.
- W. MCKINNEY. – **Analyse de données en Python.**
N° 14109, 2015, 488 pages.
- M.-R. AMINI, E. GAUSSIER. – **Recherche d'information.**
N° 67376, 2^e édition, 2017, 294 pages.

DANS LA MÊME COLLECTION

- B. BARRÉ. – **Concevez des applications mobiles avec React Native.**
N° 67563, 2018, 224 pages.
- R. RIMELÉ. – **HTML 5 – Une référence pour le développeur web.**
N° 67401, 3^e édition, 2017, 832 pages.
- P. FICHEUX. – **Linux embarqué – Mise en place et développement.**
N° 67484, 2017, 220 pages.
- M. BIDAULT. – **Programmation Excel avec VBA.**
N° 67401, 2017, 480 pages.
- K. NOVAK. – **Débuter avec LINUX.**
N° 13793, 2017, 522 pages.
- P. MARTIN, J. PAULI, C. PIERRE DE GEYER. – **PHP 7 avancé.**
N° 14357, 2016, 732 pages.
- R. GOETTER. – **CSS 3 Flexbox.**
N° 14363, 2016, 152 pages.

Retrouvez nos bundles (livres papier + e-book) et livres numériques sur
<http://izibook.eyrolles.com>

Juvénal Chokogoue

Maîtriser l'utilisation des technologies Hadoop

Initiation à l'écosystème Hadoop

EYROLLES

The logo for EYROLLES features the word "EYROLLES" in a bold, sans-serif font. Below the text is a horizontal line with a small circle centered underneath it.

ÉDITIONS EYROLLES
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

En application de la loi du 11 mars 1957, il est interdit de reproduire intégralement ou partiellement le présent ouvrage, sur quelque support que ce soit, sans l'autorisation de l'Éditeur ou du Centre Français d'exploitation du droit de copie, 20, rue des Grands Augustins, 75006 Paris.
© Groupe Eyrolles, 2018, ISBN: 978-2-212-67478-1

Table des matières

Avant-propos	1
Pourquoi cet ouvrage ?	1
Objectifs de cet ouvrage	2
Objectif 1. Monter en compétences sur les principales technologies de l'écosystème Hadoop	2
Objectif 2. Comprendre les concepts qui sous-tendent les technologies de l'écosystème Hadoop	3
Objectif 3. Savoir identifier la portée fonctionnelle et stratégique des technologies de l'écosystème Hadoop.	3
À qui cet ouvrage s'adresse-t-il ?	4
Comment lire cet ouvrage ?	5
Les compléments web	5
Notes de l'auteur	7
Remerciements	9
Introduction à l'écosystème Hadoop	11
Hadoop	12
L'approche conceptuelle d'Hadoop	13
Fonctionnement d'Hadoop	15
Taxonomie de l'écosystème Hadoop	19
En résumé	23
Présentation des guides d'étude	23
Guide d'étude	24
Principes clés	26

PARTIE 1

Les modèles de calcul de l'écosystème Hadoop..... 29

CHAPITRE 1

Les modèles de calcul batch 33**Principes du traitement parallèle en batch** 34

Le traitement sur disque ou batch processing..... 34

L'exécution parallèle en batch..... 36

Le MapReduce 38

Définition du MapReduce..... 38

Détails de l'exécution du MapReduce dans un cluster Hadoop..... 39

Jointure de deux tables relationnelles..... 41

Mahout et Hama 43

Mahout..... 43

Hama..... 46

En résumé 46**Guide d'étude** 47**Principes clés** 52

CHAPITRE 2

Les modèles de calcul interactifs 53**Principes du traitement parallèle en in-memory** 54

Définition de l'in-memory..... 54

Calcul in-memory parallèle..... 55

Spark : le moteur in-memory distribué d'Hadoop..... 57

Définition de Spark..... 57

Fonctionnement général de Spark..... 58

Fonctionnement de Spark sur Hadoop..... 61

Tutoriel Spark avec Scala..... 62

Tez : le moteur d'optimisation du MapReduce 65**En résumé** 67**Guide d'étude** 69**Principes clés** 73

PARTIE 2

Les abstractions des modèles de calcul d'Hadoop 75

CHAPITRE 3

Les langages d'abstraction d'Hadoop 79**Généralités sur les langages d'abstraction 80****Hive 81**

Infrastructure technique de Hive 81

Écriture des requêtes HiveQL 83

Pig 87

Fonctionnement de Pig 88

Programmation en Pig Latin 89

Différence entre Hive et Pig 90**En résumé 91****Guide d'étude 92****Principes clés 96**

CHAPITRE 4

Le SQL sur Hadoop 97**Les moteurs SQL MPP 98**

Fonctionnement des bases de données parallèles 98

Exécution des requêtes SQL dans les bases parallèles 102

Les moteurs natifs SQL sur Hadoop 106

Fonctionnement des moteurs natifs SQL sur Hadoop 106

Impala : le moteur SQL de Cloudera 108

HAWQ : le moteur SQL de Pivotal 112

Phoenix : le moteur SQL de Salesforce.com 115

En résumé 116**Guide d'étude 117****Principes clés 123**

PARTIE 3

Le stockage de données en Hadoop 125

CHAPITRE 5

Généralités sur le stockage des données 129**La donnée** 130

Le format de codage des données 130

Le format de stockage de données 132

Les formats de fichiers non structurés 142

Le type de données 142

De la donnée à la base de données 145

Problèmes soulevés par la collecte des données 145

Les bases de données 148

De la base de données aux SGBD 151

De la base de données au SGBDR 151

Du SGBD SQL au SGBD NoSQL 154

Panorama des SGBD NoSQL 157

En résumé 160**Guide d'étude** 161**Principes clés** 167

CHAPITRE 6

HBase 169**Concepts de base de HBase** 170

Définition de HBase 170

La base de données en HBase 171

Architecture et fonctionnement de HBase 181**Utilisation de HBase** 184

Règles de modélisation d'une table HBase 184

Exploitation d'une table HBase 186

Mapping d'une base de données relationnelle aux tables HBase 189

Phoenix et HBase 190

En résumé 191

Guide d'étude	192
Principes clés	195
CHAPITRE 7	
L'indexation de contenu	197
Principes de l'indexation de contenu	198
Les bases de l'indexation de contenu	199
Techniques de construction du dictionnaire de l'index	201
Utilisation de l'index lors des requêtes	203
Les approches informatiques de l'indexation de contenu	203
La construction de l'index dans le système informatique	204
Les composants d'un système informatique d'indexation de contenu	208
Les SGBD NoSQL d'indexation de contenu	209
Définition des moteurs NoSQL d'indexation de contenu	210
Fonctionnalités offertes par les moteurs NoSQL d'indexation de contenu	213
En résumé	216
Guide d'étude	217
Principes clés	219
CHAPITRE 8	
Apache Lucene	221
Développement d'une application de recherche de contenu	222
Architecture d'une application de recherche de contenu	222
Étapes de développement d'une application de recherche de contenu	224
Le moteur Apache Lucene	229
Définition d'Apache Lucene	229
Développement de modules de recherche et d'indexation de contenu avec Lucene	231
En résumé	236
Guide d'étude	237
Principes clés	240

CHAPITRE 9

ElasticSearch	243
Spécificités d'ElasticSearch	244
Indexation et recherche de contenu avec ElasticSearch	245
Concepts clés.	245
Indexation de contenu dans ElasticSearch	246
Architecture d'ElasticSearch.	249
Recherche de contenu dans ElasticSearch	252
Exploitation d'ElasticSearch	253
Fondamentaux du REST.	253
Utilisation d'ElasticSearch	260
En résumé	266
Guide d'étude	268
Principes clés	273

PARTIE 4

La gestion du cluster Hadoop	275
---	-----

CHAPITRE 10

YARN	279
Limites d'Hadoop	280
Le modèle de calcul du Hadoop	280
Le HDFS	280
Haute disponibilité du cluster	281
La sécurité du cluster	281
YARN et les développements en cours sur Hadoop	282
Définition de YARN	283
Fonctionnement de YARN	284
En résumé	287
Guide d'étude	288
Principes clés	291

CHAPITRE 11

Apache ZooKeeper	293
Problèmes du développement d'applications en environnement distribué ..	294
Le problème de panne partielle	294
Le problème de coordination	295
ZooKeeper	295
Principes	295
Espace de noms	297
Fonctionnement	298
Méthodes exposées	299
En résumé	300
Guide d'étude	301
Principes clés	303

PARTIE 5

Le streaming en temps réel dans Hadoop	305
---	-----

CHAPITRE 12

Apache Storm	309
Définition et principes	309
Fonctionnement	311
Les topologies	312
Utilisation de Storm	315
Le SQL	315
Flux	316
Trident	316
En résumé	317
Guide d'étude	318
Principes clés	320

PARTIE 6

Les outils annexes de l'écosystème Hadoop..... 321

CHAPITRE 13

Oozie et Sqoop 325**La planification des jobs Hadoop : Oozie** 326

Définition 326

Fonctionnement..... 326

Utilisation 328

La planification de l'ingestion des données dans Hadoop : Sqoop 331

Définition 331

Fonctionnement..... 331

Utilisation 335

En résumé 339**Guide d'étude** 340**Principes clés** 344

CHAPITRE 14

Hue et Ambari..... 345**Hue : l'interface graphique d'Hadoop** 345**Ambari : l'interface d'administration du cluster** 347**En résumé** 348**Guide d'étude** 349**Principes clés** 349

PARTIE 7

Adoption à grande échelle d'Hadoop..... 351

CHAPITRE 15

Distributions d'Hadoop..... 355**La distribution Cloudera** 356**La distribution Hortonworks** 357

La distribution MapR	357
Offres comparées des trois distributions	360
Guide de sélection de sa distribution Hadoop	361
En résumé	365
Guide d'étude	366
Principes clés	368
CHAPITRE 16	
Solutions Hadoop embarquées	371
Les stratégies d'intégration logicielles	371
L'interfaçage	372
L'intégration	372
IBM InfoSphere BigInsights	373
Pivotal Big Data Suite	374
SAS High Performance Analytics	375
En résumé	377
Guide d'étude	378
Principes clés	380
CHAPITRE 17	
Hadoop dans le Cloud	381
Le Cloud computing et Hadoop	381
Contexte d'émergence du Cloud computing	382
Définition du Cloud computing	383
Taxonomie des services Cloud	384
Hadoop en Cloud	385
Amazon Elastic MapReduce (EMR)	388
Fonctionnalités	388
Tarification	390
Microsoft Azure HDInsight	392
Fonctionnalités	392
Tarification	393

Guide d'évaluation d'une offre Hadoop dans le Cloud	396
Critère 1. Viabilité à long terme du fournisseur	397
Critère 2. Composants Hadoop disponibles	397
Critère 3. Performance	397
Critère 4. Propriété des données	398
Critère 5. Sécurité des données	398
Critère 6. Niveaux de services (SLA)	399
Critère 7. Tarification	399
En résumé	400
Guide d'étude	401
Principes clés	405
 CHAPITRE 18	
Le big data	407
Contexte général	407
Big data, comment en est-on arrivé là ?	408
Qu'est-ce que le big data ?	409
Les facteurs du big data	410
L'évolution culturelle	410
Les facteurs économiques	411
L'utilisation croissante des smartphones et l'ubiquité d'Internet	412
Le Cloud computing et la virtualisation	414
L'open source	414
En résumé	416
Guide d'étude	417
Principes clés	419
Conclusion générale de l'ouvrage	421
Votre avis compte !	423
Liens et références utiles	424
Liens web	424
Ouvrages de référence	428
Index	429

Avant-propos

Pourquoi cet ouvrage ?

Lors de l'une de nos missions, nous avons rencontré Joël Belafa, architecte big data au sein de l'entreprise Dataiku ; la rédaction de cet ouvrage tire son origine d'une anecdote toute simple qu'il nous a alors racontée.

Alors qu'il était en entretien avec un candidat pour un poste dans l'entreprise, il posa la question suivante : « *Avez-vous des compétences en Apache Dauphin ?* ». Le candidat lui répondit : « *Oui, j'ai fait un TD de 6 h dessus lors de ma formation* ». Joël n'a pas retenu le candidat... Par cette question, il voulait juger de son degré de veille concernant l'évolution des technologies Hadoop ; mais l'écosystème Hadoop, pourtant riche en technologies baptisées de noms d'animaux, n'en a aucune à ce jour appelée Apache Dauphin.

Cette petite anecdote, qui parlera sûrement à certains d'entre vous, met en évidence un fait : l'explosion de données qui caractérise le monde dans lequel nous vivons actuellement a entraîné un foisonnement de problématiques qui nécessitent des réponses technologiques aussi différentes les unes que les autres. Hadoop est le socle technologique du stockage et du traitement de ces données, mais il n'est pas capable à lui seul de répondre à toutes ces problématiques. C'est pourquoi un ensemble de technologies regroupées sous le nom d'écosystème Hadoop a été développé ; il comprend, entre autres, SPARK, Tez, YARN, Lucene, HBase, Accumulo, Kafka ou encore Storm. La fondation Apache en est aujourd'hui le dépositaire.

La particularité d'Hadoop et de son écosystème technologique, c'est qu'ils s'appuient sur des approches conceptuelles et techniques différentes de celles sur lesquelles les technologies traditionnelles sont fondées. Là où le développement d'une base de données passait par un MCD transformé en MPD, le développement d'une « base de données » en HBase repose sur la dénormalisation et le partitionnement d'une table. Or, la plupart de ces approches n'ont jamais été étudiées de façon sérieuse au cours des formations des professionnels de la génération actuelle et sont rarement abordées, même dans des tutoriels. Combinez à cela le rythme d'innovation soutenue d'Hadoop et des technologies du big data en général, et vous vous retrouvez rapidement sur un marché présentant une forte disparité entre le niveau d'innovation technologique et les compétences dans ces domaines.

Enfin, le marché a tendance à classer les technologies d'Hadoop d'un point de vue purement technique, négligeant leur aspect fonctionnel et métier, ce qui n'aide pas toujours à en percevoir le rôle et la valeur véritable.

Objectifs de cet ouvrage

Objectif 1. Monter en compétences sur les principales technologies de l'écosystème Hadoop

D'un point de vue fonctionnel, l'écosystème Hadoop est composé d'une centaine de technologies regroupées en 14 catégories selon leur but :

- langages d'abstraction ;
- SQL sur Hadoop ;
- modèles de calcul ;
- traitement en temps réel ;
- bases de données ;
- outils d'*ingestion streaming* ;
- intégration des données ;
- coordination de *workflow* ;
- coordination de services distribués ;
- administration de *clusters* ;
- fonctionnalités liées à l'interface utilisateur ;
- indexation de contenu ;
- systèmes de fichiers distribués ;
- gestionnaires de ressources.

Les principales technologies auxquelles fait appel chacune de ces catégories sont étudiées indépendamment, en 7 parties distinctes (voir carte heuristique illustrant les technologies de l'écosystème Hadoop, page 22). À la fin de l'ouvrage, vous aurez développé les compétences qui vous permettront de les exploiter et de les utiliser de façon professionnelle. Nous citerons notamment Spark, MapReduce, Pig, Hive, Impala, Phoenix, HBase, YARN, ZooKeeper, Sqoop ou encore Storm.



Attention

Cet ouvrage vous aidera à développer des compétences nécessaires pour utiliser ces technologies, mais il ne fera pas de vous des experts !

Objectif 2. Comprendre les concepts qui sous-tendent les technologies de l'écosystème Hadoop

Il n'y a pas qu'Hadoop qui s'appuie sur une approche de calcul différente des méthodes traditionnelles. Les technologies des 14 catégories énoncées précédemment reposent sur des principes et des approches conceptuelles du calcul massivement parallèle.

Ce livre est le deuxième réalisé dans le cadre du projet Data-Transition Numérique (<http://www.data-transitionnumerique.com/>). Il a pour but d'aider les professionnels de la donnée à développer les compétences nécessaires pour assurer leur transition vers le numérique et à bénéficier des opportunités offertes par le big data.

Dans cet ouvrage en particulier, vous découvrirez les approches et les principes qui structurent les technologies de l'écosystème Hadoop. Par exemple, si nous prenons la catégorie « Indexation de contenu », l'ouvrage ne commence pas directement par vous aider à développer des compétences sur Elasticsearch et Lucene, mais il vous explique d'abord les principes de la recherche de contenu et les approches distribuées de l'indexation de contenu.

L'enjeu est de rendre vos compétences indépendantes des outils que vous utiliserez, car vous l'avez certainement remarqué, l'industrie du logiciel open source – et particulièrement celle d'Hadoop – est sujette à une forte volatilité. Les outils sont amenés à changer, voire disparaître, mais les principes sur lesquels ils s'appuient, eux, destinés à rester. Les nouvelles aptitudes acquises vous permettront de rapidement monter en compétences sur n'importe quelle technologie de la catégorie (autre ou future), indépendamment de ses versions.

Pour nous assurer que cet objectif soit atteint, nous nous sommes appuyés sur deux principes fondamentaux.

- ***Personne ne connaît mieux une technologie que son créateur.*** En rédigeant cet ouvrage, nous avons fait en sorte de nous rapprocher le plus possible de l'esprit ou de la pensée originale des auteurs des différentes technologies concernées. Cela nous a emmenés loin, bien au-delà des travaux de la fondation Apache.
- ***On n'évalue pas une technologie hors du contexte dans lequel elle a été créée.*** Évaluer une technologie hors de son contexte de création, c'est exactement comme évaluer un singe sur sa capacité à nager. En d'autres termes, le contexte d'une technologie est tellement important pour sa compréhension qu'il est nécessaire de le comprendre pour se l'approprier dans sa pleine mesure. Pour rendre vos compétences aussi indépendantes que possible des technologies étudiées, cet ouvrage accorde donc une place prépondérante à l'analyse du contexte de création de chacune.

Objectif 3. Savoir identifier la portée fonctionnelle et stratégique des technologies de l'écosystème Hadoop

Les technologies de l'écosystème Hadoop sont généralement catégorisées et confrontées uniquement sur la base de leurs aspects techniques. On comparera par exemple Spark et le MapReduce en observant leurs vitesses ; et il en sera de même pour Hive, Pig, Impala et HAWQ. Ce procédé tend malheureusement à éluder à la fois la raison pour laquelle la technologie a été développée au départ, et sa valeur métier, c'est-à-dire son rôle concret pour l'entreprise et la façon dont elle

peut être rentabilisée. L'enjeu, c'est que vous puissiez vous détacher de la technique pour porter un regard métier, managérial sur Hadoop. Vous serez alors capable d'exercer des activités de conseil autour des technologies de l'écosystème et de dire avec précision quelle technologie correspond à quel besoin métier, en justifiant vos choix.

Au sortir de cet ouvrage :

- vous maîtriserez les approches et les principes qui sont à la base de l'écosystème technologique d'Hadoop ;
- vous aurez développé des compétences qui vous permettront d'être opérationnel sur l'ensemble des technologies Hadoop ;
- le rythme d'innovation soutenu de l'écosystème Hadoop aura un sens pour vous, et vous serez capable de le suivre ;
- vous serez capable d'exercer des activités de conseil (technique et métier) autour de l'écosystème Hadoop ;
- vous saurez bénéficier des opportunités offertes par le big data.

À qui cet ouvrage s'adresse-t-il ?

Cet ouvrage est un livre d'apprentissage technique. Il a été rédigé à l'intention de toute personne qui souhaite développer des compétences sur une ou plusieurs technologie(s) de l'écosystème Hadoop.

Avec la transition numérique en cours, il n'est pas difficile de prédire qu'Hadoop va progressivement devenir le standard en matière de traitement des données, un peu comme Microsoft Excel entre 1995 et aujourd'hui. Voilà pourquoi cet ouvrage s'adresse à toute personne qui souhaite se positionner dès maintenant face aux nouvelles approches de traitement des données. Il peut s'agir de tout professionnel des données, qu'il occupe la fonction de chargé d'étude, de statisticien, de consultant, de développeur ou même de manager.

Pendant, il est recommandé d'avoir les prérequis suivants :

- maîtrise de SQL, des concepts des bases des données relationnelles et du développement d'applications de base de données ;
- connaissances de base en programmation orientée objet, et bonne connaissance d'au moins un langage de programmation tel que Java, Python, Ruby, R ou autre ;
- connaissances de base s'agissant du Web et des systèmes d'information.

Vous n'aurez alors aucune difficulté à lire et comprendre l'ouvrage. De plus, il a été écrit dans un style initiatique de type « pas à pas », qui permettra d'initier le débutant pour l'amener progressivement vers la maîtrise des technologies exposées. Par ailleurs, même si l'ouvrage est un livre d'initiation, comme son titre l'évoque, il n'en exige pas moins une certaine rigueur dans l'approche. Nous conseillons donc aux lecteurs simplement curieux de s'orienter vers des ouvrages plus généraux, étant donné que la relative nouveauté d'Hadoop et de son écosystème, ainsi que le niveau de profondeur avec lequel nous avons abordé les technologies et leurs approches conceptuelles, impliquent un plan d'apprentissage rigoureux.

Comment lire cet ouvrage ?


Les sept parties de l'ouvrage sont écrites indépendamment les unes des autres. Vous n'avez donc pas spécialement besoin de les lire dans l'ordre. Cependant, veuillez à lire dans l'ordre les chapitres de chaque partie, car ils correspondent à une problématique précise et une catégorie de l'écosystème Hadoop distincte.

Chaque partie contient un ou plusieurs paragraphe(s) introductif(s) spécifiant le but de la catégorie de technologie concernée et son contexte. Chaque chapitre possède quant à lui un guide d'étude que nous vous recommandons de suivre.

Pour lire ce livre de façon efficace, vous pourrez adopter l'une des deux stratégies suivantes.

- Dans chaque partie, vous étudierez chaque chapitre indépendamment et de façon rigoureuse et, à la fin de chacun, vous vérifierez vos acquis grâce à au guide d'étude correspondant, en validant à l'aide du guide des réponses.
- Vous ferez une première lecture rapide de tous les chapitres de la partie. Ensuite, dans une seconde lecture plus minutieuse, vous étudierez progressivement l'ensemble. C'est seulement à la fin de la partie que vous répondrez aux questions figurant dans chacun des guides d'étude, en vous appuyant sur les réponses.

Dans les deux cas de figure, nous vous recommandons de considérer chaque partie de l'ouvrage comme un minilivre à part entière. L'application de l'une ou l'autre de ces méthodes vous permettra de faire le lien à chaque fois entre les technologies et les approches conceptuelles sur lesquelles elles s'appuient. Utilisez la méthode qui vous arrange. Nous insistons simplement sur l'importance de ne pas lire l'ouvrage chapitre après chapitre, mais partie par partie.

Faites également attention aux symboles  : ils marquent un point important, en soulignant un détail fondamental du concept expliqué. Prenez le temps de bien les analyser et, si possible, recopiez-les dans votre cahier pour y revenir plus tard.

Parce que cet ouvrage a été rédigé pour vous – et parce que la mesure du talent d'un auteur n'est pas liée à ce qu'il a écrit, mais à ce que ses lecteurs en ont compris –, nous avons mis à votre disposition un site Internet non censuré, www.data-transitionnumerique.com, dans lequel vous pourrez communiquer vos retours et poser vos questions. Nous vous remercions d'avance pour votre collaboration.

Les compléments web

Pour télécharger le code source des exemples de cet ouvrage, ainsi que les réponses aux guides d'étude, veuillez-vous rendre à cette adresse : <http://www.editions-eyrolles.com/dl/0067478>.

Notes de l'auteur

Les références aux marques, aux entreprises et aux universités citées dans cet ouvrage n'ont, en aucune façon que ce soit, un but publicitaire ou promotionnel. Elles sont utilisées exclusivement à des fins pédagogiques et restent entièrement la propriété de leurs détenteurs.

Les marques citées et les logos d'entreprises sont des marques déposées d'entreprises situées en France, aux États-Unis ou partout dans le monde.

Les jeux de données utilisés pour illustrer le fonctionnement des technologies sont purement fictifs. Ils n'ont aucun caractère promotionnel et ne renvoient à aucune réalité concrète.

Les conseils, les tableaux comparatifs, les *benchmarks* de solutions et les prises de position présents dans l'ouvrage renvoient au point de vue personnel de l'auteur à la date de publication de ce livre. Aucun favoritisme n'a été fait lors des benchmarks et des comparaisons. Étant donné la vitesse avec laquelle évolue le monde de la technologie, beaucoup de ces conseils et tableaux peuvent devenir obsolètes après la publication de l'ouvrage. Ainsi, l'auteur ne peut être tenu pour responsable des dommages qu'aurait causés l'application de ces conseils après la date de publication de l'ouvrage.

En raison des changements rapides du marché, le contenu des sites web fournis peut être modifié ou les sites web eux-mêmes peuvent devenir indisponibles. L'auteur ne peut donc vous donner aucune garantie quant à la durabilité des sites Internet fournis en référence.

Il y a beaucoup d'anglicismes dans l'ouvrage. C'est un choix personnel de l'auteur qui veut, par là même, conserver la teneur sémantique originale des mots dans leur jargon.

Remerciements

Ce livre s'est inspiré de plus de 300 références d'articles, de 3 ouvrages, de multiples vidéos, blogs, articles de recherche. Ces ressources ont été pour moi une grande source d'enrichissement personnel et professionnel. Je souhaite donc exprimer ma gratitude à l'égard de tous ces auteurs pour le partage et la mise à disposition de leurs connaissances. Bien entendu, je porte la responsabilité de toute erreur qui se serait glissée dans l'ouvrage, même après le travail minutieux de toute l'équipe d'Eyrolles. Je tiens particulièrement à remercier :

- mon Père céleste, la source ultime de mon inspiration ;
- Alexandre Habian, pour avoir cru au projet, et toute l'équipe éditoriale d'Eyrolles, pour m'avoir accompagné dans la rédaction et la publication de l'ouvrage ;
- la vibrante communauté open source d'Hadoop, pour le développement des technologies qui aident les entreprises à valoriser leurs données ;
- l'équipe Insights & Data de Capgemini, pour m'avoir donné l'opportunité d'affiner mes compétences en big data ;
- Ariane Liger Belair, directrice académique de SAS Institute, pour le soutien lors de la rédaction de l'ouvrage et la validation des informations sur l'offre SAS High Performance Analytics ;
- Franck Kouemeni, CEO de l'agence de communication digitale, M. Kouems & associés et son équipe pour le développement du site web de l'ouvrage www.data-transitionnumerique.com ;
- Joël Belafa, *lead architect* chez Dataiku, dont l'anecdote a inspiré l'idée de ce livre et dont l'amitié indéfectible m'a aidé tout au long de la rédaction du livre.

Le succès de la publication de cet ouvrage est aussi le vôtre.

Introduction à l'écosystème Hadoop

La clé de la compréhension d'une technologie, ce n'est pas la technologie elle-même, mais le contexte dans lequel elle a été créée.

La solution conceptuelle au problème du traitement des données au sein de l'économie numérique est la suivante : les traitements/calculs sont divisés en tâches, et leur exécution est parallélisée au sein d'un *cluster* (grappe) d'ordinateurs complètement tolérants aux pannes. Dès lors, la démarche de centralisation qui avait prévalu jusqu'ici est tout simplement inenvisageable.

La tolérance aux pannes est fournie par un tout nouveau type de système de fichiers appelé *système de fichiers distribués* (DFS). Le découpage et la parallélisation des tâches reposent sur un nouveau modèle de programmation appelé *MapReduce*. Une *application générique de gestion de ressources* permet d'aller au-delà du MapReduce et d'utiliser d'autres modèles de calcul sur le cluster.

Ces trois éléments sont aux fondements de toutes les approches technologiques développées à l'ère du numérique.

De facto, Hadoop – l'implémentation MapReduce la plus populaire et la plus mature du marché – est en passe de devenir le standard. Nous pouvons donc affirmer, sans trop prendre de risques, qu'Hadoop va devenir la plate-forme de traitement des données par défaut pour l'utilisateur, un peu comme Excel est devenu progressivement le logiciel par défaut en matière d'analyse de données. Problème : à la différence d'Excel, Hadoop n'a pas été conçu à destination des *analystes métier* à la base, mais pour les développeurs.

Par *analyste métier*, nous entendons toute profession liée à un métier qui implique de près ou de loin le traitement des données : statisticien, analyste marketing, analyste de crédit, contrôleur de gestion, comptable, analyste financier, etc.

Selon la loi de Metcalfe, « la valeur d'un standard est proportionnelle au carré du nombre de systèmes qui l'utilisent ». Nous pouvons contextualiser cette citation en précisant que « la valeur d'une technologie est proportionnelle au carré du nombre de personnes qui l'utilisent ». En d'autres

termes, l'adoption d'Hadoop à grande échelle et son succès ne dépendent pas des développeurs, mais des analystes métier ; et la fondation Apache l'a bien compris. Voilà pourquoi elle s'évertue, depuis qu'elle a repris Hadoop en 2009, à rapprocher le plus possible le MapReduce (et ses dérivés) des utilisateurs métier.

Face à un marché fortement concurrentiel, la réponse ne s'est pas fait attendre. Car sur le marché, deux catégories d'acteurs poursuivent cet objectif : le monde de l'*open source*, centralisé autour de la fondation Apache, et le monde des éditeurs logiciels. Les deux offrent un ensemble d'outils qu'il est aujourd'hui raisonnable de qualifier d'écosystème Hadoop.

Dans le chapitre de présentation de cet ouvrage exclusivement réservé à l'apprentissage des outils de l'écosystème technologique d'Hadoop, nous allons d'une part revenir sur les fondements d'Hadoop et, d'autre part, expliquer la philosophie et la taxonomie¹ des outils qui constituent son écosystème. Vous verrez ainsi en quoi ces outils facilitent l'adoption d'Hadoop par un public de non-développeurs.

Hadoop

Pensez à toutes les données que nous avons aujourd'hui : en 2012 déjà, IDC émettait le postulat selon lequel, de 2005 à 2020, le volume de données croîtrait d'un facteur 300, de 130 exaoctets à 40 000 exaoctets, soit 40 trillions de gigaoctets, ce qui représente plus de 5 200 gigaoctets créés pour chaque homme, femme et enfant en 2020. Actuellement, le volume des données en circulation connaît une démultiplication permanente : 5 exaoctets de données sont désormais produits tous les deux jours, soit le même volume que l'ensemble des données produites de l'aube de la civilisation jusqu'à 2003. En 2014, 90 % de toutes les données jamais générées par l'homme l'ont été au cours des deux dernières années. Cisco renchérit sur ce constat lorsqu'il estime le trafic réseau global annuel à 1,3 zettaoctets en 2016.

Cette intensification du trafic réseau est attribuée à l'accroissement du nombre des smartphones, tablettes et autres appareils connectés à Internet, à la croissance des communautés d'utilisateurs, à l'augmentation de la bande passante, aux débits en hausse constante offerts par les opérateurs de télécommunications, ainsi qu'à la prolifération de la disponibilité et de la connectivité du Wi-Fi. Nous ne parlerons même pas de la variété des données créées !

L'échelle de cette croissance des données surpasse la capacité raisonnable des technologies traditionnelles – et plus précisément celle des Systèmes de gestion de bases de données relationnelles (SGBDR) – ou même la configuration matérielle typique permettant l'accès à ces données. Plus encore, les données canalisées vers Internet créent de la pression en vue d'une capture cohérente et rapide.

Les entreprises doivent trouver le moyen de maîtriser et traiter efficacement ces données pour continuer à servir fidèlement leur clientèle et rester compétitives. Google fait partie des entreprises qui ont très tôt ressenti le besoin de gérer efficacement les gros volumes de données liés aux requêtes des utilisateurs. Rappelons que le moteur de recherche de Google doit restituer en temps quasi réel les résultats de plus de 3 millions de recherches effectuées par minute. Pour

¹ taxonomie : « classification, suite d'éléments formant des listes qui concernent un domaine, une science » (*Larousse*)

répondre à cette exigence, il doit donc indexer l'intégralité des pages web disponibles et rechercher à l'intérieur de chacune les mots demandés par l'utilisateur.

Aujourd'hui, le nombre de sites Internet dans le monde est estimé à plus d'un milliard, avec une croissance de 5,1 % (estimation faite par Netcraft), sachant qu'au-delà de 5 secondes d'attente, l'utilisateur considère que la requête a échoué et passe à autre chose. Résoudre ce problème ne peut se faire avec les approches de centralisation des données au cœur d'un seul serveur – comme on le faisait dans le passé – ni même avec les approches centralisées de traitement telles que l'*in-memory*. Il est dès lors indispensable de penser à d'autres approches.

Le traitement in-memory fait référence à une approche qui consiste à charger et traiter toutes les données en mémoire vive (RAM).

L'approche conceptuelle d'Hadoop

Pour répondre à ces défis, l'idée de Google est de développer une approche conceptuelle consistant, d'une part, à distribuer le stockage des données et, d'autre part, à paralléliser le traitement de ces données sur plusieurs nœuds d'une grappe de calcul (un cluster d'ordinateurs).

L'emploi d'une grappe de calcul n'est pas anodin. En effet, tout en étant l'infrastructure qui sert de support au traitement massivement parallèle, son utilisation permet de profiter des économies d'échelle engendrées par la baisse des coûts d'ordinateurs. Ainsi, la croissance des données est gérée en augmentant simplement les nœuds dans le cluster. Cette approche conceptuelle a été adoptée par le marché ; elle sous-tend l'ensemble des technologies du big data actuellement.

- **Au niveau du traitement.** Google découpe le problème d'indexation des pages web en tâches ou sous-problèmes à distribuer dans le cluster pour exécution. Pour ce faire, il construit un *index inversé* pour chaque mot-clé contenu sur chaque page web. Pour faire simple, un index inversé, c'est un peu comme la page d'index d'un livre ; pour chaque mot-clé, sont référencées ses différentes localisations sur l'ensemble des sites web. Ainsi, chaque fois que vous recherchez un mot, l'index fournit les pages où il se situe ; cela vous évite de fouiller tout le Web page par page pour retrouver le mot.

Aujourd'hui, la recherche Internet fonctionne selon ce principe : le moteur de recherche constitue une base d'index inversés pour chaque mot. Cependant, construire un index inversé n'est pas simple. Prenons un exemple : si le moteur de recherche indexe 1 000 pages contenant chacune 5 000 mots, il doit construire un fichier de 5 millions de mots (1 000 × 5 000).

Pour construire l'index inversé, Google passe par un raisonnement en trois phases consécutives.

- La phase **Map** consiste à assigner à chaque nœud du cluster la tâche d'attribuer à chaque mot de la page web un indice correspondant à la page dans laquelle il est. Cet indice peut être le titre de la page, son numéro ou n'importe quel élément qui permet de l'identifier de façon unique parmi toutes les pages qui constituent le site web. Cette tâche s'exécute parallèlement (en même temps) dans tout le cluster.

- La phase *Shuffle* consiste, pour chaque nœud, à trier par ordre alphabétique les mots auxquels il a affecté un index. Cette étape intermédiaire vise à faciliter le travail effectué par la troisième phase.
- La phase *Reduce* consiste, pour chaque mot contenu dans l'ensemble des nœuds du cluster, à regrouper tous ces indices. Ainsi, on obtient l'index inversé.
- **Au niveau du stockage.** Google décide de ne pas centraliser le stockage de toutes les pages Internet vers un seul serveur pour la construction des index inversés. De toute façon, cette approche n'est pas envisageable, puisque le volume de données généré aujourd'hui dépasse largement la capacité des serveurs traditionnels.

Comme la tâche de construction des index est partagée entre les nœuds du cluster, les fichiers contenant les mots (les pages web) doivent être divisés, et chaque morceau de fichier doit être stocké de façon redondante sur tous les disques durs du cluster. Ainsi, si un nœud tombe en panne au cours du traitement, cela n'affecte pas les autres tâches.

Techniquement, le stockage de fichiers sur un disque dur se fait à l'aide de ce que l'on appelle un *système de fichiers (file system)*. Le système de fichiers est unique pour chaque ordinateur, ce qui pose problème dans un cluster où l'ensemble des nœuds doivent être vus comme une seule machine. Pour résoudre ce problème et gérer la redondance des données sur plusieurs disques durs, Google a mis sur pied le *système de fichiers distribués (Distributed File System ou DFS)*, qui est installé sur le cluster.

La figure suivante résume le travail de construction d'un index inversé, par Google, au cours de la procédure en trois phases décrite plus haut. La phase de normalisation retire la ponctuation et les *stop words* (mots sémantiquement vides) et convertit tous les mots en minuscules. La phase Map produit des paires clé/valeur constituées du mot et de son index, tandis que la phase Shuffle trie ces mêmes paires clé/valeur. La phase Reduce, enfin, regroupe dans un fichier de sortie l'ensemble des index.

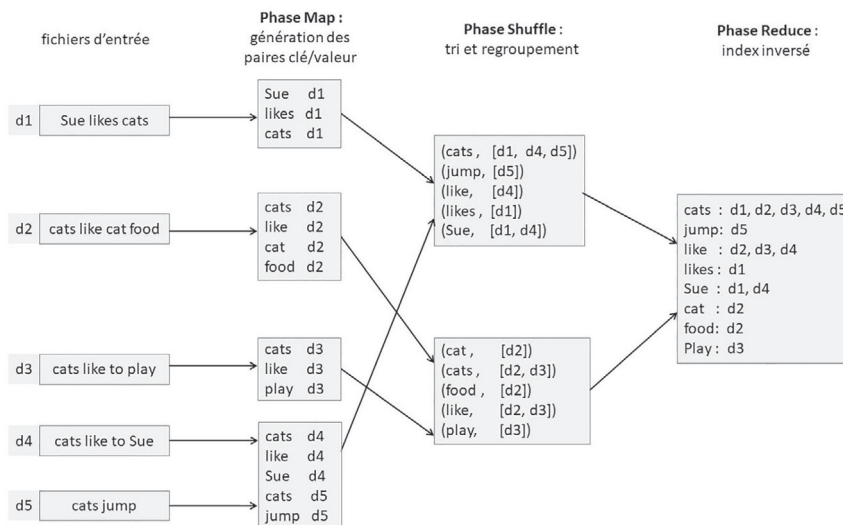


Figure I-1 – Utilisation de l'approche MapReduce pour créer un index inversé

À travers cette approche conceptuelle, Google parvient à gérer les problématiques liées à l'indexation des pages web et à tirer parti des données générées dans l'espace numérique. Le découpage des traitements en plusieurs tâches et l'exécution en parallèle de ces dernières sur un grand nombre de nœuds se font à l'aide du modèle en trois étapes présenté plus haut, que Google a baptisé **MapReduce**. La distribution, le stockage et la redondance des fichiers sur le cluster sont pris en charge par le système de fichiers distribués mis au point par Google et qu'il a appelé *Google File System (GFS)*.

À l'origine, ces deux éléments étaient utilisés en interne chez Google. Plus tard, un ingénieur de l'entreprise, **Doug Cutting**, implémentera en Java le MapReduce et le GFS et leur donnera le nom d'une des peluches de son fils : **Hadoop**. Le **HDFS** (*Hadoop Distributed File System*) constitue son système de fichiers distribués, l'équivalent du GFS.

Depuis 2009, le projet Hadoop a été repris par la fondation Apache. Aujourd'hui, il constitue officiellement un *framework* open source (c'est-à-dire qu'il peut être utilisé librement par tout le monde).

Fonctionnement d'Hadoop

Comme précisé plus haut, le MapReduce est une approche conceptuelle ; il a besoin d'être implémenté pour être utilisé. Hadoop répond à cette demande : c'est l'implémentation la plus populaire et la plus mature actuellement sur le marché.

En réalité, Hadoop est un ensemble de classes² écrites en Java pour la programmation des tâches MapReduce et HDFS, dont les implémentations sont disponibles dans d'autres langages de programmation : en Scala et C# notamment. Ces classes permettent à l'analyste d'écrire des fonctions Map et Reduce qui vont traiter les données, sans avoir à savoir comment ces fonctions sont distribuées et parallélisées dans le cluster. Dans cette section, nous allons expliquer le fonctionnement d'Hadoop : les étapes d'exécution d'un programme MapReduce dans un cluster et la façon dont il est parallélisé.

Terminologie d'Hadoop

Avant de parler de l'exécution des *jobs* MapReduce dans Hadoop, nous allons en présenter la terminologie.

Un *job MapReduce* est une unité de travail que le client veut exécuter. Il consiste en trois choses :

- le fichier des données à traiter (*input file*) ;
- le programme MapReduce ;
- les informations de configuration (métadonnées).

Le cluster exécute le job en divisant le programme MapReduce en deux : les tâches Map d'un côté et les tâches Reduce de l'autre. Dans le cluster Hadoop, il y a deux types de processus qui contrôlent l'exécution du job : le *JobTracker* et un ensemble de *TaskTrackers*.

² Une « classe » est une notion renvoyant aux méthodes orientées objet. C'est une structure abstraite qui décrit un objet du monde réel sous deux angles : ses propriétés (caractéristiques) et ses méthodes (les actions qu'il peut effectuer). Par exemple, la classe *Vehicule* représente un véhicule, *couleur* est l'une de ses propriétés et *accélérer/freiner* est l'une de ses méthodes.

Le JobTracker est le processus central qui est démarré sur le nœud de référence (*name node*) ; il coordonne tous les jobs qui s'exécutent sur le cluster, gère les ressources de ce dernier et planifie les tâches à exécuter sur les TaskTrackers.

Ces derniers sont les processus qui traitent le programme MapReduce de l'analyste ; ils sont démarrés au niveau des nœuds de données, exécutent les tâches Map ou Reduce et envoient des rapports d'avancement au JobTracker, qui garde une copie du progrès général de chaque job. Si une tâche échoue, le JobTracker peut la replanifier sur un TaskTracker différent.

En fait, le JobTracker désigne le nœud de référence et le processus Master qui y est démarré, tandis que le TaskTracker fait référence au nœud de données et au processus Worker qui y est lancé.

La figure ci-après illustre les relations entre le JobTracker et les TaskTrackers dans le cluster. Le cluster Hadoop présente cinq nœuds (celui de référence secondaire n'est pas comptabilisé) : les nœuds de données sont des TaskTrackers, tandis que le nœud de référence est un JobTracker.

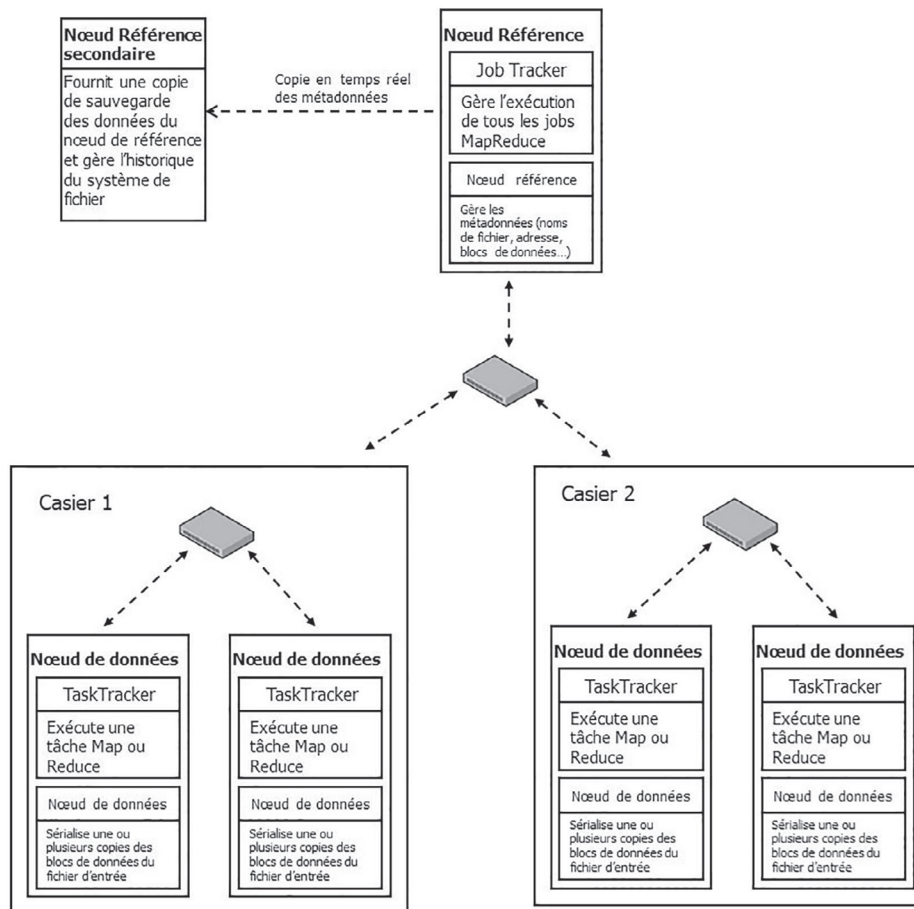


Figure I-2 – Le cluster Hadoop et ses cinq nœuds

Chaque tracker possède une partie stockage, assurée par le nœud, et une partie traitement, effectuée par le processus (Worker ou Master) démarré sur le nœud. Dans la terminologie Hadoop, le nom du processus est associé à la machine ; on parle de JobTracker et de TaskTracker.



Attention

Le *nœud de référence secondaire* n'est pas un nœud de relais ; il ne prend pas la place du nœud de référence si celui-ci tombe en panne. Il effectue seulement la sauvegarde des métadonnées contenues dans le nœud de référence. En cas de panne du nœud de référence, l'administrateur du cluster Hadoop se sert de ces métadonnées, pour restaurer manuellement le cluster.

Maintenant que nous avons réglé le problème de terminologie, regardons ensemble la façon dont le MapReduce y est exécuté.

Détails d'exécution d'un modèle de calcul dans Hadoop

Même si Hadoop correspond à l'implémentation du MapReduce, il a, depuis sa version 2, la capacité d'exécuter plusieurs modèles de calcul sur le cluster. Cela est possible grâce à un gestionnaire de ressources tel que Mesos ou YARN. Nous y reviendrons dans le chapitre 11. Néanmoins, tous les modèles de calcul sont exécutés de la même façon par Hadoop.

Illustrons ici les étapes d'exécution d'un modèle de calcul sur Hadoop avec le MapReduce. Le job MapReduce écrit par l'utilisateur s'exécute en sept étapes.

1. Au départ, l'utilisateur configure le job : il écrit la fonction Map, la fonction Reduce, spécifie le nombre r de tâches Reduce, le format de lecture du fichier d'entrée, le format des r fichiers de sortie, éventuellement la taille des blocs du fichier d'entrée et le facteur de réplication. Une fois que tout cela est fait, le JobTracker démarre les r TaskTrackers qui vont effectuer les r tâches Reduce spécifiées par l'utilisateur.

Le *facteur de réplication* est le nombre de fois qu'un bloc de fichier est répliqué dans le cluster par le HDFS. Il est défini par l'utilisateur, et par défaut il est égal à 3. Prenons un exemple : un facteur de réplication égal à 3 signifie que chaque bloc de fichier est répliqué par le HDFS 3 fois dans 3 nœuds différents du cluster.

2. Le HDFS découpe le fichier d'entrée en M blocs de taille fixe, généralement 64 Mo. Ensuite, le HDFS réplique ces blocs selon le facteur défini par l'utilisateur (3, par défaut) et les distribue de façon redondante dans des nœuds différents du cluster. Diviser le fichier d'entrée en blocs de taille fixe permet de répartir de façon équilibrée la charge de traitement parallèle entre les nœuds du cluster ; ainsi, le traitement s'achève à peu près au même moment dans l'ensemble des nœuds du cluster.
3. Par défaut, le JobTracker déclenche M TaskTrackers sur les M nœuds de données dans lesquels ont été répartis les M blocs du fichier d'entrée, pour exécuter les tâches Map, soit un TaskTracker Map pour chaque bloc de fichiers. Chaque TaskTracker lit le contenu du bloc de fichiers par rapport au format d'entrée spécifié par l'utilisateur, puis le transforme en

paires clé/valeur par le processus de hachage défini dans la fonction Map. Ce processus de hachage s'effectue dans la mémoire locale du nœud.

4. Périodiquement, dans chaque nœud, les paires clé/valeur sont sérialisées dans un fichier sur le disque dur local du nœud. Ensuite, ce fichier est partitionné en r régions (correspondant aux r tâches Reduce spécifiées) par une fonction de hachage qui va affecter à chacune une clé correspondant à la tâche Reduce à laquelle elle a été. Les informations sur la localisation de ces régions sont transmises au JobTracker, qui les fait suivre aux r TaskTrackers qui vont effectuer les tâches Reduce.



Attention

Le HDFS et le système de fichiers du nœud de données sont deux choses bien distinctes.

Chaque TaskTracker Map écrit ses paires clé/valeur sur le disque dur local du nœud sur lequel il est exécuté (le système de fichiers local du nœud), et non sur le HDFS (le système de fichiers global du cluster). Pourquoi ? Parce que ces résultats sont intermédiaires et sont utilisés comme entrées des tâches Reduce pour produire le résultat final du job.

Une fois que le job est complet, les résultats du Map ne servent plus à rien et peuvent être supprimés. Ainsi, les stocker dans le HDFS revient à les répliquer, ce qui ne sert strictement à rien.

5. Lorsque les r TaskTrackers Reduce reçoivent les informations de localisation, ils utilisent des appels de procédures distantes (protocole RPC) pour lire – depuis le disque dur des nœuds sur lesquels les tâches Map se sont exécutées – les régions des fichiers Map leur correspondant. Ensuite, ils les trient par clé. Notez au passage que le tri s'effectue en mode batch dans la mémoire du TaskTracker Reduce. Si les données sont trop volumineuses, alors cette étape peut augmenter de façon significative le temps total d'exécution du job.



Attention

Bien que les TaskTrackers Reduce téléchargent par appel de procédure distante les données des TaskTrackers Map, les traitements Reduce ne peuvent pas commencer tant que les tâches Map ne sont pas complètement achevées.

6. Les TaskTrackers Reduce itèrent à travers toutes les données triées, puis ils passent chaque clé unique rencontrée, avec sa valeur, à la fonction Reduce écrite par l'utilisateur. Les résultats du traitement de la fonction Reduce sont alors sérialisés dans le fichier r_i (avec i , l'indice de la tâche Reduce) selon le format de sortie spécifié par l'utilisateur. Cette fois-ci, les fichiers ne sont pas sérialisés dans le disque dur du nœud TaskTracker, mais dans le HDFS, et ce pour des raisons de résilience (tolérance aux pannes).
7. Le job s'achève là ; à ce stade, les r fichiers Reduce sont disponibles et Hadoop applique, selon la demande de l'utilisateur, soit un `Print Ecran`, soit leur chargement dans un SGBD, soit encore leur passage comme fichiers d'entrée à un autre job MapReduce.

La figure suivante récapitule ces sept étapes. Le jaune traduit les traitements, le vert la RAM, le blanc les opérations d'accès aux données et les cylindres bleus les fichiers Map.

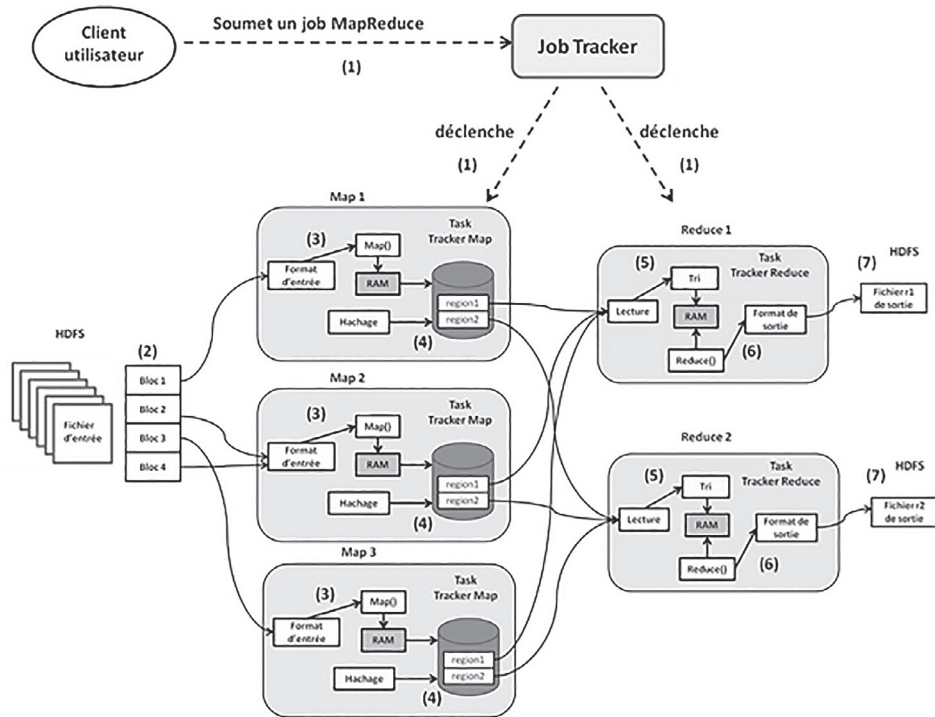


Figure I-3 – Étapes d'exécution d'un job MapReduce dans un cluster Hadoop

Voilà, à ce stade, vous avez les informations générales concernant le fonctionnement d'Hadoop. Si vous souhaitez en apprendre plus sur le fonctionnement d'Hadoop, nous vous recommandons l'ouvrage ***Hadoop – Devenez opérationnel dans le monde du big data***, J. Chokogoue, ENI, 2017.

Passons maintenant au nerf de la guerre et examinons l'écosystème Hadoop, afin de comprendre son importance, ses enjeux et pourquoi il requiert à lui seul tout un ouvrage.

Taxonomie de l'écosystème Hadoop

En réalité, Hadoop est une plate-forme qui implémente des modèles de calcul parallèle comme le MapReduce et fournit un système de fichiers distribués redondant, fiable et optimisé pour la gestion des fichiers volumineux (HDFS). Comme précisé plus haut, Hadoop est un ensemble de classes écrites en Java pour la programmation des tâches MapReduce et HDFS. Ces classes permettent à l'analyste de décrire des fonctions de traitement des données, sans avoir à se préoccuper de la façon dont ces fonctions sont distribuées et parallélisées dans le cluster.

Pour tirer le meilleur parti d'un cluster Hadoop, la fondation Apache y a intégré une série de logiciels et d'outils. Cet ensemble forme aujourd'hui l'écosystème Hadoop ou *framework Hadoop*. Sans lui, il reviendrait à chaque entreprise, en fonction de son besoin, de développer elle-même

des outils compatibles avec Hadoop afin de déployer ses solutions sur le cluster, ce qui serait très complexe et l'éloignerait probablement de son activité principale.

Actuellement, de nombreux développeurs travaillent sur l'écosystème Hadoop et offrent leur concours à la fondation Apache : Yahoo!, par exemple, qui a développé ZooKeeper (service de coordination distribué), Cloudera, le créateur d'Impala (moteur de calcul SQL massivement parallèle), ou LinkedIn, le développeur de Kafka (système de messagerie publish/subscribe distribué).

L'écosystème Hadoop fournit une collection d'outils et de technologies spécialement conçue pour faciliter le développement, le déploiement et la prise en charge des solutions big data. La définition de cet écosystème est importante, car l'adoption d'Hadoop par les entreprises s'en trouve facilitée et leur permet de surmonter les défis du numérique.

Tous ces outils peuvent être rangés selon 14 catégories, en fonction de la problématique qu'ils résolvent.

- **Langages d'abstraction.** Ils servent à développer des jobs MapReduce à l'aide d'un langage similaire au SQL. Dans cette catégorie, on distingue principalement Hive, Pig et Cascading.

Nous étudierons principalement Hive et Pig un peu plus loin.

- **SQL sur Hadoop.** Il s'agit de technologies permettant d'exécuter nativement du SQL sur un cluster Hadoop. À ce jour, trois projets sont en incubation à la fondation Apache, bien qu'opérationnels : Impala, Phoenix et HAWQ.

Nous étudierons les trois technologies.

- **Modèles de calcul.** Ce sont des moteurs d'exécution d'algorithmes en parallèle ou des modèles de programmation des tâches distribuées sur un cluster. Actuellement, Hadoop dispose de cinq modèles de calcul (chacun s'exécutant à l'aide de son propre moteur) : MapReduce, le modèle original d'Hadoop, Spark, le moteur in-memory distribué créé par l'université de Berkeley, Mahout, Hama et Tez.

Puisque le modèle de calcul est l'élément central du cluster, et compte tenu des problématiques rencontrées à l'heure du numérique, nous les étudierons tous.

- **Outils de traitement en temps réel.** Ils servent à traiter immédiatement des données générées en *streaming* (au fil de l'eau). Quatre outils sont disponibles dans cette catégorie : Storm, Samza, S4 et Spark Streaming.

Dans cette catégorie, nous étudierons Storm.

- **Bases de données.** Ce sont des systèmes de gestion de bases de données distribuées (SGBDD), adaptés au fonctionnement sur un cluster. Plusieurs SGBDD open source existent, mais ils ne font pas partie de la fondation Apache. Les seuls systèmes attachés à la fondation sont, à notre connaissance, HBase, Cassandra et Accumulo.

Nous présenterons ici HBase.

- **Outils d'ingestion & streaming.** Il s'agit d'outils capables d'« ingérer » des données générées en streaming. La fondation Apache en propose deux : Kafka et Flume.

En raison de leur complexité, nous leur avons consacré un ouvrage entier. Il sera disponible bientôt.

- **Outils d'intégration des données.** Ils permettent de déplacer les données d'un SGBDR vers le HDFS et vice versa.

Le seul outil disponible dans cette catégorie est Scoop, que nous étudierons bien évidemment.

- **Outils de coordination de workflow.** Ils servent à planifier et chaîner l'exécution de plusieurs jobs dans le cluster Hadoop.

Oozie est le seul outil disponible dans cette catégorie. Nous l'analyserons dans le détail.

- **Outils de coordination de services distribués.** Ils coordonnent les échanges entre les nœuds d'un cluster, ou entre plusieurs applications.

ZooKeeper est l'outil de coordination de service distribué le plus populaire à ce jour. Nous lui consacrerons un chapitre entier.

- **Outils d'administration de cluster.** Ils servent notamment à gérer l'approvisionnement des ressources du cluster et les permissions des utilisateurs, à y suivre l'exécution des tâches, etc.

Plusieurs outils indépendants ont été développés pour l'administration Hadoop, combinés dans Ambari, que nous présenterons.

- **Outils d'interface utilisateur.** C'est grâce à eux que l'utilisateur interagit avec le cluster et les outils d'Hadoop.

Actuellement, la fondation Apache offre une interface web appelée HUE. Nous prendrons un moment pour la décrire également.

- **Outils d'indexation de contenu.** Ils rendent possible le *full text search*, c'est-à-dire la recherche en texte intégral. Ils permettent d'effectuer des recherches de contenu, d'indexer le texte des pages Web et fournissent des fonctionnalités de recherche sur ces dernières. Dans cette catégorie, on distingue les moteurs Lucene, Solr et Lucy. En dehors de ces trois moteurs, un autre est également très utilisé, mais il ne fait pas partie de la fondation Apache : Elasticsearch, qui s'appuie sur Lucene.

Nous étudierons donc Lucene et Elasticsearch.

- **Systèmes de fichiers distribués.** Il s'agit de couches abstraites, installées sur le cluster pour gérer le stockage des données distribués. Hadoop est fourni avec son système de fichiers distribués initial, le *HDFS*, que nous n'étudierons pas (cela exigerait qu'on s'attarde sur les détails techniques du fonctionnement d'Hadoop, ce qui dépasse le cadre de cet ouvrage).

Un chapitre entier est dédié à l'étude des systèmes de fichiers distribués et du HDFS dans l'ouvrage suivant : *Hadoop – Devez-vous être opérationnel dans le monde du Big Data*, J. Chokogoue, ENI, 2017.

- **Gestionnaires de ressources.** Ce sont des outils qui exploitent au mieux les ressources d'un cluster. Ils permettent d'exécuter plusieurs modèles de calcul sur un cluster. Souvenez-vous qu'Hadoop est originellement l'implémentation du MapReduce ; en d'autres termes, il est limité à l'exécution de ce modèle. Les gestionnaires de ressources lèvent cette limite. Actuellement, la fondation Apache en fournit deux : YARN et Mesos.

Nous les étudierons tous les deux, en insistant sur YARN.

Une **distribution Hadoop** – version commerciale du framework – est composée d'un outil de chacune des catégories annoncées précédemment. Leur maîtrise vous permet d'embrasser les problématiques du traitement de gros volumes de données à l'ère du numérique.