Кібернетика та системний аналіз

УДК 004.415.2 DOI: 10.30748/zhups.2020.63.11

С.В. Мінухін

Харківський національний економічний університет ім. С. Кузнеця, Харків

ДОСЛІДЖЕННЯ ПРОДУКТИВНОСТІ КЛАСТЕРА APACHE SPARK НА ПЛАТФОРМІ AZURE ДЛЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ

Розглянуто та досліджено питання підвищення продуктивності застосування моделей та методів задач машинного навчання з використанням Apache Spark Azure HDInsight. Для підвищення обгрунтованості отриманих результатів використано один з найбільш відомих бенмарків для тестування бібліотек машинного навчання Spark-Perf. Наведені кроки щодо встановлення, розгортання та налаштування Apache Spark на платформі Azure. Для оцінки ефективності розподілених обчислень використано метрики продуктивності щодо середнього часу навчання та тестування та їх відношення. Проведений порівняльний аналіз результатів розв'язку задач з бібліотеки MLlib для кластерів з гомогенною та гетерогенною архітектурою, які свідчать про високу ефективність їх використання.

Ключові слова: *модель, машинне навчання, Apache Spark, Azure HDInsight, Spark-Perf, MLlib, дерева рішень, навчання, тестування.*

Вступ

Постановка проблеми. Хмарні обчислення – один з напрямів інформаційних технологій, що стрімко розвивається та все більше використовується. Хмарні обчислення – це надання обчислювальних потужностей, сховищ для баз даних, додатків та інших IT-ресурсів на платформах хмарних сервісів через Інтернет з оплатою за фактом використання. Платформи хмарних сервісів надають швидкий доступ до гнучких і недорогих IT-ресурсів. Хмарні обчислення дозволяють позбутися великих попередніх витрат на обладнання і заощадити час, необхідний для управління ним. Замість цього можна розподілити обчислювальні ресурси таких типів і розмірів, які необхідні для реалізації ідей користувача або для управління IT-відділом. Можна практично миттєво отримувати доступ до необхідної кількості ресурсів з оплатою за фактом використання. Хмарні обчислення забезпечують простий доступ до серверів, сховищ, баз даних і великого набору сервісів додатків в Інтернеті.

Хмарні обчислення мають велику кількість переваг: капітальні витрати перетворюються в змінні. Немає необхідності вкладати великі кошти в центри обробки даних (ЦОД) і сервери, не знаючи заздалегідь, які потужності згодом будуть потрібні. Можна платити тільки за фактичне використання обчислювальних ресурсів; значна економія при великих обсягах. При використанні хмарних обчислень можна досягти нижчої змінної вартості, ніж при створенні власної обчислювальної інфраструктури. Оскільки сотні тисяч споживачів спільно використовують хмарні ресурси, постачальники хмарних послуг можуть забезпечити значну економію за рахунок масштабу, що дозволяє знизити вартість фактичного використання ресурсів; немає необхідності прогнозувати, який обсяг ресурсів інфраструктури буде потрібним. Приймаючи рішення щодо обсягу ресурсів для розгортання застосувань, часто в кінцевому рахунку можна зіткнутися з простоєм дорогих ресурсів або нестачею потужностей. При використанні хмарних обчислень ці проблеми зникають. Можна використовувати тільки потрібні ресурси з можливістю масштабування в бік збільшення або зменшення всього за кілька хвилин. Доступ до нових IT-ресурсів для розробників в середовищі хмарних обчислень можна отримати досить швидко, що дозволяє скоротити час, необхідний для їх впровадження. В результаті організація стає більш гнучкою, оскільки на експерименти і розробку витрачається набагато менше часу і коштів. До найбільш розвинутих надпродуктивних систем оброблення великих даних є Apache Spark – програмне забезпечення з відкритим кодом для розвязку задач Bigdata, що має переваги перед іншими фреймвьорками, зокрема: є динамічним за природою, підтримує обчислення в пам'яті на основі технології RDD та надає можливість повторного використання, толерантності до помилок, обробки потоку в реальному часі тощо [1–3]. Фреймворк Spark – це сучасна платформа обчислювальних кластерів та влючає API, яке допомагає клієнтам, що працюють з великими да-

ними, виконувати потокове, машинне навчання або робочі високонавантажені запити SQL, які вимагають повторного доступу до наборів даних та може виконувати одночасно пакетну обробку та обробку потоків. Пакетна обробка відноситься до обробки раніше зібраної роботи в єдиному пакеті, в той час як потокова обробка означає роботу з потоковими даними Spark. Він може отримати доступ до будь-якого джерела даних Hadoop, а також може працювати на кластерах Hadoop. Крім того, Apache Spark поширює можливості Hadoop MapReduce, що включає в себе ітераційні запити та обробку потоків (рис. 1) [1–3].

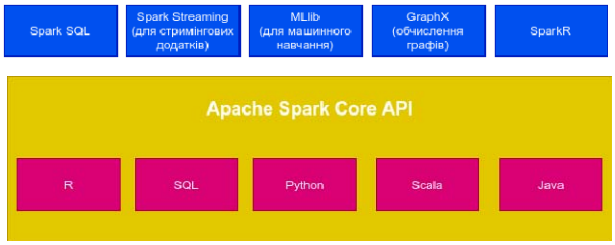

Рис. 1. Склад компонент Apache Spark

Spark використовує архітектуру майстер-робітник: існує драйвер, який спілкується з одним координатором – майстром, який керує робітниками (робочими вузлами). Драйвер і робітники працюють в своїх власних процесах Java.

Можна запустити їх на одній машині (вертикальний кластер) або на окремих машинах (горизонтальний кластер) або в змішаній конфігурації комп'ютерів. Одним із сервісів що надаються хмарними платформами, є машинне навчання. відомим терміном для платформ такого роду є абревіатура MLaaS (Machine learning as a Service) – машинне навчання як сервіс [4]. Під цим визначенням розуміють різні хмарні платформи, які охоплюють більшість інфраструктурних питань, таких як попередня обробка даних, навчання та оцінка моделей, а також прогнозування. Постачальники послуг пропонують такі інструменти, як предиктивна аналітика, глибоке навчання, API, візуалізація даних, обробка природних мов тощо. Сервіси, що були перераховані, можна використовувати у випадках, коли потрібно зробити аналіз невеликого обсягу даних, але не можуть бути використані для оброблення великих даних. Для цих задач існує ряд інших інструментів та сервісів, які оптимізовані для розподіленої обробки даних у реальному часі.

Одними з найбільш відомих представників сервісів цього класу є Azure HDInsight, Amazon EMR, Google Cloud Dataproc тощо. Azure HDInsight [5–6] є хмарним дистрибутивом компонентів Hadoop. Azure HDInsight забезпечує просту, швидку і економічну обробку великих даних. Він надає можливість реалізувати різні сценарії, такі як отримання, перетворення, завантаження та зберігання даних, машинне навчання та Інтернет речей за допомогою таких платформи з відкритим кодом, як Hadoop, Spark, Hive, Kafka, Storm, R та інших.

Azure HDInsight надає можливість створити власну конфігурацію кластеру [5], використовуючи різні класи віртуальних машин: початкового рівня, загального призначення, для над продуктивних обчислень та оптимізовані по пам’яті. **Метою даної роботи** є дослідження продуктивності кластера Apache Spark в Azure HDInsight для розв’язку задач машинного навчання.

Виклад основного матеріалу

Огляд бібліотеки машинного навчання Spark MLlib та тестових завдань (бенчмарків)

Spark MLlib [7] – це бібліотека машинного навчання, яка забезпечує як ефективність, так і високоякісні алгоритми, а її здатність до обробки даних в пам'яті значно покращує продуктивність ітеративного алгоритму. В Spark 2.0 API RDD на базі пакету spark.mllib увійшов до режиму технічного обслуговування. У цьому випуску API на основі DataFrame є основним API для навчання Spark. Причина переходу MLlib на API на основі DataFrame полягає в тому, що він є більш зручним для користувача, ніж RDD. Однією з переваг використання DataFrames є джерела даних Spark Data, SQL. На рис. 3 зображена типова схема взаємодії компонентів Spark з пакетом Spark MLlib.

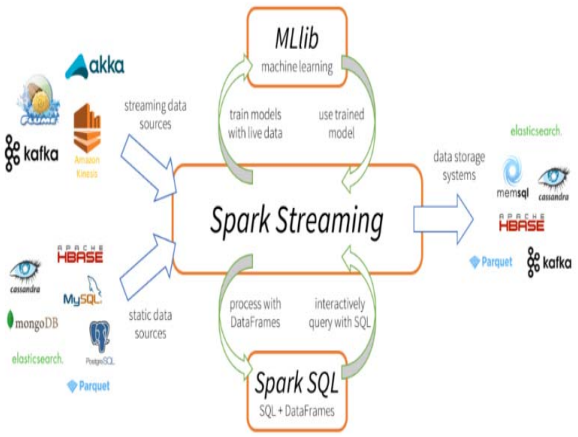

Рис. 2. Схема взаємодії компонентів Spark з пакетом MLlib

Бібліотека MLlib складається з двох програмних пакетів: spark.mllib – містить оригінальний API, побудований на RDD; spark.ml – реалізація API більш високого рівня, побудований з використанням DataFrames для конструювання конвеєрів машинного навчання, ML Pipelines. Задачі машинного навчання, які можна вирішити вбудованими в Spark MLlib алгоритмами машинного навчання, включа-

ють: регресію: лінійна регресія (Linear regression); узагальнену лінійну регресію (Generalized linear regression); дерева рішень (Decision tree regression); регресію випадкового лісу (Random forest regression); регресію дерева градієнтного бустингу (Gradient-boosted tree regression); регресію виживання (Survival regression); ізотонічну регресію (Isotonic regression); базова статистику; алгоритми оптимізації: градієнтний спуск (Gradient descent); стохастичний градієнтний спуск (Stochastic gradient descent, SGD); алгоритм BFGS з обмеженою пам'яттю (Limited-memory BFGS, L-BFGS), тощо. Для тестування продуктивності кластеру Apache Spark існує декілька бенчмарків [8]: Spark-Bench, HiBench, BigDataBench-Spark та Spark-Perf. Spark-Bench [9] - це гнучка система для моделювання, порівнян-ня, тестування та бенчмаркінгу додатків Spark, а також самої системи Spark, від IBM. Spark-Bench спочатку починався як набір бенчмаркінгу, для отримання часових значень для алгоритмів, переважно машинного навчання. Але з часом він перетворився на дуже гнучкий фреймворк, придатний для багатьох випадків використання. Spark-Bench має чотири категорії тестування навантажень: додатки машинного навчання, що підтримуються Spark MLLib, додатки для обчислення графів, що підтримуються Spark GraphX; SQL- додатки, що підтримуються стандартною службою Spark SQL та Hive; потокові додатки, що підтримуються Spark DStream. Тестування бібліотеки машинного навчання виконується за допомогою трьох найбільш поширених алгоритмів для регресії, класифікації та рекомендацій. HiBench [10] – це набір бенчмарків для обробки великих даних, розроблений компанією Intel, який допомагає оцінювати різні фреймворки великих да-них з точки зору швидкості, пропускної спроможності і використання систем-них ресурсів. Він містить набір робочих навантажень Hadoop, Spark і наванта-жень потокової передачі, включаючи такі алгоритми як Sort, WordCount, TeraSort, Sleep, SQL, PageRank, індексування Nutch, Bayes, Kmeans, NWeight, розширений DFSIO та інші. HiBench має 6 категорій тестування навантаження: мікро-бенчмарки; бенчмарки машинного навчання; бенчмарки роботи SQL запитами; бенчмарки веб-пошуку; бенчмарки роботи з графами; бенчмарки для оцінки потокової передачі. Spark-Perf [11] – це бенчмарк для комплексної оцінки продуктивності кластеру, розроблений компанією Databricks. До бенчмарку входять комплекти тестів на перевірку ефективності таких компонент як ядро Spark (Core Spark), PySpark, Spark Streaming та MLlib.

Особливостями бенчмарку Spark-Perf є те, що він має можливість настроїти потрібну конфігурацію тестів а також автоматично завантажувати та створювати потрібну конфігурацію кластеру Apache Spark. Spark-Perf має найбільше покриття тестами алгоритмів машинного навчання серед аналогічних пакетів бенчмарків для тестування великих даних (табл. 1).

Таблиця 1

Склад тестів машинного навчання бенчмарку Spark-Perf

Ідентифікатор тесту	Назва тесту	Кількість випробувань	Кількість тестових примірників
Алгоритми регресії та класифікації			
glm-regression	Generalized Linear Regression Model	10	5000
glm-classification	Generalized Linear Classification Model	10	5000
naive-bayes	Naive Bayes	10	5000
Алгоритми дерева рішень			
decision-tree	Decision Tree	16 x 10	5000
Алгоритми рекомендацій			
als	Alternating Least Squares	10	500
Алгоритми кластеризаії			
kmeans	K-Means clustering	10	5000
lda	Latent Dirichlet allocation	10	
pic	Power Iteration Clustering	10	
gmm	Gaussian Mixture Model	10	5000
Статистичні алгоритми			
summary-statistics	Summary Statistics	10	
pearson	Pearson's Correlation	10	50000
spearman	Spearman's Correlation	10	50000
chi-sq-feature/gof/mat	Chi-square Tests	10	100000
Алгоритми лінійної алгебри			
svd	Singular Value Decomposition	10	
pca	Principal Component Analysis	10	

Закінчення табл. 1

block-matrix-mult	Matrix Multiplication	10	
Алгоритми пошуку асоціативних правил			
fp-growth	FP-growth frequent item sets	10	
prefix-span	PrefixSpan	10	
Алгоритми пошуку ознак			
word2vec	Word2Vec distributed presentation of words	10	

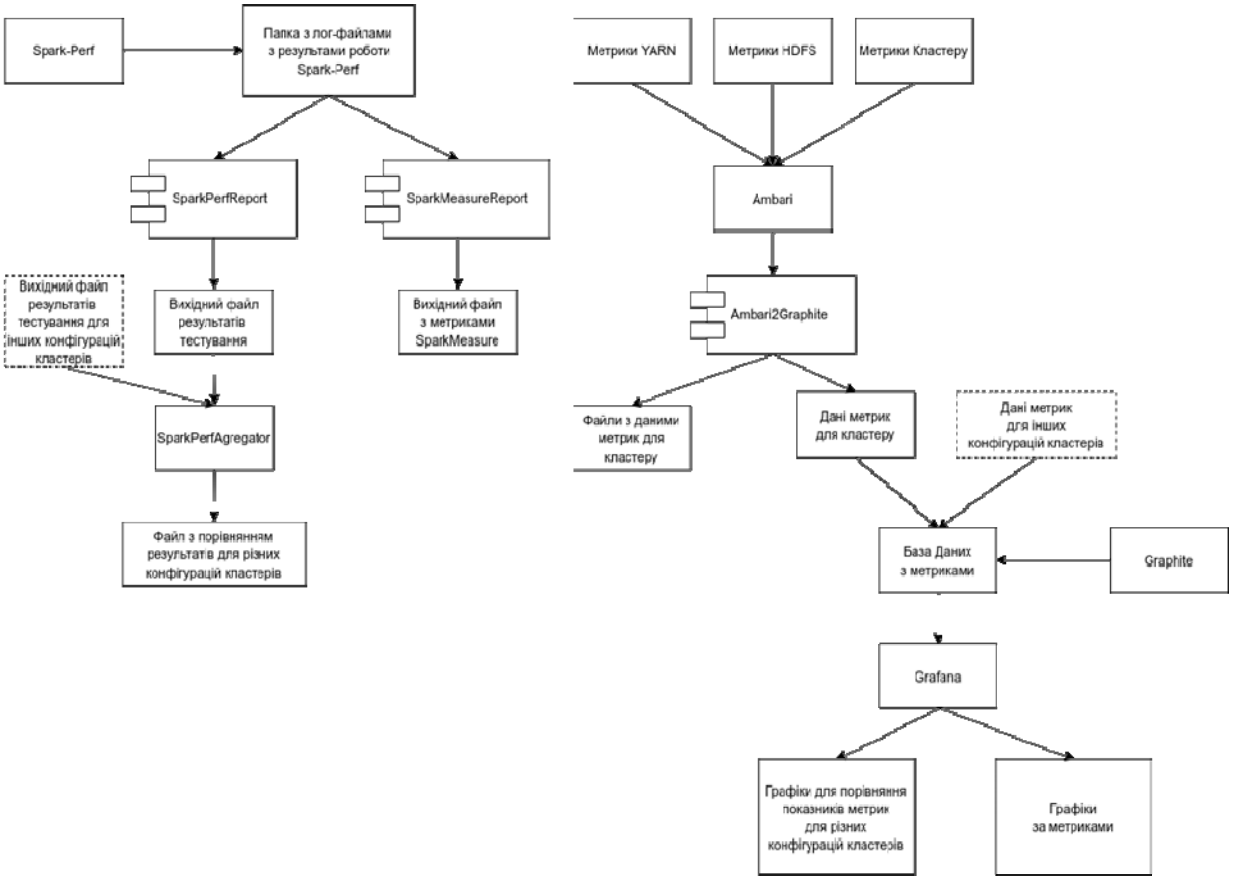

Рис. 3. Схема зв'язків програмного забезпечення для тестування Spark-Perf

Для аналізу результатів, які отримуються у результаті роботи бенчмарку Spark-Perf, та моніторингу ресурсів Apache Spark розроблено програмне забезпечення, структура якого наведена на рис. 3.

Загальний звіт результатів представляє собою текстовий файл, в якому записані найбільш значущі данні про кожен тест. До нього входять назва тесту, параметри запуску, та результати тестування. В папці з логами зберігаються по два файли для кожного тесту з розширеннями ".out" та ".err". В першому з них знаходиться детальна інформація про виконаний тест, а в файл з розширенням ".err" записується інформація про помилки при виконанні тесту, а також вся службова інформація, яка може стати корисною в разі помилки.

Модуль SparkPerfReport. Для збору та обробки результатів тестування, що записані у об'єкті результату, розроблено на мові Python 3.7 програмний модуль Spark-PerfReport. Вхідними даними для програми є шлях до папки з результатами тестування. Далі модуль ітеративно йде по внутрішнім папкам та шукає файли з розширенням .out, розбирає файл, збирає відповідну інформацію та формує звіт у вигляді excel-документу. У модулі SparkPerfReport до звіту виводиться статистичний аналіз отриманих результатів для кожного тесту, а саме середнє арифметичне значення, медіана та середньоквадратичне відхилення для процесів тестування та навчання, а також відношення середнього часу тестування до середнього часу навчання.

Модуль SparkMeasureReport. Формує звіт за результатами роботи системи SparkMeasure. Вхідні параметри у модулі такі ж, як і у модуля SparkPerfReport. Поля метрик, які буде створювати модуль, визначені в окремому файлі, а структура таблиці для бази даних, у яку зберігаються отримані дані, має інший вигляд. Для тесту є декілька метрик - поле з назвою метрики, яке винесене в окреме по-

ле. У звіті модулі SparkMeasureReport формується зведена таблиця з метриками для всіх проведених тестів, а також графіки, побудовані на її основі.

Модуль AmbariMetricsReport. Apache Ambari - це інструмент адміністрування з відкритим вихідним кодом, що розгортається на кластерах Hadoop, і використовується для відстеження запущених програм та їх стану. Apache Ambari - це веб-додаток, який управляє, контролює та забезпечує гараноздатність кластерів Hadoop. Він забезпечує високоінтерактивну інформаційну панель, яка дозволяє візуалізувати хід та стан виконання кожної програми, що працює над кластером Hadoop. Його гнучкий і масштабований користувальницький інтерфейс дозволяє встановлювати на кластер цілу низку інструментів, таких як Pig, MapReduce, Hive тощо. Сервіс моніторингу Ambari надає можливість отримувати метрики роботи кластеру за певний проміжок часу в форматі CSV. Для збору та аналізу метрик для моніторингу ресурсів та завдань кластеру використаний комплекс програмних модулів, який включає модулі Ambari [12], Grafana [13], Graphite, Ambari2Grafana (рис. 3). Graphite – це безкоштовний інструмент з відкритим кодом, який здійснює моніторинг та візуалізацію числових даних часових рядів, таких як продуктивність комп'ютерних систем. Grafana – це безкоштовне програмне забезпечення, що дозволяє візуалізувати та форматувати метричні дані, яке розповсюджується за ліцензією Apache 2.0.2. Це дозволяє створювати інформаційні панелі та графіки з різних джерел, включаючи бази даних часових рядів, такі як Graphite, InfluxDB та OpenTSDB. Для збору інформації з Ambari у базу даних Graphite використовується програма Ambari2Graphite, яка була розроблена на мові Python3.7. Вхідними даними для програми є файл налаштування, який містить доступи до Ambari та Graphite, дату початку моніторингу та дата початку тестування. Модуль використовує API Ambari для завантаження значень за наданими метриками.

Експериментальні дослідження продуктивності кластеру Apache Spark в Azure HDInsight

Проведення експериментальних досліджень виконувалося на основі наступних кроків.

Крок 1. Вибір кластеру Apache Spark в Azure HDInsight. Існує декілька шаблонів для розгортання кластеру Spark у Azure. Для того, щоб побачити всі з них, потрібно ввести у поле пошуку шаблонів слово “Spark” та обрати шаблон “101-hdinsight-spark-linux” та налаштувати параметри: headNode.targetInstanceCount – кількість master-вузлів; headNode.hardwareProfile.vmSize – обрана конфігурація розміру віртуальних машин для master-вузлів; workerNode.targetInstanceCount – кількість worker-вузлів; workerNode.hardwareProfile.vmSize – обрана конфігурація розміру віртуальних машин (примірників) для worker-вузлів.

Крок 2. Розгортання та налаштування кластера на основі Configs. Налаштовуються значення параметрів spark.driver.memoryOverhead; spark.executor.cores; spark.executor.instances; spark.executor.memory; executor.memoryOverhead, ступінь паралелізму.

Крок 3. Вибір архітектури кластерів Apache Spark. Для тестування були обрані конфігурації кластерів з гомогенною та гетерогенною архітектурою, наведені в табл. 2.

Таблиця 2

Конфігурації кластерів

Тип кластеру	Master-вузли		Worker-вузли	
	Кількість, шт	Конфігурація розміру	Кількість, шт	Конфігурація розміру
Гомогенний	2	Standard_D3_v2	2	Standard_D3_v2
	2	Standard_D3_v2	3	Standard_D3_v2
Гетерогенний	2	Standard_D3_v2	2	Standard_D4_v2
	2	Standard_D3_v2	3	Standard_D4_v2

Крок 4. Налаштування середовища для тестових випробувань. Тестові установки для гомогенних кластерів наведені в табл. 3, для гетерогенних кластерів – в табл. 4.

Таблиця 3

Налаштування гомогенних кластерів

Назва характеристики	Значення характеристики для кластеру	
	2 обчисл. вузли	3 обчисл. вузли
Конфігурація master -вузлів	2 x Standard_D3_v2	2 x Standard_D3_v2
Конфігурація worker -вузлів	2 x Standard_D3_v2	3 x Standard_D3_v2
Планувальник ресурсів	Capacity Scheduler	Capacity Scheduler
Ступінь паралелізму	32	40
Кількість вузлів	4	5
Застосовано ядер	8 + 8 = 16	8 + 12 = 20
Застосовано оперативної пам’яті	28 + 28 = 56	28 + 42 = 70
Процесор на вузлах	4	4
Оперативна пам’ять на вузлах	14	14

Таблиця 4

Налаштування гетерогенних кластерів

Назва характеристики	Значення характеристики для кластеру	
	2 обчисл. вузли	3 обчисл. вузли
Конфігурація master-вузлів	2 x Standard_D3_v2	2 x Standard_D3_v2
Конфігурація worker-вузлів	2 x Standard_D4_v2	3 x Standard_D4_v2
Планувальник ресурсів	Capacity Sched-uler	Capacity Sched-uler
Ступінь паралелізму	48	64
Кількість вузлів	4	5
Застосовано ядер	8 + 16 = 24	8 + 24 = 32
Застосовано опе-ративної пам'яті	28 + 56 = 84	28 + 84 = 112
Процесор на вузлах	8	8
Оперативна пам'ять на вузлах	28	28

У файлі "config.py" налаштовується конфігурація для тестування шляхом визначення наступних опцій:

SPARK_CLUSTER_URL – URL кластеру Spark, який буде використаний при запуску задач. Має бути задано значення "yarn", оскільки замовчуванням кластери Spark на Azure HDInsight використовують цей планувальник;

секції "RUN" та "PREP", в яких можна вказати які тести треба запустити та які треба підготовляти. Процес підготовки складається з завантаження third-party пакетів і компіляції коду (тільки для Scala).

Доступні такі набори тестів:

SPARK_TESTS – набір тестів Spark Core на Scala;

PYSPARK_TESTS – набір тестів Spark Core на Python;

STREAMING_TESTS – набір тестів Spark Streaming на Scala;

MLLIB_TESTS – набір тестів Spark MlLib на Scala;

PYTHON_MLLIB_TESTS – набір тестів Spark MlLib на Python.

В нашому випадку вмикаємо лише тести SPARK_TESTS та MLLIB_TESTS

Крок 4. Проведення тестувальних випробувань.

В процесі тестування було використано тестові набори "Дерево рішень" (Decision Trees, DR), які включали 14 різних методів з бібліотеки машинного навчання.

Результатом є порівняння середнього часу навчання та середнього часу тестування для різних конфігурацій кластерів (рис. 4–7).

Рис. 4. Порівняльний аналіз часу навчання на гомогенних кластерах з 2 та 3 обчислювальними вузлами (workers)

Рис. 5. Порівняльний аналіз часу навчання на гетерогенних кластерах з 2 та 3 обчислювальними вузлами (workers)

Рис. 6. Порівняльний аналіз часу тестування на гомогенних кластерах з 2 та 3 обчислювальними вузлами (workers)

Рис. 7. Порівняльний аналіз часу тестування на гетерогенних кластерах з 2 та 3 обчислювальними вузлами (workers)

Із залежностей часу навчання та тестування можна зробити висновок, що продуктивність кластерів з 3 вузлами більша, ніж для кластерів з 2 вузлами, але для більш ефективного розподілу навантаження між робочими вузлами потрібне більш ретельніше налаштування параметрів щодо ступені паралелізму відповідно до існуючих рекомендацій щодо вибору їх значень в координації з іншими параметрами кластеру, які можна налаштовувати як в інтерфейсі Ambari, так і вручну.

Висновки

Розглянуті питання щодо підвищення продуктивності розв'язку задач машинного навчання з застосуванням високопродуктивних розподілених обчислень на базі хмарної платформи Microsoft Azure довели досить великі можливості цієї платформи. Для підвищення продуктивності розв'язку різних задач машинного навчання потрібне підключення, налаштування та розгортання обчислювального кластера, що потребує виконання довільно трудомістських завдань. Важливим при використанні розподілених режимів роботи розглянутих кластерів є вибір величин параметрів налаштувань, що можна зробити за допомогою існуючих засобів, та вручну.

Існуючі рекомендації щодо оптимізації або підвищення продуктивності кластерів, зокрема Apache Spark в Azure HDInsjght, не завжди дозволяють зробити це з певною гарантованою ймовірністю внаслідок того, що кількість параметрів, які можна налаштовувати, сягає більш ніж 100 параметрів.

Багато з них пов'язані між собою, в значній мірі опосередковано, та отже цей зв'язок повинен бути визначений, для чого вже задіяні такі сучасні інструменти, як наприклад, моделі машинного навчання. Вони дозволяють створювати прогнозні моделі, але й при цьому бажане яка можна "тонке" налаштування величин та вибору найбільш вагомих з точки зору особливостей розв'язувальних задач параметрів.

В подальшому дослідження будуть спрямовані на дослідження впливу масштабованості та гетерогенності кластерів на час навчання та тестування методів та моделей машинного навчання.

Список літератури

1. The official site of SPARK.APACHE. Apache Spark™ is a unified analytics engine for large-scale data processing. – Available at: https://spark.apache.org/.

2. The official site of APACHE.ORG. RDD programming guide. – Available at: https://spark.apache.org/docs/latest/rdd-programming-guide.html.

3. The official site of FLAIR.TRAINING. Apache Spark Ecosystem and Spark Components. – Available at: https://data-flair.training/blogs/apache-spark-ecosystem-components/.

4. The official site of ANALYTICSINDIAMAG.COM. What is machine learning as a service (MLAAS). – Available at: https://analyticsindiamag.com/what-is-machine-learning-as-a-service-mlaas/.

5. The official site of MICROSOFT.COM. Azure HDInsight documentation. – Available at: https://docs.microsoft.com/en-us/azure/hdinsight/.

6. The official site of AZURE.MICROSOFT. Azure HDInsight pricing. – Available at: https://azure.microsoft.com/en-us/pricing/details/hdinsight/.

7. The official site of DATABRICKS.COM. Apache Spark MLlib – Databricks Documentation. – Available at: https://docs.databricks.com/applications/machine-learning/mllib/index.html.

8. The official site of IBM.COM. Benchmarking Spark. – Available at: https://www.ibm.com/support/knowledgecenter/en/STXKQY_BDA_SHR/bl1bda_benchmarkspark.htm.

9. A Comprehensive Benchmarking Suite For In Memory [Electronic resource] / M. Li, J. Tan, Y. Wang, L. Zhang // Cluster Computing. – 2017. – № 20. – Available at: http://surl.li/bdep.

10. The official site of GITHUB.COM. HiBench. – Available at: https://github.com/Intel-bigdata/HiBench.

11. The official site of GITHUB.COM. Spark-Perf - Performance tests for Apache Spark. – Available at: https://github.com/databricks/spark-perf.

12. The official site of AMBARI.APACHE. Apache Ambari. – Available at: https://ambari.apache.org/.

13. The official site of GRAFANA.COM. Grafana: The open observability platform. – Available at: https://grafana.com/.

References

1. The official site of SPARK.APACHE. *Apache Spark™ is a unified analytics engine for large-scale data processing*, available at: www.spark.apache.org/.

2. The official site of APACHE.ORG. *RDD programming guide*, available at: www.spark.apache.org/docs/latest/rdd-programming-guide.html.

3. The official site of FLAIR.TRAINING. *Apache Spark Ecosystem and Spark Components*, available at: www.data-flair.training/blogs/apache-spark-ecosystem-components/.

4. The official site of ANALYTICSINDIAMAG.COM. *What is machine learning as a service (MLAAS)*, available at: www.analyticsindiamag.com/what-is-machine-learning-as-a-service-mlaas/.

5. The official site of MICROSOFT.COM. *Azure HDInsight documentation*, available at: www.docs.microsoft.com/en-us/azure/hdinsight/.

6. The official site of AZURE.MICROSOFT. *Azure HDInsight pricing*, available at: www.azure.microsoft.com/en-us/pricing/details/hdinsight/.

7. The official site of DATABRICKS.COM. *Apache Spark MLlib – Databricks Documentation*, available at: www.docs.databricks.com/applications/machine-learning/mllib/index.html.

8. The official site of IBM.COM. *Benchmarking Spark*, available at: www.ibm.com/support/knowledgecenter/en/STXKQY_BDA_SHR/bl1bda_benchmarkspark.htm.

9. Li, M., Tan, J., Wang, Y. and Zhang, L. (2017), A Comprehensive Benchmarking Suite For In Memory, *Cluster Computing*, No. 20, available at: www.surl.li/bdep.

10. The official site of GITHUB.COM. *HiBench*, available at: www.github.com/Intel-bigdata/HiBench.

11. The official site of GITHUB.COM. *Spark-Perf – Performance tests for Apache Spark*, available at: www.github.com/databricks/spark-perf.

12. The official site of AMBARI.APACHE. *Apache Ambari*, available at: www.ambari.apache.org/.

13. The official site of GRAFANA.COM. *Grafana: The open observability platform*, available at: www.grafana.com/.

Надійшла до редколегії 07.02.2020
Схвалена до друку 10.03.2020

Відомості про автора:

Мінухін Сергій Володимирович
доктор технічних наук доцент
професор Харківського національного
економічного університету ім. С. Кузнеця,
Харків, Україна
https://orcid.org/0000-0002-9314-3750

Information about the author:

Sergii Minukhin
Doctor of Technical Sciences Associate Professor
Professor of Simon Kuznets Kharkiv National
University of Economics,
Kharkiv, Ukraine
https://orcid.org/0000-0002-9314-3750

ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ КЛАСТЕРА APACHE SPARK НА ПЛАТФОРМЕ AZURE ДЛЯ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

С.В. Минухин

Исследованы вопросы повышения производительности применения моделей и методов задач машинного обучения с использованием Apache Spark в Azure HDInsight. Для повышения обоснованности полученных результатов использованы один из самых известных бенчмарков для тестирования библиотеки машинного обучения Spark-Perf. Приведены шаги установления, развертывания и настройки Apache Spark на платформе Azure. Для оценки эффективности распределенных вычислений использованы метрики производительности среднего времени обучения и тестирования и их отношения. Проведенный сравнительный анализ результатов решения задач из библиотеки MLlib для кластеров с гомогенной и гетерогенной архитектурой, которые свидетельствуют о высокой эффективности их использования.

Ключевые слова: *модель, машинное обучение, Apache Spark, Azure HDInsight, Spark-Perf, MLlib, деревья решений, обучение, тестирование.*

PERFORMANCE RESEARCH OF APACHE SPARK CLUSTER ON AZURE PLATFORM FOR MACHINE LEARNING METHODS

S. Minukhin

An analysis is made of the use of distributed computing capabilities to solve machine learning tasks. The issues of increasing the productivity of application of models and methods of machine learning tasks using Apache Spark in Azure HDInsight are explored. To increase the validity of the results and evaluate the quality of Apache Spark fireworks, a brief analysis was conducted and one of the most famous benchmarks was used to test Spark-Perf machine learning libraries. A feature of the selected test data is the widespread representation of machine learning models, in particular decision trees mehtods. The steps for installing, deploying, and configuring Apache Spark Parameters on the Azure platform. To evaluate the results of the calculations, a set of software models was developed to work with Spark, Ambari and monitoring system SparkMeasure. A scheme of interaction of software models for obtaining and processing data from various sources is developed. The MLlib Machine Learning Library was selected to test the models. Cluster architectures with homogeneous and heterogeneous architecture have been configured, for which parameter values have been selected, which allow to increase the productivity of solving machine learning tasks due to fine tuning (tuning). Developed a set of software modules to receive and process data directly from Spark log files, metrics obtained by the Sparkmeasue monitoring system, Ambari system, Grafana, Graphite and Ambari2Grafana together providing system administrators for making effective economic decisions on cluster management. Metrics using training and test report data are offered to evaluate the performance of the calculations - average training time and average test time, as well as their relationship. The results and their analysis of the comparative analysis of the obtained results of the solution of some problems of the MLlib library for clusters with homogeneous and heterogeneous architecture, which indicate the high efficiency of their use, are presented.

Keywords: *model, machine learning, Apache Spark, Azure HDInsight, Spark-Perf, MLlib, decision trees, training, testing.*