Apache Spark: Mesin Pemrosesan Data Besar

Eman Syaikh
Sekolah Tinggi Teknik dan Sains Komputer,
Universitas Pangeran Mohammad Bin Fahd,
Khobar, Arab Saudi.
emanshaikh26@gmail.com

Iman Mohiuddin
Sekolah Tinggi Teknik dan Sains Komputer,
Universitas Pangeran Mohammad Bin Fahd,
Khobar, Arab Saudi.
iman28198@gmail.com

Yasmeen Alufaisan
Sekolah Tinggi Teknik dan Sains Komputer,
Universitas Pangeran Mohammad Bin Fahd,
Khobar, Arab Saudi.
yalufaisan@pmu.edu.sa

Irum Nahvi
Sekolah Tinggi Teknik dan Sains Komputer,
Universitas Pangeran Mohammad Bin Fahd,
Khobar, Arab Saudi.
inahvi@pmu.edu.sa

Abstrak—Analisis data besar telah mempengaruhi pasar industri. Hal ini memiliki dampak yang signifikan pada kumpulan data yang besar dan beragam untuk menunjukkan pola tersembunyi dan pengungkapan lainnya. Apache Hadoop, Apache Flink dan Apache Storm adalah beberapa kerangka kerja yang umum digunakan untuk analisis data besar. Apache Spark adalah mesin analisis data besar yang terkonsolidasi dan menyediakan paralelisme data absolut. Makalah ini membahas tinjauan teknis tentang analisis data besar menggunakan Apache Spark dan bagaimana ia menggunakan komputasi dalam memori yang membuatnya jauh lebih cepat dibandingkan dengan kerangka kerja terkait lainnya. Selain itu, Spark juga menyediakan kemampuan pemrosesan batch dan pemrosesan aliran yang luar biasa. Selain itu, juga dibahas tentang kemampuan multithreading dan konkurensi Apache Spark. Inti dari konvergensi adalah arsitektur, persyaratan perangkat keras, ekosistem, kasus penggunaan, fitur Apache Spark, dan penggunaan Spark dalam teknologi baru.

Ketentuan Indeks—Analisis Big Data, Pembelajaran Mesin, Pemrosesan Aliran, Kumpulan Data Terdistribusi yang Tangguh.

I. PENDAHULUAN

Riset analisis data besar telah memberikan pengaruh besar pada pasar industri. Ini adalah strategi mengambil data dalam jumlah besar dari berbagai sumber, mengorganisasikan data yang sama secara lengkap dan kemudian menganalisis kumpulan data yang besar tersebut untuk menemukan fakta dan angka yang bermakna dari data yang dikumpulkan. Sebagai hasil dari agregasi data yang heterogen, banyak organisasi yang menyediakan layanan web seperti Amazon dan Microsoft menggunakan cluster server komoditas.

Istilah big data digunakan untuk merujuk pada sejumlah besar kumpulan data kompleks yang dikembangkan dari sumber data baru dan diperiksa untuk perangkat lunak aplikasi pemrosesan data tradisional. Big data digunakan untuk strategi dan teknologi non-konvensional yang memerlukan pengumpulan wawasan dari kumpulan data besar serta pengorganisasian dan pemrosesan data tersebut. Singkatnya, analisis big data adalah proses menemukan pengetahuan dari sebagian besar data. Oleh karena itu, untuk menganalisis data sebesar itu, perlu menggunakan semacam alat analisis atau kerangka pemrosesan.

Kerangka pemrosesan dipandang sebagai salah satu konstituen terpenting dari sistem data besar. Kerangka pemrosesan menentukan data dalam sistem, baik dengan mengonsumsinya data ke dalam sistem apa adanya atau dengan membaca data dari penyimpanan nonvolatile. Kerangka kerja pemrosesan dikategorikan berdasarkan jenis dan kondisi data yang dirancang untuk dioperasikan. Jika beberapa sistem menangani data dalam batch, sistem lain menangani data dalam aliran yang tidak terputus saat data tersebut berpindah ke dalam sistem. Ada beberapa sistem yang juga dapat menangani data dengan kedua cara [1]. Secara umum, kerangka kerja pemrosesan data besar dapat dibagi menjadi tiga kategori berikut: kerangka kerja batch saja, kerangka kerja streaming saja, dan kerangka kerja hybrid.

Sistem pemrosesan batch mengumpulkan semua data ke dalam kelompok yang kemudian disimpan dan diproses di masa mendatang. Di sisi lain, sistem pemrosesan aliran memproses data segera setelah data tiba, yaitu pemrosesan waktu nyata. Dalam sistem pemrosesan hibrid, beban kerja batch dan streaming dapat ditangani. Hal ini menghasilkan pemrosesan data yang lebih sederhana dan umum karena kami dapat menerapkan fitur atau API yang sama untuk data batch dan streaming.

Apache Spark adalah mesin analitik terkonsolidasi yang substansial untuk pemrosesan data terdistribusi yang komprehensif dan beban kerja pembelajaran mesin. Ini telah membentuk domain yang luas untuk memecahkan masalah ilmu data dan teknik menggunakan bahasa pemrograman seperti Python. Apache Spark memperkuat teknik seperti pemrosesan dalam memori, aliran, dan pemrosesan batch beban kerja data besar. Teknik-teknik ini akan dibahas lebih lanjut pada Bagian III. Apache Spark dengan cepat digunakan oleh berbagai industri yang tak terbatas. Ini bukan hanya proyek aktif di Apache Software Foundation tetapi juga proyek sumber terbuka yang diterima secara luas. Tindakan mengumpulkan, memproses, dan menyimpan data dalam jumlah besar disebut data besar.

Dalam tulisan ini, kami fokus secara khusus pada kerangka pemrosesan data besar Apache Spark. Kami mendiskusikan kemampuan pemrosesan batch dan aliran Spark. Kami menjelaskan lebih lanjut berbagai fitur yang menjadikan Spark kerangka kerja unik. Kami juga membahas beberapa kasus penggunaan utama Apache Spark. Setelah itu, kami meninjau persyaratan ekosistem, arsitektur, dan perangkat keras Spark. Terakhir, kami membahas kemampuan multi-threading dan konkurensi Spark serta penggunaan Spark dalam teknologi baru.

Makalah yang tersisa disusun sebagai berikut: Bagian II

978-1-7281-3687-5/19/$31.00 ©2019 IEEE

memperkenalkan karya literatur yang terkait dengan makalah ini, Bagian III membahas Apache Spark, dan Bagian IV menyimpulkan makalah.

II. PEKERJAAN YANG BERHUBUNGAN

Dalam [2], penulis memulai proyek Apache Spark yang menyediakan mesin analitik terintegrasi untuk berbagai pemrosesan data yang disebarluaskan. Spark memungkinkan pemrograman seluruh cluster secara paralel. Ia memperluas modelnya ke struktur data dasar yang disebut Kumpulan Data Terdistribusi Tangguh (RDD), meskipun memiliki model pemrograman yang sama dengan MapReduce. Spark adalah sistem pemrosesan informasi terdepan untuk SQL komprehensif, pemrosesan aliran, pemrosesan grafik, dan pembelajaran mesin. Oleh karena itu, model Apache Spark dapat secara efektif membantu beban kerja saat ini dan memberikan banyak manfaat bagi pengguna.

Salloum dkk. [3] berfokus pada komponen utama dan karakteristik khas Apache Spark dalam analisis data besar. Beberapa fungsi heterogen untuk desain dan implementasi dihasilkan oleh Apache Spark, yang terdiri dari API pipeline pembelajaran mesin. Apache Spark adalah kerangka kerja komputasi cluster yang tersebar luas dan populer tidak hanya di komunitas akademis tetapi juga di pasar industri. Oleh karena itu, makalah ini mengarahkan banyak perhatian terhadap penelitian dan pertumbuhan yang terkait dengan Apache Spark dalam analisis data besar.

Dalam [4], penulis mengusulkan solusi untuk mengatasi konfrontasi signifikan yang dihadapi selama analisis data besar. Dalam pekerjaannya, mereka menggunakan framework Apache Storm dengan sampel data Twitter. Apache Storm berhasil mengatasi tantangan tersebut dan membuktikan bahwa ia dapat memproses streaming real-time dengan latensi yang sangat rendah.

Dalam [5], penulis mengembangkan jalur baru untuk pencitraan resonansi magnetik fungsional (fMRI) menggunakan PySpark pada satu node. PySpark adalah bahasa untuk analisis data dan saluran pipa, yang memaparkan model pemrograman Spark ke Python. Dalam saluran ini, jaringan otak dipisahkan dari data fMRI dengan pencocokan templat dan metode jumlah perbedaan kuadrat (SSD). Dalam hal waktu pemrosesan, pipeline ini empat kali lebih cepat dibandingkan yang dikembangkan dengan Python. Ini telah meningkatkan pemrosesan data dalam memori secara paralel, mengonversi data menjadi Kumpulan Data Terdistribusi Tangguh, dan menyimpan hasilnya dalam format lain seperti bingkai data.

Gopalani dkk. [6] terutama menyajikan perbandingan antara kerangka Apache Hadoops Map Reduce dan Apache Spark karena kedua opsi tersebut digunakan dalam analisis data besar. Selanjutnya makalah ini membandingkan kedua framework pada berbagai parameter serta memberikan analisis performa terhadap keduanya menggunakan algoritma K-Means dan disimpulkan bahwa framework Apache Spark akan membawa perubahan besar dalam dunia big data karena kemampuannya. pemrosesan dalam memori.

Pada [7], penulis melakukan studi perbandingan Apache Spark dan Apache Flink. Secara khusus, makalah ini berfokus pada membandingkan pustaka pembelajaran mesin dalam kerangka kerja ini untuk pemrosesan data batch. Algoritma machine learning yang digunakan pada penelitian ini adalah Support Vector Machines dan Linear Regresi. Makalah ini menunjukkan dengan hasil empiris bahwa Spark mengungguli Flink dalam hal kinerja.

Dari perspektif kegunaan dan kemudahan penggunaan, platform berorientasi aliran data terdistribusi - Apache Hadoop MapReduce, Apache Spark dan Apache Flink dibandingkan pada [8]. MapReduce menjawab tantangan seperti skalabilitas dan redundansi bawaan, sedangkan dua tantangan terakhir berfokus pada kebutuhan aliran data yang efisien, cache data, dan operator pemrosesan data deklaratif. Tujuan utamanya adalah untuk memberikan jalan untuk memilih platform yang sesuai dan memberikan pemahaman yang lebih baik tentang fungsionalitas sistem pemrosesan data besar.

Makalah kami berbeda dengan penelitian sebelumnya karena mengulas Apache Spark dari berbagai aspek. Kami fokus pada fitur-fitur utama Sparks, kemampuan pemrosesan batch dan pemrosesan aliran, kasus penggunaan, ekosistem, arsitektur, kemampuan multi-threading dan konkurensi. Terakhir kami juga menyebutkan bagaimana Spark telah menjadi alat vital yang digunakan dalam teknologi baru saat ini.

AKU AKU AKU. APACHE SPARK

Apache Spark adalah platform pemrosesan data besar yang kuat yang mengadaptasi kerangka kerja hybrid. Kerangka kerja hibrid menawarkan dukungan untuk kemampuan pemrosesan batch dan streaming. Meskipun Spark menggunakan banyak prinsip yang mirip dengan mesin Hadoops MapReduce, Spark mengungguli mesin Hadoops MapReduce dalam hal kinerja. Misalnya, dengan beban kerja pemrosesan batch yang sama, Spark bisa lebih cepat daripada MapReduce karena fitur “perhitungan dalam memori penuh” yang digunakan oleh Spark dibandingkan dengan pembacaan dan penulisan tradisional ke disk yang digunakan oleh MapReduce. Spark dapat berjalan dalam mode mandiri atau dapat dikombinasikan dengan Hadoop untuk menggantikan mesin MapReduce.

1) Model Pemrosesan Batch Percikan: Keuntungan terkuat Spark dibandingkan MapReduce adalah komputasi dalam memori. Spark berinteraksi dengan disk hanya untuk dua tugas: memuat data awalnya ke dalam memori dan menyimpan hasil akhir kembali ke memori. Semua hasil lain di antaranya diproses dalam memori. Pemrosesan dalam memori ini membuat Spark jauh lebih cepat dibandingkan kerangka pemrosesan batch kompetitifnya, Hadoop. Selain itu, pengoptimalan holistik yang digunakan oleh Spark berkontribusi lebih jauh pada kecepatan tingginya di mana serangkaian tugas lengkap dapat dianalisis sebelumnya. Hal ini dicapai dengan menghasilkan Directed Acyclic Graphs (DAGs) yang digunakan untuk mewakili semua operasi, data, dan hubungan di antara keduanya [1]. Untuk mendukung fitur komputasi dalam memori, Spark menggunakan Kumpulan Data Terdistribusi Tangguh (RDD). RDD adalah struktur data hanya-baca yang disimpan dalam memori untuk menjadikan Spark kerangka kerja toleransi kesalahan tanpa harus menulis ke disk setelah setiap operasi.

2) Model Pemrosesan Aliran Percikan : Selain pemrosesan batch, Spark menyediakan kemampuan pemrosesan aliran dengan penggunaan batch mikro. Dalam batch mikro, aliran data diperlakukan sebagai sekelompok batch yang sangat kecil yang kemudian ditangani sebagai tugas rutin oleh mesin batch Spark [1]. Meskipun prosedur micro-batching ini berfungsi dengan baik, prosedur ini masih dapat menyebabkan beberapa perbedaan dalam hal kinerja dibandingkan dengan kerangka pemrosesan aliran yang sebenarnya.

A.FITUR

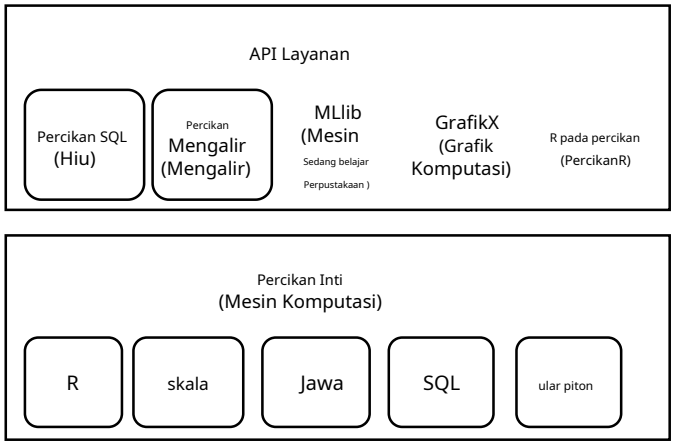
Apache Spark memiliki banyak fitur yang dapat dibedakan. Berikut penjelasan beberapa fitur tersebut:

- *Kecepatan:*Apache Spark adalah alat yang dapat digunakan untuk menjalankan aplikasi Spark di cluster Apache Hadoop. Apache Spark seratus kali lebih cepat dibandingkan Apache Hadoop dan sepuluh kali lebih cepat dibandingkan mengakses data dari disk. Spark memanfaatkan gagasan Resilient Distributed Dataset (RDD), dan memungkinkannya menyimpan data secara jelas di dalam memori [9].
- *Kegunaan:*Spark memungkinkan pengguna dengan cepat menulis aplikasi dalam berbagai bahasa pemrograman seperti Java, Scala, R dan Python. Hal ini membantu pemrogram untuk mengembangkan dan menjalankan aplikasi mereka dalam bahasa yang mereka kenal sehingga memudahkan pengembangan aplikasi paralel
- *Analisis tingkat lanjut:*Selain operasi peta dan pengurangan yang sederhana, Sparks menyukai kueri SQL, streaming data, dan analitik rumit lainnya seperti pembelajaran mesin, dan algoritma grafik
- *Berjalan di mana-mana:*Apache Spark dapat dijalankan di berbagai platform seperti Apache Hadoop YARN, Mesos, EC2, Kubernetes atau di cloud menggunakan mode cluster mandiri Apache Spark. Itu dapat mengambil beberapa informasi data yaitu HDFS, Cassandra, HBase dll.
- *Komputasi dalam memori:*Komputasi klaster dalam memori memungkinkan Spark menjalankan algoritme pembelajaran mesin berulang dan membantu kueri bilateral dan analisis streaming data dengan kecepatan super cepat. Spark menyimpan data di RAM server sehingga data yang disimpan dapat diakses dengan cepat
- *Pemrosesan streaming waktu nyata:*Spark streaming memahami pemrosesan aliran real-time bersama dengan konfigurasi lain yang menyimpulkan bahwa spark streaming itu sederhana, toleran terhadap kesalahan, dan tidak terpisah [9].

B. KASUS PENGGUNAAN APACHE SPARK

- *Kesehatan:*Spark digunakan di sektor perawatan kesehatan karena menyediakan analisis menyeluruh terhadap catatan pasien bersama dengan data medis sebelumnya. Hal ini membantu untuk mengidentifikasi pasien mana yang rentan menghadapi komplikasi kesehatan dalam waktu dekat dan oleh karena itu menghindari rawat inap kembali di rumah sakit sehingga mengurangi biaya bagi rumah sakit dan pasien karena sekarang dimungkinkan untuk menerapkan layanan rumah bagi pasien yang teridentifikasi [10]. Selain itu, Spark juga digunakan dalam pengurutan genom karena dapat mengurangi waktu pemrosesan yang diperlukan untuk memproses data genom yang sebelumnya memerlukan waktu beberapa minggu untuk mengatur semua senyawa kimia dengan gen. MyFitnessPal adalah perusahaan yang menggunakan Spark [11].
- *Keuangan:*Apache Spark memberikan wawasan yang membantu membuat pilihan yang tepat atas berbagai masalah seperti segmentasi pelanggan, penilaian risiko kredit, dan periklanan bertarget [10]. Lembaga keuangan sering kali menggunakan data besar untuk mengetahui waktu yang tepat frauha d oksaat ini D,jadi th di model sudah ada dengan transaksi dan mayoritas dan lokasi kapan hal itu dapat dihentikan. Berbagai cara digunakan untuk mendeteksi palsu mereka dikerahkan secara batch lingkungan. Dengan bantuan Apache Spark di Hadoop, lembaga keuangan dapat mendeteksi transaksi palsu secara realtime, berdasarkan jejak penipuan sebelumnya [11].
- *Perdagangan elektronik:*Spark digunakan dalam industri e-commerce untuk mencari informasi mengenai transaksi real-time yang diteruskan ke algoritma pengelompokan streaming seperti algoritma pengelompokan Kmeans atau kuadrat terkecil bergantian. Ini juga meningkatkan rekomendasi kepada pelanggan berdasarkan tren terkini. Alibaba dan eBay adalah contoh perusahaan yang menggunakan Spark dalam e-commerce.
- *Hiburan:*Dalam industri game, Apache Spark membantu mengenali pola peristiwa dalam game secara real-time dan kemudian meresponsnya untuk menghasilkan peluang bisnis yang bermanfaat seperti iklan selektif, retensi pemain, atau perubahan otomatis level game berdasarkan tingkat kesulitannya. Selain itu, Spark yang dikombinasikan dengan MongoDB juga digunakan di situs berbagi video seperti Pinterest, Netflix, dan Yahoo. Situs web ini menampilkan iklan terkait kepada penggunanya berdasarkan video yang dilihat, dibagikan, dan dijelajahi oleh pengguna [10].

C.EKOSISITEM



Gambar 1. Ekosistem Apache Spark

Ekosistem Apache Spark terdiri dari komponen utama berikut:

- *Percikan SQL.*Sebelumnya dikenal sebagai Hiu. Spark SQL adalah kerangka kerja terdistribusi yang bekerja dengan data terstruktur dan semi-terstruktur. Ini memfasilitasi aplikasi analitis dan interaktif untuk streaming dan data historis yang dapat diakses dari berbagai sumber seperti tabel JSON, Parket dan Hive [12].
- *Percikan Streaming:*Ini memungkinkan pengguna untuk memproses streaming data secara real time. Untuk melakukan analisis streaming, streaming Spark meningkatkan kemampuan penjadwalan cepat Apache Spark dengan memasukkan data ke dalam batch mini. Operasi yang dikenal sebagai transformasi kemudian diterapkan pada batch mini yang dapat diperoleh dengan mudah dari streaming langsung dan sumber data seperti Twitter, Apache Kafka, sensor IoT, dan Amazon Kinesis.

- *MLlib:*Ini memberikan algoritme berkualitas tinggi dengan kecepatan tinggi dan membuat pembelajaran mesin mudah digunakan dan diskalakan. Beberapa algoritma pembelajaran mesin seperti regresi, klasifikasi, pengelompokan, dan aljabar linier hadir. Ini juga menyediakan perpustakaan untuk pembelajaran mesin primitif tingkat rendah seperti algoritma optimasi penurunan gradien umum. Ini juga menyediakan fungsi lain seperti evaluasi model dan impor data. Ini dapat digunakan di Java, Scala, dan Python.
- *GrafikX:*Ini adalah mesin komputasi grafik yang memungkinkan pembuatan, manipulasi, transformasi, dan eksekusi data terstruktur grafik dalam skala besar. Ini terdiri dari berbagai API Spark RDD yang memfasilitasi pembuatan grafik terarah.
- *Percikan Inti:*Berbagai fungsi Apache Spark dibangun di atas inti Spark. Ini menyediakan berbagai macam API serta aplikasi untuk bahasa pemrograman seperti Scala, Java, dan Python API untuk memfasilitasi kemudahan pengembangan. Komputasi dalam memori diimplementasikan dalam inti Spark untuk memberikan kecepatan dan memecahkan masalah MapReduce.
- *percikanR:*Ini adalah paket untuk R yang memungkinkan data scientist memanfaatkan kekuatan Spark dari shell R. Karena DataFrame adalah struktur data dasar untuk pemrosesan data di R, SparkR DataFrame juga merupakan unit dasar SparkR. Ia dapat melakukan berbagai fungsi pada kumpulan data besar seperti seleksi, pemfilteran, dan agregasi [12].

D. ARSITEKTUR

Seperti diilustrasikan pada Gambar 2, arsitektur Apache Spark terdiri dari master node yang memiliki program driver yang bertugas memanggil program utama suatu aplikasi. Program driver adalah kode yang ditulis oleh pengguna atau jika shell interaktif digunakan, itu adalah shell. Program driver ini bertanggung jawab untuk menciptakan konteks Spark. Konteks Spark berperilaku seperti gerbang ke semua fungsi Apache Spark. Ia bekerja dengan manajer cluster yang bertanggung jawab untuk mengelola pekerjaan yang berbeda [13]. Konteks Spark dan program driver secara kolektif menangani eksekusi pekerjaan dalam cluster.

Manajer cluster pertama-tama menangani pekerjaan pengalokasian sumber daya. Kemudian pekerjaan tersebut dipecah menjadi beberapa tugas yang selanjutnya dialokasikan ke node pekerja atau budak. Saat RDD dibuat dalam konteks Spark, RDD dapat dialokasikan ke seluruh node budak yang berbeda dan juga dapat di-cache di sana. Node budak berperan dalam melaksanakan tugas yang diberikan kepadanya oleh manajer cluster. Mereka kemudian mengembalikan tugas-tugas ini ke konteks Spark. Pelaksana melaksanakan pelaksanaan tugas. Masa hidup eksekutornya sama dengan Spark. Untuk meningkatkan kinerja sistem, jumlah node pekerja harus ditingkatkan sehingga pekerjaan dapat dibagi lebih lanjut menjadi lebih banyak bagian logis [14].

E. PERSYARATAN PERANGKAT KERAS

- *Sistem Penyimpanan:*Penting untuk menjaga sistem penyimpanan tetap dekat dengan sistem Spark, karena sebagian besar pekerjaan Spark membaca data input dari sistem penyimpanan eksternal seperti HDFS (Hadoop Distributed File System) atau HBase [15].
- *Disk Lokal:*Meskipun Spark melakukan banyak komputasi di memori, Spark masih menggunakan disk lokal, yang tidak muat di RAM, untuk menyimpan data dan mempertahankan keluaran perantara antar tahapan. Lebih baik memiliki 4-8 disk per node, yang dikonfigurasi tanpa RAID.
- *Penyimpanan:*Spark berjalan dengan baik pada memori mulai dari 8 Gigabyte hingga ratusan Gigabyte per mesin. Lebih baik mengalokasikan paling banyak 75% memori untuk Spark dalam semua kasus, dan sisanya harus ditetapkan ke sistem operasi dan cache buffer.
- *Jaringan:*Banyak aplikasi Spark terikat jaringan ketika data ditemukan di dalam memori. Aplikasi ini dapat dibuat lebih cepat dengan menggunakan jaringan 10 Gigabit atau lebih tinggi. Hal ini terutama berlaku untuk aplikasi pengurangan terdistribusi seperti gabungan SQL.
- *Inti CPU:*Spark melakukan pembagian minimal antar thread dan oleh karena itu ia dapat diskalakan dengan baik hingga puluhan inti CPU per mesin. Setidaknya 8-16 core per mesin harus disediakan. Penyediaan inti bergantung pada biaya beban kerja CPU [15].

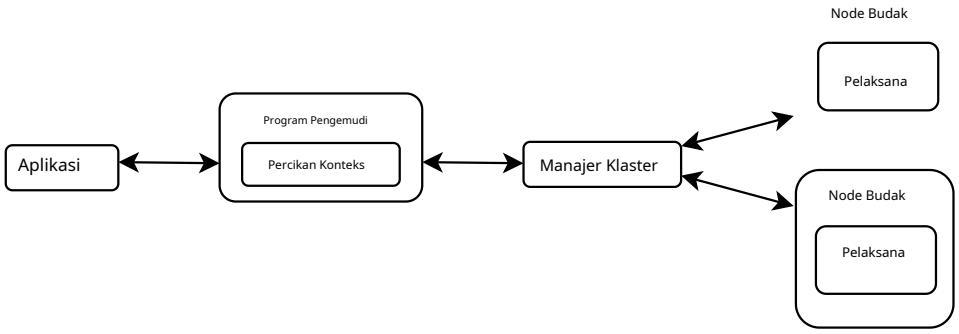
F. MULTITHREADING DAN KONKURENSI

*1) Multithread:*Apache Spark memiliki API untuk banyak bahasa seperti Scala, Python Java, dan R. Penggunaan Spark yang paling populer adalah dengan Scala dan Python. Memilih bahasa mana yang akan digunakan dengan Spark bergantung pada fitur yang ingin kita manfaatkan [16]. Mengenai multithreading, Python berada pada posisi yang kurang menguntungkan dibandingkan Scala karena Python tidak mendukung multithreading. Scala di sisi lain mendukung multithreading. Memiliki program multithread berarti kita dapat menjalankan lebih dari satu tugas secara bersamaan. Thread adalah proses ringan yang menggunakan lebih sedikit memori dibandingkan proses kelas berat. Membuat thread di Scala dapat dilakukan dengan dua cara. Hal ini dapat dilakukan dengan memperluas kelas Thread atau antarmuka Runnable. Metode run() dapat digunakan setelah itu [17]. Sebuah thread Scala dapat melewati lima status berbeda selama masa pakainya yang dijelaskan sebagai berikut:

1) Baru: Keadaan awal thread.
2) Runnable: Thread siap dijalankan tetapi belum dipilih oleh penjadwal.
3) Berjalan: Thread sedang dieksekusi.
4) Diblokir: Thread sedang menunggu beberapa peristiwa seperti input atau sumber daya.
5) Dihentikan: Thread selesai dieksekusi.

Selain itu, aliran thread Scala dapat dikontrol menggunakan metode thread Scala yang berbeda [17]. Misalnya, kita dapat menggunakan metode sleep() untuk membuat thread tertidur selama jangka waktu tertentu dan metode join() untuk membiarkan thread menunggu hingga thread lain berakhir.

*2) Konkurensi:*Dalam konkurensi, suatu tugas dikatakan telah selesai ketika semua thread dan sub thread yang berfungsi telah selesai diproses. Di Spark, semua tugas dijalankan di dalam JVM pelaksana. Jumlah tugas yang dapat dijalankan pada saat yang sama dikontrol oleh jumlah inti yang ditangani oleh



Gambar 2. Arsitektur Apache Spark

pelaksana. Pengaturan ini ditentukan selama penyerahan pekerjaan dan bersifat konstan secara umum namun dapat bervariasi jika tugas menggunakan alokasi dinamis. Secara umum, tugas adalah thread tunggal yang menjalankan kode serial yang ditulis untuk tugas tertentu. Kode dalam tugas adalah single-threaded dan sinkron kecuali jika kode tersebut secara langsung menunjukkan bahwa kode tersebut tidak sinkron [18]. Saat Scala berjalan di JVM, ia memiliki akses penuh ke semua kemampuan multithreadingnya. Namun, tidak seperti Java, Scala tidak secara default hanya terbatas pada konsep thread untuk mencapai konkurensi. Ini memberikan opsi lanjutan lainnya yang dapat mencapai konkurensi seperti Futures dan Aktor. Sebaliknya, bahasa Python dan R tidak mendukung konkurensi dan multi-threading yang sebenarnya. Multi-threading hanya dapat berjalan secara paralel untuk beberapa I\|O tugas, tetapi hanya dapat dijalankan satu per satu untuk beberapa tugas inti yang terikat CPU. Oleh karena itu, lebih banyak overhead yang dihasilkan dalam pengelolaan memori dan data [19] .

G. APACHE SPARK DALAM TEKNOLOGI YANG BERKEMBANG

*1) Komputasi Kabut:*Komputasi kabut memerlukan latensi yang sangat rendah, pemrosesan pembelajaran mesin paralel, dan algoritme analisis grafik kompleks yang disediakan oleh Apache Spark. Spark streaming bersama dengan MLlib dan Apache Kafka menjadi dasar deteksi transaksi keuangan palsu. Transaksi kartu kredit seseorang dapat diperoleh untuk mengklasifikasikan pola pengeluaran individu. Model dapat dibentuk dan dilatih lebih lanjut untuk memperkirakan kelainan apa pun dalam transaksi kartu dan bersamaan dengan streaming Kafka dan Spark secara real time. Spark juga dapat digunakan dalam analisis interaktif karena sangat cepat dibandingkan dengan MapReduce yang menyediakan alat seperti Pig dan Hive untuk analisis interaktif [20].

*2) Pembelajaran Mesin:*Apache spark memiliki API yang sangat kuat untuk aplikasi pembelajaran mesin yang dikenal sebagai MLlib yang terdiri dari beberapa algoritma pembelajaran mesin. Misalnya, kita dapat menggunakan Support Vector Machine (SVM) di Spark. SVM adalah algoritma pembelajaran mesin yang digunakan untuk klasifikasi dan analisis regresi. Satu-satunya pengoptimal yang tersedia untuk SVM di Spark adalah pengoptimal SGD [21]. Selanjutnya, Staman juga o mendukung algoritme pembelajaran mesin lain yang disebut XGBoost atau eXtreme Gradient Boosting. Algoritma ini memungkinkan pengguna untuk membangun saluran terpadu dengan menyematkan XGBoost ke dalam sistem pemrosesan data yang berbasis Apache Spark [22].

Contoh lain penggunaan pembelajaran mesin di Spark adalah Deep Learning (DL). Karena DL secara komputasi sangat berat, mendistribusikan prosesnya adalah solusi yang baik dan Apache Spark adalah salah satu cara termudah untuk mengimplementasikannya. DL dapat diimplementasikan di Apache Spark dengan banyak cara, beberapa contohnya adalah sebagai berikut [23]:

- Elephas menggunakan DL Terdistribusi dengan Keras dan PySpark
- Yahoo Inc. menggunakan TensorFlowOnSpark
- CERN menggunakan Keras Terdistribusi
- Qubole menggunakan tutorial Keras dan Spark

Selain itu, Deep Learning Pipelines adalah pustaka sumber terbuka yang menyediakan API tingkat tinggi yang dapat melakukan pembelajaran mendalam yang dapat diskalakan dengan Python dengan Apache Spark. Beberapa kelebihan perpustakaan ini adalah [23]:

- Dengan bantuan pustaka ML Spark, kita dapat memiliki API yang mudah digunakan yang menghasilkan pembelajaran mendalam dalam beberapa baris kode.
- Itu tidak mengurangi kinerja sambil berfokus pada kemudahan penggunaan dan integrasi.
- Karena dibuat oleh pembuat Apache Spark, ia memiliki peluang lebih besar untuk digabungkan sebagai API resmi.
- Karena Python adalah bahasa yang digunakan, maka lebih mudah untuk berintegrasi dengan semua perpustakaan terkenalnya. Selain itu, ia menggunakan kekuatan dua perpustakaan utama untuk Deep Learning yaitu TensorFlow dan Keras.

IV. KESIMPULAN

Big data adalah istilah yang mengacu pada kumpulan data dalam jumlah yang sangat besar yang digunakan untuk mengungkap pola dan tren secara komputasi. Untuk menganalisis dan menemukan pengetahuan dari sebagian besar data ini, diperlukan kerangka pemrosesan. Ada berbagai jenis kerangka data besar yang umum digunakan seperti Apache Hadoop, Apache Storm, Apache Spark, Apache Flink dll. Dalam tulisan ini kita berbicara tentang kemampuan pemrosesan batch dan pemrosesan aliran Apache Spark, kasus penggunaan, ekosistem, arsitektur, multi-threading dan kemampuan konkurensi dan terakhir penggunaan Spark dalam teknologi baru.

RREFERENSI

[1] J. Ellingwood, "Hadoop, storm, samza, spark, dan flink: Kerangka data besar dibandingkan," Situs web, 10 2016. [Online]. Tersedia: https://www.digitalocean.com/community/tutorials/hadoopstorm-samza-spark-and-flink-big-data-frameworks-compared

[2] M. Zaharia, R. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, MJ Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, dan I. Stoica, "Apache spark: mesin terpadu untuk pemrosesan data besar,"*Komunitas. ACM*, jilid. 59, hal. 56–65, 2016.

[3] S. Salloum, R. Dautov, X. Chen, PX Peng, dan JZ Huang, "Analisis data besar di apache spark,"*Jurnal Internasional Ilmu Data dan Analisis*, jilid. 1, tidak. 3, hlm. 145–164, November 2016. [Online]. Tersedia: https://doi.org/10.1007/s41060-016- 0027-9

[4] M. Iqbal dan T. Soomro, "Analisis data besar: perspektif badai Apache,"*Jurnal Internasional Tren dan Teknologi Komputer*, jilid. 19, hal. 9–14, 01 2015.

[5] S. Sarraf dan M. Ostadhashem, "Aplikasi data besar dalam pencitraan resonansi magnetik fungsional menggunakan apache spark," di *Konferensi Teknologi Masa Depan (FTC) 2016*, Des 2016, hlm.281–284.

[6] S. Gopalani dan R. Arora, "Membandingkan apache spark dan pengurangan peta dengan analisis kinerja menggunakan k-means," *Jurnal Internasional Aplikasi Komputer*, jilid. 113, hal. 8–11, 03 2015.

[7] D. García-Gil, S. Ramírez-Gallego, S. García, dan F. Herrera, "Perbandingan skalabilitas untuk pemrosesan batch data besar di apache spark dan apache flink,"*Analisis Data Besar*, jilid. 2, tidak. 1, hal. 1 Maret 2017. [Online]. Tersedia: https://doi.org/10.1186/s41044-016-0020-2

[8] B. Akil, Y. Zhou, dan U. Rhm, "Tentang kegunaan hadoop mapreduce, apache spark apache flink untuk ilmu data," di *Konferensi Internasional IEEE 2017 tentang Big Data (Data Besar)*, Des 2017, hlm.303–310.

[9] VS Jonnalagadda, P. Srikanth, K. Thumati, dan SH Nallamala, "Sebuah studi tinjauan tentang percikan apache dalam pemrosesan data besar," *Jurnal Internasional Tren dan Teknologi Ilmu Komputer (IJCST)*, jilid. 4, tidak. 3, hlm. 93–98, Juni 2016.

[10] "Apache memicu kasus penggunaan secara real time," Situs Web, 11 2018. [Online]. Tersedia: https://data-flair.training/blogs/spark-usecases/

[11] "5 kasus penggunaan apache spark teratas," Situs web, 6 2016. [Online]. Tersedia: https://www.dezyre.com/article/top-5-apache-sparkuse-cases/271

[12] "Spark note untuk pemula & berpengalaman," Website, 6 2016. [Online]. Tersedia: https://data-flair.training/blogs/spark-notes/

[13] Naveen, "Arsitektur Apache spark," Situs Web, 2 2017. [Online]. Tersedia: https://intellipaat.com/blog/tutorial/sparktutorial/spark-architecture/

[14] N. Vaidya, "Arsitektur Apache spark menjelaskan arsitektur cluster percikan," Website, 5 2019. [Online]. Tersedia: https://www.edureka.co/blog/spark-architecture/

[15] "Penyediaan perangkat keras," Situs web. [On line]. Tersedia: https://spark.apache.org/docs/2.1.0/hardware-provisioning.html

[16] P.Gandhi, "Apachespark: Python vs. scala," Situs Web, 2018. [Online]. Tersedia: https://www.kdnuggets.com/2018/05/apache-spark-pythonscala.html

[17] "Apa itu scala thread & multithreading: Penanganan file dalam scala," Website, 9 2018. [Online]. Tersedia: https://dataflair.training/blogs/scala-thread/

[18] R. Spitzer, "Konkurensi di dalam percikan," Situs Web,2017.[Online]. http:// Tersedia: www.russellspitzer.com/2017/02/27/Concurrency-In-Spark/

[19] S. Vaid, "Memilih bahasa pemrograman yang tepat untuk algoritma pembelajaran mesin dengan apache spark," Website, 6 2018. [Online]. Tersedia: https://blogs.opentext.com/choosing-theright-programming-lingual-for-machine-learning-algorithmswith-apache-spark/

[20] N. Kumar, "Kasus & aplikasi penggunaan Apache spark," Situs web, 6 2019. [Online]. Tersedia: https://www.knowledgehut.com/blog/big-data/spark-usecases-applications

[21] S. Yasrobi, J. Alston, B. Yadranjiaghdam, dan N. Tabrizi, "Analisis kinerja perpustakaan pembelajaran mesin percikan," *Transaksi Pembelajaran Mesin dan Penambangan Data*, jilid. 2, hal.67–77, 2017.

[22] J. Brownlee, "Pengenalan lembut tentang xgboost untuk terapan mesin sedang belajar," Situs web, 8 2016. [On line]. Tersedia: https://machinelearningmastery.com/gentleintroduction-xgboost-applied-machine-learning/

[23] F. Vzquez, "Pembelajaran mendalam dengan apache spark part 1," Website, 4 2018. [Online]. Tersedia: https://towardsdatascience.com/deeplearning-with-apache-spark-part-1-6d397c16abd