W ramach projektu zdecydowałem się przeanalizować dane VCF. Tym razem biorąc pod uwagę moje ostatnie problemy z pamięcią, wybrałem instancję: m7i-flex.large na Ubuntu z 64 gb pamięci RAM, dało się to odczuć na każdym etapie analizy.

Do analizy przygotowałem taki kod:

cat > vcf\_analysis\_v1.py << 'PY'

from pyspark.sql import SparkSession

from pyspark.sql.functions import col

spark = SparkSession.builder.appName("GenomicDataAnalysisV1").getOrCreate()

vcf\_path = "data/ALL.chr2.phase3\_shapeit2\_mvncall\_integrated\_v5b.20130502.genotypes.vcf.gz"

df = spark.read.csv(vcf\_path, sep="\t", comment="#", header=False)

cols = ["CHROM","POS","ID","REF","ALT","QUAL","FILTER","INFO"]

if len(df.columns) > 8:

extras = [f"COL{i}" for i in range(9, len(df.columns)+1)]

cols = cols + extras

df = df.toDF(\*cols)

snp\_df = df.filter( col("REF").rlike("^[ACGT]$") &col("ALT").rlike("^[ACGT]$"))

snp\_count = snp\_df.count(

print(f"Liczba mutacji SNP (REF i ALT jednonukleotydowe): {snp\_count}")

spark.stop()

PY

Dostałem output z policzoną liczbą SNPs z chromosomu 12

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez AI może być niepoprawna.

Odpowiadając na pytania

• Jak Spark ułatwia pracę na dużych danych w porównaniu do tradycyjnych metod?

Można równolegle przetwarzać w wielu wątkach danych, co według danych, które sprawdziłem może przyspieszyć pracę nawet 100x. Możliwe jest automatyczne rozdzielanie zapytań przez system oraz optymalizacja zapytań. Dane w wielu formatach są obsługiwane od razu, nie ma potrzeby wcześniejszego rozpakowywania danych.

• Wyzwania w skalowaniu analizy (np. pamięć, dzielenie danych).

Sądzę, że konfiguracja samego procesu zajmuje sporo czasu, co nie opłaca się przy małych wolumenach danych. Jeśli mamy też proste pytania do odpowiedzi (nie wymagające dużej mocy obliczeniowych), to samo przetwarzanie w chmurze i uruchomienie procesu może być dłuższe niż sam proces lokalnie. Łatwo jest też narazić się na zbyt duży koszt, dynamicznie skalująć. Istotne jest ustawienie odpowiedniego zarządzania pamięcią (ja dostałem OutofMemoryError). Przesyłanie danych między węzłami też może spowalniać analizę, warto o tym pamiętać. Analiza rozproszona i właściwa konfiguracja YARN czy SparkConf potrafi mocno zdenerwować ( wiem z własnego doświadczenia 😊). Dane wymagają odpowiedniego partycjonowania przed załadowaniem, żeby to wszystko faktycznie działało szybciej.

• Przykłady zastosowań takich analiz w bioinformatyce, medycynie, IT.

Bioinformatyka jest oczywistym przykładem, zwiększająca się liczba danych i ich złożoność oraz konieczność stosowania wymagających systemów ML/AI do ich uproszczenia i redukcji wymiarowości, to idealne zastosowanie dla klastrów i chmury.

Kolejnym obszarem może być Medycyna spersonalizowana i ta codzienna -szpitalna, gdzie automatyczne systemy mogły by przyjmować dane od pacjentów z całego szpitala i wykrywać interakcje lekowe czy potencjalne błędy zanim przyniosłyby one negatywne konsekwencje dla pacjentów. Innym przykładem może być dopasowanie leku pod landscape genetyczny guza.

Ostatnim przykładem mogą być finanse i śledzenie zmian na giełdzie real-time oraz analiza funkcjonalna na bieżąco czy działania wojenne, do monitoringu pola walki.

• Rola chmury w udostępnianiu środowiska i zasobów do analizy dużych zbiorów danych.

Można dynamicznie skalować zasoby, płaci się tylko za zużycie, a nie za sprzęt, mamy gotowe klastry do procesowania danych, można przechowywać duże zbiory danych, nie trzeba konfigurować środowiska lokalnego, można współpracować wspólnie w chmurze.

• Jakie dalsze kroki można podjąć, aby usprawnić analizę i skalowalność?

Stosować klastry zwieloma węzłami, a nie single node, zapis danych w szybko procesowalnym formacie – np. Parquet, użycie modeli uczenia maszynowego do odkrywania ukrytych wątków w danych, partycjonowanie danych np. według próbki, używanie narzędzi do monitoringu wydajności jak Amazon CloudWatch, równoległa i rozproszona analiza danych

• Jak chmura ułatwia pracę z dużymi zbiorami danych w porównaniu do lokalnych rozwiązań?

Pamięc i CPU jest ograniczona na komputerze, nieograniczona w chmurze

Skalowanie jest tańsze w chmurze niż na komputerze lokalnym, dodatkowo mniej czasochłonne

Dostęp do danych i współpraca są szybsze w chmurze

Koszt w chmurze przy dużych danych jest teoretycznie mniejszy, bo nie wymaga inwestycji sprzęt i nie płaci się za amortyzację, ale w praktyce należy uważać żeby nie zapłacić zbyt dużego rachunku przy nieograniczonych możliwościach skalowania (na komputerze lokalnym nie ma takiego ryzyka)

Integracja w chmurze może być prostsza jeśli stosujemy gotowe rozwiązania, choć ja bezpośrednio tego nie odczułem (często klasycznie musiałem wiele konfigurować)

Dziękuję za przygotowanie przedmiotu i przepraszam za opóźnienie 😊