Abstract - In this paper, we present a system for batch processing and visualization of Formula 1 telemetry data. The goal of this system is to facilitate easier processing and visualization of Formula 1 telemetry data. To achieve this, a system designed for batch processing was implemented. Main features of the system are that the system can work independently with the help of a job orchestrator, which fetches and processes data after every race, as well as visualization of processed data.

Keywords: Big data batch processing, Data visualization, Formula 1, Apache Spark, Big data.

1. INTRODUCTION

Sport, as one of the most popular and competitive sports, has become a significant industry in recent years. The sport's popularity is not only due to its high level of competition but also due to the interest in the data generated during the race. These data include information about the drivers, the car's performance, and the race conditions. The data is collected using various devices, including speed sensors, lap-time clocks, and video cameras. The data is used by teams to analyze their performance and make decisions on how to improve their strategy for the next race.

In this paper, we present a system for batch processing and visualization of Formula 1 telemetry data. The system is designed to help teams analyze the data more efficiently and effectively. The system includes a data processing component that fetches and processes data after every race, as well as a visualization component that allows teams to visualize the processed data in a more intuitive way.

The system is implemented using Apache Spark, a popular distributed computing framework that is widely used in the big data industry. The system is designed to be scalable and can handle large amounts of data. The system is also designed to be easily maintainable and extensible, allowing teams to add new features and functionality as needed.

2. OVERVIEW OF THE SYSTEM

The system is composed of two main components: a data processing component and a visualization component. The data processing component is responsible for fetching and processing the telemetry data after every race. The data is then stored in a distributed file system, which is accessible to the visualization component.

The visualization component is responsible for visualizing the processed data. The system includes a web-based user interface that allows teams to interact with the data. The user interface is designed to be intuitive and easy to use, allowing teams to quickly understand the data and make informed decisions.

3. CONCLUSION

In conclusion, we have presented a system for batch processing and visualization of Formula 1 telemetry data. The system is designed to help teams analyze their data more efficiently and effectively. The system is implemented using Apache Spark, a popular distributed computing framework that is widely used in the big data industry. The system is designed to be scalable and can handle large amounts of data. The system is also designed to be easily maintainable and extensible, allowing teams to add new features and functionality as needed.
Dakle, analiza na nivou jedne cele trke, više trka ili pak cele sezone nije moguća. Analiza postojećih alata koji poseduju timovi koji su učesnici F1 nije moguća zbog toga što ne postoji pristup niti uvid u njihove mogućnosti ili arhitekturu.

3. ARHITEKTURA SISTEMA

Zadatak ovog sistema je da potomčke o telemetriji Formule 1 iz spoljnih izvora prikupi, izvrši različita izračunavanja i agregacije, skladišti u bazi podataka i na kraju prikaže pomoću alata za pravljenje grafikona i vizualizaciju podataka. Na Slici 1 je prikazana arhitektura sistema, dok će u nastavku teksta biti opisani prikazani moduli.

Slika 1. Arhitektura sistema

Podaci se preuzimaju sa jednog od eksternih izvora podataka. Nakon toga, orkestrator poslova izdaje zadatke komponenti za paketnu obradu podataka da skladišti ove pristupčene podatke u CSV formatu u sirovu zonu podataka na distribuiranom sistemu datoteka. Iz sirove zone podataka se podaci učitavaju u komponentu za paketnu obradu koja ih obradjuje i transformiše i tako transformirane ih čuva u CSV formatu u direktorijumima zone transformacije. Poslednji korak u ovom sistemu je upisivanje transformisanih podataka u trajno skladište podataka.

3.1 Orkestrator poslova

Zadatak orkestratora tokova poslova je da zadaci koji su definisani budu izvršeni u zakazano vreme. Ove zadatke definiše korisnik i mogu biti Python ili beš skripte, kao i beš komande. Apache Airflow kao tehnologija namenjena za orkestraciju tokova poslova ima mnogobrojne alternativne. Neke od najpopularnijih tehnologija pored Airflow-a jesu Luigi, Apache NiFi, AWS Step Functions i Prefect. U uži odaber za korišćenje tehnologije za izvršavanje tokova poslova ušuši su Airflow i Luigi, te će u narednom delu teksta biti dato poredenje ove dve tehnologije.

Suštinski najbitnija karakteristika kod ove komponente je način i mogućnost za zakazivanje zadataka i tokova poslova. Distribuirana priroda Airflow omogućava i distribuirano izvršavanje zadataka koje je potrebno podsiti na izvršiocu nakon čega je dovoljno prepustiti zakazivaču koji je sastavni deo Airflow da sam pokreće zadatke. Luigi ne podržava mogućnost da sam pokreće zadatke, već je moguće namestiti cron posao (engl. Cron Job) putem komandne linije kako bi on pokrenuo izvršavanje poslova. Iako distribuirano izvršavanje u ovom slučaju ne igra ključnu ulogu, sposobnost Airflow-a da se zakaže i izvrši je ključna u odabiru ove tehnologije. Osim toga, Airflow ima mogućnost da pokrene izvršavanje sledećeg zadatka iako se njegov prethodnik i dalje nije potpuno završio. Ovo preklapanje ubrzava izvršavanje tokova, ali je kompleksno za implementirati.

3.2 Distribuirani sistem datoteka

Distribuirani sistem datoteka u ovom sistemu služi za skladištenje podataka u sirovom i transformiranom obliku. Podaci se čuvaju u CSV formatu i u zoni transformacije i u sirovoj zoni. U sirovu zonu se upisuju tek preuzeti, neobrađeni podaci. Komponenta za paketnu obradu učitava sirove podatke i nad njima vrši transformacije. Nakon transformisanja, podaci se čuvaju u zoni transformacije, odakle se prepisuju u bazu podataka.

Kao rezultat pretrage alternativa za distribuirani sistem datoteka Hadoop najčešći odgovor na koji se nailazi je HBase. HBase je nerelacioni sistem za upravljanje bazom podataka usmerenom na kolone (engl. Column-oriented). HBase kao svoju osnovu ima HDFS i namenjen je za skladištenje i upravljanje velikom količinom strukturiranih i polu strukturiranih podataka, a takođe svoju otpornost na greške eksploataše iz HDFS-a.

HDFS je sistem namenjen za skladištenje velikih skupova podataka na klastru izgrađenom od kućnih uređaja i namešnjen je za paketnu obradu podataka, dok je sa druge strane HBase NoSQL baza podataka. HBase je optimizovan za nasumičan pristup pisanju i čitanju velikih skupova podataka i može da podrži veći broj korisnika istovremeno.

HDFS skladišti podatke na osnovu hijerarhijske strukture datoteka pri čemu su te datoteke izdeljene u blokove podatke i distribuirane u klastru. Za razliku od HDFS-a, HBase podatke organizuje po redovima i kolonama unutar sebe.

3.3. Paketna obrada podataka

Paketna obrada podataka istovremeno obrađuje veliku količinu podataka koji se prvo skladište, a zatim se čitaju u paketima [2]. Specifikacija veličine paketa zavisi od implementacije sistema, te veličina može biti ograničena na različite načine kao što je broj podataka, vremenski okvir ili polu strukturiranih podataka, a takođe svoju otpornost na greške ekspoataše iz HDFS-a. HBase je sistem namenjen za skladištenje velikih skupova podataka, a HDFS je optimizovan za nasumičan pristup pisanju i čitanju velikih skupova podataka.

Kao jedna od mogućih alternativa u kojoj je moguće impiementirati paketnu obradu podataka je Apache Flink. Spark i Flink su radni okviri otvorenog koda namenjeni za procesiranje velikih skupova podataka i jedni su od vodećih kodaka u pitanju njihova upotreba. Obe ove tehnologije namenjene su za obradu velikih skupova podataka u paralelnom režimu pritom pružaajući podršku putem prog.
ramskog interfejsa tokom analize i manipulacije podacima.

**Slika 2. Konceptualni prikaz paketne obrade podataka**

**Spark** je više optimizovan za paketnu obradu podataka i koristi se u slučajevima gde je za poslove koji vrše paketnu obradu prihvatljivo da imaju nešto višu latenciju u odnosu na obrađivače u realnom vremenu [3]. Upravljanje memoriom kod Spark-a oslanja se na model upravljanja memoriom poznat kao RDD Caching. RDD kešing omogućava skladištenje rezultata međukoraka u toku obradivanja podataka što ovom sistemu veoma pogođuje jer je potrebno imati više međurezultata tokom agregiranja telemetrijskih podataka. Model obrade podataka (engl. Data Processing Model) koji koristi Spark je distribuirani skup podataka otporan na otkaze (engl. Resilient Distributed Dataset, RDD). RDD model predstavlja kolekciju elemenata otpornih na otkaze koje je moguće obradivati u paralelnom režimu.

Glavna prednost Spark-a je to da su performanse u poređenju sa Hadoop-om i do 100 puta brže za skladištenje pomoćnih rezultata u memoriji, kao i 10 puta kada se podaci čuvaju na disku [4] Prethodno napisana osobina je u ovom slučaju od velikog značaja jer u toku svake iteracije kreiraju privremene rezultate koje koristimo u toku transformisanja telemetrijskih podataka.

**3.4 Komponenta za vizuelizaciju**

Postoji mnogo alternativa koje je moguće koristiti za vizuelizaciju podataka poput Tableau, Looker, Power BI, Superset i mnogih drugih.

Jedan od kriterijuma koji je bio bitan pri odabiru alata za vizuelizaciju svakako mora biti izgled samog korisničkog interfejsa. Metabase je alat koji je namenjen za korišćenje „obr“ vrste korisnika. Prilagođen je i ljudima koji ne dolaze iz sveta informacionih tehnologija, te nemaju nikakvo predznanje o uticajima na različite grafikone. Koristeći alate za pravljenje upita i već gotovih grafikona, nikakvo predznanje o upitima nad bazom podataka.

**4. PRIKAZ IZGLEDA SISTEMA**

Implementiran sistem omogućava interaktivnu analizu prikupljenih telemetrijskih podataka. U okviru ovog poglavlja cilj je prikazati moguće izglede aplikacije i njene mogućnosti.

Na Slici 3 prikazan je stubičasti grafikon koji prikazuje koji je najveći broj promena stepena prenosa na Velikoj nagradi Monaka za 5 vozača. Ovaj grafikon je moguće dobiti u svega 4 koraka koristeći Metabase-ov editor za pravljenje upita nad bazom podataka.

Slika 3. Broj promena stepena prenosa na VN Monaka

Pomoću ovog sistema moguće je veoma lako generisati različite grafikone koristeći telemetriju ne samo iz jedne sesije, već i iz cele sezone, što je glavno ograničeno svih trenutno dostupnih alata. Koristeći ugrađeni vizuelni editor za upite moguće je brzo i lako vizualizovati unapred pripremljene podatke.

U sklopu Metabase-a moguće je i sačuvati generisane grafikone za kasnije korišćenje i grupisati ih u kontrolne table. Kontrolne table su veoma pogodne za korišćenje u slučajevima kada je potrebno brzo pristupiti različitim tipovima grafikona. Dodatna funkcionalnost koju Metabase pruža svojim korisnicima je da se pretplate na neku od kontrolnih tabli i da pretplaćenim korisnicima pošalje nove obaveštenje kada se novi grafikon formira.

**5. ZAKLJUČAK**

Formula 1 je specifičan sport iz više razloga. Prvi razlog jeste da za dobar rezultat nije dovoljan samo sposoban takmičar, već i performantan bolid kojim takmičar upravlja. Drugi je da za razliku od ostalih sportova u kom se tim ili pojedinac takmiče u izolovanim događajima koji se na kraju odražavaju na finalno stanje u šampionatu, u Formuli 1 se svi takmičari takmiče u isto vreme sa svim ostalim protivnicima tokom svih događaja.

Poboljšavanje bolida dugo je zavisilo isključivo od sposobnosti vozača da prenese svoje utiske mehaničarima. Sada, inženjeri pretežno se oslanjaju na rezultate analiza i obrada podataka koji skupljaju tokom slobodnih treninga i trka kako bi bili u mogućnosti da shvate šta se tačno događa. Drugi je da za razliku od ostalih sportova u kom se tim ili pojedinac takmiče u izolovanim događajima koji se na kraju odražavaju na finalno stanje u šampionatu, u Formuli 1 se svi takmičari takmiče u isto vreme sa svim ostalim protivnicima tokom svih događaja.

Poboljšavanje bolida dugo je zavisilo isključivo od sposobnosti vozača da prenese svoje utiske mehaničarima. Sada, inženjeri pretežno se oslanjaju na rezultate analiza i obrada podataka koji skupljaju tokom slobodnih treninga i trka kako bi bili u mogućnosti da shvate šta se tačno događa. Drugi je da za razliku od ostalih sportova u kom se tim ili pojedinac takmiče u izolovanim događajima koji se na kraju odražavaju na finalno stanje u šampionatu, u Formuli 1 se svi takmičari takmiče u isto vreme sa svim ostalim protivnicima tokom svih događaja.

Ovaj rad usmeren je na razvijanje sistema koji gleda ocima ambicije za uspesima u ovom sportu. Odgovarajuće rezultate čine neizostavne činioce svakog tima koji ima dozvoljenost vozača da prenese svoje utiske mehaničarima. Peti je da za razliku od ostalih sportova u kom se tim ili pojedinac takmiče u izolovanim događajima koji se na kraju odražavaju na finalno stanje u šampionatu, u Formuli 1 se svi takmičari takmiče u isto vreme sa svim ostalim protivnicima tokom svih događaja.
trkačkog vikenda. Subjektivni osećaji koji se dobija putem prenosa trka na televiziji ili tokom boravka na stazi mogu biti potpuno različiti od onoga što pokazuju podaci. Zbog toga je potrebno da se prilikom svake diskusije ili analize onoga što je viđeno na trci uključu i nepobitne činjenice kako bi se dobila potpuna slika.

Realizovani sistem sastoji se od nekoliko ključnih delova. Prvi deo je za dobavljanje podataka sa izvora podataka bez kojih vršiti bilo kakvu analizu. Dalje sledi sposobnost sistema da prikupljene podatke učita, obradi, agregira i pripremi za vizualizaciju. Poslednji, ali možda najbitniji deo, jeste čuvanje i prikaz tih podataka. Bez dobre vizualizacije nema ni analize koja bi mogla dublje dočarati što se tačno desilo na stazi.

Smer u kom ide dalji napredak ovog sistema fokusiran je na razvijanje striming aspekta obrade podataka. Obrada podataka u realnom vremenu dala bi potpuno novu dimenziju analize podataka jer bi gledaoci mogli dobiti poređenja za manje od minute od trenutka dešavanja na stazi. Dalji napredak ovog sistema mogao bi se ogledati u povezivanju postojećih informacija o telemetriji sa podacima o vremenskim uslovima na i oko staze. Ovime bi se mogao dobiti dublji uvid u to koji bolidi i vozači se kako ponašaju pri različitim vremenskim uslovima. Primer ove analize mogao bi dovesti do pravljenja relacija vezane za kakve performanse ima određeni tim na stazama koji su na višoj nadmorskoj visini poput Meksiko Sitija ili kako se koji vozač nosi sa visokom vlažnošću vazduha.

6. LITERATURA

[1] theOehrl. (2022). FastF1 Documentation [Onlajn]. Dostupno na: https://theoehrly.github.io/Fast-F1/

[2] Packt. (2022). Distributed batch processing [Onlajn]. Dostupno na: https://subscription.packtpub.com/-book/big-data-and-business-intelligence/9781784391409/1/ch01lvl1sec13/distributed-batch-processing.

[3] ProjectPro. (2023). Apache Flink vs Spark - Will one overtake the other? [Onlajn] Dostupno na: https://www.projectpro.io/article/apache-flink-vs-spark-will-one-overtake-the-other/282.

[4] Vaidja, N. (2022). Apache Spark Architecture - Spark Cluster Architecture Explained [Onlajn]. Dostupno na: https://www.edureka.co/blog/spark-architecture/

Kratka biografija:

Aleksa Vučaj rođen je 8. septembra 1998. godine u Novom Sadu, Vojvodina. Pohađao je gimnaziju „Isidora Sekulić”. Fakultet tehničkih nauka upisao je 2017. godine na smeru Računarstvo i automatika, da bi 2021. diplomirao. Master studije na studijskom programu Računarstvo i automatika usmerenje Računarstvo visokih performansi upisuje 2021. i polaže sve planom i programom predviđene predmete.