A2 - Computação Escalável

André Costa
Emanuel Bissiatti
João Lucas Duim
Rafael Portácio
Victor Bombarda

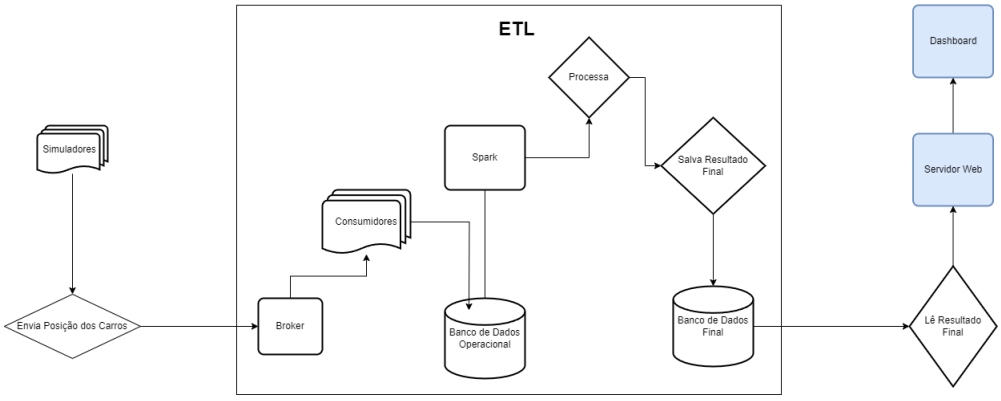
Junho de 2023

1 Introdução

O projeto visa desenvolver um sistema distribuído de monitoramento de rodovias capaz de processar grandes volumes de dados em tempo real. Calculando métricas como velocidades, acelerações, velocidade média, tempo médio de travessia. E também com detecção de eventos como direção perigosa e risco de colisão e a partir desses eventos aplicar multas ou proibir circulação, detectando a presença de carros que estão proibidos.

2 Modelagem

A modelagem do projeto inclui a configuração de múltiplos simuladores (um para cada rodovia) para gerar dados de posição dos veículos, o uso de um broker para distribuir esses dados aos consumidores, o armazenamento dos dados em um banco de dados operacional de escrita e leitura de dados frenética, o processamento dos dados pelo Spark e o armazenamento do resultado final em um banco de dados relacional. Por fim, um servidor web é configurado para disponibilizar um dashboard que permite a visualização dos resultados do monitoramento.



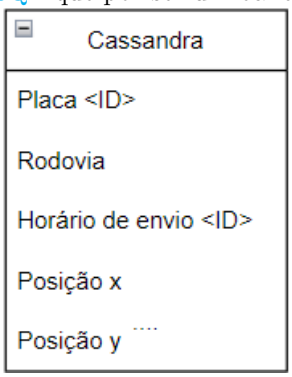
2.1 Broker

O broker usado é o RabbitMQ. É um broker escalável, garante que podemos inserir consumidores e produtores sem afetar o seu funcionamento, pode-se controlar o consumo de mensagens e possui alta confiança contra perda de mensagens.

Implementado com Celery no python, o broker é um intermediário que recebe as posições dos veículos dos simuladores, garante a comunicação e o processamento assíncrono distribuindo as tarefas eficientemente entre os consumidores.

2.2 Banco Operacional

Os consumidores inserem o dado num banco operacional Cassandra para posterior processamento. Na versão AWS, encontramos problemas de permissão em usar o Amazon Keyspaces e por isso trocamos para MySQL que por ser um banco relacional requiriu algumas adaptações.



2.3 Processamento

O processamento dos dados é realizado pelo Spark, um poderoso framework de processamento distribuído. Ele é responsável por aplicar transformações nos dados recebidos de forma vetorizada, permitindo a análise em tempo real de um grande volume de dados.

Etapas principais de processamento:

- Cálculo de velocidade e aceleração com funções de lag numa Window.
- Cálculo da posição prevista.
- Calcula-se a média das velocidades calculadas por rodovia e com ela atualiza-se as velocidade médias.
- Detecção dos riscos de colisão ao comparar a posição prevista dos carros consecutivos em uma mesma faixa com Window.
- Join com os parâmetros da rodovia para encontrar veículos acima da velocidade máxima ou de aceleração elevada (para direção perigosa).
- Detecção de mudança de faixa e de ultrapassagem da velocidade máxima (para aplicação de multa).
- Detecção de entrada e saída da rodovia e registro delas no df_travessia
- Calcula-se o tempo de travessia das placas em que ambas entradas e saídas foram registradas.

2.4 Banco Final

Após o processamento, o resultado final é salvo em um banco de dados final em MariaDB, que serve como um repositório de dados consolidados para posterior consulta ou acompanhamento. Na versão AWS encontramos maior facilidade ao utilizar MySQL e por isso usamos ele no lugar.

rodovias		
nome_rodovia	varchar(20)	PK
horario_registro	bigint	PK
velocidade_media	float(4,2)	
tempo_medio_cruzamento	float(4,2)	
tempo_processamento	float(6,4)	
total_veiculos	int	N
veiculos_acima_vel	int	
veiculos_colisao	int	

carros		
placa	varchar(10)	PK
horario_registro	bigint	PK
rodovia	varcha(20)	
pos_x	float(4,3)	
pos_y	float(4,3)	
aceleracao	float(3,3)	N
velocidade	float(3,3)	N
multas	int	N
risco_colisao	bool	N
direcao_perigosa	bool	N
velocidade_acima	bool	N

2.5 Servidor Web

Para visualizar os resultados do monitoramento, um servidor web é configurado para disponibilizar um dashboard feito em JavaScript. O servidor web atua como uma interface para os usuários, permitindo a visualização dos dados em tempo real de forma amigável e intuitiva. Os resultados finais são lidos pelo servidor web a partir do banco de dados final e são apresentados no dashboard, fornecendo informações valiosas sobre o tráfego nas rodovias.

3 Docker

Utilizamos Docker para encapsular todo o ambiente necessário para a execução local do projeto, incluindo Spark, Celery, Cassandra, MariaDB, RabbitMQ e Networks. Essa abordagem garante que o projeto funcione de maneira consistente em todas as máquinas do grupo, eliminando possíveis inconsistências de configuração e ausência de dependências.

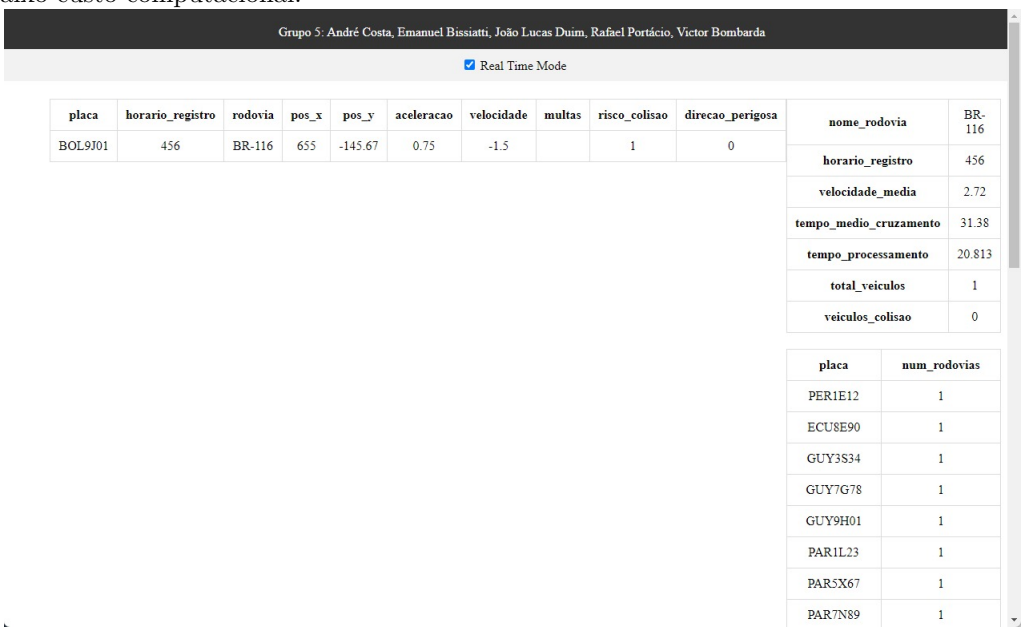
4 AWS

A forma pela qual ficou implementado na AWS foi simples: se utilizou máquinas EC2 e bancos RDS. Como o serviço Amazon MQ não estava funcionando, optamos por utilizar uma máquina EC2 que está servindo como Broker e provedor de consumidores/workers.

Esta máquina então abre as portas para permitir que os produtores/simuladores se conectem com este e enviem as mensagens. Na implementação em nuvem optamos por fazer ambos os bancos relacionais, seguindo a mesma lógica anterior: um banco operacional MySQL e o outro, ainda MariaDB, que serve para se conectar direto com o frontend.

5 Dashboard

O dashboard foi desenvolvido em nodeJS responsável por ler as informações do banco de dados final e realizar as requisições HTML para o dashboard web. O dashboard busca pelas informações mais recentes do banco a cada 1 segundo, atualizando-se quando necessário. Além disso, a análise dos Top 100 veículos que passaram por mais rodovias também é realizada aqui, visto a necessidade de armazenamento das informações e baixo custo computacional.



6 Problemas

6.1 Problemas no Processamento

- Baixa performance utilizando abordagem frame-a-frame
  - Solução: Alteramos para uma abordagem por batch
- Lógica do histórico de multas
  - Solução: Criação de dataframe armazenando as últimas 10 multas de um carro e seu tempo, verificando a diferença de tempo da primeira para a última recebida.
- Necessidade de enviar informações para próximas iterações para o cálculo do tempo de travessia
  - Solução: Criação de um dataframe de registro de entrada e saída da rodovia
- Verificação de carros com direção perigosa
  - Solução: Usar lógica similar a desenvolvida para o sistema de multas, mas incluindo coluna definindo o tempo limite da punição.

6.2 Problemas na AWS

- Impossibilidade de utilizar banco de dados NoSQL
  - Solução: Adaptamos o projeto para utilizar MySQL como banco operacional
- Configuração da rede interna
  - Solução: Foi necessário aprender a conectar devidamente todos os componentes necessários de uma instância a outra.