Midterm Project Demo
DEBS - 1

March 14, 2022

Manos, Vivek, Jax, Karan

Project Overview

Trading on financial markets vary sensitively by precise real time event data

Aim to address the problem of detecting price variation pattern from the given query and provide buy/sell advice for another query given the previous pattern

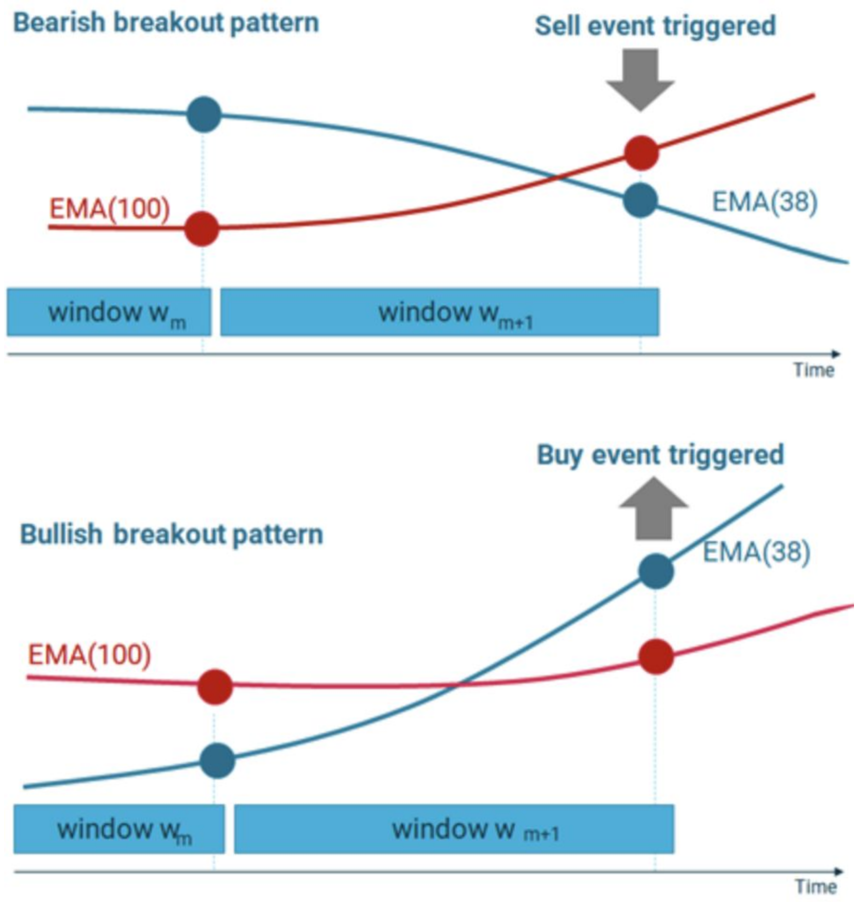
Project Overview - Query 1

The first query defines one of the most essential indicators per symbol used in technical analysis to identify trends: the exponential moving average (EMA)

$$EMA^{j}_{w_i} = \left[Close_{w_i} \cdot \left(\frac{2}{1+j}\right)\right] + EMA^{j}_{w_{i-1}}\left[1 - \left(\frac{2}{1+j}\right)\right]$$

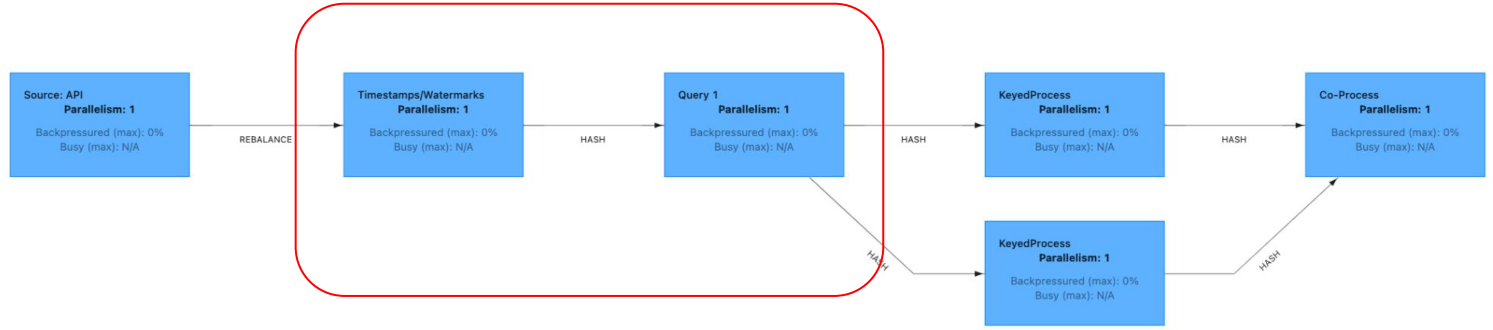
Project Overview - Query 2

- Works in succession with Query 1.
- When timestamp hits every 5 minutes, we receive a pair of EMA values (EMA38, EMA100) for each symbol stock.
- Checks Bearish ("Sell") and Bullish("Buy") conditions for possible crossovers.
- Stores last 3 crossovers per symbol into a data structure & constantly updates it.
- Returns the last 3 crossovers through GRPC protocol per symbol stock & per batch.



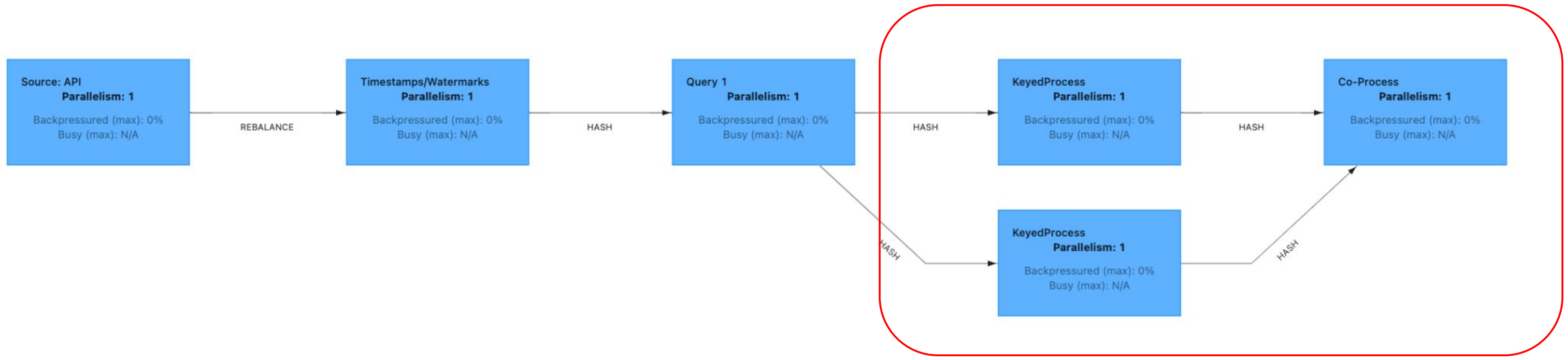
Status Update - Query 1

- Implemented base classes for each event type - Stock Measurement, EMA object and EMA stream object.
- Completed the logic of EMA calculation for a specific stock
- Designed the operator specification for Query 1
    - Used Tumbling windows with custom Event timestamp specification
    - Used Value state to store value from previous tumbling window for EMA calculation



Status Update - Query 2

- Implemented functions to check for bullish and bearish crossover events.
  - Tried both parallelized and sequential implementations
- Both implementations use the EMA stream object output by Query 1.
- Use value state to store the past EMA, window ID and symbol
- Store last 3 crossover events for each symbol and send results to GRPC client benchmark.



Project Demo

Challenges & RoadMap - Query 1

- Query 1 is currently using the default watermarking strategy (monotonous timestamps) which is not applicable for actual evaluation data.
- Dummy events are used to signal the end of a batch of event for benchmarking purposes causing higher number of events to process
- Process function uses unnecessary memory even though we only need the last event of the window
- Currently throughput of query 1 is low which needs to be investigated

Challenges & Roadmap - Query 2

- Decide which architecture strategy to follow:
  - (**Parallel**) Calculate Bearish-Bullish conditions in parallel & connect the streams afterwards.
    - (+) Parallelize the conditions to work on same time.
    - (-) Need for extra data structures to store the intermediate results.
    - (-) Increase the number of conditions more than having them in a single operator.
  - (**Sequential**) Calculate Bearish-Bullish conditions in a single operator
    - (+) No need for extra data structures for intermediate results.
    - (-) The parallelization of Bearish-Bullish conditions is not an option.

- Pass the correct timestamp in order to send it back to the GRPC server as protobuf requires.

- Send the data of Query 2 per batch and not at the end of all batches of the dataset.

References

https://cs551-gitlab.bu.edu/cs551/spring22/team-debs-1/debs2022

Future Roadmap - Query 2

- Test and choose the better implementation strategy to detect crossover events - parallelized vs. sequential.
- Implement a batch check, so as to send crossover events per symbol for each batch.
- Send the correct timestamp as required by the GRPC client for the results of query 2.