Blockhouse Quantitative Developer Work Trial Task

Description: Reconstructing MBP-10 data from MBO data

Background

With the objective of optimizing execution quality for orders submitted by a client, the state of the limit order book is often used as an input to models. With knowledge of how many orders are at a certain price, traders can gauge the liquidity of the stock. And with knowledge of how the order book changes across time, traders can potentially learn the microstructures of the asset for short term alpha following.

However, oftentimes the limit order book is not readily streamed by all exchanges. What is provided by the exchanges, is a stream of trade actions that takes place on the exchange. Traders ingest this stream of actions, such as “Trades”, “Fills”, “Cancellations” etc. and reconstruct their own running orderbook.

Difficulty arises due to the sheer speed of actions being taken on the exchange for a given ticker. It is then important for the orderbook reconstruction to keep up with the incoming stream of actions in order to have the most accurate picture of market state at any given moment.

Task

Your task is to use C++ to compute the MBP-10 orderbook from a given set of MBO data.

You will receive 2 .csv files: mbo.csv and mbp.csv that correspond to sample input and output (See following section for more precise description of data).

Please format your output csv columns such that it aligns exactly with the sample mbp.csv

We are expecting both the source files and the executable binary file in order to evaluate your code performance. We should be able to build your executable binary file with your source code using an included Makefile. Make sure that your code compiles. Afterwards, we should be able to run your executable by passing in the mbo.csv input data.

Example usage:

```
./reconstruction_<name> mbo.csv
```

Please also provide a Readme.txt that explains the overall optimization steps taken and any special things to take note when running your code.

*Finally Upload your code to a **private** GitHub repo and attach the link in your submission.*

IMPORTANT NOTE: There are certain special operations we like you to perform during the reconstruction.

1. Ignore the initial row as this is a “clear[R]” action to clear the orderbook
   a. -> Assume we are starting the day with an empty orderbook
2. In the data (provided by Databento + more details below), both actions [T]rade and [F]ill do not affect the orderbook, only the [C]ancel action followed by it does. For our internal use, we prefer to combine these 3 actions in the MBO to a single T action in the MBP-10 output that *does* reflect in the orderbook.
   a. Note: in the MBO, for a sequence of actions in the order T -> F -> C, the T will be incoming from the *opposite* side that is currently in the book.
   b. For example: a T action may appear on the ASK side at 10 dollars. However, this level does not actually exist on the ASK side of the book. Because the trade has to be matched to an order in the book, the existing order is on the BID side of the book. We see that both F and C actions are on the ASK side
   c. *Therefore*, we store the T action on the BID side as that is the side whose change is actually reflected in the book.
3. If the side of the row with action ‘T’ is ‘N’, we should not alter the orderbook.

Evaluation

1. Correctness
2. Speed
3. Coding style
4. Readme Insights
5. Unit Tests (optional)

You will firstly be tested on correctness, as speed without correct output is not useful. However, given the high frequency nature of this computation, the critical component of this trial will be speed and space efficiency of the reconstruction. This is where overperformers will get the highest marks. Speed across all submissions will be evaluated using a consistent test bench.

We will also look at your source code too and use that to evaluate code cleanliness and interpretability.

And finally, your Readme will be reviewed to evaluate your thought process in writing performant code. If something didn’t work as great as you liked, feel free to add that here too! And consider what could have been better! Any limiting factors? Try to be concise, but we’d like to know your thought process!

For extra points, you can include your own unit tests that you used to check for correctness and performance.

Date Description and Source

As we cannot share internal data, data is provided by Databento which is a standard and reliable provider for market data.

- MBO: Market by order. In lieu of live market data, this is a timeseries dataset that tracks all the actions that have taken place on the exchange.
- MBP-10: Market by price - top ten levels. This simply means the orderbook with both bid and ask side up to ten levels of depth.
    - **NOTE**: Databento provides a MBP-10 dataset by default however it is not sufficient for our purposes. The attached mbp.csv is the expected output.

For much more granular details, Databento provides their own documentation here:

- **MBO**: https://databento.com/docs/schemas-and-data-formats/mbo?historical=python&live=python&reference=python
- **Action code**s: https://databento.com/docs/standards-and-conventions/common-fields-enums-types#action?historical=python&live=python&reference=python

In fact, there is already a **starting point** for the orderbook reconstructor here:

https://databento.com/docs/examples/order-book/limit-order-book/example

DO NOTE: Copy pasted as-is will obviously not be accepted. We must see attempts to improve the speed and efficiency of the code and associated explanation in the Readme. The output must also match the expected output as shown in mbp.csv.

If there are any questions, please reach out!