Programming Assignment

harshit@quantboxtrading.com

2023-07-10

Instructions

1. Code has to be written in C++ for g++ on linux
2. Do not use any non standard libraries like boost or folly
3. Complete code with build/run instructions (preferably makefiles) have to be zipped and mailed back to us.
4. In case you use any resources from the internet, mention those.
5. Please mention all assumptions you take in your solutions
6. Your solutions would be judged on quality/comments/assertions along with space/time efficiency.
7. You need to provide the test cases for your code

Question 1

You have been given market data in binary format for a new exchange. It consists of a PacketHeader that contains a number of messages inside the packet, followed by that many messages.

The format is as follows:

Table 1: Data Types

Size	Data Type
8 bytes	uint64_t
4 bytes	uint32_t
1 byte	char/uint8_t

Table 2: Packet Header

Name	Length	Data Type	Notes
Timestamp	8 bytes	uint64_t	Timestamp of packet (time since epoch in nanoseconds)
NumMessages	4 bytes	uint32_t	Number of messages

Messages

Table 3: New - This message is sent when a New order is added to book

Name	Length	Data Type	Notes
MsgType	1 byte	char	This is ‘N’ for new message
Price	8 bytes	uint64_t	Price of order
Quantity	8 bytes	uint32_t	Quantity of order
SymbolId	4 bytes	uint32_t	Id of Symbol
OrderId	8 bytes	uint64_t	Order id of order - unique across symbols
Side	1 byte	uint8_t	Side of order

Table 4: Cancel - This message is sent when a order is completely cancelled from book

Name	Length	Data Type	Notes
MsgType	1 byte	char	This is ‘C’ for cancel message
Orderid	8 bytes	uint64_t	Order id of order - unique across symbols

Table 5: Trade - This message is sent when an order is completely executed

Name	Length	Data Type	Notes
MsgType	1 byte	char	This is ‘T’ for trades
Orderid	8 bytes	uint64_t	Order id of order - unique across symbols

Table 6: Modify - This message is sent when an order is replaced - all shares from original order must be removed and the new quantity is added instead. OrderId, Price, Symbol, Side remains same

Name	Length	Data Type	Notes
MsgType	1 byte	char	This is ‘M’ for modify
Price	8 bytes	uint64_t	New Modified Price of order (remains same for now)
Quantity	8 bytes	uint64_t	New Quantity of Order
OrderId	8 bytes	uint64_t	Orderid of order

Table 7: Query - This message is a query message - whenever you get this message dump a response message corresponding to the symbolId of the query message

Name	Length	Data Type	Notes
MsgType	1 byte	char	Type of message(‘Q’)
SymbolId	4 bytes	uint32_t	Symbolid of the symbol

Notes:

1. This data has been split across different files: data0.dat, data1.dat ... data10.dat
2. OrderIds are unique and incremental across all symbols
3. You have to decode all packets [10 points]
4. You have to read all files and read data in sorted order of timestamp. All files are individually sorted by time. [10 points]
5. You have to create an optimised market book for all symbols Market book is a journal of orders that you use to keep track of market. [10 points]

   Below is an example of sequence of messages and corresponding market book:

- New: symbol: 1, side: Buy, price: 100, qty: 20, orderid:1000

Buy total qty	Buy price	Sell price	Sell total qty
20	100	0	0

- New: symbol: 1, side: Sell, price: 110, qty: 12, orderid:1001

Buy total qty	Buy price	Sell price	Sell total qty
20	100	110	12

- New: symbol: 1, side: Buy, price: 101, qty: 30, orderid: 1003

Buy total qty	Buy price	Sell price	Sell total qty
30	101	110	12
20	100		

- New: symbol: 1, side: Sell, price: 110, qty: 32, orderid: 1005

Buy total qty	Buy price	Sell price	Sell total qty
30	101	110	44
20	100		

- Modify: symbol: 1, side: Buy, price: 101, qty: 10, orderid: 1006

Buy total qty	Buy price	Sell price	Sell total qty
10	101	110	12
20	100		

6. You have to generate an ‘output.dat’ file in binary format that contains responses to any Query in the same order. You can use the given responsereader binary to make sure your output.dat is in correct format. [10 points]

You have to generate responses in the following format:

Table 13: Response - This is the response packet that you need to dump corresponding to each query message

Name	Length	Data Type	Notes
BidPrice	8 bytes	uint64_t	Current Best bid price
BidQty	8 bytes	Uint64_t	Total quantity at best bid price
BidCount	4 bytes	uint32_t	Total number of orders at best bid count
AskPrice	8 bytes	uint64_t	Current best ask price
AskQty	8 bytes	uint64_t	Total quantity at best ask price
AskCount	4 bytes	uint32_t	Total number of orders at best ask price

Example provided below:

Data0.dat

```
[PacketHeader:timestamp:1686674777270400799|numMessage:4]
[New|msgType:N|price:123|quantity:336|symbolid:0|orderid:1|side:1]
[New|msgType:N|price:2100|quantity:10|symbolid:2|orderid:2|side:0]
[New|msgType:N|price:2200|quantity:60|symbolid:2|orderid:3|side:1]
[New|msgType:N|price:2100|quantity:427|symbolid:2|orderid:4|side:0]
```

Data1.dat

```
[PacketHeader:timestamp:1686674777270400890|numMessage:4]
[Trade|msgType:T|orderid:1]
[Query|msgType:Q|symbolid:0]
[Modify|msgType:M|price:2200|quantity:70|orderid:3]
[Query|msgType:Q|symbolid:2]
```

After reading both these files, the final output should consist of 2 responses in output.dat file corresponding to each query: Output.dat

```
[Response:|bidprice:0|bidqty:0|bidcount:0|askprice:0|askqty:0|askcount:0]
[Response:|bidprice:2100|bidqty:437|bidcount:2|askprice:2200|askqty:70|askcount:1]
```

To verify your decoder, you can check the first packet of data0.dat should be:

fileindex:0

```
[PacketHeader |timestamp:1686674777270482713|numMessage:5]
[New|msgType:N|price:100|quantity:997|symbolid:0|orderid:12|side:0]
[New|msgType:N|price:470|quantity:85|symbolid:1|orderid:13|side:1]
```

```
[New|msgType:N|price:100|quantity:847|symbolid:0|orderid:14|side:0]
[Cancel|msgType:C|orderId:12]
[New|msgType:N|price:122|quantity:546|symbolid:0|orderid:15|side:1]
```