Read Me

David Shaw

Unfortunately both of these still need work to become production ready. The 2 reasons are:

1. Endpoints are unreliable and sometimes do not return any data.
2. The secound is that both of these take for granted that each trade that is generated is executed which is not true because of the live nature of orderbooks. Really I should be taking the reply form the exchange and using that to update size etc. I have used IOC limit orders to try and limit damage.

1 Question 2

This was a more difficult problem than I initially thought mostly because of issues connecting to exchanges which are working better but sometimes don't work. Really the best way to do this seems like it would be some kind of concurent program. Where we have seperate process continuily quering the endpoint and updating paramaters as it does so .This is so if the endpoint doesn't return it will mess up the whole script. I have tried to overcome this with time.sleeps and and async programming when I can as in the next question.

There are 6 total cases when looking at this exchange problem for 2 exchanges although some are more likely than others but to make sure that the process is as complete as possible I will account for all of them. Say you want to buy coin1 with coin2 any of the below exchange dynamics are possible:

1. Coin1/Coin2 market exists on both exchanges. In this case amalgamate both exchanges order books and execute.
2. Coin2/Coin1 exists on both, the same as above but flip bids to offers.
3. Coin1/Coin2 on one an Coin2/Coin1 on the other. Invert the offers to turn them into bids (buying in Coin1/Coin2 is selling in Coin2/Coin1) and amalgamate.
4. Coin1/Coin2 only exists on one easy. Just execute on this exchange.
5. Coin2/Coin1 only exists on one.
6. Neither exchange has it just revert (could try a path like) $coin1 \rightarrow coin3 \rightarrow coin2$ but that is beyone the scope.

Initially I figured just to query all endpoints and then keep a dict of previous results to check if the best method for each pair was already figured on a previous attempt like would be done in a dynamic programming problem. Which may be better in practice.

So taking into account all of these possible scenarios we have to be easily able to convert (ask, size) pairs in one market into (bid, size) pairs in another for easy comparison. I have converted all asks to bids but this would have been just as viable if I had converted them all to asks. Converting prices is easy enough as one just takes the reciprocal. If I would buy 1 BNB at .05 BTC it implies that I think that 1 BTC is worth 20 BNB. Quantity is a little harder to see but if we think of a bid as being say 10 BNB at a price of .05 BTC/BNB. That means that I am looking to buy 10 BNB*.05 BTC/BNB = .5 BTC so I would sell .5 BTC at a price of 20 BNB in the BTC/BNB market. The best way to see this I think is below a mapping that maps bid, size (b, s_1) to ask, new size (a, s_2) and that can be done through a function phi, where $\phi(x, y) \to (\frac{1}{x}, xy)$. And we can see

$$\phi(\phi(b, s_1)) \to \phi(\frac{1}{b}, bs_1) \to (\frac{1}{\frac{1}{b}}, \frac{1}{b}bs_1)) \to (b, s_1) \tag{1}$$

So it is entirely symmetic. The idea is to take all of these then take a sumproduct of each of these combined orderbooks and find the one that offers the lowest cost and then send that to the exchange. These are all calculated with fees included as a mult on prices, bringing up asks and bringing bids down. Although in this question we only look at asks.

The main function looks like this *to_execute_main(trading_vol_ftx, trading_vol_bin ,coin_from, coin_to, size)* . The first 2 paramaters take the volume that you trade on each exchange and use that to determine your fee bucket. Then it takes the coin you have and the one you want and takes the total amount of the secound coin that you want and finds the best execution.

Initailly I had planned to also include a dex in the computation like uniswap but didn't because of time limits. I have included a picture illustraating how to compare a Dex to an orderbook exchange. Basically for each price point on an orderbook you figure how much you need to put in or take out to move the price to that point and then what you get out is the size and the quotent of in to out is the price. Then you always execute on the dex between the orderbook prices as your average fill on the dex is always better, until you rech a point where the dex price is the same as the exchanges price. This is because in a dex you are moving the price when buying where as in an orderbook it is all at the same price. Note that your average fill price which is (Tokens in)/(Tokens out) is not the same as the exchage price. (Tokens pool1)/(Tokens pool2)

2 Question 3

There are a few things in this question that should be noted. The first is that we are only looking at coins vs USD because that is the only context being short really makes much sense in and the other is that on Binance there is no USD markets for many coins including bitcoin. Instead I have used BUSD markets (although you can in Euro) . I'm not sure if this is a ploy to generate more trading but I would say it is more likley that it is a regulatory issue. I have had to use asyn clients to pull live Funding data from binance so please await the main function to not get an error. Also I have assumed frictionless tradding between exchanges with no withdrawl fees or that you hold enough colateral in each exchange to always be able to put these positions on.

The main idea is to take the current market we are paying in see how much it cost per day and compaire that to alternatives. I have not taken timing into account as some positions especially in bianance with an 8 hour funding window will be closed before the payment time to avoid payment. This should in general, cause spot and index to move closer as the window draws closer and as such the funding rate to decrease. This is not taken into account. Also we assume that funding is constant. This is obviously not true but it can be modeled as a random walk with the current rate being the mean just like a stock in Black-Scholes. If we ignore the likley patterns mentioned above just as BS ignores stock trends.

We check each of these differences by calcualating IndexPrice*(rate * time) to get how much you are paying and could save. time is an estimate of how much longer you expect to hold the position and is important to figure wether to trade or not. If you are likley closing the short tomorrow paying to change position makes little sense. This is likley unknown but in some case can be given a resonable estimete like people trading ETH-PERP vs stETH on the merge. If you think you will hold forever use np.inf in time.

Then we order these by how much we save and send to the exchange. Here I have not done this optimally as I have spent too much time on this already. I trade as many as I can for the best interest saving. Looking at the fronth of the bid and the ask books seeing if their difference is less than the saving and then trading min(size_bid, siz_ ask) if it is and repeating until I have gone though the book or have done all my size. Then I look at the secound best instrument to hold by interest rate and so on. Really I should not be just looking at rate alone but on each iteration comparing the saving on interest vs bid size and ask size for all books to decide the optimal trade. The other problem with this is that the more time spent checking the higher the chence that the order book has changed. Note we are only looking at asks from the book we are moving from and bids in the book we are moving to.

Then I send these orders to the exchange. On binance loans have to be taken out manually but on FTX it all happens automatically. The main function looks like this *`interest_main(coin, current, total_size, time_days, trading_vol_ftx, trading_vol_bin, mark=0)`* The coin is what you are trading, the total is what the current position is in either `ftx_spot, bin_spot, ftx_future or bin_future` and these are the paramaters that it accepts. The total_size is the amount of the coin you are currently holding short. Time days is the amont of time you intend to hold. The 2 vol paramaters determine fees as before and the mark paramater is there as you are paying interest on a different spot price to what it is currently being traded at. If this is 0 we just use the current mark.