

HETHA.IO project

HOW DOES IT WORK?

Table of contents

What is arbitration?	3
How to make money on arbitration?.....	3
How our product helps you?	3
What problems it is possible to face in arbitration?.....	4
How it works?	4
What do chains look like?	5
Where can you find chains?	6
How to get an access to the server?	7
Where you can find logs?.....	7
Example of chains in logs	7
Setting the parameters	17
Where you have to write your account keys?	21

What is arbitration?

In short, cryptocurrency arbitrage is the profit earned from the price difference of cryptocurrencies across exchanges. By selling on one exchange and buying on another, you have the opportunity to make a profit.

How to make money on arbitration?

The most important factor in arbitrage is speed, making it impractical to execute trades manually. Exchange rates on cryptocurrency exchanges update instantly, so the faster a trade is executed, the higher the potential profit.

To profit from arbitrage, the first step is to analyze which exchanges and currency pairs are profitable at a given moment. How do you know which currency pairs are profitable? It's simple—you just need to check the exchange rates and pick the best option. However, even though the process itself isn't complicated, it takes time, and by the time the market analysis is complete, most potential deals may already be outdated. The next and most challenging step is to execute the trade as quickly as possible. The difficulty at this stage lies solely in the speed of execution.

How our product helps you?

We offer two product variations, depending on what you already have available and how much personal time you're willing to spend on arbitrage.

The first option is the **"Free"** plan. It is completely free. With this plan, the client gains access to the "Market" section. This section contains all the analytics for the data types "Stream Full," "Stream Filtered." Additionally, access to viewing chains and orderbooks is provided. If you are only interested in analytical and statistical data, this option is for you.

The second option is the **"Pro"** plan. In addition to full access to the "Market" section, the client gets access to the "Server" section. This is a fully ready-to-use product that requires nothing else. A robot fully automates market analysis, executes trades, and maintains the balance of currencies across all connected accounts. All trade parameters can be customized in the "Server" - > "Settings" section. This way, each user can make the service more convenient according to their individual needs.

What problems it is possible to face in arbitration?

There are several difficulties encountered in arbitration. The most common include non-closed orders, capital allocation, and identical pairs. Below is a description of these issues:

1. **Non-closed Orders:** These occur when an order has not been filled at the requested price. Since exchange rates change instantly, if an order is placed at a previous price, it may remain nonclosed if the price has since changed. This does not mean the deal will never close, as the currency rate can return to the requested level. However, there are cases where the rate changes significantly, making it unlikely to return to the desired position.
2. **Capital Allocation:** This refers to distributing assets across accounts involved in arbitration. Proper allocation is crucial for efficient transaction execution. While larger deals generally yield higher profits, it's important to ensure that your capital is divided among the accounts participating in the arbitration.
3. **Identical Pairs:** Many traders choose to use the same currency pair for arbitration simultaneously. This issue is challenging to address because there is no centralized database for transactions due to security reasons. When multiple traders use the same pair, the probability of non-closed deals increases significantly because the rate changes after each transaction. Consequently, the more traders involved with the same currency pair, the more the exchange rate will fluctuate.

How it works?

With constant access to exchanges, we can receive order books continuously, forming chains with a delay of a few seconds. If the delay in data transmission from any exchange increases, work with that exchange is temporarily suspended. Chains, which are sequences of potential arbitrage deals occurring sequentially across various cryptocurrency exchanges and based on currency pair transactions, are delivered to customers within an average of 1 second after publication on the exchange. Chains are sent to clients via WebSocket. The maximum size of chains is three, and the maximum depth is 2 BTC.

These chains are referred to as unfiltered. To access them, go to the Downloads section in your personal account and click the Download button. A folder containing all the necessary keys to connect to our server will be downloaded to your PC. Once connected, the chains will begin to be broadcasted. Unfiltered chains are then ready for further use.

The program evaluates each chain based on several parameters, including stability and profitability, and then filters the chains, selecting the best options. These are called filtered chains. To access them, go to the Downloads section in your personal account and click the Download button to download the folder. Connect to our server, and the filtered chains will start arriving with an average delay of 2-4 seconds after being published on the exchange. Filtered chains are now ready for use.

Once a chain has been filtered, the program begins executing arbitrage deals. It can also generate reports for reverse deals, where transactions are made to ensure the number of assets of the same order remains balanced across all connected accounts. There are two types of deals:

- Lite: Transactions occur at the best available price, regardless of volume.
- Full: Deals are executed at a weighted average price. The full type is more profitable for large amounts, while the lite type is suitable for smaller amounts.

What do chains look like?

All chains have the same structure and encoding. Below is a part of the customer chain with an explaining the encoding.

```
[{
  "orderBook": "bfx",
  - name of the exchange (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)
  "pair": "btcusd",
  - currency pair that takes part in the deal
  "amount": 0.016,
  - input deal amount
  "currency": "btc",
  - input currency
  "arr_size": 1,
  - dimension of the part of cup, depending on the type of operation (Bid or Ask) "rate": 12370,
  - rate at which the deal was made
  "side": "sell",
  - input currency operation
  "time": 1597690279466,
  - deal timecode
  "market": {
    "bB": [
      ["12370", 6.23317874]
    ],
  - Bid book – bid – orders list
    "bW": 12370,
  - Bid weight – average-weighted price of all bid orders in a glass "bL": "12370",
  - Bid last – best price
    "bSW": 12370,
  - Not used

  - Not used

  "bSL": "12370", "aB": [
    ["12371", 27.0618301]
  ],
  - Ask book – ask – orders list
    "aW": 12371,
    "aL": "12371",
    "aSW": 12371,
  - average weighted price for reduced position
    "aSL": "12371",
  - Best Ask price for reduced position
    "sV": "0.015",
  - Reduced position volume
    "s": 0,
  - Status – only cups with status "0" are accepted for processing. Statuses can also be 1,2,3 but they are incorrect.
```

- "t": 1597690279466
- Timecode
 - },
 - "market_ping": 0.472,
- connection speed at a given time
 - "average_ping": 0.6154,
- average connection speed with the exchange
 - "long_tick": 17.04413025210084,
- average long-term gap between ticks
 - "short_tick": 6.138285714285715,
- average short-term gap between ticks
 - "current_volume": 0.016,
- normalized deal volume adjusted to BTC "market_volume": 27.0618301,
- normalized cup volume adjusted to BTC
 - "long_volume": 11.721547461176456,
- average long-term market volume
 - "short_volume": 27.60135947428572,
- average short-term market volume
 - "current_deep": 1,
- current market depth
 - "long_deep": 1.1134453781512605,
- long-term market depth
 - "short_deep": 1,
- short-term market depth
 - "estimate": 0.7965347766876221
- parameter is calculated using a neural network taking into account all the above parameters for determining order execution order. The range of the indicator from zero to one. The lower the indicator, the more unstable the market and the faster you need to make a deal. In the case of chain processing by sequential orders, it is recommended that first was executed orders in which the "estimate" parameter tends to zero (less than 0.1).

Where can you find chains?

The server is provided only if you purchase a complete product. If you purchase filtered and unfiltered chains, you need to download them from your personal account in the Downloads section. In these files there is an example code for JS Node, with which you can connect to our server to get chains. If you purchase a complete version you can see examples of chains on the server.

How to get an access to the server?

To log in to the provided server, you need to install a program that will give you access to the server. We recommend using the puTTY program as it is free and quite easy to use. So, after you open the program, you need to enter the IP address for the connection, which is in your personal account in the section Settings/Server. After setting all the desired settings and pressing "OPEN", opens the console. There are requested your login and password, which you also can find in your personal account in the same section. Then press "Enter" and get an access to the server. All the folders and files you are interested in are located in the /home/trader folder.

Where you can find logs?

After you have installed the program to connect to the server, opened it and entered all necessary data (they are placed in your personal account in the Server/Logs section), you need to write the mc command, thereby opening the file manager. This path follows /home/trader/logs. Now you need to choose the date you are interested in and then the number of the group of logs.

Example of chains in logs

Now we are looking at several variants of the logs you can meet.

The first is locked chains. What does this mean and why are chains blocked? The minimum allowable profit in this example that our program passes is for Full - 0.1%, and for Lite - 0.2%. If you want to change these parameters, you need to enter the params.json file. You need to change the value in the line "percent_accept". If the chain does not satisfy the minimum profit requirements, it is automatically blocked. Also, the chain can be temporarily blocked until the balance information on the accounts is updated. Let see the example:

Chain №1

Incoming chain: lite profit: 100.18528302013945 chain: bnc:btc/usd-bts:xrp/usd-bnc:xrp/btc time: 2.342

That's actually is the chain itself. lite – it a type of chain (Remember that there are Lite and Full) profit: 100.18528302013945 – it is the profit that this chain can make. As we see, profit is 0.18%, respectively, the chain is blocked. chain: bnc:btc/usd-bts:xrp/usd-bnc:xrp/btc - a chain that takes part in deal. bnc: btc/usd-bts: xrp/usd-bnc: xrp/btc - cryptocurrency exchange (bfx - Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex - CEX.IO): currency pair that takes part in the first deal (btc/usd)

- cryptocurrency exchange: currency pair that takes part in the second deal (xrp/usd) - cryptocurrency exchange: currency pair that takes part in the third deal (xrp/btc).

time: 2.342 – timecode

Chain №2

NOT BLOCKED: Chain: bnc:btc/usd-bts:xrp/usd-bnc:xrp/btc timestamp: 1606377703681 {"orderBook":"bnc","Pair":"btcusd","Side":"sell","Timestamp":1606377703681,"b_lock":0,"a_lock":0,"Status":"Unlocked"} Check equity sell direct:

{"orderBook":"bnc","pair":"btcusd","amount":0.032,"currency":"btc","arr_size":20,"rate":17563.57822095,"side":"sell","time":1606377700519} Balance a: 0.09880529

Amount LESS than balance. Accepted

Actual price is more that acceptable (type BUY): bts:xrpusd:0.51302:0.51353302:0.513648

NOT BLOCKED - the same chain was not launched during the period specified in the parameters.

Chain: bnc:btc/usd-bts:xrp/usd-bnc:xrp/btc – *the chain that takes part in deals. bnc: btc/usd-bts: xrp/usd-bnc: xrp/btc - cryptocurrency exchange (bfx - Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex - CEX.IO): currency pair that takes part in the first deal (btc/usd) - cryptocurrency exchange: currency pair that takes part in the second deal (xrp/usd) - cryptocurrency exchange: currency pair that takes part in the third deal (xrp/btc).*

timestamp: 1606377703681 – *timecode*

{"orderBook":"bnc","Pair":"btcusd","Side":"sell","Timestamp":1606377703681,"b_lock":0,"a_lock":0,"Status":"Unlocked"}

{"orderBook":"bnc" – *exchange from which the orderbook was taken*

"Pair":"btcusd" – *currency pair*

"Side":"sell" – *input currency operation*

"b_lock":0 – *if the timestamp is greater than the current timestamp, the first element of the currency pair is temporarily locked*

"a_lock":0 – *if the timestamp is greater than the current timestamp, the second element of the currency pair is temporarily locked*

"Status":"Unlocked" – *Chain status – not locked because of using this currency after the last balance update.*

Now the program is checking sold currency.

Check equity sell direct:

{"orderBook":"bnc","pair":"btcusd","amount":0.032,"currency":"btc","arr_size":20,"rate":17563.57822095,"side":"sell","time":1606377700519} Balance a: 0.09880529

Amount LESS than balance. Accepted

{"orderBook":"bnc" - *exchange from which the orderbook was taken (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)*

"pair":"btcusd" – *currency pair*

"amount":0.032 – *amount of sold currency*

"currency":"btc" – *input currency*

"arr_size":20 – *dimension of the part of cup, depending on the type of operation (Bid or Ask)*

"rate":17563.57822095 – *rate at which the deal was made*

"side":"sell" – *deal with input currency*

"time":1606377700519} – *timestamp*

Balance a: 0.09880529 – *balance of sold currency*

Amount LESS than balance. Accepted – *the amount of the sold currency must not exceed the set value of the total balance on the currency account. You can change this parameter in the settings.*

Actual price is more than that acceptable (type BUY): - *parameter of deviation in the cup. Initially it is set 0.05. You can change the parameter in your personal account.*
bts:xrpusd:0.51302:0.51353302:0.513648

1st parameter - *rate at which the deal is to be made (rate from the chain)*

2nd parameter - *calculated allowed deviation*

3rd parameter - *current deviation*

The second option is not blocked chains, which take part in deals.

Incoming chain: lite profit: 100.25263472578926 chain: bts:btc/usd-bnc:xlm/btc-bnc:xlm/usd time: 2.206 - цепочка

NOT BLOCKED – the same chain was not launched during the time period specified in the parameters.

Chain: bts:btc/usd-bnc:xlm/btc-bnc:xlm/usd - *the chain that takes part in deals.*

timestamp: 1606379598874 – *timecode*

```
{"orderBook":"bts","Pair":"btcusd","Side":"buy","Timestamp":1606379598874,"b_lock":0,"a_lock":0,"Status":"Unlocked"}
```

{"orderBook":"bts" - exchange from which the orderbook was taken (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)

"Pair":"btcusd" – currency pair

"Side":"buy" – input currency operation

"Timestamp":1606379598874 – timecode

"b_lock":0 – if the timestamp is greater than the current timestamp, the first element of the currency pair is temporarily locked

"a_lock":0 – if the timestamp is greater than the current timestamp, the second element of the currency pair is temporarily locked

"Status":"Unlocked" – Chain status – not locked because of using this currency after the last balance update.

Now the program is checking sold currency.

Check equity sell opposite:

```
{"orderBook":"bts","pair":"btcusd","amount":200,"currency":"usd","arr_size":4,"volume":"0.22000000","rate":16615.54,"side":"buy","time":1606379596668} Balance b: 444.41
```

"pair":"btcusd" – currency pair

"amount":200– sold currency amount

"currency":"usd" – input currency

"arr_size":4 – dimension of the part of cup, depending on the type of operation (Bid or Ask)

"volume":"0.22000000" – cup volume reduced to BTC

"rate":16615.54, - rate at which the deal was made

"side":"buy" - deal with input currency

"time":1606379596668 – timecode

Balance b: 444.41 – sold currency balance

Amount LESS than balance. Accepted – *the amount of the sold currency must not exceed the set value of the total balance on the currency account. You can change this parameter in the settings.*

```
{"orderBook":"bnc","Pair":"xlmBTC","Side":"buy","Timestamp":1606379598875,"b_lock":0,"a_lock":0,"Status":"Unlocked"}
```

{"orderBook":"bts" - exchange from which the orderbook was taken (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)

"pair":"xlmBTC" – currency pair

"Side":"buy" - deal with input currency

"Timestamp":1606379598875 – timecode

"b_lock":0 – if the timestamp is greater than the current timestamp, the first element of the currency pair is temporarily locked

"a_lock":0 – if the timestamp is greater than the current timestamp, the second element of the currency pair is temporarily locked

"Status": "Unlocked" – Chain status – not locked because of using this currency after the last balance update.

Now the program is checking sold currency Check

equity sell opposite:

```
{"orderBook": "bnc", "pair": "xlbtc", "amount": 0.01201285062056364, "currency": "btc", "arr_size": 5, "volume": 10020, "rate": 0.00000896, "side": "buy", "time": 1606379596483} Balance b: 0.11486897
```

"pair": "xlbtc" – currency pair

"amount": 0.01201285062056364 – sold currency pair

"currency": "btc" – input currency

"arr_size": 5 - dimension of the part of cup, depending on the type of operation (Bid or Ask)

"volume": "10020" – cup volume reduced to BTC

"rate": 0.00000896 – rate at which the deal was made

"side": "buy" – deal with input currency

"time": 1606379596483 – timecode

Balance b: 0.11486897 – sold currency balance

Amount LESS than balance. Accepted – the amount of the sold currency must not exceed the set value of the total balance on the currency account. You can change this parameter in the settings.

```
{"orderBook": "bnc", "Pair": "xlmusd", "Side": "sell", "Timestamp": 1606379598875, "b_lock": 0, "a_lock": 0, "Status": "Unlocked"}
```

"orderBook": "bnc" - exchange from which the orderbook was taken (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)

"Pair": "xlmusd" – currency pair

"Side": "sell" – deal with input currency

"Timestamp": 1606379598875 – timecode

"b_lock":0 – if the timestamp is greater than the current timestamp, the first element of the currency pair is temporarily locked

"a_lock":0 – if the timestamp is greater than the current timestamp, the second element of the currency pair is temporarily locked

"Status": "Unlocked" – Chain status – not locked because of using this currency after the last balance update.

Now the program is checking sold currency Check

equity sell direct:

```
{"orderBook": "bnc", "pair": "xlmusd", "amount": 1339.3792153954325, "currency": "xlm", "arr_size": 8, "volume": 4087.3, "rate": 0.14985, "side": "sell", "time": 1606379595503} Balance a: 4145.92365885
```

"pair": "xlmusd" – currency pair

"amount": 1339.3792153954325 – sold currency pair

"currency": "xlm" – input currency

"arr_size": 8 - dimension of the part of cup, depending on the type of operation (Bid or Ask)

"volume": "4087.3" – cup volume reduced to BTC

"rate":0.14985 – rate at which the deal was made
"side":"sell" – deal with input currency
"time":1606379595503 – timecode
Balance a: 4145.92365885 – bought currency balance

Amount LESS than balance. Accepted – *the amount of the sold currency must not exceed the set value of the total balance on the currency account. You can change this parameter in the settings.*

Node sequence estimation: 0.32777199149131775
Node sequence estimation: 0.2544032037258148 Node
sequence estimation: 0.3438597321510315

The parameter is calculated using a neural network, taking into account all the above parameters for determining the execution order. The indicator ranges from zero to one. The lower the indicator, the more unstable the market, and the faster a deal needs to be made. In the case of chain processing by sequential orders, it is recommended that orders in which the "estimate" parameter tends toward zero be executed first (less than 0.1).

```
{ "0": { "chain": { "orderBook": "bts", "pair": "btcusd", "amount": 200, "currency": "usd", "arr_size": 4, "volume": "0.22000000", "rate": 16615.54, "side": "buy", "time": 1606379596668, "market_ping": 0.288, "average_ping": 0.26513513513513526, "long_tick": 3.3233552400947666, "short_tick": 1.8105900621118012, "current_volume": 0.0075448500000000005, "market_volume": 0.15, "long_volume": 0.7094511138309072, "short_volume": 2.4360050367701853, "current_deep": 1, "long_deep": 1, "short_deep": 1, "estimate": 0.32777199149131775,
```

"orderBook": "bts",

- *name of the exchange (bfX – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO) "pair": "btcusd",*
- *currency pair that takes part in the deal*
"amount": 200,
- *input deal amount*
"currency": "usd",
- *input currency*
"arr_size": 4,
- *dimension of the part of cup, depending on the type of operation (Bid or Ask) "rate": 16615.54,*
- *rate at which the deal was made*
"side": "buy",
- *input currency operation*
"time": 1606379596668,
- *deal timecode*
"market_ping": 0.288,
- *current connection speed*
"average_ping": 0.26513513513513526,
- *average connection speed with the exchange*
"long_tick": 3.3233552400947666,
- *average long-term gap between ticks*
"short_tick": 1.8105900621118012,
- *average short-term gap between ticks*
"current_volume": 0.0075448500000000005,
- *normalized deal volume adjusted to BTC "market_volume": 0.15,*

- *normalized cup volume adjusted to BTC*
"long_volume": 0.7094511138309072,
- *average long-term market volume*
"short_volume": 2.4360050367701853,
- *average short-term market volume*
"current_deep": 1,
- *current market depth*
"long_deep": 1,
- *long-term market depth* "short_deep": 1,
- *short-term market depth*
"estimate": 0.32777199149131775
- The parameter is calculated using a neural network, taking into account all the above parameters for determining the execution order. The indicator ranges from zero to one. The lower the indicator, the more unstable the market, and the faster a deal needs to be made. In the case of chain processing by sequential orders, it is recommended that orders in which the "estimate" parameter tends toward zero be executed first (less than 0.1).

```
{
  "chain": {
    "orderBook": "bnc",
    "pair": "xlmBTC",
    "amount": 0.01201285062056364,
    "currency": "BTC",
    "arr_size": 5,
    "volume": 10020,
    "rate": 0.00000896,
    "side": "buy",
    "time": 1606379596483,
    "market_ping": 0.516,
    "average_ping": 0.7484459459459453,
    "long_tick": 3.191377478543947,
    "short_tick": 1.82190625,
    "current_volume": 0.0075448500000000005,
    "market_volume": 0.0075448500000000005,
    "long_volume": 0.2913235742113055,
    "short_volume": 0.1484563523749999,
    "current_deep": 1,
    "long_deep": 1,
    "short_deep": 1,
    "estimate": 0.2544032037258148,
    "orderBook": "bnc",

```

- *name of the exchange (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)* "pair": "xlmBTC",
- *currency pair that takes part in the deal*
"amount": 0.01201285062056364,
- *input deal amount*
"currency": "BTC",
- *input currency*
"arr_size": 5,
- *dimension of the part of cup, depending on the type of operation (Bid or Ask)* "rate": 0.00000896,,
- *rate at which the deal was made*
"side": "buy",
- *input currency operation*
"time": 1606379596483,
- *deal timecode*
"market_ping": 0.516,
- *current connection speed*
"average_ping": 0.7484459459459453,
- *average connection speed with the exchange*
"long_tick": 3.191377478543947,
- *average long-term gap between ticks*
"short_tick": 1.82190625,
- *average short-term gap between ticks*
"current_volume": 0.0075448500000000005,

- normalized deal volume adjusted to BTC
"market_volume": 0.0075448500000000005,
- normalized cup volume adjusted to BTC
"long_volume": 0.2913235742113055,
- average long-term market volume
"short_volume": 0.1484563523749999,
- average short-term market volume
"current_deep": 1,
- current market depth
"long_deep": 1,
- long-term market depth
"short_deep": 1,
- short-term market depth
"estimate": 0.2544032037258148
- The parameter is calculated using a neural network, taking into account all the above parameters for determining the execution order. The indicator ranges from zero to one. The lower the indicator, the more unstable the market, and the faster a deal needs to be made. In the case of chain processing by sequential orders, it is recommended that orders in which the "estimate" parameter tends toward zero be executed first (less than 0.1).

This text informs us about the currency pairs that are currently blocked. They are blocked due to the initiation of a new chain. These currency pairs will remain blocked until the balance information is updated.

```
"ltcbtc":["b_lock",1606379778875],"ethbtc":["b_lock",1606379778875],"zecbtc":["b_lock",1606379778875],"dashbtc":["b_lock",1606379778875],"xrpbtc":["b_lock",1606379778875],"xmrbtc":["b_lock",1606379778875],"btcusd":["a_lock",1606379778875],"eosbtc":["b_lock",1606379778875],"xlmbtc":["b_lock",1606379778875],"trxbtc":["b_lock",1606379778875],"neobtc":["b_lock",1606379778875],"etcbtc":["b_lock",1606379778875]},"lockedCurrency":"btc"},"2"
```

```
{"chain":{"orderBook":"bnc","pair":"xlmusd","amount":1339.3792153954325,"currency":"xlm","arr_size":8,"volume":4087.3,"rate":0.14985,"side":"sell","time":1606379595503,"market_ping":0.516,"average_ping":0.7484459459459453,"long_tick":3.1471942980136074,"short_tick":1.83337106918239,"current_volume":0.0075448500000000005,"market_volume":0.6962812594888916,"long_volume":0.10486859781497908,"short_volume":0.16531981790056835,"current_deep":1,"long_deep":1,"short_deep":1,"estimate":0.3438597321510315,
```

"orderBook": "bnc",

- *name of the exchange (bfx – Bitfinex; bnc – Binance; btx – Bitrex; bts – Bitstamp; cex – CEX.IO)*
"pair": "xlmusd",
- *currency pair that takes part in the deal*
"amount": 1339.3792153954325,
- *input deal amount*
"currency": "xlm",
- *input currency*
"arr_size": 8,
- *dimension of the part of cup, depending on the type of operation (Bid or Ask)*
"rate": 0.14985,

- *rate at which the deal was made*
"side": "sell",
- *input currency operation*
"time": 1606379595503
- *deal timecode*
"market_ping": 0.516,
- *current connection speed*
"average_ping": 0.7484459459459453,
- *average connection speed with the exchange*
"long_tick": 3.1471942980136074,
- *average long-term gap between ticks*
"short_tick": 1.83337106918239,
- *average short-term gap between ticks*
"current_volume": 0.0075448500000000005,
- *normalized deal volume adjusted to BTC*
"market_volume": 0.6962812594888916,
- *normalized cup volume adjusted to BTC*
"long_volume": 0.10486859781497908,
- *average long-term market volume*
"short_volume": 0.16531981790056835,
- *average short-term market volume*
"current_deep": 1,
- *current market depth*
"long_deep": 1,
- *long-term market depth*
"short_deep": 1,
- *short-term market depth*
"estimate": 0.3438597321510315
- The parameter is calculated using a neural network, taking into account all the above parameters for determining the execution order. The indicator ranges from zero to one. The lower the indicator, the more unstable the market, and the faster a deal needs to be made. In the case of chain processing by sequential orders, it is recommended that orders in which the "estimate" parameter tends toward zero be executed first (less than 0.1).

Now the chain is ready to start. Below is the represented the described chain. The code is exactly the same as in the chain above.

Chain is ready to start:

```
{
  "lite": true,
  "history": [
    {
      "orderBook": "bts",
      "pair": "btcusd",
      "amount": 200,
      "currency": "usd",
      "arr_size": 4,
      "volume": "0.22000000",
      "rate": 16615.54,
      "side": "buy",
      "time": 1606379596668,
      "market_ping": 0.288,
      "average_ping": 0.26513513513513526,
      "long_tick": 3.3233552400947666,
      "short_tick": 1.8105900621118012,
      "current_volume": 0.0075448500000000005,
      "market_volume": 0.15,
      "long_volume": 0.7094511138309072,
      "short_volume": 2.4360050367701853,
      "current_deep": 1,
      "long_deep": 1,
      "short_deep": 1,
      "estimate": 0.32777199149131775,
      "btcusd": [
        "b_lock",
        1606379778875
      ],
      "ethusd": [
        "b_lock",
        1606379778875
      ],
      "ltcusd": [
        "b_lock",
        1606379778875
      ],
      "xrpusd": [
        "b_lock",
        1606379778875
      ],
      "bchusd": [
        "b_lock",
        1606379778875
      ]
    },
    {
      "orderBook": "bnc",
      "pair": "xlbtc",
      "amount": 0.01201285062056364,
      "currency": "btc",
      "arr_size": 5,
      "volume": 10020,
      "rate": 0.00000896,
      "side": "buy",
      "time": 1606379596
    }
  ]
}
```

483,\"market_ping\":0.516,\"average_ping\":0.7484459459459453,\"long_tick\":3.1913774785439477,\"short_tick\":1.82190625,\"current_volume\":0.0075448500000000005,\"market_volume\":0.0075448500000000005,\"long_volume\":0.2913235742113055,\"short_volume\":0.1484563523749999,\"current_deep\":1,\"long_deep\":1,\"short_deep\":1,\"estimate\":0.2544032037258148,\"lcbtc\":[\"b_lock\",1606379778875],\"ethbtc\":[\"b_lock\",1606379778875],\"zecbtc\":[\"b_lock\",1606379778875],\"dashbtc\":[\"b_lock\",1606379778875],\"xrpbtc\":[\"b_lock\",1606379778875],\"xmrbtc\":[\"b_lock\",1606379778875],\"btcusd\":[\"a_lock\",1606379778875],\"eosbtc\":[\"b_lock\",1606379778875],\"xlmbtc\":[\"b_lock\",1606379778875],\"trxbtc\":[\"b_lock\",1606379778875],\"neobtc\":[\"b_lock\",1606379778875],\"etcbtc\":[\"b_lock\",1606379778875]},{\"orderBook\":\"bnc\", \"pair\":\"xlmusd\", \"amount\":1339.3792153954325, \"currency\":\"xlm\", \"arr_size\":8, \"volume\":4087.3, \"rate\":0.14985, \"side\":\"sell\", \"time\":1606379595503, \"market_ping\":0.516, \"average_ping\":0.7484459459459453, \"long_tick\":3.1471942980136074, \"short_tick\":1.83337106918239, \"current_volume\":0.0075448500000000005, \"market_volume\":0.6962812594888916, \"long_volume\":0.10486859781497908, \"short_volume\":0.16531981790056835, \"current_deep\":1, \"long_deep\":1, \"short_deep\":1, \"estimate\":0.3438597321510315, \"xlmusd\":[\"a_lock\",1606379778875]}], \"deep\":3, \"profit\":100.25263472578926, \"chain\":\"bts:btc/usd-bnc:xlm/btc-bnc:xlm/usd\", \"base\":\"usd\", \"num\":\"2\", \"created\":1606379598874, \"time\":1606379596668}

Limit order - Market: bts pair: btcusd amount: 0.01203692 rate: 16615.54 type: buy
 Limit order - Market: bnc pair: XLMBTC amount: 1341 rate: 0.00000896 type: buy rate round: 8 lot round: 0
 Limit order - Market: bnc pair: XLMUSD amount: 1339.4 rate: 0.14985 type: sell rate round: 5 lot round: 1

Limit order

– order type Market:

bts

- name of the exchange (*bf*x – Bitfinex; *bnc* – Binance; *btx* – Bitrex; *bts* – Bitstamp; *cex* – CEX.IO) "pair": "xlmusd",
- currency pair that takes part in the deal
"amount": 1339.4,
- input deal amount
"rate": 0.14985,
- rate at which the deal was made
"side": "sell",
- input currency operation

Replies from exchanges BTS

response:

```
{\"id\":\"1300662781132800\", \"price\":\"16615.54\", \"amount\":\"0.01203692\", \"pair\":\"btcusd\", \"type\":\"buy\", \"response\":{\"\"price\": \"16615.54\", \"amount\": \"0.01203692\", \"type\": \"0\", \"id\": \"1300662781132800\", \"datetime\": \"2020-11-26 08:33:18.990621\"}}
```

"id": "1300662781132800" – deal id
"price": "16615.54" – purchase price
"amount": "0.01203692" – input deal amount
"pair": "btcusd" – currency pair
"type": "buy" – deal type (input currency)

","datetime": "2020-11-26 08:33:18.990621 – order time

Order processing spent time: 0.132 Chain processing time: 2.338

Order processing spent time: 0.132 – order processing time

Chain processing time: 2.338 – chain processing time

This text informs us that the position has been placed, but the deal has not yet been executed because the order type is Limit, and the price has already changed. The deal will be completed as soon as the rate returns to its previous value.

BNC response:

```
{ "id": 192832386, "price": 0, "amount": "0.00000000", "pair": "xlmBTC", "type": "buy", "response": { "symbol": "XLMBTC", "orderId": 192832386, "orderListId": -1, "clientOrderId": "jDIE3MJh4PLUX5sADhtn4D", "transactTime": 1606379599011, "price": "0.00000896", "origQty": "1341.00000000", "executedQty": "0.00000000", "cumulativeQuoteQty": "0.00000000", "status": "NEW", "timeInForce": "GTC", "type": "LIMIT", "side": "BUY", "fills": [] } }
```

"id": "192832386" – deal id
"price": "0" – purchase price
"amount": "0.00000000" – input deal amount
"pair": "xlmBTC" – currency pair
"type": "buy" – deal type (input currency)
"response" – reply from the exchange. All values before "response" decrypt the response from the exchange.

Order processing spent time: 0.971 – order processing time

Chain processing time: 3.177 – chain processing time

BNC response:

```
{ "id": 391186997, "price": 0, "amount": "0.00000000", "pair": "xlmUSD", "type": "sell", "response": { "symbol": "XLMUSDT", "orderId": 391186997, "orderListId": -1, "clientOrderId": "XDJXMyT6yIzPDOW2vxv6NI", "transactTime": 1606379599016, "price": "0.14985000", "origQty": "1339.40000000", "executedQty": "0.00000000", "cumulativeQuoteQty": "0.00000000", "status": "NEW", "timeInForce": "GTC", "type": "LIMIT", "side": "SELL", "fills": [] } }
```

"id": "391186997" – deal id
"price": "0" – purchase price
"amount": "0.00000000" – input deal amount
"pair": "xlmUSD" – currency pair
"type": "sell" – deal type (input currency)
"response" – reply from the exchange. All values before "response" decrypt the response from the exchange.

Order processing spent time: 0.975 – order processing time

Chain processing time: 3.181 – chain processing time

Setting the parameters

When you get a product, you also get a set of default settings, but you can change them if you want. To do this, go to the server, go to the/home/arby.trader/libs folder. There you will find the file called params.json. To change the settings, you must open the file and change its values to the desired.

Below are all these parameters with an explanation of the values:

ROOT Module

- "release": 2.4.5,
 - Version.
- "mode pro": true,
 - Version type.
- "debug": false,
 - Enable/disable system messages.
- "lock_timeout": 3,
 - Timeout when new chains using the same currency as the previous one will not start. The time is in milliseconds, it means that by default it is set 3 minutes from the last deal in the last chain to start a new deal in the new chain using the same currency as a previous one.
- "percent_balance": 0.9,
 - The deal amount must not exceed the specified value of funds on the account. By default it is set 90%.
- "gentime limit": 1,
 - Max. allowable chain generation time, sec.
- "profit limit": 110,
 - Display in the logs the chains with profitability from VALUE, %.
- "chainlog limit": 1000,
 - Display VALUE rows in chains.log.
- "ws interval": 1,
 - Reconnect to the server in VALUE, min.
- "balance interval": 2,
 - Balance information is requested every VALUE, min.
- "allow get balance": true,
 - Enable/disable displaying balances in the personal account.
- "allow update": false,
 - Enable/disable "Update" button.
- "callback port": 3003,
 - Callback port.
- "ws url": ws://feed.hetha.io:8080/ws/,
 - Server address.

WS PORT Module

- "pro": 8080,
 - Pro version port.
- "standard": 8000,
 - Standard version port.

WS PARAMS Module

- "event": cups,chains,
Event - If "chain" is specified, all chains with profitability greater than 100% will be transmitted. If the minimum profitability percentage is higher, it is specified as "chain_100.05". "cups" - all cups are transmitted.
- "market": bnc knc bbt bfx,
Receive data from VALUE exchanges.
- "currency": btc usd ada avax eth sol sui trx, -
Receive chains with VALUE currencies.
- "profit": 100.02,
Receive chains with profitability above VALUE, %.

CHAIN BASE Module

- "0,1,2": btc usd
Base currency. If the chain doesn't include the base currency, it won't be processed.
- "balance": true,
Permit or prohibit the retention of balance information.
- "balance backup hour": 18,
Balance information is saved every day at VALUE, h. "chain": true,
- Permit or prohibit the retention of chains.
- "order": true,
Enable/disable order saving.

AI Module

- "accept": true,
Enable/disable AI usage. If this setting is enabled, order data will be available to the service for model training.
- "pro enable": true,
Enable/disable forecasting of chain execution time.
- "pro threshold": 0,
Threshold value for chain execution time. Below this value - allow execution, above this value - prohibit execution, min.
- "base enable": true,
Enable/disable forecasting of chain execution. "base threshold": 0,
- Threshold value. Chains are not executed with a coefficient higher than the VALUE.

FORCE REVERSE Module

- "accept": true,
Enable/disable balance distribution mode.
- "chain length": 2,
Max. allowable chain length.
- "percent_accept": 0,
Min. allowable profit for the reverse deal.
- "activator": 4,
Force reverse is activated if the number of base dimensions (\$100 or equivalent) is less than

VALUE.

- "multiplier": 8,
- Increase the sum of the chain by VALUE times relative to the base dimension (\$100 or equivalent).

LITE Module

- "accept": true,
- If the previous conditions were executed or were not executed, the chain is either accepted or not accepted.
- "percent_accept": 0.2,
- Minimum profit is not less than the specified value. By default it is set 0.2%. "force_trading": 3,
- Use "Market" order placement type if profit exceeds set value (%). "rate_diff": 0.05,
- The chain is accepted if the deviation from the initial rate is not more than the specified value (%) "multiplier": 2

TREND Module

- "accept": false,
- Accept Trend deals.
- "usd": long,
- Long - Buy at the current price (limited order) and sell when the price increases by «Offset by VALUE, %». Short - Sell at the current price (limited order) and buy when the price decreases by «Offset by VALUE, %». Flat - Sell at price « $\frac{1}{2}$ *Offset by VALUE, %» higher than current, and buy « $\frac{1}{2}$ *Offset by VALUE, %» lower.
- "btc": flat,
- Long - Buy at the current price (limited order) and sell when the price increases by «Offset by VALUE, %». Short - Sell at the current price (limited order), and buy when the price decreases by «Offset by VALUE, %». Flat - Sell at price « $\frac{1}{2}$ *Offset by VALUE, %» higher than current, and buy « $\frac{1}{2}$ *Offset by VALUE, %» lower.
- "force against": true,
- If True is selected, then:
Long – Buy at the current price (market order) and sell when the price increases by «Offset by VALUE, %».
Short - Sell at the current price (market order), and buy when the price decreases by «Offset by VALUE, %». "offset by": 0.2,
- Sale or purchase when the price changes by VALUE, %.

FULL Module

- "accept": true,
- If the previous conditions were executed or were not executed, the chain is either accepted or not accepted.
- "percent_accept": 0.1,
- Minimum profit is not less than the specified value. By default it is set 0.1%. "force_trading": 1.5,
- Use "Market" order placement type if profit exceeds set value (%). "rate_diff": 0.1,

- The chain is accepted if the deviation from the initial rate is not more than the specified value (%)
"multiplier": 4
- Multiplies chain sum on a specified number of times.

Where you have to write your account keys?

To start working with the program, you need to enter the necessary data for connection to accounts on exchanges. If these keys set on the rented server, no one except you has an access to them. To do this, go to the server, go to the /home/arby.trader/libs folder. There you will find the file params.json. When you open the file, you will see a list of exchanges and places where you need to enter the keys.

```
"bfx": {
    "key": ".....",
    "secret": "....."
- Bitfinex keys },
  "bnc": {
    "key": " ..... ",
    "secret": " ..... "
- Binance keys
  },
  "bts": {
    "key": " ..... ",
    "secret": " ..... ",
    "client_id": " ..... ",
    "timeout": .....,
    "host": www.bitstamp.net
- Bitstamp keys
  },
  "btx": {
    "key": " ..... ",
    "secret": " ..... "
- Bittrex keys },
  "cex": {
    "user_id": " ..... ",
    "key": " ..... ",
    "secret": "....."
- CEX.IO keys
```

API. The result of a call to the server

Here are some examples. The latest data can be found in your personal account.

- **server/credentials** - Getting script settings.

Description:

Returns the server settings that are in the personal account. The personal account settings are set in JSON format. Note, that in the personal account the correctness of the JSON format is not checked. Therefore, an incorrect format may cause an error when running the script.

Incoming Parameters:

—

Result:

```
{
  "lock_timeout": 180000,
  "percent_balance": 0.9,
  "lite": {
    "accept": true,
    "percent_accept": 0.2,
    "force_trading": 3,
    "rate_diff": 0.05,
    "multiplier": 2
  },
  "full": {
    "accept": true,
    "percent_accept": 0.1,
    "force_trading": 1.5,
    "rate_diff": 0.1,
    "multiplier": 4
  },
  "bfx": {
    "key": "",
    "secret": ""
  },
  "bnc": {
    "key": "",
    "secret": ""
  },
  "bts": {
    "key": "",
    "secret": "",
    "client_id": "", "timeout":
    10000,
    "host": "www.bitstamp.net"
  },
  "btx": {
    "key": "",
    "secret": ""
  },
  "cex": {
    "user_id": "",
    "key": "",
    "secret": "" }
}
```

Call Example:

```
api.request('server/credentials', {}, function ( response ) {});
```

- **save/chain** – Saving chains to appear in the user's personal account.

Description:

The request transfers the data in a chain that need to be saved to display information in the user's personal account. All information in chains is transferred directly from the chains, but the fields orderId (varchar 25), status (varchar 50), comment (text) can be arbitrary and transfer any value the user need.

Incoming Parameters:

lite – 0 (full) or 1 (lite); orderId
– arbitrary field;
history – JSON, The whole chain's history;
deep – deep of the chain; profit – chain's
profit; chain – edited chain's transcript;
status – arbitrary field; comment - arbitrary
field;

Result:

Ok — no saving's errors.
Error — in any other cases.

Call Example:

```
var params = {'lite': feed.lite, 'orderId': orderId, 'history': feed.history, 'deep': feed.deep, 'profit':  
feed.profit, 'chain': feed.chain, status:"done", comment: ""}; api.request('save/chain',  
params, function ( response ) {});
```

- **market/status** – Obtaining the current status of the exchanges.

Description:

The request returns JSON with data about all exchanges, including the speed of connection, average speed parameters for the last period, e.t.c. Incoming Parameters: –

Result:

```
[  
  {  
    id: 1,  
    short_name: 'bts',  
    name: 'Bitstamp',  
    av_ping: 1.53,  
    last_ping: 1.8,  
    uptime: 100,  
    status: 0 },
```

```

{      id:      2,
  short_name: 'bfx',
  name: 'Bitfinex',
  av_ping: 1.522,
  last_ping: 1.878,
  uptime: 100,
  status: 0 },
{      id:      3,
  short_name: 'bnc',
  name: 'Binance',
  av_ping: 0.474,
  last_ping: 0.486,
  uptime: 100,
  status: 0 },
{      id:      4,
  short_name: 'btx',
  name: 'Bittrex',
  av_ping: 0.551,
  last_ping: 0.52,
  uptime: 100,
  status: 0 },
{      id:      5,
  short_name: 'cex',
  name: 'CEX.IO',
  av_ping: 0.686,
  last_ping: 0.786,
  uptime: 100,
  status: 0 }
]

```

Call Example:

```
api.request('market/status', {}, function ( response ) {});
```

- **report/chains** – Obtaining data from a chain analyzer. Data can also be obtained in the personal account.

Description:

The request returns JSON with data to the direct and reverse chains.

Incoming Parameters:

Chain – The parameter fetches a portion of the chain. You can use different choices to search the chain field. Example "xrp," "bnc: trx," "usd-cex," etc.;

dateFrom, dateTo — the search takes place between these dates. Parameter value " - remove the search boundary. Note that the search for the specified parameters can be extremely long. Therefore, the call should be done carefully to avoid blocking the request. In the case of frequent or long requests, this API feature may be restricted in use; lite – permissible values: -1 – leave out, 0 — full, 1 – lite;

profit – transactions profitability is not lower than declared; reverse – true/
false - check reverse chains within the same time period.

What is a reverse chain?

If a new chain completely mirrors a direct chain in reverse order, it is called a reverse chain. These chains are necessary to understand the frequency of events where funds transferred to one market using a direct chain will return using reverse chains. The presence of reverse chains automatically balances the distribution of funds across markets. Ideally, there should be an equal number of direct and reverse chains. Examples of direct and reverse chains:
bts:bch/usd-cex:bch/usd и cex:bch/usd-bts:bch/usd

Result:

```
[
  { chain: 'btx:btc/usd-bfx:xrp/usd-bnc:xrp/btc',
    direct_count: 2, direct_profit: 0.6,
    direct_days: 1,
    direct_history: '2020-09-03:2', reverse_count: 10,
    reverse_profit: 3.05, reverse_days: 3, reverse_history: '2020-
    09-02:3,2020-09-03:6,2020-09-23:1' },
  { chain: 'btx:xrp/btc-bfx:xrp/usd-bnc:btc/usd',
    direct_count: 5, direct_profit: 1.68,
    direct_days: 2,
    direct_history: '2020-09-02:3,2020-09-
    23:2', reverse_count: 1, reverse_profit: 0.23,
    reverse_days: 1, reverse_history: '2020-09-
    03:1' },
  { chain: 'cex:btc/usd-cex:xrp/usd-bnc:xrp/btc',
    direct_count: 2, direct_profit: 0.47,
    direct_days: 1,
    direct_history: '2020-08-18:2',
    reverse_count: 151,
    reverse_profit: 41.93,
    reverse_days: 9,
    reverse_history:
    '2020-08-25:1,2020-09-03:11,2020-09-04:25,2020-09-05:19,2020-09-13:16,2020-09-
    14:1,2020-09-16:16,2020-08-19:9,2020-08-22:53' } ]
```

Call Example: params = {"chain": "xrp", "dateFrom": "2020-08-15", "dateTo": "2020-09-24",
"reverse": true, "profit": "100.2", "lite": "0"}; api.request('report/chains', params, function (response) {})