ATLANTIS PRESS

Advances in Engineering Research, volume 150

4th International Conference on Machinery, Materials and Computer (MACMC 2017)

Design and Implementation of Simulation System for Program Trading Strategy of Stock Exchange

Ruilin Wang[1, a]

[1] School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, 100876, China

[a]email

Keywords: Program trading, VN.PY, Strategy simulation

Abstract. Strategy operation is one of the most important aspects in program trading. In this paper, the program trading strategies of the high-frequency trading, algorithmic trading, and carry trading strategies were investigated. Through investigation and analysis, the principles, advantages and disadvantages of the three strategies were analyzed. This paper introduced the features of the program trading system architecture VN.PY based on Python language and ctaalgo module, focused on learning and analyzing the strategy module od VN.Trader trading platform in VN.PY, and put forward the shortcomings of the module. In view of the insufficiency of the function of ctaalgo strategy module, this paper put forward the revision idea of the corresponding module and function according to the project requirements. This article has made the corresponding improvement to the VN.PY open source code, added and revised the strategy. The back-testing function has been improved and the functions related with strategy simulation has been improved. Finally, the completion of the system testing and description were explored in the paper.

Introduction

Over the past few decades, the models of investors' trades on the capital market have undergone a radical change. In today's capital markets, many people choose to use electronic systems for trading. This way is more efficient and simple, more in line with modern trends, but also more and more people's attention and favor. The traditional mode of asset management by manual trading and the trading model with the experience of investment managers have been challenged by some of the problems that have puzzled investors for a long time. Therefore, the introduction of program trading system is the popular sentiment, only program trading system can greatly improve the operating efficiency and reduce risk management. With the maturity of the domestic financial market and the diversification of futures varieties, the operability of programmed trading in China will be further expanded.

Strategy Research on Program Trading

Carry Trading. Carry trading refers to the simultaneous operation of two or more different markets or the same market in order to gain profits through favorable spreads. Its characteristic is the risk is small, but the income is relatively low. Taking futures and stock markets as an example, it is assumed that the market price is in a stable and balanced state, and that the prices of futures commodities will be fairer when they are reflected in the stock market. When there is a trend in the price of a commodity in the market, it will represent a positive price information. This information will be passed to the market by trading in the stock and futures markets. At this point, prices in the two markets are going up. For several reasons, futures prices tend to move faster in the stock market, so futures prices are higher than their fair value relationships with the stock index. Using arbitrage strategies, selling futures that have become relatively expensive and buying relatively cheap shares. The effect is to bring price back to fair value at a new high level. Considering arbitrage stocks, the program purchase order is triggered by price differences, and traders do not have to care about the price movements caused by the factors.

High-Frequency Trading. High-frequency trading is now widely used in financial transactions and hedge funds. High frequency trading strategies are also widely used. For high-frequency trading, the industry is always difficult to give a unified standardized definition, using the characteristics of high-frequency transactions to describe this type of programmed trading strategy. Among them, the U.S. Securities and Exchange Commission (SEC) definition of high-frequency trading is the use of professional traders, trading in Japan many times Trading strategy. High frequency traders use quantitative methods and algorithms to acquire and process trading instruction information quickly, generate and send trading orders, and buy and sell many times in a short period of time to gain profits. Extremely frequent transactions and small spreads are the source of profits.

Algorithmic Trading. The algorithmic trading is controlled by the algorithm from the subscription to the market to open positions and withdrawal. In the process of a series of transactions in all types of transactions without manual intervention. The algorithm relies on the computer and the Internet trading platform, it needs through a series of algorithms for instruction set computer trading platform, and set up trading strategies in loading operation, through the trading strategy implementation of orders and judgments, withdrawals and other decision. In this process, the timing, price, price limit, or pricing of a commodity transaction has been determined by computer programs. The core idea, or feature, of algorithmic trading is risk differentiation. The computer system by computing its ability is much higher than that of the human brain, the success of the transaction amount will be converted to the decomposition of large transactions more than a single small transaction, and cannot achieve the same kind of goods between the hedge through the reasonable algorithm, thus reducing the investment risk. The algorithm transaction is divided into passive algorithm transaction and active algorithm transaction, and the representative strategies of passive transaction are VWAP and TWAP strategy. Active algorithmic trading is represented by the Swiss credit Sniper strategy.

Design of Simulation System for Program Trading Strategy

System Overall Design. Based on the research of the system structure and function and key technology of VN.PY, this paper uses the structure design of VN.PY and the encapsulation technology and interface of the underlying API, and designs a programming oriented transaction strategy simulation system. In this paper, the main functions of strategy implementation and back test are improved on the basis of VN.PY. It does not lose its original balance and multi market features in the system characteristics, and makes it more comprehensive and more usable on key functions. This system follows the structure pattern of the upper, lower and lower three layers of VN.PY, the top layer is the policy module, the GUI interface module and the data record module. The middle layer is the system engine layer, which consists of an event engine, an order routing, and a data engine.

The upper application level includes the CTA policy module, the GUI interface module, the market data recording module and other expansion modules. Among them, the middle engine layer includes the core engine and database, and the database adopts MongoDB database. The core engine includes event engine, order routing, and data engine.

In the core engine, the Gateway component translate the parameters required for events in the event list to the required parameters and put the event into the event engine. Order routing is used to distinguish the need orders should be connected to a trading platform, such as CTP is mainly for futures, futures and options trading, flying gold exchange, precious metals trading, when the user is connected with the CTP port and flying port, if the next single noble metal orders may be connected to the CTP port, resulting in transaction information errors, not a single. In order to avoid this situation, the function of order routing is to identify which exchange the product belongs to, and avoid transaction errors.

At the bottom of the API interface C++ interface package, not directly in the Python, many C++ call parameters in the API interface function is defined and the presence of ApiStruct.h, and these structures in Python we cannot call them to achieve learning function, also cannot create a pathway in these structures, the primary CTP, Pegasus and other trading platforms of C++ API cannot be loaded

in the Python. VN.PY will interface with a python package, called active function, the Python variable into the C++ variable, in the passive function call, the API C++ variable into the Python variable, so we can achieve the interactive CTP interface and program.

System Function Design. This section mainly introduces the design idea and technology of the innovative function which is different from VN.PY in the VN.PY system.

The main working process of the back-measurement module is to import historical data, and to replay historical data on the back-testing module, and calculate the profit ratio and other indicators through a series of functions. The back-testing module, the main function for the strategy evaluation, it simulates the operation strategy in the historical data, transaction system, profitability, and generate documentation, user can analyze the strategy of excellent.

In VN.PY, the system provides the initial realization of back test function, but did not achieve the complete test indicators show that has failed to make a transaction and back testing integration, this paper improves the test module function, add back test statistics calculation function, and unified test function interface module and strategy module, realizes the BACKPROBE firm integration function, the user can also more intuitive evaluation strategies.

After the test is completed, the system can automatically generate K line diagram according to the measured data to help users observe the running effect and profitability of the game more intuitively.

Download historical quotes from the data communications function of the current VN.PY VN.PY, but the history of the market load function also has many limitations. According to the previous research and analysis of the historical data module (ctaHistoryData.py), we will make the following improvements to this module.

Due to the latest date for download from the communications data for the history of the market was in 2015, the data is outdated. We use it as a strategy and running back to the test data, there is a big error, and far from reality. Therefore, modifying and refining the historical data loading function enables the system to load and use more recent data, enabling policies to run in better environments.

VN.PY provides the function of data loading test functions: user manual from the communications data download history of the market, in MongoDB, after loading the historical data back test.

The historical data engine of this paper is designed to add a historical data engine into the main engine and provide the ability to automatically download data from the Interactive Brokers for nearly five days after the user logs in.

Implementation of Simulation System for Program Trading Strategy

Integration Implementation of Actual Operation and Back Test. In this system, the function name and parameters corresponding to the policy engine module and the underlying API interface are unified, so as to realize the integration of the function of the policy feedback and the operation of the firm. When the policy is running, the back-testing module and the policy engine module can call the same data and implement the policy back test function while the firm is running.

Perfection of Back Test Function. In this thesis, the function of this section is to set a good initial environment for the policy and to prepare the test data for the operation of the strategy so that the strategy can be loaded and run normally. Therefore, the following tasks need to be performed:

(1) Instantiate the back-up engine classes
engine = BacktestingEngine()
(2) Set the engine back test model to K line
engine.setBacktestingMode(engine.BAR_MODE)
(3) Set the data start date for the back test
engine.setStartDate('20110101')
(4) Load history data into the engine
engine.setDatabase(MINUTE_DB_NAME, 'IF0000')
(5) Create policy objects in the engine
engine.initStrategy(DoubleEmaDemo, {})

The core function of the back test is runBacktesting (). It has a startup strategy called signal, which simulates the opening transaction after the policy is running, and records the transaction record and calculates the value. The function of this function is varied, and we can divide it into two parts according to the function. The first part is the preparation of the strategy before the operation of the strategy, and the other part is the boot strategy and virtual transaction, which leads to the simulation data.

In the first part of the initialization function, the functions are as follows:

(1) (datetime,open,high,low,close,vol）

(2) Backtesting.loadHistoryData()

(3) Backtesting.strategy.onInit()

CtaTemplate.initData = loadBar(self.initDays)

(4) CtaTemplate.ctaEngine.loadBar(self.barDbName, self.vtSymbol, days)

(5) Backtesting.strategy.onStart()

The main function of the core test section is for each bar, the system first sets up the order, then calculates the signal, issues the order and waits for the next bar cycle. The core idea is:

(1) for d in self.dbCursor

(2) data = dataClass()

data.__dict__ = d

(3) If self.mode == self.BAR_MODE:

dataClass = CtaBarData

func = self.newBar

self.crossLimitOrder()

self.crossStopOrder()

self.strategy.onBar(bar)

The function of this function is to make the system call the limit order transaction function first, then call the stop call transaction function, and finally call the onBar function of the policy to send out the signal.

In the onBar function of the policy, you usually call buy, sell, cover, short, four order functions according to the signal, and call the putEvent function after the status is updated.

ctaDemo.putEvent()

ctaEngine.putStrategyEvent(strategy_name)

event = Event(EVENT_CTA_STRATEGY+name)

We put cta event in the event engine:

ctaEngine.eventEngine.put(event).

Summary and Outlook

Summary. The research of this thesis contains the research of the above two problems, and this subject is also aimed at this direction. In the thesis I completed during the first Python language and program trading strategies in the financial knowledge of learning, especially on program trading strategies on how to use historical data back to the test was studied, then the VN.PY source code of the VN.Ttrader client module for a certain learning program flow to it carefully the research and understanding of strategy simulation to program trading have a more profound understanding of. This thesis focuses on the open source software VN.PY cannot import lack of recent history of market data, added a historical data engine, through the Interactive Brokers download option futures market to local data, and according to the characteristics of Interactive Brokers using EST time to preserve the history of the market, the design time conversion function, used for time series into Interactive Brokers, the it is the history of the market can be used in this system. Through the system test and verification, we have successfully downloaded the recent market target automatically. Second, aiming at the existing system BACKPROBE insufficiency, increased MySQL database links, and increase the back-testing index, in the back test generated after BACKPROBE index report, and display the K map, reflected in the strategy operation test in the back. Through the system test and

verification, the indexes of back test and K line diagram have been successfully obtained, and the function of program transaction system strategy test has been realized.

Outlook. At present, many functions need to be improved for the function of the system, and the functions of the disconnection, reconnection, and long-term continuous operation of the system have yet to be studied. In the future, we will continue to study the procedural trading strategy, combined with more in-depth understanding of the Python language, thoroughly complete the design of full-featured programming transaction strategy simulation system. We will complete the drop function improvement, the interface testing and so on.

References

[1] Kozhan, R.,Tham, W.W..Execution risk in high-frequency arbitrage [J]. Management science: Journal of the Institute of Management Sciences, 2012,58(11): 2131-2149.

[2] Savani, Rahul.High-Frequency Trading: The Faster, the Better? [J]. IEEE intelligent systems, 2012,27(4): 70-73.

[3] Fang Zhaoben, Zhen Lei. Research on algorithmic trading of Chinese stock market based on asymmetric ACD model and multi-interval VWAP algorithm [J]. Journal of University of Science and Technology of China, 2011, 41(9): 753-759.

[4] Liu Wei, Shen Chungen. Research on Program Trading Strategies Based on Combination of Technical Analysis Indicators [J]. Journal of Quantitative Economics, 2015, 32(3): 87-92.