Backtester Documentation:

The backtesting of day trading strategies is performed by backtester.py, which uses daily and intraday candlestick data that were imported using dailyprocessor.py and dataprocessor.py, respectively. The example strategies that I provide are simulated between January 2018 and December 2022. For each strategy, one trading account is created. While the program is running, a dynamic chart showing the balances of each trading account plotted against time is displayed. This plot also shows relevant performance metrics such as the number of trades executed, the percentage of winning trades, and the maximum drawdown percentage.

Diagram

Description automatically generated

Below is a flowchart of the main operations. After the user sets the parameters, daily data for the duration of the simulation is loaded. The program then begins to iterate through each month; loading that month’s intraday data before iterating through each timestamp. The entirety of this loop, which is contained in the dashed box in the flowchart, is translated to machine code using Numba for better performance. At 9:30 am, symbols with missing data are filtered out and the trading day begins. During each period, orders placed previously are executed before new orders are placed. At the end of the trading day, all open orders and positions are canceled and liquidated. Once the end of the dataset is reached (at the end of the month), the intraday candlestick charts corresponding to each trade are saved if plotting is enabled. This process is repeated until all months are completed.

A *jitclass* is a C-compatible object that is compiled to machine code:

* IntradayData: Contains intraday market and technical indicator data.
* DailyData: Contains daily market and technical indicator data.
* Broker: Executes orders when their buy/sell conditions are met.
* Trader: Represents one trading account which contains lists of:
  + Order: Contains all information related to a single open order. Deleted when the position is liquidated.
  + Trade: A record of each trade.
* Strategy: Handles the implementation of each specified strategy. This includes checking for entry signals, placing new orders, and modifying existing orders.

The backtester comprises of five Python modules:

* backtester.py: The main module as described above. Includes the Broker class.
* marketinfo.py: Used for storing market data. Includes IntradayData and DailyData.
* accountinfo.py: Contains the classes that define the characteristics of a trading account. These include Trader, Order and Trade.
* strategymanager.py: Implements each trading strategy. Includes Strategy.
* tradeplotter.py: Contains objects used to plot the dynamic chart showing account balances and the candlestick plots of trades. Since saving image in Matplotlib is very slow, multiprocessing is used to distribute the task across every core.