Objective

Demonstrate your foundational skills in high-frequency trading (HFT), quantitative analysis, and software engineering by implementing a basic market-making strategy using the HftBacktest framework.

Data Source

Use the sample tick data provided at https://reach.stratosphere.capital/data/usdm/ for your implementation.

Task Description

Implement a basic high-frequency market-making strategy for a single crypto trading pair. Your implementation should use HftBacktest (https://github.com/nkaz001/hftbacktest) and include:

1. Data Processing:

- Load and preprocess the provided tick data from the given source.
- Implement at least one relevant feature for HFT decision-making.

2. Strategy Implementation:

- Develop a simple market-making strategy that adjusts quotes based on order book imbalance.
- Implement basic risk management (e.g., position limits, order size constraints).

3. Latency Simulation:

- Utilize HftBacktest's latency simulation features to model feed and order latencies.

4. Performance Optimization:

- Implement at least one performance-critical component in Rust or C++.

5. Backtesting:

- Use HftBacktest to perform a backtest of your strategy.
- Calculate and report basic performance metrics (e.g., PnL, Sharpe ratio).

Written Component

Include a brief report (max 2 pages) covering:
1. Overview of your market-making strategy and its implementation.
2. Analysis of your strategy's performance, including the impact of latency.
3. Discussion of your optimization approach and its effects.
4. Brief description of how you'd improve the strategy given more time.

Requirements

1. Use Python for the overall structure, with Rust or C++ for at least one optimized component.
2. Provide clear documentation and instructions for running your code.
3. Use Git for version control and submit your solution as a GitHub repository.

Time Allocation

2 - 4 days.

Evaluation Criteria

1. Code quality and organization
2. Understanding of HFT and market-making principles
3. Effective use of HftBacktest's features
4. Quality of analysis in the written component

Submission

1. GitHub repository with your code

2. Written report (PDF format)
3. README file with instructions for running your code and listing any dependencies