⚙️

Hardware You Need to Run a High-Frequency Trading

📈 Algorithmic Trading Strategies: Notes

🪙 1. Market Making

> Providing liquidity and profiting from the spread.

✅ How it Works

- Place **limit orders** on **both sides** of the order book (*buy & sell*).
- Earn profit from the **bid-ask spread** when orders fill.

🌟 Best For

- Pairs with:
  - **Low volatility**
  - **High liquidity**

⚠️ Risks

- Sudden price spikes can hit your **buy side** and leave you exposed.

⚡ 2. Latency Arbitrage

> Exploiting price differences across exchanges.

✅ How it Works

- Detect price **discrepancies** between two (or more) exchanges.
- Example:

    - BTC = **$30,000** on Exchange A
    - BTC = **$30,050** on Exchange B
    - You **buy on A** and **sell on B** instantly to capture the $50 spread.

🌟 Requirements

- **Ultra-fast connection**
- Ideally use **co-location** to minimize latency.

⚠️ Risks

- Execution delay can cause the price to move before both trades complete.

📊 3. Statistical Arbitrage

> Profiting from price inefficiencies using statistical relationships.

✅ How it Works

- Use **mean reversion** or **correlation models** across **related markets/pairs**.
- Look for temporary mispricings that are likely to revert.

🧪 Example

- ETH/BTC vs ETH/USDT vs BTC/USDT
- Trade based on relative mispricing between these pairs.

🔍 Tools & Signals

- **Z-Score**
- **Cointegration tests**
- **PCA (Principal Component Analysis)**

🚀 4. Momentum Ignition

> Riding short-term micro-momentum triggered by market activity.

✅ How it Works

- Detect and ride **micro momentum** during:
    - Sudden **order book imbalance**
    - Large **whale trades** or volume spikes

🔍 Key Insight

- Enter early when momentum ignites and exit before it fades.

🧊 5. Iceberg Order Detection *

> Spotting hidden large orders and following smart money.

✅ How it Works

- Use algorithms to detect **iceberg orders** (large hidden orders broken into smaller visible parts).
- Anticipate institutional moves and position yourself early.

🔍 Why It Works

- Hidden liquidity often signals key levels and institutional interest.

🫧 6. Quote Stuffing Detection

> Exploiting fake depth & order book manipulation.

✅ How it Works

- Detect **quote stuffing**: rapid fake orders intended to confuse and manipulate the market.
- Use **signal processing** to spot and trade against manipulation.

🔍 Strategy

- Trade the **quick reversal** after the fake orders are withdrawn.

📊 Types of Algorithmic Trading Models

Model	Description	Speed Requirement	Best Market
Market Making	Profit from bid/ask spread	Medium	High liquidity pairs
Arbitrage	Exploit price gaps across venues	Ultra-fast	BTC/ETH across exchanges
Trend-Momentum	Detect micro-trends	Fast	Mid-cap pairs
Mean Reversion	Bet on price returning to average	Fast	Range-bound assets
Order Book Imbalance	Detect heavy buying/selling pressure	Real-time	Futures / Perpetual swaps
Event-Driven HFT	React to news, tweets, Fed rates	Fast	Large-cap coins

---

⚙️ HFT Infrastructure & Optimization

🚀 1. Ultra-Low Latency Infrastructure

> Build the fastest possible pipeline to stay ahead of the competition.

✅ Best Practices

- 🖥️ **Co-location**
  - Deploy your servers **physically near exchange matching engines** (in the same data center) for minimal latency.
- 🧑‍💻 **Low-Level Languages**
  - Use **C++** or **Rust** for microsecond-level execution speed.
- 🖤 **Bare Metal / Dedicated VPS**

    - Avoid shared cloud resources.
    - Deploy on **bare-metal servers** or **dedicated virtual private servers (VPS)** for maximum performance.
- ⏱️ **Every Millisecond Counts**
    - Faster bots consistently beat slower ones to the fill — even a **1ms** improvement can make a huge difference.

📡 2. Smarter Signal Models

> Combine advanced order book & statistical techniques to improve trading signals.

✅ How to Build Smarter Signals

- Combine:
    - **Order book imbalance**
    - **Time & sales (tape)**
    - **Volume delta**
- Use **statistical models** for:
    - Predicting **short-term direction**
    - Detecting **spoofing** or **iceberg orders**
- Apply **statistical arbitrage** with:
    - **Cointegration**
    - **PCA (Principal Component Analysis)**

to improve your edge.

💰 3. Fee Optimization

> Reduce or eliminate trading fees to maximize profits.

✅ How to Optimize Fees

- Use exchanges with **rebates** (e.g., **maker rebates**)

- Trade with **0% or negative maker fees**:
  - Some venues pay you to add liquidity
- Eliminate fees using **native token discounts**:
  - Examples: **BNB**, **KCS**

💧 4. Deep Liquidity Pools

> Trade in high-volume markets to minimize slippage and ensure execution.

✅ Best Practices

- Trade in **high-volume markets** to avoid slippage.
- 📈 **Examples**:
  - BTC/USDT
  - ETH/USDT
  - Top perpetual contracts
- ⚠️ Avoid **altcoins with thin order books** unless your model handles slippage effectively.

🎯 5. Dynamic Position Management

> Actively adjust positions to market conditions and risk.

✅ Best Practices

- Use **real-time trailing stop-losses**:
  - e.g., 0.05%–0.2% trailing SL
- Dynamically adjust:
  - Take Profit (TP) and Stop Loss (SL) based on **volatility spikes**
- Auto-scale down your position size during:
  - High spreads
  - Major news events

🐺 6. Avoid Getting Hunted

> Make your trading behavior less predictable to avoid being targeted.

✅ How to Protect Your Positions

- Don't leave **obvious stop-losses** or **predictable patterns**.
- Randomize:
  - Order sizes
  - Order timing (slightly)
- Use:
  - **Iceberg orders**
  - **Limit laddering** to hide your intent from other traders.

🧪 7. Backtest at Tick-Level

> Perform detailed and realistic backtests to validate your strategies.

✅ Best Practices

- Use **tick-by-tick** or **Level-2 (L2)** data to avoid false signals.
- Run millions of trade simulations, incorporating:
  - Slippage
  - Latency
  - Fees
- Target performance:
  - **Sharpe ratio > 2**
  - **Win rate > 50%**
  - Low standard deviation of returns

🤖 8. Portfolio of Micro-Edge Bots

> Diversify your risk & capture multiple uncorrelated edges.

✅ Why Use Micro-Edge Bots?

- Instead of relying on **one big strategy**, deploy **10+ smaller bots**:
    - Each bot runs a **small, specific edge**.
    - Bots are **uncorrelated**, reducing overall risk.
    - Smoothens your equity curve across regimes.

🔄 Rotate Bots Based On:

- 📈 **Volatility Regime**
    - Activate bots tuned for high/low volatility.
- 🕒 **Market Session**
    - Run region-specific bots for:
        - Asia
        - Europe
        - US
- 📰 **News Sensitivity**
    - Enable/disable bots based on event risks and scheduled announcements.

🖥️ HFT Hardware Specification & Setup

🧪 1. Development + Simulation Beast

> Purpose: Ultra-smooth multi-strategy development & fast backtesting.

🔷 Specs

- **CPU:** AMD Threadripper PRO 7975WX (32 cores / 64 threads)

- **RAM:** 256 GB DDR5 ECC
- **Storage:**
  - 4TB NVMe Gen4 (datasets)
  - 2TB SSD (OS + backups)
- **GPU:** NVIDIA RTX 4090 (visual backtesting & research)
- **Networking:** 1Gbps minimum leased line, dual NIC setup
- **Cooling:** Custom liquid cooling (silent & efficient)

💸 Estimated Cost: ₹8–12 Lakh (~$10,000–$15,000)

⚡ 2. Real-Time Execution Server

> Purpose: Live strategy deployment, with microsecond-level response time.

🔷 Specs

- **CPU:** Intel Xeon Gold 6426Y (24 cores @ 3.5 GHz base)
- **RAM:** 512 GB DDR5 ECC
- **Storage:** 4TB NVMe RAID 0 (ultra-fast access)
- **Network:** Dual 10Gbps Mellanox NICs + FPGA Support (Solarflare or Napatech)
- **Extras:**
  - Hardware timestamping
  - Realtime Linux kernel
  - BIOS-tuned latency settings
- **OS:** Ubuntu Server tuned for low-latency

💸 Estimated Cost: ₹25–40 Lakh (~$30,000–$50,000+)

⚡ 3. Scalping / Execution Server

> Combines advanced models with ultra-fast execution for scalping strategies.

🔷 Purpose

- Run latency-sensitive strategies and complex models simultaneously.
- Designed for **scalping + real-time model inference** at microsecond-level latency.

🔷 Specs

- **CPU:** Intel Xeon Platinum 8460Y (48 cores)
- **RAM:** 1TB DDR5 ECC
- **GPU:** Dual NVIDIA H100 (for real-time model computations)
- **Storage:**
  - 8TB NVMe RAID (ultra-fast active storage)
  - 10TB SAS (archival & model checkpoint storage)
- **Network:** 10Gbps fiber + FPGA-integrated NIC
- **Cooling:** Fully liquid-cooled rack-mounted unit

🔷 Estimated Cost

- ₹70 Lakh – ₹1.2 Crore (~$85,000 – $145,000)

🖥️ Programming Languages for HFT

🚀 1. C++

- **Use:**
  - Core HFT engine
  - Order execution
  - Latency-sensitive code

    - DMA, FIX protocols
    - FPGA interfacing
- **Why:**
    - Fastest execution
    - Handles microsecond/nanosecond-level trades
    - Preferred by institutional HFT firms
- **Example:**
    - Writing custom order book logic
    - Latency optimization at nanosecond level

🐍 2. Python

- **Use:**
    - Backtesting
    - Strategy building
    - Data processing
    - Web dashboards
- **Why:**
    - Fast to write & iterate
    - Excellent libraries support (e.g., `pandas`, `numpy`)
- **Example:**
    - Testing moving average crossovers
    - Crypto backtesting
