In-Memory Computing SUMMIT NORTH AMERICA 2018

In-Memory Techniques Low-Latency Trading

Kevin A. Goldstein R.

Neeve Research

Kevin Goldstein

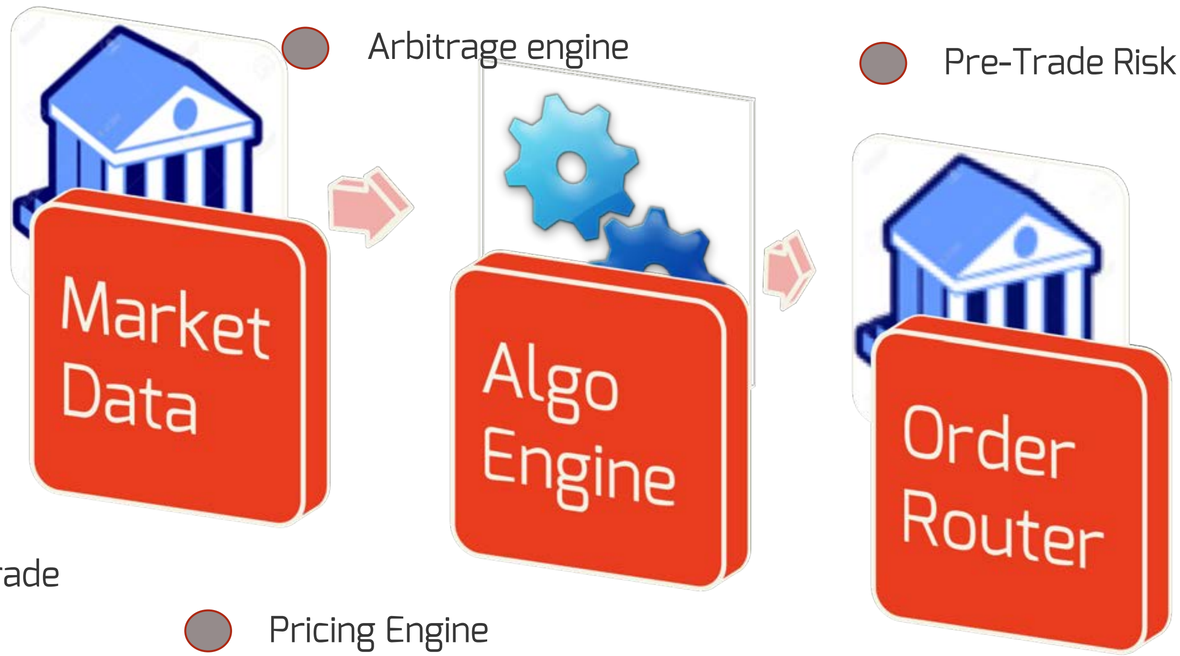
- Live in NYC
- +18 years on Wall St.
- Extensive low latency development for market makers, LL trading shops, Banks…
- Extensive performance tuning for distributed trading applications
  - *Head G2 architecture Hedgefund Management System*
  - *Head of dev USA at FlowTraders*
- Sr. Solutions Architect at Neeve Research
- Frequent speaker at Industry Events
  - *(10/25 NYC IMC Meetup)*
  - *(11/14 NYC IMC Meetup HTAP)*

Agenda

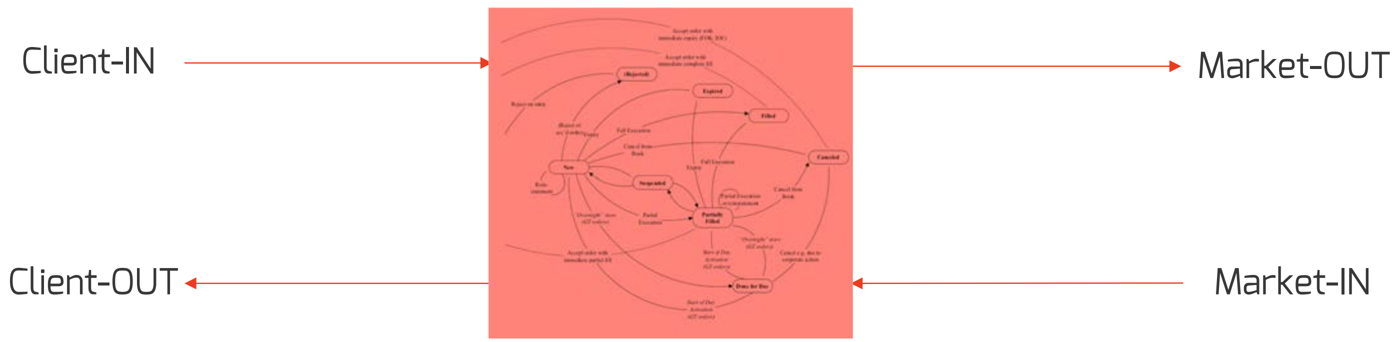
- Introduce trading systems
- Top concerns for trading systems
- IMC applied to trading systems
- Q & A

Trading System at a Glance:

- Config engine
- P&L Calc
- Arbitrage engine
- Pre-Trade Risk
- Positions
- Hedger
- Best Ex.
- Pricing engine
- Post-trade
- Pricing Engine
- Trade reconciliator
- LOPER
- OATS
- Drop copier



Basic Order Manager



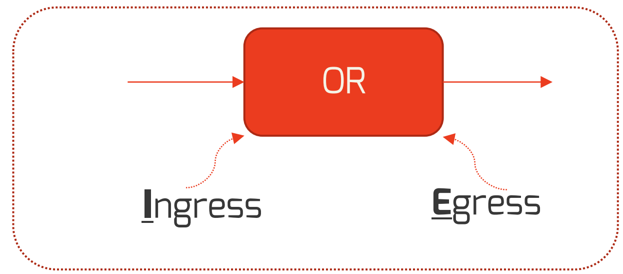
Top Three Requirements for Trading Systems

Priority

Complexity

Performance

- Low 5-20 microseconds



Consistency

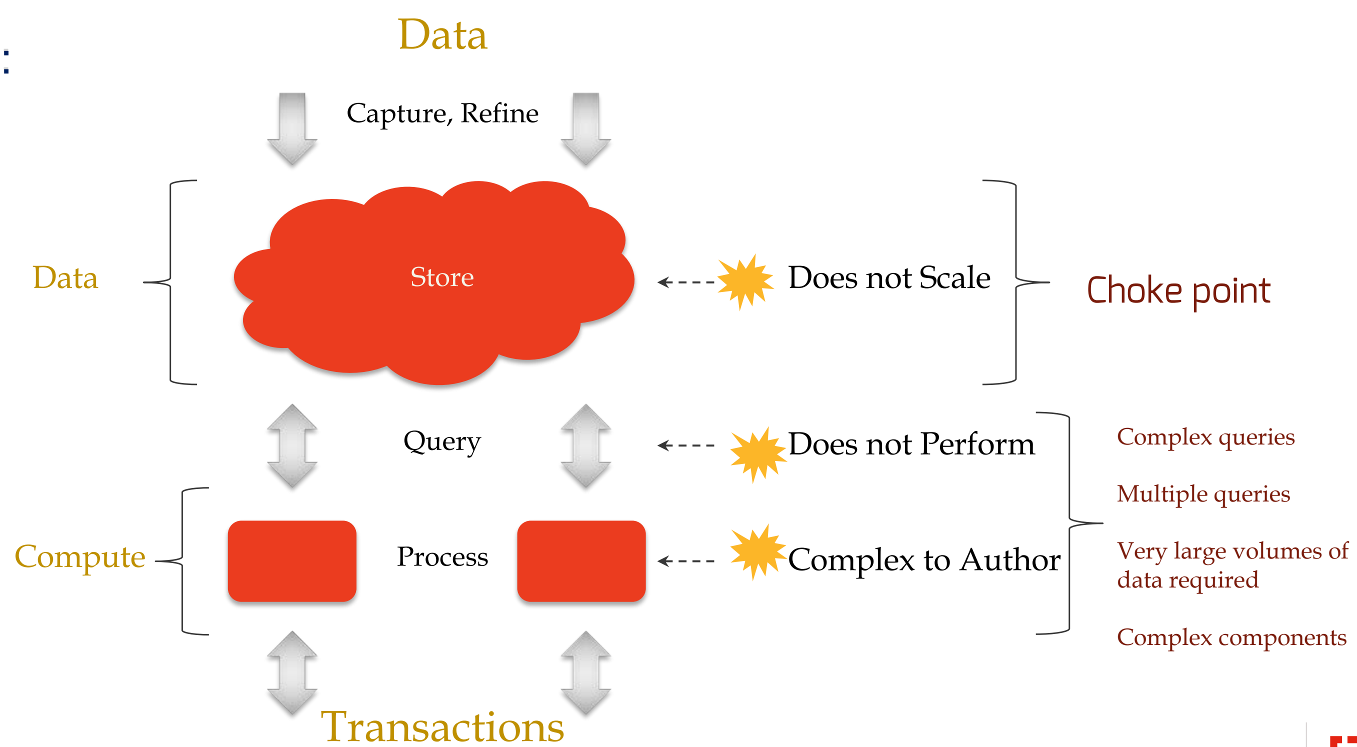
- Perform the same with 10K mps as with 100K mps
- 1mic std deviation for I2E

Reliability

- Message Reliability
- Survive process and machine failure

IMC Applied to Data Management for Performance

BEFORE:



IMC Applied to Data Management for Performance

Memory	Latency
L1 Cache	~1ns
L2 Cache	~3ns
L3 Cache	~12ns
Remote NUMA Node	~40ns
Main Memory	~100ns
Random SSD Read 4K	150µs
Data Center Read	500µs*
Mechanical Disk Seek	10ms

MEMORY ORIENTED COMPUTING!

All State in Memory All The Time!

Non Starters For Performance We're Talking About!

Sources: https://gist.github.com/jboner/2841832
http://mechanical-sympathy.blogspot.com/2013/02/cpu-cache-flushing-fallacy.html

IMC Applied to Data Management for Performance

Responsible for updating any consumers

Ownership

Publication

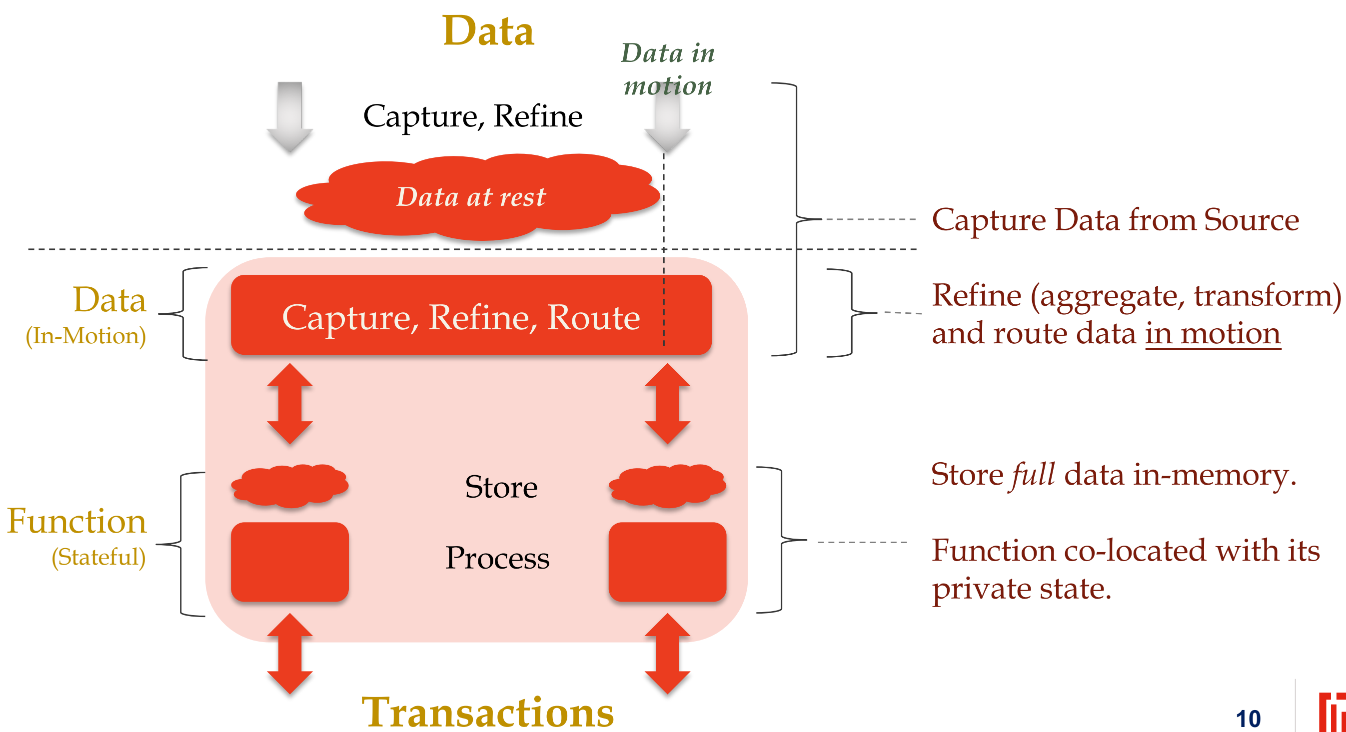
Consumption

How do you consume the data in the most efficient manner possibly

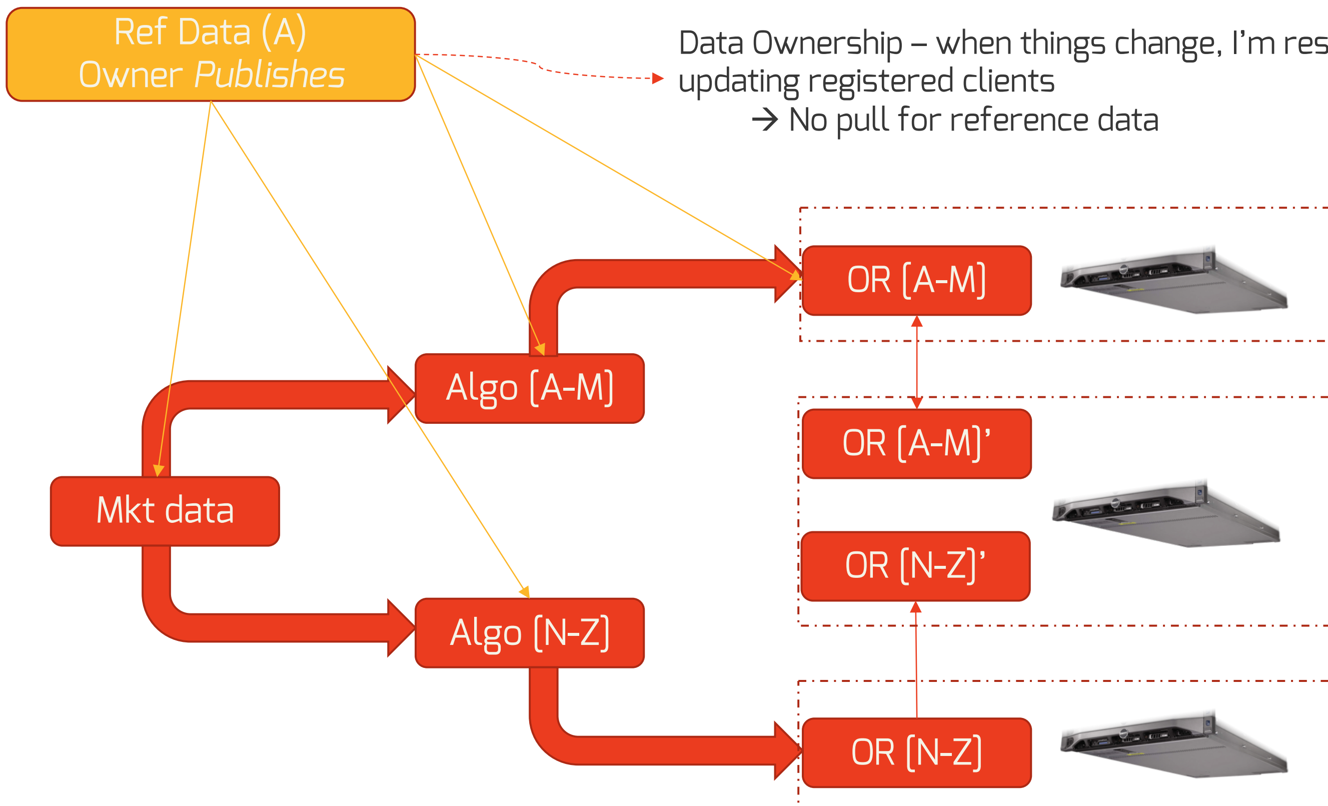
Data gravity

IMC Applied to Data Management for Performance

AFTER



IM Applied for Reliability



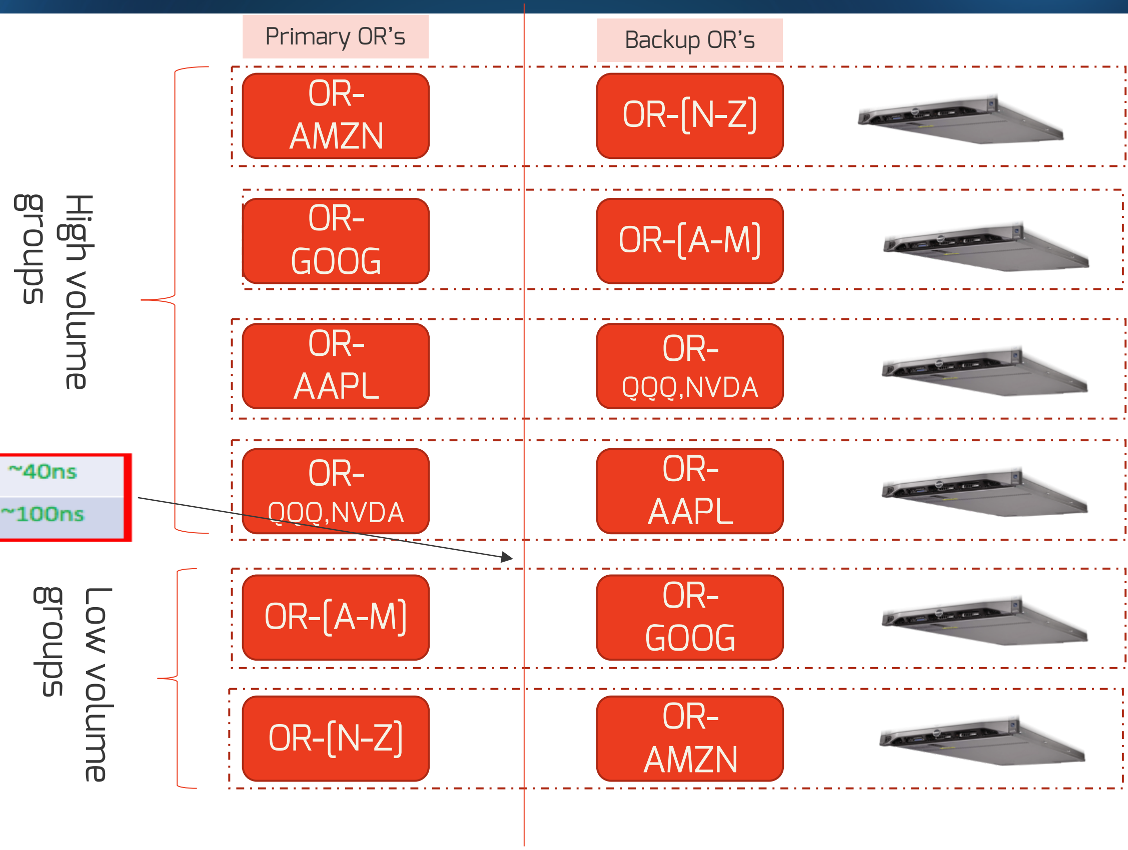
Data Ownership – when things change, I'm responsible for updating registered clients

→ No pull for reference data

- Reduce the amount of noise to deal with
- Opens the door for efficient HA
- Much smaller memory foot print
- faster access times & smaller machines

IM Applied for Reliability, Performance and Consistency

Remote NUMA Node	~40ns
Main Memory	~100ns

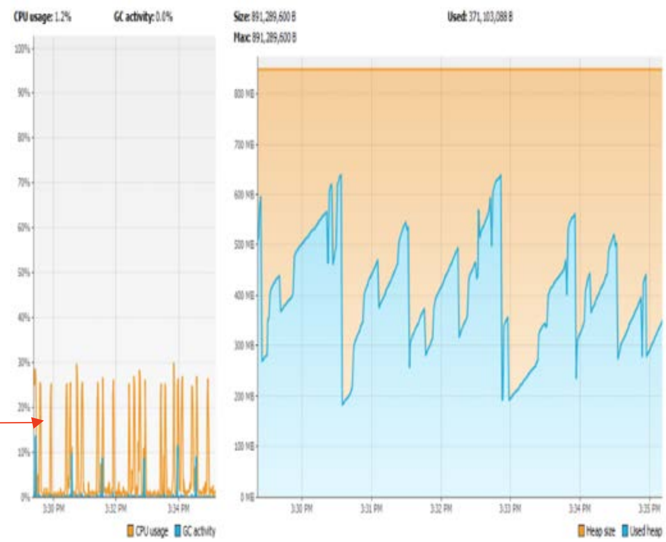


Benefits:

- Symbology flexibility
- Hardware risk
- Scaling flexibilit

IM Applied for Performance

AVOID GC

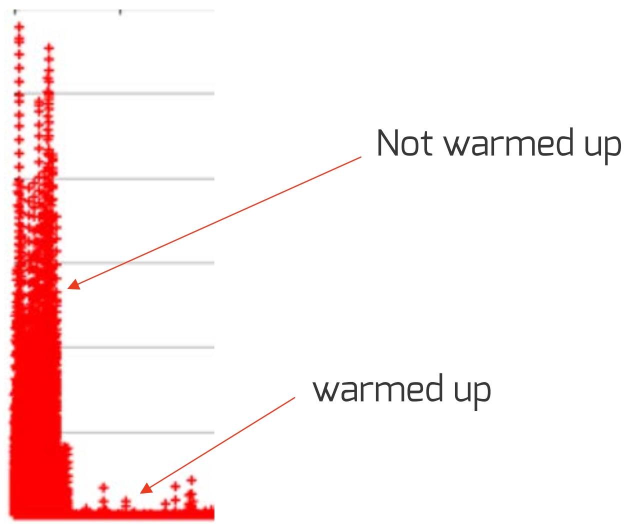


CPU is bound to memory cleanup

- Pooling is the way to go
- Leverage Off-heap memory
- Actively manage live objects
- Warmups are key

Warmups & POOLING

Warmups are a must:



IM Benefits We Capitalize On

- ✓Message Driven
- ✓Stateful
- ✓Multi-Agent
- ✓Zero Garbage
- ✓Zero Loss
- ✓Fully Fault Tolerant
- ✓Horizontally Scalable
- ✓Ultra Performance

Questions?

kevin@neeveresearch.com
@neeveresearch
@kevgol0
www.neeveresearch.com

In-Memory Computing SUMMIT NORTH AMERICA 2018