



A Better Alternative Trading System

# Matching Performance Study

October 28, 2005

## Introduction

This document describes performance testing on BATS matching infrastructure done in late October 2005. The aim of the test was to show that our software architecture realized our goal to create a scalable platform on which to run an ATS. Specifically, we wanted to show that our software could support the volume of message flow seen by the busiest ECNs while maintaining acceptable order and quote latency to our subscribers.

**Legal Disclaimer: Test results are intended for illustration purposes only. BATS Trading makes no guarantees as to the performance of the system. The simulations used for this test may or may not be indicative of system performance under real world matching conditions. Hardware and/or software changes may alter the performance of the system in the future.**

## Software Design

BATS second-generation trading platform was designed from the ground up to handle the high speed, high volume, anonymous, reliable, algorithmic trading that has become increasingly common in the equity markets today.

Subscribers connect to the system via an **Order Handler** using the industry standard FIX protocol, with BATS specific extensions. The Order Handler validates an incoming order, converts it to an internal proprietary format, and forwards it to the appropriate matching unit.

At the core of the system are several **Matching Units**. Each matching unit handles a group of symbols, which can be reallocated to balance the load across the system. The matching unit compares the limit price of an incoming order with the price of resting limit orders on the **BATS Order Book** and the price of other market's displayed quotes. If the order is immediately marketable against the BATS Order Book, and the BATS quote is equal to or better than the consolidated quote, an immediate match is made and communicated back to the subscribers. If another market is displaying a better quote, the order is forwarded to the other market, directly or indirectly, to attempt to fill the order at the best available price.

Resting orders are matched according to the following priorities:

- Price – the displayed limit price of the order
- Display – displayed orders have priority over non-displayed (reserve) orders
- Time – the time the order was received by BATS

**Full depth of book** market data is available via the BATS PITCH protocol.

## Technology

The BATS trading platform is housed in a world-class data center in Weehawken, New Jersey maintained by Savvis Communications. The same building is used to host the Archipelago Exchange.

The data center provides a high level of security, redundant power supplies including an on-site backup generator, and reliable cooling. Savvis personnel are available 24/7 to maintain and service the system.

The BATS trading platform is distributed across several blade servers to maximize throughput and minimize bottlenecks. The blade servers are basically commodity hardware running the Linux operating system. More blades can be added easily to expand system capacity.

Fast, reliable data storage is provided by an EMC Symmetrix Storage Area Network (SAN).

## Goals

For the October tests, we set out to accomplish the following goals:

- Show that a pair of matching units could sustain 10,000 messages per second over 100,000,000 messages.

## Matching Performance Study

- Show that order latency is acceptable in low, moderate and high volume situations.
- Show that market data latency is acceptable in low, moderate and high volume situations.

### Methods

For throughput tests, we started with a full day log of quote data from a leading ECN. We wrote a program to read this data and convert it to representative FIX messages. These messages were sent to our matching unit through an Order Handler, just as a client FIX message would be sent, but at a controlled data rate.

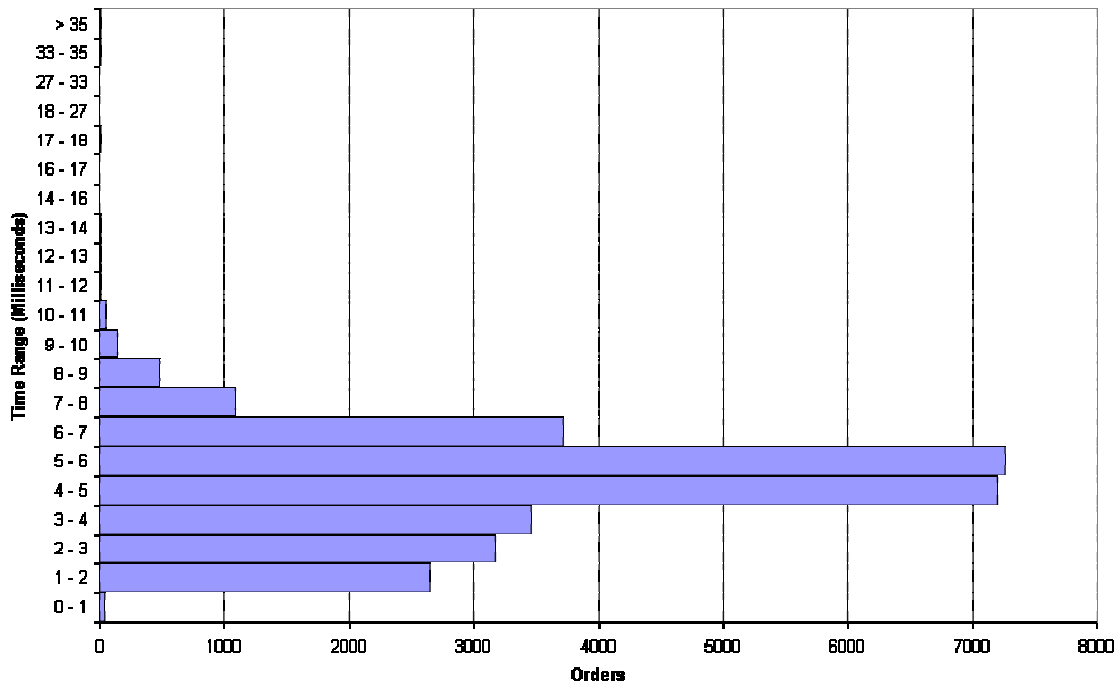
For latency tests, we created a load in the same manner as our throughput test and measured order accept and cancel times on a simulated client connection running at a rate more typical of a single subscriber. At the same time, we measured the time between sending an order or cancel, and seeing the corresponding change on our PITCH feed. Our load rates were 1400, 3000 and 4400 messages per second. Our client rates were 50, 100, 150 and 200 messages per second.

### Results

In our throughput test we were able to run a full day of activity, approximately 110,000,000 messages, through two matching engines in about 3 ½ hours with a sustained throughput of nearly 10,000 messages per second. For the purposes of this test, we considered this a success and concentrated on the latency tests.

The following histograms show the performance of our matching engine from the perspective of the client. The chart immediately below shows the distribution of response times to new orders. A large majority of order acknowledgements were received between 4 and 6 milliseconds after sending the order.

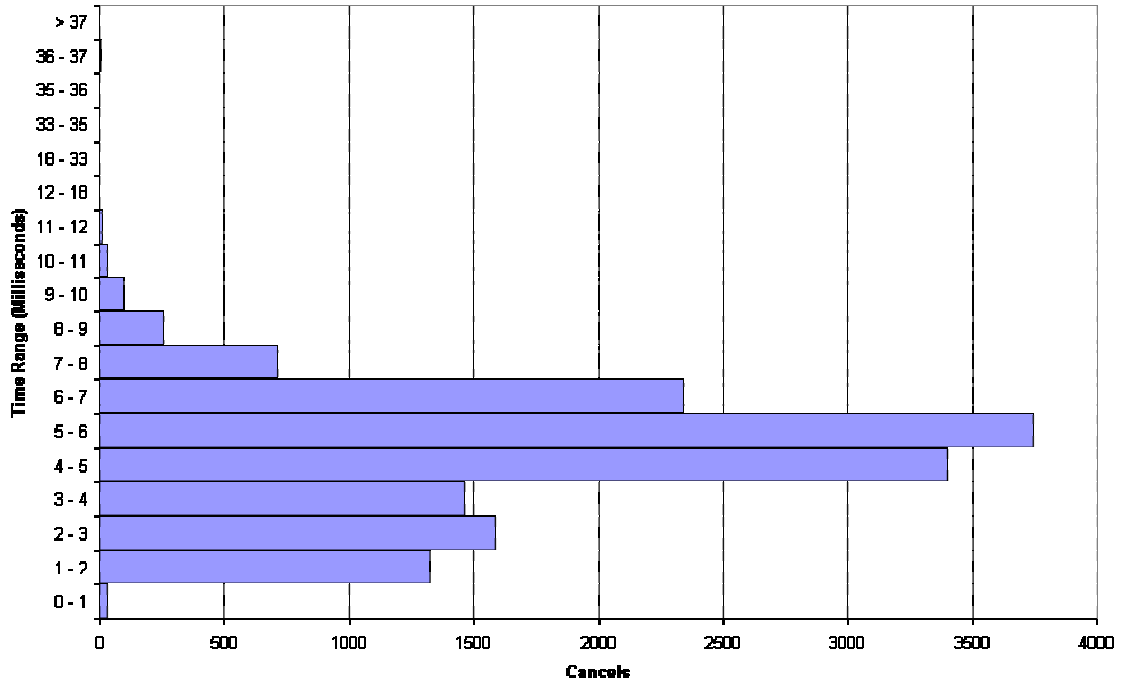
**Order Acknowledgement Latency - Low Load/Low Data Rate**



The next chart shows the distribution of response times to order cancel requests.

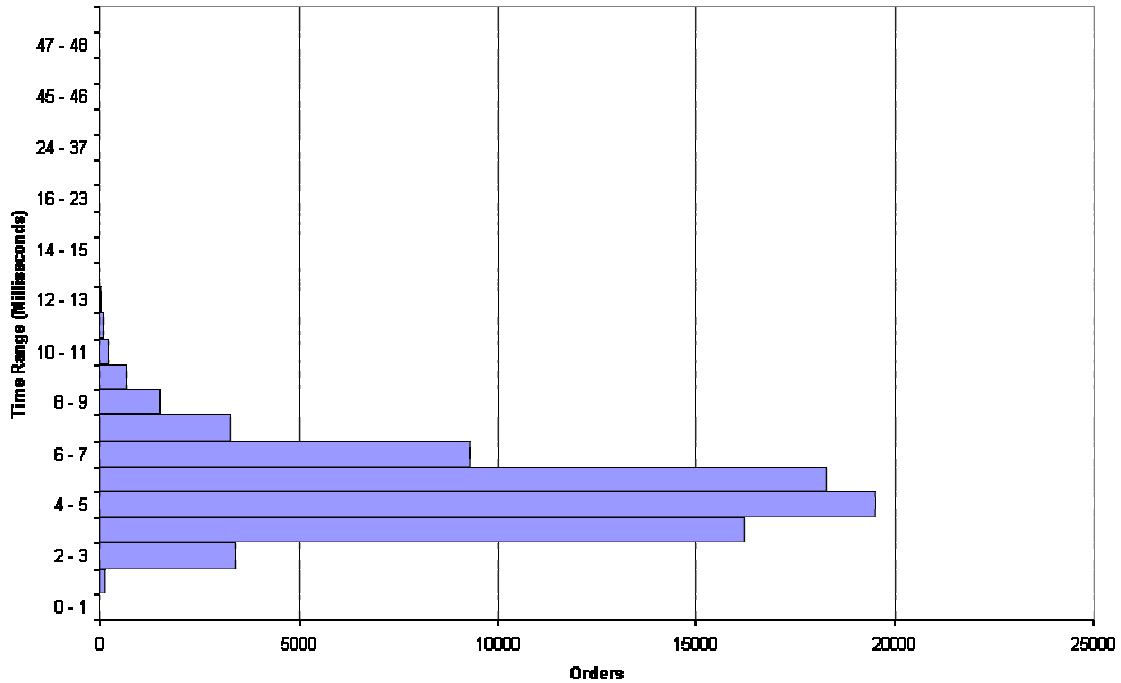
# Matching Performance Study

## Order Cancel Latency - Low Load/Low Data Rate



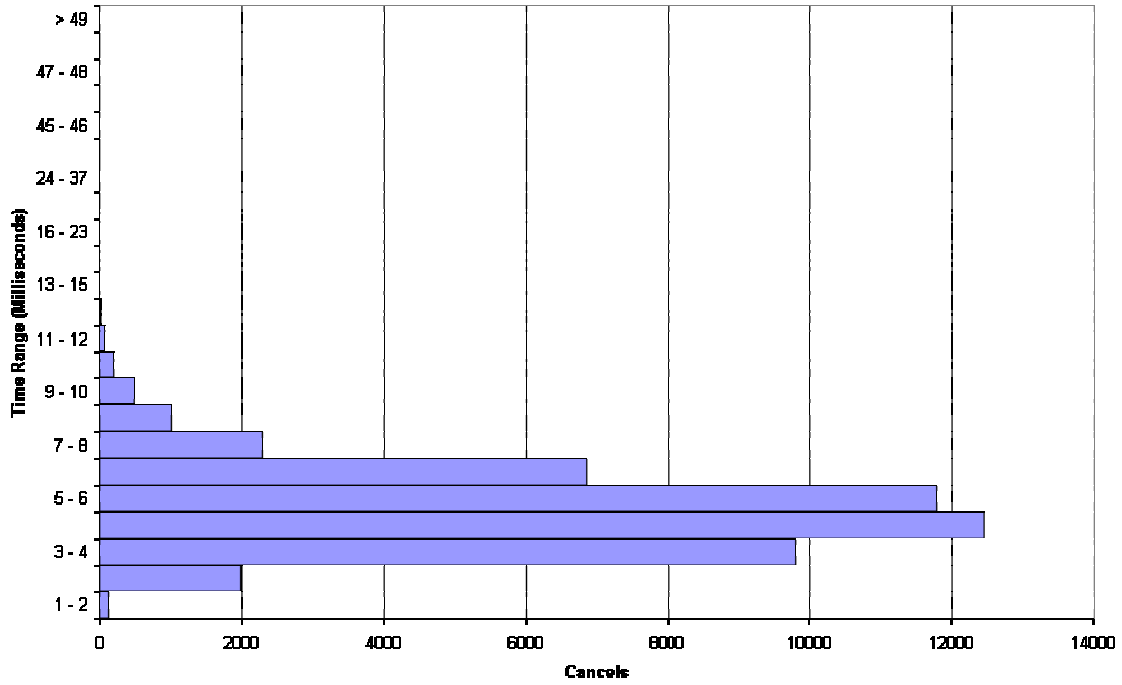
The first two histograms were at the lowest load (1400 messages/second) and the lowest client data rate (50 messages/second). The next two histograms are the results of the same test at the highest load (4400 messages/second) and the highest client data rate (200 messages/second).

## Order Acknowledgement Latency - High Load/High Data Rate



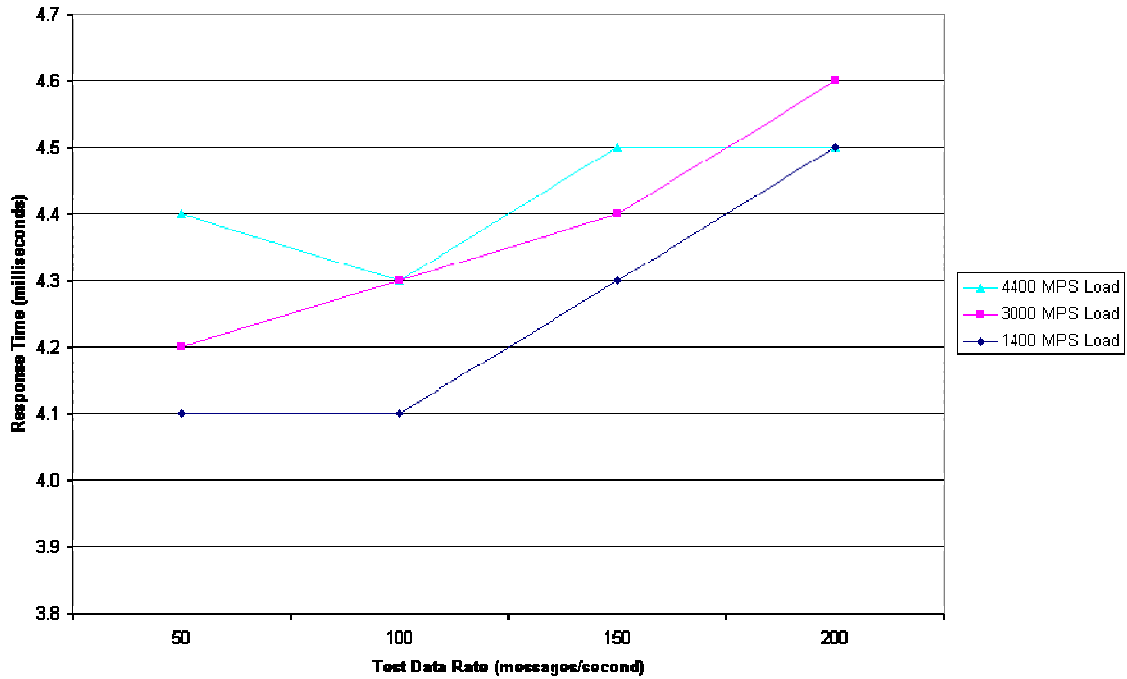
# Matching Performance Study

## Order Cancel Latency - High Load/High Data Rate

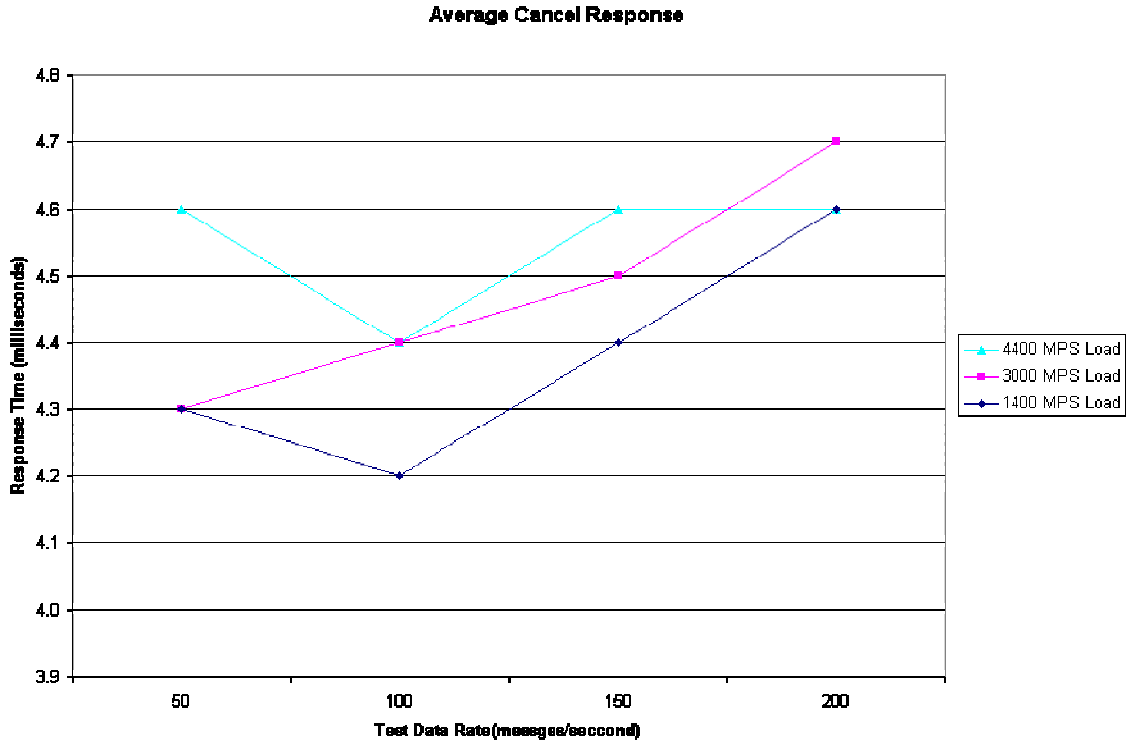


The next two graphs show the average response times of orders and cancels for each load and each client data rate.

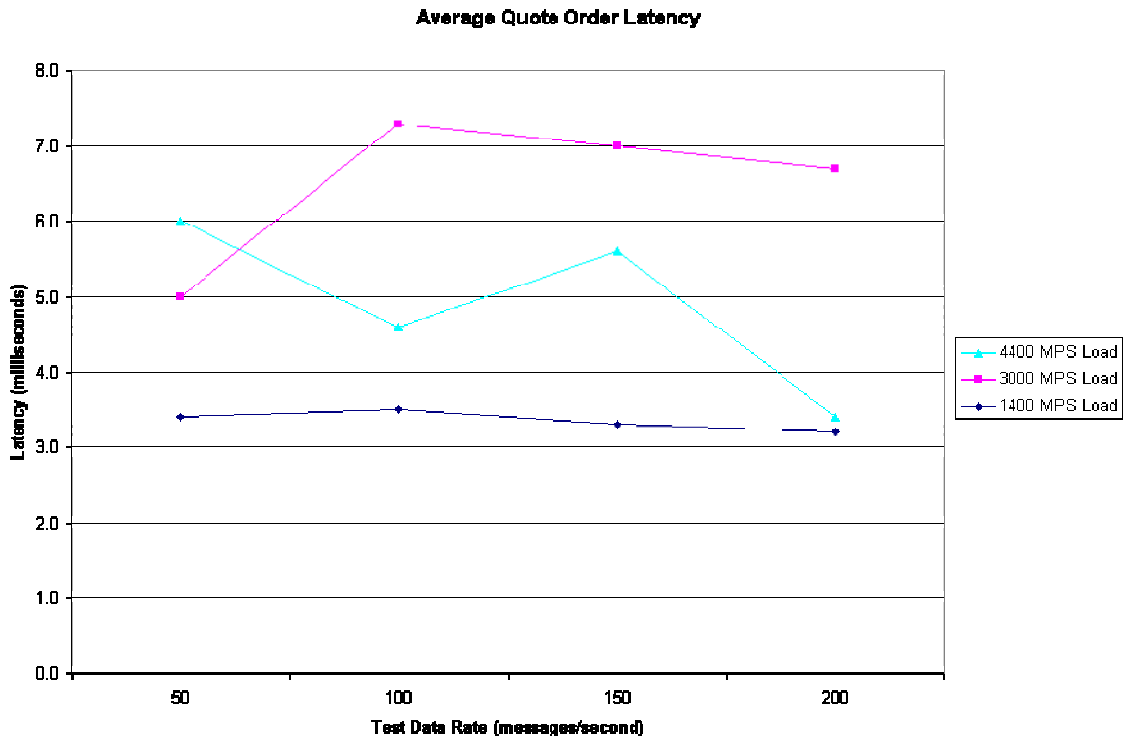
## Average Order Response



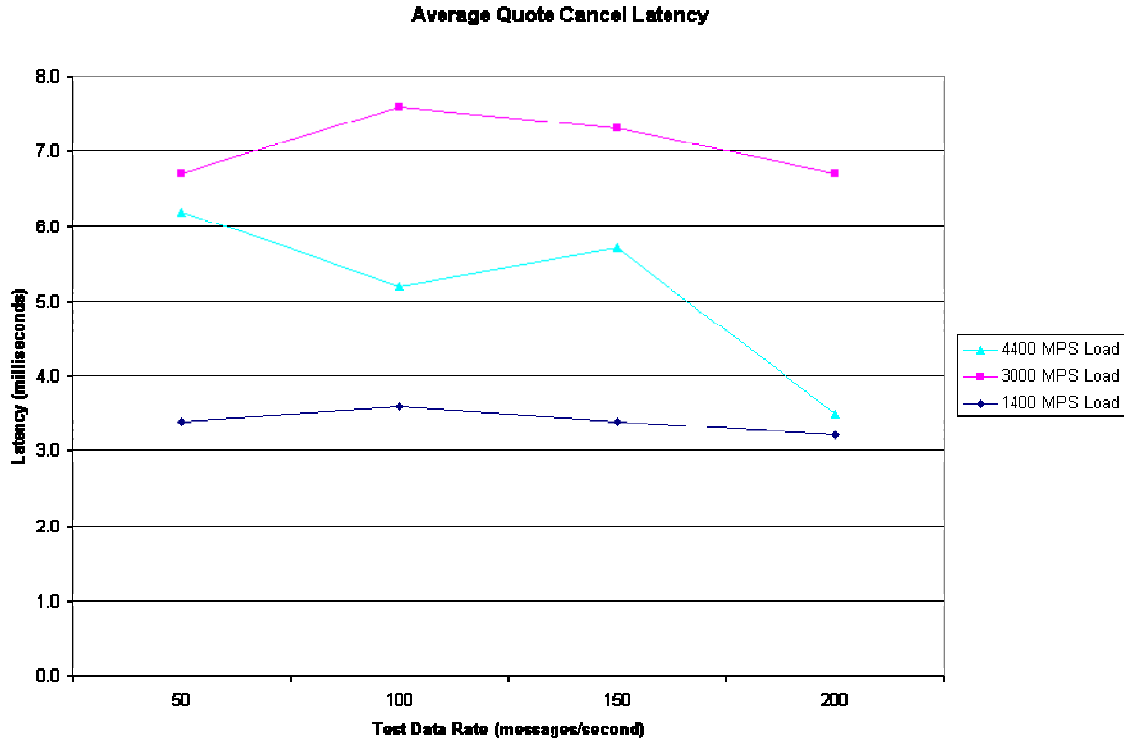
# Matching Performance Study



The next two graphs show the average time from sending a new order or cancel message to receiving the corresponding PITCH message.



## Matching Performance Study



## Conclusions

Over the course of the test, we were able to accomplish our three stated goals. Our test system was able to sustain 10,000+ messages per second while processing over 100,000,000 messages in the scope of one trading day. Our realized order latency, at real world data rates, met our response time expectations and is in line with the top ECN's operating in the market today. In addition, market data delivery latency also performed to our expectations and rivaled the top ECN's in the market. Even though our real-world tests proved that our matching engines are already on par with the best of our competitors, we believe that additional engineering efforts we have planned will allow us to improve our system's performance even more.

We were very pleased with the scalability of the matching engine. There was a very small difference between response times at the highest and lowest engine loads. The largest spread was 0.3ms. Likewise, there was a very small difference between response times at the different client data rates. The largest spread was 0.4ms. With our parallel matching architecture, we will be able to add more matching engines as necessary to support a very large data flow while providing low latency response to subscribers.