Using TPC-C to study Firebird Performance

Paul Reeves
IBPhoenix

mail: preeves at ibphoenix.com

IBPhoenix
THE POWER WITHIN

IBPhoenix
THE POWER WITHIN

About the speaker

I work for IBPhoenix providing technical support.

I maintain the windows installer for Firebird and do the Windows builds.

Introduction

The aim of this talk is to use the TPC-C benchmark to study :

- How does Firebird perform under load?
- Can we use the data collected from the tests to make evidence based decisions that will improve application performance?

IBPhoenix
THE POWER WITHIN

What is TPC-C

- Models typical OLTP application
- Old fashioned “bricks'n'mortar” business – perhaps a wholesaler providing stock to shops?
- Five randomly generated workloads
  - New Orders (45%)
  - Payments (43%)
  - Deliveries (4%)
  - Stock-level checks (r/o) (4%)
  - Order Status (r/o) (4%)
- Its main metric is the number of new orders per minute.

What's good about the benchmark ?

- Simple
- Synthetic
- (Fairly) consistent, despite a high degree of randomisation.
- Stable platform to generate hundreds of hours of test data. (500+ so far.)
- Studying real data under load is always better than guess work.

IBPhoenix
THE POWER WITHIN

What's bad about the benchmark ?

- No blobs
- No stored procedures
- Nothing special at all, really
- Very few business rules
- Very simple data model
- Very short rows
- Difficult to overload the hardware
- And, of course, it is not your data or your application.

The Test Harness

- Provides a consistent unchanging platform
- Server is 4-core x64 CPU with 8 GB RAM
- H/W Raid controller with
  - 4 * HDDs configured in RAID 10
  - 2 * SSDs configured in RAID 1
- Dual boots to
  - Windows 2012
  - openSUSE 13.1
- Firebird 2.5.3 is installed with SS,CS and SC open on different ports, using a single configuration file.

- Network connection is 1 Gbit.
- Client is another 4-core x64 CPU with 8 GB RAM
- The Benchmark app is written in Java executed from the client
- Test details and test results are stored in a separate Firebird database (on a remote server) for analysis.

IBPhoenix
THE POWER WITHIN

Outline of the tests

- **Firebird defaults except :**
  - 3000 buffers hardcoded into each DB
  - Sweep set to 0
  - SS tied to two CPU (Windows Only)
- **Each test run consists of**
  - Sweep
  - gstat full before test
  - 15 minute test
  - gstat full after test
- **No special configuration of host O/S**
  - But updates applied.
- Test Series are fully automated

Test Coverage

- Windows, Linux
- HDD (RAID 10), SSD (RAID 1)
- SuperClassic, Classic, SuperServer
- Small, Large and Very Large Databases
  - 1 GB (effectively in memory)
  - 10 GB (must use the file system cache.
  - 40 GB (too large for fs cache so lots of swapping.)
- 10..100 connections in steps of 10 connections

That is a lot of test combinations (360)

IBPhoenix THE POWER WITHIN

Caveats - I

Results are specific to :

- Firebird 2.5.3
- This test harness

The results can only be a guide, not a rule.

The main message to take away is the patterns the graphs produce, not the actual numbers.

Caveats – II

Connections are NOT users

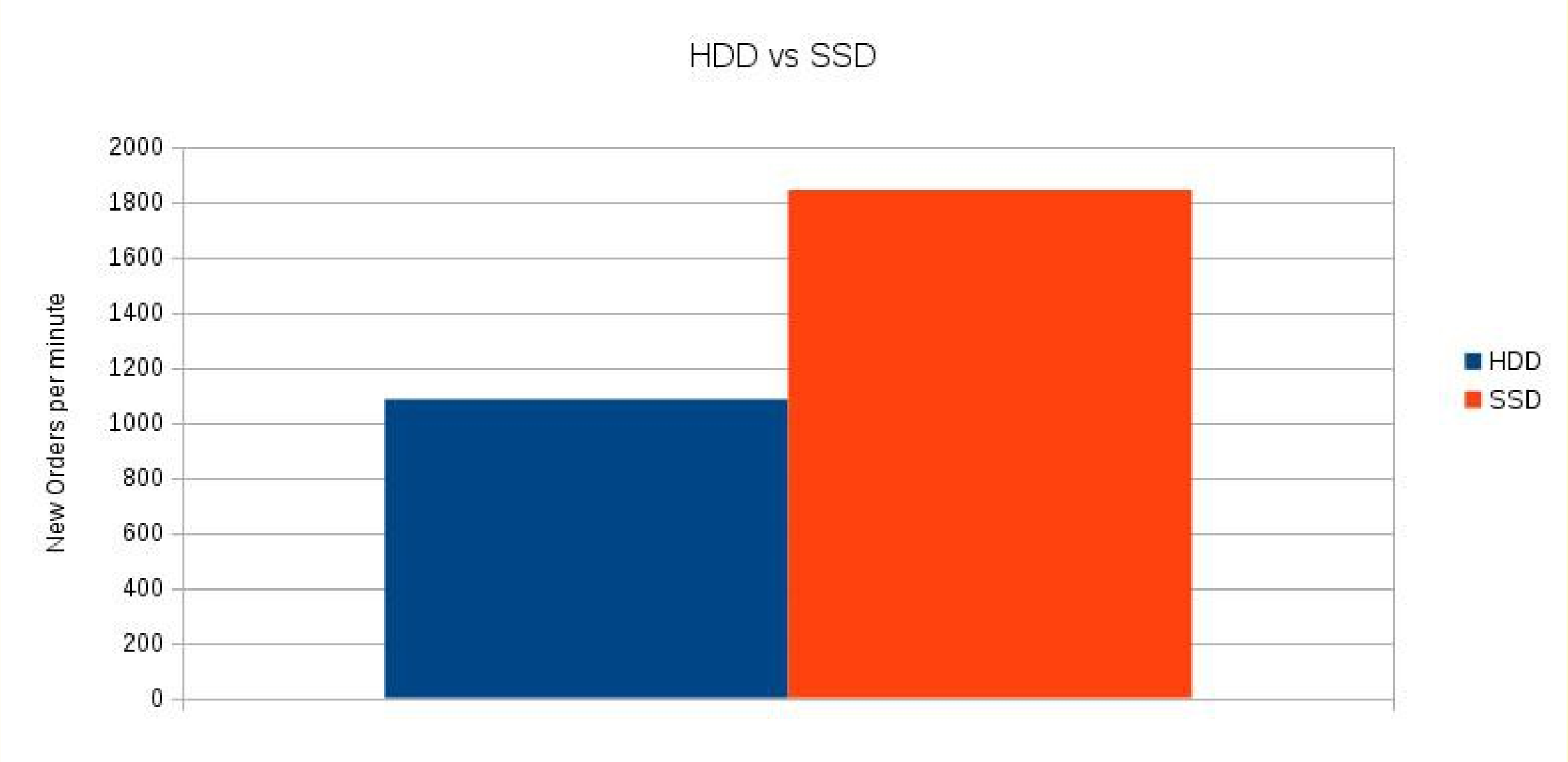
Basically the test harness is using a connection pool

IBPhoenix THE POWER WITHIN

At last, let's look at some of the results

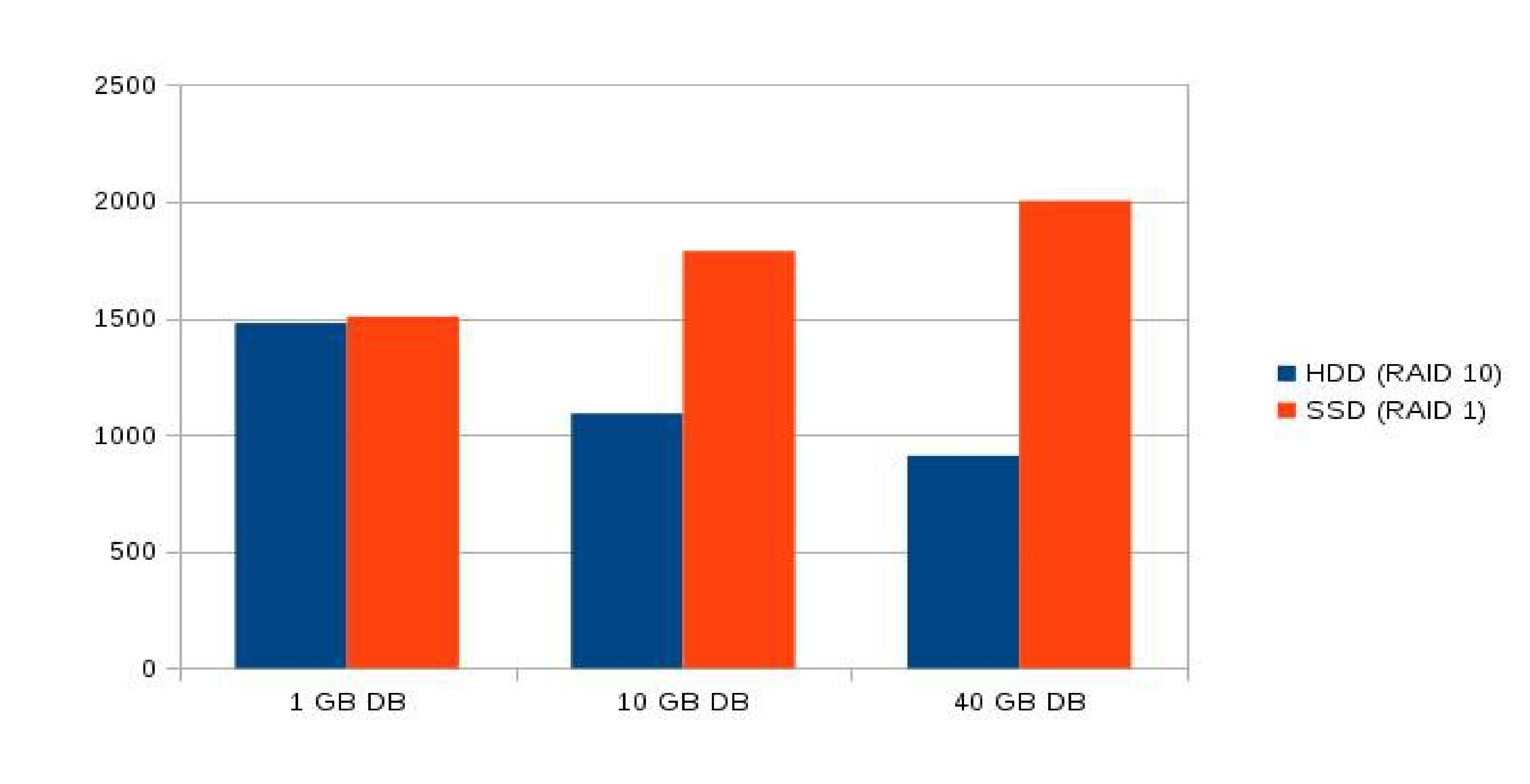
HDD vs SSD

Overall, SSD is clearly a winner

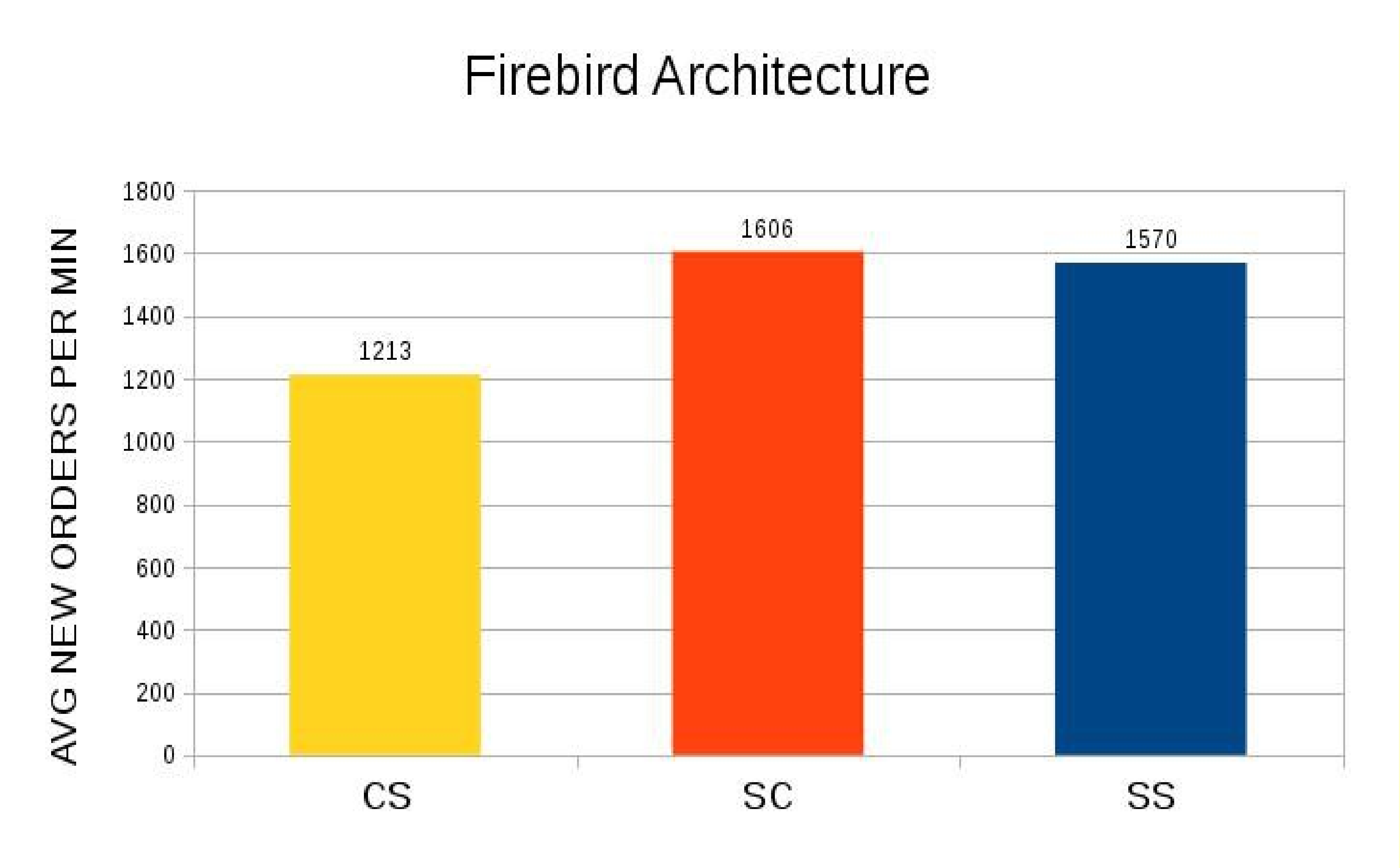


Database Size and HDD vs SSD

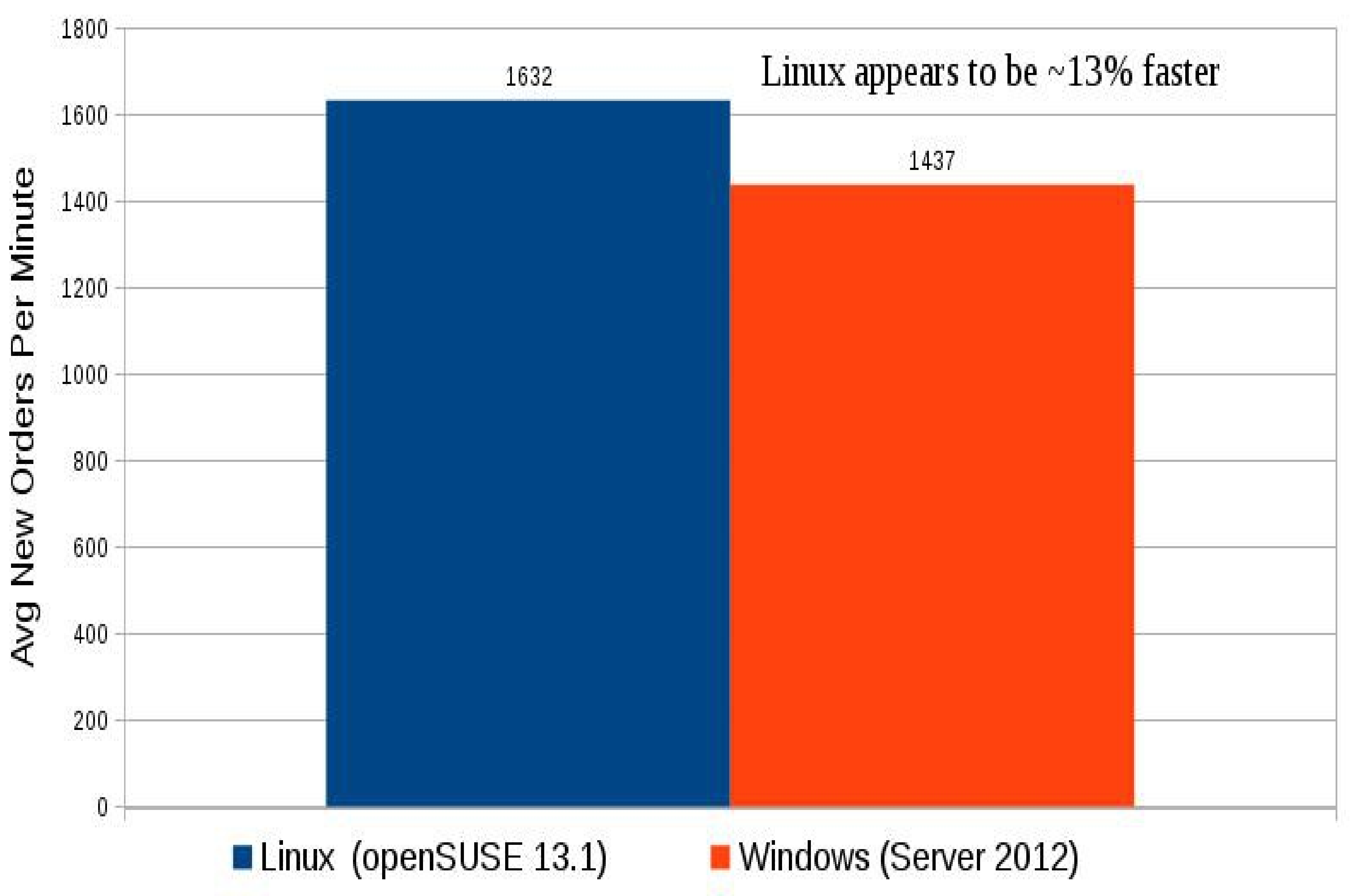
The story is not so simple...



Architecture

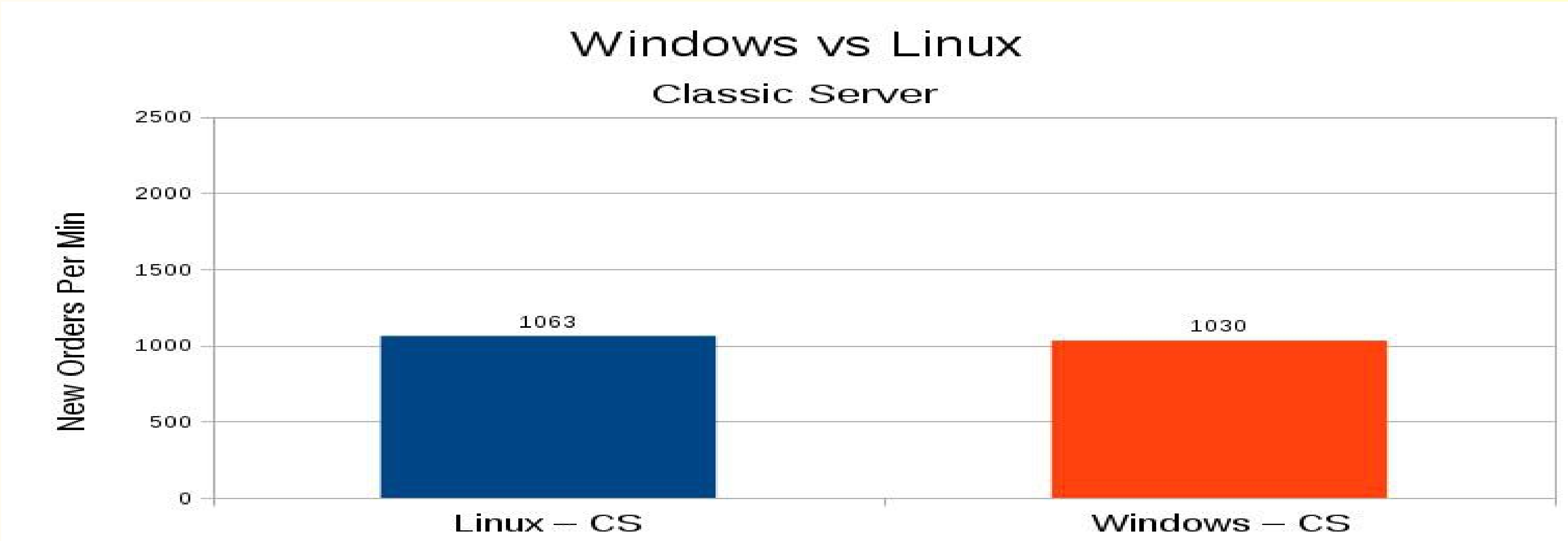


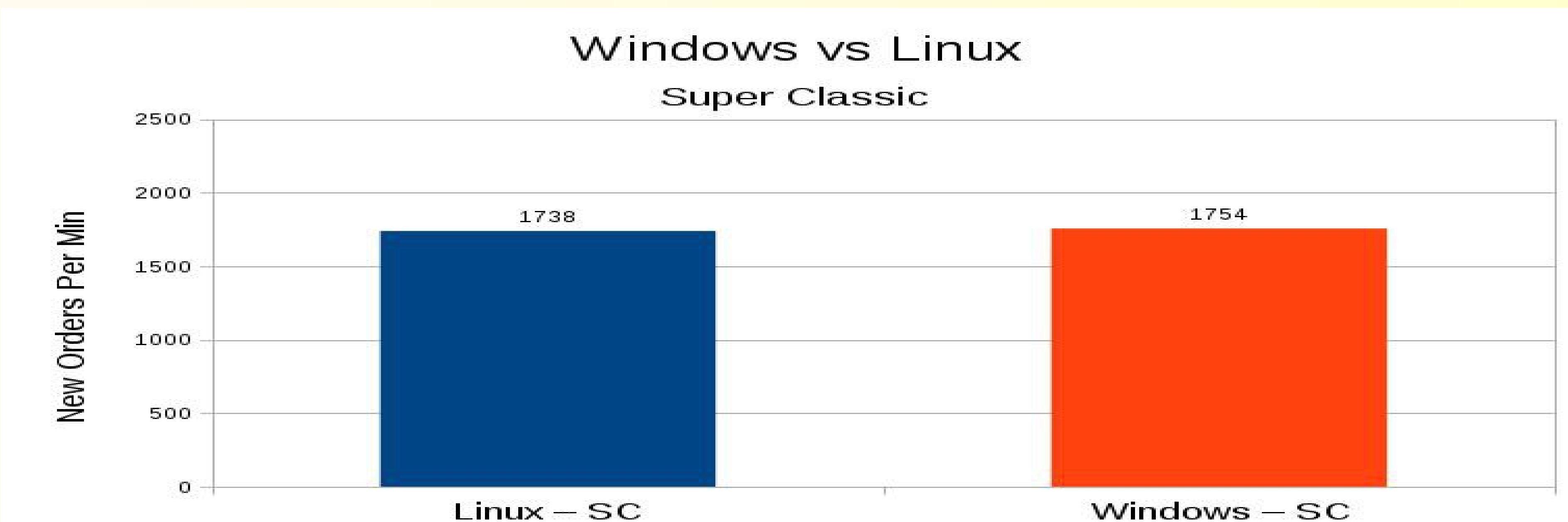
Windows vs Linux



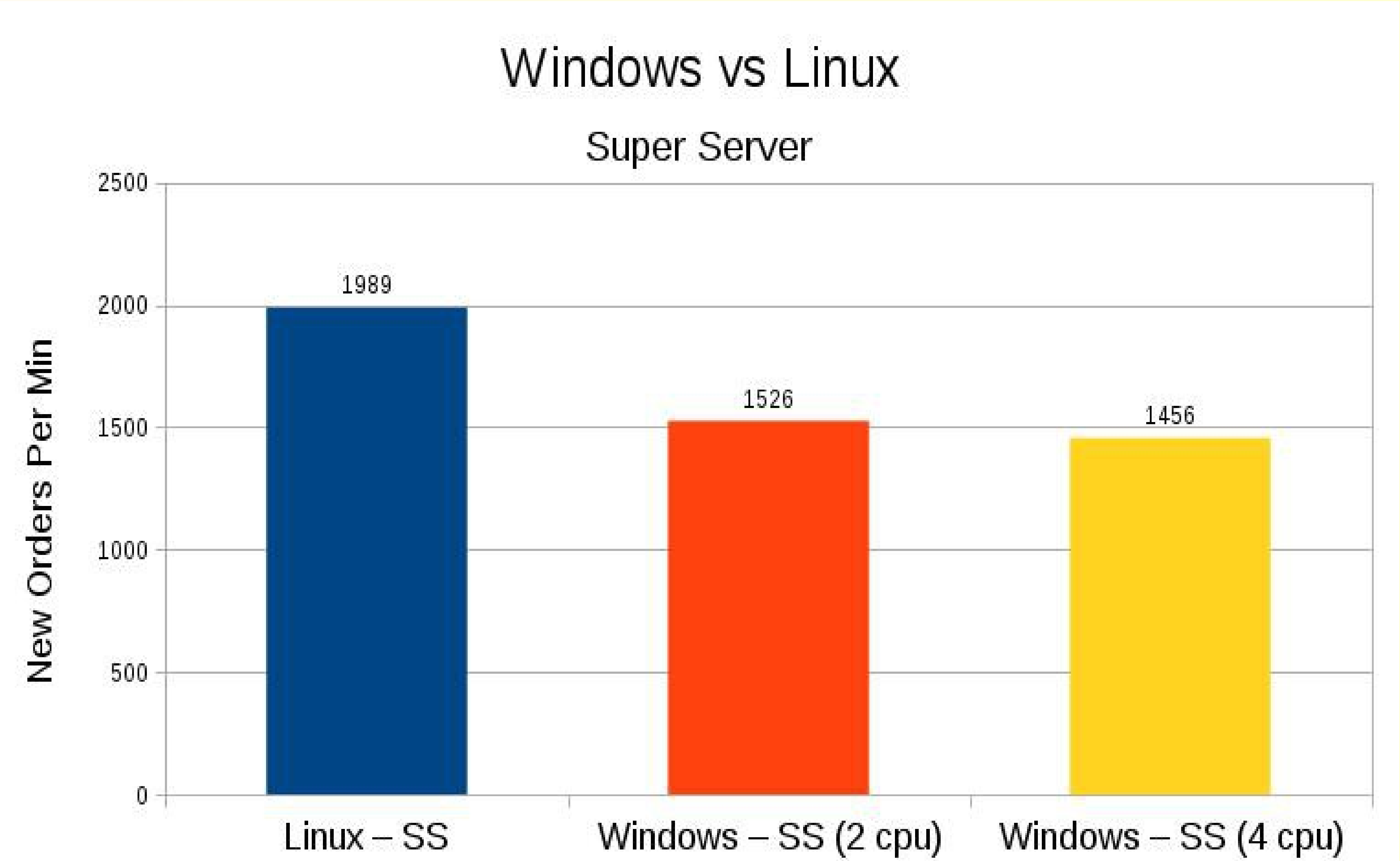
IBPhoenix
THE POWER WITHIN

Where is the problem with Windows performance?

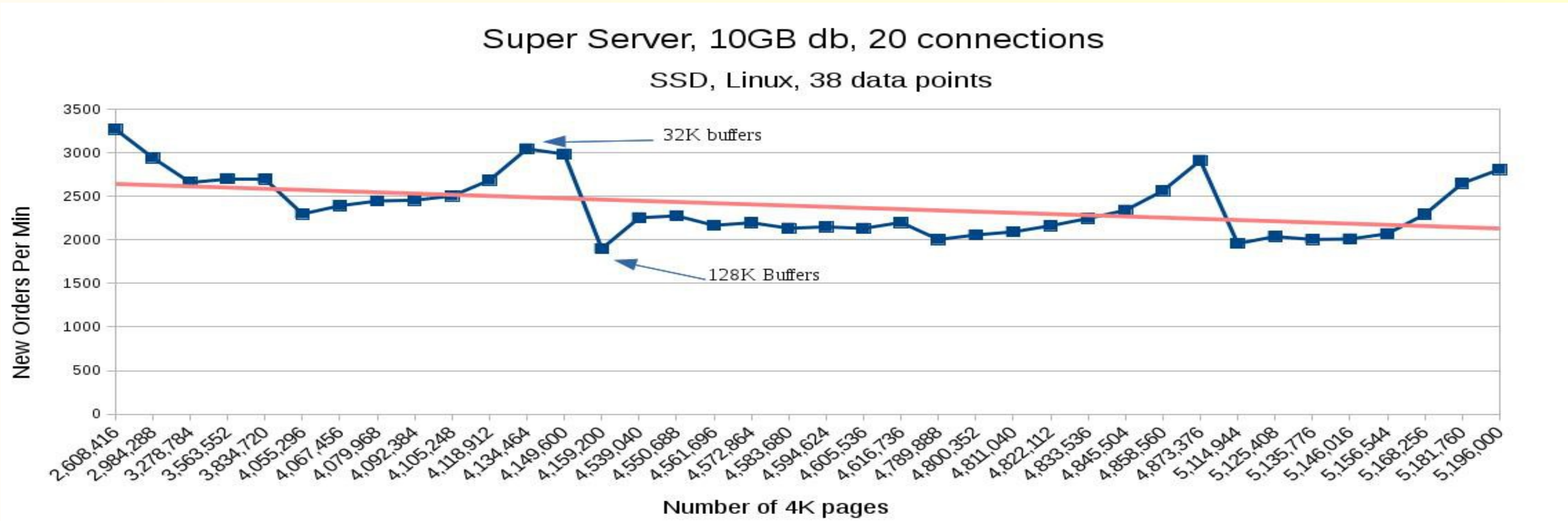


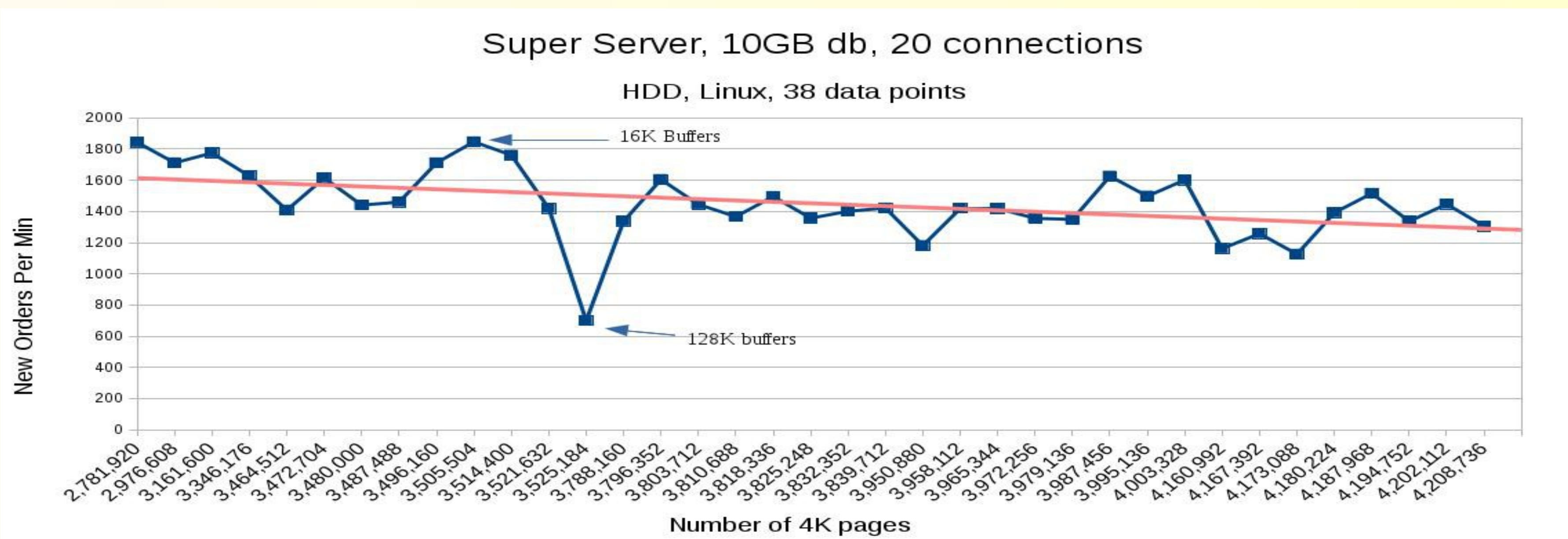


Windows and Super Server still have a problem...



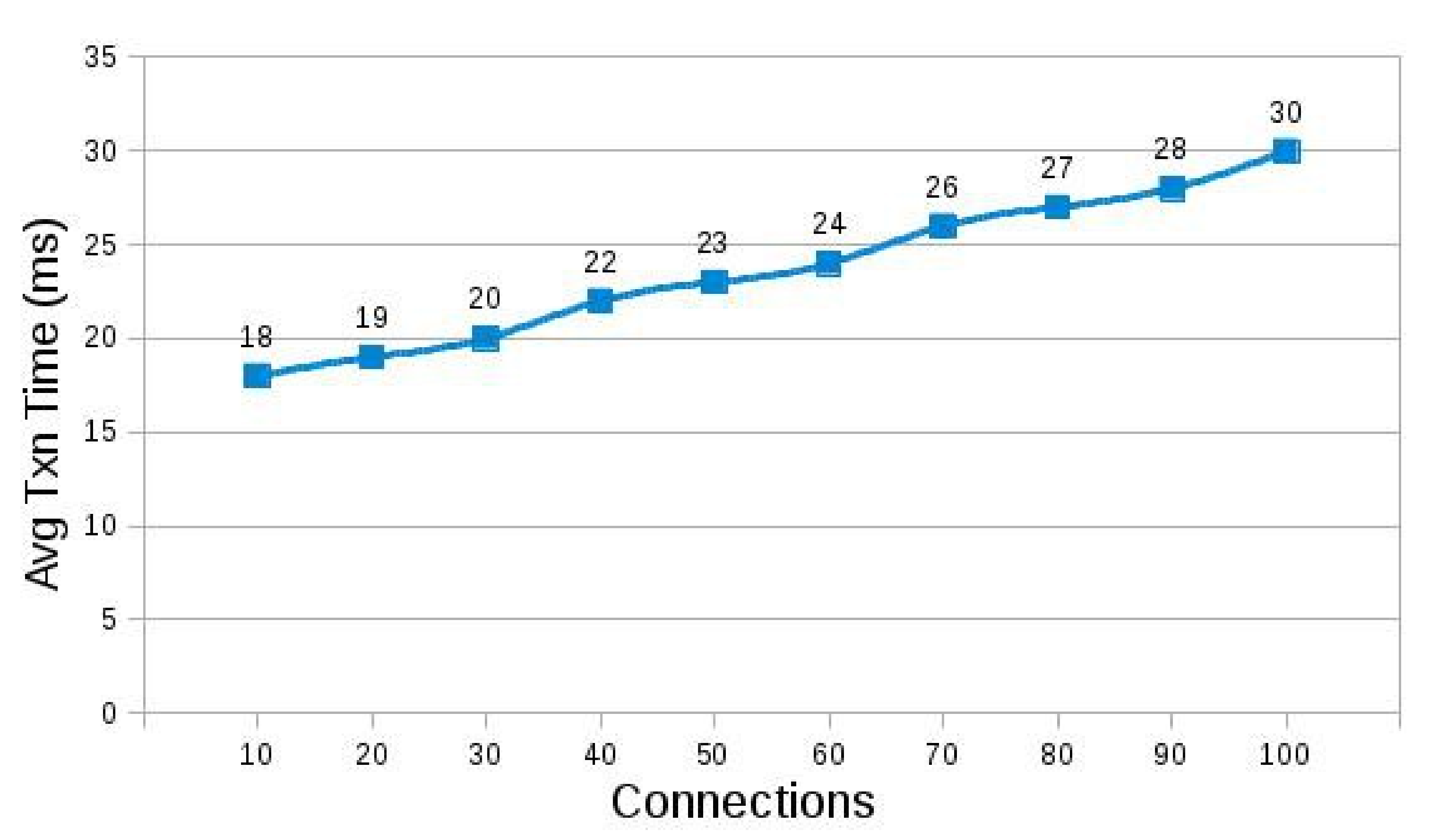
Influence of growing DB on performance



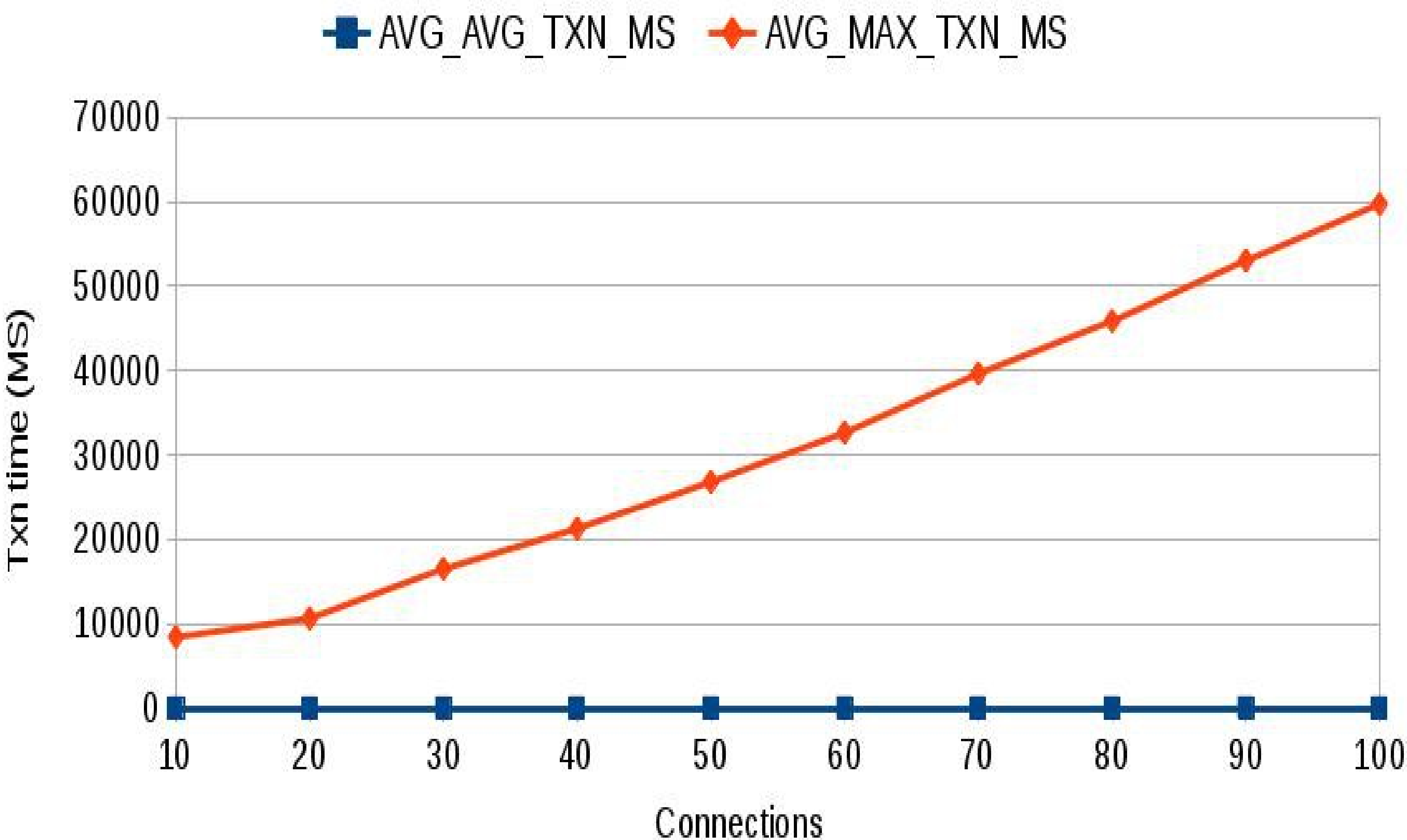


IBPhoenix
THE POWER WITHIN

Impact of increased connections on TXN time

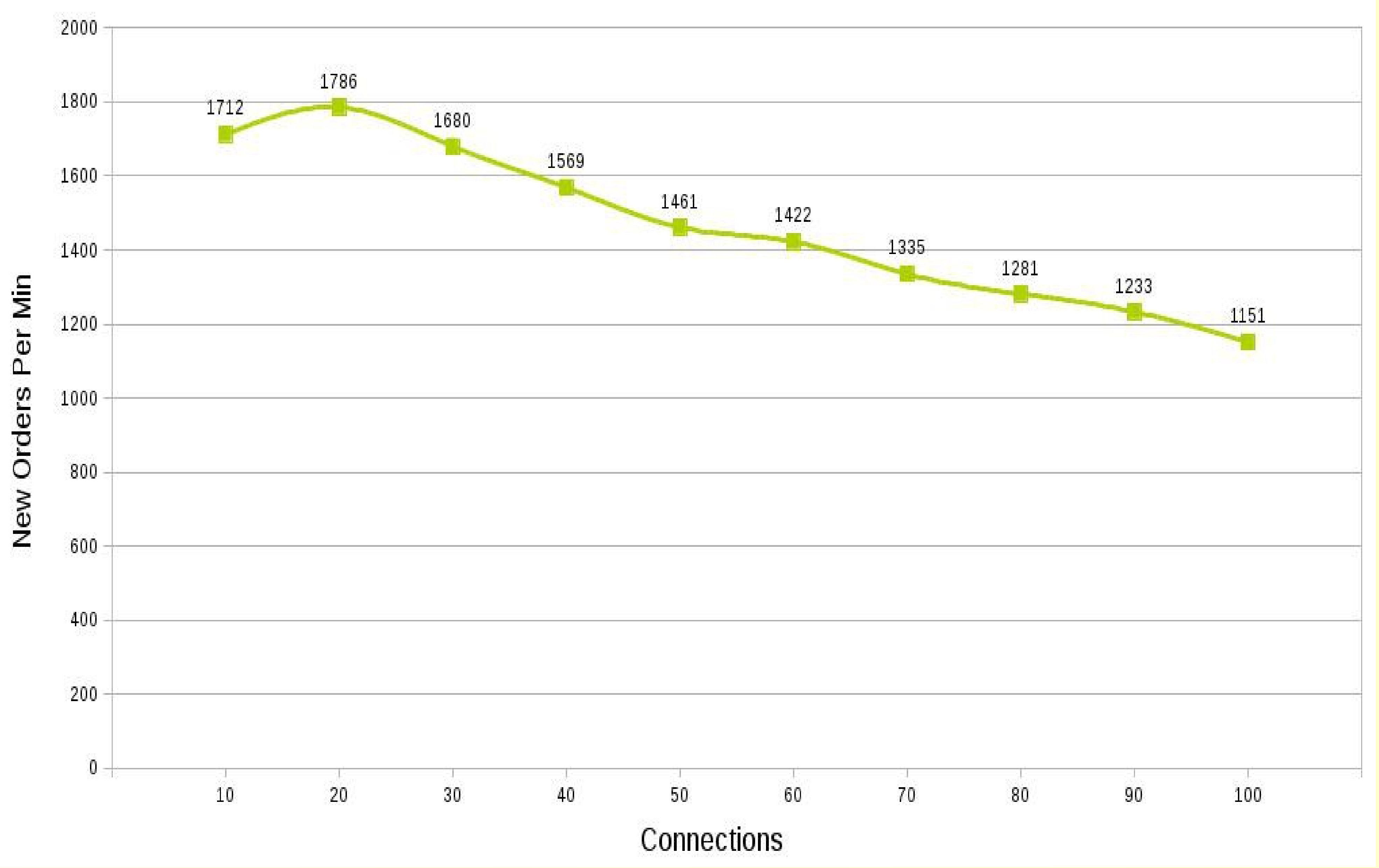


Impact on max average txn times as contention increases



IBPhoenix
THE POWER WITHIN

Impact of increased connections on NO PM



So, why the slow down as connections increase?

- New orders randomly add ~10 line items per order.
- Each line item requires an update of the quantity in the stock table.
- There are ~100,000 stock items.
- Even so, two txns could each order 10 items, and just one of which is identical to each txn.
- So we have 18 items locked for update and one dead-locked.
- A third txn comes along and tries to lock on of these 19 items and so we now have 28 or 29 products locked.
- No order can commit until it has updated stock levels for all line items.
- And so it goes...

IBPhoenix
THE POWER WITHIN

What can we learn from this?

- Fundamentally database architecture and application design have a profound effect on application performance.
- Ideally performance issues should be fixed at this level.
- For the TPC-C benchmark this means looking at other ways to manage the update of the stock levels.
- Of course this takes the most time and effort and doesn't solve the immediate problem.

IBPhoenix
THE POWER WITHIN

Can we use the test harness to advise us on how to improve performance?

The Hypothesis

By running lots of tests with different configurations we can take averages of each test series and derive an optimal configuration.

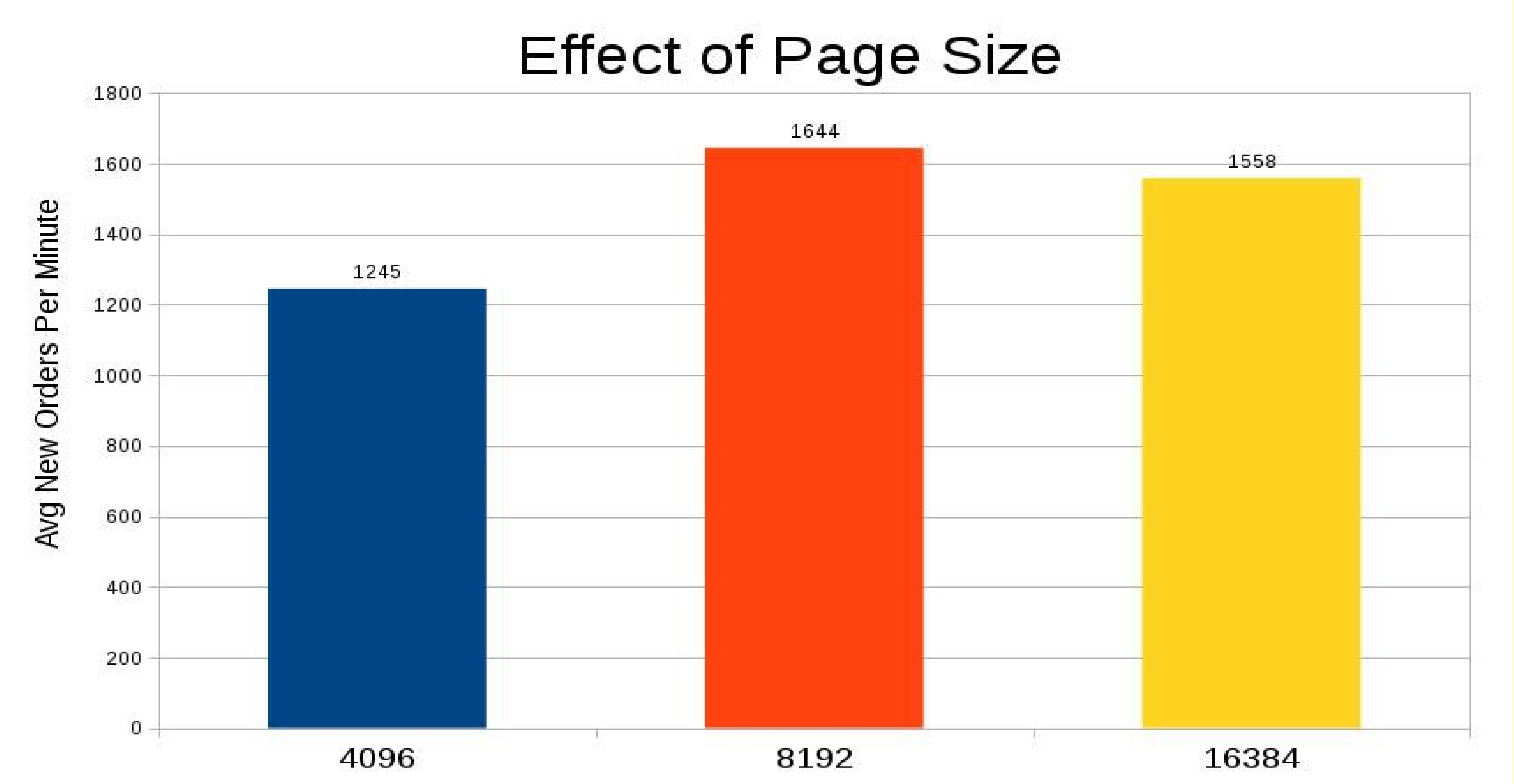
We will look at three configuration parameters

- Page Size
- Buffers
- Hash Slots

- For each parameter we will run our test series, changing a single value each time.

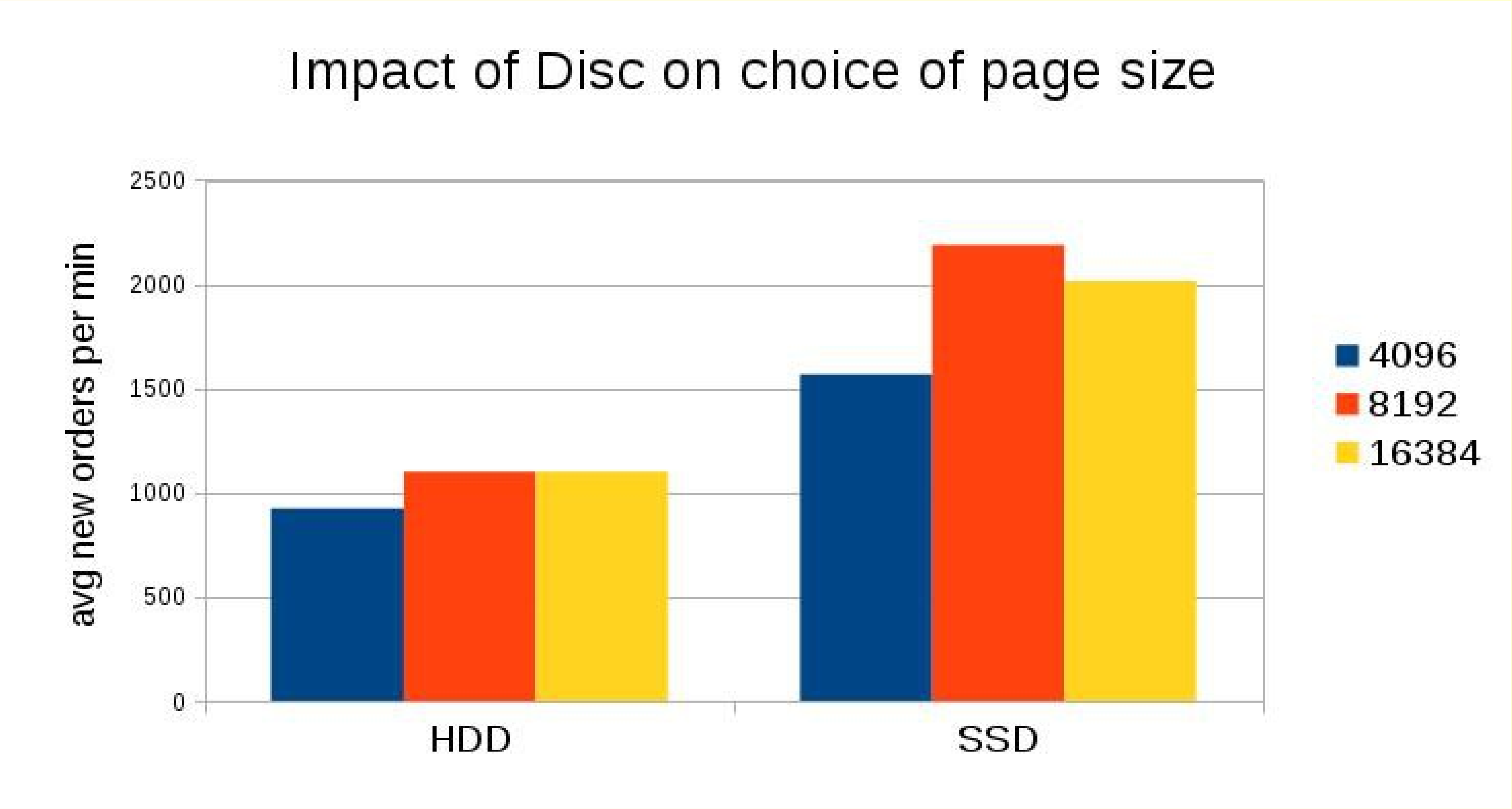
Page Size

8K appears to be ~17% better than 4K.
And 16K not so interesting.



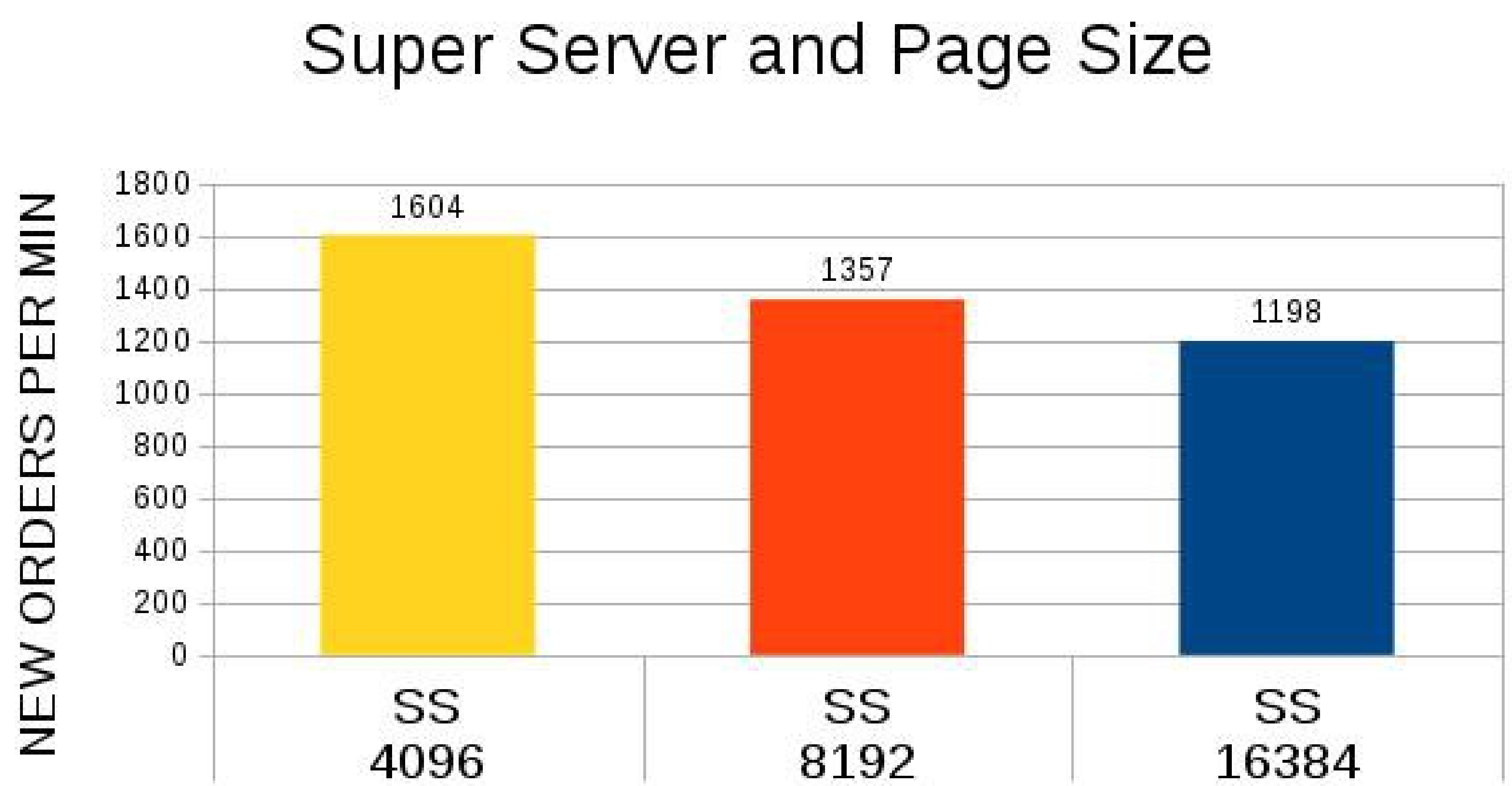
Page Size and Disc

But again, things are not so simple...



Super Server and Page Size

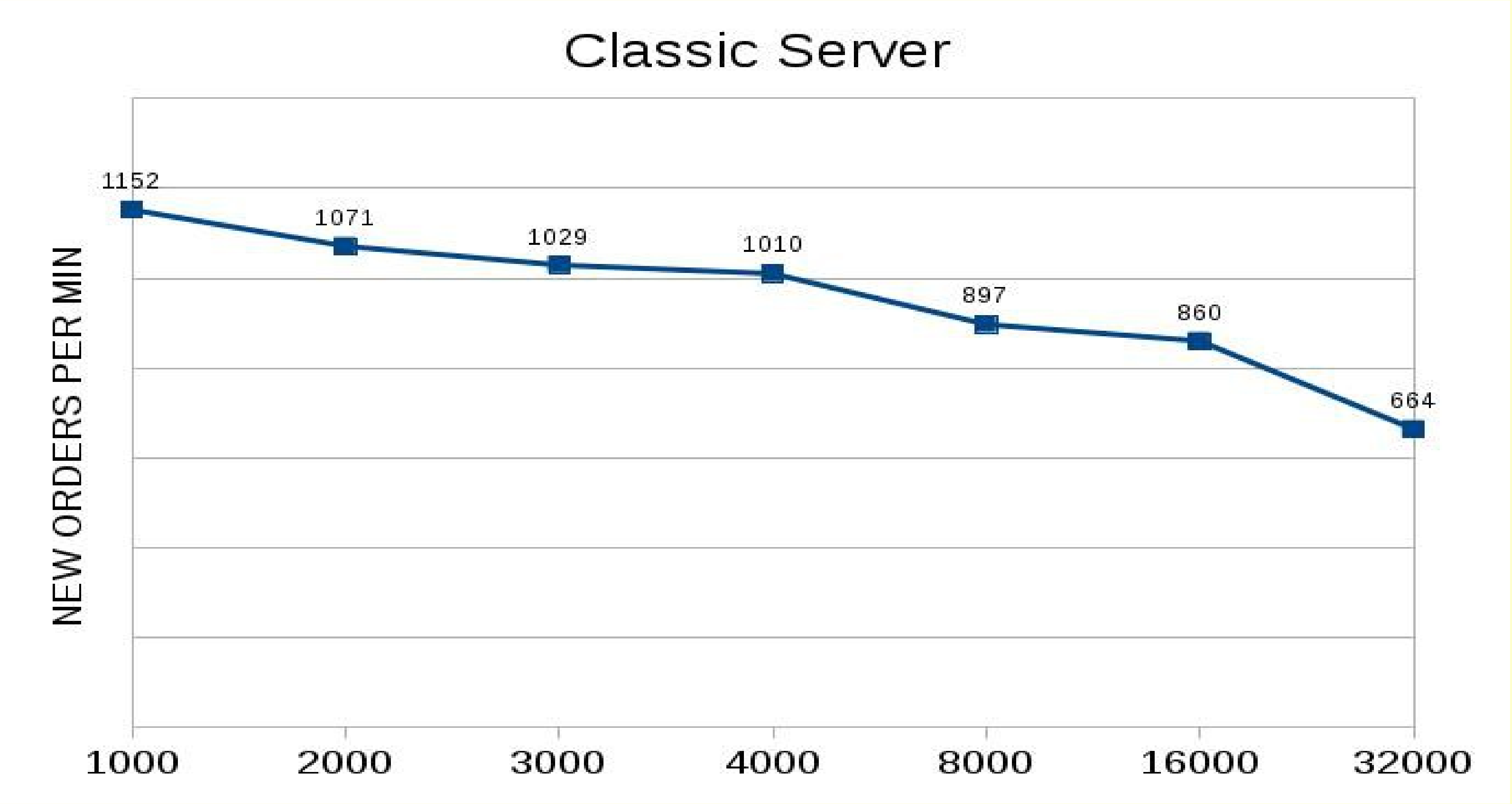
The previous slide indicated that 8K page size was optimum, but apparently this is not true for SS.



Buffers

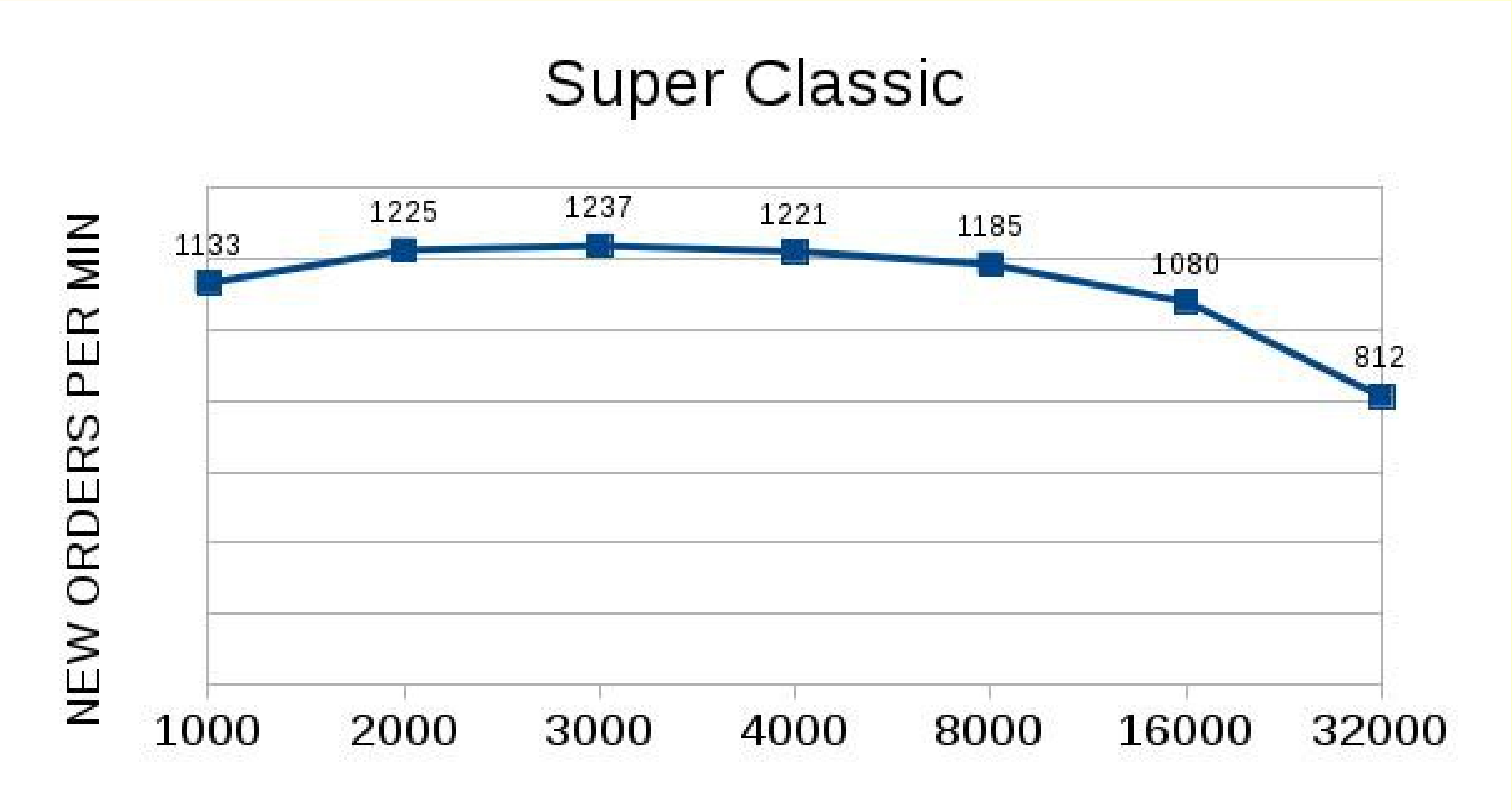
Buffers – Classic Server

Less is more

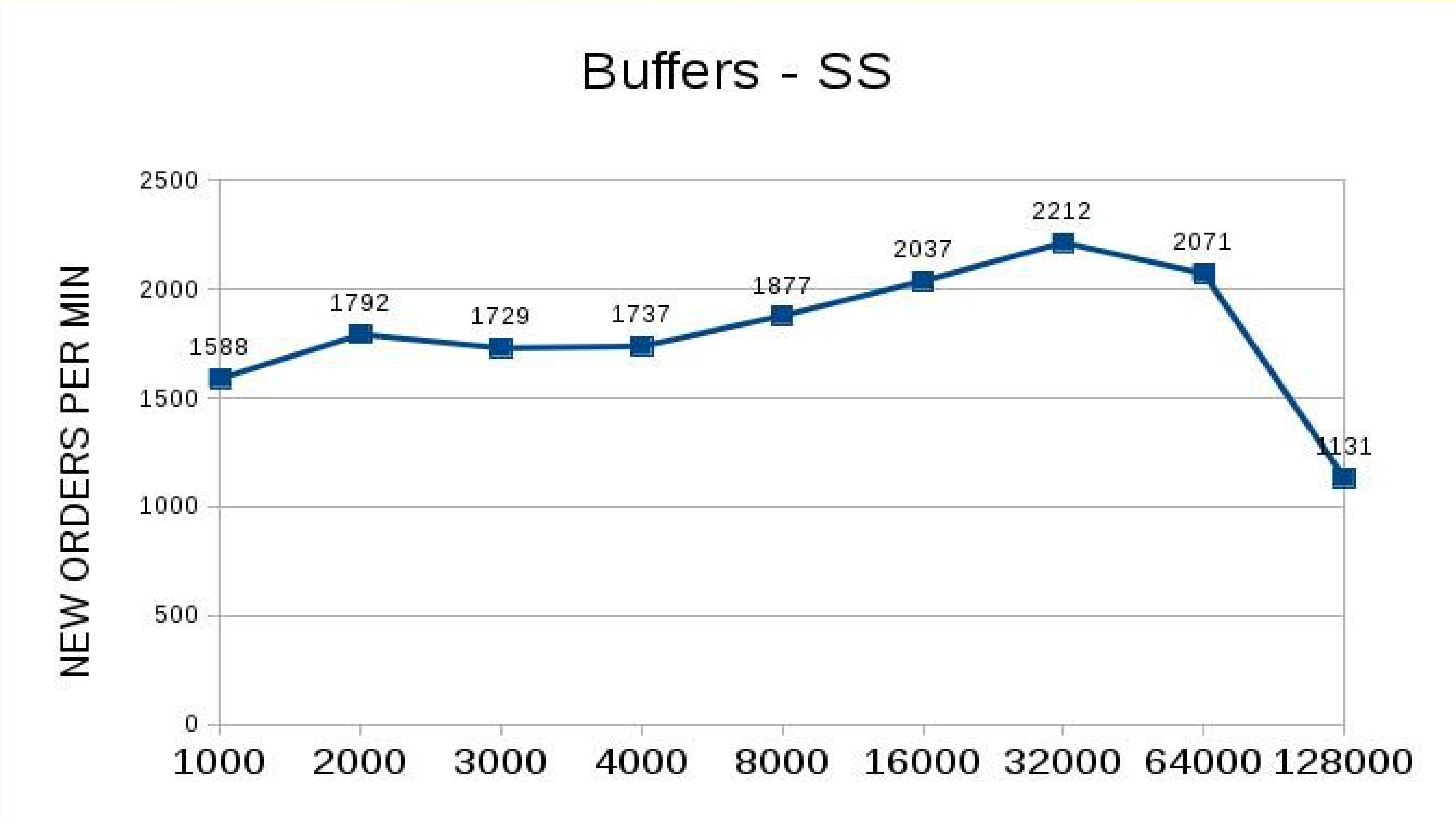


Buffers – Super Classic

- Can use more buffers
- ~7 % improved performance over classic



Buffers – Super Server



Chosen correctly can lead to 80% performance improvement over SC

Note impact of 128K buffers – disables file system caching!

Buffers

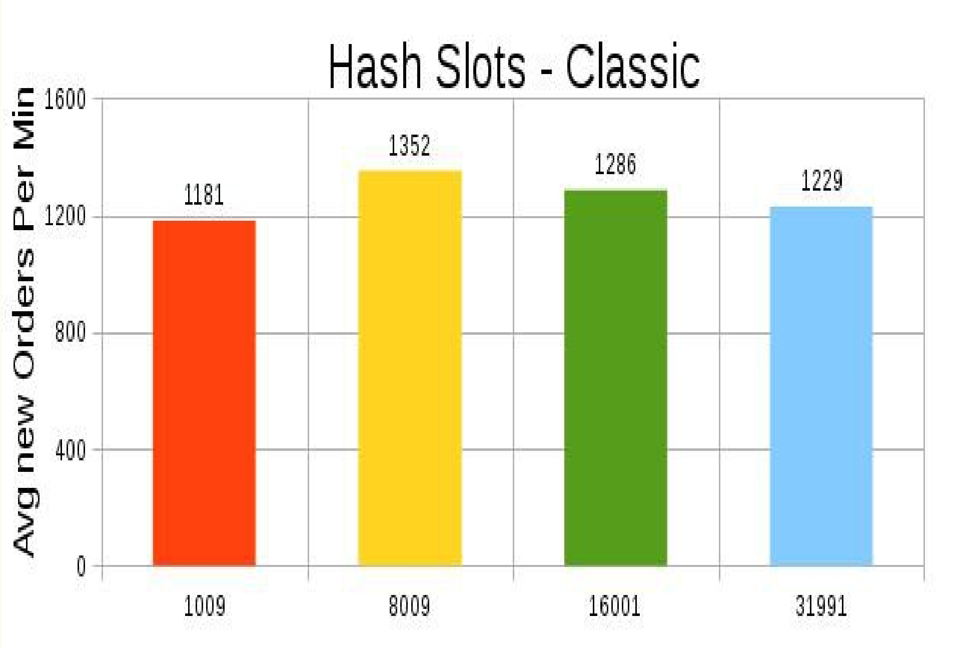
- Incorrect settings have a massive (bad) impact
- Each architecture has different behaviour
- Must analyse by architecture
- CS – smaller is better
- SC (2.5 only) prefers smaller over larger
- SS – increase buffers to look for sweet spot – more is not better.
- (Tests carried out on 10 GB DB)

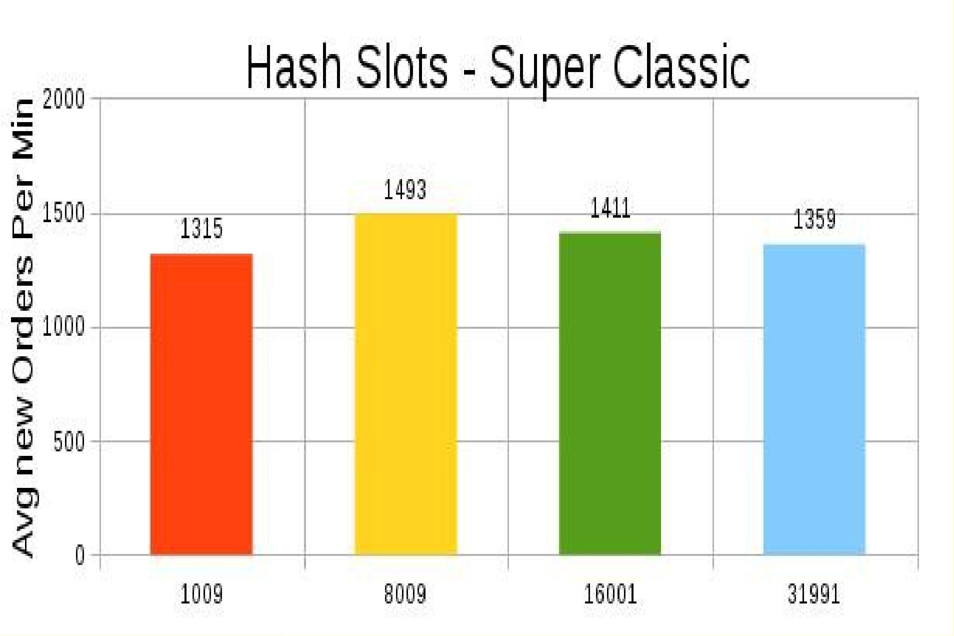
Hash Slots

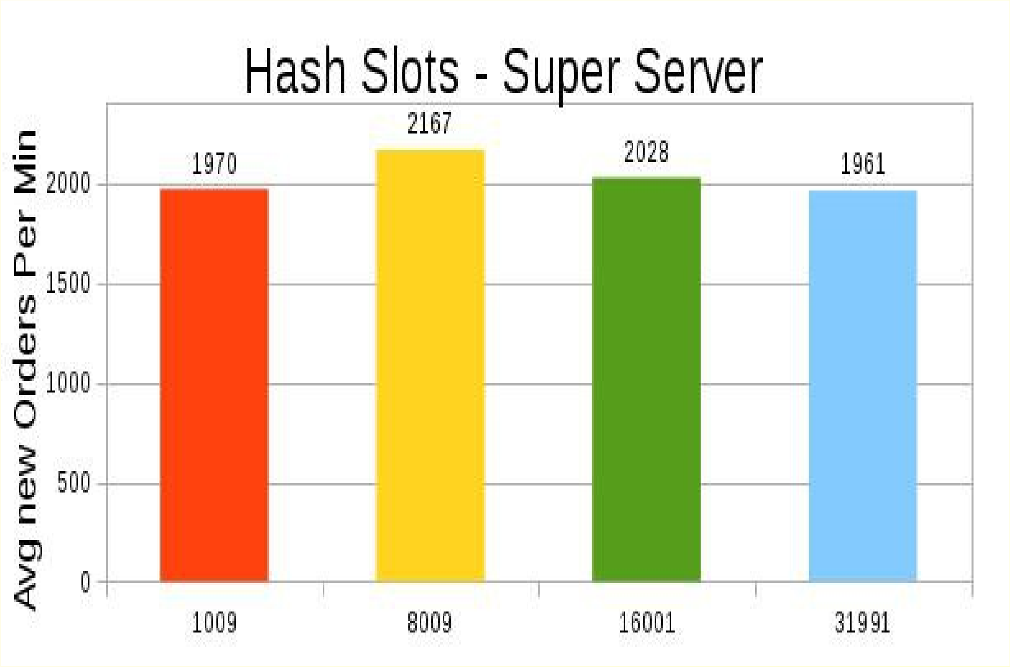
- All database access generates lock table activity, even just simple selects.
- Locks are located via a hash table.
- They are linked in chains.
- The chains are searched sequentially.
- More hash slots allows for shorter chains.

So in theory as connections increase we have more lock contention, and therefore more hash slots should improve performance.

The effect of different Hash Slots values







IBPhoenix
THE POWER WITHIN

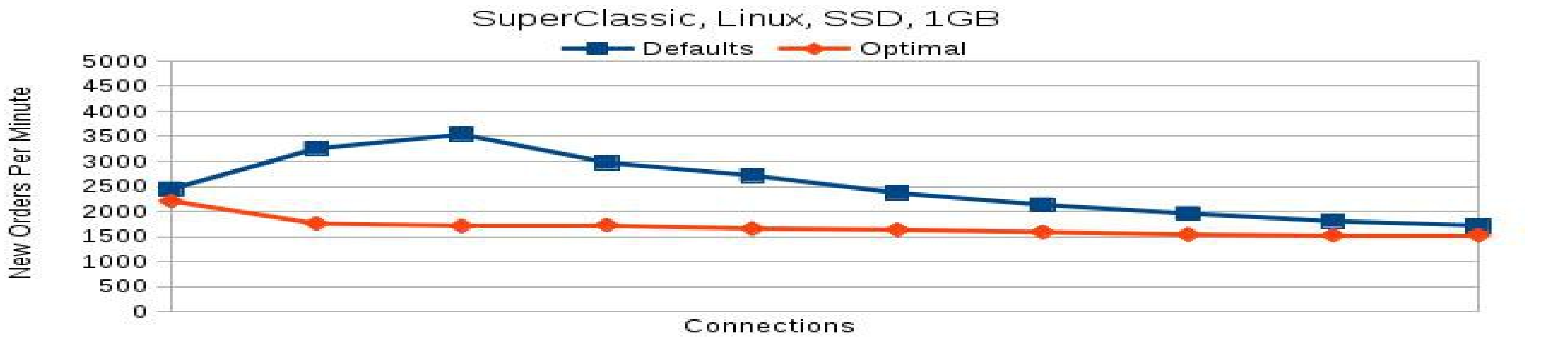
Towards an Optimal Config?

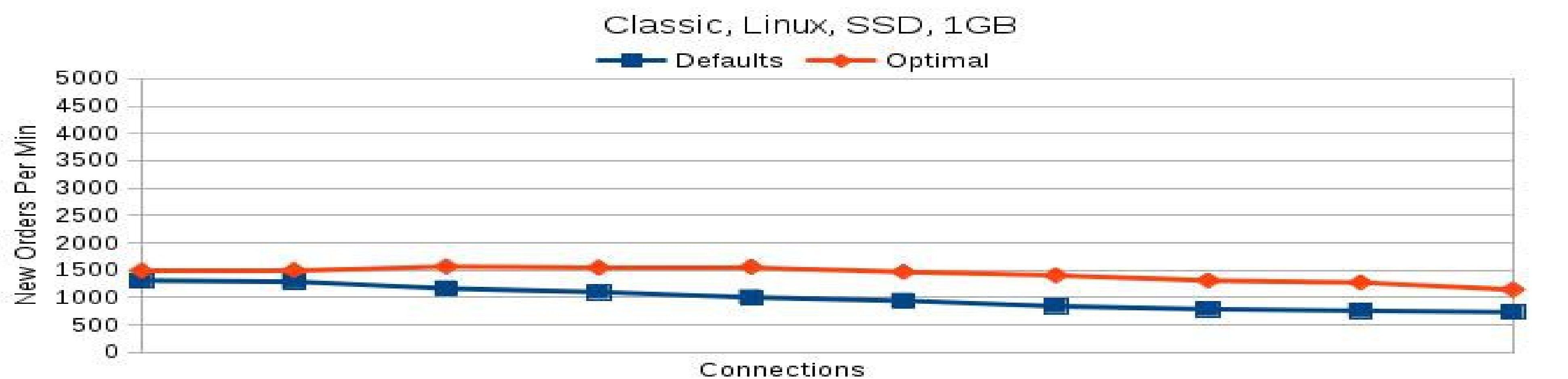
To summarize:

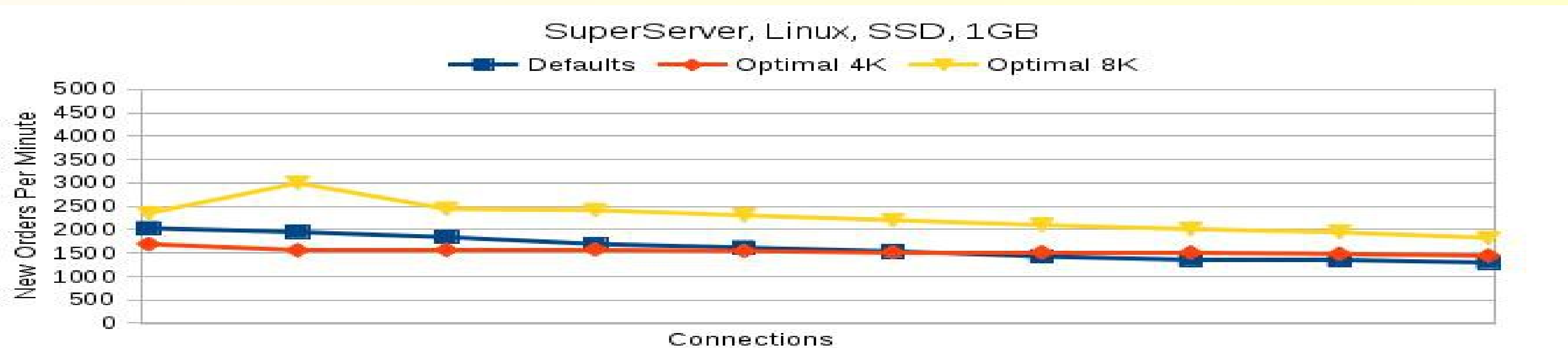
- 8K page size seems preferable for SC and CS
- 4K page size seems better for SS but we'll test both
- 8009 Hash Slots seems to improve performance for all architectures.
- Each arch. Has specific sweet spots for buffers
  - SC – 3000.
  - CS – 1000, perhaps 1500 ?
  - SS – 32000.

So, lets see how that works...

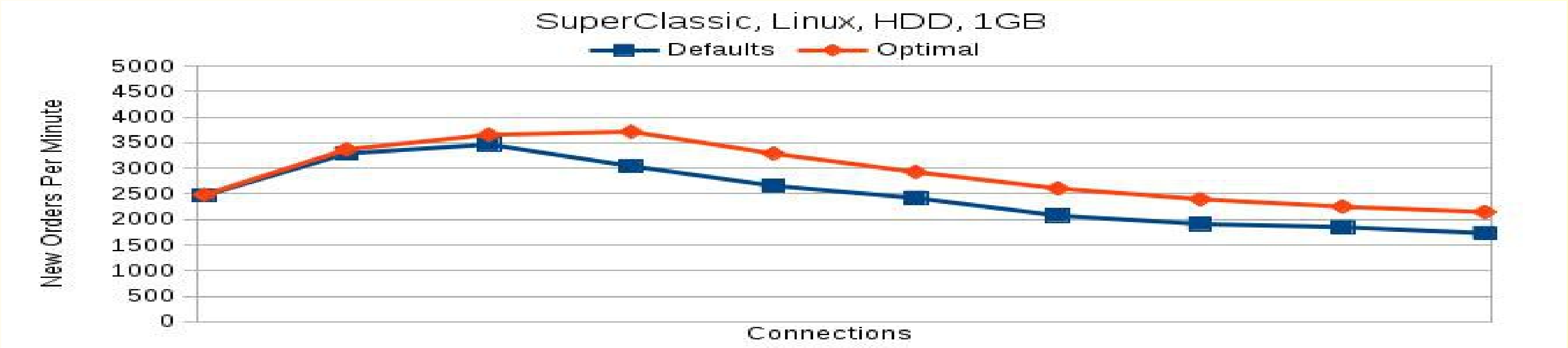
Compare Optimal to Defaults – Linux, SSD 1GB DB

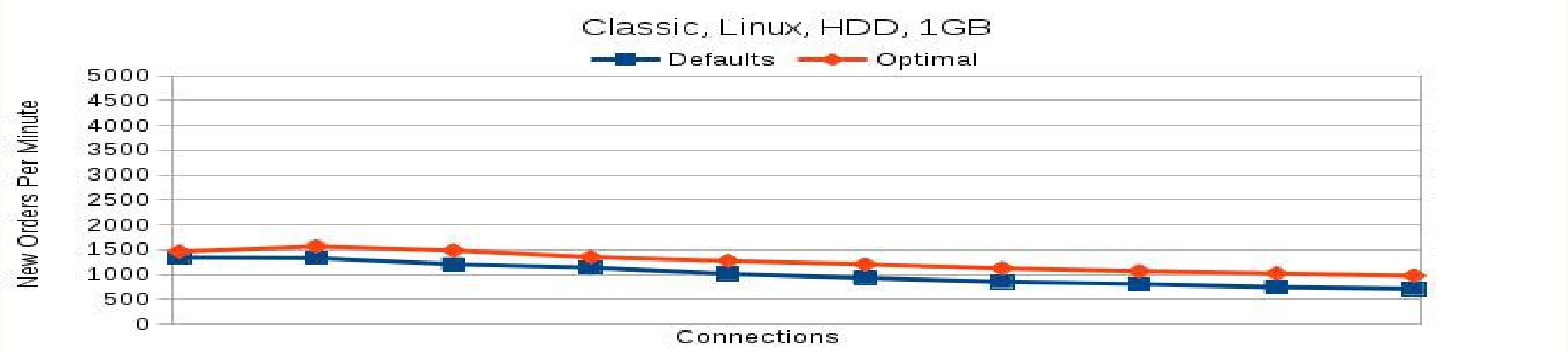


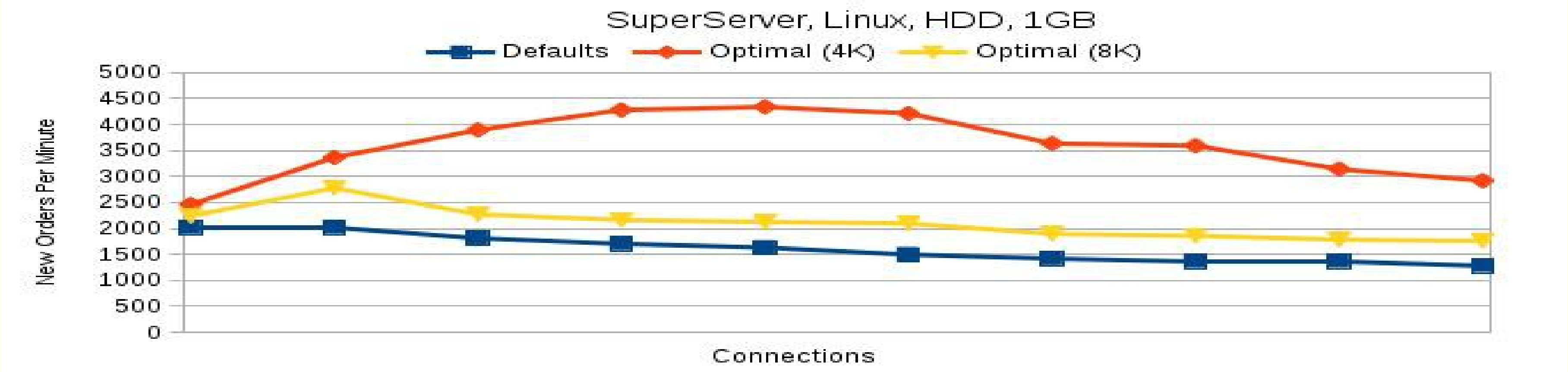




Compare Optimal to Defaults – Linux, HDD 1GB DB

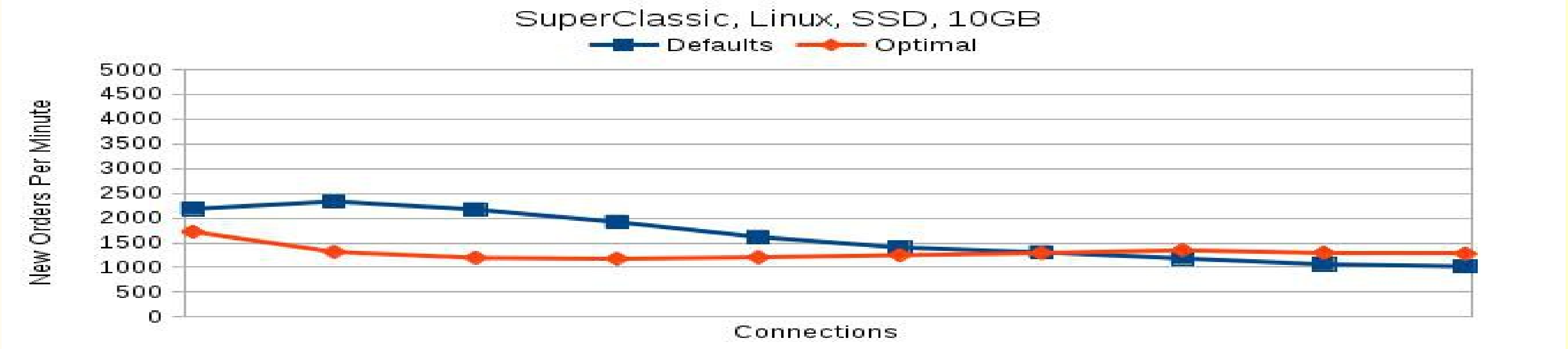


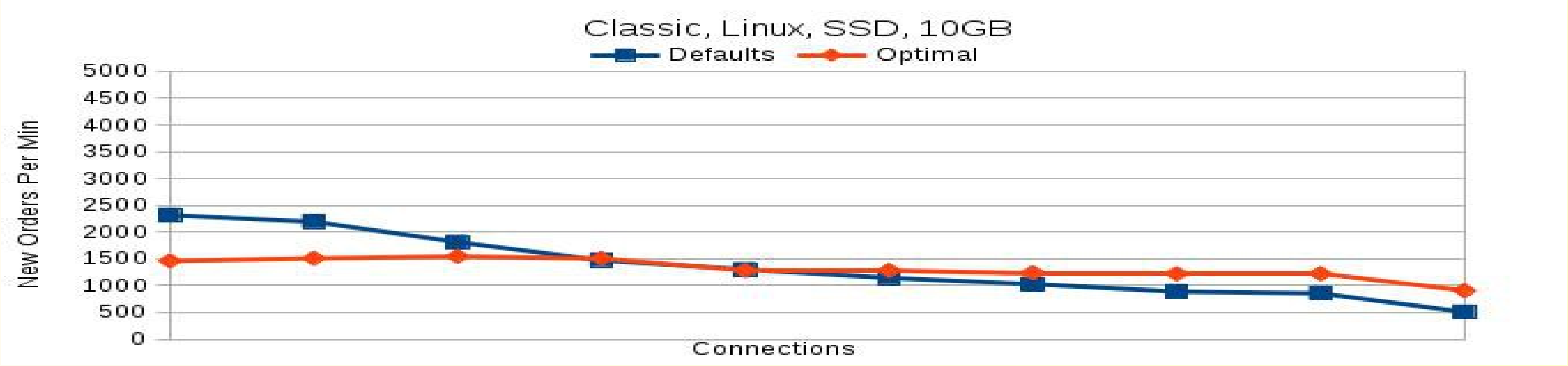


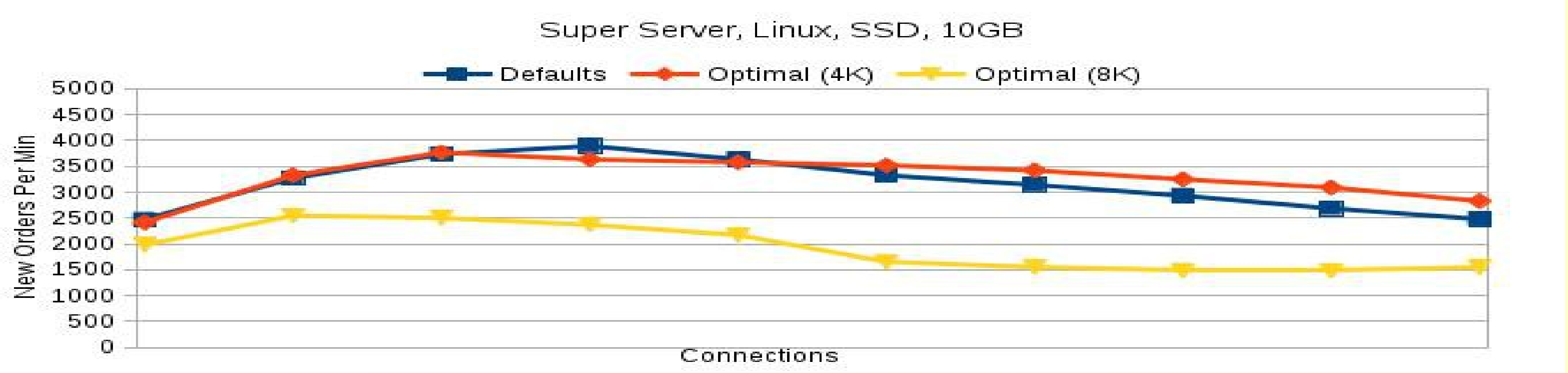


IBPhoenix
THE POWER WITHIN

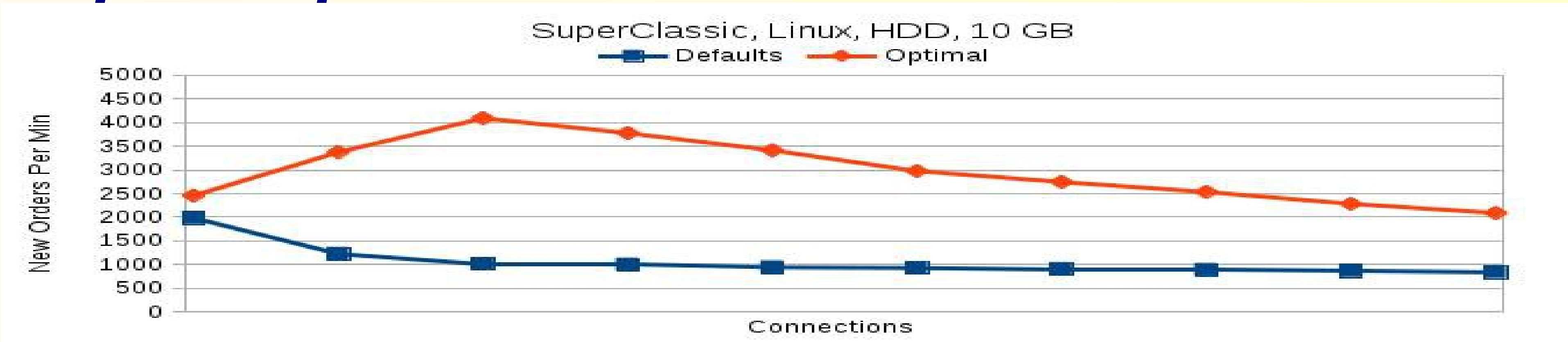
Compare Optimal to Defaults – Linux, SSD, 10GB

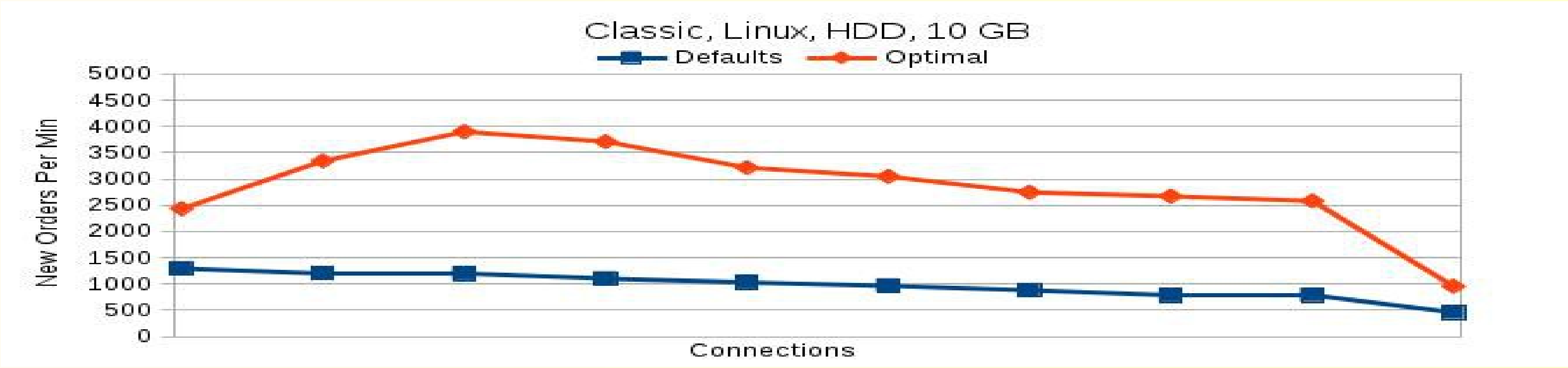


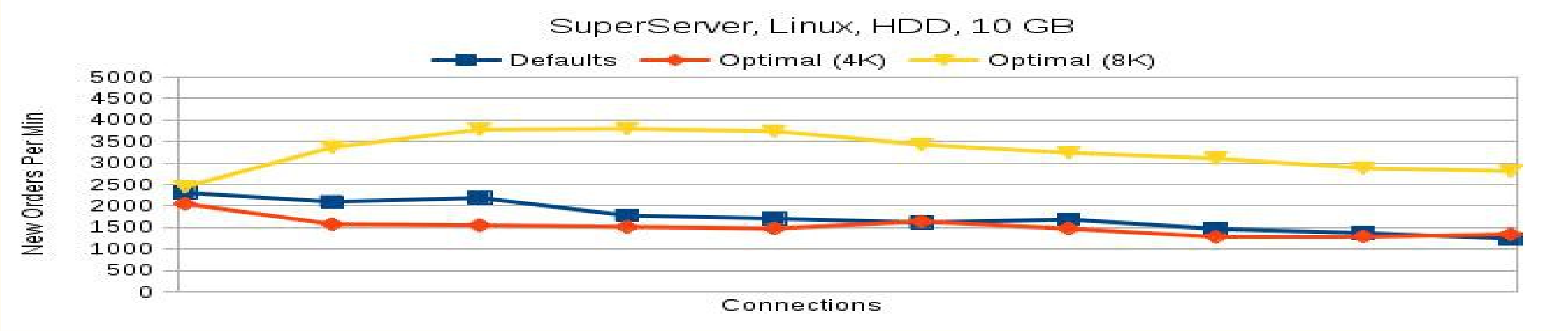




Compare Optimal to Defaults – Linux, HDD, 10GB

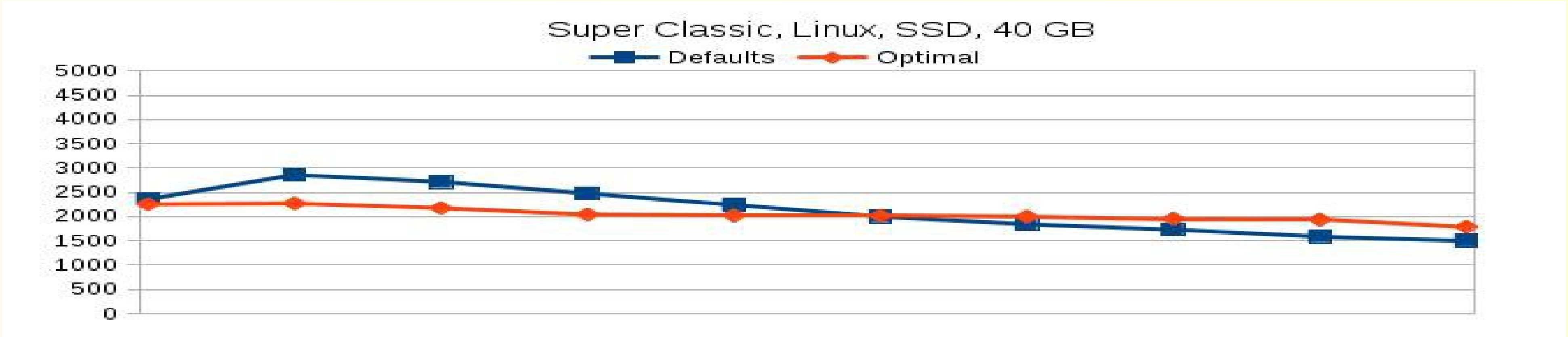


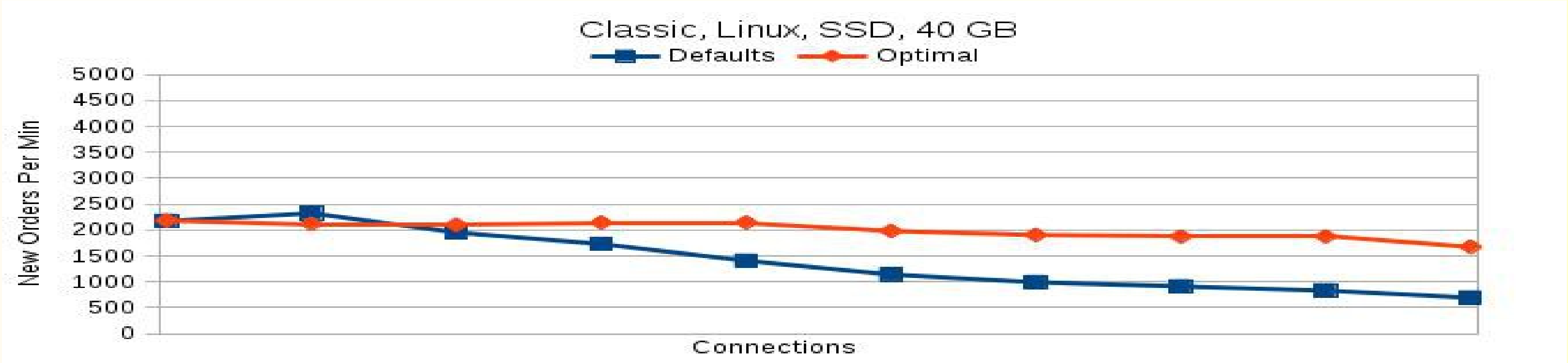


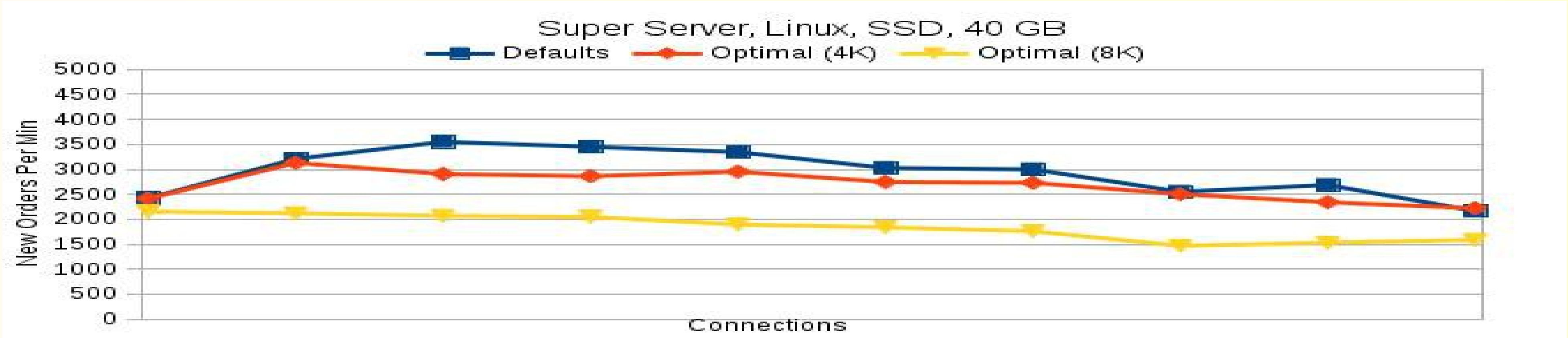


IBPhoenix
THE POWER WITHIN

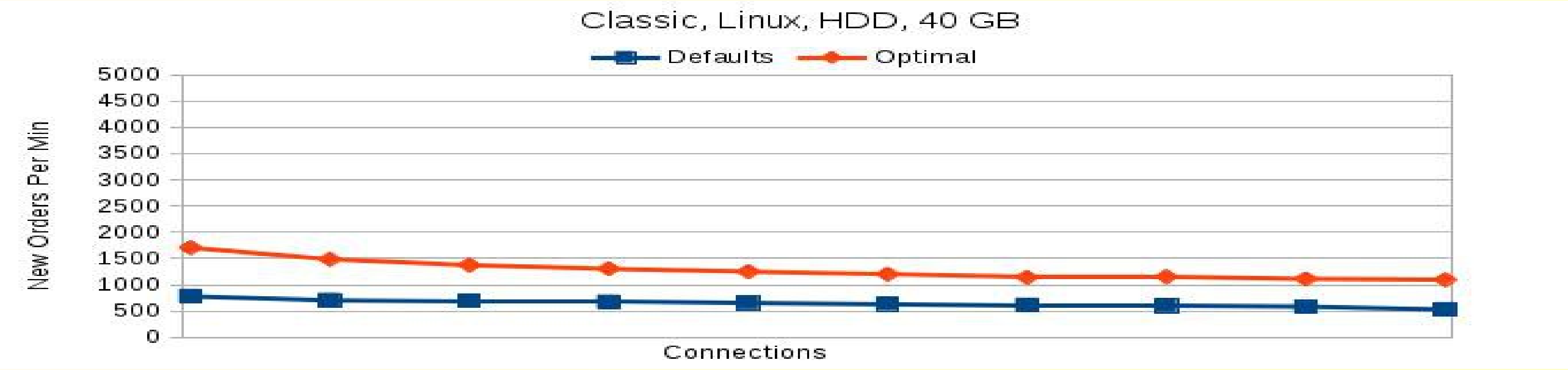
Compare Optimal to Defaults – Linux, SSD, 40GB

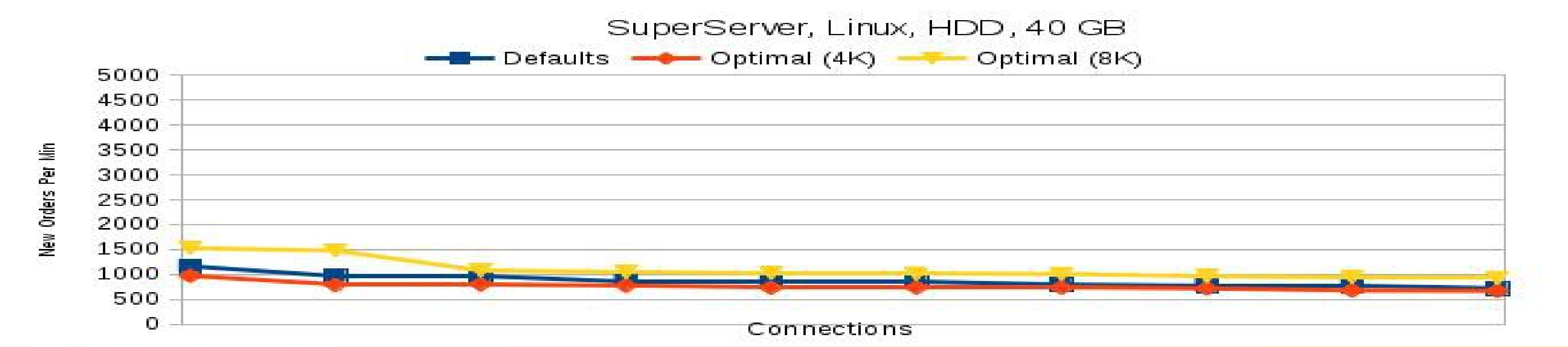






Compare Optimal to Defaults – Linux, HDD, 40GB



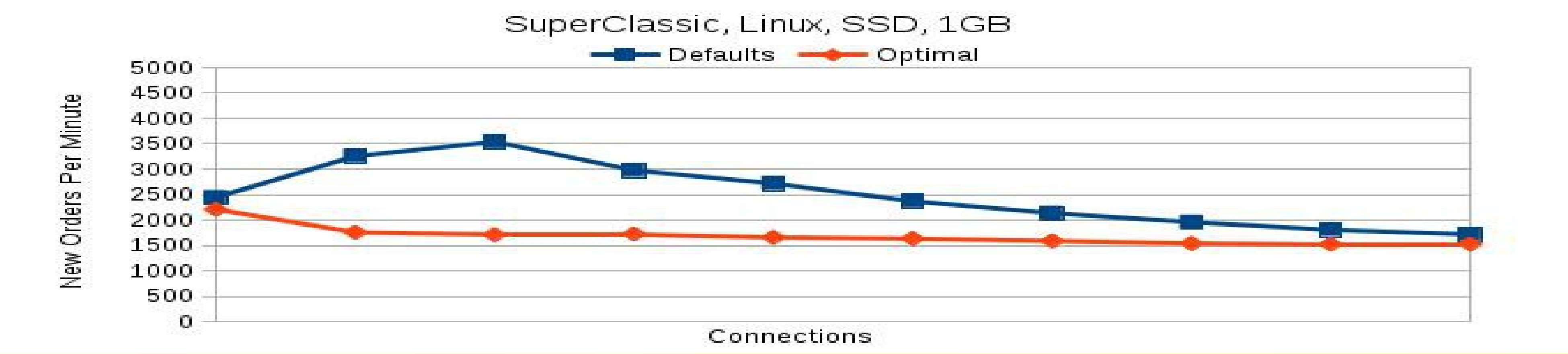


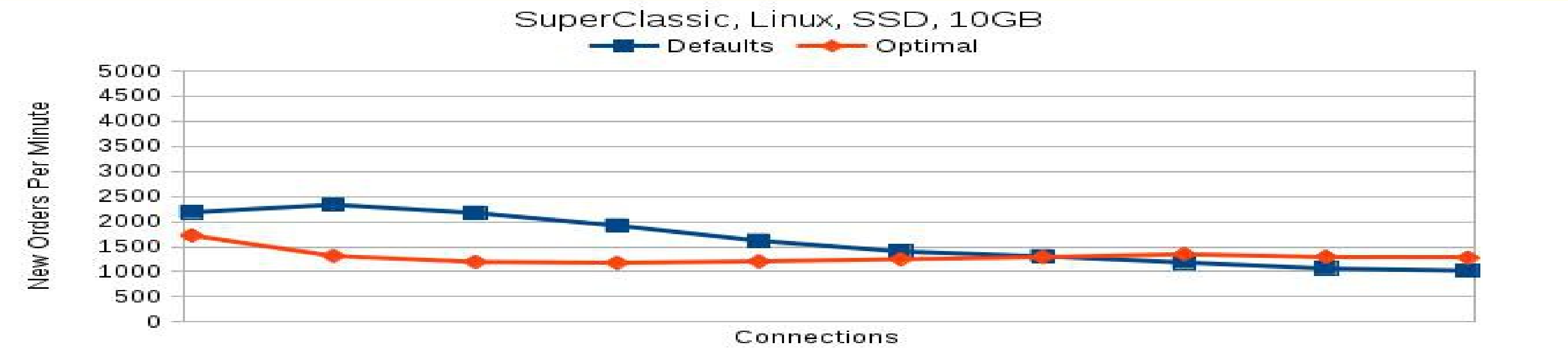
IBPhoenix
THE POWER WITHIN

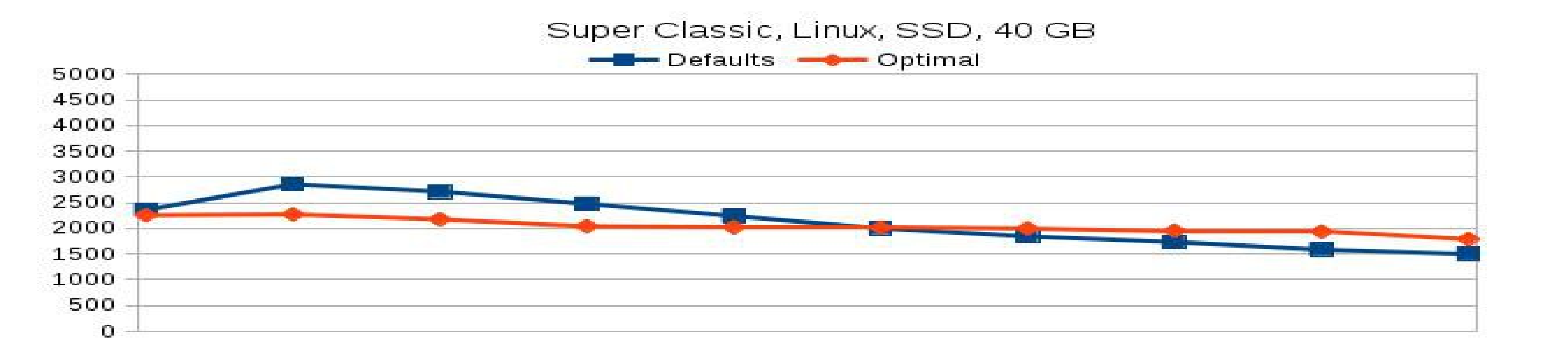
The big question

- Why don't SSDs seem to respond to our configuration techniques?

SSDs and SuperClassic - a recap

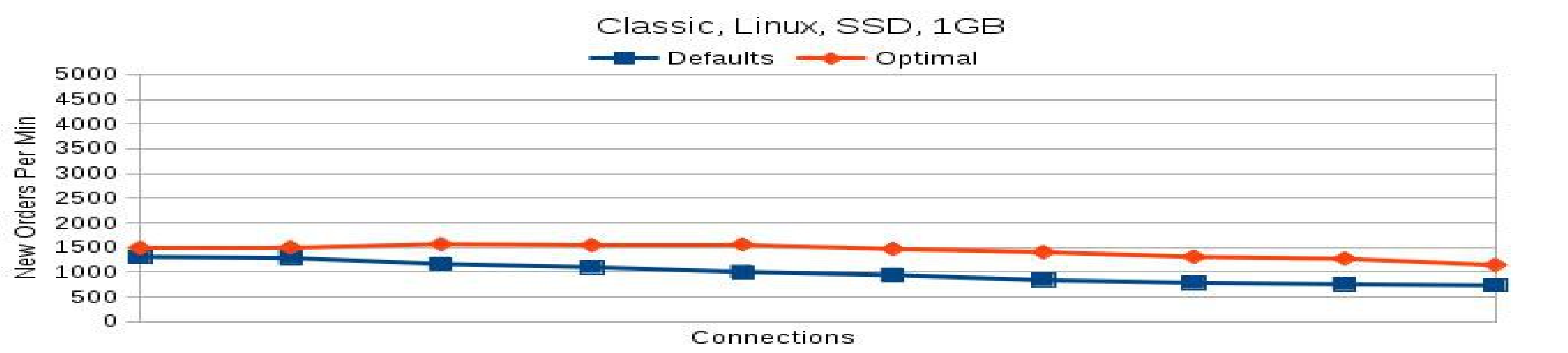


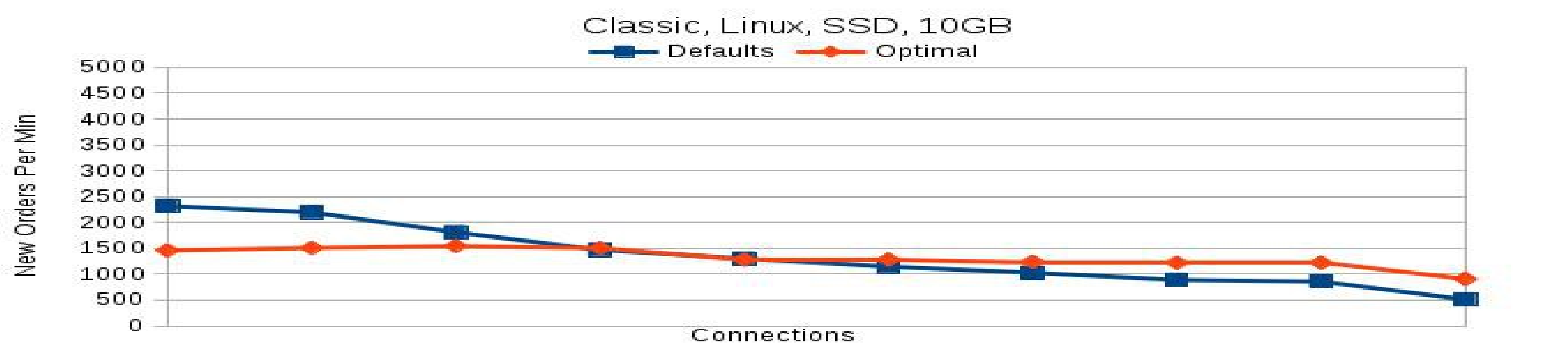


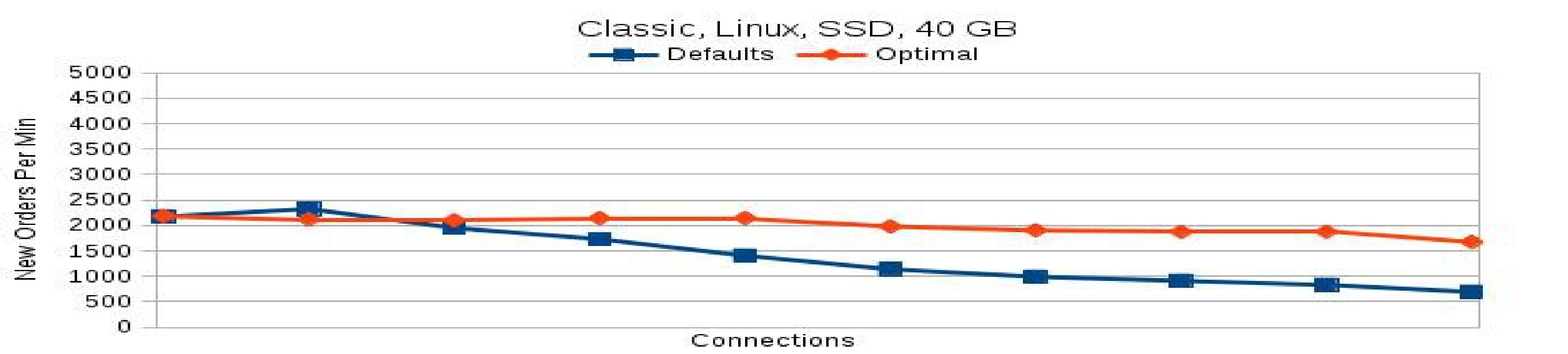


IBPhoenix
THE POWER WITHIN

SSDs and Classic - a recap

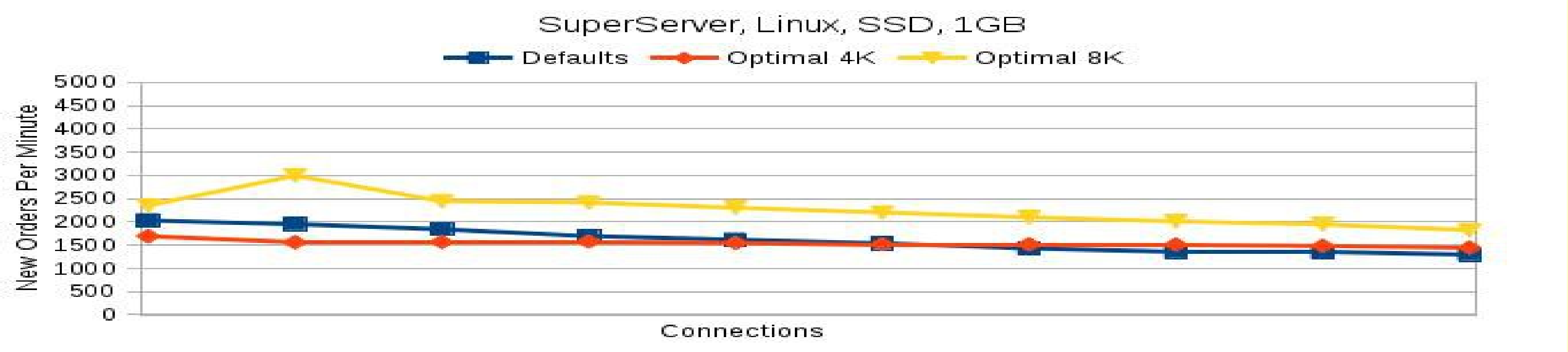


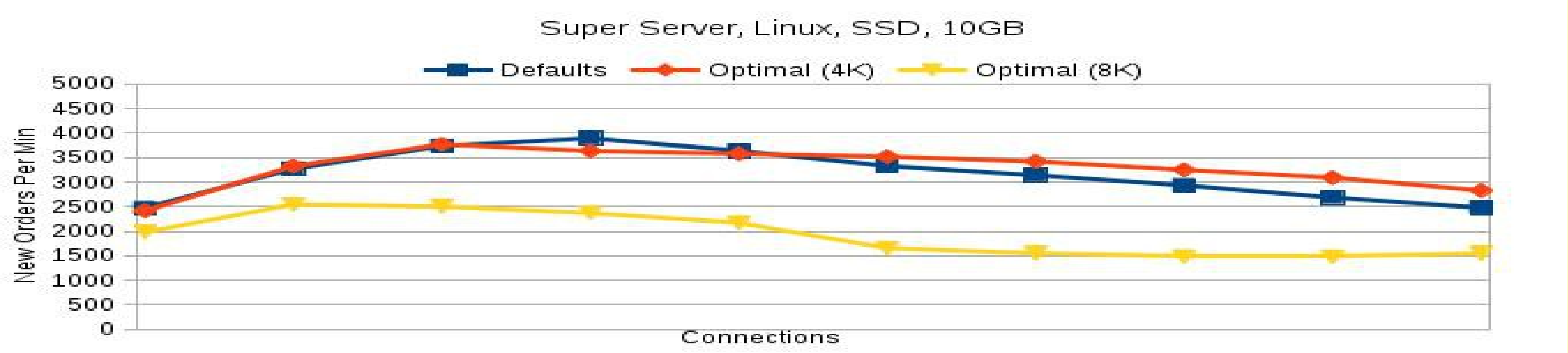


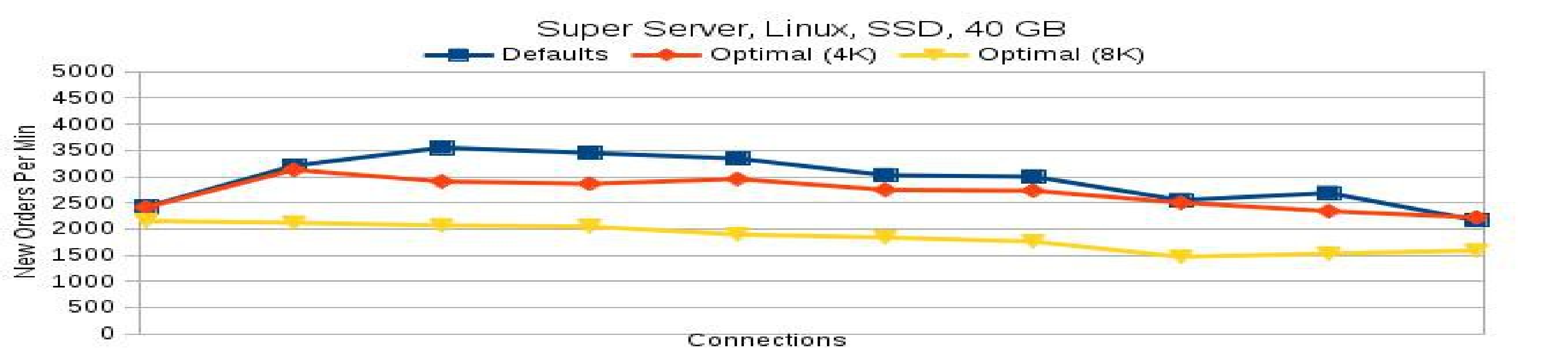


IBPhoenix
THE POWER WITHIN

SSDs and SuperServer - a recap







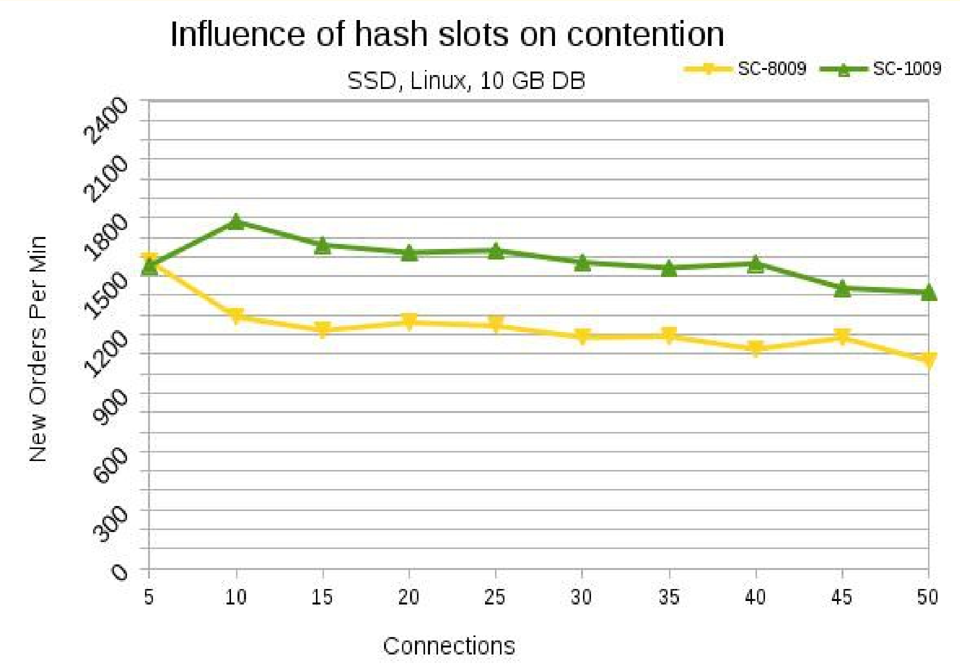
IBPhoenix
THE POWER WITHIN

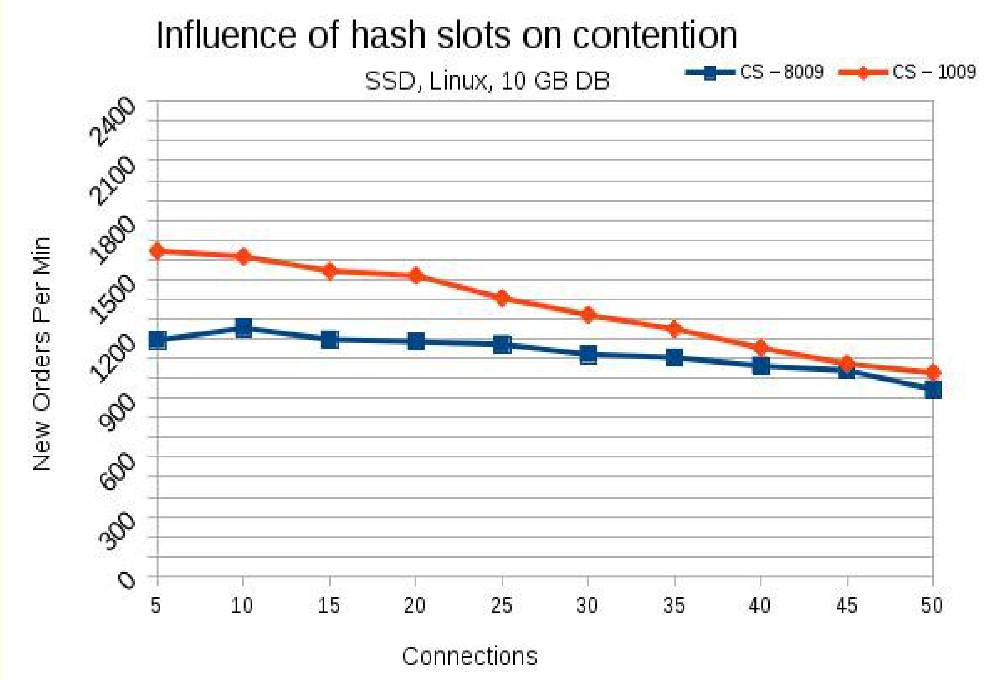
So why has SSD performance degraded?

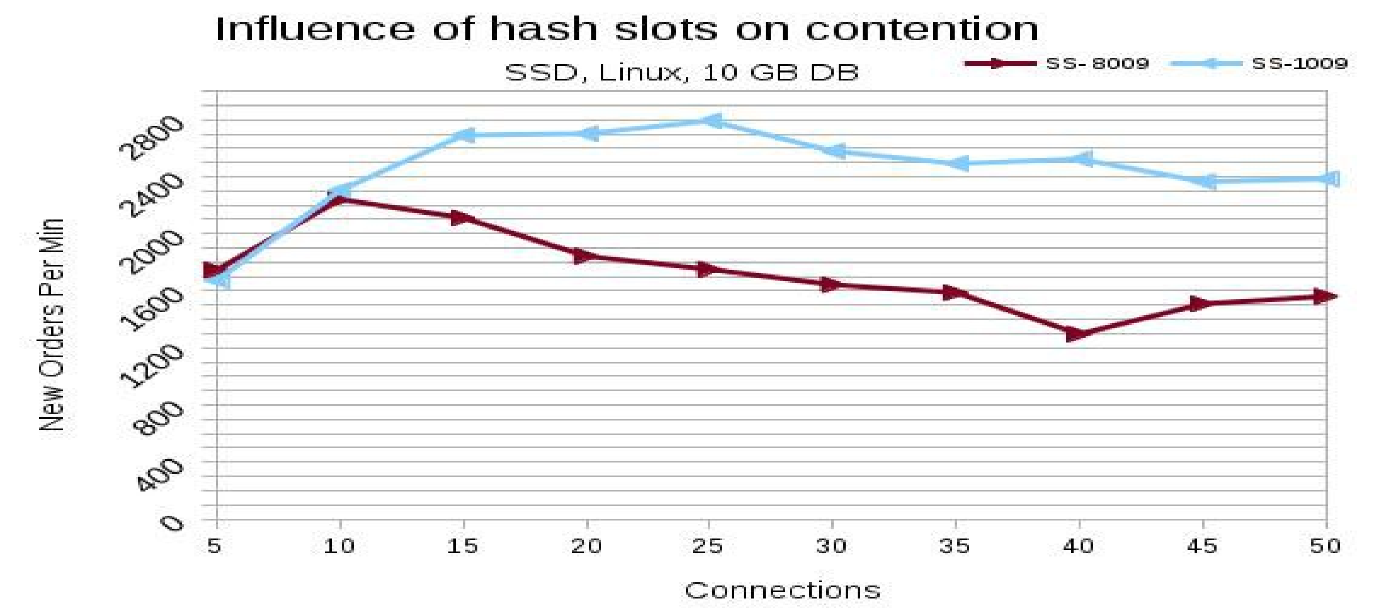
- While reviewing this presentation I noticed that there was no analysis of the hash slots data.
- Perhaps the answer lies there?
- Let's take a look.

IBPhoenix
THE POWER WITHIN

Perhaps the hash slots change is the problem with diminished SSD performance?







IBPhoenix
THE POWER WITHIN

And our hypothesis?

- It clearly worked for HDDs
- SSDs did not respond or actually performed more poorly due to inadequate analysis (but we didn't know that until we had done the tests.)
- Ultimately this hypothesis failed but that is not a bad thing – we have learnt that:
  - SSDs perform very differently to HDDs
  - Determining optimal configurations requires much more refined data analysis.
  - Optimal Settings do not transfer automatically to a different setup.
  - Bad configuration choices have just as much an impact on performance as good ones do. ☺

Where next with this research?

- Obviously work needs to be done to understand better how to get the best performance out of SSDs
- Can the optimal configuration be refined further?
  - What happens when we try different hash slots with our 'optimal' page size and buffers?
  - Ditto for a different page size.
- What happens if we play around a bit with the File System Cache size and the number of buffers?
- What happens if we remove the sources of lock contention in the application/data model?

Lots of questions that still need answers.

IBPhoenix THE POWER WITHIN

Summary

For Firebird 2.5.3 and this test harness...

- SSDs are better than HDDS, especially for VLDBs
- Linux and Windows perform similarly, except for SS under Windows.
- Usually SS is better than SC which is better than CS
- 8K page size is usually better than 4K except for SS for HDDs
- Smaller buffers are better for CS
- SC doesn't care neither for large buffers nor small
- SS likes large buffers but not so big as to disable the file system cache.
- SSDs do not appear to respond to the same performance tweaks as HDDs.

IBPhoenix
THE POWER WITHIN

Conclusion

- There is a fine balance to be had in all performance tweaking.
- Test everything.
- There is no universal optimised config.

IBPhoenix
THE POWER WITHIN

Questions?

And finally, a big thankyou to all the sponsors who have helped make this conference possible...

MOSCOW EXCHANGE
Platinum

SITA SOFTWARE
Platinum

IBPhoenix
Platinum

IBSurgeon
Platinum

REDSOFT

IBObjects

CopyCat

Upscene

IBPhoenix
THE POWER WITHIN