
CERN

A Latency Picture is Worth a Thousand Storage Metrics

Luca Canali – CERN

About Me

- Senior DBA and Team Lead at CERN IT
  - Joined CERN in 2005
- Working with Oracle RDBMS since 2000

- Sharing knowledge with the Oracle community
- Home page: http://cern.ch/canali
- Blog: http://externaltable.blogspot.com
- Twitter: @LucaCanaliDB

Outline

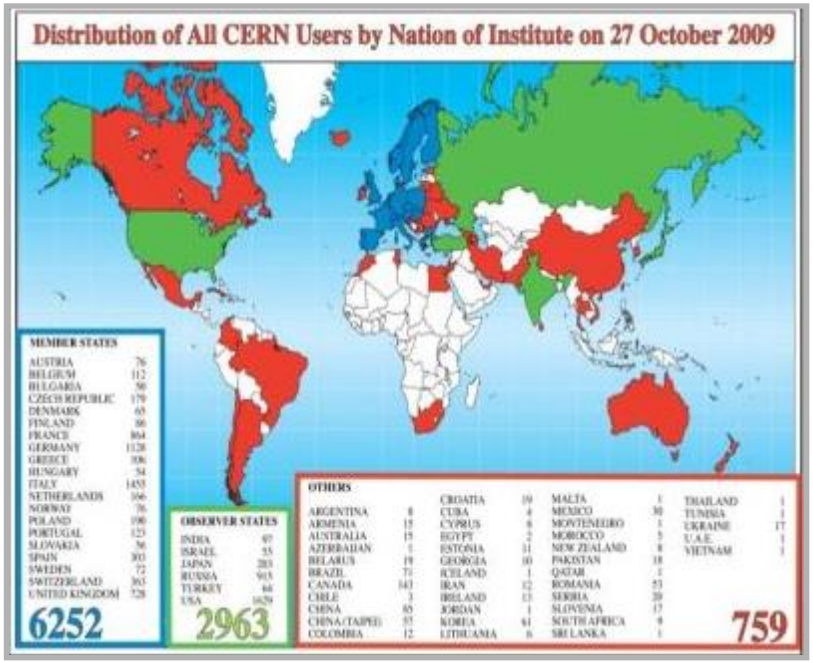
- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

Outline

- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

CERN

- European Organization for Nuclear Research founded in 1954
- Membership: 21 Member States + 7 Observers
- 60 Non-member States collaborate with CERN
- 2400 staff members work at CERN as personnel + 10000 researchers from institutes world-wide



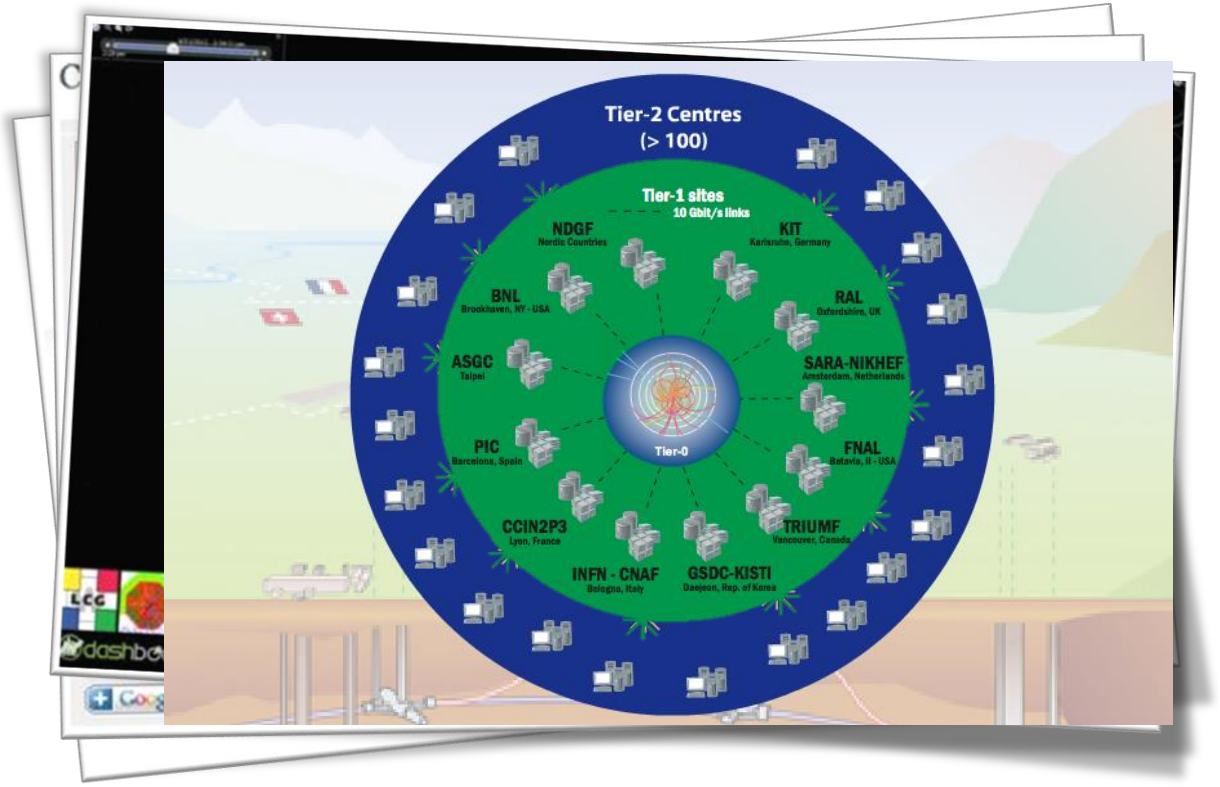
LHC, Experiments, Physics

- Large Hadron Collider (LHC)
  - World's largest and most powerful particle accelerator
  - 27km ring of superconducting magnets
  - Currently undergoing upgrades, restart in 2015
- The products of particle collisions are captured by complex detectors and analyzed by software in the experiments dedicated to LHC
- **Higgs boson discovered!**

- The Nobel Prize in Physics 2013 was awarded jointly to François Englert and Peter W. Higgs *"for the theoretical discovery of a mechanism that contributes to our understanding of the origin of mass of subatomic particles, and which recently was confirmed through the discovery of the predicted fundamental particle, by the ATLAS and CMS experiments at CERN's Large Hadron Collider"*

WLCG

- The world's largest scientific computing grid



More than 100 Petabytes
of data stored in a custom file system and analysed.
Increasing: 20+ Petabytes/year

CPU: over 250K cores
Jobs: 2M per day

160 computer centres in 35 countries

More than 8000 physicists with real-time access to LHC data

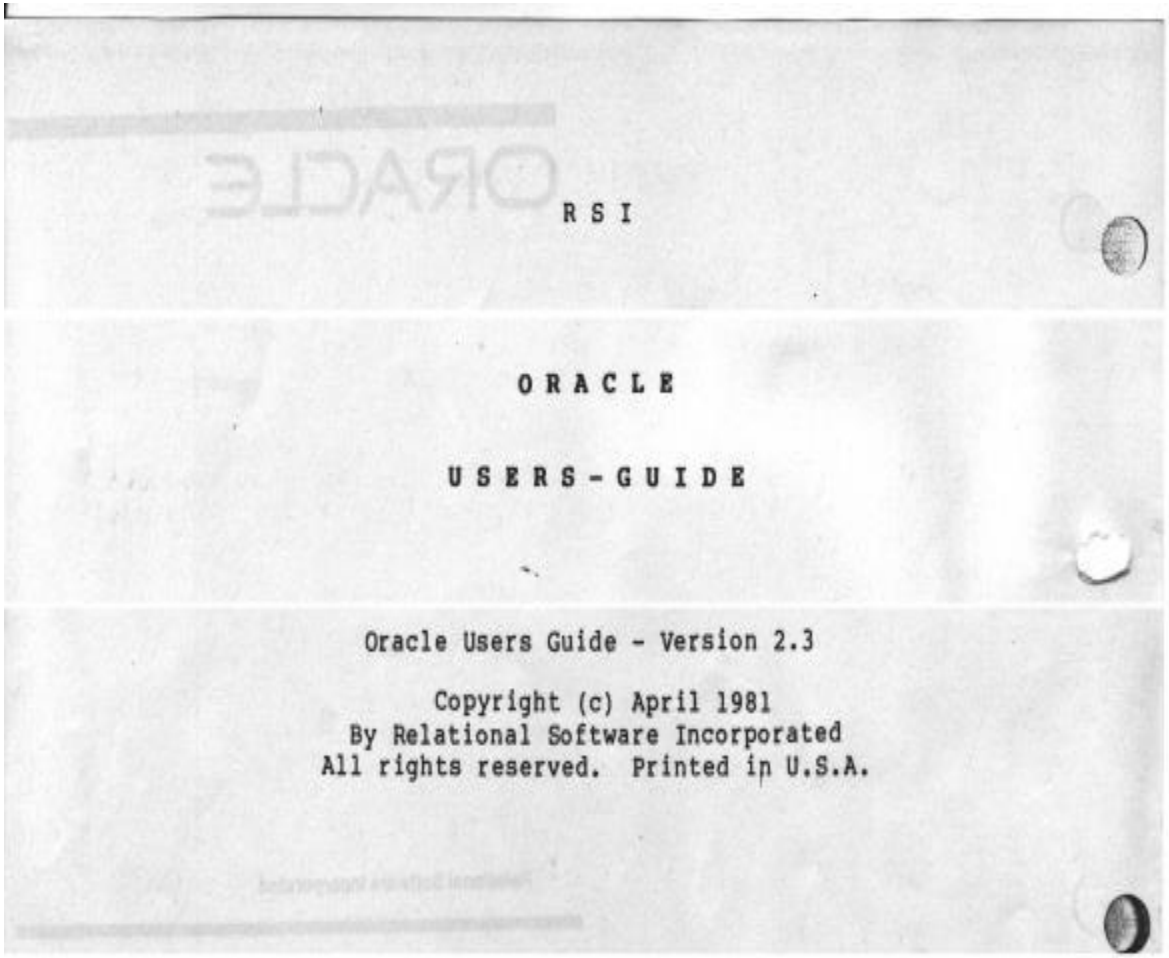
Oracle at CERN

Since 1982

(*accelerator controls*)

More recent:

use for Physics



RSI

ORACLE

USERS-GUIDE

Oracle Users Guide - Version 2.3

Source: N. Segura Chinchilla, CERN

CERN's Databases

- ~100 Oracle databases, most of them RAC
  - Mostly NAS storage plus some SAN with ASM
  - ~500 TB of data files for production DBs in total

- Examples of critical production DBs:
  - LHC logging database ~170 TB, expected growth up to ~70 TB / year
  - 13 production experiments' databases ~10-20 TB in each
  - Read-only copies (Active Data Guard)

Database SERVICES

At the heart of CERN, LHC and Experiment Operations

CERN IT Department

http://cern.ch/it-dep/db/

IT/DB GROUP

Experiment Offline Databases
9 Production DBs, 7 Integration DBs
5 Tests DBs, 8 (Active) Data Guards

Streams

Streams

Tier-1 Centres

NDGF, GridKa, BNL, IN2P3, ASGC, SARA-NIKHEF, PIC, FNAL, CCIN2P3, TRIUMF, INFN - CNAF

Experiment Online Databases
4 Production DBs, 6 (Active) Data Guards

Data

LHC Experiments

LHCb

CMS

ATLAS

ALICE

RAW Data

CASTOR
CERN Advanced STORage manager

21 Production DBs
4 Development DBs
1 Data Guard

Middleware
73 Application Servers

Administrative/IT/ Engineering Databases
17 Production DBs
11 Development DBs
4 Ref/Test DBs

Accelerators ACC
12 Production DBs

LHC Operations

PARTNERS:

CERN openlab

ORACLE

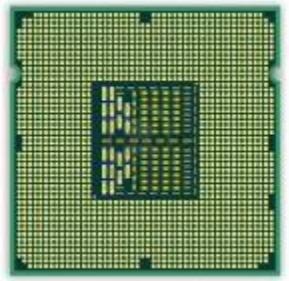
Outline

- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

Latency

- Latency, a measure of time.
  - In the context of this presentation: time to access data






Why is Storage Latency Important?

- Performance analysis and tuning:
  - Where is the time spent during a DB call?
  - What response time do the user sessions experience from the DB?

- OLTP-like workloads:
  - Response time can be dominated by I/O latency
  - Examples: index-based access, nested loops joins

Physical Sources of Latency

- Blocking I/O calls:
  - Think access to a large table via an index
  - Random access
  - HDD: head movement and disk spinning latency

What can we do: Hardware

- Current trends for HW
  - Size I/O subsystem for IOPS not TB
  - Large flash cache in storage
  - Flash cards on servers
  - Servers with large amounts of memory
    - Avoid reading from storage!

- A balance act performance vs. cost

What can we do: Software

- Big gains in application/SQL optimization
  - SW optimizations beat HW optimizations most of the times

- Oracle tuning:
  - Understanding when single-block access is or is not optimal
  - Full scan vs. index-based access
  - Hash join vs. nested loops
  - In general: get a good execution plan!

So Where is the Problem?

DB Admin:

- Storage is slow!

Storage Admin:

- The problem is with the DB!

- Reality check:
  - Lack of clear storage performance data.
  - Changing database workloads.

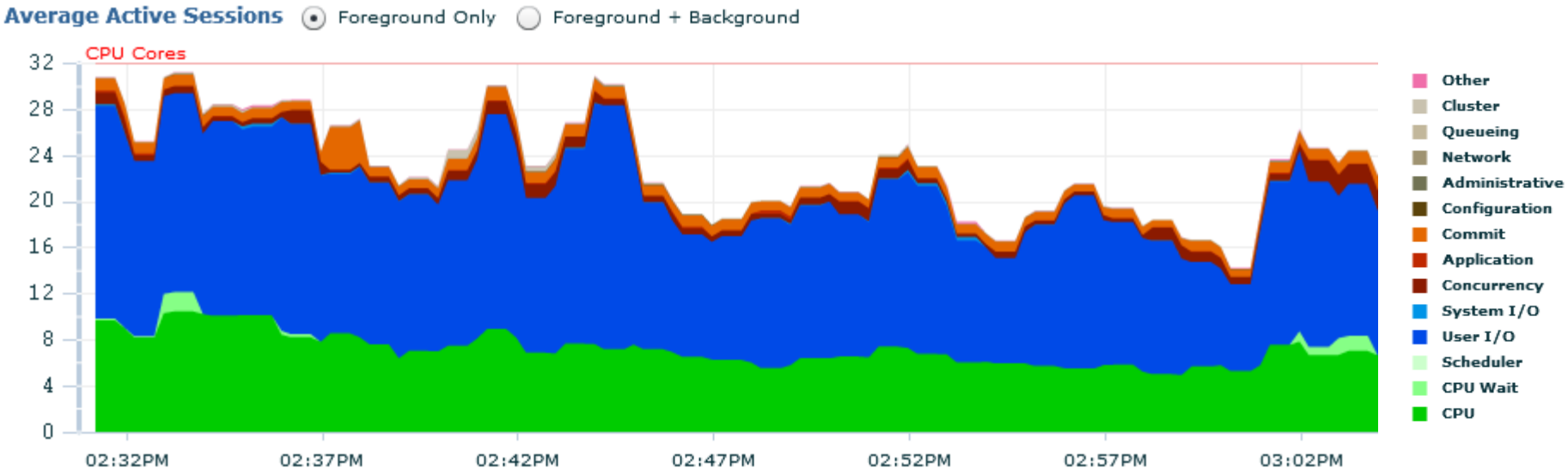
Performance Analysis

- Finding the root causes can be hard:
  - Applications are slow because of storage?
  - Or Storage is slow because overloaded by a runaway application?

- We want to build a model of what is happening
  - That can be proven with data
  - That we can use for tuning

Transactional Workload

- We will focus on systems like this example (OLTP):



  - DB time dominated by ‘db file sequential read’
    - CPU usage is not a bottleneck
    - We can ignore locks and other serialization events

Wait Event Drill-Down

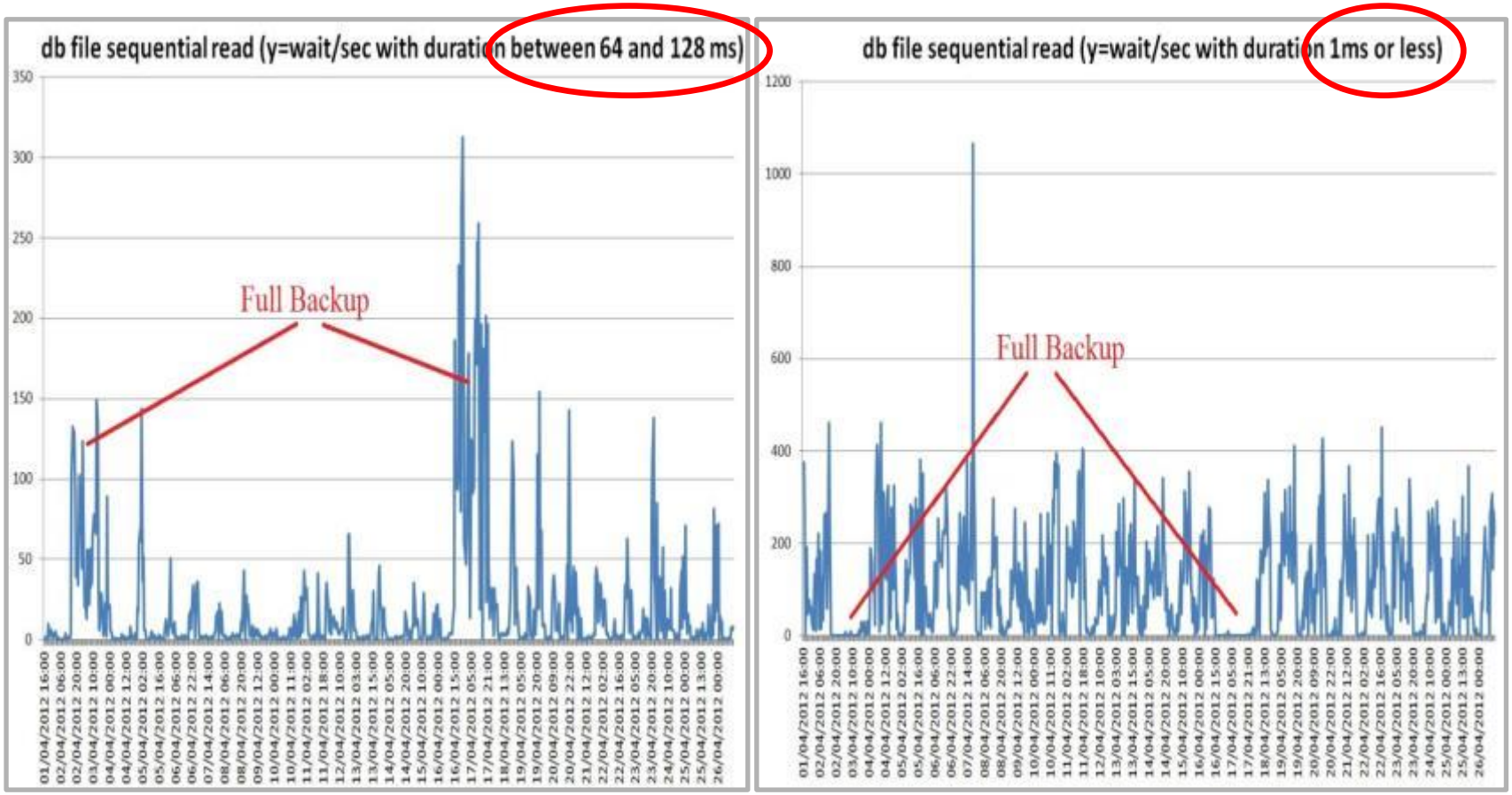
- What is available?
  - Drill down on ‘db file sequential read’ wait event data
  - Using event histogram data

- What can we gain?
  - Make educated guesses of what is happening on the storage
  - Attack the root causes when investigating performance problem

A Story From Production

- Migrated to a new storage system
  - NAS storage with SSD cache
  - Good performance: because of low-latency reads from SSD
- Issue:
  - From time to time production shows unacceptable performance
- Analysis:
  - The issue appears when the backup runs!

Wait Event Drill Down



Very slow reads appear

Reads from SSD cache go to zero

Our Analysis

- AWR data used to examine the issue
  - DBA_HIST_EVENT_HISTOGRAM
  - Wait event: db file sequential read
- I/O slow during backups
  - because fewer I/O requests were served from SSD
- Note on how we worked around this issue
  - Short term: moved backup to Active Data Guard
    Medium term: upgraded filer model

Real-Time Monitoring

- Problem:
  - Hourly AWR average is often too coarse
  - How to perform real-time monitoring of the event latency?

- Answer: data from GV$EVENT_HISTOGRAM
  - CLI script to compute deltas from cumulative counters

Monitoring Latency - Snapshots

- Custom script: ehm.sql

```
primary:system@orclrac1> @ehm 60 db%sequential

waiting for 60 sec (delta measurement interval = 60 sec)

Wait (ms)   N#          Event                     Last update time
----------- ----------- ------------------------- -----------------------------------
1            12588      db file sequential read 20-NOV-13 04.52.02.549024 PM +02:00
2            638        db file sequential read 20-NOV-13 04.52.02.323209 PM +02:00
4            241        db file sequential read 20-NOV-13 04.52.00.017278 PM +02:00
8            1032       db file sequential read 20-NOV-13 04.52.02.407010 PM +02:00
16           6128       db file sequential read 20-NOV-13 04.52.02.520877 PM +02:00
32           3865       db file sequential read 20-NOV-13 04.52.02.526403 PM +02:00
64           622        db file sequential read 20-NOV-13 04.52.02.475484 PM +02:00
128          48         db file sequential read 20-NOV-13 04.52.02.454875 PM +02:00
256          2          db file sequential read 20-NOV-13 04.51.35.738163 PM +02:00
512          1          db file sequential read 20-NOV-13 04.51.54.617231 PM +02:00
1024         13         db file sequential read 20-NOV-13 04.52.01.560293 PM +02:00
2048         0          db file sequential read 20-NOV-13 03.19.40.350234 PM +02:00
4096         0          db file sequential read 15-NOV-13 02.25.22.371191 AM +02:00
8192         0          db file sequential read 31-OCT-13 01.01.10.757675 AM +02:00
16384        0          db file sequential read 28-OCT-13 11.51.50.122887 PM +02:00
32768        0          db file sequential read 11-OCT-13 12.42.21.599088 PM +02:00
65536        0          db file sequential read 11-OCT-13 12.42.21.601458 PM +02:00
131072       0          db file sequential read 11-OCT-13 12.42.21.606092 PM +02:00

Avg_wait(ms) N#          Tot_wait(ms) Event
------------ ---------- ------------ ------------------
8.5          25177      214095.1     db file sequential read
```

Script can be downloaded from: http://cern.ch/resources.htm

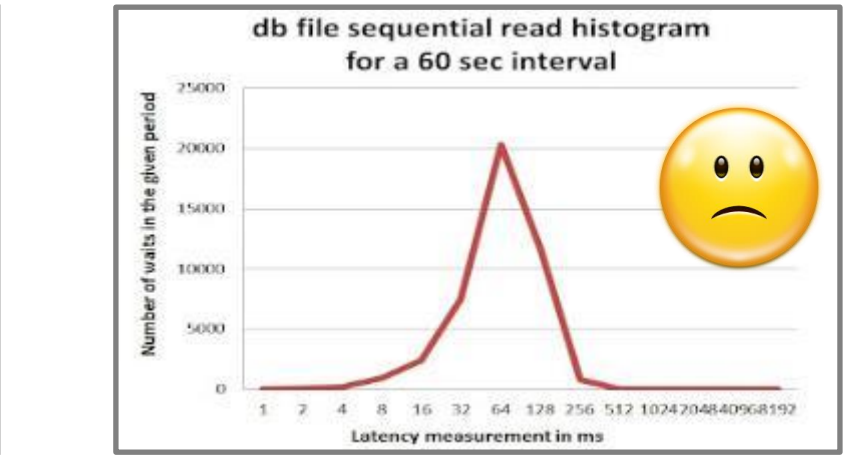
Monitoring Latency - Snapshots

```
primary:system@orclraci> @ehm 60 db%sequential

waiting for 60 sec (delta measurement interval = 60 sec)

Wait (ms)    N#          Event
-----------  ----------  ------------------------
1            15958       db file sequential read
2            1317        db file sequential read
4            1355        db file sequential read
8            2590        db file sequential read
16           9845        db file sequential read
32           8339        db file sequential read
64           1607        db file sequential read
128          124         db file sequential read
256          7           db file sequential read
512          1           db file sequential read
1024         15          db file sequential read
2048         0           db file sequential read
4096         0           db file sequential read
8192         0           db file sequential read

Avg_wait(ms) N#          Tot_wait(ms) Event
-----------  ----------  ------------ --------------------
10.3         41159       423786       db file sequential read
```




```
primary:system@orclraci> @ehm 60 db%sequential

waiting for 60 sec (delta measurement interval = 60 sec)

Wait (ms)    N#          Event
-----------  ----------  ------------------------
1            16          db file sequential read
2            61          db file sequential read
4            230         db file sequential read
8            930         db file sequential read
16           2427        db file sequential read
32           7488        db file sequential read
64           20352       db file sequential read
128          11616       db file sequential read
256          814         db file sequential read
512          20          db file sequential read
1024         22          db file sequential read
2048         0           db file sequential read
4096         0           db file sequential read
8192         0           db file sequential read
16384        0           db file sequential read
32768        0           db file sequential read
65536        0           db file sequential read
131072       0           db file sequential read

Avg_wait(ms) N#          Tot_wait(ms) Event
-----------  ----------  ------------ --------------------
53.4         43992       2350745.9    db file sequential read
```

Checkpoint #1

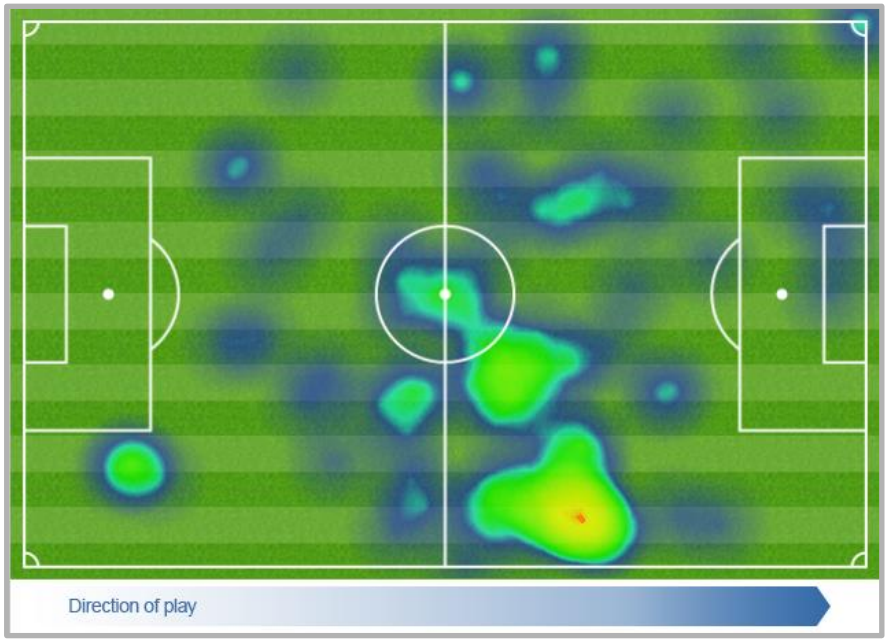
- Average latency can hide details
  - Multi-modal distributions in modern storage (think HDD arrays with large SSD cache)
- How-to drill down on the latency dimension:
  - Use Latency histograms
- Take care of the time dimension too:
  - Collect data over short time intervals

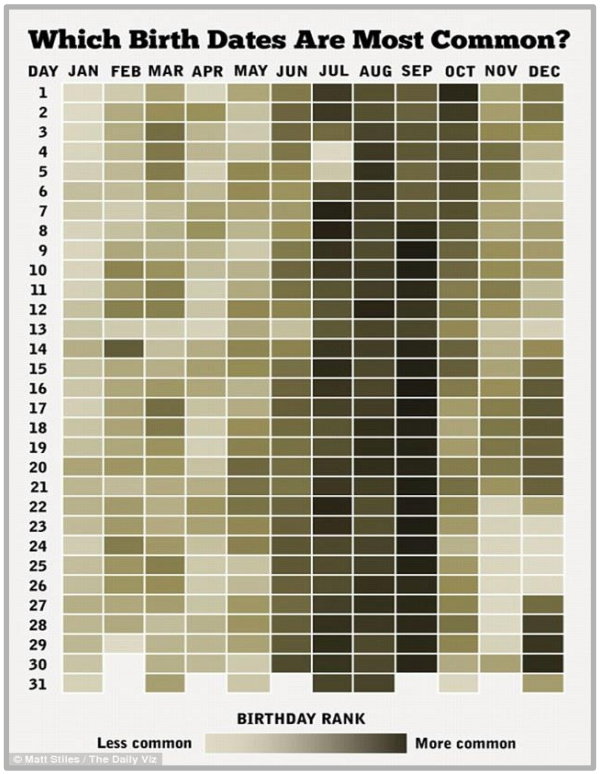
Display Latency Data over Time

- It's a 3D visualization problem:
  - (1) Latency bucket, (2) time, (3) value
- Heat Map representation
  - Used for example in Sun ZFS Storage 7000 Analytics
  - Reference: Brendan Gregg, Visualizing system latency, Communications of the ACM, July 2010

Heat Maps

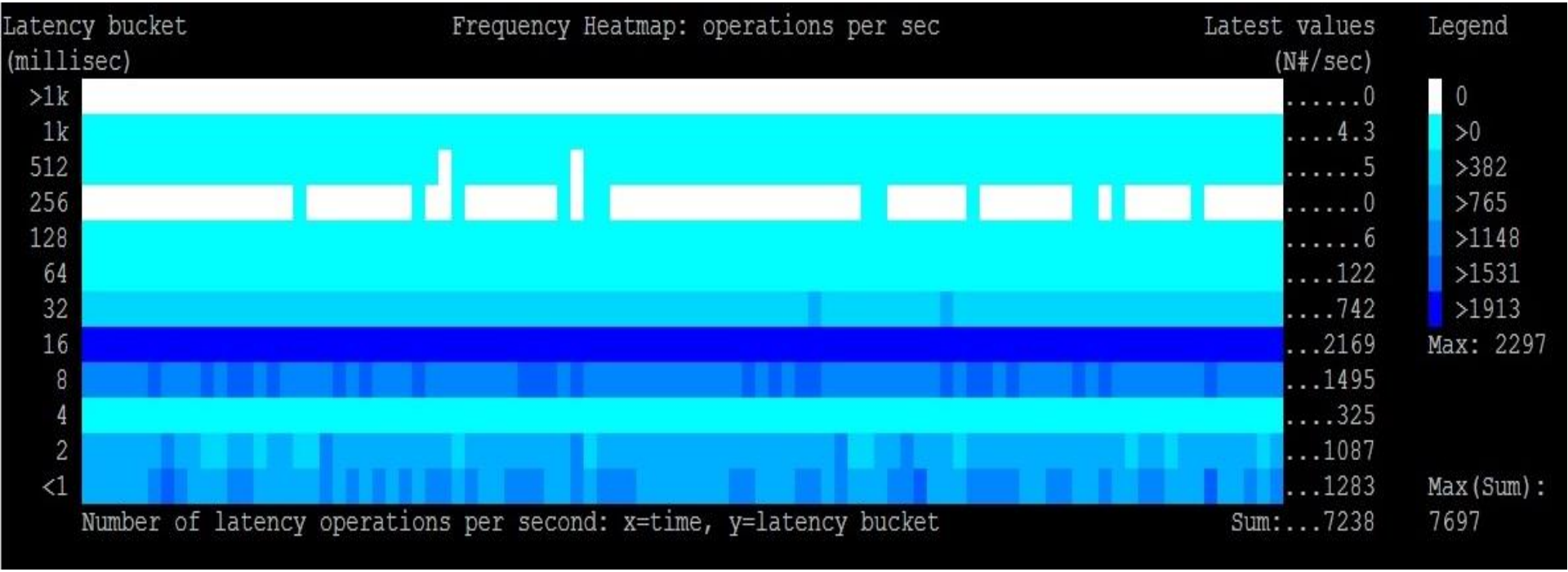
- Definition:
  - *graphical representation of data where the individual values contained in a matrix are represented as colors (wikipedia)*
- Examples:





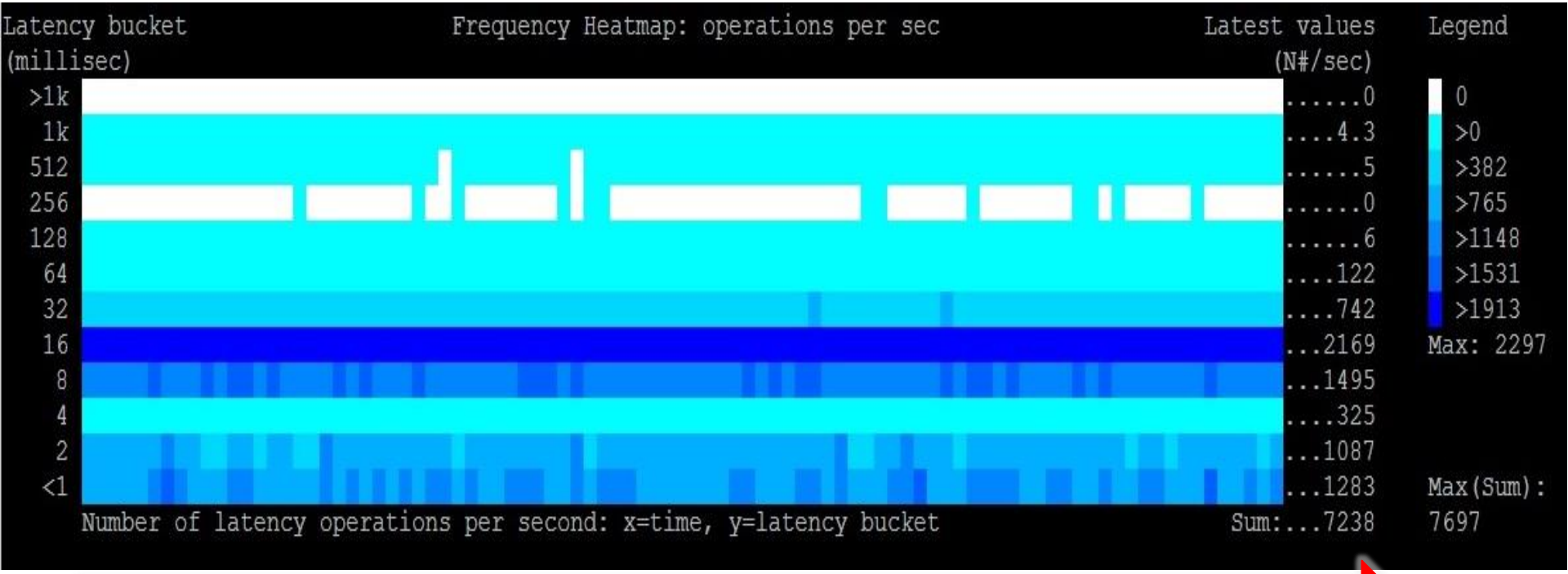
Latency Heat Maps - Frequency

- X=time, Y=latency bucket
- Color=events per second (e.g. IOPS)



Latency Heat Maps - Frequency

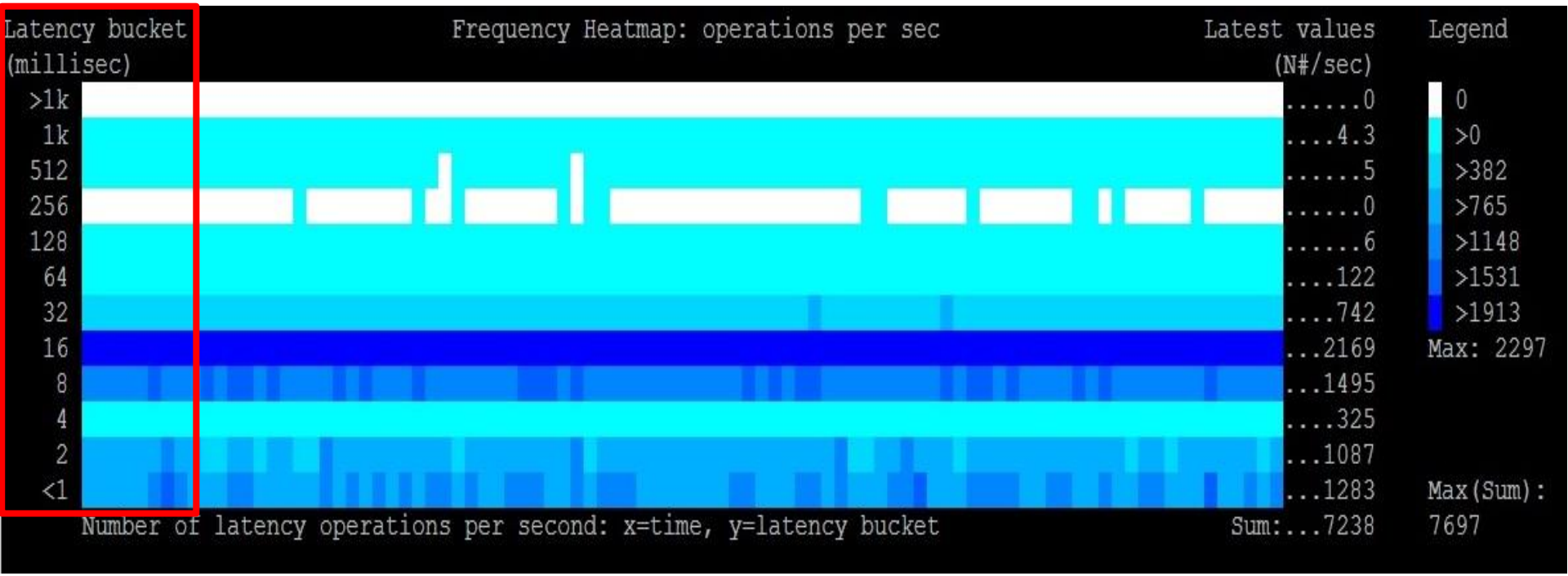
- X=time, Y=latency bucket
- Color=events per second (e.g. IOPS)



T I M E

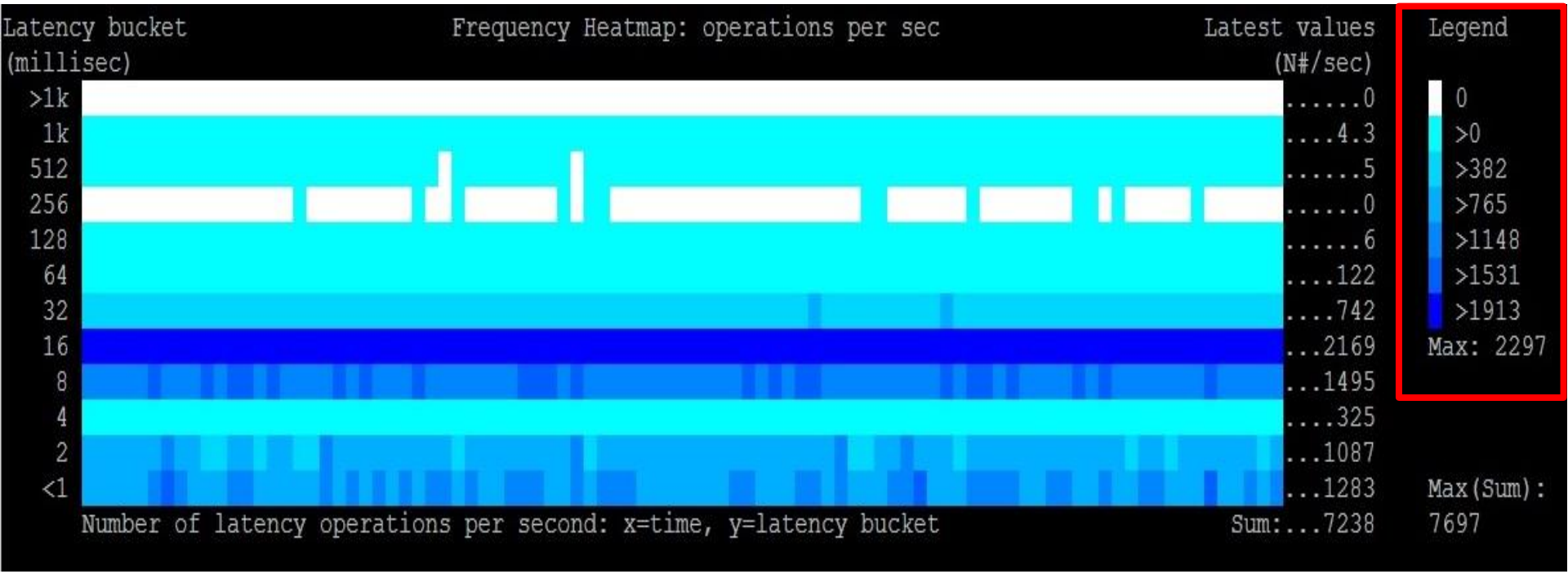
Latency Heat Maps - Frequency

- X=time, Y=latency bucket
- Color=events per second (e.g. IOPS)



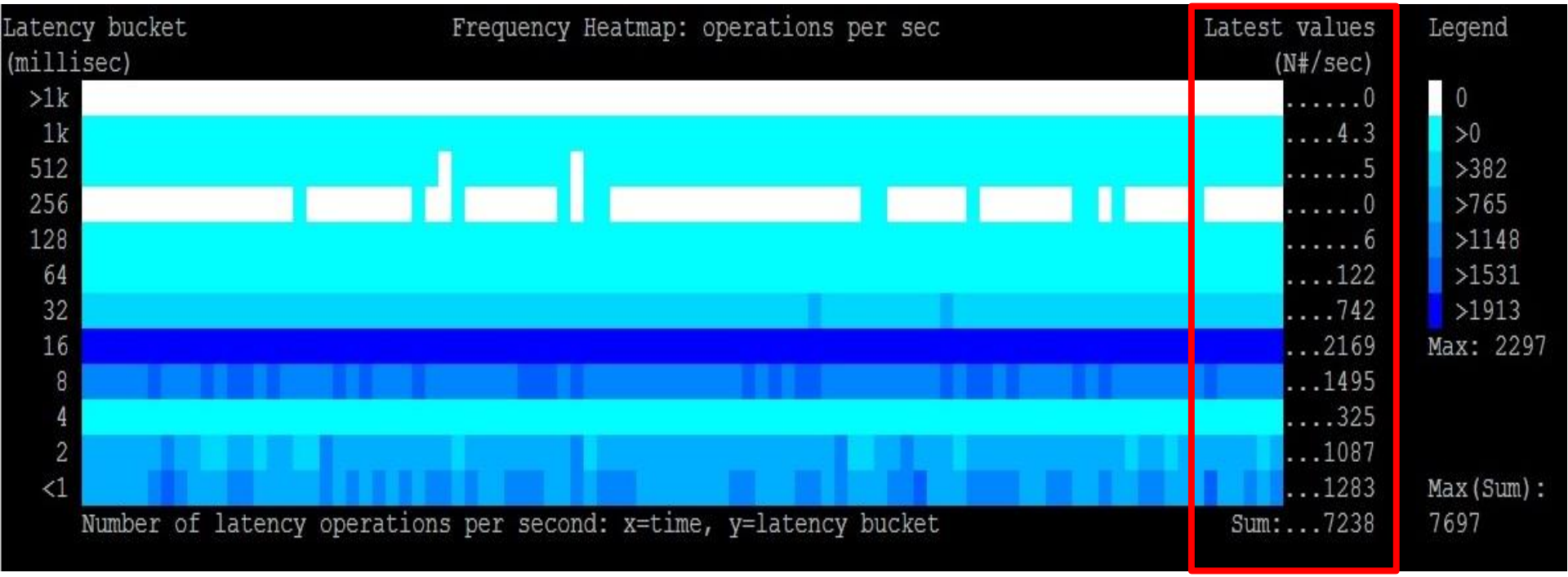
Latency Heat Maps - Frequency

- X=time, Y=latency bucket
- Color=events per second (e.g. IOPS)



Latency Heat Maps - Frequency

- X=time, Y=latency bucket
- Color=events per second (e.g. IOPS)



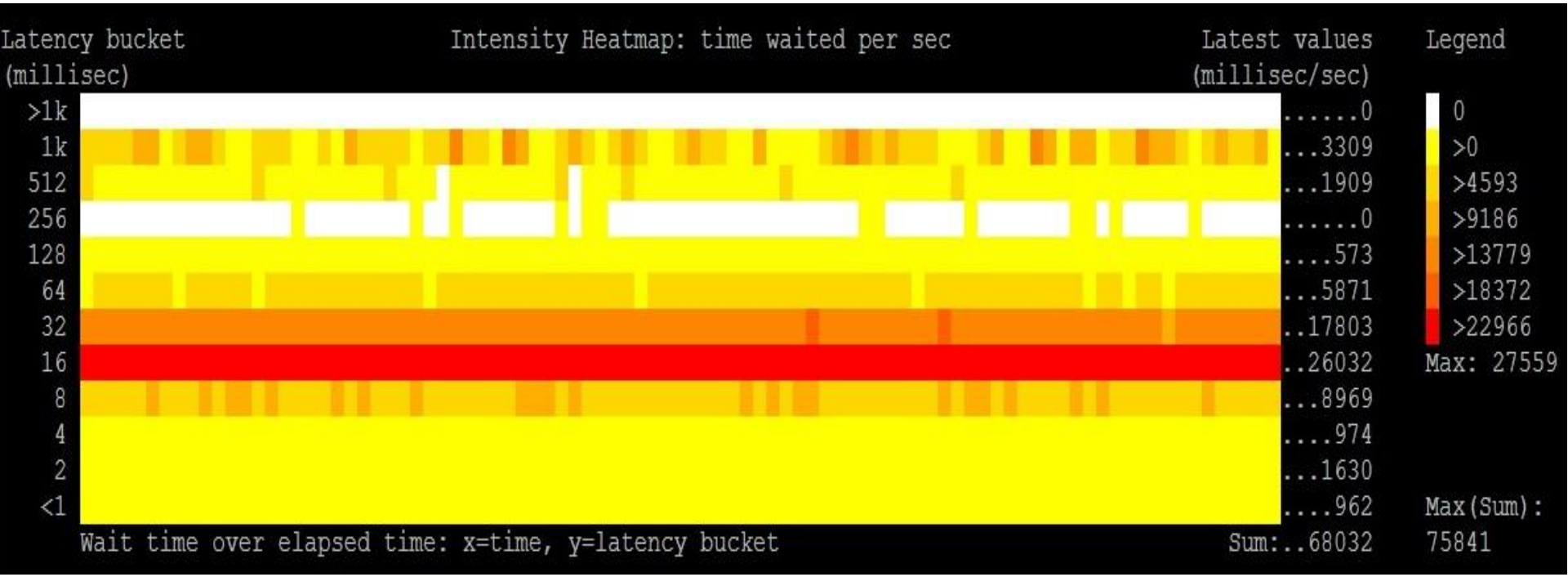
Time-Waited Histogram

- How much time do we wait in a given bucket?
  - Not directly available in gv$event_histogram
- How to estimate it? Example:
  - 100 waits in the bucket 8ms means
  - Wait time between 100*4 ms and 100*8 ms
  - Approximate: 100 * 6 ms [that is 100 * ¾ * 8 ms]

- Definition:
  - Intensity = 0.75 * bucket_value * frequency_count

Latency Heat Maps - Intensity

- X=time, Y=latency bucket
- Color= intesity [time waited per bucket]



Checkpoint #2

- Heat Maps
  - A representation of latency histograms over time
- Frequency Heat Map
  - IOPS details per latency bucket and time
- Intensity Heat Map
  - Time waited details
  - Visual representation of the weight of each bucket

Outline

- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

Stress Testing

- Use case:
  - Investigate HW performance
  - Compare different systems
  - Example: compare current storage with a new system

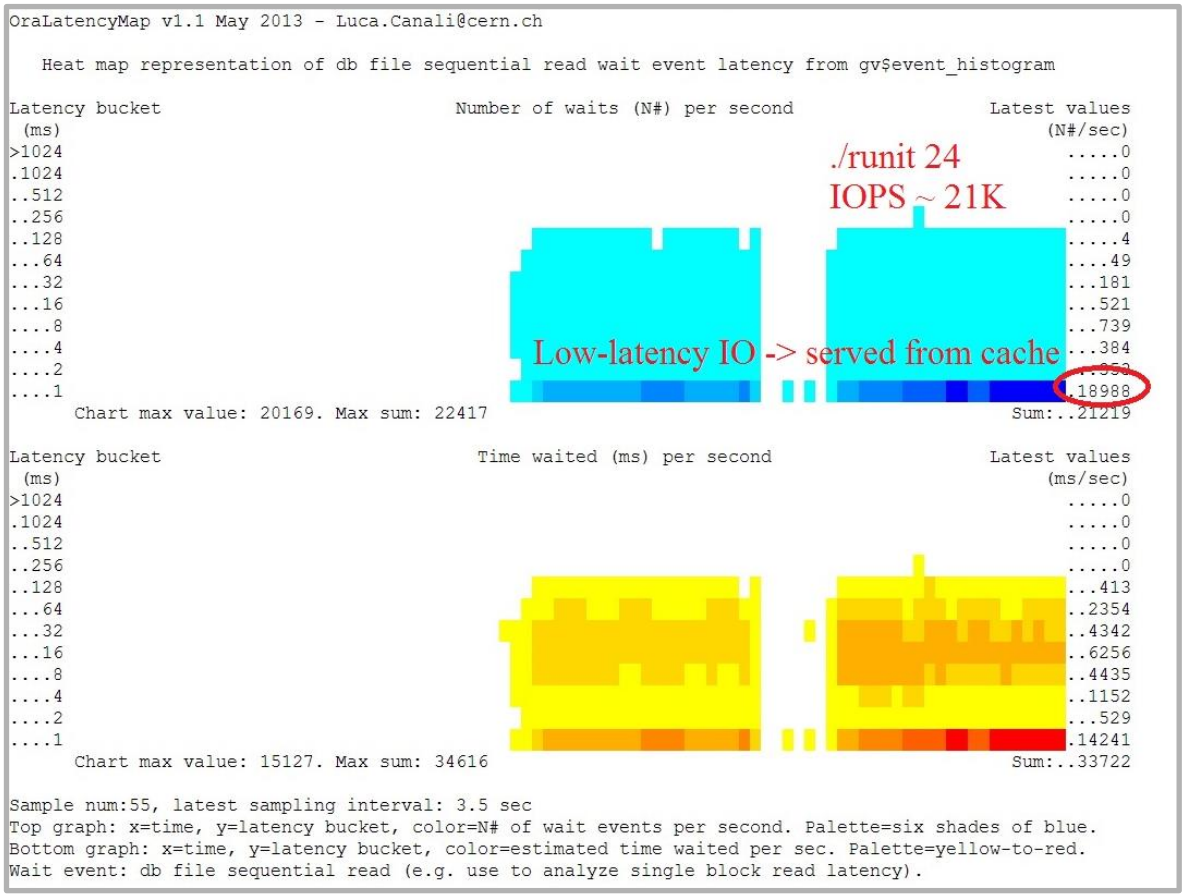
- It can be hard:
  - Choosing test workloads that are representative of production usage
  - Understanding the effects of caching

SLOB 2

- An Oracle-based stress testing tool
  - Search: “SLOB 2 by Kevin Closson”
- Great tool to generate lots of random IO
  - I/O directly from Oracle processes
  - Physical reads
    - Visible as Oracle’s wait events db file sequential read
- Size of test data is configurable
- Concurrency is configurable

Example: “Too Good To Be True”

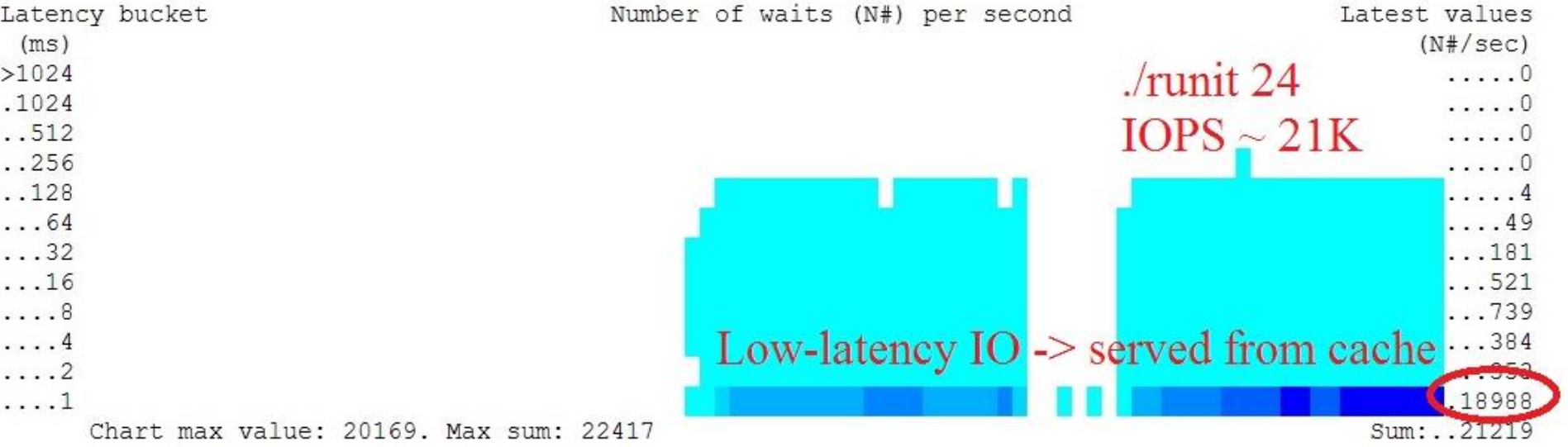
- 23 SAS disks delivering 20K IOPS?
- It doesn’t make sense
- Latency is the clue
- Reads served by controller cache!



- Lesson learned: test data size was too small

OraLatencyMap v1.1 May 2013 - Luca.Canali@cern.ch

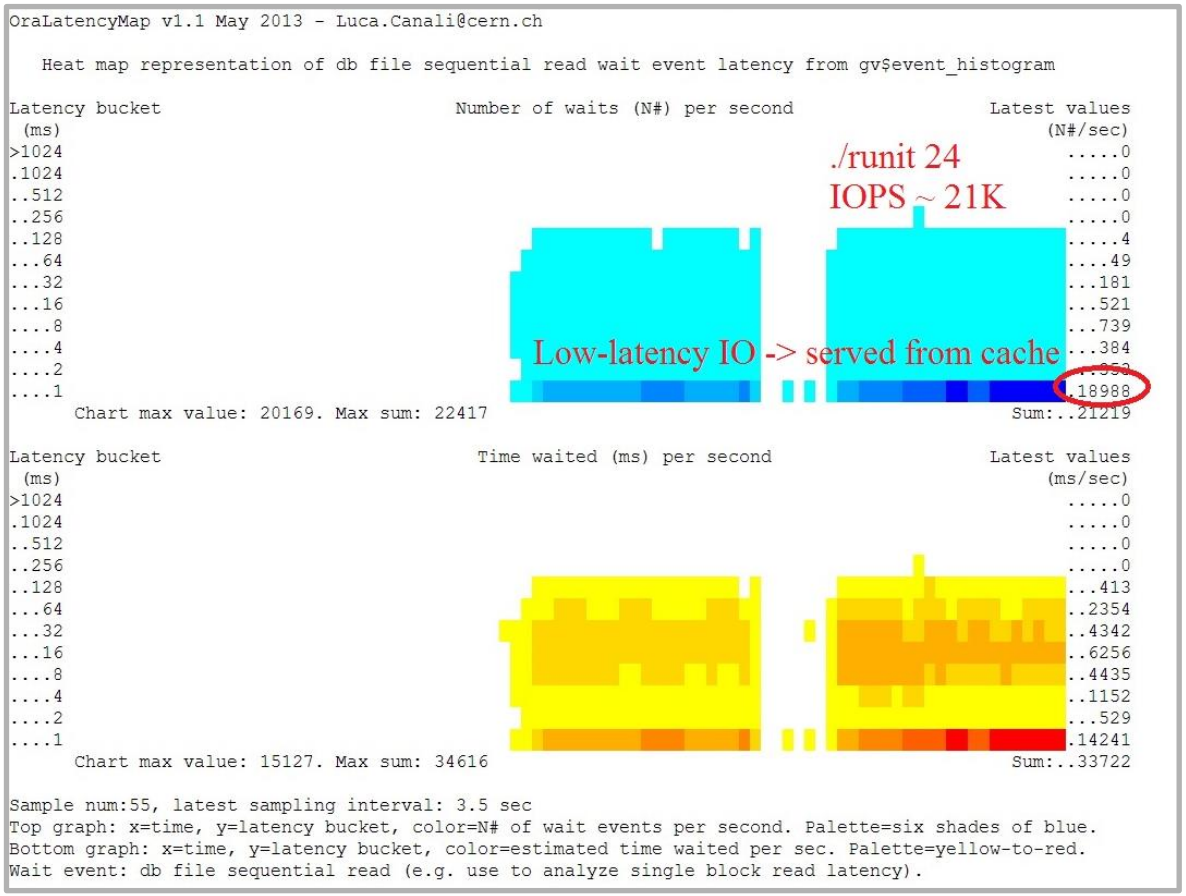
Heat map representation of db file sequential read wait event latency from gv$event_histogram



Sample num:55, latest sampling interval: 3.5 sec
Top graph: x=time, y=latency bucket, color=N# of wait events per second. Palette=six shades of blue.
Bottom graph: x=time, y=latency bucket, color=estimated time waited per sec. Palette=yellow-to-red.
Wait event: db file sequential read (e.g. use to analyze single block read latency).

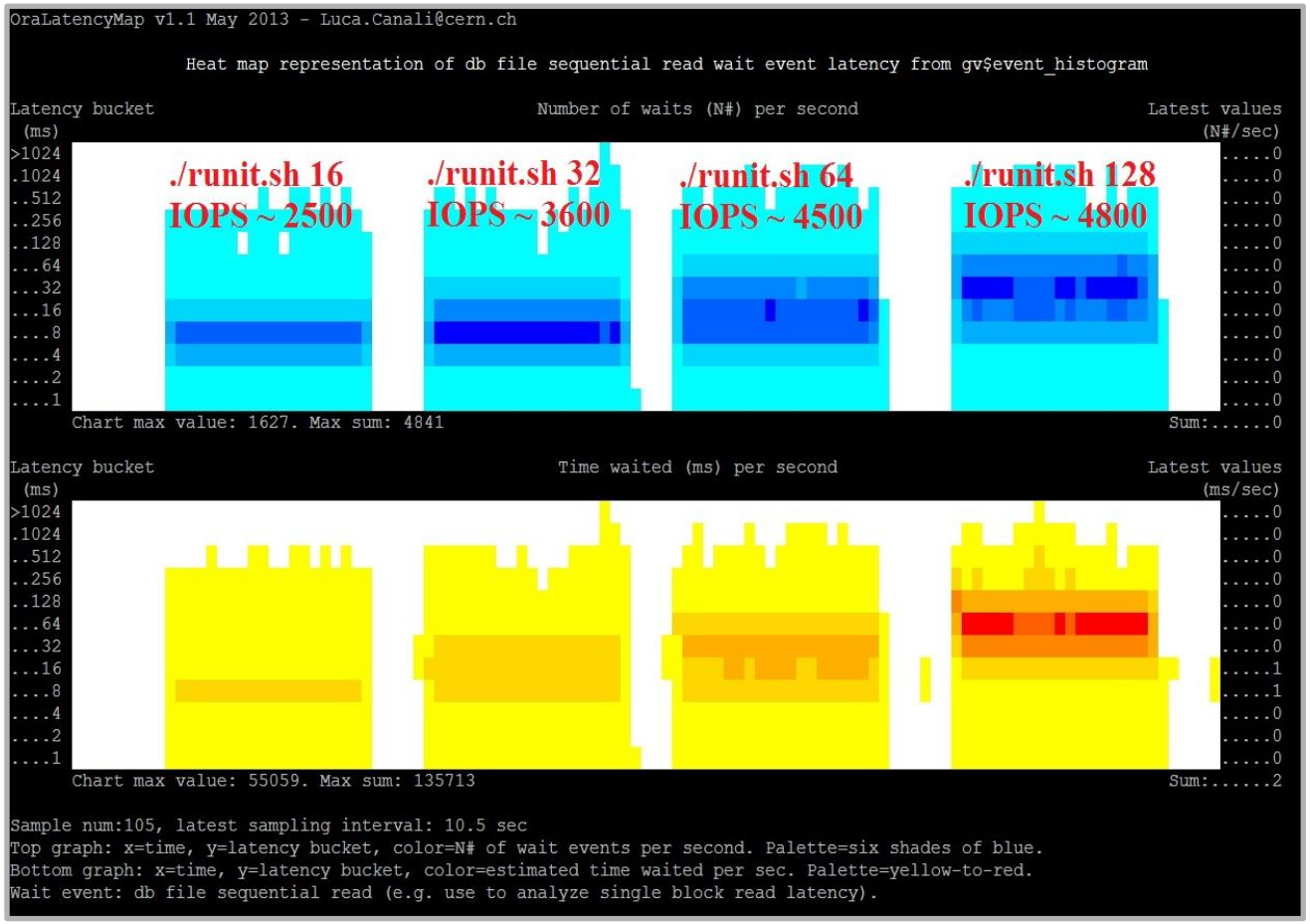
Example: “Too Good To Be True”

- 23 SAS disks delivering 21K IOPS?
- It doesn’t make sense
- Latency is the clue
- Reads served by controller cache!



- Lesson learned: test data size was too small

Example: “Load Till Saturation”



23 SAS disks

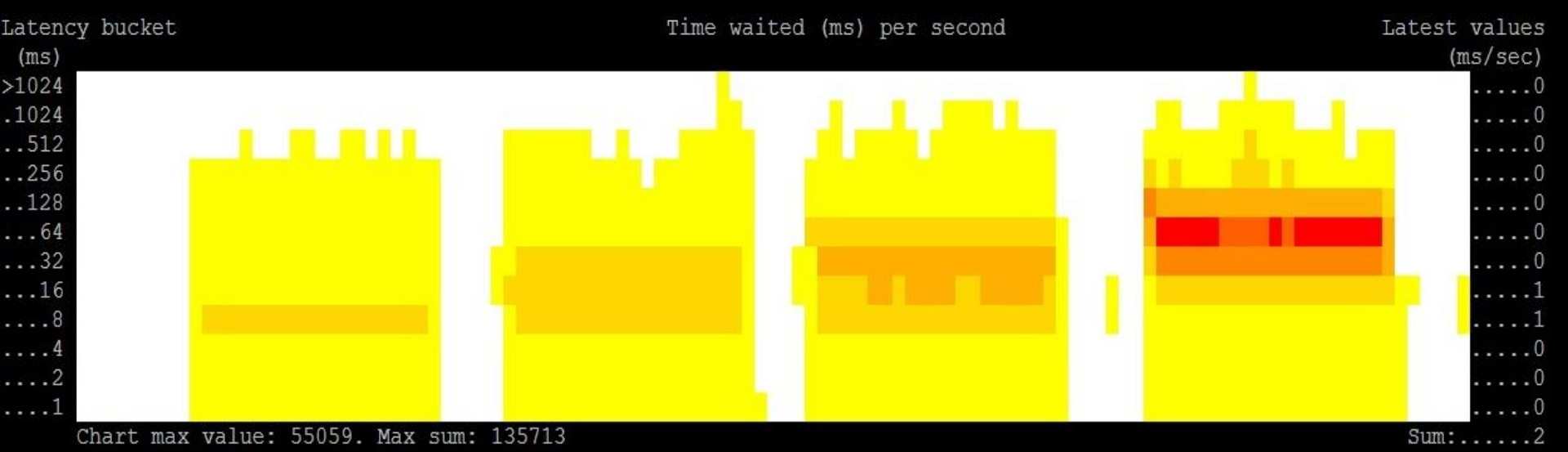
JBOD & ASM

4 consecutive tests with increasing load

Lesson learned: don’t accept IOPS numbers without latency values

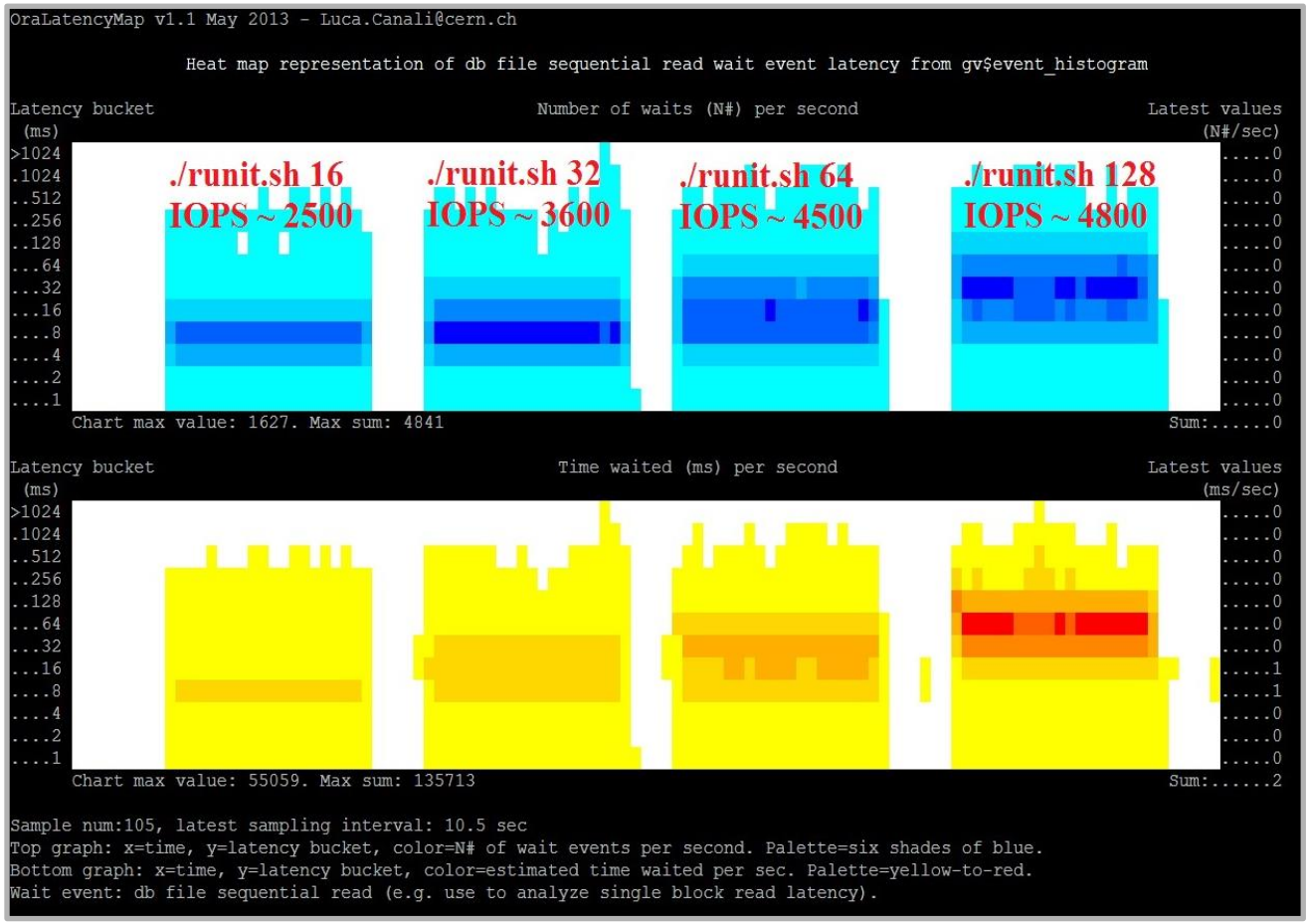
OraLatencyMap v1.1 May 2013 - Luca.Canali@cern.ch

Heat map representation of db file sequential read wait event latency from gv$event_histogram



Sample num:105, latest sampling interval: 10.5 sec
Top graph: x=time, y=latency bucket, color=N# of wait events per second. Palette=six shades of blue.
Bottom graph: x=time, y=latency bucket, color=estimated time waited per sec. Palette=yellow-to-red.
Wait event: db file sequential read (e.g. use to analyze single block read latency).

Example: “Load Till Saturation”



23 SAS disks

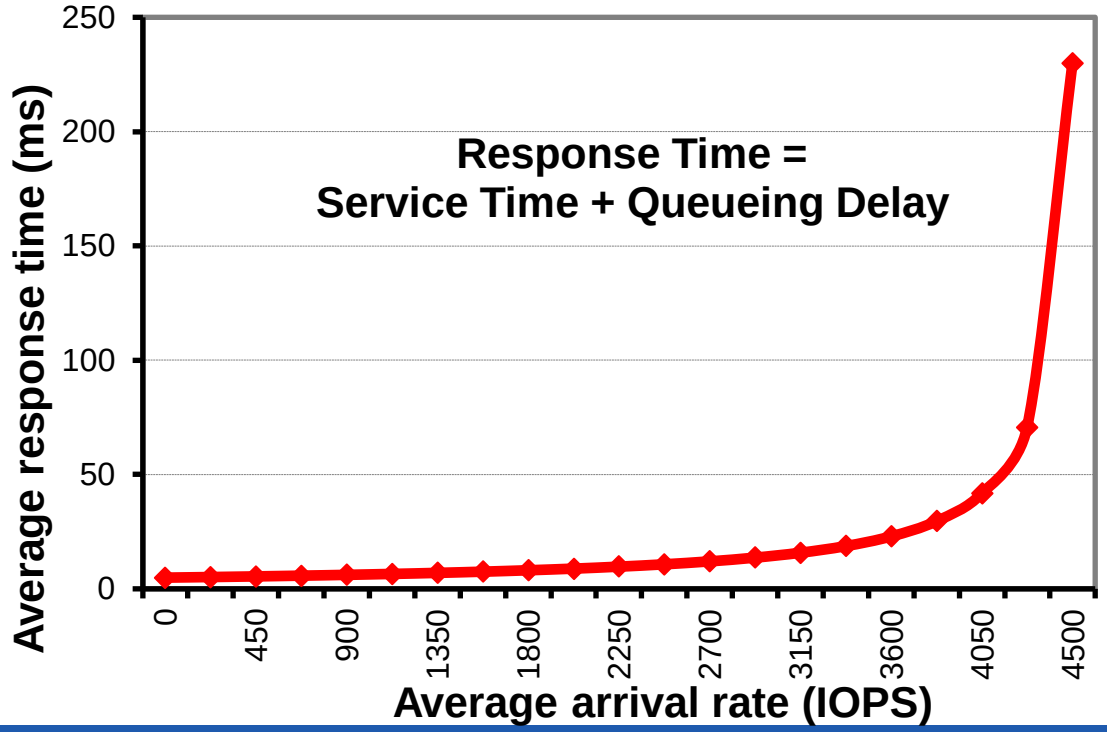
JBOD & ASM

4 consecutive tests with increasing load

Lesson learned: don’t accept IOPS numbers without latency values

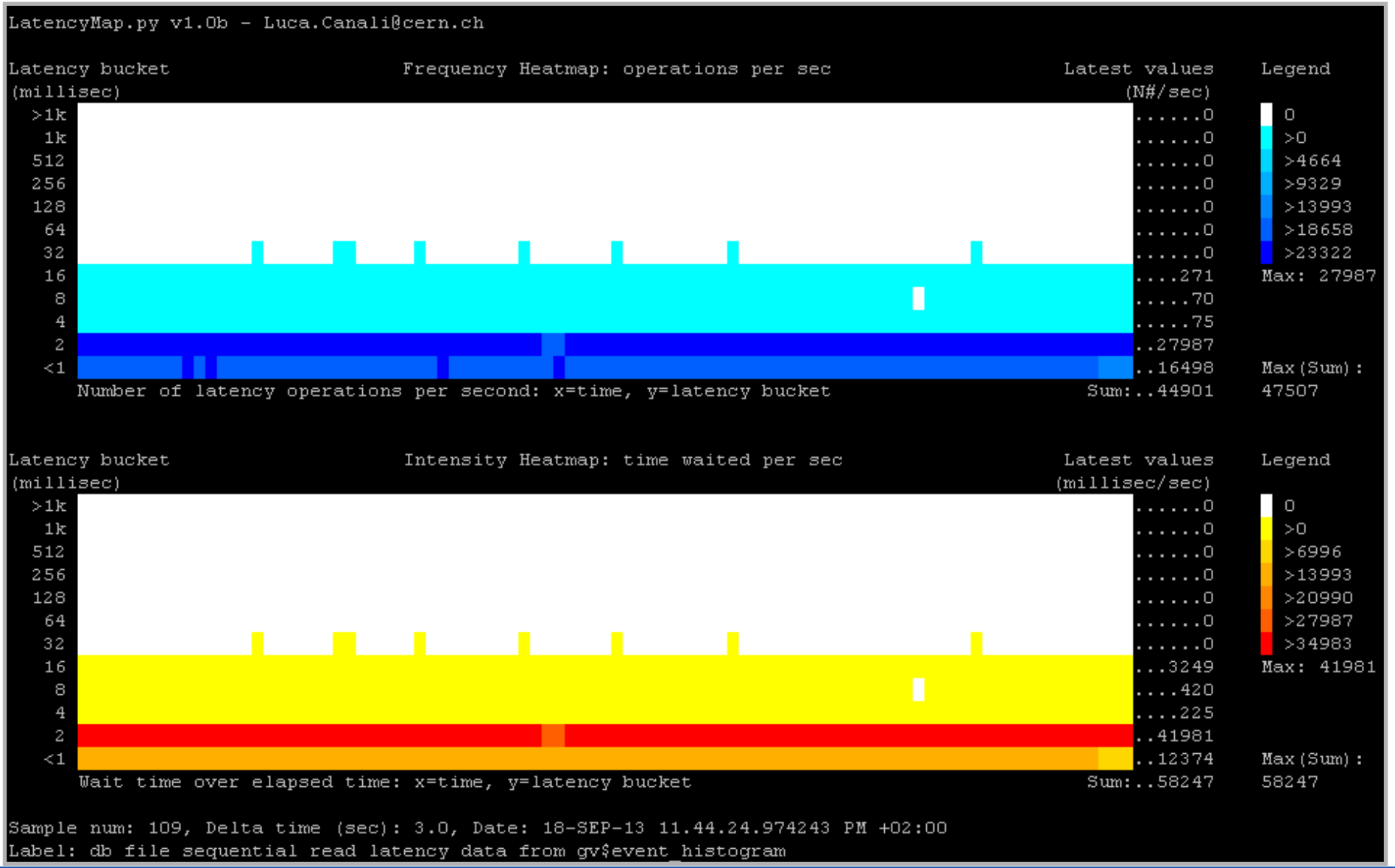
IOPS and Latency are Related

- Observation: as IOPS approach saturation latency increases very fast
- Data fits a simple model from queueing theory:

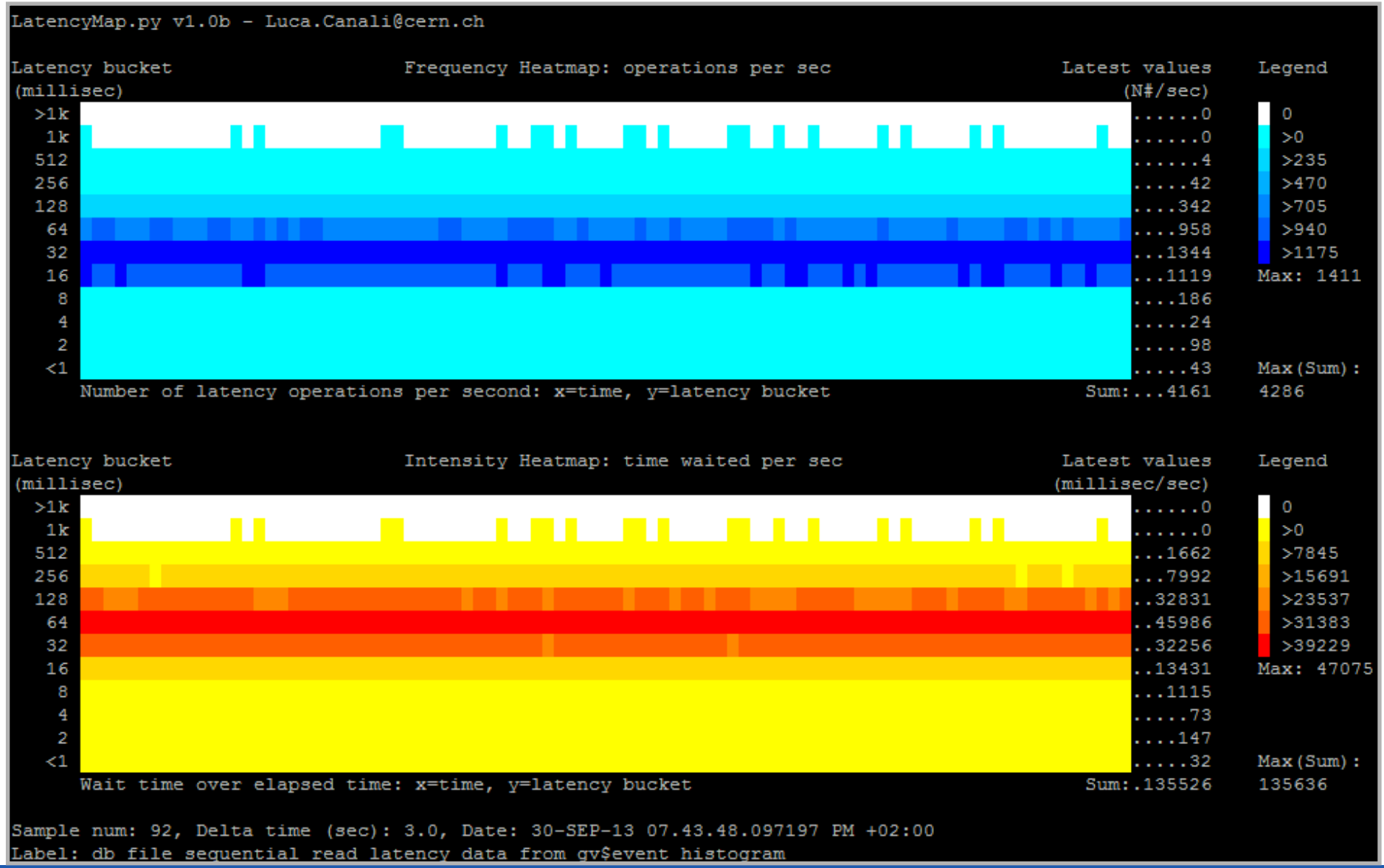


Calculation performed using
MMm Multiserver model
By Cary Millsap, 2003

0.5 TB dataset, 100% in SSD cache, 56 sessions, random reads - NAS system



10TB dataset, 128 sessions, random reads, onset of disk I/O saturation - NAS

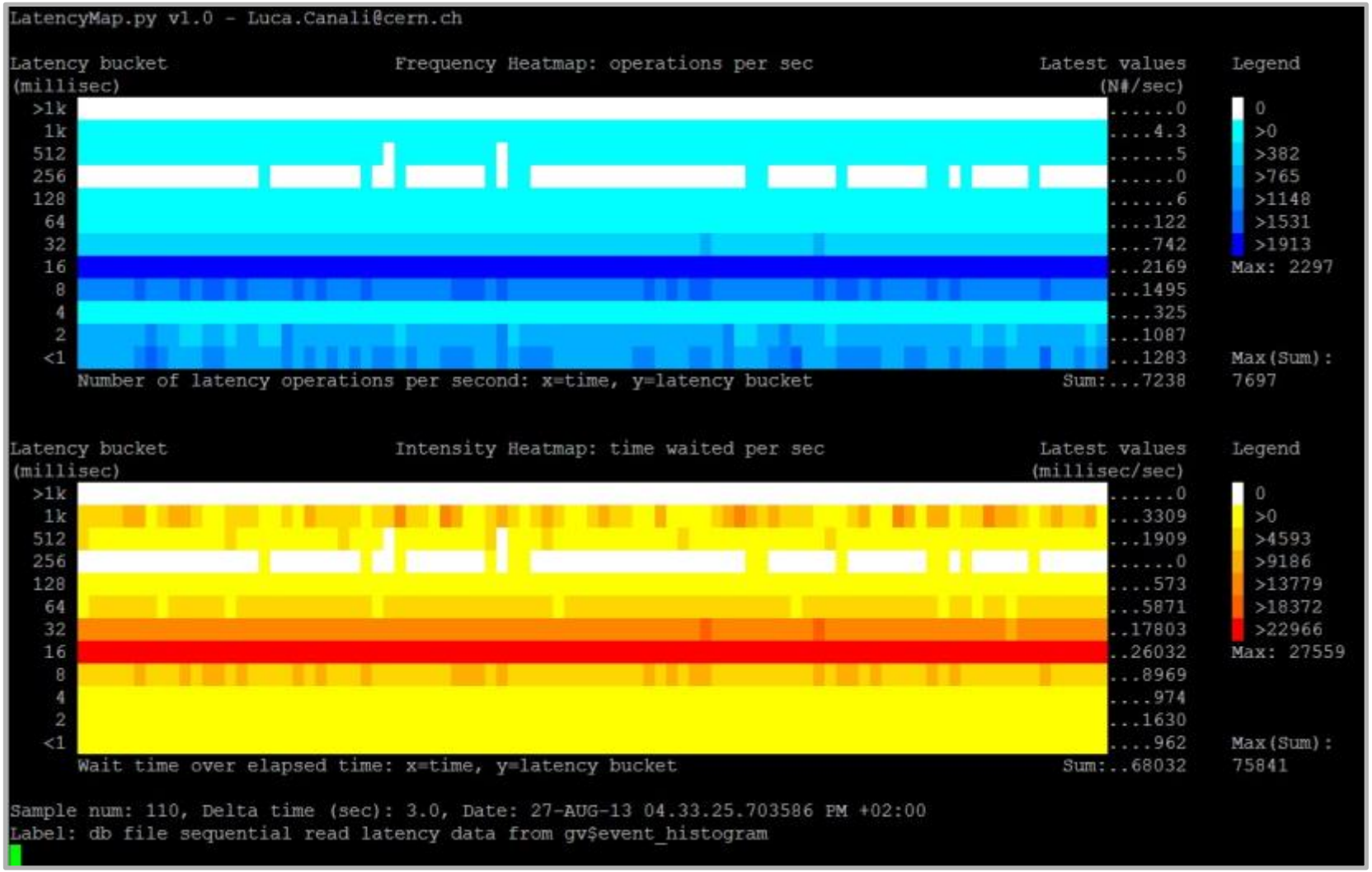


Monitoring Production Systems

- Understand I/O response time
  - Help for tuning and capacity planning
  - Attack questions like: is the storage slow?

- Drill down on three areas:
  - I/O served by SSD/controller cache
  - I/O served by physical disks ‘spindles’
  - I/O with very large latency: outliers

An Example of a Loaded Storage

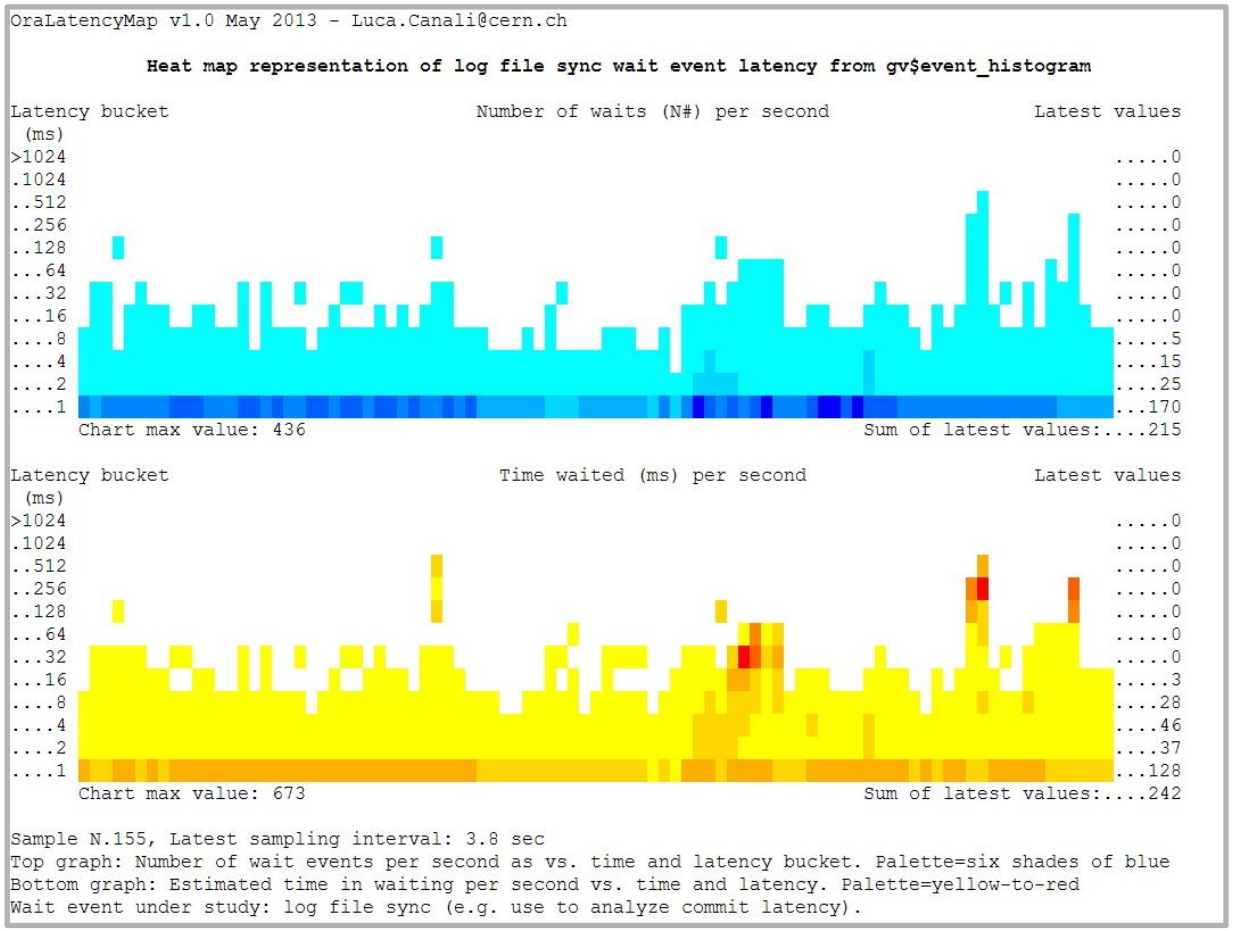


Example Analysis

- Are disks close to saturation?
  - NO, but latency high (SATA disks)
- I/O outliers?
  - YES, Further investigation on controller needed
- Do we have SSD/cache?
  - YES, ~30% reads with low latency
  - We could profit from a larger SSD cache maybe?

Heat Maps for Log File Sync

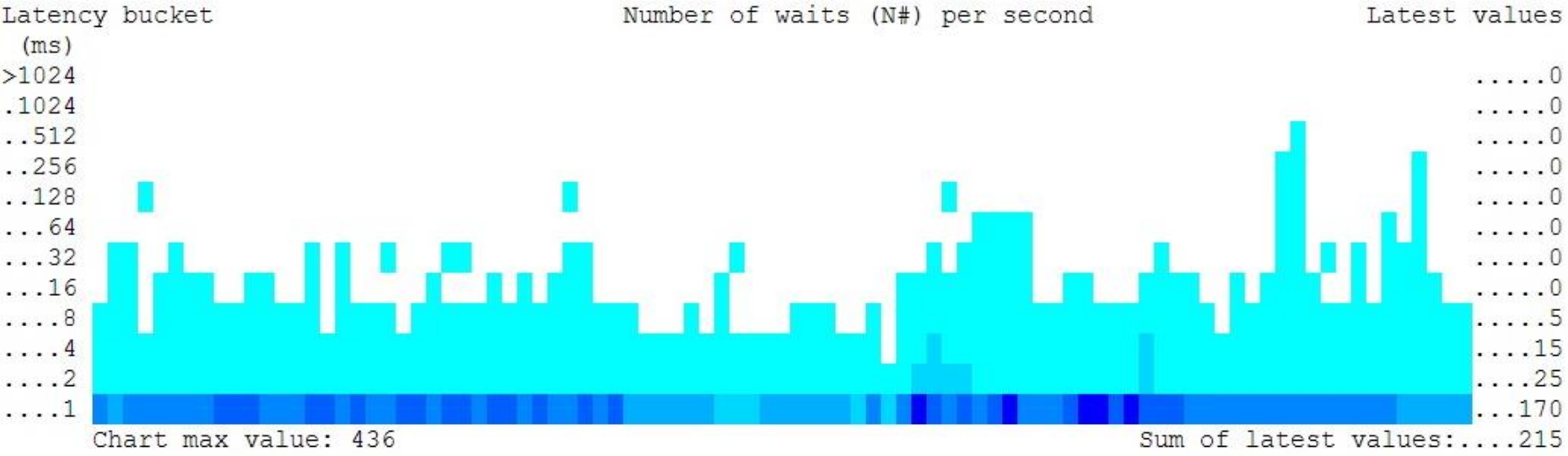
- Example from a production system



Low latency from writes because of storage cache

OraLatencyMap v1.0 May 2013 - Luca.Canali@cern.ch

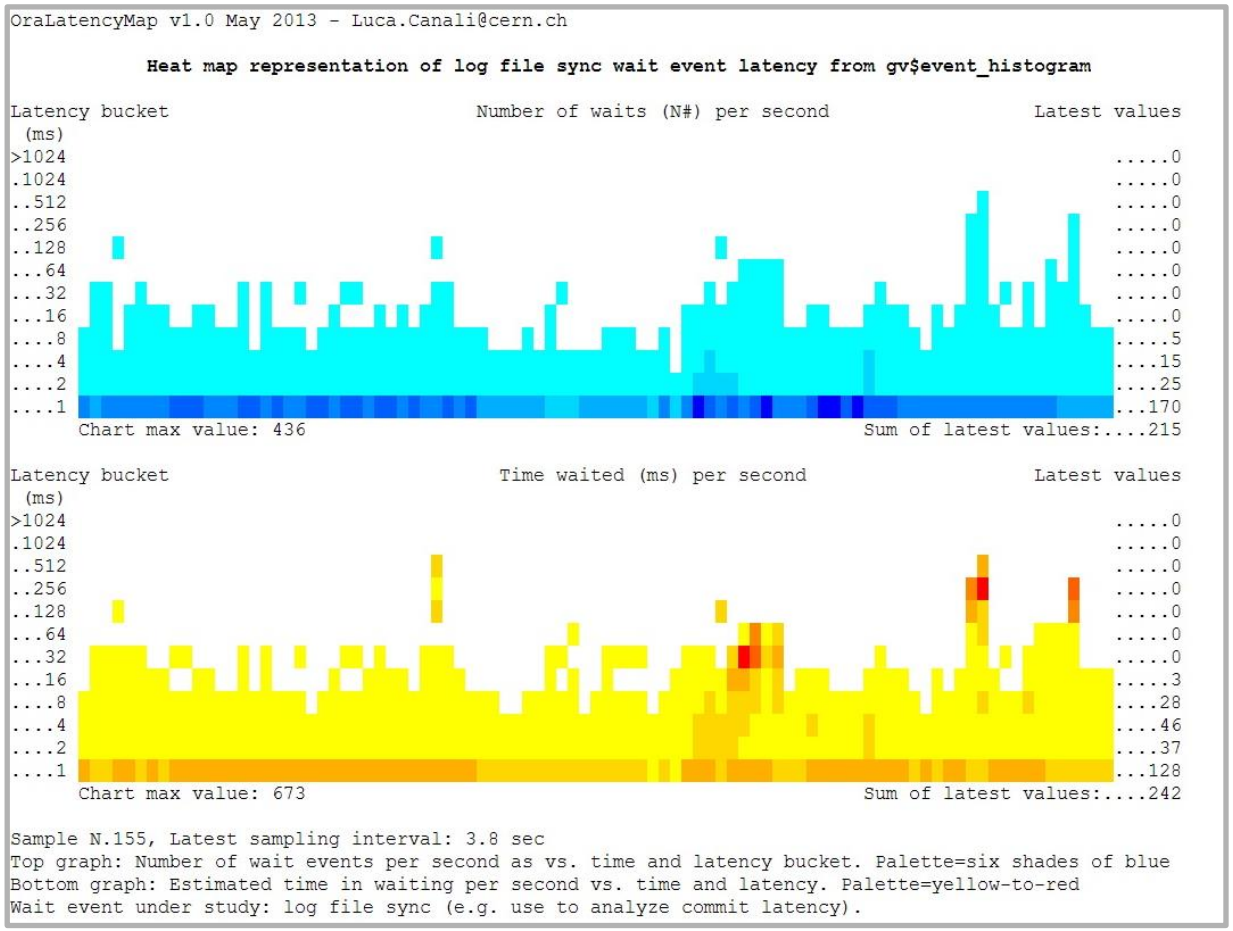
Heat map representation of log file sync wait event latency from gv$event_histogram



Sample N.155, Latest sampling interval: 3.8 sec
Top graph: Number of wait events per second as vs. time and latency bucket. Palette=six shades of blue
Bottom graph: Estimated time in waiting per second vs. time and latency. Palette=yellow-to-red
Wait event under study: log file sync (e.g. use to analyze commit latency).

Heat Maps for Log File Sync

- Example from a production system

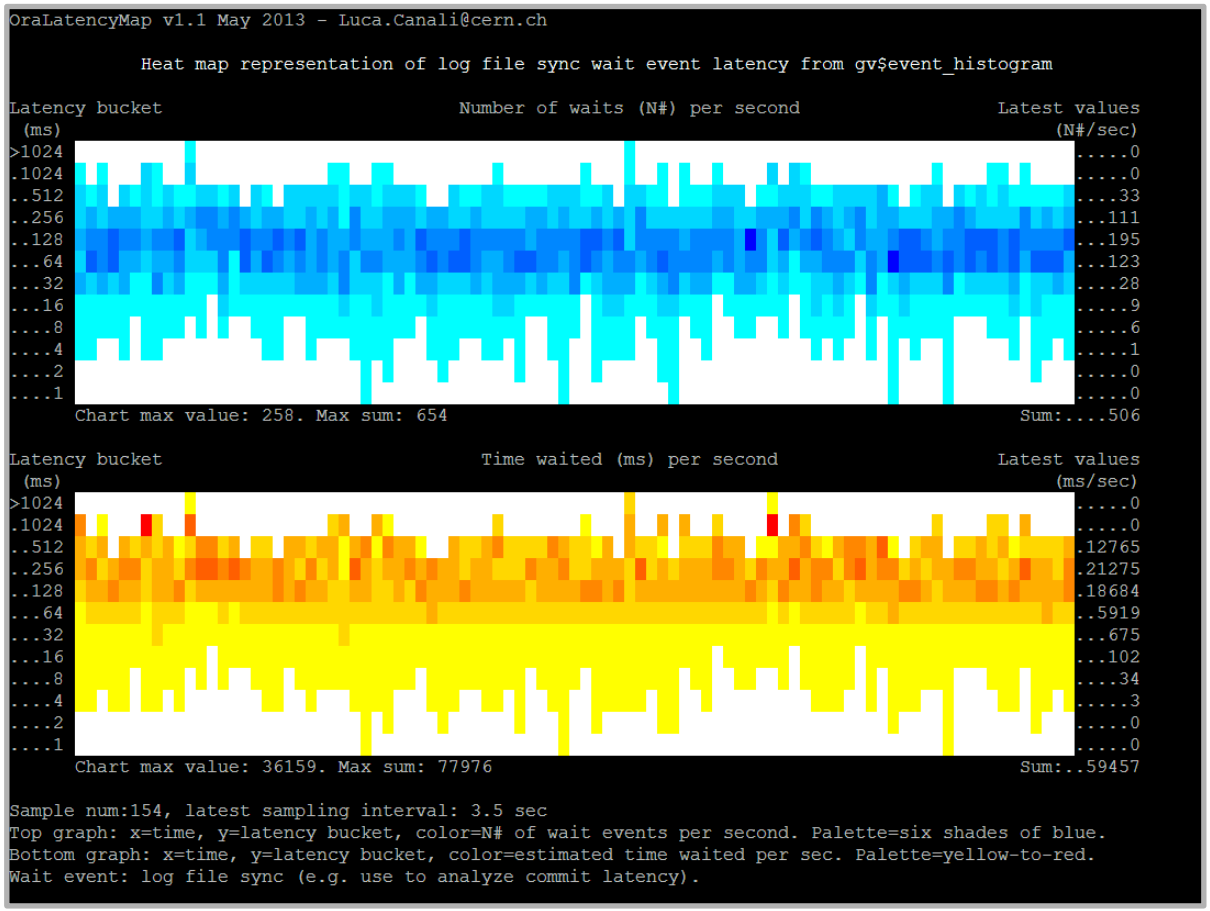


Low latency from writes because of storage cache

Log File Sync Troubleshooting

- Anomaly, on a test system

High latency caused by high load from Oracle and a few faulty HW components



Checkpoint #3

- IOPS data alone can be misleading because of saturation effects
  - Need IOPS with latency breakdown

- We can get insights on the storage performance
  - With latency histograms and heat map visualization
  - Analysis of random reads by drilling down ‘db file sequential read’
  - Use this method also for commit wait: ‘log file sync’

Limitations of Using Wait Events

- Wait event timing is done by the Oracle code
  - May not reflect actual I/O response time
  - CPU time may also be accounted as I/O wait
  - Server high load can distort the timing values
  - V$event_histogram lowest bucket is 1 millisecond

Random Reads

- Db file sequential read
  - Is the easiest event to relate to I/O service time
  - Instruments single-block reads, blocking I/O

- *Limitations:*
  - *In some cases Oracle can use async I/O for random reads, e.g. for prefetching and batching.*
  - *The wait event used in that case is 'db file parallel read'*

Log File Sync

- Log file sync
  - Big part of the commit-time wait
  - Complex: it's not a pure I/O event
- The root causes of high latency for log file sync can be
  - CPU starvation
  - LGWR behaviour
    - with inter-process communication and/or bugs
- See also Kevin Closson's blog

Limitations for Multi-Block I/O

- Asynchronous I/O
  - Wait events of this family are very hard or impossible to utilize precisely in the context of latency
  - See Frits Hoogland's work
- Multi-block I/O
  - Hard to compare latency for I/Os of different sizes
- Keeping in mind these limitations.
  - Heat maps of db file parallel write can be used to study DBWR performance issues (e.g. high write latency)

Outline

- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

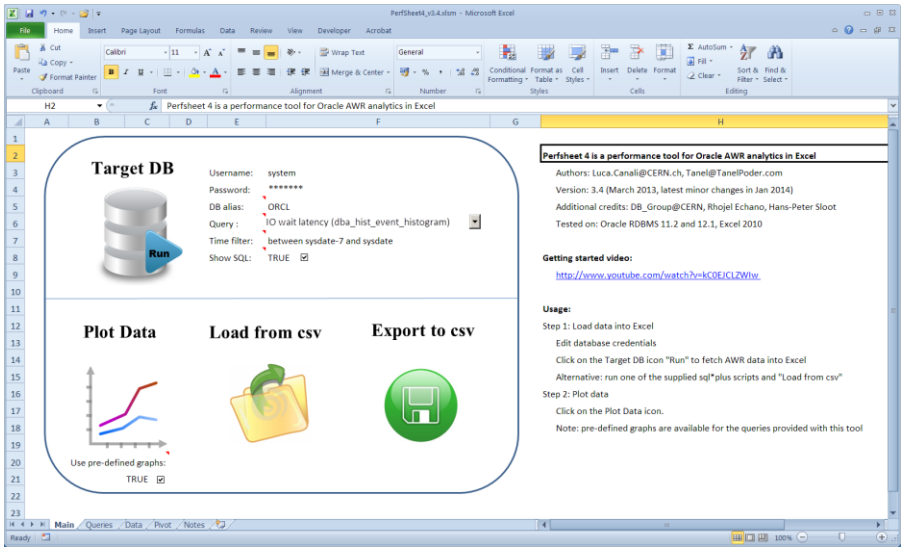
Tools

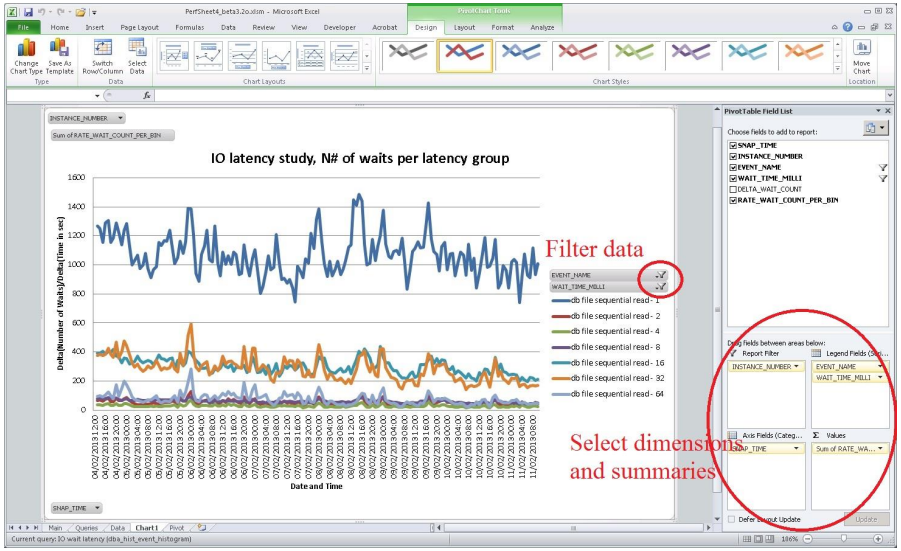
- Automate tedious tasks
  - Data collection
  - Visualisation

- Three tools I have built and shared:
  - PerfSheet4 -> AWR analytics
  - OraLatencyMap -> Heat Maps in SQL*Plus
  - PyLatencyMap -> Advanced version

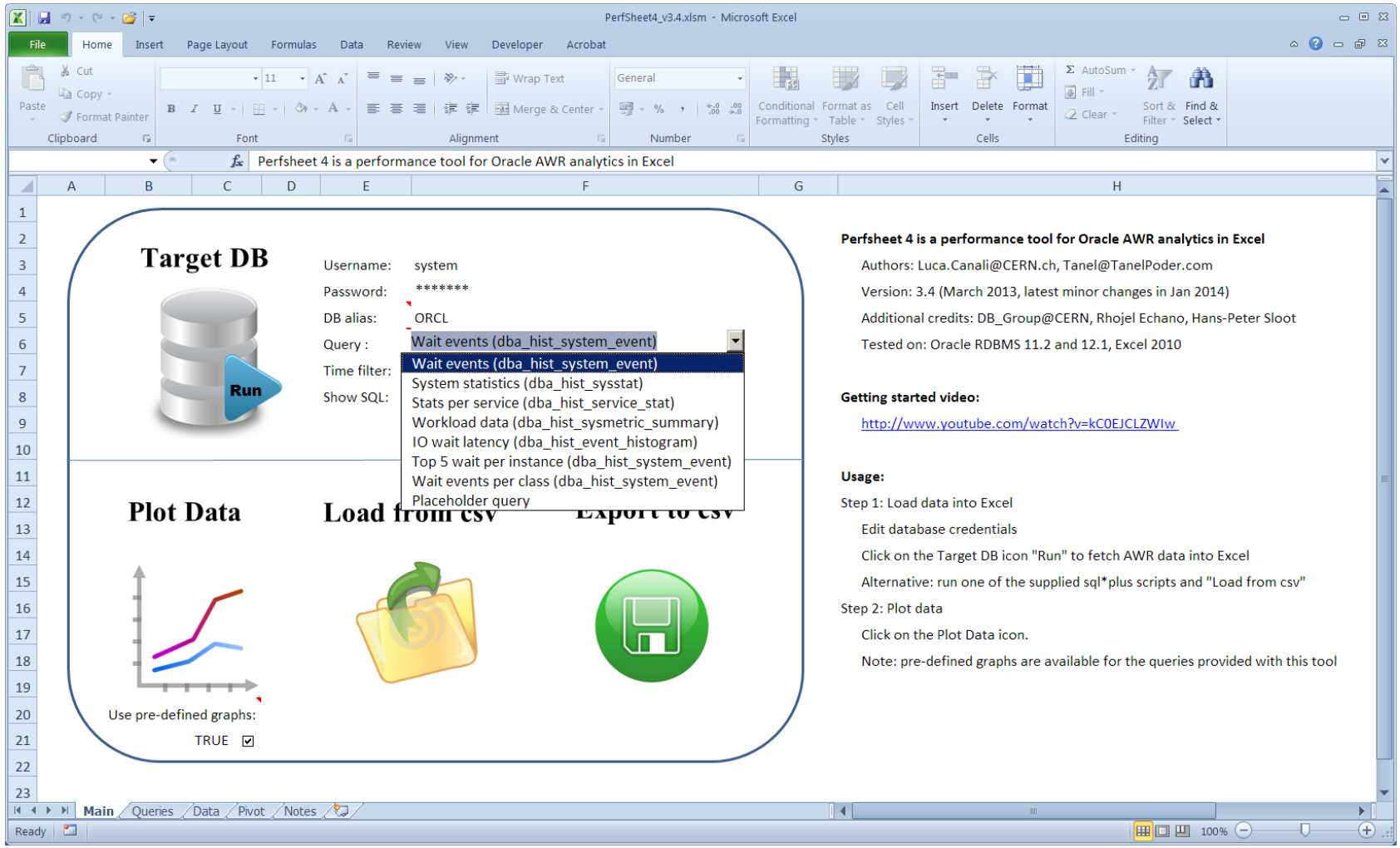
Tools: PerfSheet 4

- Simple analytic platform for AWR data
- Predefined queries and graphs
- Power of Pivot Charts

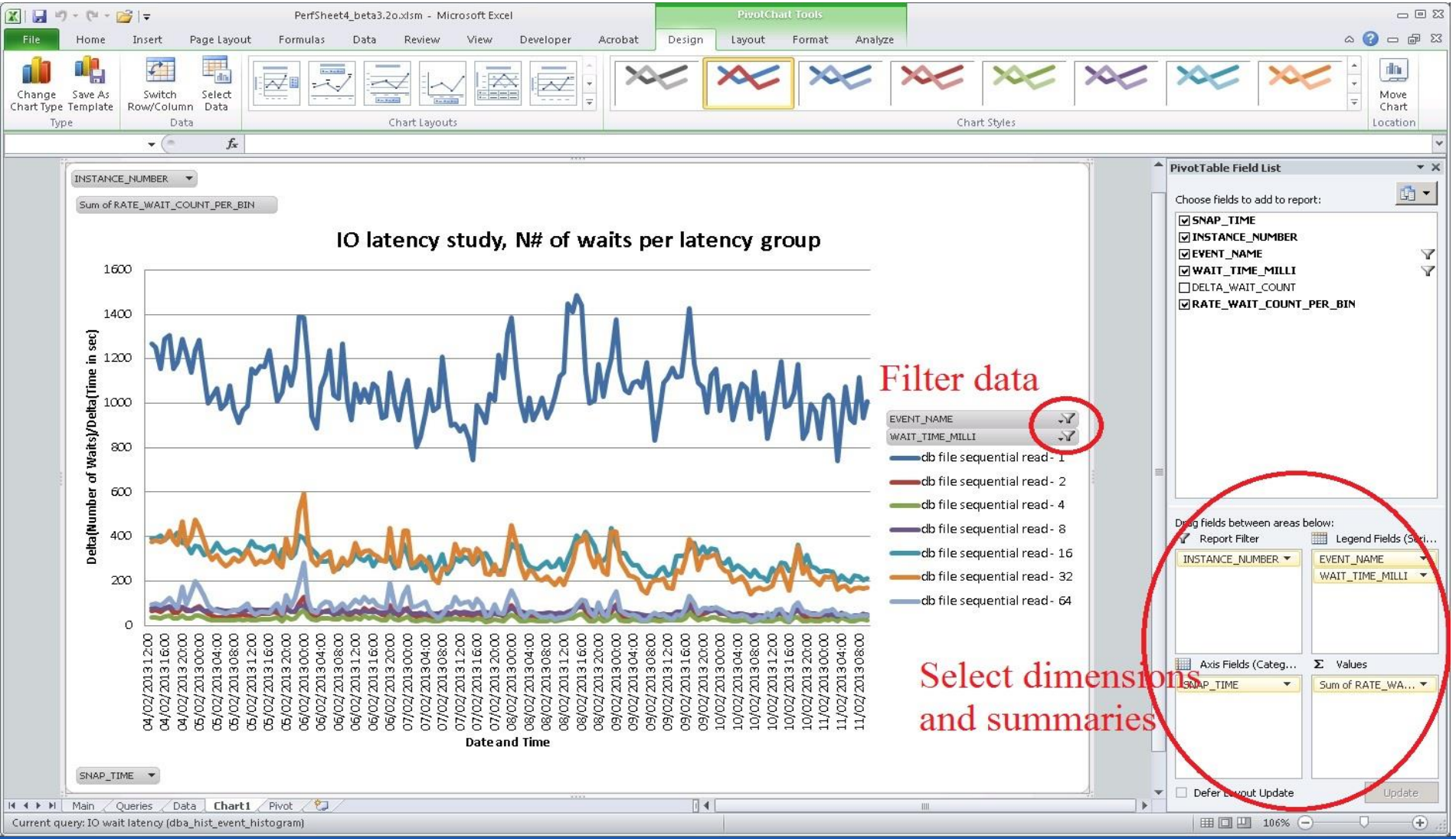




Tools: PerfSheet 4



Tools: PerfSheet 4



Tools: OraLatencyMap

- Run on SQL*Plus, core written in PL/SQL
  - Lightweight, does not require any installation
  - Command line interface
  - Heat maps generated using ANSI escape codes

- Getting started:

```
SQL> @OraLatencyMap
```

Tools: PyLatencyMap

- It's written in Python + SQL*Plus scripts
  - No installations required, CLI, lightweight
  - It's an advanced version of OraLatencyMap

- Can be used for generic latency sources
  - Oracle v$, trace files, AWR, DTrace data, etc
  - Pre-built examples available
  - Additional features: record and replay

Getting Started with PyLatencyMap

- Modular architecture
  - Source | <optional filter> | visualization engine

- Getting started

```
$ ./Example1_oracle_random_read.sh
```

- Video, getting started with PyLatencyMap
- http://www.youtube.com/watch?v=-YuShn6ro1g

Outline

- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

More Latency Sources

- Latency can be collected from various other sources
  - From the OS
  - From Oracle RDBMS trace files and X$ tables
  - From the storage instrumentation

OS Level

- DTrace
  - Great performance tool, coming to Linux too
  - Naturally fits to gathering I/O latency data
    - Histograms collected with the quantize operator

```
dtrace -n '
syscall::pread64:entry { self->s = timestamp; }
syscall::pread64:return /self->s/ { @pread["ns"] =
quantize(timestamp -self->s); self->s = 0; }

tick-10s {
printa(@pread);
trunc(@pread);
}'
```

More DTrace

- DTrace Oracle executable for wait event data
  - This scripts uses the DTrace PID provider

```
dtrace  -n '
pid2349:oracle:kews_update_wait_time:entry,pid2350:oracl
e:kews_update_wait_time:entry {
     self->ela_time = arg1;
}
pid2349:oracle:kskthewt:entry,pid2350:oracle:kskthewt:en
try {
     @event_num[arg1] = quantize(self->ela_time);
}
tick-10s {
printa(@event_num);
trunc(@event_num);
}'
```

Sample DTrace Output

```
1 294099                                    :tick-10s
   146 (edited: db file sequential read)
 value  ------------- Distribution ------------- count
   128 |                                         0
   256 |@                                        14
   512 |@@@@@                                    120
  1024 |@@@@@@@@                                 182
  2048 |@@@@@@@@@@@@@@@@                         320
  4096 |@@@@@@@@@@@@@@                           302
  8192 |                                         5
 16384 |                                         4
 32768 |                                         0
```

```
dtrace -s DTrace/pread_tracedata.d |python
DTrace/dtrace_connector.py |python LatencyMap.py
```

Latency From 10046 Trace Files

- Latency data is available in 10046 trace files
  - Wait event data with micro second precision
  - Allows the drill-down to session level

```
SQL> exec dbms_monitor.session_trace_enable(sid,serial#)
```

  - Snippet from the trace file:

```
nam='db file sequential read' ela= 977 file#=7
block#=29618220 blocks=1 obj#=82015 tim=1377520823036634
```

  - Plug into PyLatencyMap for heat map display

```
cat tracefile.trc| python 10046_connector.py |python
LatencyMap.py
```

Sampling Instead of Tracing

- We want to process a stream of wait events
- ASH data OK but sampling frequency too low
  - Not fast enough for capturing all I/O events
  - "_ash_sampling_interval"=1000 -> 1 Hz
- I propose a different method:
  - High frequency sampling of v$session_wait_history
  - 10 latest wait events for each session
    - "_session_wait_history"=10

Latency Data From X$KSLWH

- V$session_wait_history is based on X$KSLWH
- Use X$KSLWH as ring buffers indexed by SID
  - Extract all captured wait events, wrap on kslwhwaitid

Selected columns from X$KSLWH:

kslwhsid	kslwhwaitid	kslwhetext	kslwhetime
42	21016	db file sequential read	300
42	21015	db file sequential read	1000
42	21014	db file sequential read	16000
42	21013	db file sequential read	400

Custom Event Histograms

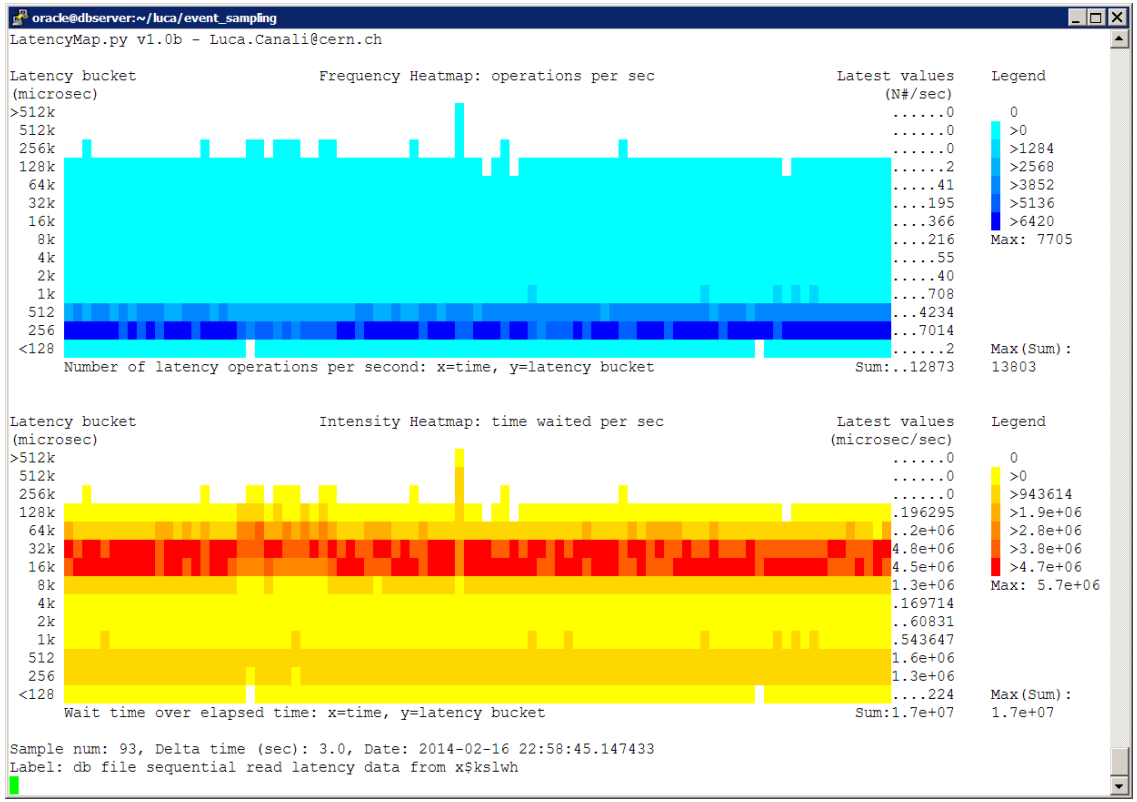
- X$KSLWH sampling with a custom python script
  - SIDs as input, latency histograms as output
  - Latency is measured in microseconds

```
$ python event_sampler_latency_histogram.py -i 3 -c

sampling rate: 6333 Hz, iops: 10396
bucket (microsec), wait count
...
256 , 20535
512 , 7708
1024 , 1923
2048 , 129
4096 , 78
...
```

Sampling Events and Heat Maps

- Heat maps from X$KSLWH
  - High frequency sampling of 16 busy SLOB sessions



NetApp ONTAP 8

- Latency data from the storage OS
  - Latency measured at the source

```
mynas::> system node run -node dbnas1 stats show -r -n
100 -i 3 volume:myvol1:nfs_protocol_read_latency

volume:myvol1:nfs_protocol_read_latency.<20us:2656
volume:myvol1:nfs_protocol_read_latency.<40us:25885
volume:myvol1:nfs_protocol_read_latency.<60us:1040
volume:myvol1:nfs_protocol_read_latency.<80us:95
volume:myvol1:nfs_protocol_read_latency.<100us:16
volume:myvol1:nfs_protocol_read_latency.<200us:14
volume:myvol1:nfs_protocol_read_latency.<400us:11
volume:myvol1:nfs_protocol_read_latency.<600us:1043
volume:myvol1:nfs_protocol_read_latency.<800us:37
…
```

Outline

- CERN and Oracle
- Storage latency investigations and Oracle
- Examples
- Tools
- More sources of latency data
- Conclusions

Summary

- I/O latency data
  - Allows DBAs to get insights on storage performance
  - We need histograms, average latency is not enough
- ‘Db file sequential read’ event histogram
  - Great for investigations of random reads
  - Visualize data using latency heat maps
- Scripts to automate this process in Oracle:
  - Download OraLatencyMap or PyLatencyMap

Wish List:

- More awareness of I/O latency
  - No more IOPS data without latency values!
  - 'Average latency' hides details, use histograms instead
- We need more features for measuring and analysing I/O latency for Oracle DB workloads
  - A PL/SQL interface?
  - DTrace static probes for Oracle RDBMS?
- OEM could make use of Heat Maps for I/O studies
- V$event_histogram should be extended
  - Latency details extended to the microsecond bucket

Acknowledgements

- CERN Database Group
  - In particular for their contributions to this presentation: Marcin Blaszczyk and Ruben Gaspar

- Many thanks for sharing their work and ideas to:
  - Brendan Gregg, Tanel Poder, Kevin Closson, Frits Hoogland, Marcin Przepiorowski, James Morles, Kyle Hailey, Cary Millsap

Thank you!

Luca.Canali@CERN.ch

Download scripts and tools from:

http://cern.ch/canali/resources.htm

www.cern.ch