performing
databases

Do Not Run-Away!
Oracle Performance for the Uncertain

Martin Klier

Performing Databases GmbH
Mitterteich / Germany

Speaker

- Martin Klier
- Solution Architect and Database Expert
- My focus:
  - Performance + Tuning
  - Highly available systems
  - Cluster and Replication

- Linux since 1997
- Oracle Database since 2003

ORACLE ACE Director

Performing Databases

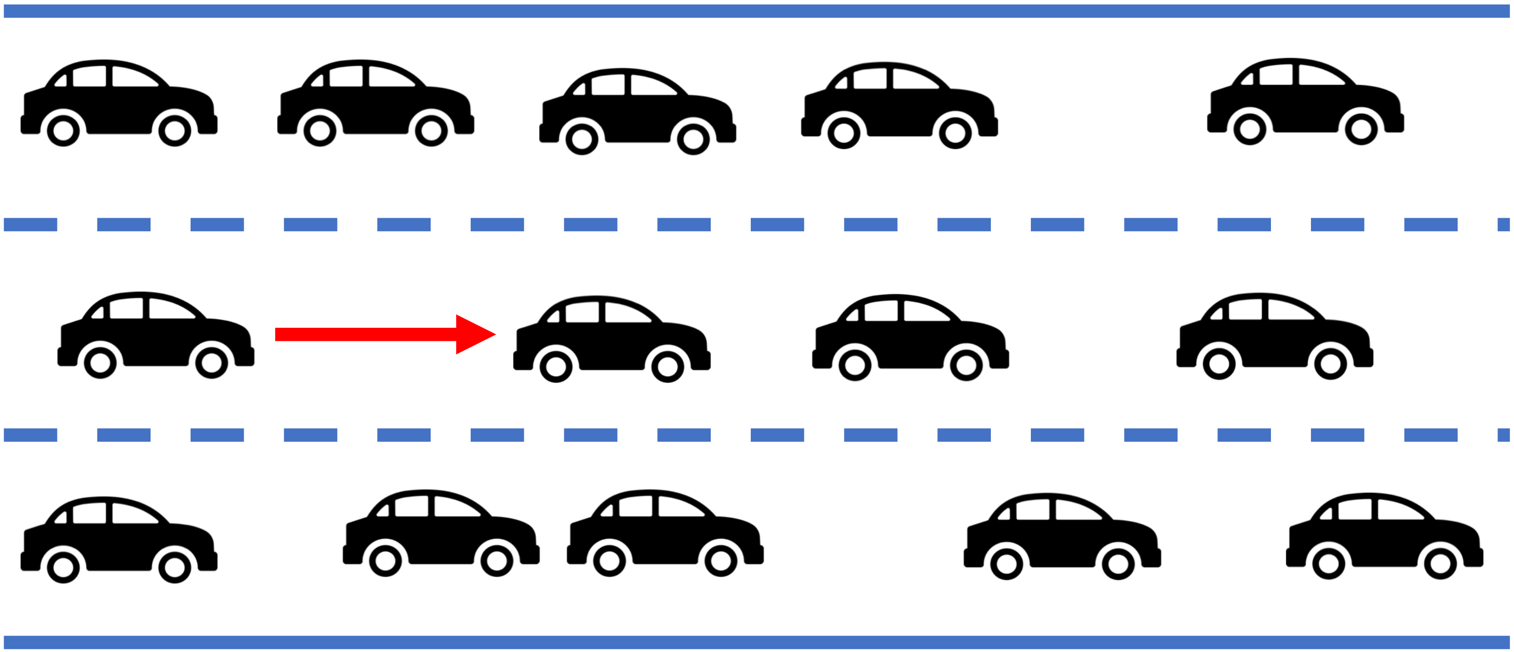
- Three Experts for Database technology
  - Concepts and Project Competence
  - Architecture- and System planning
  - Licensing
  - Implementation and Troubleshooting

- Get in touch
  - Performing Databases GmbH
    Wiesauer Strasse 27
    95666 Mitterteich // Germany
  - http://www.performing-databases.com
  - Twitter: @PerformingDB

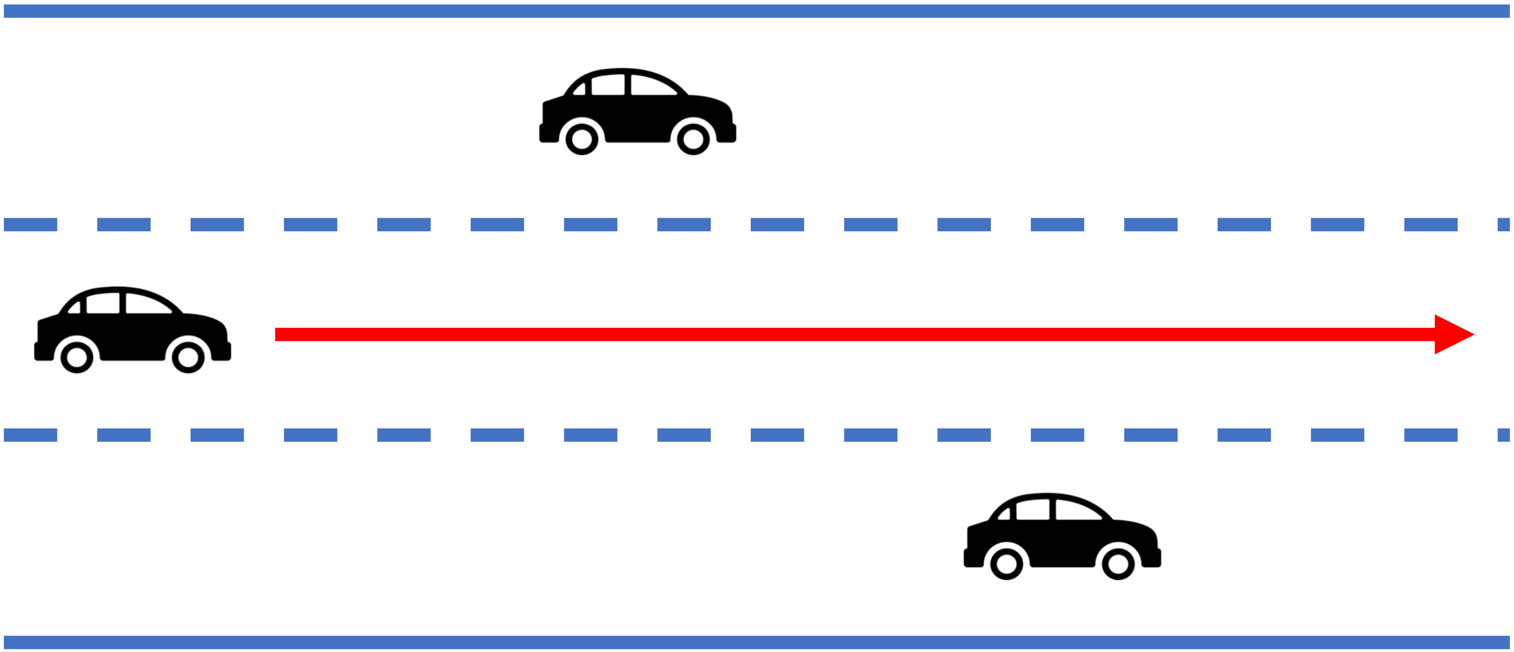
Id	Operation	Name	Starts	E-Rows	E-Bytes	Cost (%CPU)	E-Time	A-Rows	A-Time	Buffers
0	SELECT STATEMENT		1			1832 (100)		70	00:00:04.48	613K
1	SORT ORDER BY		1	5	2590	1832 (1)	00:00:22	70	00:00:04.48	613K
2	VIEW	VIEWRTPICKING	1	5	2590	1831 (1)	00:00:22	70	00:00:04.48	613K
3	SORT UNIQUE		1	5	1376	1831 (75)	00:00:22	70	00:00:04.48	613K
4	UNION-ALL		1					70	00:00:04.48	613K
* 5	TABLE ACCESS BY INDEX ROWID	PICKORDERPOS	70	1	9	3 (0)	00:00:01	70	00:00:00.01	120
* 6	INDEX UNIQUE SCAN	XPKPICKORDERPOS	70	1		2 (0)	00:00:01	70	00:00:00.01	50
7	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	24	2 (0)	00:00:01	35	00:00:00.01	73
* 8	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
9	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	24	2 (0)	00:00:01	35	00:00:00.01	73
* 10	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
11	NESTED LOOPS		35					35	00:00:00.01	77
12	NESTED LOOPS		35	1	16	3 (0)	00:00:01	35	00:00:00.01	42
* 13	INDEX RANGE SCAN	I_STORAGELOCATION_TUNING_1	35	1	9	2 (0)	00:00:01	35	00:00:00.01	38
* 14	INDEX UNIQUE SCAN	XPKWAREHOUSE	35	1		0 (0)		35	00:00:00.01	4
15	TABLE ACCESS BY INDEX ROWID	WAREHOUSE	35	43	301	1 (0)	00:00:01	35	00:00:00.01	35
16	NESTED LOOPS		35	1	25	3 (0)	00:00:01	35	00:00:00.01	117
17	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	13	2 (0)	00:00:01	35	00:00:00.01	73
* 18	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
19	TABLE ACCESS BY INDEX ROWID	RACK	35	11276	132K	1 (0)	00:00:01	35	00:00:00.01	44
* 20	INDEX UNIQUE SCAN	XPKRACK	35	1		0 (0)		35	00:00:00.01	9
21	NESTED LOOPS		35	1	25	3 (0)	00:00:01	35	00:00:00.01	117
22	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	13	2 (0)	00:00:01	35	00:00:00.01	73
* 23	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
24	TABLE ACCESS BY INDEX ROWID	RACK	35	11276	132K	1 (0)	00:00:01	35	00:00:00.01	44
* 25	INDEX UNIQUE SCAN	XPKRACK	35	1		0 (0)		35	00:00:00.01	9
26	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	33	1	9	2 (0)	00:00:01	33	00:00:00.01	69
* 27	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	33	1		1 (0)	00:00:01	33	00:00:00.01	36
28	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	2	1	9	2 (0)	00:00:01	2	00:00:00.01	6
* 29	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	2	1		1 (0)	00:00:01	2	00:00:00.01	4
30	SORT AGGREGATE		1	1	8			1	00:00:00.01	376
* 31	INDEX FAST FULL SCAN	XAK1STORAGELOCATION	1	20257	158K	80 (2)	00:00:01	20773	00:00:00.01	376
32	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	9	2 (0)	00:00:01	35	00:00:00.01	73
* 33	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
34	HASH UNIQUE		36	1	8	145K (1)	00:29:09	36	00:00:04.48	610K
* 35	COUNT STOPKEY		36					36	00:00:04.45	610K
* 36	FILTER		36					36	00:00:04.45	610K
37	TABLE ACCESS FULL	PACKINGUNIT	36	28340	221K	210 (1)	00:00:03	556K	00:00:00.01	15311
38	NESTED LOOPS		556K					36	00:00:03.42	595K
39	NESTED LOOPS		556K	1	32	6 (0)	00:00:01	87	00:00:03.17	595K
40	TABLE ACCESS BY INDEX ROWID	LU	556K	1	19	4 (0)	00:00:01	87	00:00:03.00	595K
* 41	INDEX RANGE SCAN	I_TUNING_LU_3	556K	1		3 (0)	00:00:01	87	00:00:02.90	595K
* 42	INDEX UNIQUE SCAN	XPKTU	87	1		1 (0)	00:00:01	87	00:00:00.01	214
* 43	TABLE ACCESS BY INDEX ROWID	TU	87	1	13	2 (0)	00:00:01	36	00:00:00.01	87
44	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	8	2 (0)	00:00:01	35	00:00:00.01	73
* 45	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
46	NESTED LOOPS		1					70	00:00:00.01	1135
47	NESTED LOOPS		1	1	272	473 (1)	00:00:06	140	00:00:00.01	1088
48	NESTED LOOPS OUTER		1	1	263	469 (1)	00:00:06	70	00:00:00.01	1020
49	NESTED LOOPS		1	1	256	467 (1)	00:00:06	70	00:00:00.01	912
50	NESTED LOOPS		1	1	245	466 (1)	00:00:06	70	00:00:00.01	812
51	NESTED LOOPS		1	1	230	463 (0)	00:00:06	70	00:00:00.01	664
52	NESTED LOOPS		1	1	189	462 (0)	00:00:06	70	00:00:00.01	564
53	NESTED LOOPS		1	1	182	461 (0)	00:00:06	70	00:00:00.01	388
54	NESTED LOOPS		1	1	139	460 (0)	00:00:06	70	00:00:00.01	245
55	NESTED LOOPS		1	1	113	458 (0)	00:00:06	70	00:00:00.01	97
56	NESTED LOOPS		1	41	3116	344 (0)	00:00:05	6	00:00:00.01	52
57	NESTED LOOPS		1	41	2747	303 (0)	00:00:04	6	00:00:00.01	38
58	NESTED LOOPS		1	193	9457	27 (0)	00:00:01	6	00:00:00.01	26
59	NESTED LOOPS		1	5	195	4 (0)	00:00:01	6	00:00:00.01	11
60	NESTED LOOPS		1	1	22	2 (0)	00:00:01	1	00:00:00.01	4
* 61	INDEX UNIQUE SCAN	XPKWAREHOUSE	1	1	4	0 (0)		1	00:00:00.01	1
* 62	TABLE ACCESS BY INDEX ROWID	RACK	1	1	18	2 (0)	00:00:01	1	00:00:00.01	3
* 63	INDEX RANGE SCAN	IDXRACK_1	1	1		1 (0)	00:00:01	1	00:00:00.01	2
64	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	1	5	85	2 (0)	00:00:01	6	00:00:00.01	7
* 65	INDEX RANGE SCAN	XIF1STORAGELOCATION	1	5		1 (0)	00:00:01	6	00:00:00.01	2
66	TABLE ACCESS BY INDEX ROWID	TU	6	38	380	15 (0)	00:00:01	6	00:00:00.01	15
* 67	INDEX RANGE SCAN	XIF2TU	6	170		2 (0)	00:00:01	6	00:00:00.01	14
* 68	TABLE ACCESS BY INDEX ROWID	PICKTUPOS	6	1	18	3 (0)	00:00:01	6	00:00:00.01	12
* 69	INDEX RANGE SCAN	IDXPICKTUPOS_2	6	2		1 (0)	00:00:01	6	00:00:00.01	8
* 70	TABLE ACCESS BY INDEX ROWID	PICKTU	6	1	9	1 (0)	00:00:01	6	00:00:00.01	14
* 71	INDEX UNIQUE SCAN	XPKPICKTU	6	1		0 (0)		6	00:00:00.01	8
* 72	TABLE ACCESS BY INDEX ROWID	PICKORDERPOS	6	1	37	6 (0)	00:00:01	70	00:00:00.01	45
* 73	INDEX RANGE SCAN	XIF1PICKORDERPOS	6	22		2 (0)	00:00:01	255	00:00:00.01	14
* 74	TABLE ACCESS BY INDEX ROWID	MAILORDERPOS	70	1	26	2 (0)	00:00:01	70	00:00:00.01	148
* 75	INDEX UNIQUE SCAN	XPKMAILORDERPOS	70	1		1 (0)	00:00:01	70	00:00:00.01	74
76	TABLE ACCESS BY INDEX ROWID	SKU	70	1	43	1 (0)	00:00:01	70	00:00:00.01	143
* 77	INDEX UNIQUE SCAN	XPKSKU	70	1		0 (0)		70	00:00:00.01	73
78	TABLE ACCESS BY INDEX ROWID	SKUPRJ	70	1	7	1 (0)	00:00:01	70	00:00:00.01	176

Latency vs. Throughput

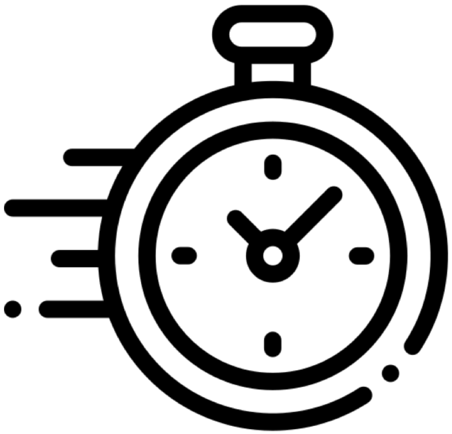
High Throughput
High Latency



Low Throughput
Low Latency



Units of Work



Throughput = work / time

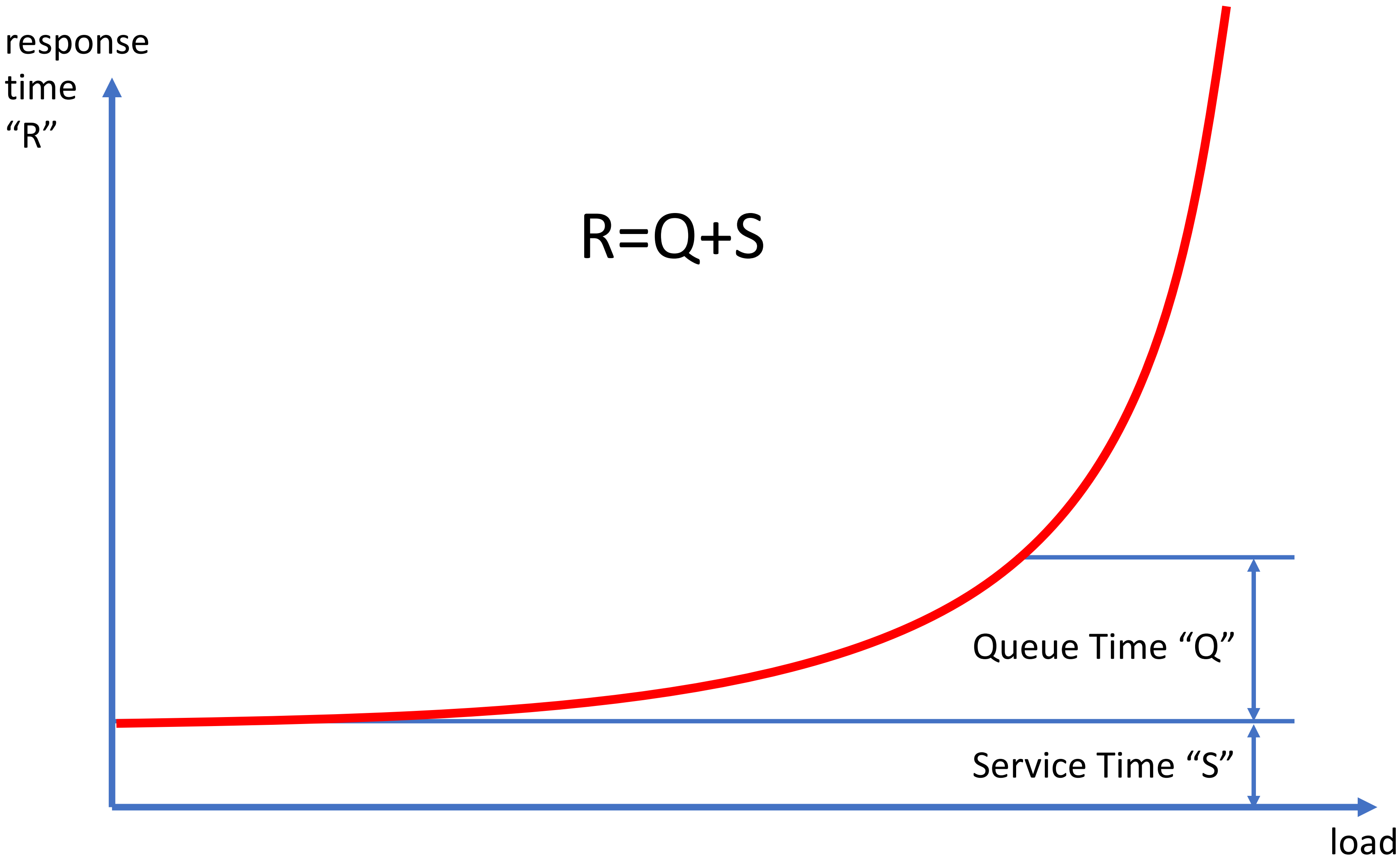
like:
Megabytes per Second
Floating Point Ops per Second
Cars per Hour

Latency = time / work

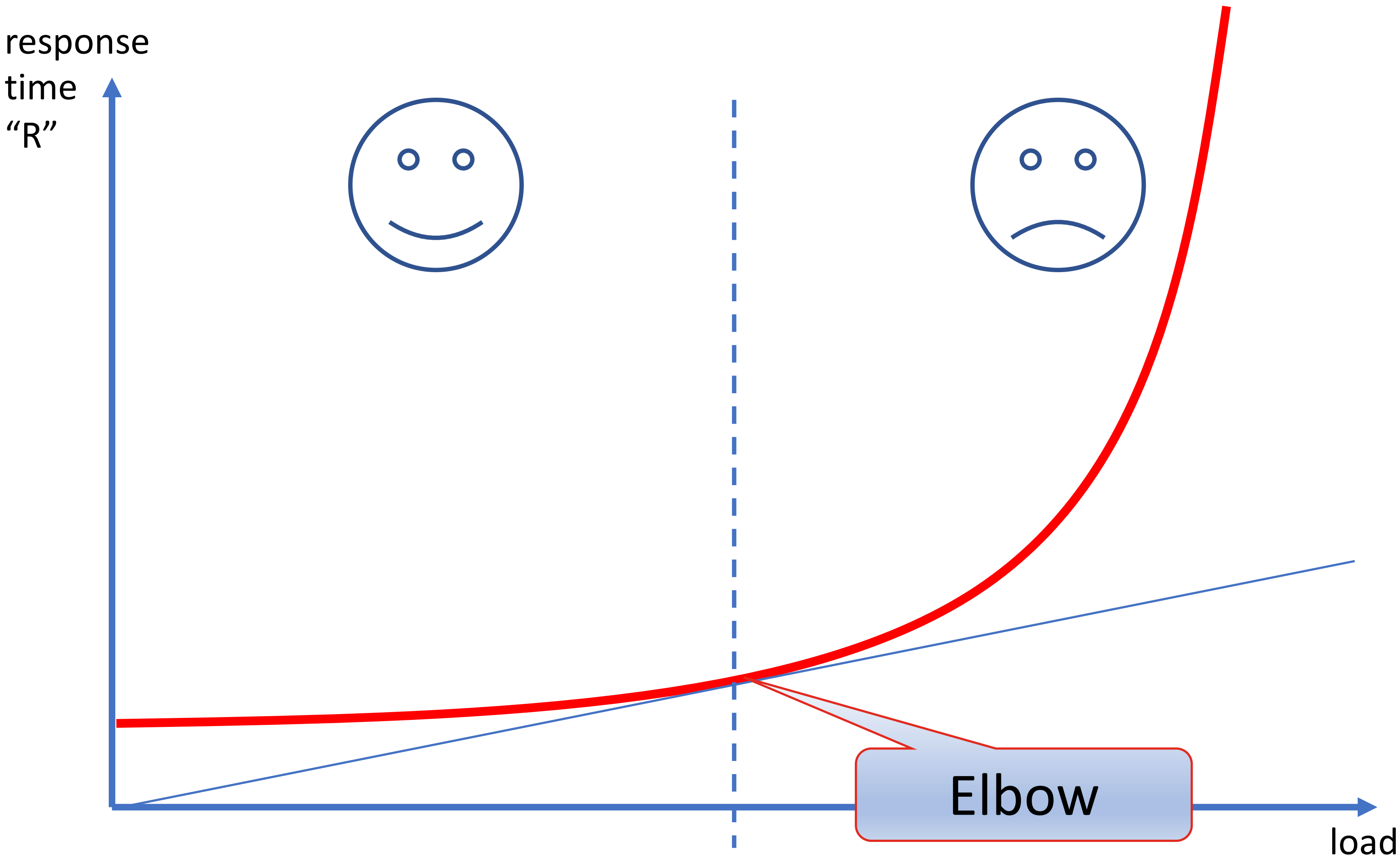
like:
Milliseconds per Read IO
Time per Instruction
Minutes for a Commute

Queueing Theory

Queueing

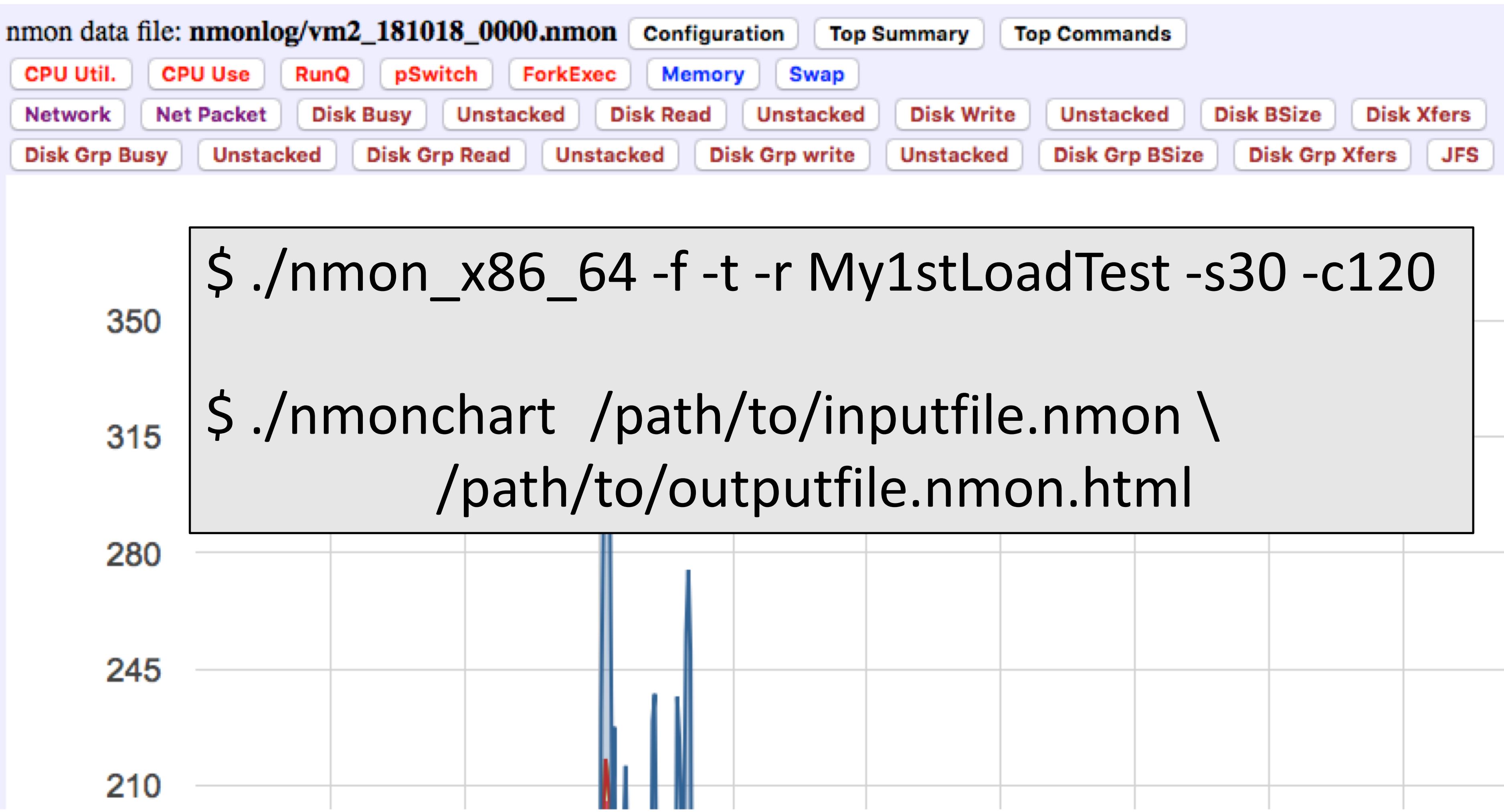


Queueing



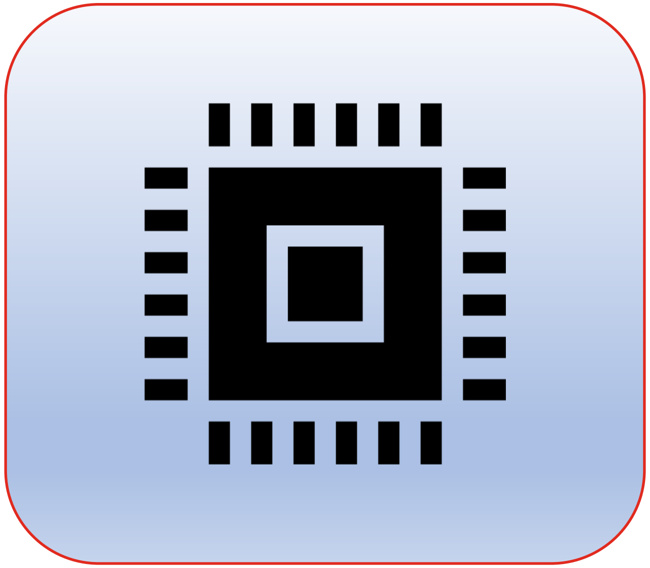
OS Level
History Tool

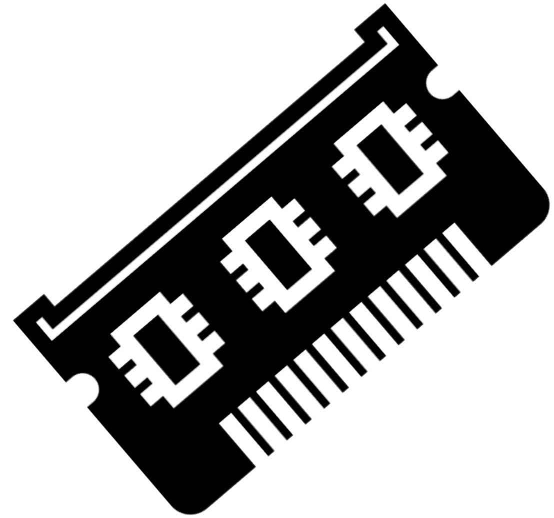
nmon + nmonchart

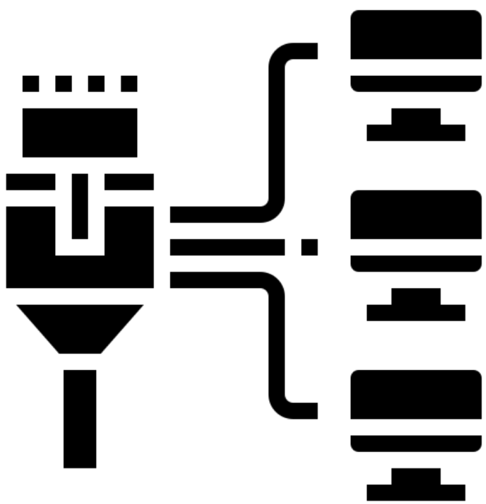


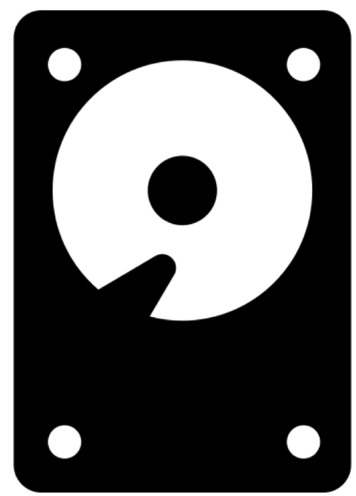

```
$ ./nmon_x86_64 -f -t -r My1stLoadTest -s30 -c120

$ ./nmonchart  /path/to/inputfile.nmon \
            /path/to/outputfile.nmon.html
```

Resources









CPU Time - time

```
$ time unzip V839960-01.zip >/dev/null

real    0m23.360s
user    0m20.492s
sys     0m2.846s
```

Elapsed, “wall clock”, overall time

spent in userland as process incl. wasted time ☺

Kernel time spent for sys calls incl. IO

CPU Time - top

```
top - 11:56:07 up 31 min,  2 users,  load average: 0,16,
Tasks: 110 total,   3 running, 107 sleeping,   0 stopped,
%Cpu0  :  0,0 us,  0,0 sy,  0,0 ni,100,0 id,  0,0 wa,                0,0 si,  0,0 st
%Cpu1  :  0,0 us,  1,0 sy,  0,0 ni, 86,9 id, 11,1 wa,  0,0 hi,  1,0 si,  0,0 st
%Cpu2  :  0,0 us,  0,0 sy,  0,0 ni,100,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
%Cpu3  : 86,4 us, 13,6 sy,  0,0 ni,  0,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
KiB Mem :  8173544 total,  2848276 free,    82904 used,  5242364 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.  8013688 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1131 oracl     20   0  111300           1880 R 100,0  0,0   0:10.62 unzip V839960-01.zip
  281 root      20   0       0              0 D   2,0
    1 root      20   0  193640   828          S   0,0  0,
    2 roo       20   0       0      0         S   0,0  0,0
    3 ro        20   0       0      0             0,0  0,0
    5 r          0 -20       0      0             0,0  0,0
```

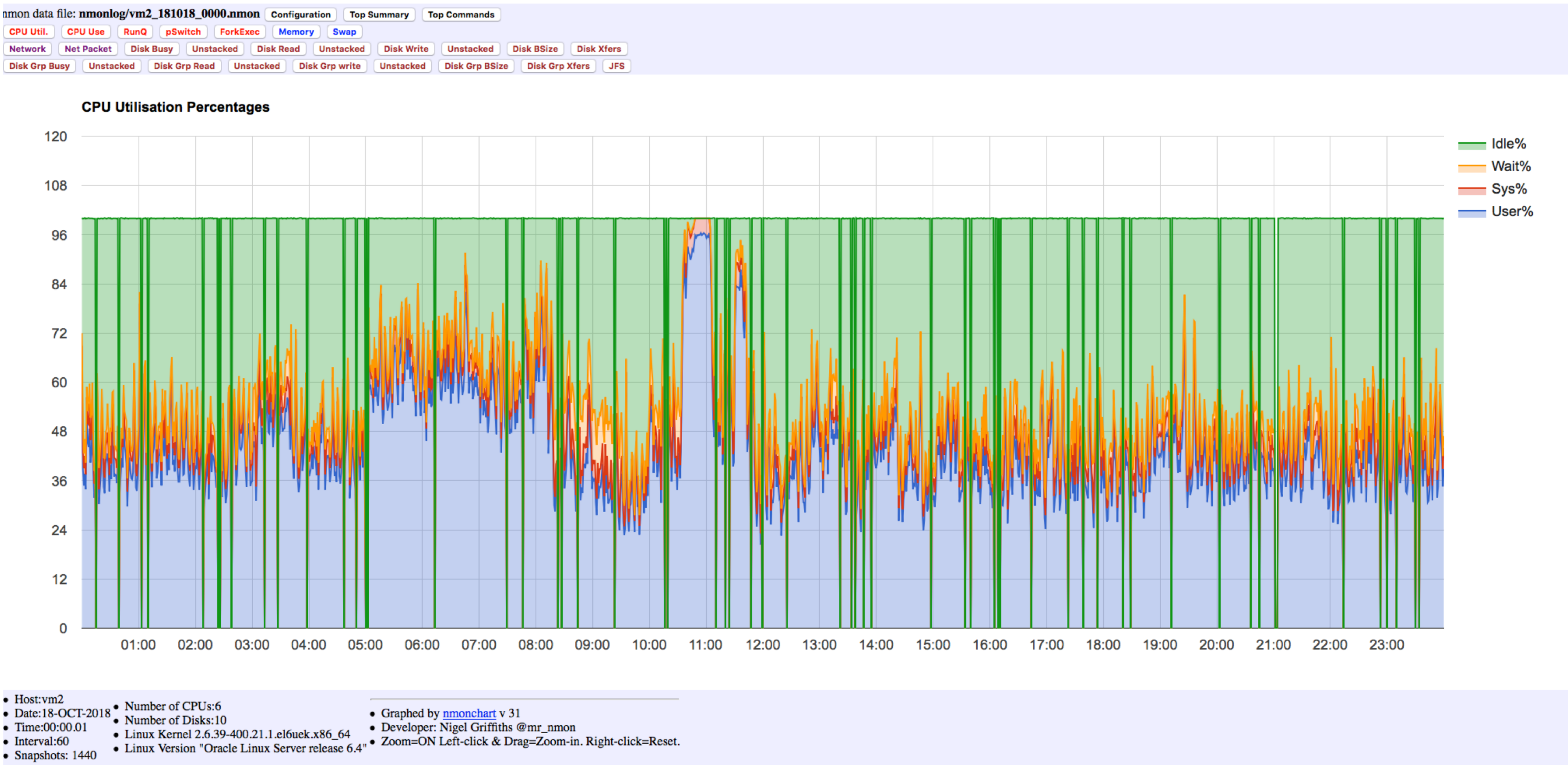
Waiting for IO

Using 100% of 1 core

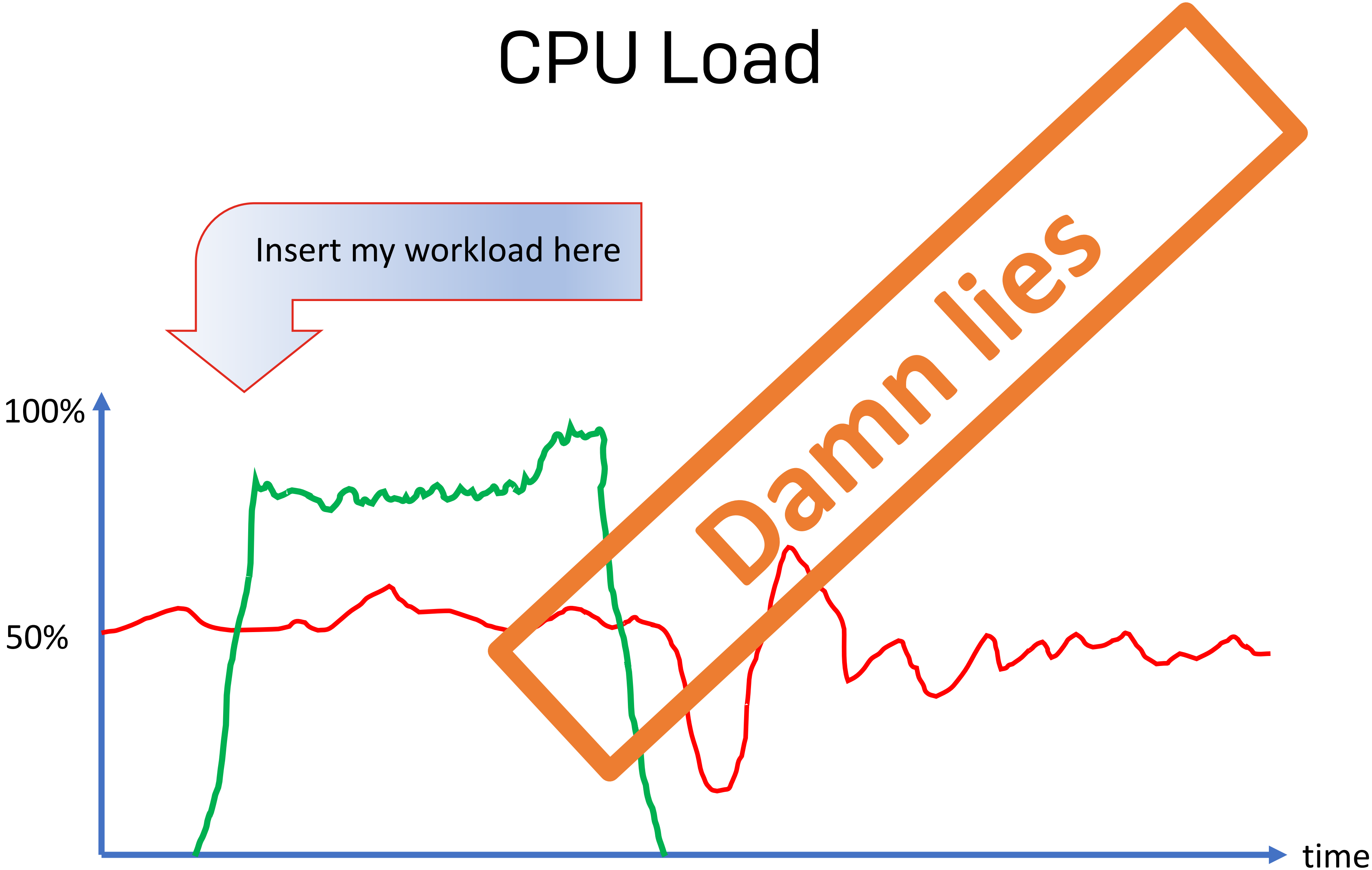
spent in userland as process

Kernel time spent for sys calls

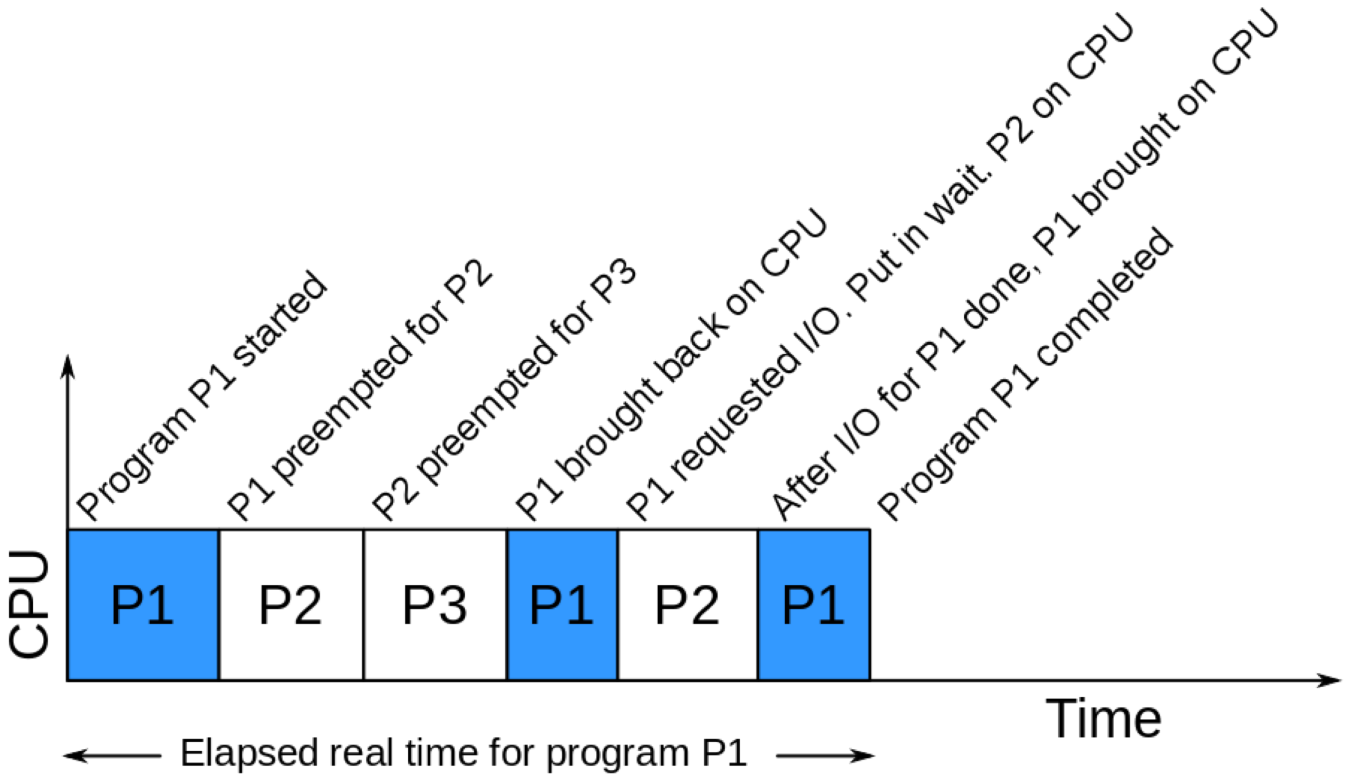
History: nmon + nmonchart



CPU Load

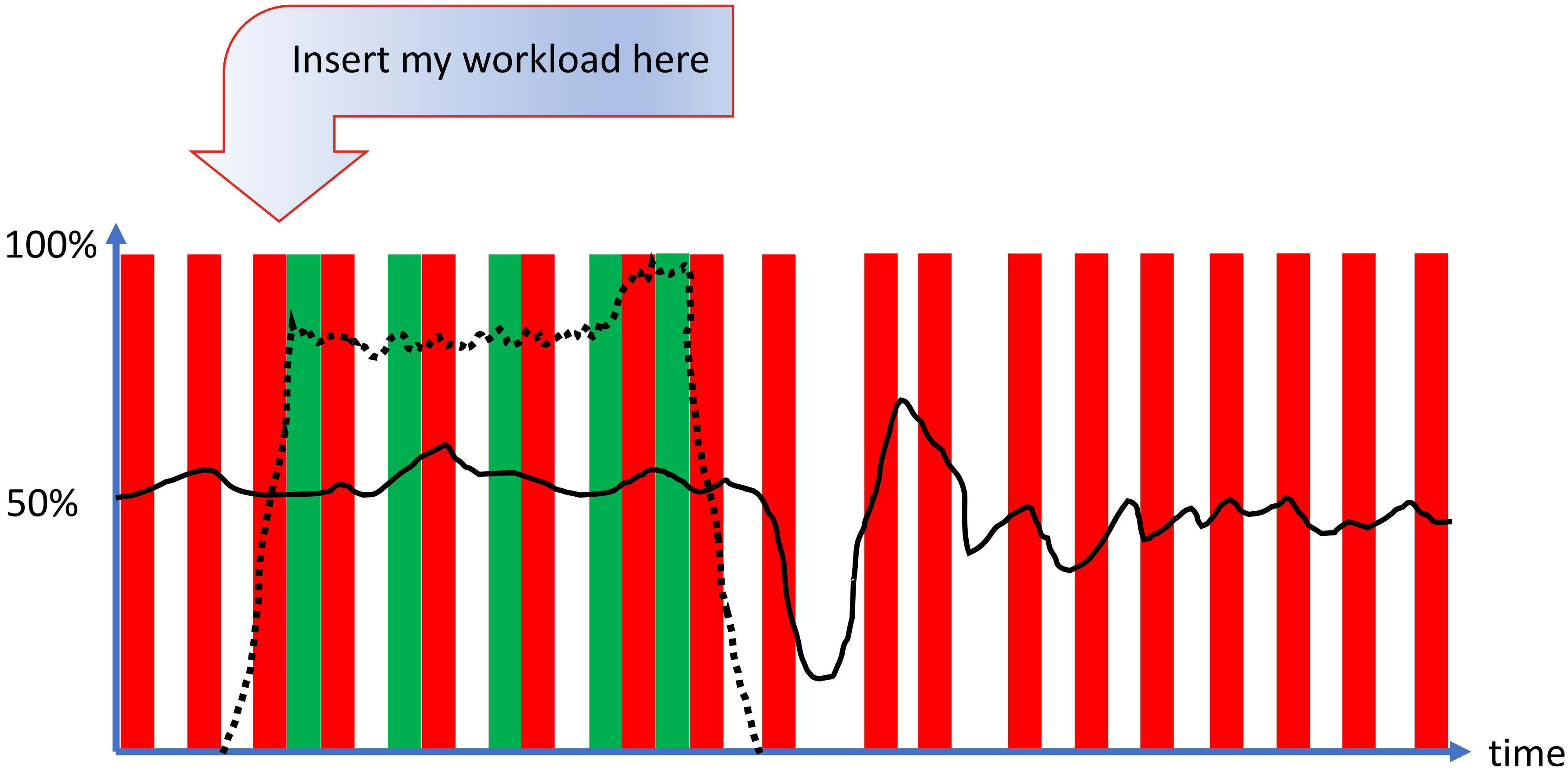


CPU Time Slots

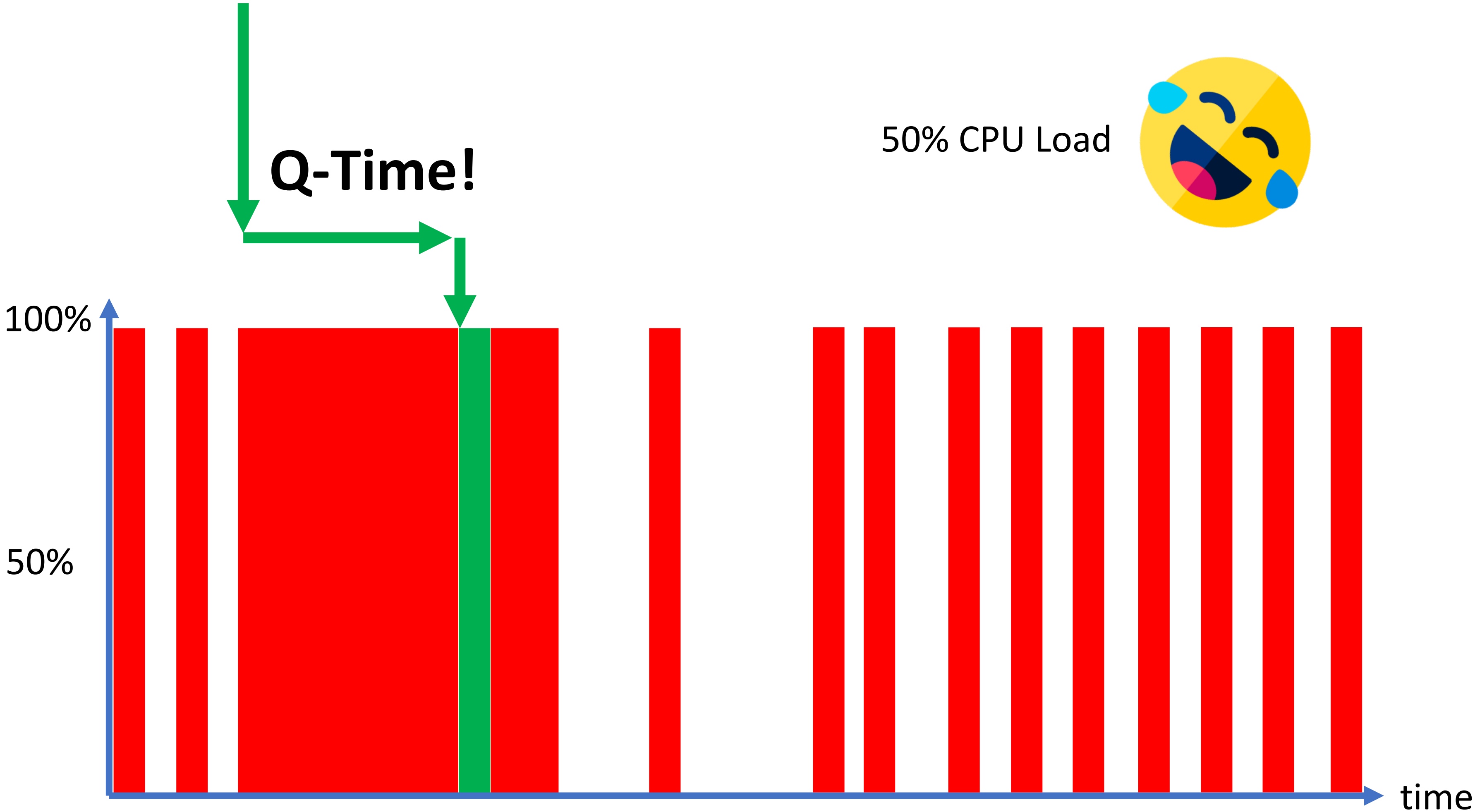


Picture: Wikipedia

CPU Load

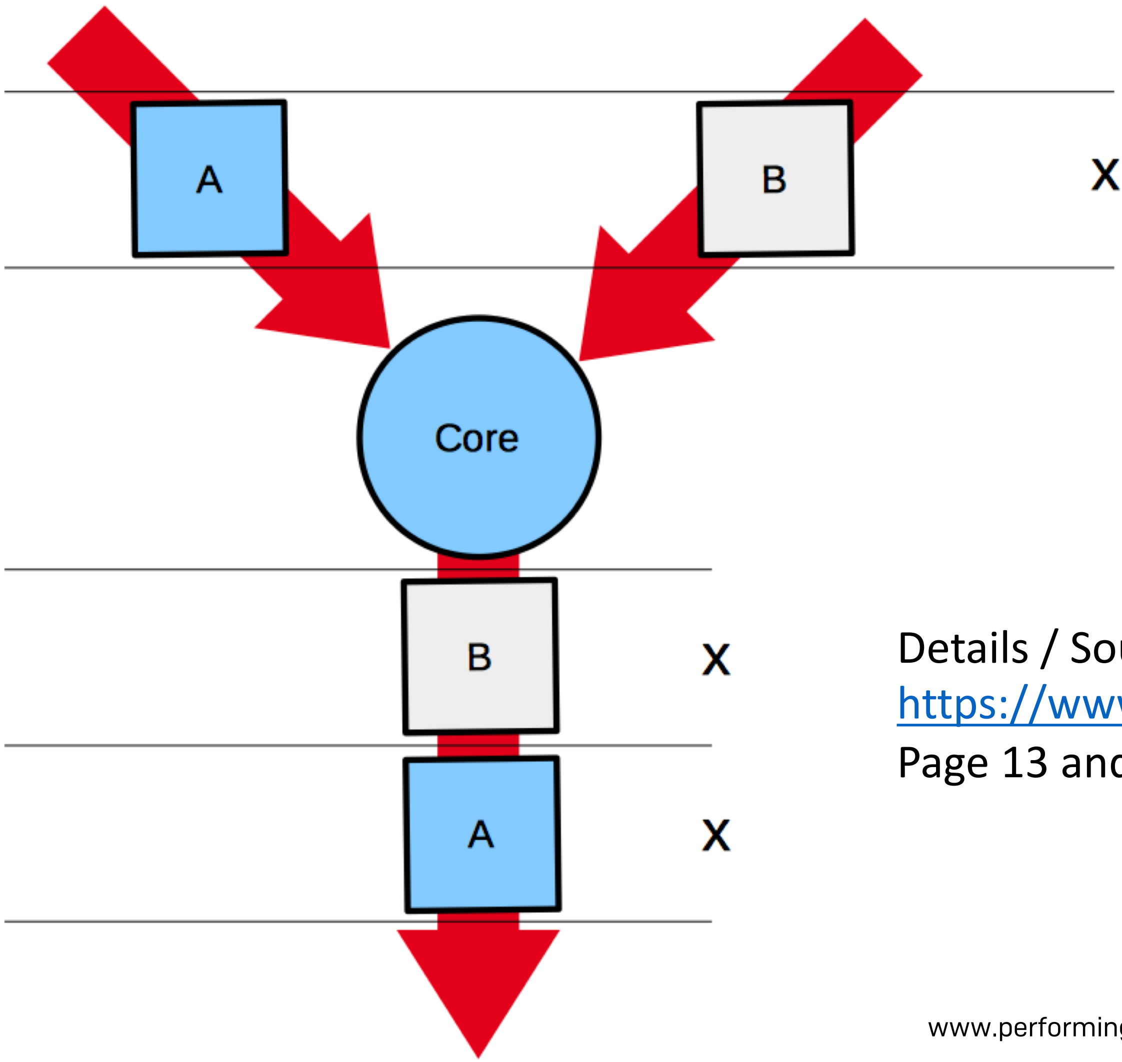


CPU Latency



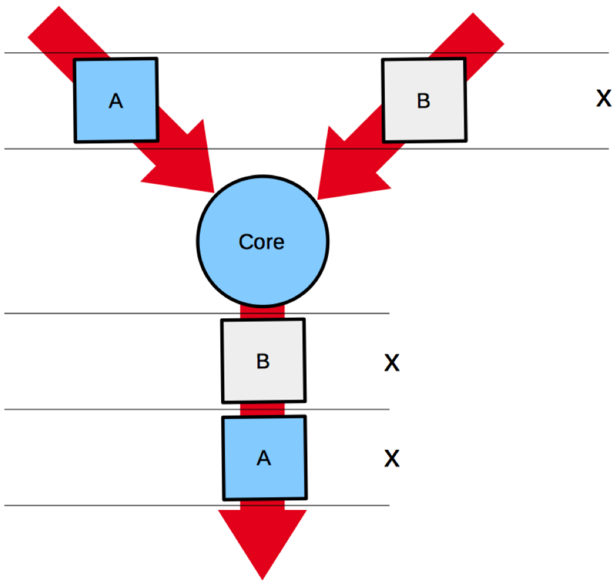
50% CPU Load

Latency in HyperThreading



Details / Source:
https://www.performing-datal
Page 13 and following

Latency in HyperThreading



Details / Source – my presentation about VMware performance
https://www.performing-databases.com/wp-content/uploads/2018/05/Klier-Oracle-on-VMware.pdf
Page 13 and following

Linux Load Average

```
top - 15:06:40 up 2 min,  2 users,  load average: 0,66, 0,22, 0,08
Tasks: 119 total,   3 running, 116 sleeping,   0 stopped,   0 zombie
%Cpu0  :  0,0 us,  0,0 sy,  0,0 ni,100,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
%Cpu1  : 31,6 us, 68,4 sy,  0,0 ni,  0,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
%Cpu2  :  0,0 us,  0,0 sy,  0,0 ni,100,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
%Cpu3  :  0,0 us,  0,0 sy,  0,0 ni,100,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
KiB Mem :  8173544 total,  7956652 free,    80216 used,   136676 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.  8024340 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  999 root      20   0    4580   1500   1208 R  97,1  0,0   1:00.39 gzip
 1016 root      20   0  157748   4212   3536 R   1,0  0,1   0:00.14 top
    1 root      20   0  193640   8160   5396 S   0,0  0,1   0:01.06 /usr/lib/systemd/sy
    2 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [kthreadd]
    3 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [ksoftirqd/0]
    4 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [kworker/0:0]
    5 root       0 -20       0      0      0 S   0,0  0,0   0:00.00 [kworker/0:0H]
    6 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [kworker/u8:0]
    7 root      20   0       0      0      0 S   0,0  0,0   0:00.02 [rcu_sched]
    8 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [rcu_bh]
    9 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [rcuos/0]
   10 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [rcuob/0]
   11 root      rt   0       0      0      0 S   0,0  0,0   0:00.00 [migration/0]
   12 root      rt   0       0      0      0 S   0,0  0,0   0:00.00 [watchdog/0]
```

Linux Load Average

```
# uptime
 15:10:03 up 6 min,  2 users,  load average: 1,01, 0,62, 0,27
```

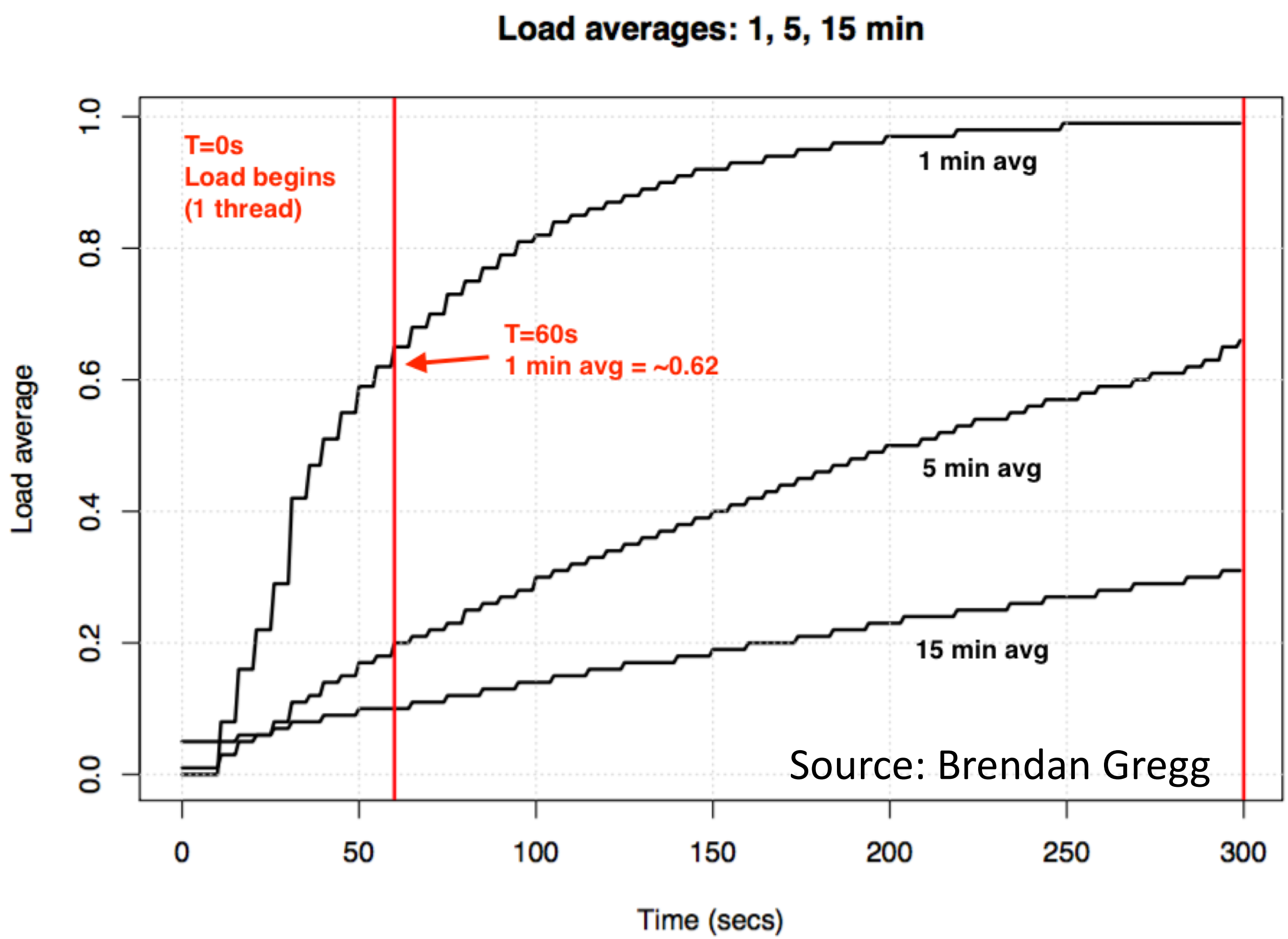
Last Minute Average

Last 5 Minutes Avg

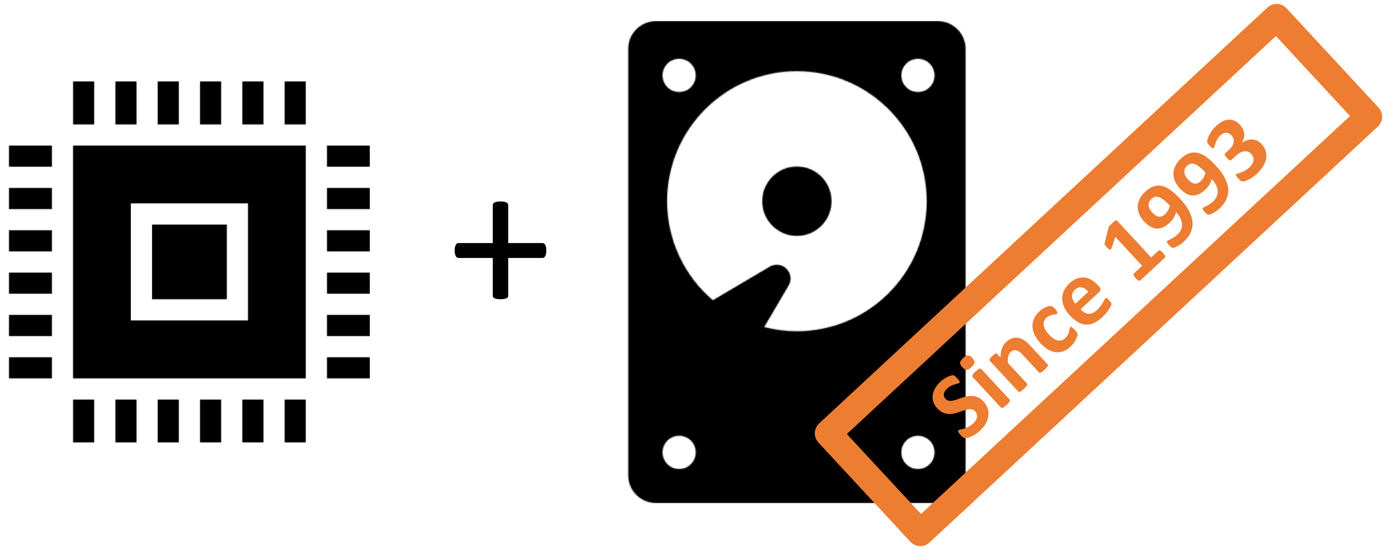
Last 15 Minutes Avg.

Damn lies

Linux Load Average



Linux Load Average



It’s a **SYSTEM load average**, not just CPU!

It contains RUNNING and QUEUED processes

Thus, also SWAPPING

and NONINTERRUPTIBLE (I/O)

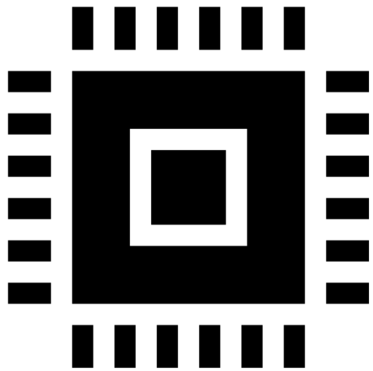
Linux Load Average

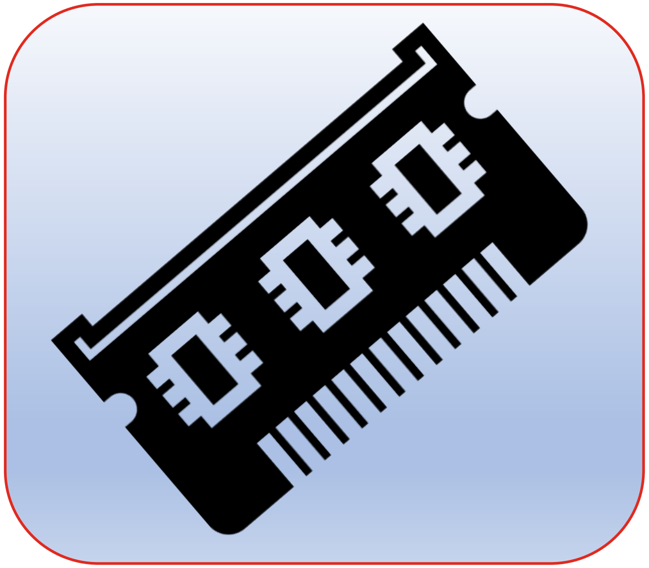
Complete your CPU picture with:

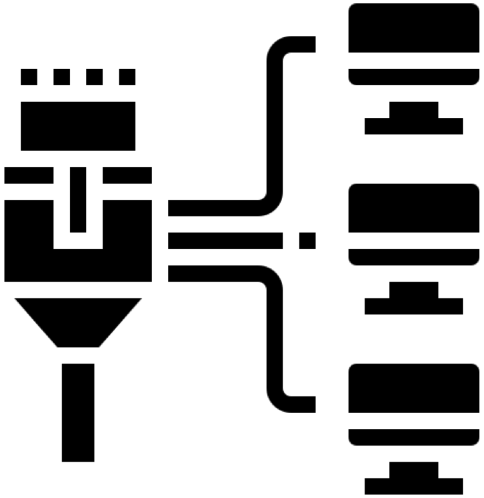
- **mpstat** 1 –P ALL
- **top** (try “t” in modern ones)
- **vmstat** 1 (r column = runqueue)

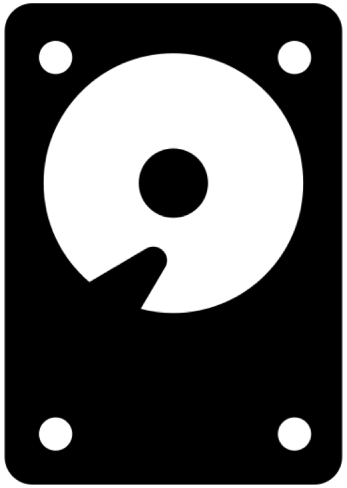
Plus I/O picture with:

- **iostat** 1







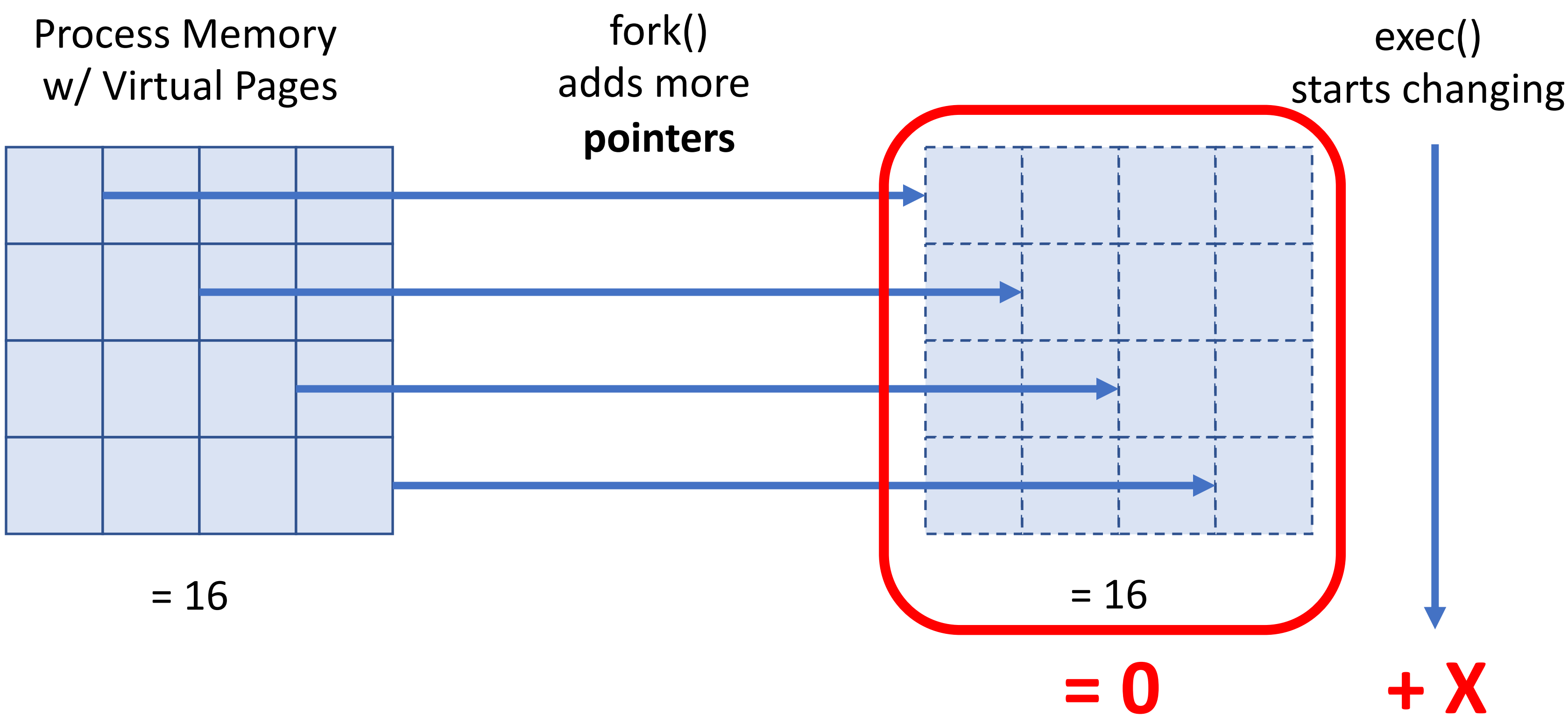


RAM Usage

```
top - 15:46:32 up 42 min,  2 users,  load average: 0,96, 0,70, 0,78
Tasks: 177 total,   2 running, 175 sleeping,   0 stopped,   0 zombie
%Cpu0  :  34,4/3,1    38[                                                        ]
%Cpu1  :  30,9/3,1    34[                                                        ]
%Cpu2  :  78,3/10,9   89[                                                        ]
%Cpu3  :  27,1/3,1    30[                                                        ]
KiB Mem :  8173544 total,  4166380 free,   542984 used,  3464180 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.  5093464 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1606 oracle    20   0 5882108  94608  17240 S 177,0  1,2   0:01.77 /u01/app/oracle/product/12.2.0/dbhome_+
 1112 oracle    -2   0 2893832  57984  55320 S   2,0  0,7   0:00.92 ora_vktm_KLM12201
   18 root      20   0       0      0      0 S   1,0  0,0   0:00.02 [rcuos/1]
 1170 oracle    20   0 2893836  61900  59236 S   1,0  0,8   0:00.05 ora_mmnl_KLM12201
 1444 oracle    20   0 3008532 761096 656376 S   1,0  9,3   1:01.93 ora_j004_KLM12201
 1471 root      20   0  157748   4396   3656 R   1,0  0,1   0:00.11 top
    1 root      20   0  193640   8160   5396 S   0,0  0,1   0:01.11 /usr/lib/systemd/systemd --switched-ro+
    2 root      20   0       0      0      0 S   0,0  0,0   0:00.00 [kthreadd]
    3 root      20   0       0      0      0 S   0,0  0,0   0:00.03 [ksoftirqd/0]
    5 root       0 -20       0      0      0 S   0,0  0,0   0:00.00 [kworker/0:0H]
    7 root      20   0       0      0      0 S   0,0  0,0   0:00.10 [rcu_sched]
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 [rcu_bh]
```

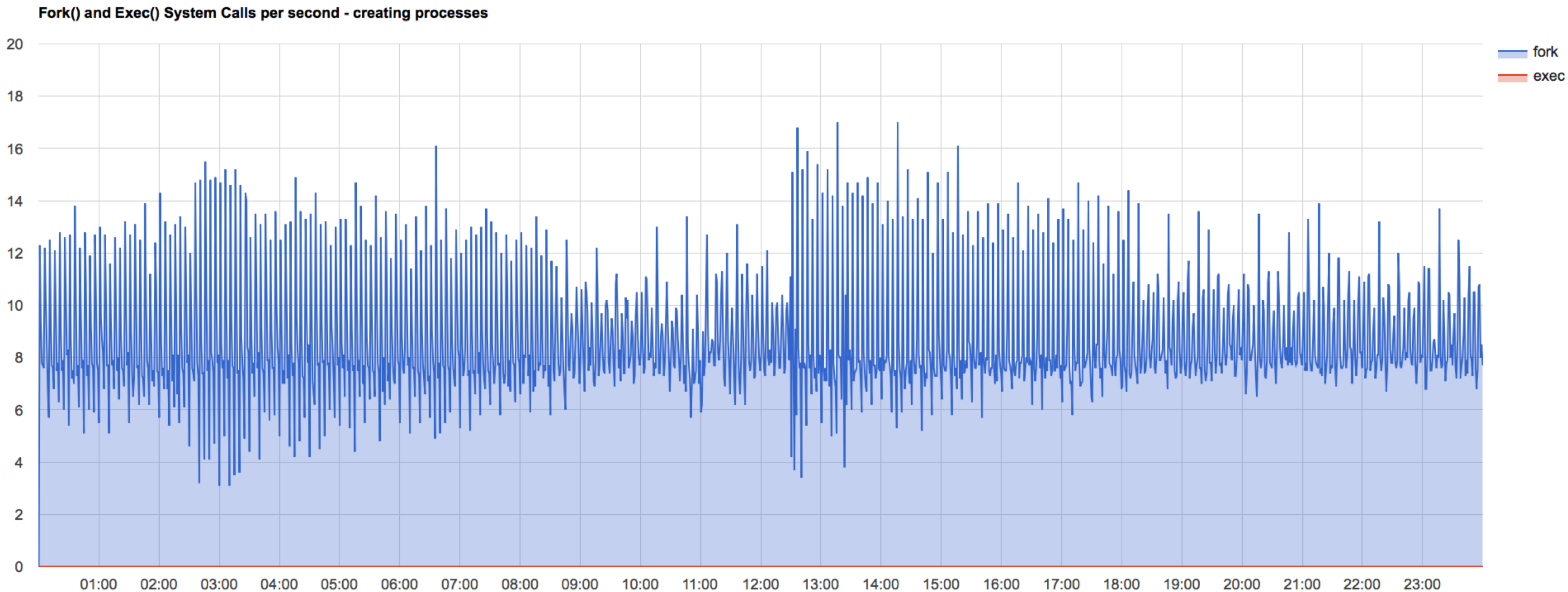
???

Linux Virtual Memory Management



Further reading: Linux Demand Paging

History: nmon + nmonchart



performing databases

@MartinKlierDBA
www.performing-databases.com

RAM Usage

VSZ = Virtual Memory Size
“all mem accessible for this process”

PSS = Proportional Set Size
“private plus prop. of shared” => crap

RSS = Resident Set Size
“including shared libs // no swap”

RAM Usage

USS = Unique Set Size

“which is guaranteed to be private”

=> my friend

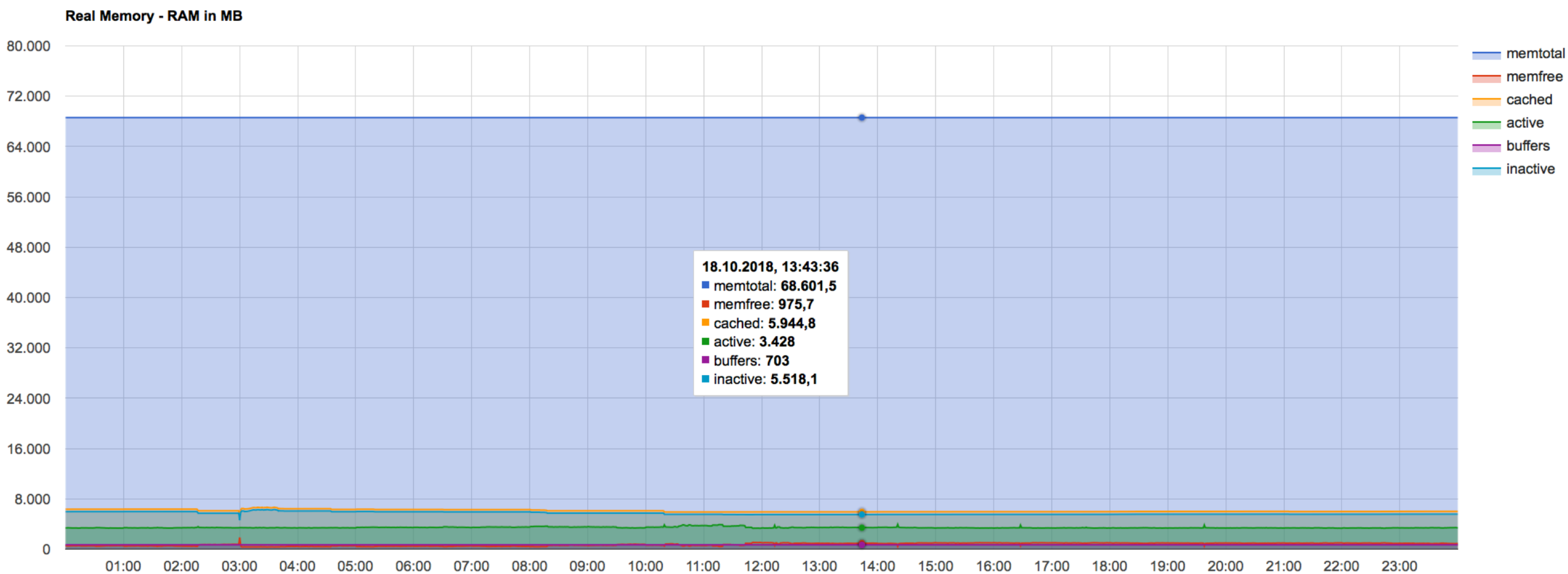
```
# ./smem -tkr
  PID User     Command                         Swap      USS      PSS      RSS
 1120 oracle   ora_mman_KLM12201                  0   172.8M   190.3M   269.1M
 1168 oracle   ora_mmon_KLM12201                  0    58.8M    80.9M   194.0M
 1142 oracle   ora_dbw0_KLM12201                  0    58.2M    75.1M   156.6M
 1184 oracle   oracleKLM12201 (DESCRIPTION        0    27.0M    47.1M   154.5M
 1206 oracle   ora_cjq0_KLM12201                  0    23.6M    36.1M   133.7M
 1100 oracle   sqlplus   as sysdba @/home/        0    19.0M    19.0M    21.6M
 1785 oracle   ora_w003_KLM12201                  0     5.9M    15.1M   105.7M
 1468 oracle   ora_q002_KLM12201                  0     8.9M    14.8M    94.3M
  607 root     /sbin/dhclient -d -q -sf /u        0    13.9M    14.5M    18.8M
```

RAM Usage

Never forget the Shared Memory
Use **ipcs** !

```
# LANG=C ipcs -m --human

------ Shared Memory Segments --------
key        shmid      owner      perms      size       nattch
0x00000000 0          oracle     600          8.2M     130
0x00000000 32769      oracle     600          2.3G     65
0x00000000 65538      oracle     600          7.8M     65
0x2a02cc10 98307      oracle     600           16K     65
```

History: nmon + nmonchart

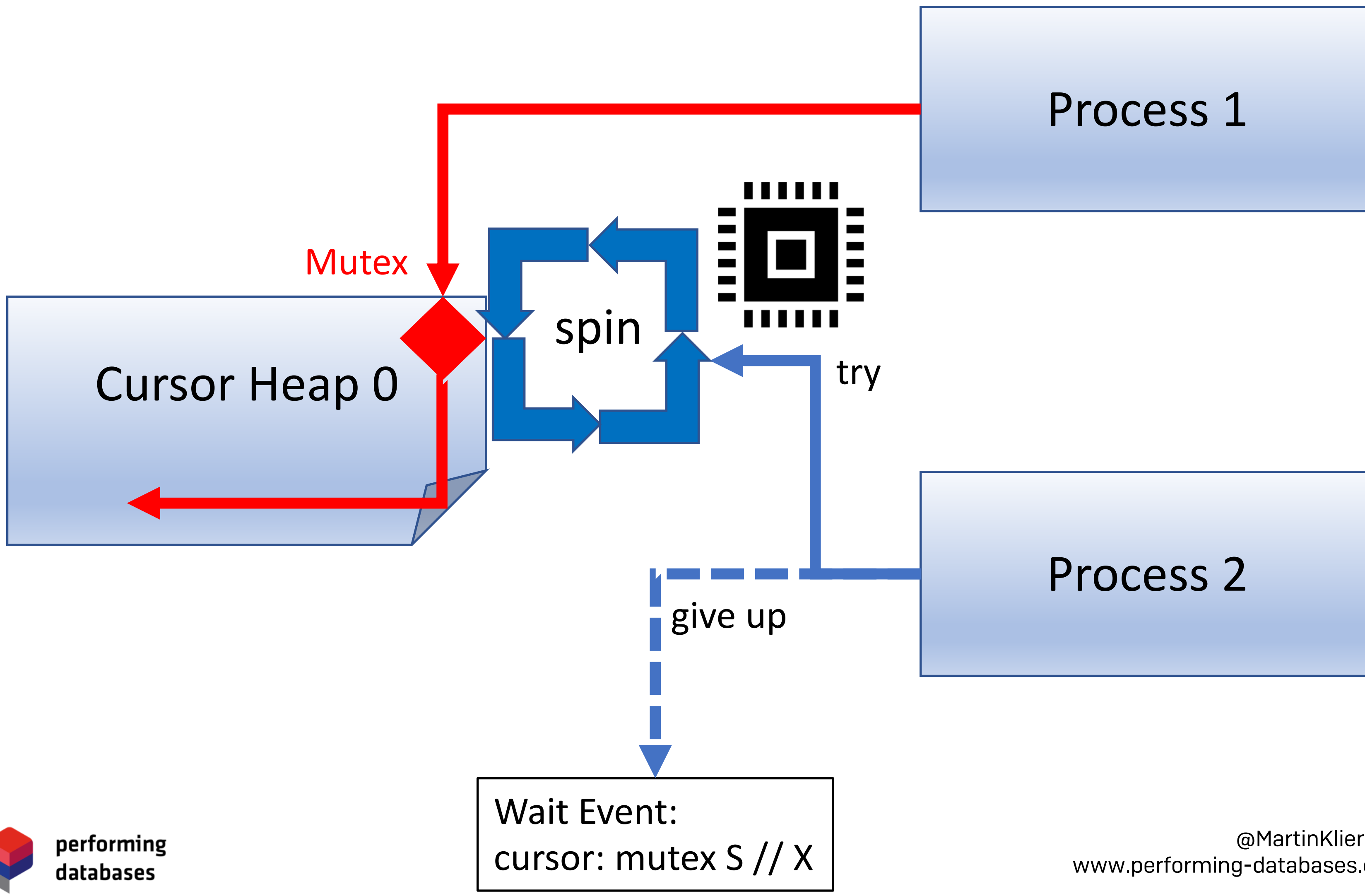


Only of limited use… :/

Finally, a bit more Oracle ☺

Wait Events

Waiting on LC Object



CPU Time in AWR

```
DB Name         DB Id    Instance     Inst Num Startup Time    Release     RAC
------------ ----------- ------------ -------- --------------- ----------- ---
                                             1 23-Feb-19 14:42 12.1.0.2.0  NO

Host Name        Platform                         CPUs Cores Sockets Memory(GB)
---------------- -------------------------------- ---- ----- ------- ----------
                 Linux x86 64-bit                   14    14       1      62.67

              Snap Id      Snap Time      Sessions Curs/Sess
            --------- ------------------- -------- ---------
Begin Snap:    144191 15-Mar-19 00:00:51       278       2.5
  End Snap:    144197 15-Mar-19 01:00:55       265       2.7
   Elapsed:               60.06 (mins)
   DB Time:              518.46 (mins)
```

⇒ Max. capacity would have been

⇒ 14 x 60 = 840 minutes on CPU

CPU Time in AWR

Load Profile	Per Second	Per Transaction	Per Exec	Per Call
DB Time(s):	8.6	0.0	0.00	0.00
DB CPU(s):	2.0	0.0	0.00	0.00
Background CPU(s):	0.1	0.0	0.00	0.00
Redo size (bytes):	1,466,109.5	5,010.1		
Logical read (blocks):	216,598.3	740.2		
Block changes:	9,843.1	33.6		
Physical read (blocks):	1,724.9	5.9		
Physical write (blocks):	468.7	1.6		
Read IO requests:	283.2	1.0		
Write IO requests:	220.8	0.8		
Read IO (MB):	13.5	0.1		
Write IO (MB):	3.7	0.0		
IM scan rows:	0.0	0.0		
Session Logical Read IM:				
User calls:	8,934.0	30.5		
Parses (SQL):	3,802.2	13.0		
Hard parses (SQL):	1.3	0.0		
SQL Work Area (MB):	18.3	0.1		
Logons:	0.3	0.0		
Executes (SQL):	4,132.2	14.1		
Rollbacks:	265.5	0.9		
Transactions:	292.6			

2 of 14 sec/sec
= 14%

CPU Time in AWR

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
log file sync	98,661	10.3K	104.00	33.0	Commit
db file sequential read	750,371	9608.5	12.80	30.9	User I/O
DB CPU		7290		23.4	
db file scattered read	119,721	2556	21.35	8.2	User I/O
enq: TX - index contention	6,129	392.1	63.97	1.3	Concurre
enq: TX - row lock contention	344	380.6	1106.43	1.2	Applicat
control file sequential read	69,471	267.6	3.85	.9	System I
direct path read temp	22,388	227	10.14	.7	User I/O
log file switch (private stran	263	192.6	732.15	.6	Configur
SQL*Net message from dblink	98,968	121	1.22	.4	Network

7290s = 121min of 840min = 14%

7209s = 121min of 518min = 23%

Causing CPU Load

- By instructions ☺

- By parsing
- By calculating
- By accessing buffer cache
- …

Means: Actually doing something useful
Always? No.

Eat.
Sleep.
Analyze Root Causes.

Hunting!

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
log file sync	98,661	10.3K	104.00	33.0	Commit
db file sequential read	750,371	9608.5	12.80	30.9	User I/O
DB CPU		7290		23.4	
db file scattered read	119,721	2556	21.35	8.2	User I/O
enq: TX - index contention	6,129	392.1	63.97	1.3	Concurre
enq: TX - row lock contention	344	380.6	1106.43	1.2	Applicat
control file sequential read	69,471	267.6	3.85	.9	System I
direct path read temp	22,388	227	10.14	.7	User I/O
log file switch (private stran	263	192.6	732.15	.6	Configur
SQL*Net message from dblink	98,968	121	1.22	.4	Network

CPU + I/O-bound

➔ Look for Buffer Gets!

Hunting BG

```
SQL ordered by Gets                 DB/Inst:               Snaps: 180193-180194
-> Resources reported for PL/SQL code includes the resources used by all SQL
   statements called by the code.
-> %Total - Buffer Gets   as a percentage of Total Buffer Gets
-> %CPU   - CPU Time      as a percentage of Elapsed Time
-> %IO    - User I/O Time as a percentage of Elapsed Time
-> Total Buffer Gets:      78,400,791
-> Captured SQL account for   78.6% of Total

     Buffer                 Gets              Elapsed
      Gets   Executions   per Exec   %Total  Time (s)  %CPU   %IO    SQL Id
----------- ----------- ------------ ------ ---------- ----- ----- -------------
 17,249,708       9,831      1,754.6   22.0       32.4  89.4     0 5uydkx53jkps5
Module: WMServer@
SELECT pickTuId, pickTuIdParent, pickScheduleId, tuId, pickTuType, tuTypeId, pri
ority, status, reserved, mailingIdPrj, flagAssortmentPrj, flagPalletListPrintedP
rj, customerKeyPrj, flagPalletizedPrj, laneProgressPrj, dnNoSettingPrj, dnNoPrj,
 priorityGroupPrj, polybagStatusPrj, manPalSequencePrjId, genDat, genInfo, modDa
```

Buffers

Hunting BG

Id	Operation	Name	Starts	E-Rows	E-Bytes	Cost (%CPU)	E-Time	A-Rows	A-Time	Buffers
0	SELECT STATEMENT		1			1832 (100)		70	00:00:04.48	613K
1	SORT ORDER BY		1	5	2590	1832 (1)	00:00:22	70	00:00:04.48	613K
2	VIEW	VIEWRTPICKING	1	5	2590	1831 (1)	00:00:22	70	00:00:04.48	613K
3	SORT UNIQUE		1	5	1376	1831 (75)	00:00:22	70	00:00:04.48	613K
4	UNION-ALL		1					70	00:00:04.48	613K
* 5	TABLE ACCESS BY INDEX ROWID	PICKORDERPOS	70	1	9	3 (0)	00:00:01	70	00:00:00.01	120
* 6	INDEX UNIQUE SCAN	XPKPICKORDERPOS	70	1		2 (0)	00:00:01	70	00:00:00.01	50
7	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	24	2 (0)	00:00:01	35	00:00:00.01	73
* 8	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
9	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	24	2 (0)	00:00:01	35	00:00:00.01	73
* 10	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
11	NESTED LOOPS		35					35	00:00:00.01	77
12	NESTED LOOPS		35	1	16	3 (0)	00:00:01	35	00:00:00.01	42
* 13	INDEX RANGE SCAN	I_STORAGELOCATION_TUNING_1	35	1	9	2 (0)	00:00:01	35	00:00:00.01	38
* 14	INDEX UNIQUE SCAN	XPKWAREHOUSE	35	1		0 (0)		35	00:00:00.01	4
15	TABLE ACCESS BY INDEX ROWID	WAREHOUSE	35	43	301	1 (0)	00:00:01	35	00:00:00.01	35
16	NESTED LOOPS		35	1	25	3 (0)	00:00:01	35	00:00:00.01	117
17	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	13	2 (0)	00:00:01	35	00:00:00.01	73
* 18	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
19	TABLE ACCESS BY INDEX ROWID	RACK	35	11276	132K	1 (0)	00:00:01	35	00:00:00.01	44
* 20	INDEX UNIQUE SCAN	XPKRACK	35	1		0 (0)		35	00:00:00.01	9
21	NESTED LOOPS		35	1	25	3 (0)	00:00:01	35	00:00:00.01	117
22	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	13	2 (0)	00:00:01	35	00:00:00.01	73
* 23	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
24	TABLE ACCESS BY INDEX ROWID	RACK	35	11276	132K	1 (0)	00:00:01	35	00:00:00.01	44
* 25	INDEX UNIQUE SCAN	XPKRACK	35	1		0 (0)		35	00:00:00.01	9
26	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	33	1	9	2 (0)	00:00:01	33	00:00:00.01	69
* 27	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	33	1		1 (0)	00:00:01	33	00:00:00.01	36
28	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	2	1	9	2 (0)	00:00:01	2	00:00:00.01	6
* 29	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	2	1		1 (0)	00:00:01	2	00:00:00.01	4
30	SORT AGGREGATE		1	1	8			1	00:00:00.01	376
* 31	INDEX FAST FULL SCAN	XAK1STORAGELOCATION	1	20257	158K	80 (2)	00:00:01	20773	00:00:00.01	376
32	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	9	2 (0)	00:00:01	35	00:00:00.01	73

performing databases

Hunting BG

Buffers

Id	Operation	Name	Starts	E-Rows	E-Bytes	Cost (%CPU)	E-Time	A-Rows	A-Time	Buffers
0	SELECT STATEMENT		1			1832 (100)		70	00:00:04.48	613K
1	SORT ORDER BY		1	5	2590	1832 (1)	00:00:22	70	00:00:04.48	613K
2	VIEW	VIEWRTPICKING	1	5	2590	1831 (1)	00:00:22	70	00:00:04.48	613K
3	SORT UNIQUE		1	5	1376	1831 (75)	00:00:22	70	00:00:04.48	613K
4	UNION-ALL		1					70	00:00:04.48	613K
* 5	TABLE ACCESS BY INDEX ROWID	PICKORDERPOS	70	1	9	3 (0)	00:00:01	70	00:00:00.01	120
* 6	INDEX UNIQUE SCAN	XPKPICKORDERPOS	70	1		2 (0)	00:00:01	70	00:00:00.01	50
7	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	24	2 (0)	00:00:01	35	00:00:00.01	73
* 8	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
9	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	24	2 (0)	00:00:01	35	00:00:00.01	73
* 10	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
11	NESTED LOOPS		35					35	00:00:00.01	77
12	NESTED LOOPS		35	1	16	3 (0)	00:00:01	35	00:00:00.01	42
* 13	INDEX RANGE SCAN	I_STORAGELOCATION_TUNING_1	35	1	9	2 (0)	00:00:01	35	00:00:00.01	38
* 14	INDEX UNIQUE SCAN	XPKWAREHOUSE	35	1		0 (0)		35	00:00:00.01	4
15	TABLE ACCESS BY INDEX ROWID	WAREHOUSE	35	43	301	1 (0)	00:00:01	35	00:00:00.01	35
16	NESTED LOOPS		35	1	25	3 (0)	00:00:01	35	00:00:00.01	117
17	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	13	2 (0)	00:00:01	35	00:00:00.01	73
* 18	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
19	TABLE ACCESS BY INDEX ROWID	RACK	35	11276	132K	1 (0)	00:00:01	35	00:00:00.01	44
* 20	INDEX UNIQUE SCAN	XPKRACK	35	1		0 (0)		35	00:00:00.01	9
21	NESTED LOOPS		35	1	25	3 (0)	00:00:01	35	00:00:00.01	117
22	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	13	2 (0)	00:00:01	35	00:00:00.01	73
* 23	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
24	TABLE ACCESS BY INDEX ROWID	RACK	35	11276	132K	1 (0)	00:00:01	35	00:00:00.01	44
* 25	INDEX UNIQUE SCAN	XPKRACK	35	1		0 (0)		35	00:00:00.01	9
26	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	33	1	9	2 (0)	00:00:01	33	00:00:00.01	69
* 27	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	33	1		1 (0)	00:00:01	33	00:00:00.01	36
28	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	2	1	9	2 (0)	00:00:01	2	00:00:00.01	6
* 29	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	2	1		1 (0)	00:00:01	2	00:00:00.01	4
30	SORT AGGREGATE		1	1	8			1	00:00:00.01	376
* 31	INDEX FAST FULL SCAN	XAK1STORAGELOCATION	1	20257	158K	80 (2)	00:00:01	20773	00:00:00.01	376
32	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	9	2 (0)	00:00:01	35	00:00:00.01	73
* 33	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
34	HASH UNIQUE		36	1	8	145K (1)	00:29:09	36	00:00:04.48	610K
* 35	COUNT STOPKEY		36					36	00:00:04.45	610K
* 36	FILTER		36					36	00:00:04.45	610K
37	TABLE ACCESS FULL	PACKINGUNIT	36	28340	221K	210 (1)	00:00:03	556K	00:00:00.01	15311
38	NESTED LOOPS		556K					36	00:00:03.42	595K
39	NESTED LOOPS		556K	1	32	6 (0)	00:00:01	87	00:00:03.17	595K
40	TABLE ACCESS BY INDEX ROWID	LU	556K	1	19	4 (0)	00:00:01	87	00:00:03.00	595K
* 41	INDEX RANGE SCAN	I_TUNING_LU_3	556K	1		3 (0)	00:00:01	87	00:00:02.90	595K
* 42	INDEX UNIQUE SCAN	XPKTU	87	1		1 (0)	00:00:01	87	00:00:00.01	214
* 43	TABLE ACCESS BY INDEX ROWID	TU	87	1	13	2 (0)	00:00:01	36	00:00:00.01	87
44	TABLE ACCESS BY INDEX ROWID	STORAGELOCATION	35	1	8	2 (0)	00:00:01	35	00:00:00.01	73
* 45	INDEX UNIQUE SCAN	XPKSTORAGELOCATION	35	1		1 (0)	00:00:01	35	00:00:00.01	38
46	NESTED LOOPS		1					70	00:00:00.01	1135
47	NESTED LOOPS		1	1	272	473 (1)	00:00:06	140	00:00:00.01	1088
48	NESTED LOOPS OUTER		1	1	263	469 (1)	00:00:06	70	00:00:00.01	1020
49	NESTED LOOPS		1	1	256	467 (1)	00:00:06	70	00:00:00.01	912
50	NESTED LOOPS		1	1	245	466 (1)	00:00:06	70	00:00:00.01	812
51	NESTED LOOPS		1	1	230	463 (0)	00:00:06	70	00:00:00.01	664
52	NESTED LOOPS		1	1	189	462 (0)	00:00:06	70	00:00:00.01	564
53	NESTED LOOPS		1	1	182	461 (0)	00:00:06	70	00:00:00.01	388
54	NESTED LOOPS		1	1	139	460 (0)	00:00:06	70	00:00:00.01	245
55	NESTED LOOPS		1	1	113	458 (0)	00:00:06	70	00:00:00.01	97
56	NESTED LOOPS		1	41	3116	344 (0)	00:00:05	6	00:00:00.01	52
57	NESTED LOOPS		1	41	2747	303 (0)	00:00:04	6	00:00:00.01	38
58	NESTED LOOPS		1	193	9457	27 (0)	00:00:01	6	00:00:00.01	26
59	NESTED LOOPS		1	5	195	4 (0)	00:00:01	6	00:00:00.01	11
60	NESTED LOOPS		1	1	22	2 (0)	00:00:01	1	00:00:00.01	4
* 61	INDEX UNIQUE SCAN	XPKWAREHOUSE	1	1	4	0 (0)		1	00:00:00.01	1
* 62	TABLE ACCESS BY INDEX ROWID	RACK	1	1	18	2 (0)	00:00:01	1	00:00:00.01	3

613k

610k
595k

Hunting BG

Buffers

```
|* 33 |            INDEX UNIQUE SCAN          | XPKSTORAGELOCATION |   35 |     1 |      |    1  (0)| 00:00:01 |   35 |00:00:00.01 |    38 |
|  34 |        HASH UNIQUE                    |                    |   36 |     1 |    8 | 145K  (1)| 00:29:09 |   36 |00:00:04.48 |  610K|
|* 35 |         COUNT STOPKEY                 |                    |   36 |       |      |          |          |   36 |[illegible] |  610K|
|* 36 |          FILTER                       |                    |   36 |       |      |          |          |   36 |00:00:04.45 |  610K|
|  37 |           TABLE ACCESS FULL           | PACKINGUNIT        |   36 | 28340 | 221K|  210  (1)| 00:00:03 | 556K|00:00:00.01 | 15311 |
|  38 |           NESTED LOOPS                |                    | 556K|       |      |          |          |   36 |00:00:03.42 |  595K|
|  39 |            NESTED LOOPS               |                    | 556K|     1 |   32 |    6  (0)| 00:00:01 |   87 |00:00:03.17 |  595K|
|  40 |             TABLE ACCESS BY INDEX ROWID| LU                | 556K|     1 |   19 |    4  (0)| 00:00:01 |   87 |00:00:03.00 |  595K|
|* 41 |              INDEX RANGE SCAN         | I_TUNING_LU_3      | 556K|     1 |      |    3  (0)| 00:00:01 |   87 |00:00:02.90 |  595K|
|* 42 |             INDEX UNIQUE SCAN         | XPKTU              |   87 |     1 |      |    1  (0)| 00:00:01 |   87 |00:00:00.01 | [illegible] |
|* 43 |            TABLE ACCESS BY INDEX ROWID| TU                 |   87 |     1 |   13 |    2  (0)| 00:00:01 |   36 |00:00:00.01 |    87 |
```

595k

```
|* 33 |            INDEX UNIQUE SCAN            | XPKSTORAGELOCATION
|  34 |        HASH UNIQUE                      |
|* 35 |         COUNT STOPKEY                   |
|* 36 |          FILTER                         |
|  37 |           TABLE ACCESS FULL             | PACKINGUNIT
|  38 |           NESTED LOOPS                  |
|  39 |            NESTED LOOPS                 |
|  40 |             TABLE ACCESS BY INDEX ROWID | LU
|* 41 |              INDEX RANGE SCAN           | I_TUNING_LU_3
|* 42 |             INDEX UNIQUE SCAN           | XPKTU
|* 43 |            TABLE ACCESS BY INDEX ROWID  | TU
```

Hunting BG

```
|* 33 |             INDEX UNIQUE SCAN                       | XPKSTORAGELOCATION
|  34 |       HASH UNIQUE                                   |
|* 35 |        COUNT STOPKEY                                |
|* 36 |         FILTER                                      |
|  37 |          TABLE ACCESS FULL                          | PACKINGUNIT
|  38 |          NESTED LOOPS                               |
|  39 |           NESTED LOOPS                              |
|  40 |            TABLE ACCESS BY INDEX ROWID              | LU
|* 41 |             INDEX RANGE SCAN                        | I_TUNING_LU_3
|* 42 |            INDEX UNIQUE SCAN                        | XPKTU
|* 43 |           TABLE ACCESS BY INDEX ROWID               | TU
```

```
  35 - filter(ROWNUM=1)
  36 - filter( IS NOT NULL)
  41 - access("LU"."SKUID"=:B1 AND "LU"."PACKINGUNITID"=:B2 AND "LU"."STATUS"<99)
       filter("LU"."PACKINGUNITID"=:B1)
  42 - access("LU"."TUID"="TU2"."TUID")
  43 - filter(("TU2"."STORAGELOCATIONID"=:B1 AND "TU2"."STATUS"<99))
```

performing databases

Craftsmen come!

What we did today

- A little performance theory
  Remember **Latency** and **Queueing**!

- Something about (Linux) resources – CPU and RAM
  Remember **Load Average**!

- How Oracle Wait Events are generated (some)
  Remember that **spinning** is expensive

- Hunting Buffer Gets!

Meet & Greet

martin.klier@performing-db.com

www.performing-databases.com

Many national // international events

Deutsche ORACLE-Anwendergruppe e.V.

Conference&Exhibition 2020!

utah oracle users group

Efficiency is

to stay in touch!

martin.klier@performing-db.com

http://www.performing-databases.com

performing databases

Load Average:
http://www.brendangregg.com/blog/2017-08-08/linux-load-averages.html

IT Icons made by Eucalypt from www.flaticon.com
Car Icon by creaticca-creative-agency from www.flaticon.com

performing
databases