When it all Goes Wrong

Nordic PGDay 2019 — March 19 — Copenhagen

Will Leinweber

@leinweber

Citus Data (Microsoft)

bitfission.com

(warning autoplays midi)

coming from

citus cloud

heroku postgres

special thanks

citus cloud
— dan farina (@danfarina)

heroku postgres
— maciek sakrejda (@uhoh_itsmaciek)

same sorts of problems

from pages & alerts

from support tickets

this talk

more app dev who uses postgres
rather than dba

@leinweber

the problem with Postgres

it’s pretty good

you don’t get experience with how it breaks

@leinweber

what to do for a problem

7 Answers

active oldest votes

5432

Please run the following commands:

- `ps aux | grep postgres` and find the PID for the postmaster process
- `kill -9 <pid>` to shutdown postgres

You may then have to go find the postmaster.pid file and delete it too

If you're still having problems try looking at these things:

`\connect database_name` or `\c database_name`

@leinweber

what to do for a problem

Re: TIP 4: Don't 'kill -9' the postmaster

From: Tom Lane <tgl(at)sss(dot)pgh(dot)pa(dot)us>
To: Doug McNaught <doug(at)wireboard(dot)com>
Cc: Jeff Davis <list-pgsql-general(at)dynworks(dot)com>, pgsql-general(at)postgresql(
Subject: Re: TIP 4: Don't 'kill -9' the postmaster
Date: 2002-02-08 16:02:48
Message-ID: 23773.1013184168@sss.pgh.pa.us
Views: Raw Message | Whole Thread | Download mbox
Thread: 2002-02-08 16:02:48 from Tom Lane <tgl(at)sss(dot)pgh(dot)pa(dot)us>
Lists: pgsql-general

```
Doug McNaught <doug(at)wireboard(dot)com> writes:
> The tip is directed at those people for whom 'kill -9' is the first
> resort, not the last.  ;) Clean shutdown is *always* better than
> unclean if you can manage it.

Agreed.  But actually, the tip dates from several versions back, when
kill -9 was indeed dangerous.

Back then, if you killed the postmaster without letting it kill all its
child processes, it was possible to start a new postmaster (and then
have it launch new children) while old backends still remained running.
The old and new backends wouldn't know about each other, leading to
disaster if any conflicting updates were made.

There are now interlocks to prevent this scenario: a new postmaster will
look for extant backends in the same database, and refuse to start if it
finds any.  So I believe that you cannot shoot yourself in the foot
that way anymore.  (Digression: the ability to make this check is one of
the few good things about the SysV shared-memory interface.)

As of 7.1 or so, I think the tip could be rephrased as "kill -9 is not
the preferred way of shutting down the database" ;-)

                        regards, tom lane
```

complicated system

network

hardware

o/s

postgres

using the database (too much)

95% application

4% auto vacuum

1% everything else

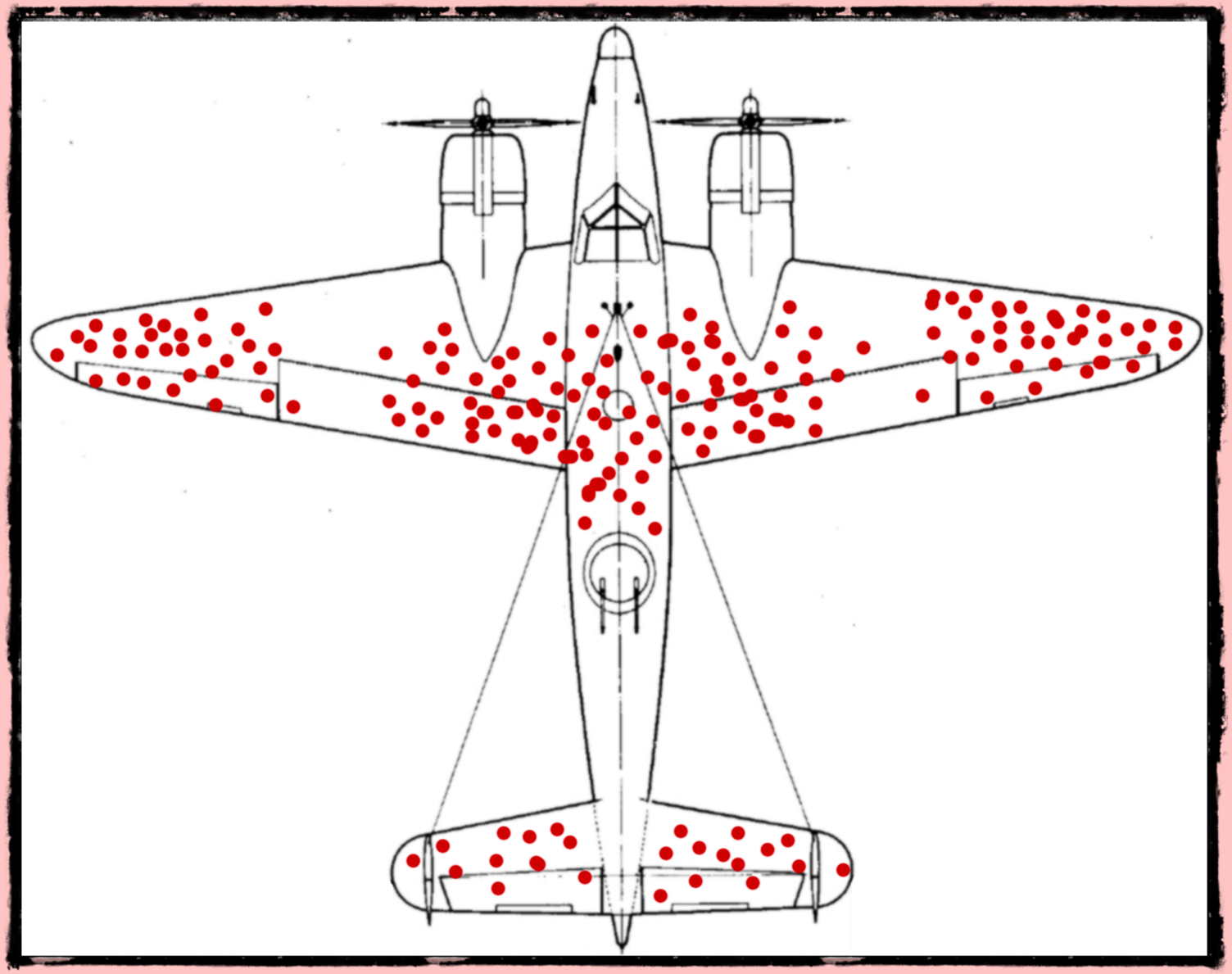
hard to convince

all the graphs saying DB is slow

and nothing has changed

…must be the database!

@leinweber



https://upload.wikimedia.org/wikipedia/commons/9/98/Survivorship-bias.png

“but I didn’t change anything”

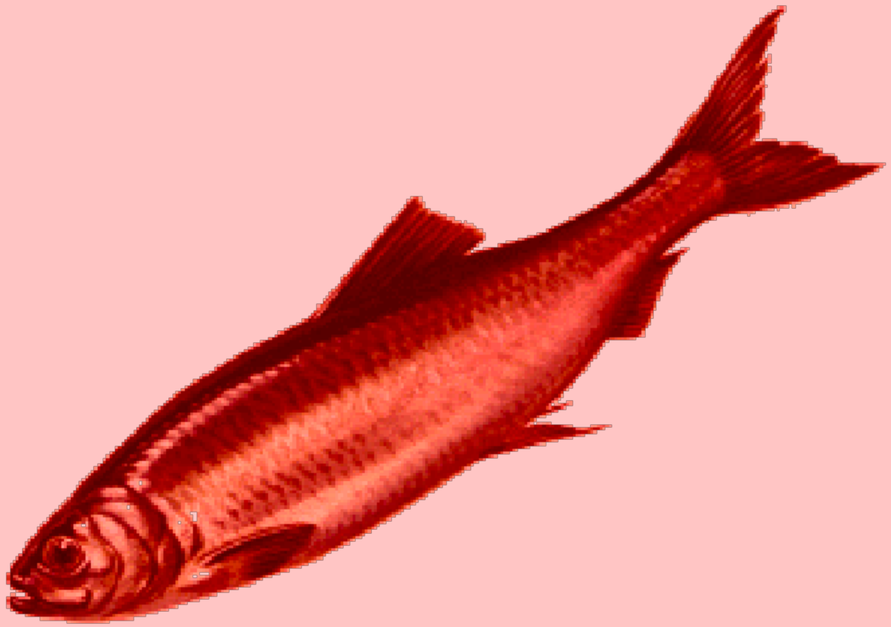
no deploys!

no database migrations!

no scaling!

@leinweber

“but I didn’t change anything”



https://upload.wikimedia.org/wikipedia/commons/0/09/Redherring.gif

“but I didn’t change anything”

more traffic?

change in access patterns?

one big user logged in?

@leinweber

run out of a resource

@leinweber

snowball

example

manageable user 1s query => 2x expensive

frequent, small queries 3ms => 12ms

@leinweber

assumptions

assumptions

postgres should not crash

…with overcommit off

large extensions increase chance

@leinweber

if not postgres, then what

system resources

cpu

memory

disk

parallelism / backends

locks

@leinweber

cpu mem disk parallelism

@leinweber

cpu mem disk parallelism

credentials wrong

networking broken

locking issue, check pg_locks

idle in transaction

cpu mem disk **parallelism**

application submitting backlogged workload

connection leak

pool sizes set too large

pg_lock issue + application backlog

cpu mem **disk** parallelism

workload skew causing thrashing

unusual sequential scan workload

failover or restart => no cache

pg_prewarm

cpu mem **disk** **parallelism**

same as just disk,

but also the application is piling on

cpu **mem** disk parallelism

large `GROUP BY`s

high disk latency due to unusual page dispersion pattern in the workload

cpu **mem** disk **parallelism**

workload has high mem (`GROUP BY`)

+ app adding backlog

lock contention slowing mem release

@leinweber

cpu **mem** **disk** parallelism

large `GROUP BY`s + paging in unusual data

cpu **mem** **disk** **parallelism**

Look for what is causing disk access

@leinweber

cpu mem disk parallelism

small, in-memory workload

- lots of seq scans on small table
- index scan w/ filter dropping lots

@leinweber

cpu mem disk **parallelism**

app backlog

+ too much processing on small data

simply a lot of work

@leinweber

cpu mem **disk** parallelism

large seq scans

@leinweber

cpu mem **disk** **parallelism**

loading cold data + application backlog

@leinweber

cpu **mem** disk parallelism

small # of backends doing a lot more work

@leinweber

cpu **mem** disk **parallelism**

entity, workload, entity*workload

soft deletes and non-conditional indexes

@leinweber

cpu **mem** **disk** parallelism

reporting query

cpu mem disk parallelism

app backlog, but with CPU/mem problems

tools of the trade

tools of the trade

C symbols

@leinweber

tools of the trade: perf

```
perf record -p <pid> && perf report
```

```
Samples: 4K of event 'cpu-clock', Event count (approx.): 1193000000
Overhead  Command     Shared Object           Symbol
   6.94%  postmaster  [kernel.kallsyms]       [k] __lock_text_start
   2.58%  postmaster  postgres                [.] base_yyparse
   2.37%  postmaster  postgres                [.] AllocSetAlloc
   2.07%  postmaster  postgres                [.] SearchCatCache
   1.95%  postmaster  libc-2.17.so            [.] __memcpy_ssse3_back
   1.66%  postmaster  [kernel.kallsyms]       [k] do_syscall_64
   1.55%  postmaster  postgres                [.] core_yylex
   1.53%  postmaster  libc-2.17.so            [.] __strcmp_sse42
   1.45%  postmaster  libc-2.17.so            [.] _int_malloc
   1.38%  postmaster  postgres                [.] hash_search_with_hash_value
   1.28%  postmaster  [kernel.kallsyms]       [k] finish_task_switch
   1.28%  postmaster  libc-2.17.so            [.] vfprintf
   1.28%  postmaster  postgres                [.] hash_seq_search
   0.90%  postmaster  libc-2.17.so            [.] __strlen_sse2_pminub
   0.78%  postmaster  postgres                [.] palloc
   0.69%  postmaster  postgres                [.] MemoryContextAllocZeroAligned
   0.65%  postmaster  postgres                [.] copyObject
   0.65%  postmaster  postgres                [.] expression_tree_walker.part.3
   0.63%  postmaster  [kernel.kallsyms]       [k] ep_send_events_proc
   0.63%  postmaster  libc-2.17.so            [.] _int_free
   0.63%  postmaster  postgres                [.] ScanKeywordLookup
Tip: To record every process run by a user: perf record -u <user>
```

@leinweber

tools of the trade: perf

`perf top`

```
Samples: 8K of event 'cpu-clock', Event count (approx.): 1421938644
Overhead  Shared Object                     Symbol
   7.72%  [kernel]                          [k] __lock_text_start
   4.06%  [kernel]                          [k] finish_task_switch
   3.79%  [kernel]                          [k] __softirqentry_text_start
   1.62%  postgres                          [.] AllocSetAlloc
   1.57%  postgres                          [.] SearchCatCache
   1.57%  postgres                          [.] base_yyparse
   1.47%  [kernel]                          [k] do_syscall_64
   1.37%  postgres                          [.] hash_search_with_hash_value
   1.35%  libc-2.17.so                      [.] __memcpy_ssse3_back
   0.96%  libc-2.17.so                      [.] __strlen_sse2_pminub
   0.94%  libc-2.17.so                      [.] __strcmp_sse42
   0.88%  postgres                          [.] core_yylex
   0.84%  libc-2.17.so                      [.] vfprintf
   0.74%  postgres                          [.] hash_seq_search
   0.66%  libc-2.17.so                      [.] _int_malloc
   0.63%  [kernel]                          [k] ena_io_poll
   0.52%  [kernel]                          [k] _raw_spin_lock
   0.51%  [kernel]                          [k] ipt_do_table
   0.48%  [kernel]                          [k] ep_send_events_proc
   0.46%  postgres                          [.] AtEOXact_GUC
   0.46%  [kernel]                          [k] tcp_ack
```

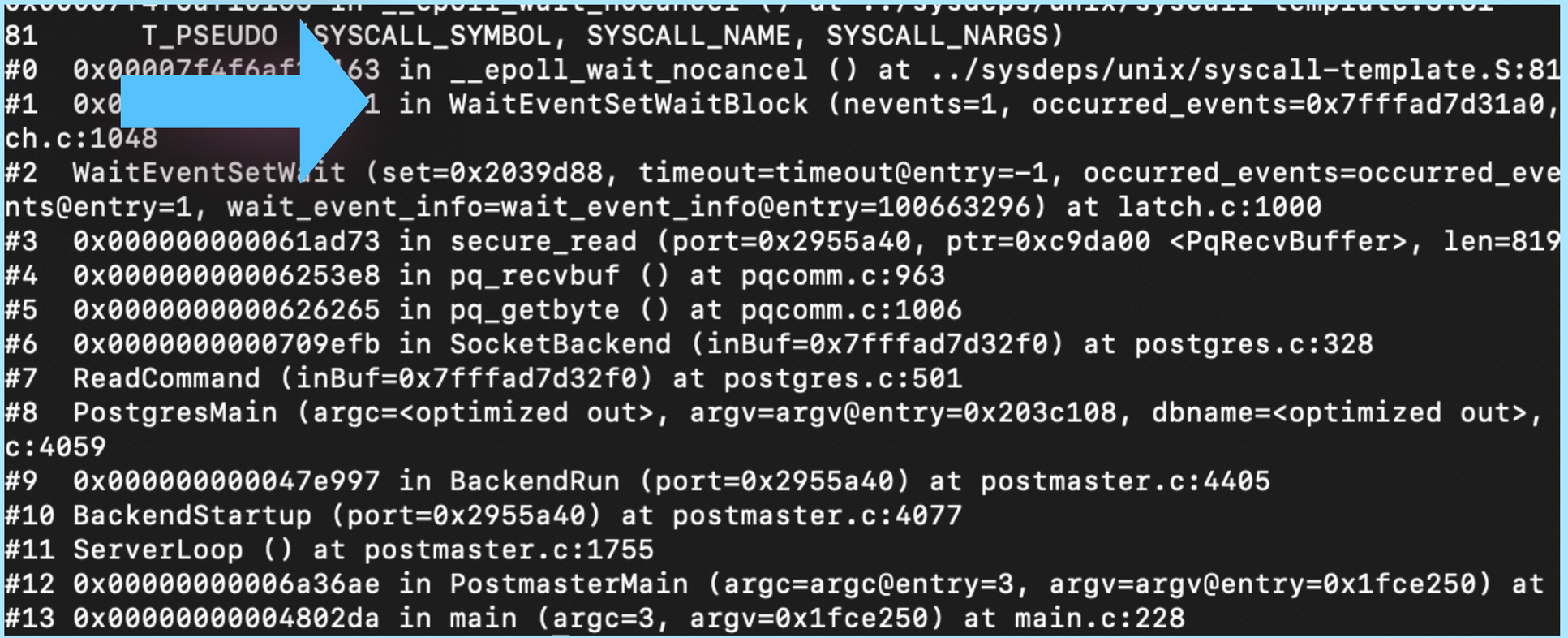
tools of the trade: perf

www.brendangregg.com/perf.html

tools of the trade: gdb

```
gdb -batch -ex 'bt' -p <pid>
```

@leinweber


81 T_PSEUDO (SYSCALL_SYMBOL, SYSCALL_NAME, SYSCALL_NARGS)
#0 0x00007f4f6af[illegible]163 in __epoll_wait_nocancel () at ../sysdeps/unix/syscall-template.S:81
#1 0x0[illegible]1 in WaitEventSetWaitBlock (nevents=1, occurred_events=0x7fffad7d31a0,
ch.c:1048
#2 WaitEventSetWait (set=0x2039d88, timeout=timeout@entry=-1, occurred_events=occurred_eve
nts@entry=1, wait_event_info=wait_event_info@entry=100663296) at latch.c:1000
#3 0x000000000061ad73 in secure_read (port=0x2955a40, ptr=0xc9da00 <PqRecvBuffer>, len=819
#4 0x00000000006253e8 in pq_recvbuf () at pqcomm.c:963
#5 0x0000000000626265 in pq_getbyte () at pqcomm.c:1006
#6 0x0000000000709efb in SocketBackend (inBuf=0x7fffad7d32f0) at postgres.c:328
#7 ReadCommand (inBuf=0x7fffad7d32f0) at postgres.c:501
#8 PostgresMain (argc=<optimized out>, argv=argv@entry=0x203c108, dbname=<optimized out>,
c:4059
#9 0x000000000047e997 in BackendRun (port=0x2955a40) at postmaster.c:4405
#10 BackendStartup (port=0x2955a40) at postmaster.c:4077
#11 ServerLoop () at postmaster.c:1755
#12 0x00000000006a36ae in PostmasterMain (argc=argc@entry=3, argv=argv@entry=0x1fce250) at
#13 0x00000000004802da in main (argc=3, argv=0x1fce250) at main.c:228

@leinweber


#4 XLogInsert (rmid=rmid@entry=0 '\000', info=info@entry=176 '\260') at xloginsert.c:
#5 0x00000000005026c7 in log_newpage (rnode=0x7f4f6d4ce808, forkNum=forkNum@entry=MAI
e@entry=0x2a00488 "" page_std=page_std@entry=1 '\001') at xloginsert.c:984
#6 0x000 in _bt_blwritepage (wstate=0x7fffad7d24c0, page=0x2a00488 "", b
#7 0x000 in _bt_buildadd (wstate=wstate@entry=0x7fffad7d24c0, state=stat
8) at nbtsort.c:576
#8 0x00000000004cfafe in _bt_load (btspool=0x2969c40, btspool2=0x0, wstate=0x7fffad7d
#9 _bt_leafbuild (btspool=0x2969c40, btspool2=0x0) at nbtsort.c:231
#10 0x00000000004c99 in btbuild (heap=0x7f4f6d4c1d30, index=0x7f4f6d4ce808, indexInf
#11 0x000 in index_build (heapRelation=heapRelation@entry=0x7f4f6d4c1d30,
4ce808, Info@entry=0x203fe98, isprimary=isprimary@entry=0 '\000', isrei
052
#12 0x00000000005113ed in index_create (heapRelation=heapRelation@entry=0x7f4f6d4c1d30
=0x203fce8 "foo_i_idx1", indexRelationId=459028, indexRelationId@entry=0, relFileNode=
0x203fe98, indexColNames=indexColNames@entry=0x203fca0, accessMethodObjectId=<optimize
tionObjectId=<optimized out>, classObjectId=<optimized out>, coloptions=<optimized out
ptimized out>, isconstraint=<optimized out>, deferrable=<optimized out>, initdeferred=
ptimized out>, skip_build=<optimized out>, concurrent=<optimized out>, is_internal=<op
) at index.c:1125
#13 0x00000000005a222f in DefineIndex (relationId=<optimized out>, relationId@entry=45
ionId=indexRelationId@entry=0, is_alter_table=is_alter_table@entry=0 '\000', check_rig
in_use=check_not_in_use@entry=1 '\001', skip_build=0 '\000', quiet=0 '\000') at indexc
#14 0x00000000070f537 in ProcessUtilitySlow (pstate=pstate@entry=0x29568a8, pstmt=pst

@leinweber

tools of the trade: iostat

iostat -xm 10

```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.26    0.04    0.19    0.04    0.00   99.47

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await  svctm  %util
nvme1n1           0.00     1.40    0.00    4.00     0.00     0.03    17.20     0.00    0.00   0.00   0.00
nvme2n1           0.00     0.00    0.00    1.70     0.00     0.02    19.76     0.00    0.00   0.00   0.00
nvme3n1           0.00     0.00    0.00    1.40     0.00     0.01    12.00     0.00    0.00   0.00   0.00
nvme4n1           0.00     0.20    0.00   11.60     0.00     0.08    14.76     0.00    0.28   0.00   0.00
nvme5n1           0.00     5.50    0.00    3.50     0.00     0.05    30.17     0.00    0.23   0.11   0.04
nvme6n1           0.00     0.00    0.00    1.00     0.00     0.01    20.00     0.00    0.00   0.00   0.00
nvme7n1           0.00     0.30    0.00    2.40     0.00     0.02    16.67     0.00    0.00   0.00   0.00
nvme8n1           0.00     0.00    0.00    0.90     0.00     0.01    29.33     0.00    0.00   0.00   0.00
nvme9n1           0.00     0.40    0.00    3.50     0.00     0.03    19.43     0.00    0.00   0.00   0.00
nvme0n1           0.00     8.00    0.00    5.70     0.00     0.06    19.79     0.00    0.56   0.00   0.00
dm-0              0.00     0.00    0.00   11.60     0.00     0.08    14.76     0.00    0.28   0.28   0.32
dm-1              0.00     0.00    0.00   23.90     0.00     0.19    15.87     0.00    0.05   0.02   0.04
```

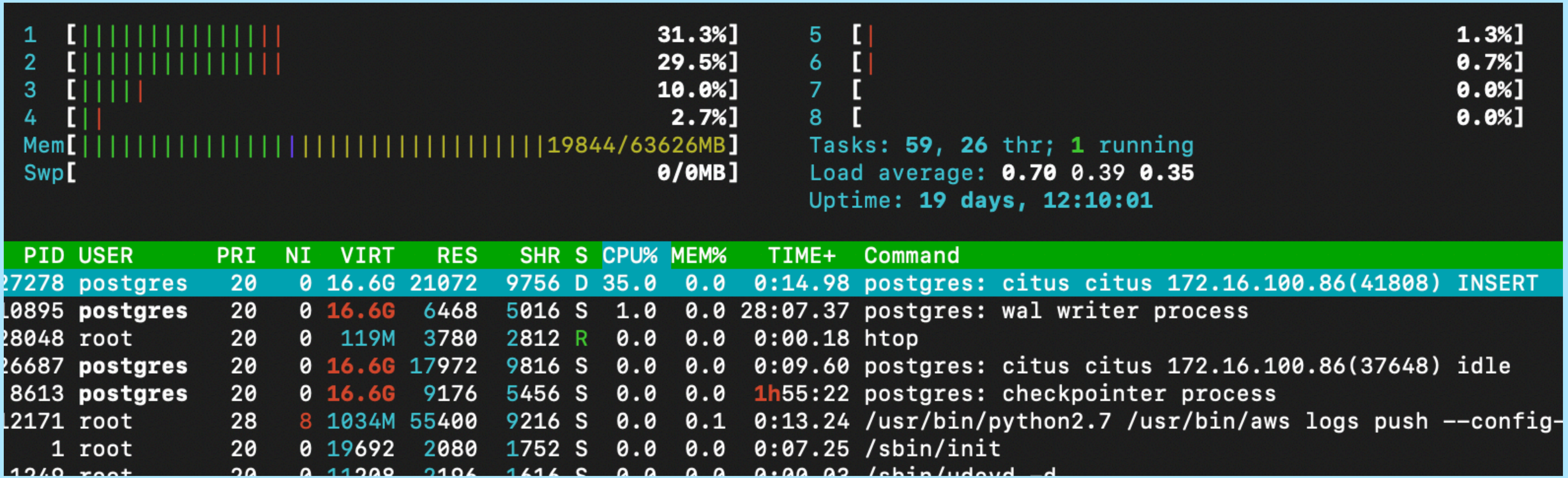
@leinweber

tools of the trade: iotop

```
Total DISK READ: 0.00 B/s | Total DISK WRITE: 836.77 K/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
24345 be/4 postgres    0.00 B/s  571.07 K/s  0.00 %  2.71 % postgres: citus citus 172.16.100.86(45232) INSERT
26513 be/4 postgres    0.00 B/s  261.74 K/s  0.00 %  1.19 % postgres: citus citus 172.16.100.86(54416) idle
 1199 be/3 root        0.00 B/s    0.00 B/s  0.00 %  0.18 % [jbd2/nvme0n1p1-]
12183 be/5 root        0.00 B/s   15.86 K/s  0.00 %  0.14 % python2.7 /usr/bin/aws logs push --al-configs-dir /e
10895 be/4 postgres    0.00 B/s   15.86 K/s  0.00 %  0.04 % postgres: wal writer process
 8444 be/4 postgres    0.00 B/s    3.97 K/s  0.00 %  0.00 % postgres: logger process
 8613 be/4 postgres    0.00 B/s  150.70 K/s  0.00 %  0.00 % postgres: checkpointer process
 8614 be/4 postgres    0.00 B/s    7.93 K/s  0.00 %  0.00 % postgres: writer process
 2560 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % supervising syslog-ng
    1 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % init
    2 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kthreadd]
 3243 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % sendmail: accepting connections
    4 be/0 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kworker/0:0H]
    6 be/0 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [mm_percpu_wq]
    7 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [ksoftirqd/0]
    8 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_sched]
    9 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_bh]
```

@leinweber

tools of the trade: htop



Tools of the trade: bwm-ng

```
bwm-ng v0.6 (probing every 10.000s), press 'h' for help
input: /proc/net/dev type: rate
/         iface                        Rx                   Tx                Total
==============================================================================
           eth0:           6.07 KB/s            3.26 KB/s            9.32 KB/s
             lo:          19.40 KB/s           19.40 KB/s           38.80 KB/s
           eth2:          29.75 KB/s            7.19 KB/s           36.95 KB/s
------------------------------------------------------------------------------
          total:          55.22 KB/s           29.85 KB/s           85.07 KB/s
```

tools of the trade: backends

```
pgrep -lf postgres + grep + wc
```

```
select * from pg_stat_activity
```

tools of the trade: pg_s_s

```
select * from pg_stat_statements
```

@leinweber

tools of the trade: summary

	cpu	mem	disk	parallelism	network
perf	x				
gdb	x				
iostat			x		
iotop			x		
htop	x	x			
bwm					x
pgrep				x	

what to do

what to do

configuration change

what to do

db change

what to do

code change

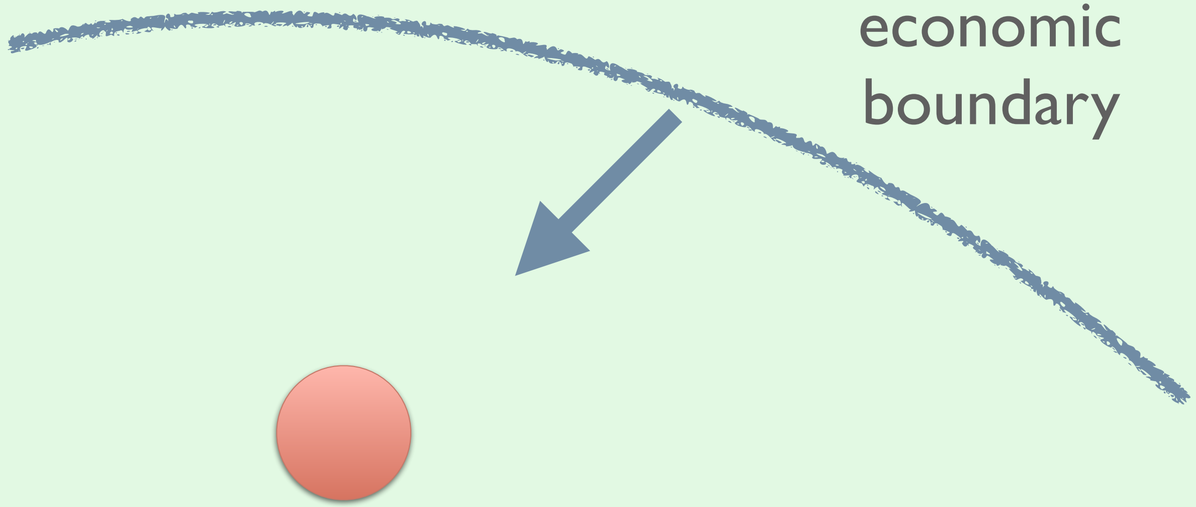
flirting with disaster

Velocity NY 2013: Richard Cook
"Resilience In Complex Adaptive Systems”

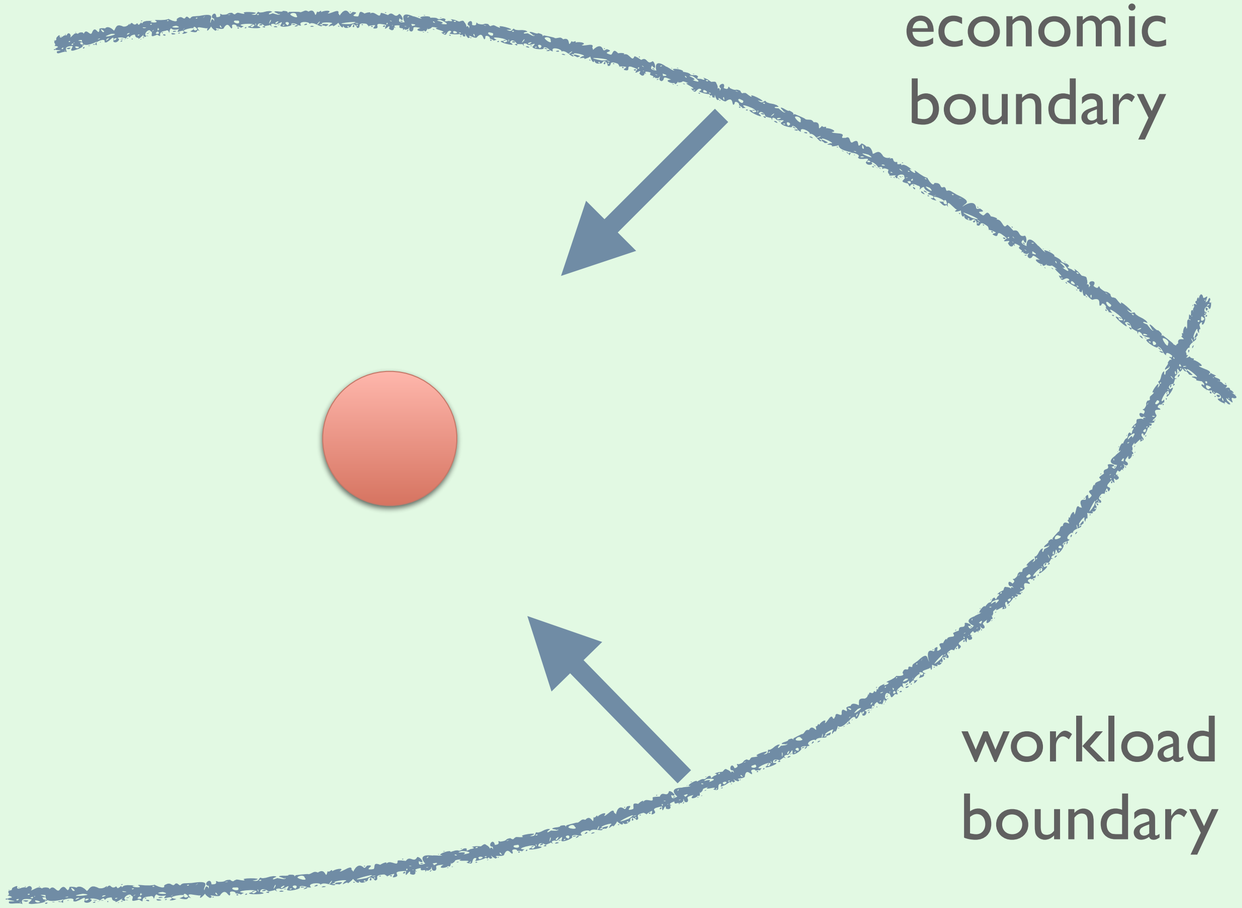
Jens Rasmussen:
Risk management in a dynamic society: a modeling problem

@leinweber

flirting with disaster

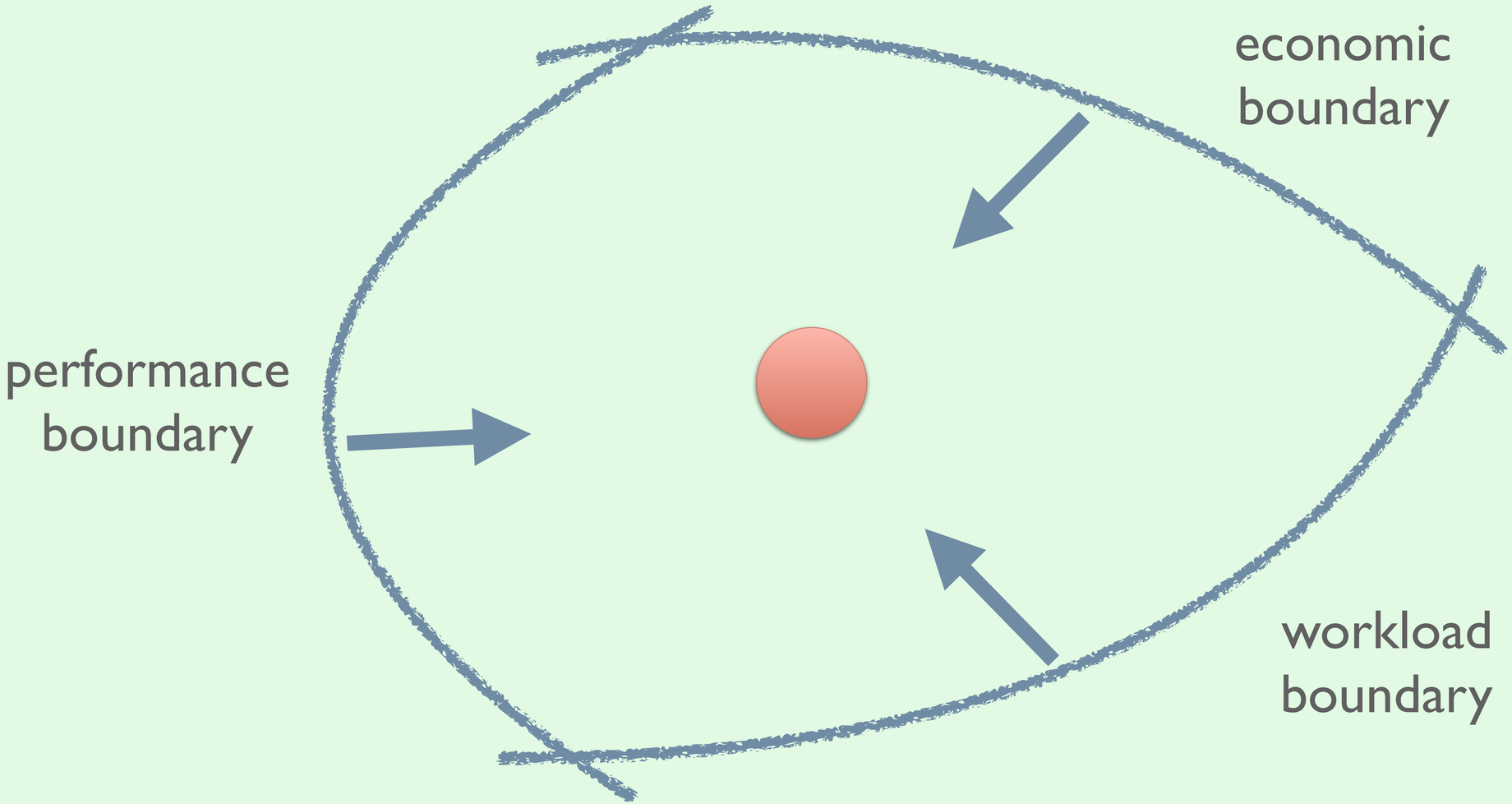


flirting with disaster



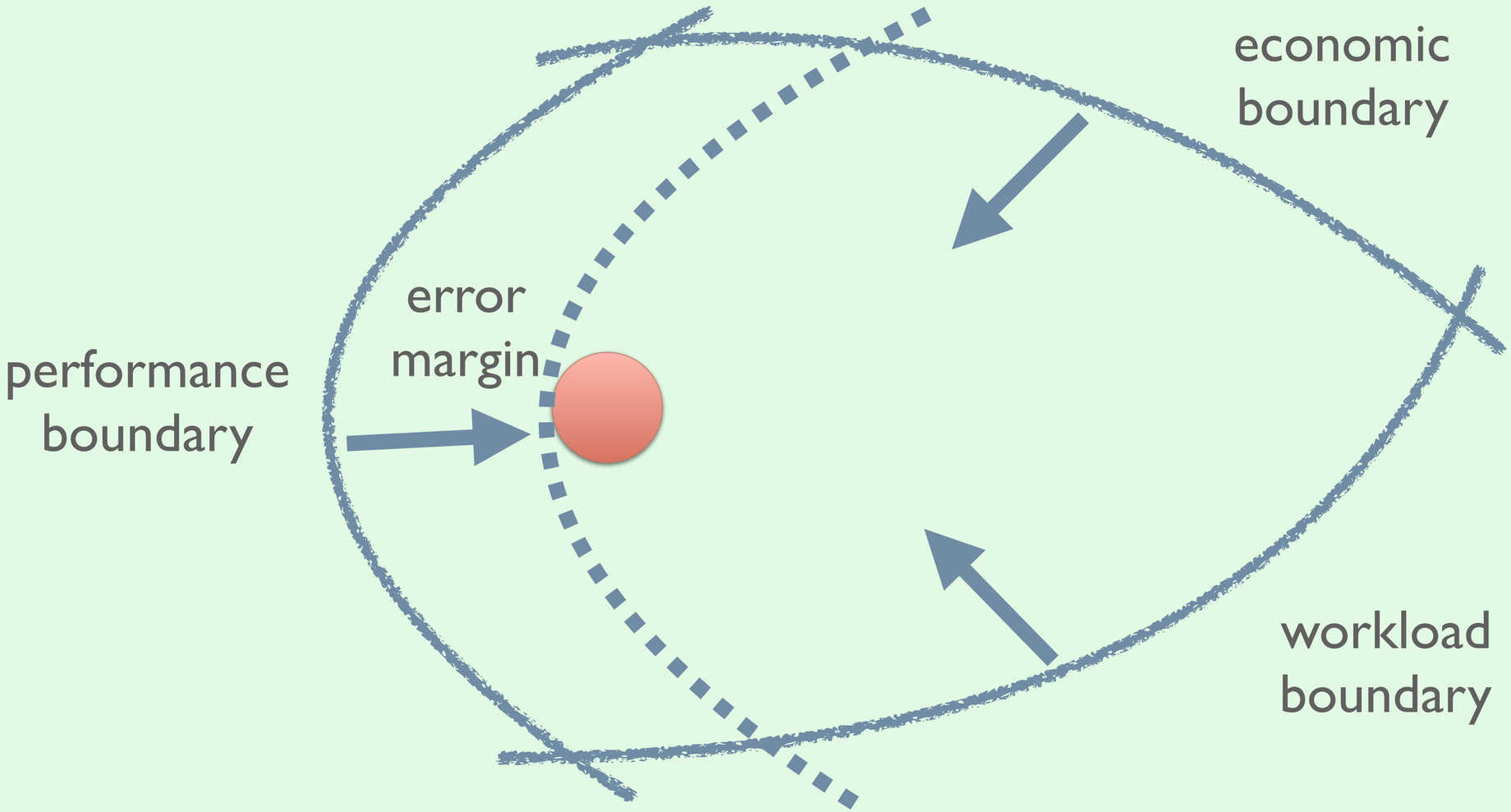
@leinweber

flirting with disaster



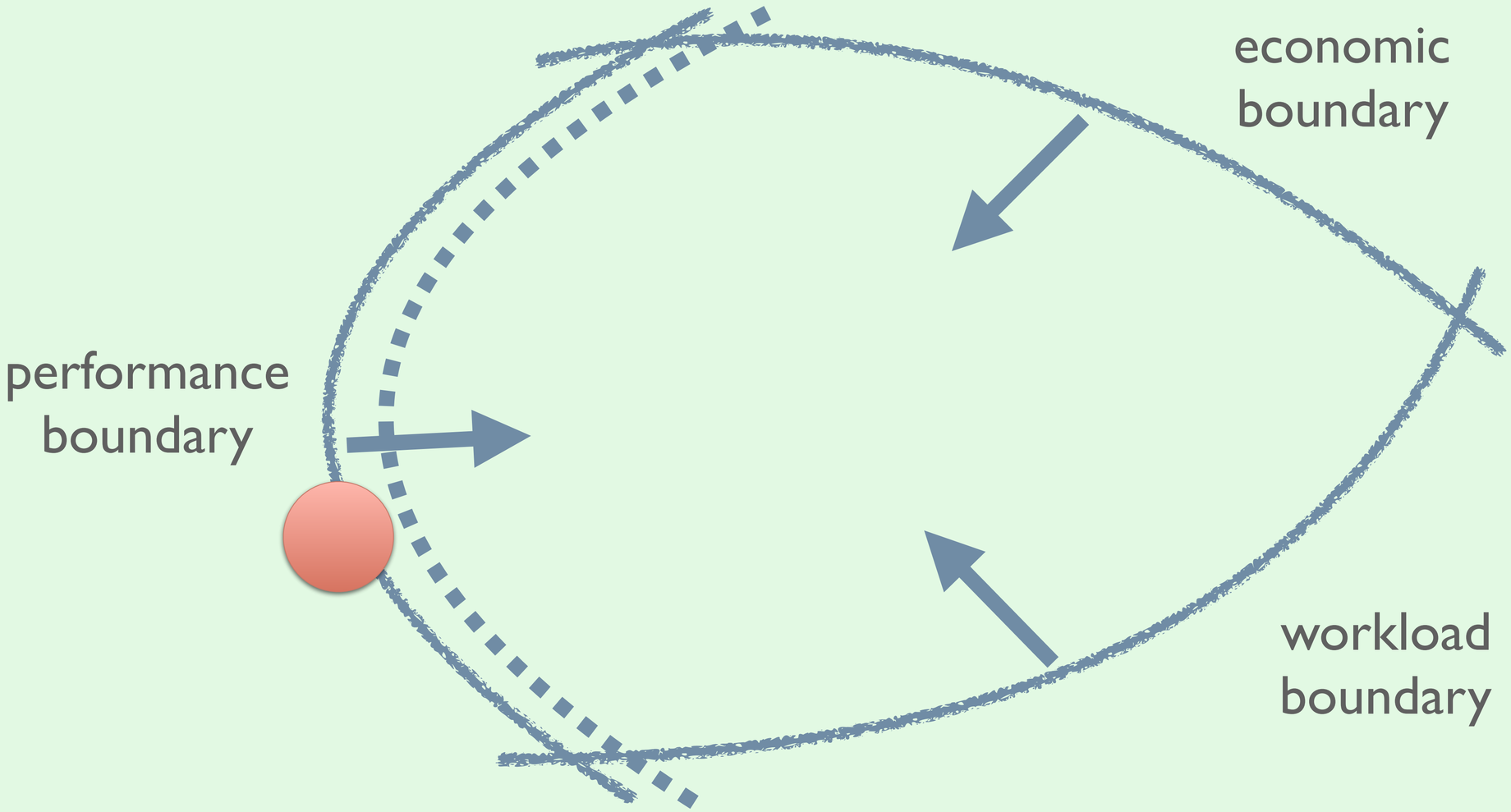
@leinweber

flirting with disaster



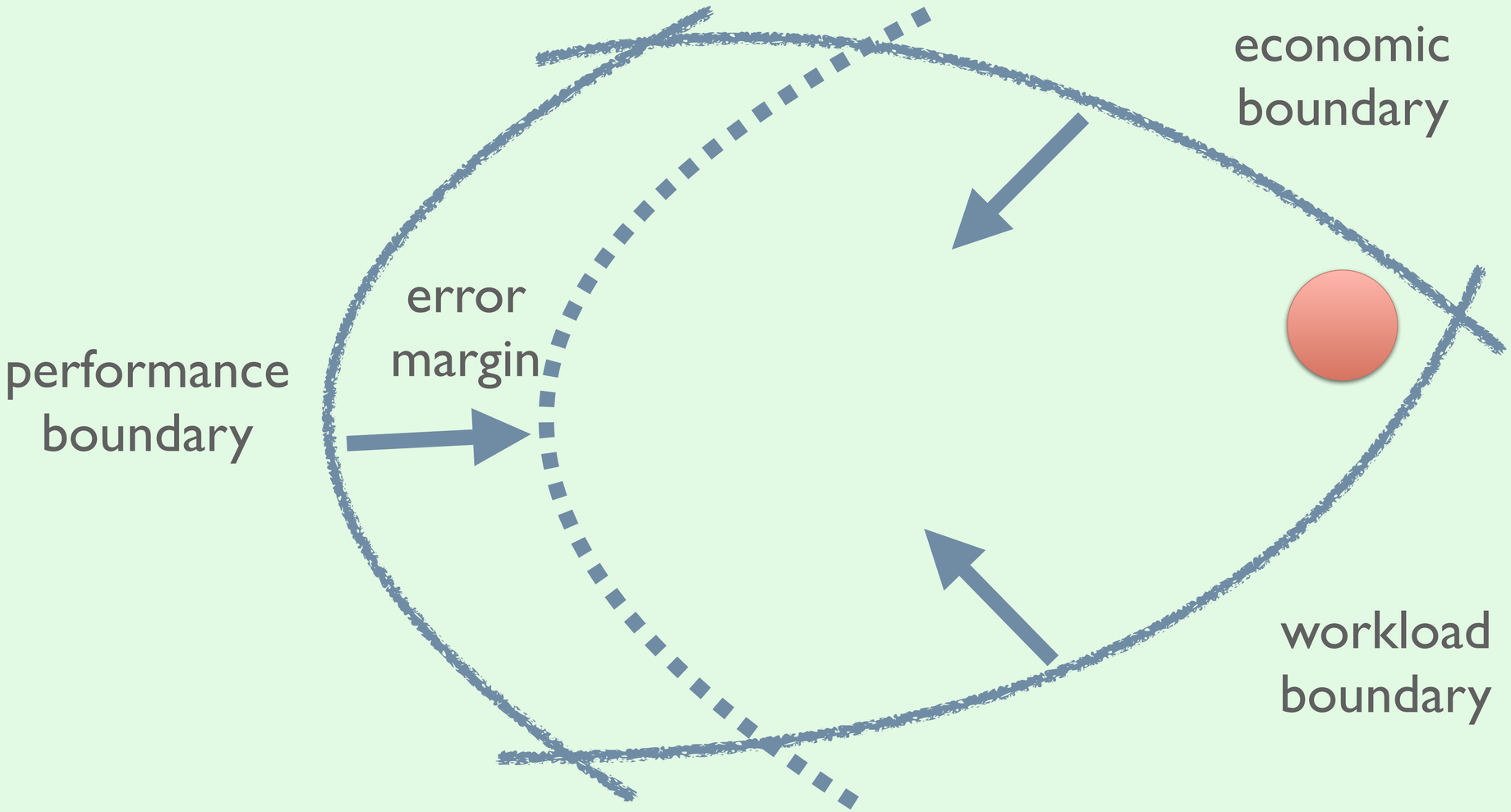
@leinweber

flirting with disaster



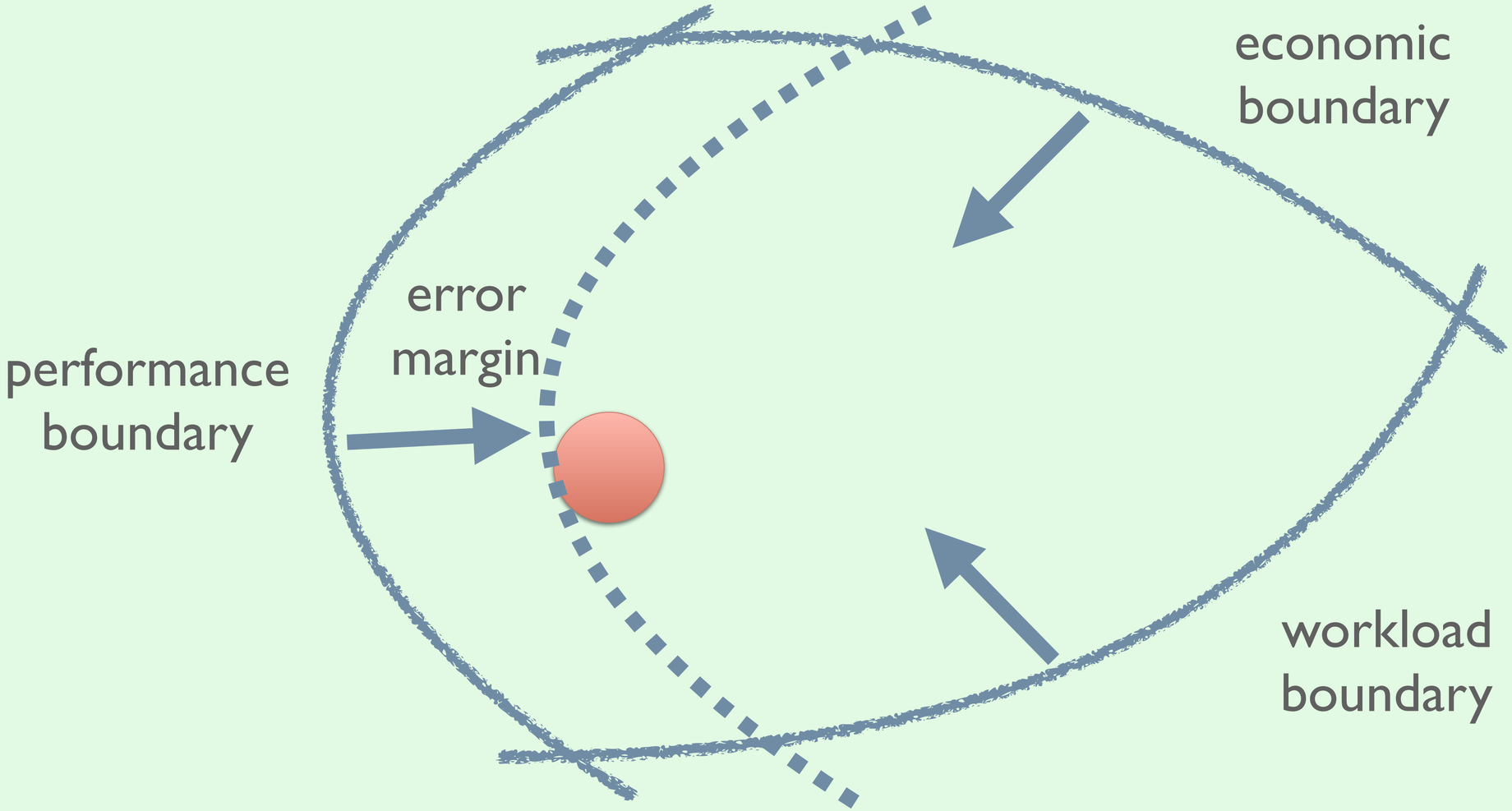
@leinweber

flirting with disaster



@leinweber

flirting with disaster



flirting with disaster

Velocity NY 2013: Richard Cook
"Resilience In Complex Adaptive Systems”

Jens Rasmussen:
Risk management in a dynamic society: a modeling problem

thank you

Will Leinweber
@leinweber
citusdata.com