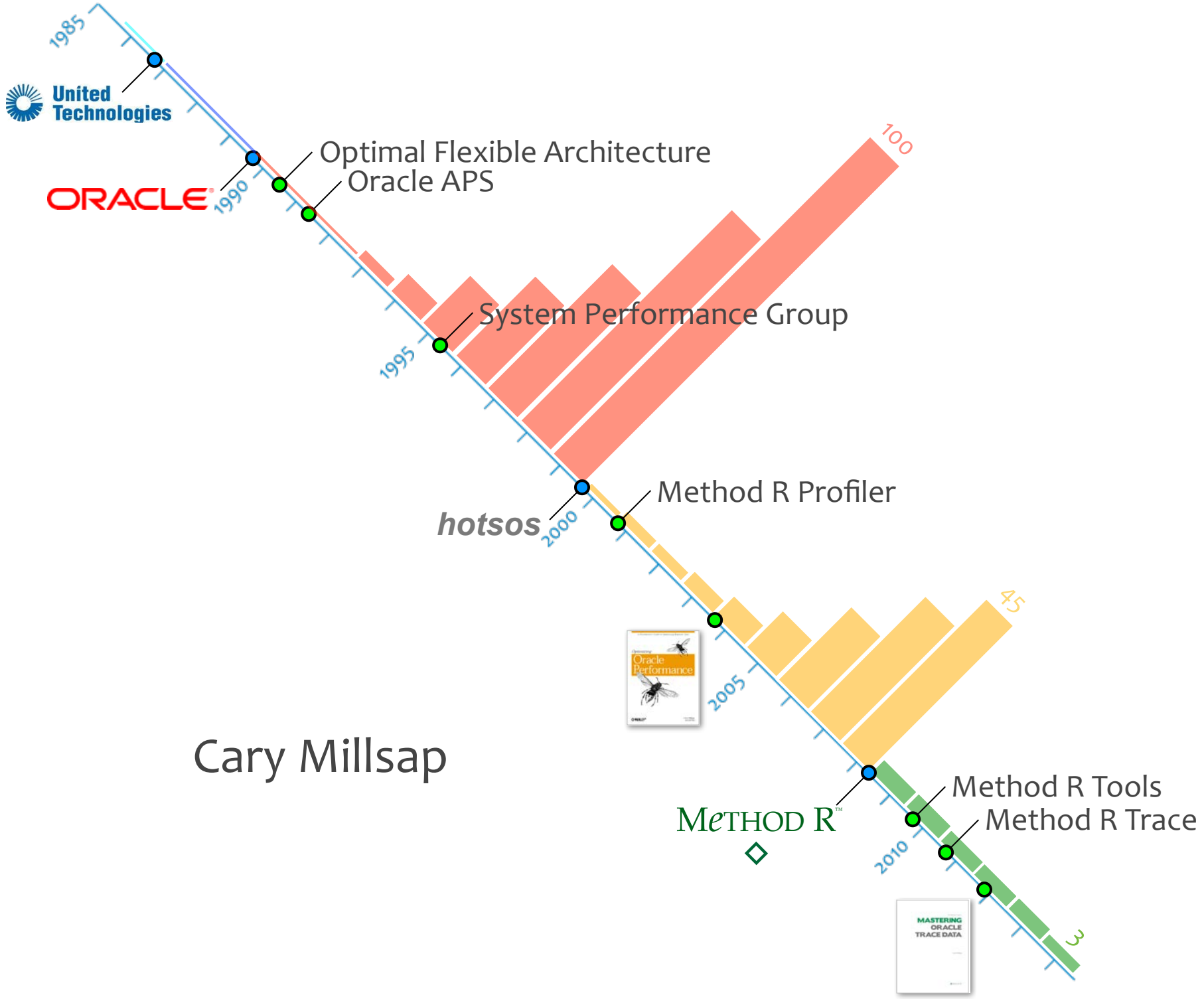
Thinking Clearly about Performance

Winner of ODTUG 2010 *Editor's Choice Award*
Published in *Communications of the ACM* and *ACM Queue*

Cary Millsap
Method R Corporation

@CaryMillsap · cary.millsap@method-r.com

Rittman Mead BI Forum
Atlanta, Georgia
5:00p–6:00p Thursday 16 May 2013


1985
United Technologies
ORACLE
1990
Optimal Flexible Architecture
Oracle APS
100
1995
System Performance Group
hotsos
2000
Method R Profiler
45
2005
Oracle Performance
Cary Millsap
METHOD R
Method R Tools
Method R Trace
2010
MASTERING ORACLE TRACE DATA
3

Method R™

method-r.com

@MethodR

create profiling tools

and other performance software

teach people

how to make software run faster

write applications

(primarily SQL and PL/SQL) for high-performance projects

fix problems

with performance on any Oracle-based system

Fundamentals of Performance (22)

1. An Axiomatic Approach
2. What is Performance?
3. Response Time vs. Throughput
4. Percentile Specifications
5. Problem Diagnosis
6. The Sequence Diagram
7. The Profile
8. Bottleneck
9. Amdahl's Law
10. Skew
11. Minimizing Risk
12. Efficiency
13. Load
14. Queueing Delay
15. The Knee
16. Relevance of the Knee
17. Capacity Planning
18. Random Arrivals
19. Coherency Delay
20. Performance Testing
21. Measuring
22. Performance is a Feature

Fundamentals of Performance (22)

1. An Axiomatic Approach
2. What is Performance?
3. Response Time vs. Throughput
4. Percentile Specifications
5. Problem Diagnosis
6. The Sequence Diagram
7. The Profile
8. Bottleneck
9. Amdahl's Law
10. Skew
11. Minimizing Risk
12. Efficiency
13. Load
14. Queueing Delay
15. The Knee
16. Relevance of the Knee
17. Capacity Planning
18. Random Arrivals
19. Coherency Delay
20. Performance Testing
21. Measuring
22. Performance is a Feature

Fundamentals of Performance (22)

1. An Axiomatic Approach
2. What is Performance?
3. Response Time vs. Throughput
4. Percentile Specifications
5. Problem Diagnosis
6. The Sequence Diagram
7. The Profile
8. Bottleneck
9. Amdahl's Law
10. Skew
11. Minimizing Risk
12. Efficiency
13. Load
14. Queueing Delay
15. The Knee
16. Relevance of the Knee
17. Capacity Planning
18. Random Arrivals
19. Coherency Delay
20. Performance Testing
21. Measuring
22. Performance is a Feature

1. An Axiomatic Approach

Given: $a \in R$, $b \in R$, $c \in R$ where $R = \{\text{Real Numbers}\}$, then:

I

Addition

① $a+b=b+a$	Commutative Property of Addition
② $a+(b+c)=(a+b)+c$	Associative Property of Addition
③ $a+(-a)=(-a)+a=0$	additive Inverse Property
④ $a+0=0+a=a$	additive Identity Element
⑤ $a+b=c$	Closure for Addition

Multiplication

① $a \cdot b = b \cdot a$	① Commutative Property of Multiplication
② $(a \cdot b) \cdot c = a \cdot (b \cdot c)$	② Associative Property of Multiplication
③ $a \cdot \frac{1}{a} = \frac{1}{a} \cdot a = 1$, $a \neq 0$	③ Multiplicative Inverse Property
④ $a \cdot 1 = 1 \cdot a = a$	④ Multiplicative Identity Element
⑤ $a \cdot b = c$	⑤ Closure for Multiplication
⑥ $a \cdot 0 = 0 \cdot a = 0$	⑥ Multiplication Property of Zero

II Axioms of Equality: ① $a=a$ Reflective Property of Equality.
② If $a=b$, then $b=a$. Symmetrical Property of Equality
③ If $a=b$, $b=c$, then $a=c$. Transitive Property of Equality.

III Distributive Property: $a[b+c] = ab+ac$

IV Axiom of opposites: $-(-a)=a$

V $-(a+b)=(-a)+(-b)$ Property of opposite of a Sum

VI $a \cdot (-1) = (-1) \cdot a = -a$ Multiplication Property of -1

VII $(-a)(b) = a(-b) = -ab$ and $(-a)(-b) = ab$ Property of opposites in Products

VIII $\frac{1}{\frac{1}{a}} = a$ and $\frac{1}{-a} = -\frac{1}{a}$, $a \neq 0$. Axiom of Reciprocals

IX $\frac{1}{ab} = \frac{1}{a} \cdot \frac{1}{b}$ Property of the Reciprocal of a Product.

X $a-b=a+(-b)$ Definition of Subtraction.

XI $\frac{a}{b} = \frac{1}{b} \cdot a = a \cdot \frac{1}{b}$, $b \neq 0$ Definition of Division

XII If $a=b$, then $a+c=b+c$ Addition Property of Equality

XIII If $a=b$, then $a-c=b-c$ Subtraction Property of Equality

XIV If $a=b$, then $ac=bc$ Multiplication Property of Equality

XV If $a=b$, $c \neq 0$, then $\frac{a}{c} = \frac{b}{c}$ Division Property of Equality

XVI For all real numbers a and b, one and only one of the following statements is true: $a<b$, $a=b$, $a>b$ Axiom of Comparison.

XVII Every decimal represents a real number and every real number has a decimal representation. Axiom of Completeness.

XVIII Between any two real numbers there is another real number. Property of Density.

1.	$\frac{7}{x}-6=20$	Given
2.	$\left(\frac{7}{x}-6\right)+6=(20)+6$	Addition property of equality
3.	$\frac{7}{x}+(-6+6)=26$	Associative property of addition
4.	$\frac{7}{x}+(0)=26$	Additive inverse property
5.	$\frac{7}{x}=26$	Additive identity element
6.	$\left(\frac{7}{x}\right)x=26x$	Multiplication property of equality
7.	$7\left(\frac{x}{x}\right)=26x$	Associative property of multiplication
8.	$7(1)=26x$	Multiplicative inverse property
9.	$7=26x$	Multiplicative identity element
10.	$\frac{7}{26}=\frac{26x}{26}$	Division property of equality
11.	$\frac{7}{26}=\frac{26}{26}x$	Associative property of multiplication
12.	$\frac{7}{26}=(1)x$	Multiplicative inverse property
13.	$\frac{7}{26}=x$	Multiplicative identity element
14.	$x=\frac{7}{26}$	Symmetrical property of equality

Axioms of Arithmetic

Given: $a \in \mathbb{R}$, $b \in \mathbb{R}$, c $\in \mathbb{R}$, where $\mathbb{R}$ = {Real Numbers}, then:

Addition		
1A	$a+b=b+a$	Commutativity
2A	$(a+b)+c=a+(b+c)$	Associativity
3A	$a+(-a)=(-a)+a=0$	Inverse Property
4A	$a+0=0+a=a$	Identity Element
5A	$a+b=c$	Closure
6A	$a-b=a+(-b)$	Subtraction (*def.*)

Multiplication		
1M	$a \times b = b \times a$	Commutativity
2M	$(a \times b) \times c = a \times (b \times c)$	Associativity
3M	$a \times \frac{1}{a} = \frac{1}{a} \times a = 1,\ a \neq 0$	Inverse Property
4M	$a \times 1 = 1 \times a = a$	Identity Element
5M	$a \times b = c$	Closure
6M	$a / b = a \times (1/b),\ b \neq 0$	Division (*def.*)

7	$a \times 0 = 0 \times a = 0$	Multiplication Property of 0
8	$a \times -1 = -1 \times a = -a$	Multiplication Property of –1
9	$-(-a) = a$	Axiom of Opposites
10	$a(b+c) = ab + ac$	Distributive Property
11	$-(a+b) = (-a) + (-b)$	Property of Opposite of a Sum
12	$(-a)b = a(-b) = -ab$ and $(-a)(-b) = ab$	Property of Opposites in Products
13	$\frac{1}{\left(\frac{1}{a}\right)} = a$ and $\frac{1}{-a} = -\frac{1}{a},\ a \neq 0$	Axiom of Reciprocals
14	$\frac{1}{ab} = \frac{1}{a} \times \frac{1}{b}$	Property of the Reciprocal of a Product

15	$a = a$	Reflexive Property of Equality
16	If $a = b$, then $b = a$	Symmetrical Property of Equality
17	If $a = b$ and $b = c$, then $a = c$	Transitive Property of Equality

18	If $a = b$, then $a + c = b + c$	Addition Property of Equality
19	If $a = b$, then $a - c = b - c$	Subtraction Property of Equality
20	If $a = b$, then $ac = bc$	Multiplication Property of Equality
21	If $a = b$ and $c \neq 0$, then $\frac{a}{c} = \frac{b}{c}$	Division Property of Equality

22	For all real numbers a and b, one and only one of the following statements is true: $a < b$, $a = b$, $a > b$.	Axiom of Comparison
23	Every decimal represents a real number and every real number has a decimal representation.	Axiom of Completeness
24	Between any two real numbers there is another real number.	Property of Density

Adapted from James R. Harkey "Properties A" (1976) by Cary Millsap. Visit *http://carymillsap.blogspot.com*.

Why?

this axiomatic approach

1

You'll solve more complex problems.

2

Knowing matters.

Proving matters more.

2. What is Performance?

per·for·mance

noun

1 a measure that relates time to the executions of individual tasks.

> Performance is an attribute of each individual experience with a system.

time

noun

1 the indefinite continued progress of existence and events in the past, present, and future regarded as a whole: *"Time is what prevents everything from happening at once." —John Archibald Wheeler (1911–2008)*

task

noun

1 a piece of work to be done or undertaken.

A task is a *business* unit of work, named and described in the language of the business.

ex·per·i·ence

noun

1 an execution of a task.

Two ways to relate experiences to time

experiences/time

time/experience

through·put

noun

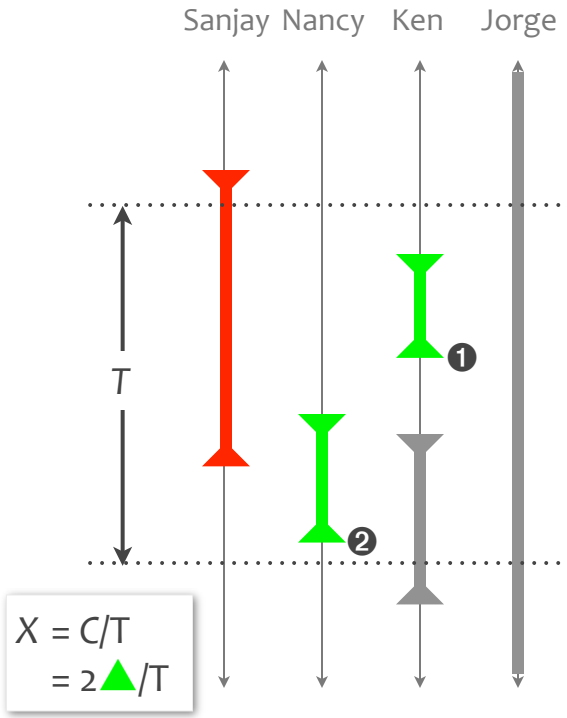
1 output or production, as of a computer program, over a period of time.

re·sponse time

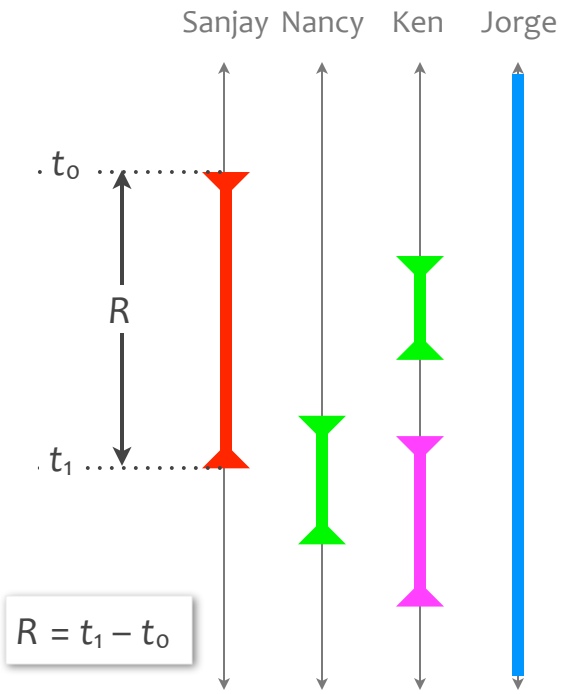
noun

1 the duration taken for a system to react to a given stimulus or event.

throughput (*X*) = experiences/time



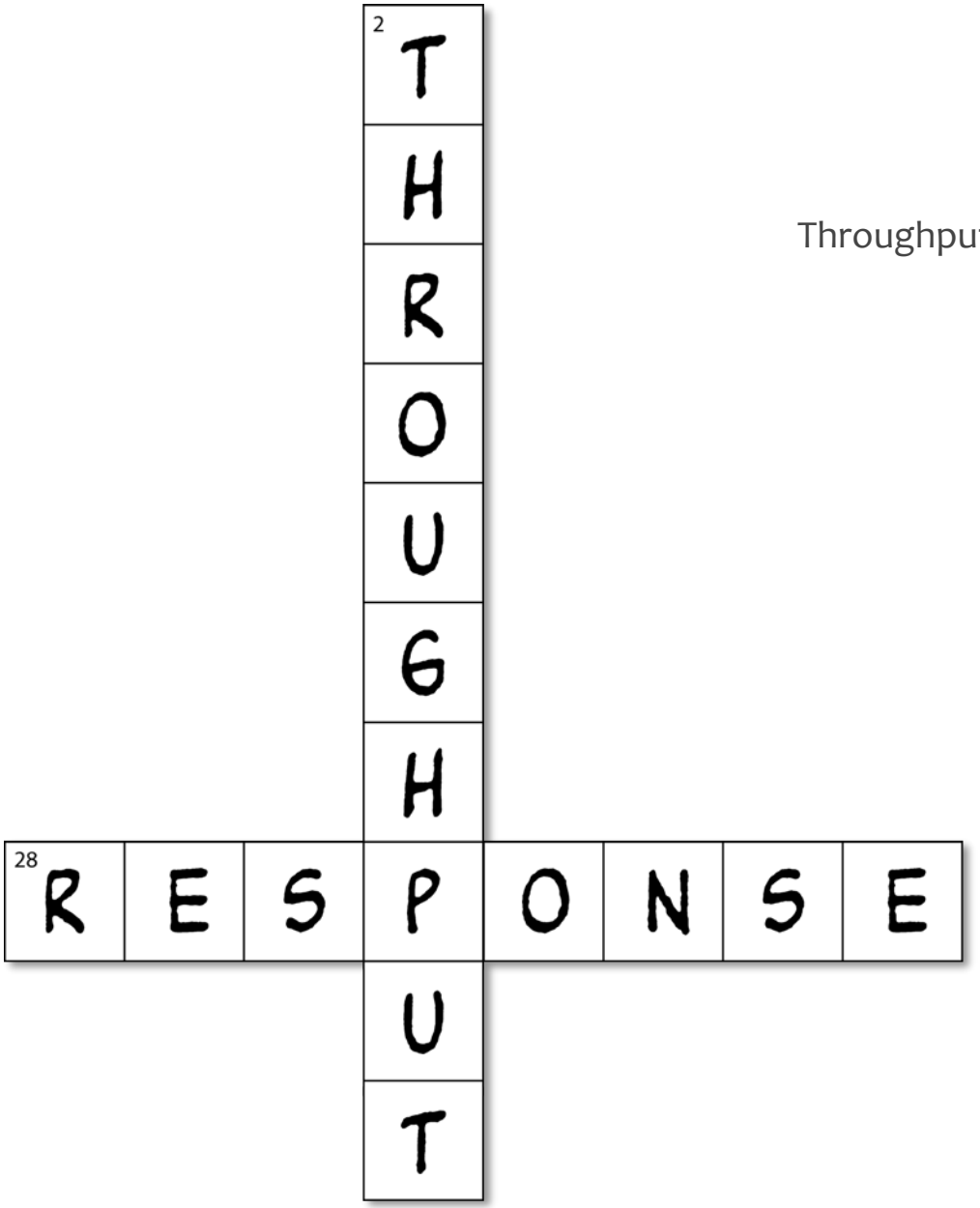
response time (R) = time/experience



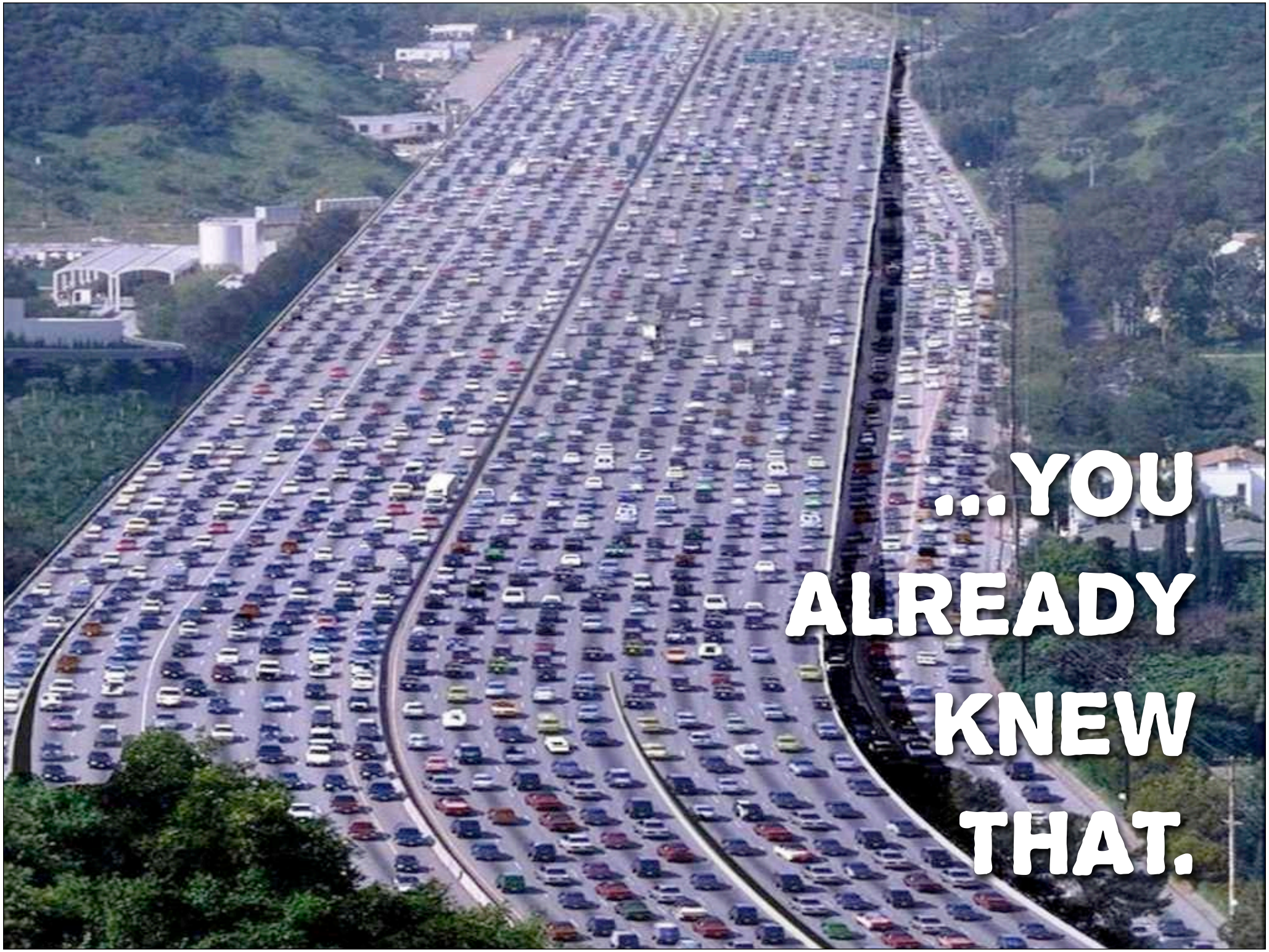
Throughput is important to **groups, leaders.**

Response time is important to **individuals, leaders.**

3. Response Time vs. Throughput



Throughput and response time are **related…**


...YOU
ALREADY
KNEW
THAT.

average throughput

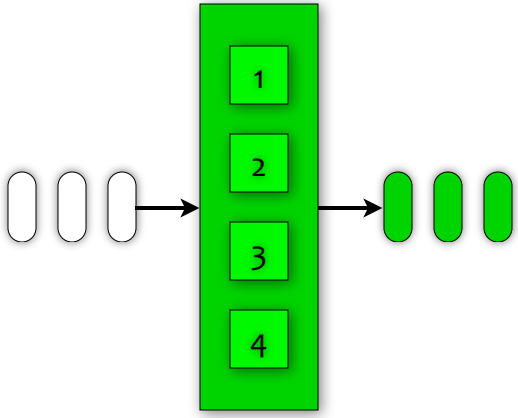
$$X(N) = \frac{N}{R(N) - Z}$$

average response time

$$R(N) = \frac{N}{X(N)} + Z$$

$N = 4$ parallel, independent, homogeneous service channels

$Z = 0$ think time



$$X(N) = \frac{N}{R(N) - Z}$$

$$R(N) = \frac{N}{X(N)} + Z$$

If $R(N) = 1$ sec/txn (average),

then $X(N) = 4/(1 - 0)$ txn/sec (average)
$= 4$ txn/sec (average).

If $X(N) = 8$ txn/sec (average),

then $R(N) = 4/8 + 0$ sec/txn (average)
$= .5$ sec/txn (average).

These formulas deal only in **averages.**

To know about **individual experiences,** you must

measure individual experiences.

4. Percentile Specifications

"Click to Order" must respond in ≤ 1.0s.

Imagine: 1-second tolerance

Which response times do you like better?

	$R(N)$
A	**1.000 s**
B	**1.000 s**

	List A	List B
1	.924	.796
2	.928	.798
3	.954	.802
4	.957	.823
5	.961	.919
6	.965	.977
7	.972	1.076
8	.979	1.216
9	.987	1.273
10	1.373	1.320

Imagine: 1-second tolerance

Which response times do you like better?

	R(*N*)	**Success rate**
A	1.000 s	**90%**
B	1.000 s	**60%**

	List A	List B
1	.924	.796
2	.928	.798
3	.954	.802
4	.957	.823
5	.961	.919
6	.965	.977
7	.972	1.076
8	.979	1.216
9	.987	1.273
10	1.373	1.320

THE FUNDAMENTAL THEOREM OF MEASURING THINGS

When obviously different experiences
yield the same measurement,
you're *measuring the wrong thing*.

"Click to Order" must respond in
≤ 1.0s for ≥ 90% of executions.

Our customers feel the variance, not the mean.

—GE

"What is Six Sigma? The Roadmap to Customer Impact"
at *http://www.ge.com/sixsigma/SixSigma.pdf*

Imagine: 1-second tolerance

Which response times do you like better?

	R(*N*)	**Success rate**
A	1.000 s	**90%**
C	1.000 s	**90%**

	List A	List C
1	.924	.091
2	.928	.109
3	.954	.134
4	.957	.136
5	.961	.159
6	.965	.172
7	.972	.185
8	.979	.191
9	.987	.207
10	1.373	8.616

THE FUNDAMENTAL THEOREM OF MEASURING THINGS

When obviously different experiences
yield the same measurement,
you're *measuring the wrong thing*.

"Click to Order" must respond in
≤ 1.0s for ≥ 90% of executions,
≤ 5.0s for ≥ 99% of executions.

Imagine: 1-second tolerance

Which response times do you like better?

	R(*N*)	1-sec tolerance success rate	**5-sec tolerance success rate**
A	1.000 s	90%	**99%**
C	1.000 s	90%	**90%**

	List A	List C
1	.924	.091
2	.928	.109
3	.954	.134
4	.957	.136
5	.961	.159
6	.965	.172
7	.972	.185
8	.979	.191
9	.987	.207
10	1.373	8.616

5. Problem Diagnosis

The right **start** is the most important thing.

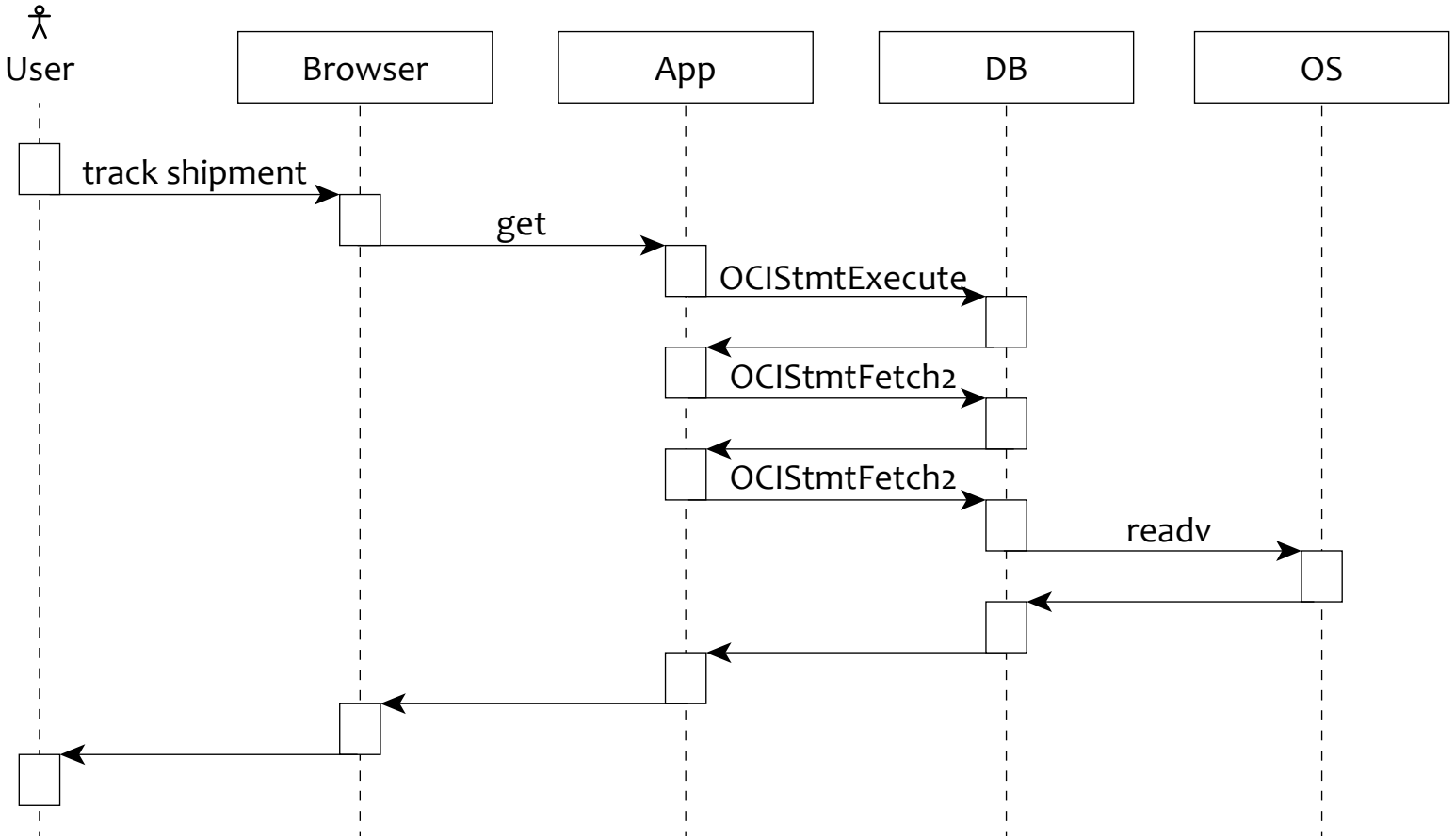
1. What is the current state?

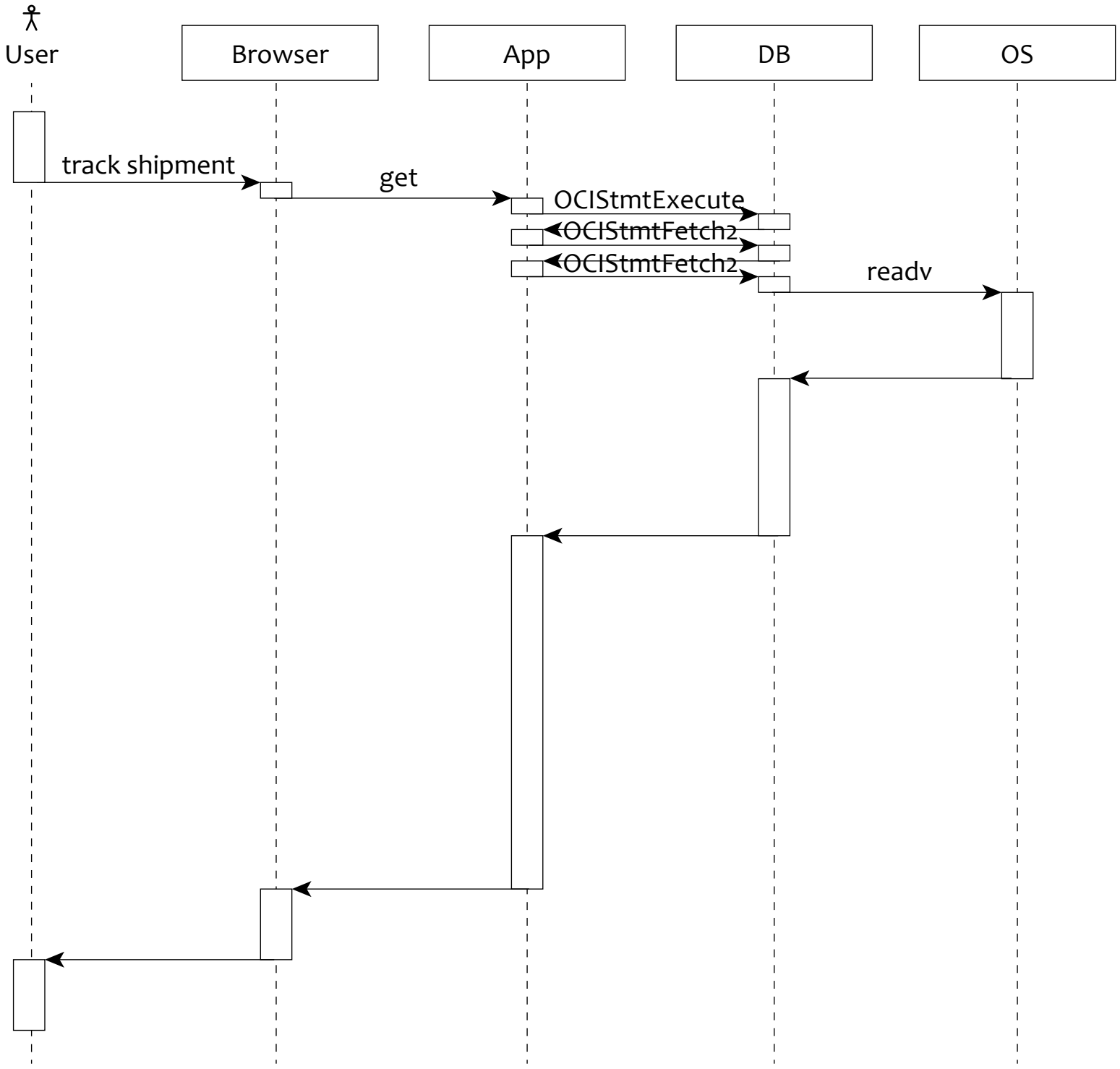
2. What is the goal state?

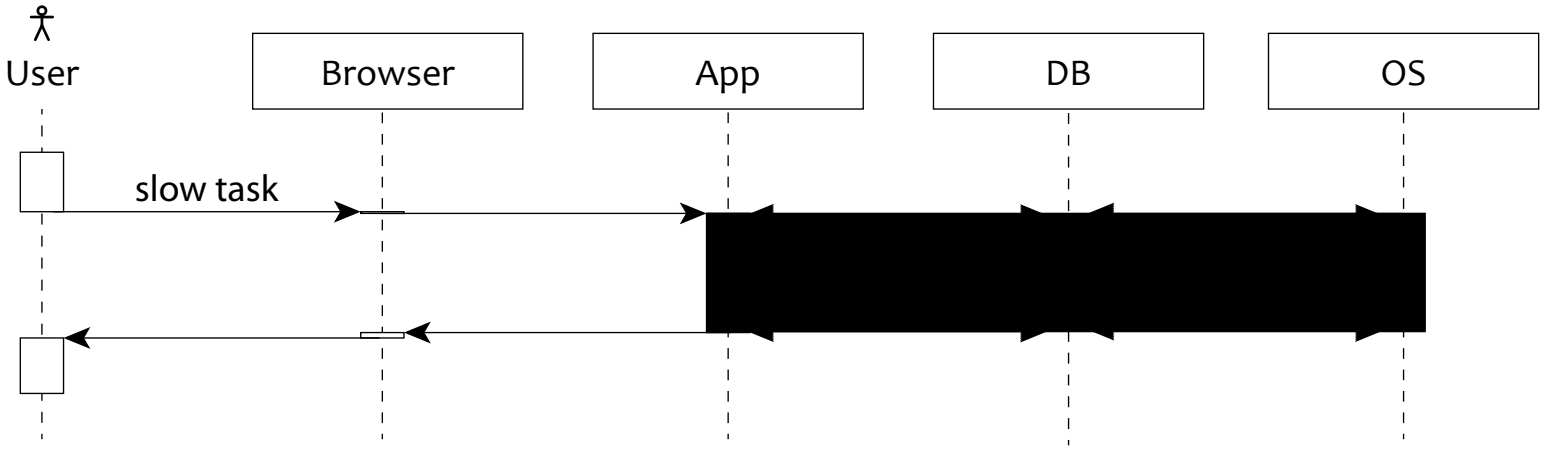
What if the goal state is **impossible**?

How can you **know**?

6. The Sequence Diagram


User
Browser
App
DB
OS
track shipment
get
OCIStmtExecute
OCIStmtFetch2
OCIStmtFetch2
readv


User
Browser
App
DB
OS
track shipment
get
OCIStmtExecute
OCIStmtFetch2
OCIStmtFetch2
readv


User
Browser
App
DB
OS
slow task

sequence diagram

...good **conceptual** tool.

sequence diagram

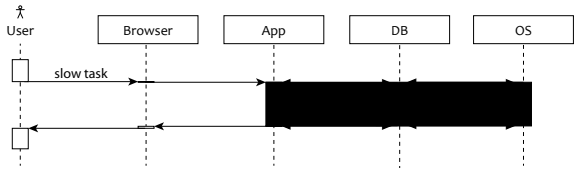
...doesn't **scale.**

7. The Profile

pro·file

noun

1 a tabular account of response time, in which the sum of component response times exactly equals the total response time being measured.



=

CALL-NAME	DURATION	%	CALLS	MEAN
db file sequential read	59,081.406102	76.6%	10,013,394	0.005900
log buffer space	6,308.758563	8.2%	9,476	0.665762
free buffer waits	4,688.730190	6.1%	200,198	0.023420
EXEC	4,214.190000	5.5%	36,987	0.113937
log file switch completion	1,552.471890	2.0%	1,853	0.837815
db file parallel read	464.976815	0.6%	7,641	0.060853
log file switch (checkpoint incomplete)	316.968886	0.4%	351	0.903045
rdbms ipc reply	244.937910	0.3%	2,737	0.089491
undo segment extension	140.267429	0.2%	1,411	0.099410
log file switch (private strand flush incomplete)	112.680587	0.1%	134	0.840900
17 others	23.367228	0.0%	58,126	0.000402
TOTAL (27)	77,148.755600	100.0%	10,332,308	0.007467

You've done this before if you've ever used…

gcc -pg … ; gprof …

java -prof … ; java ProfilerViewer …

perl -d:Dprof … ; dprofpp …

p5prof …

mrskew …

Where did my code spend my time?

Where did it ***not*** spend my time?

How long should this task run?

What if the goal state is **impossible**?

How can you **know**?

Profiling is how you can **know**.

8. Bottleneck

bot·tle·neck

noun

1 the resource that dominates a given task's response time.

In other words,

bottleneck = profile's top line

Quiz: What is the task's bottleneck?

CALL-NAME	DURATION	%	CALLS	MEAN	MIN	MAX
SQL*Net message from client	984.010000	50.3%	95,161	0.010340	0.000000	0.310000
SQL*Net more data from client	418.820000	21.4%	3,345	0.125208	0.000000	0.270000
db file sequential read	279.340000	14.3%	45,084	0.006196	0.000000	0.050000
EXEC	136.880000	7.0%	67,888	0.002016	0.000000	1.320000
PARSE	74.490000	3.8%	10,098	0.007377	0.000000	0.090000
FETCH	37.320000	1.9%	57,217	0.000652	0.000000	0.130000
latch free	23.690000	1.2%	34,695	0.000683	0.000000	0.080000
log file sync	1.090000	0.1%	506	0.002154	0.000000	0.050000
SQL*Net more data to client	0.830000	0.0%	15,982	0.000052	0.000000	0.020000
log file switch completion	0.280000	0.0%	3	0.093333	0.080000	0.110000
enqueue	0.250000	0.0%	106	0.002358	0.000000	0.020000
SQL*Net message to client	0.240000	0.0%	95,161	0.000003	0.000000	0.010000
buffer busy waits	0.220000	0.0%	67	0.003284	0.000000	0.020000
db file scattered read	0.010000	0.0%	2	0.005000	0.000000	0.010000
SQL*Net break/reset to client	0.000000	0.0%	2	0.000000	0.000000	0.000000
TOTAL (15)	1,957.470000	100.0%	425,317	0.004602	0.000000	1.320000

Quiz: What is the task's bottleneck?

CALL-NAME	DURATION	%	CALLS	MEAN	MIN	MAX
SQL*Net message from client	984.010000	50.3%	95,161	0.010340	0.000000	0.310000
SQL*Net more data from client	418.820000	21.4%	3,345	0.125208	0.000000	0.270000
db file sequential read	279.340000	14.3%	45,084	0.006196	0.000000	0.050000
EXEC	136.880000	7.0%	67,888	0.002016	0.000000	1.320000
PARSE	74.490000	3.8%	10,098	0.007377	0.000000	0.090000
FETCH	37.320000	1.9%	57,217	0.000652	0.000000	0.130000
latch free	23.690000	1.2%	34,695	0.000683	0.000000	0.080000
log file sync	1.090000	0.1%	506	0.002154	0.000000	0.050000
SQL*Net more data to client	0.830000	0.0%	15,982	0.000052	0.000000	0.020000
log file switch completion	0.280000	0.0%	3	0.093333	0.080000	0.110000
enqueue	0.250000	0.0%	106	0.002358	0.000000	0.020000
SQL*Net message to client	0.240000	0.0%	95,161	0.000003	0.000000	0.010000
buffer busy waits	0.220000	0.0%	67	0.003284	0.000000	0.020000
db file scattered read	0.010000	0.0%	2	0.005000	0.000000	0.010000
SQL*Net break/reset to client	0.000000	0.0%	2	0.000000	0.000000	0.000000
TOTAL (15)	1,957.470000	100.0%	425,317	0.004602	0.000000	1.320000

It is "SQL*Net message from client".

Bad:

“**bottleneck** = busiest resource on the system while your task runs”

Same profile as before...

CALL-NAME	DURATION	%	CALLS	MEAN	MIN	MAX
SQL*Net message from client	984.010000	50.3%	95,161	0.010340	0.000000	0.310000
SQL*Net more data from client	418.820000	21.4%	3,345	0.125208	0.000000	0.270000
db file sequential read	279.340000	14.3%	45,084	0.006196	0.000000	0.050000
EXEC	136.880000	7.0%	67,888	0.002016	0.000000	1.320000
PARSE	74.490000	3.8%	10,098	0.007377	0.000000	0.090000
FETCH	37.320000	1.9%	57,217	0.000652	0.000000	0.130000
latch free	23.690000	1.2%	34,695	0.000683	0.000000	0.080000
log file sync	1.090000	0.1%	506	0.002154	0.000000	0.050000
SQL*Net more data to client	0.830000	0.0%	15,982	0.000052	0.000000	0.020000
log file switch completion	0.280000	0.0%	3	0.093333	0.080000	0.110000
enqueue	0.250000	0.0%	106	0.002358	0.000000	0.020000
SQL*Net message to client	0.240000	0.0%	95,161	0.000003	0.000000	0.010000
buffer busy waits	0.220000	0.0%	67	0.003284	0.000000	0.020000
db file scattered read	0.010000	0.0%	2	0.005000	0.000000	0.010000
SQL*Net break/reset to client	0.000000	0.0%	2	0.000000	0.000000	0.000000
TOTAL (15)	1,957.470000	100.0%	425,317	0.004602	0.000000	1.320000

Same profile as before...

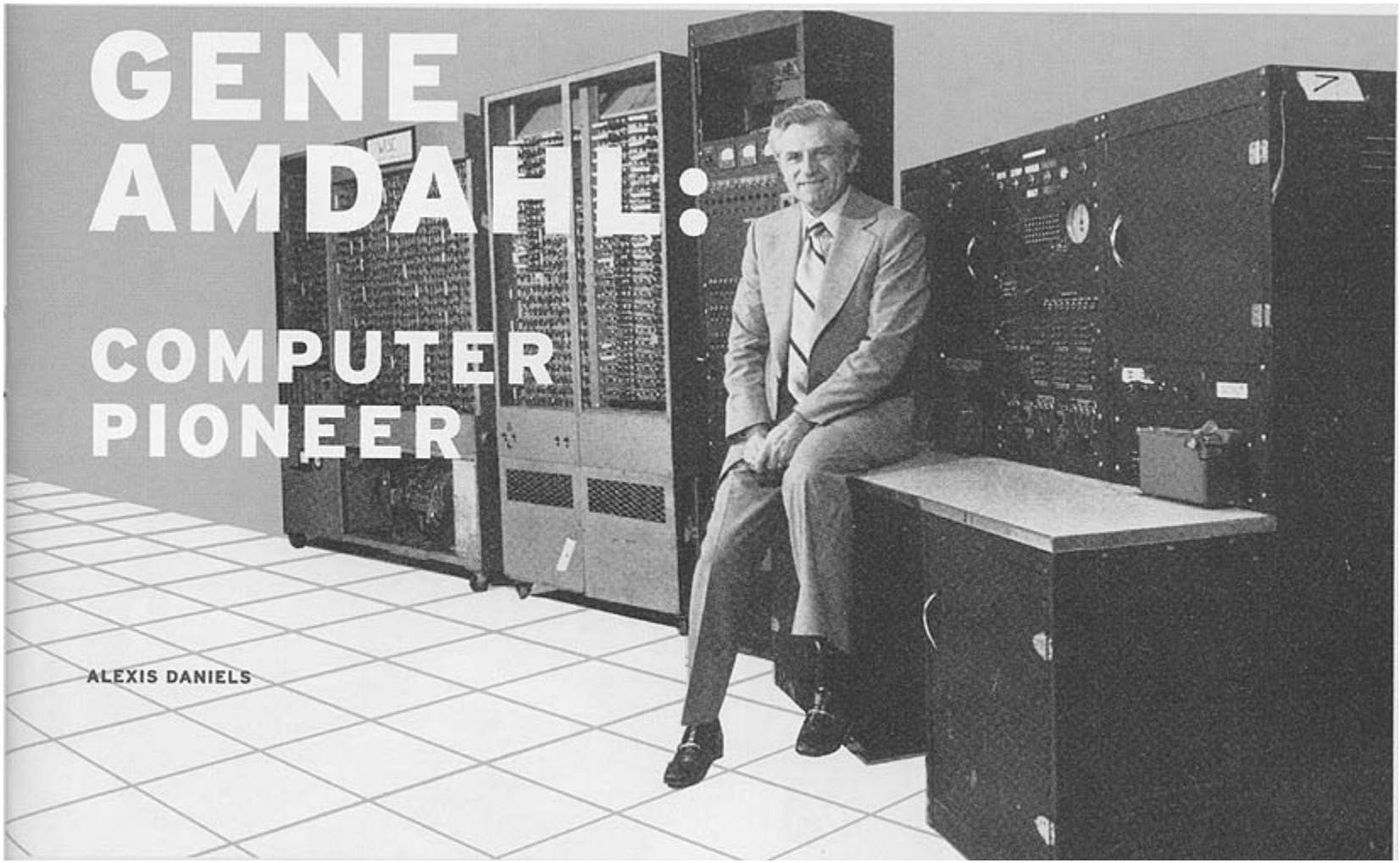
CALL-NAME	DURATION	%	CALLS	MEAN	MIN	MAX
SQL*Net message from client	984.010000	50.3%	95,161	0.010340	0.000000	0.310000
SQL*Net more data from client	418.820000	21.4%	3,345	0.125208	0.000000	0.270000
db file sequential read	279.340000	14.3%	45,084	0.006196	0.000000	0.050000
EXEC	136.880000	7.0%	67,888	0.002016	0.000000	1.320000
PARSE	74.490000	3.8%	10,098	0.007377	0.000000	0.090000
FETCH	37.320000	1.9%	57,217	0.000652	0.000000	0.130000
▶ latch free	23.690000	1.2%	34,695	0.000683	0.000000	0.080000
log file sync	1.090000	0.1%	506	0.002154	0.000000	0.050000
SQL*Net more data to client	0.830000	0.0%	15,982	0.000052	0.000000	0.020000
log file switch completion	0.280000	0.0%	3	0.093333	0.080000	0.110000
enqueue	0.250000	0.0%	106	0.002358	0.000000	0.020000
SQL*Net message to client	0.240000	0.0%	95,161	0.000003	0.000000	0.010000
buffer busy waits	0.220000	0.0%	67	0.003284	0.000000	0.020000
db file scattered read	0.010000	0.0%	2	0.005000	0.000000	0.010000
SQL*Net break/reset to client	0.000000	0.0%	2	0.000000	0.000000	0.000000
TOTAL (15)	1,957.470000	100.0%	425,317	0.004602	0.000000	1.320000

...This system was dominated by “latch free” while the task ran.

Bottleneck is defined in the context of a task execution.

Different experiences on a given system have different **bottlenecks.**

9. Amdahl's Law


GENE AMDAHL:
COMPUTER PIONEER
ALEXIS DANIELS

A task's response time can improve only in proportion to how much the task uses the thing you improve.

$$C(N) = \frac{N}{1+\alpha(N-1)}$$

	Response time		Potential improvement	Relative cost
1	1,748.229	70.8%	35.4%	1,000,000
2	338.470	13.7%	12.3%	1
3	152.654	6.2%		+∞
4	97.855	4.0%	4.0%	1
5	58.147	2.4%		+∞
6	48.274	2.0%	1.6%	1
7	23.481	1.0%		+∞
8	0.890	0.0%		+∞
	2,468.000	**100.0%**	**53.3%**	**1,000,003**

	Response time		Potential improvement	Relative cost
1	1,748.229	70.8%	35.4%	1,000,000
2	338.470	13.7%	12.3%	1
3	152.654	6.2%		+∞
4	97.855	4.0%	4.0%	1
5	58.147	2.4%		+∞
6	48.274	2.0%	1.6%	1
7	23.481	1.0%		+∞
8	0.890	0.0%		+∞
	2,468.000	**100.0%**	**53.3%**	**1,000,003**

First, assess the **whole** profile

before deciding upon your next step.

10. Skew

skew

noun

1 a non-uniformity in a list.

Skew is what fouls your ability to *predict* results.

Quiz: How much time will you save if you eliminate half of the 10,013,394 "db file sequential read" calls?

CALL-NAME	DURATION	%	CALLS	MEAN	MIN	MAX
db file sequential read	59,081.406102	76.6%	10,013,394	0.005900	0.000010	15.853019
log buffer space	6,308.758563	8.2%	9,476	0.665762	0.000004	1.010092
free buffer waits	4,688.730190	6.1%	200,198	0.023420	0.000004	1.021281
EXEC	4,214.190000	5.5%	36,987	0.113937	0.000000	5.400000
log file switch completion	1,552.471890	2.0%	1,853	0.837815	0.000006	1.013093
22 others	1,303.198855	1.7%	70,400	0.000402	0.000000	8.964706
TOTAL (27)	77,148.755600	100.0%	10,332,308	0.007467	0.000000	15.853019

...half of 59,081 seconds?

Quiz: How much time will you save if you eliminate half of the 10,013,394 "db file sequential read" calls?

	RANGE {min ≤ e < max}		DURATION	%	CALLS	MEAN	MIN	MAX
1.	0.000000	0.000001						
2.	0.000001	0.000010						
3.	0.000010	0.000100	199.445978	0.3%	9,346,059	0.000021	0.000010	0.000099
4.	0.000100	0.001000	21.420428	0.0%	108,351	0.000198	0.000100	0.000999
5.	0.001000	0.010000	612.513248	1.0%	106,319	0.005761	0.001000	0.009999
6.	0.010000	0.100000	11,193.505611	18.9%	314,869	0.035550	0.010000	0.099999
7.	0.100000	1.000000	26,057.804096	44.1%	130,471	0.199721	0.100002	0.999717
8.	1.000000	10.000000	20,804.497660	35.2%	7,308	2.846811	1.000184	9.900656
9.	10.000000	100.000000	192.219083	0.3%	17	11.307005	10.242772	15.853019
10.	100.000000	1,000.000000						
11.	1,000.000000	+∞						
		TOTAL (11)	59,081.406102	100.0%	10,013,394	0.005900	0.000010	15.853019

...Depends on *which* half:

- Eliminate the green ones (93% of calls), and you'll save 0.3% of time.
- Eliminate the red ones (just 5% of calls), and you'll save 98.5% of time.

11. Minimizing Risk

When everyone is happy except for you, make sure your local stuff is in order before you go messing around with the global stuff that affects everyone else, too.

—*Common Sense (?)*

Match the scope of the solution

to the scope of the problem.

12. Efficiency

ef·fi·cien·cy

noun

1 an inverse measure of waste.

waste

noun

1 any work that could be eliminated without sacrificing useful benefits.

Obviously, the highest type of efficiency is that which can utilize existing material to the best advantage.

—*Jawaharlal Nehru*
1889–1964

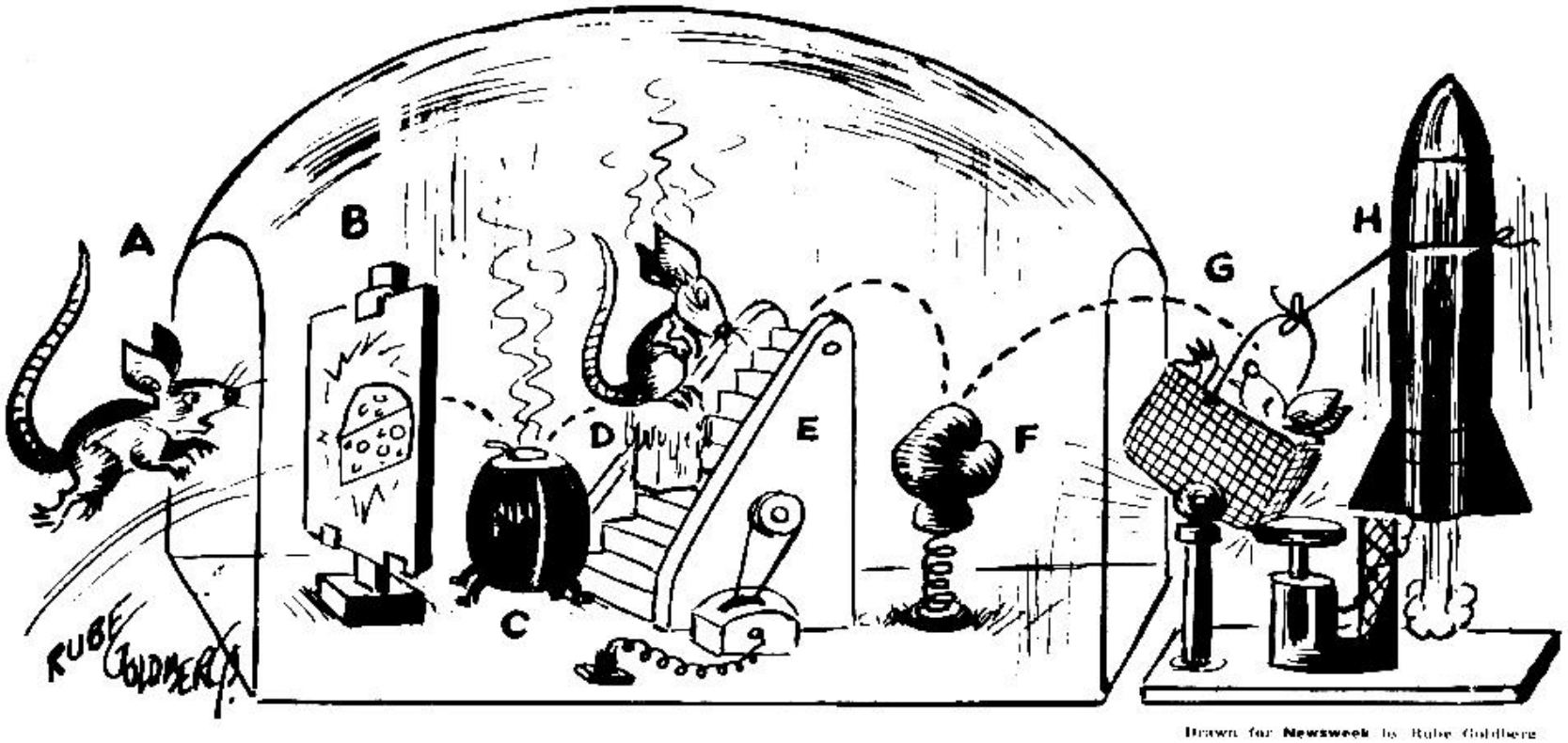
To measure efficiency,

profile.

The **fastest** way to do something is

to not do it at all.

How to Get Rid of a Mouse



Drawn for **Newsweek** by Rube Goldberg

The best mousetrap by Rube Goldberg: Mouse (A) dives for painting of cheese (B), goes through canvas and lands on hot stove (C). He jumps on cake of ice (D) to cool off. Moving escalator (E) drops him on boxing glove (F) which knocks him into basket (G) setting off miniature rocket (H) which takes him to the moon.

Is the **apparent requirement**
really a **legitimate** requirement?

Improvements that make

your program more **efficient**

can benefit

everyone on the system.

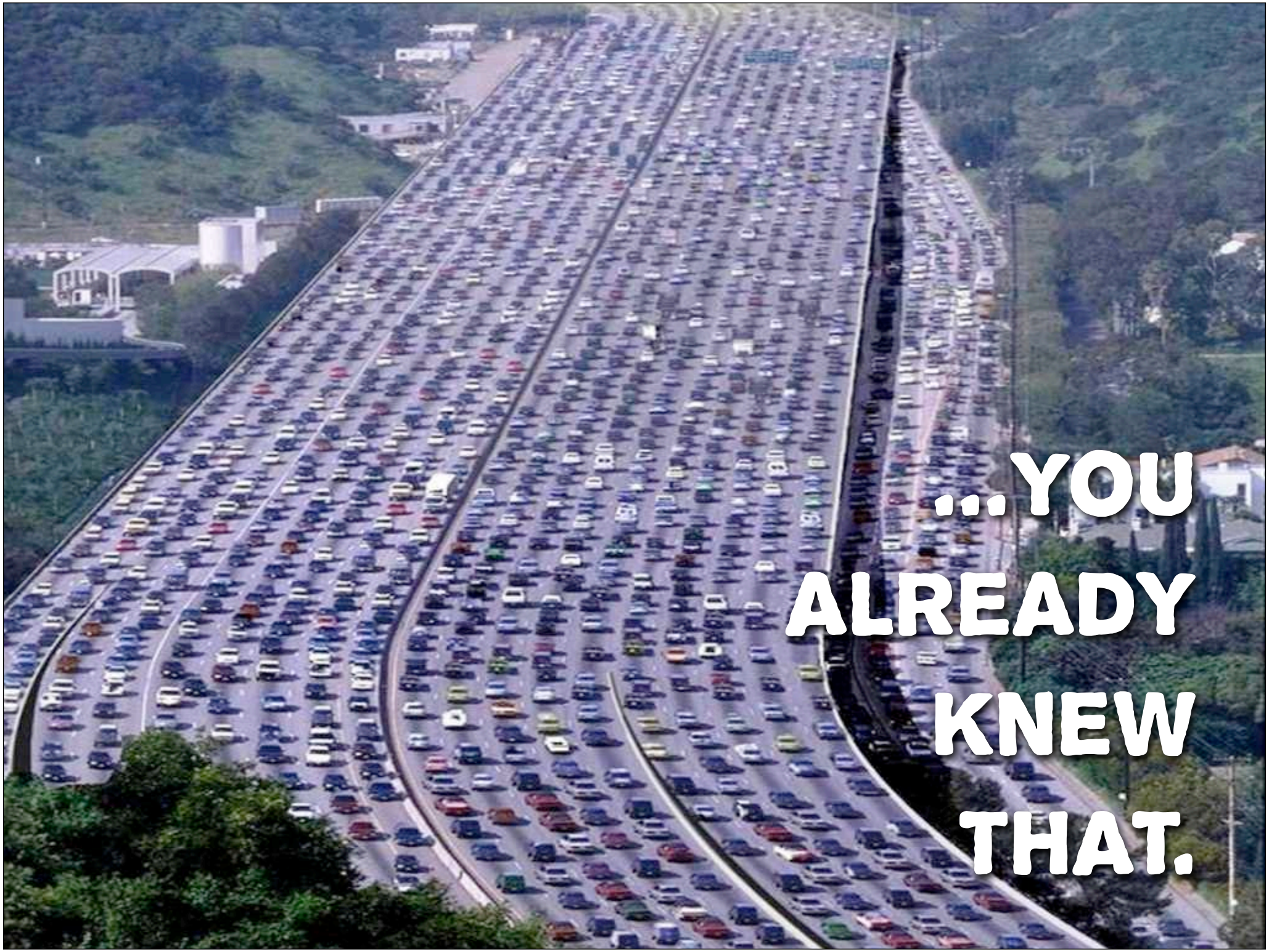
13. Load

load

noun

1 competition for a resource by concurrent task executions.

busier $\rightarrow$ more waiting


...YOU
ALREADY
KNEW
THAT.

Two types of waiting:

queueing delay

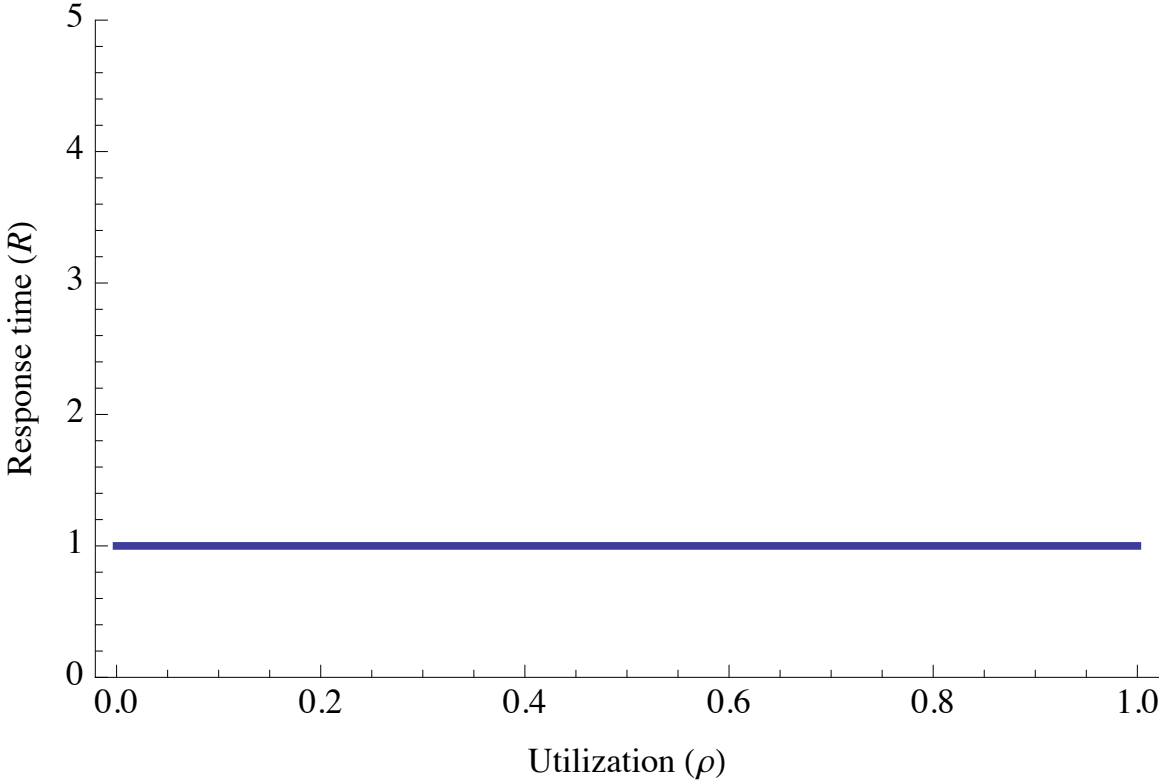
coherency delay

14. Queueing Delay

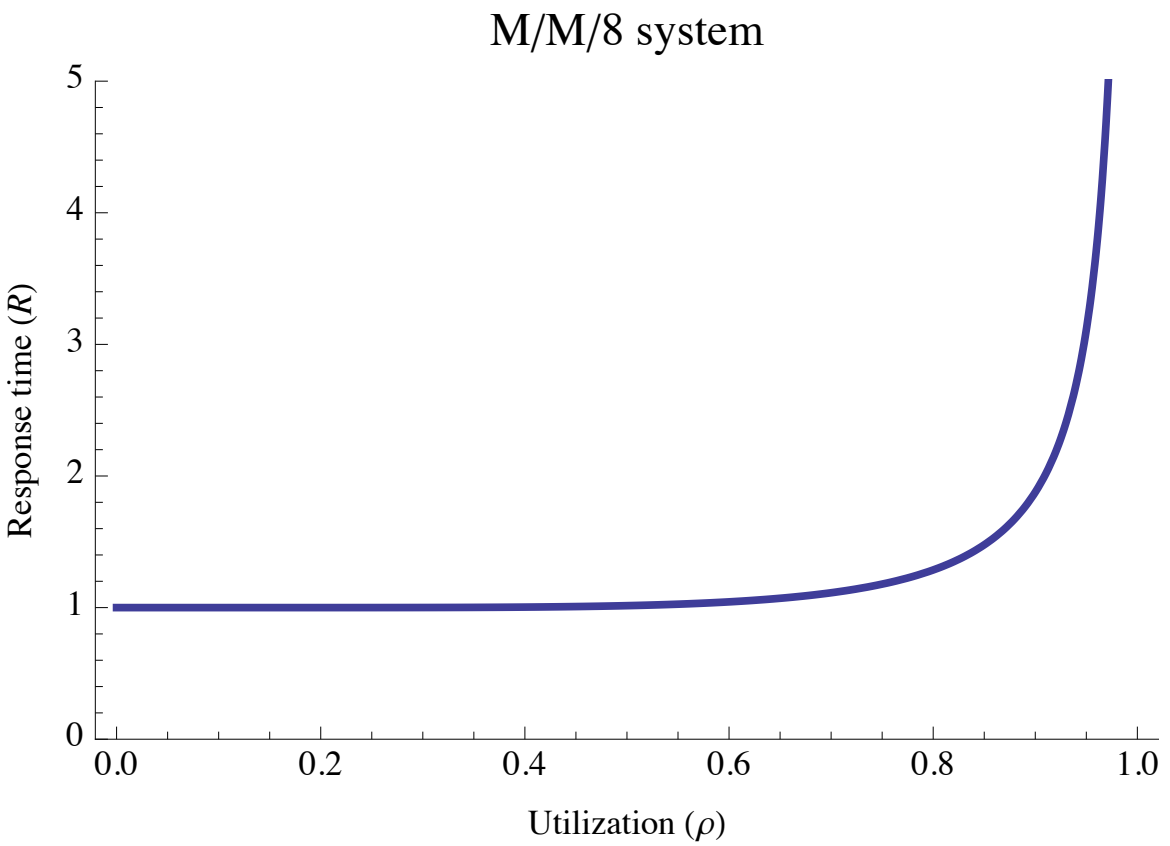
queue·ing de·lay

noun

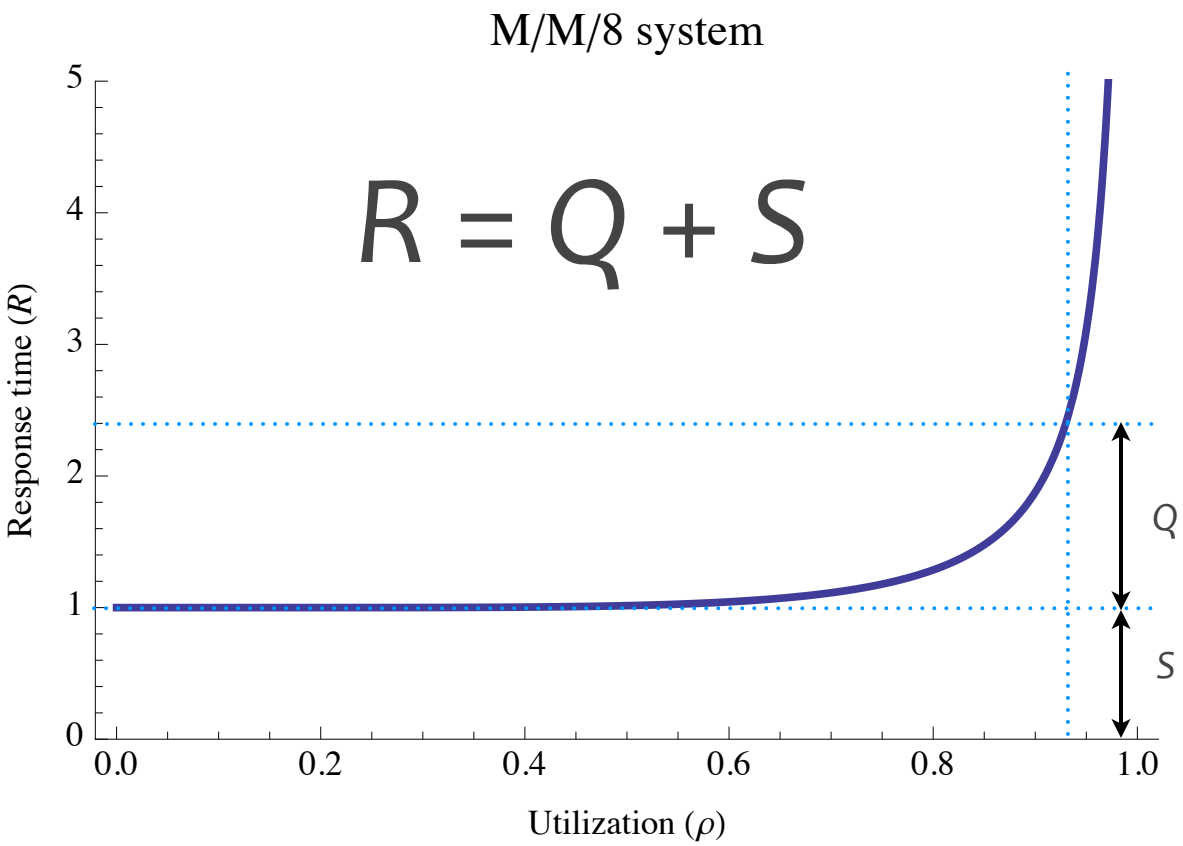
1 time spent waiting in a queue for access to a shared resource.



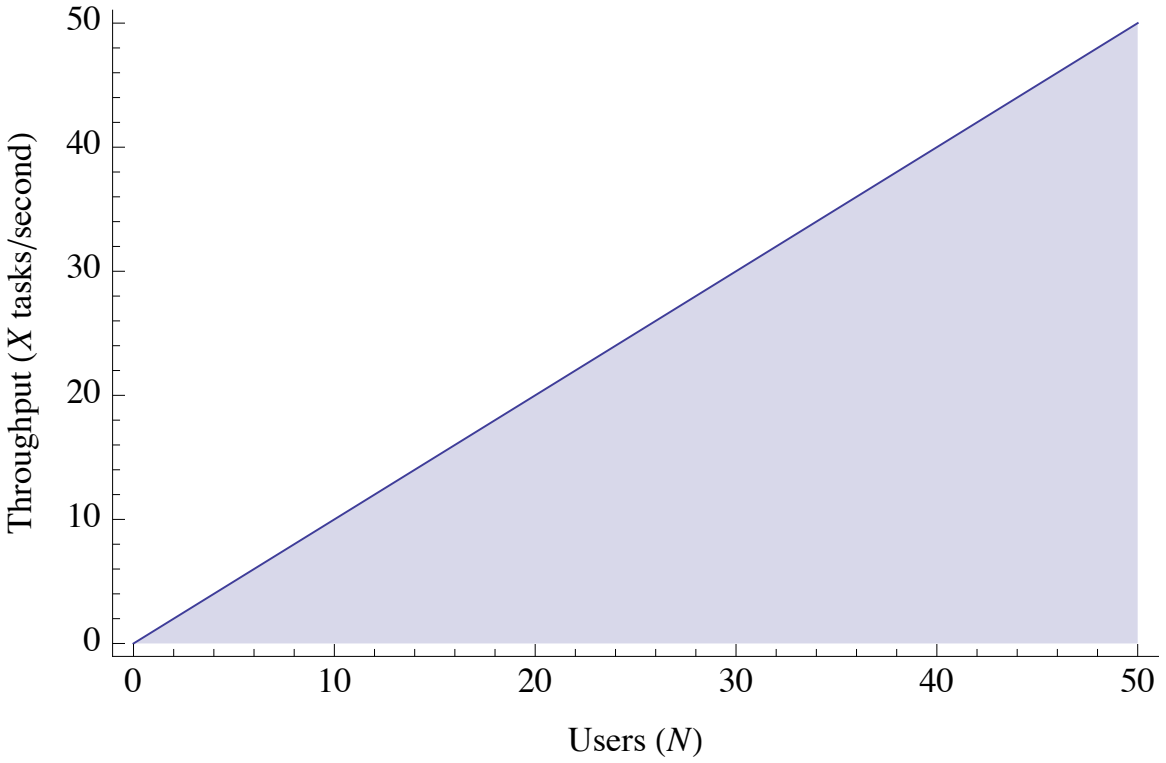
Response time without queueing delay



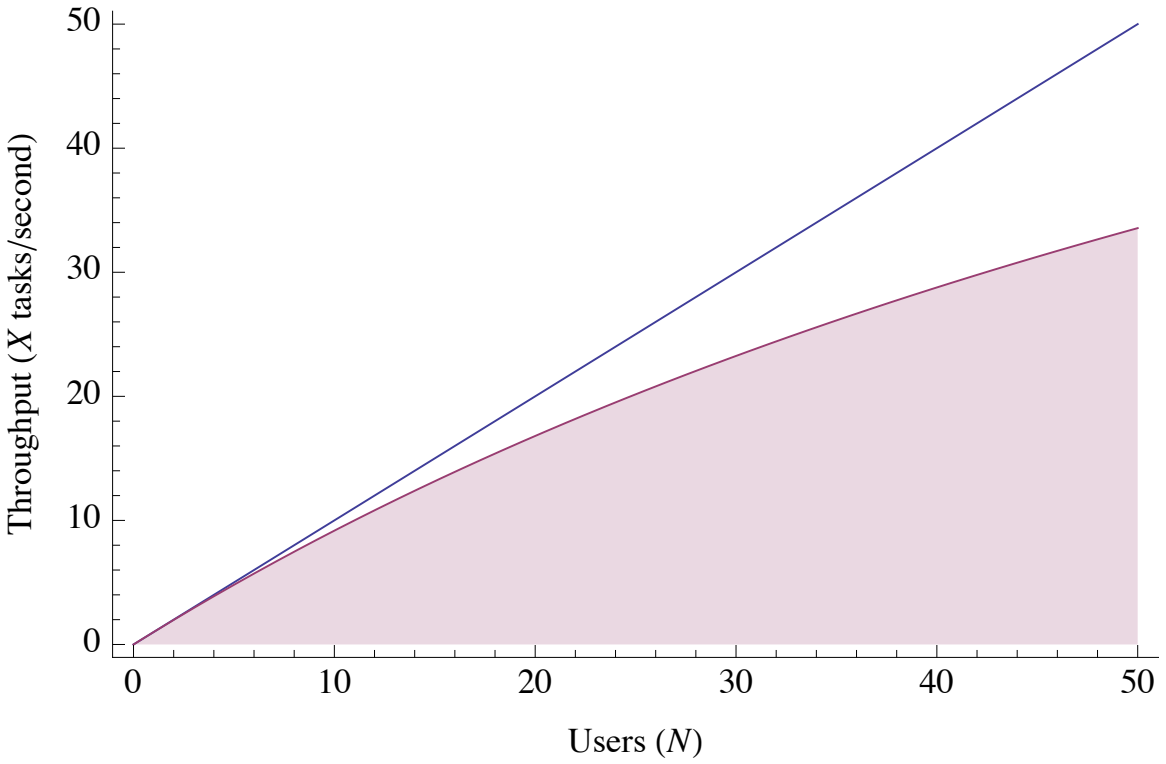
Response time with queueing delay



Response time with queueing delay



Throughput without queueing delay



Throughput with queueing delay

In Oracle…

unaccounted-for …

db file sequential read*

db file scattered read*

…

*unexpectedly high latencies

15. The Knee

Goals:

response time ↘

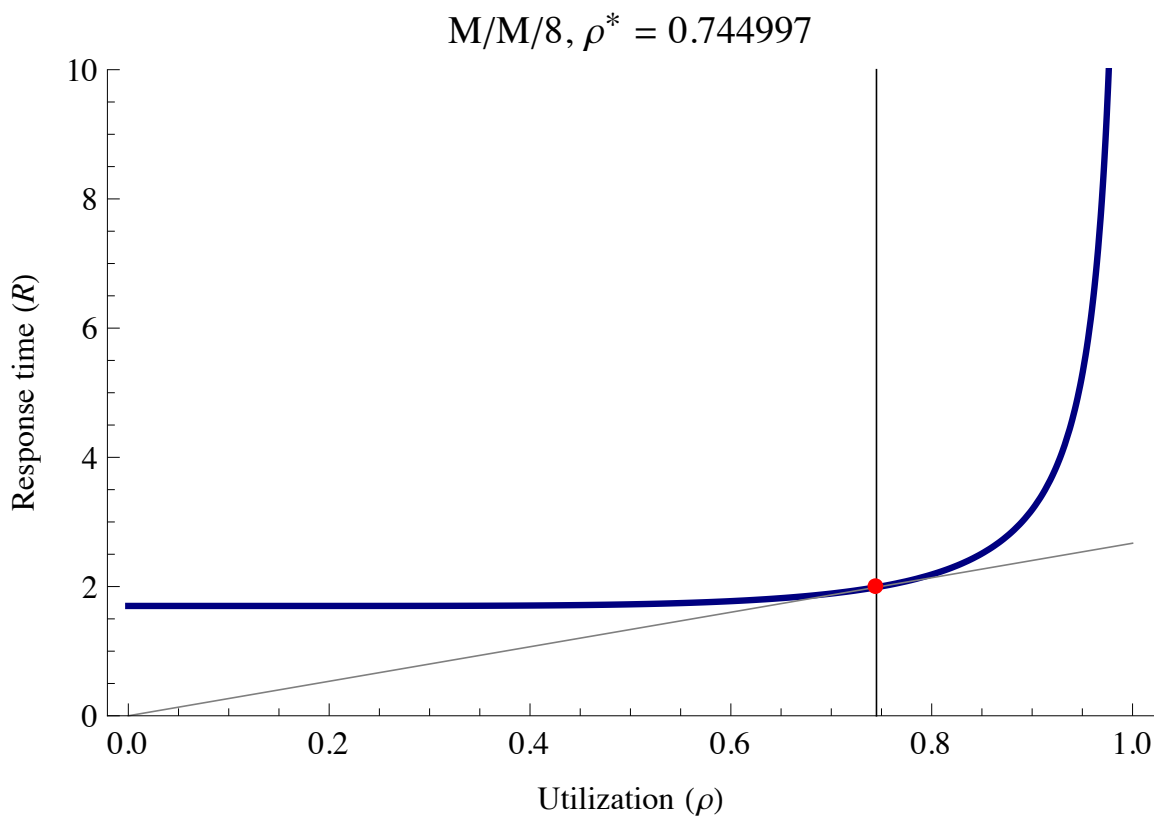
throughput ↗


better
response time
better
throughput

knee

noun

1 the resource utilization value at which throughput and response time are in optimal balance.



The knee is the ρ value at which R/ρ is minimized

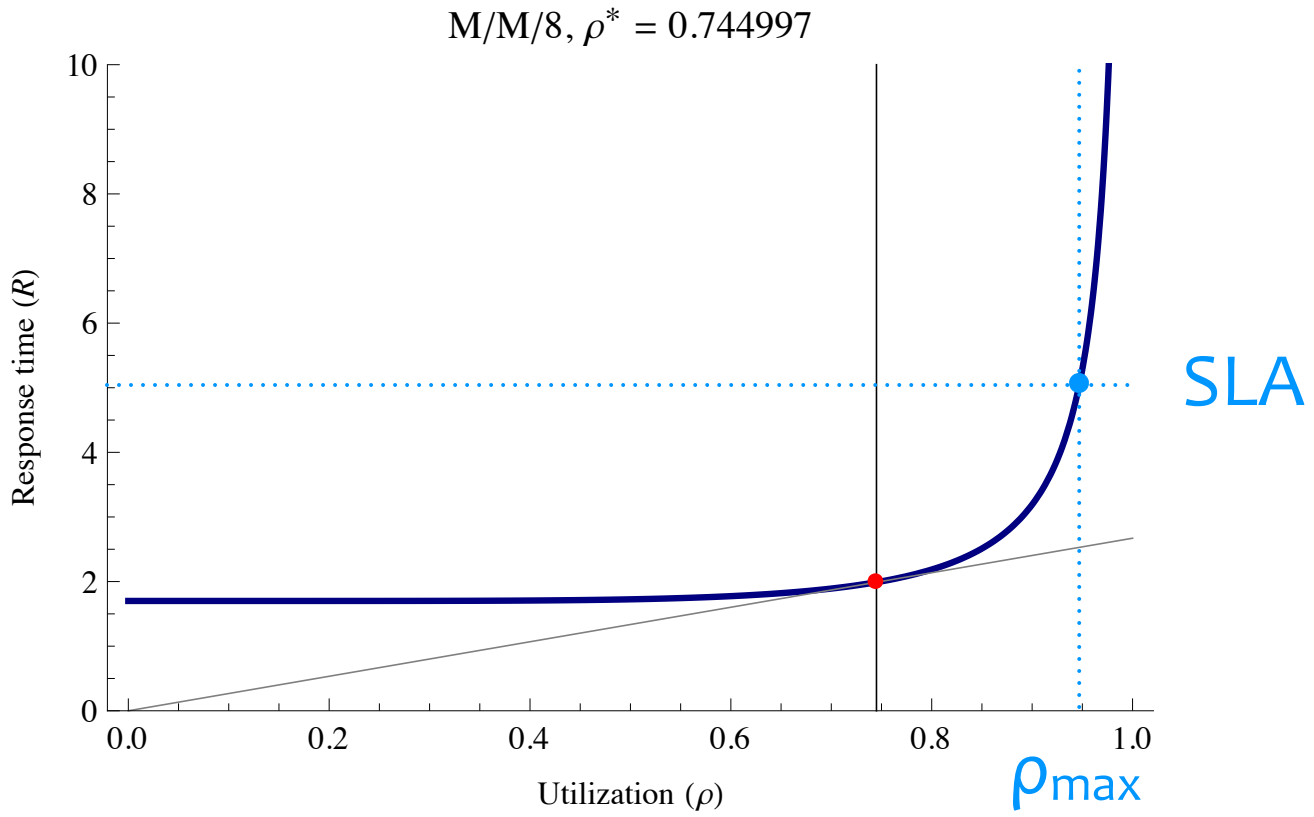
Service channel count	Knee utilization
1	50%
2	57%
4	66%
8	74%
16	81%
32	86%
64	89%
128	92%

Every resource has a knee.

There is no response time knee;
only service level agreements.

—Neil Gunther

"Watch Your Knees and Queues"
at *http://perfdynamics.blogspot.com/2008/03/watching-your-knees-and-queues.html*

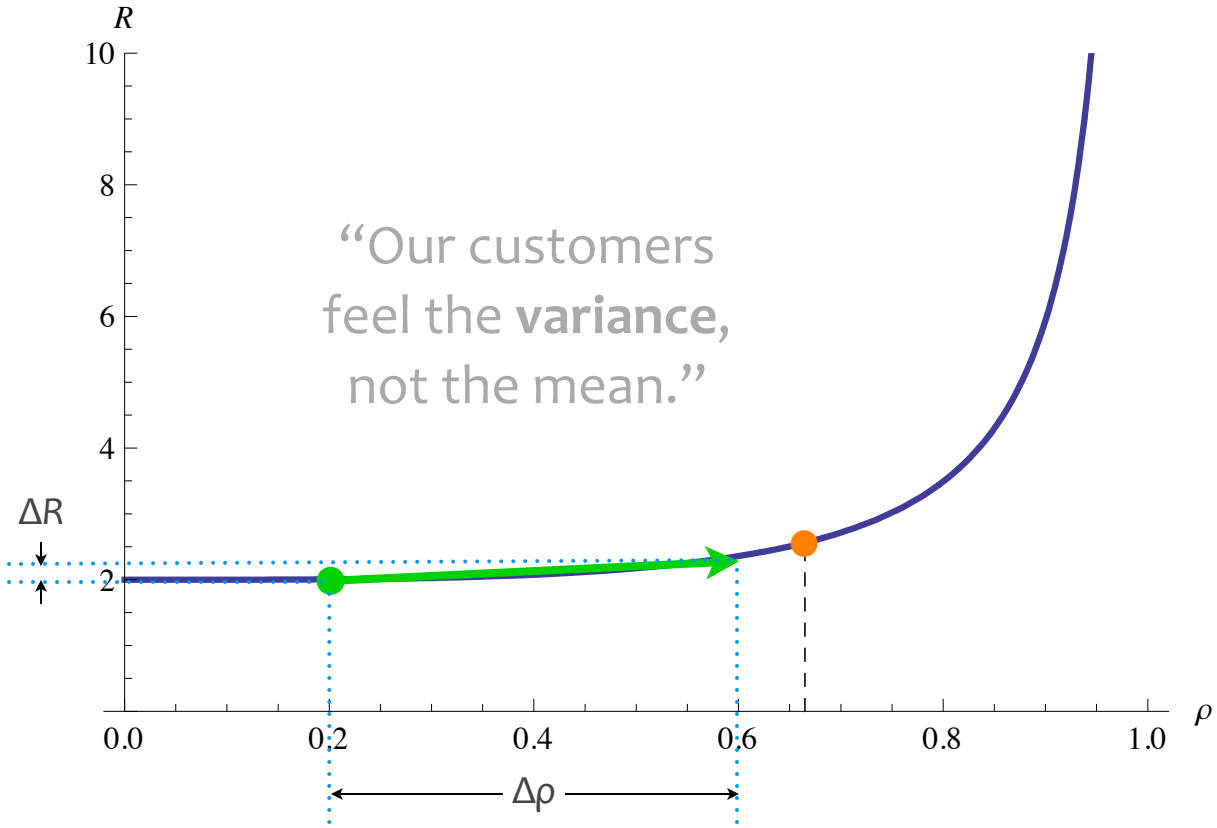


Gunther's suggested ρ_{max} is where R = SLA

16. Relevance of the Knee

$M/M/4, S = 2.$

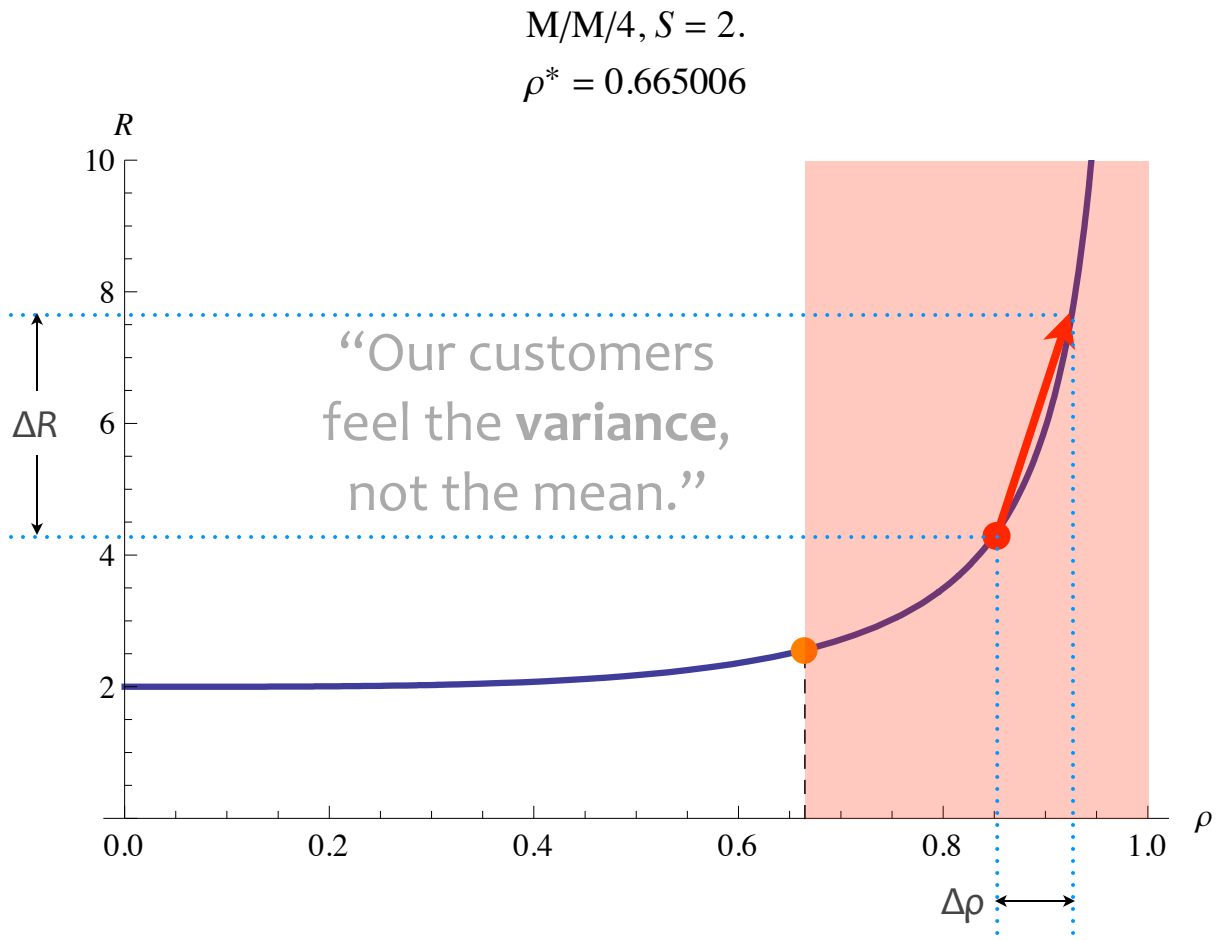
$\rho^* = 0.665006$



"Our customers feel the **variance**, not the mean."

Left of the knee, performance with random arrivals is

stable and **consistent**.



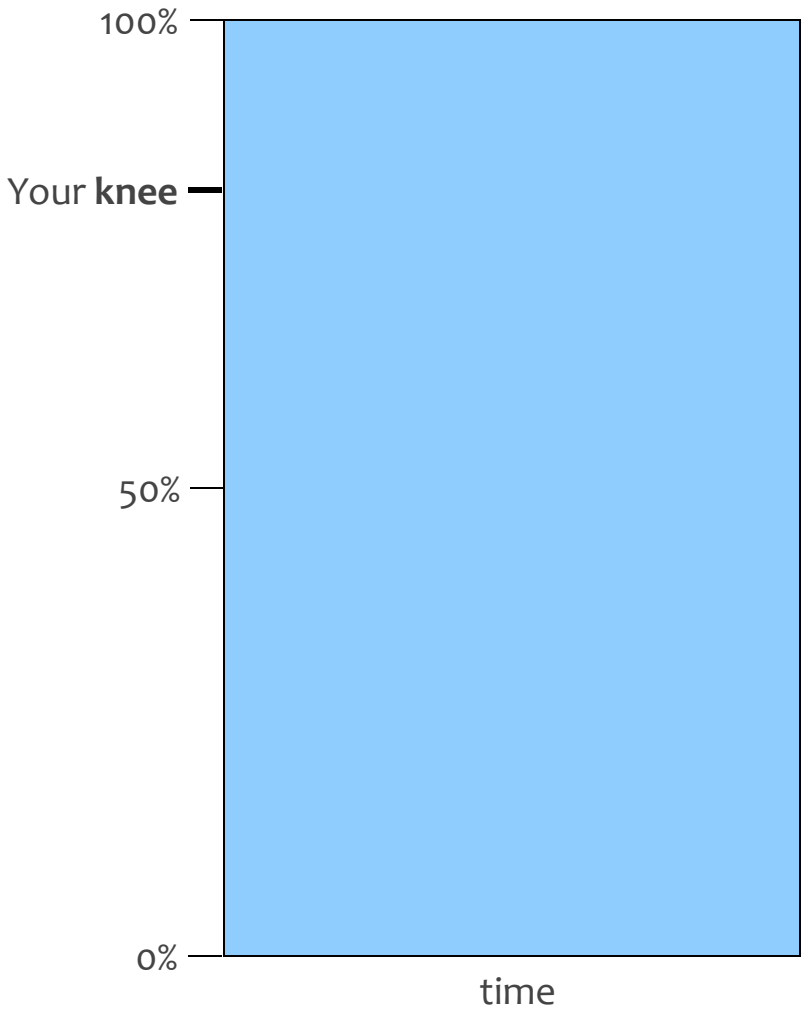
Right of the knee, performance with random arrivals is

unstable and **problematic.**

17. Capacity Planning

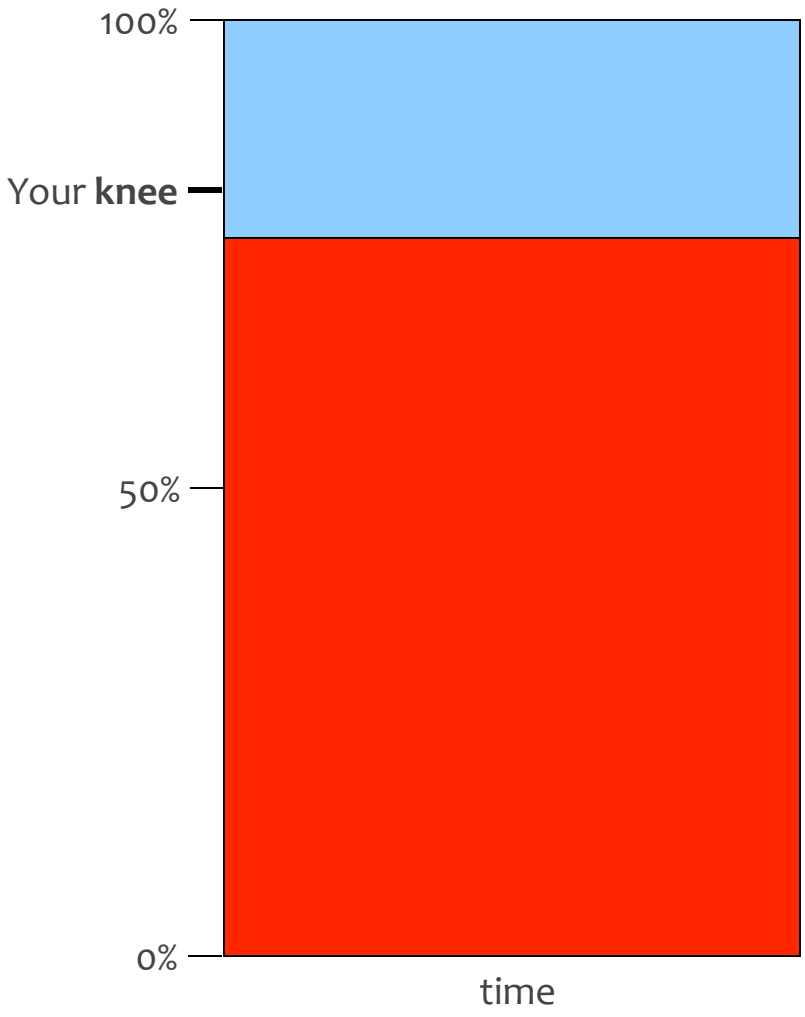
Capacity

for **each** resource on your system

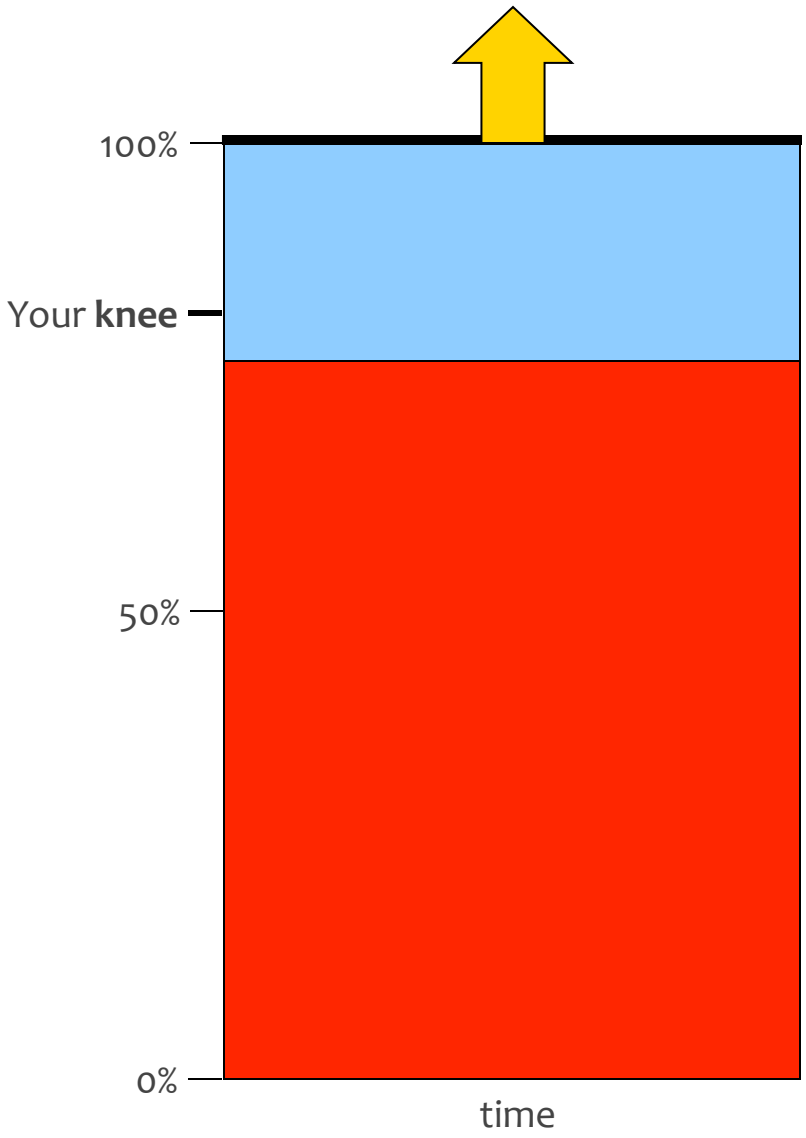


Load

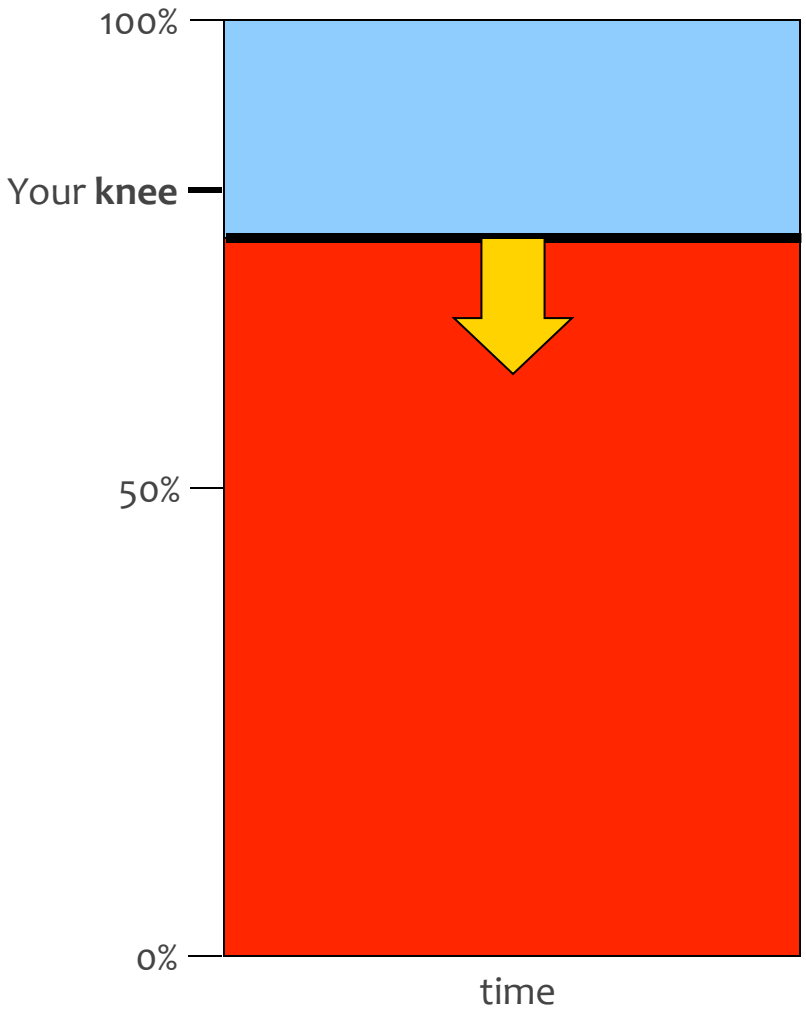
for **each** resource on your system



capacity planning =
"How big must capacity be?"



load management =
"How small must **load** be?"



To **perform well,** you must

manage your **load**

so that **utilizations**

on resources with **random arrivals**

do not exceed their **knees.**

When load exceeds a knee, you need to

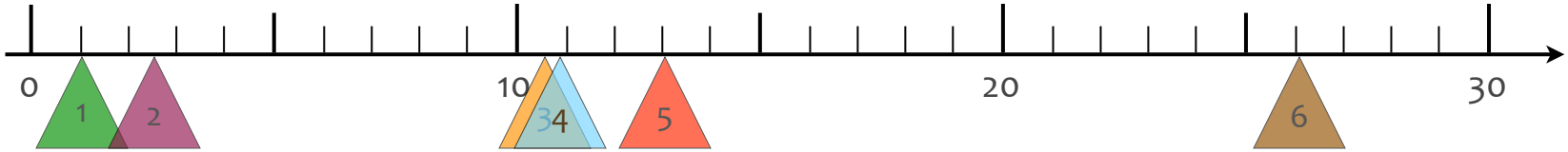
reschedule load,

or **eliminate load,**

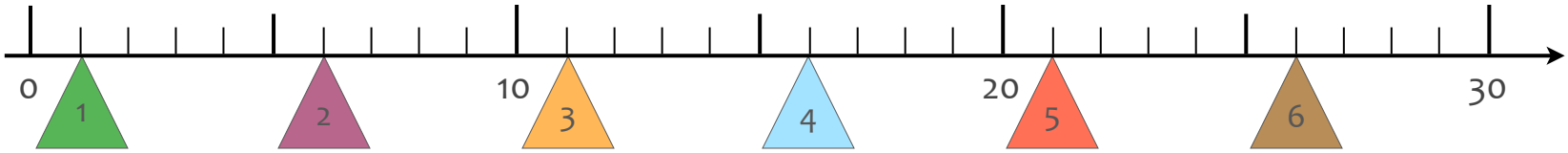
or **increase capacity.**

18. Random Arrivals

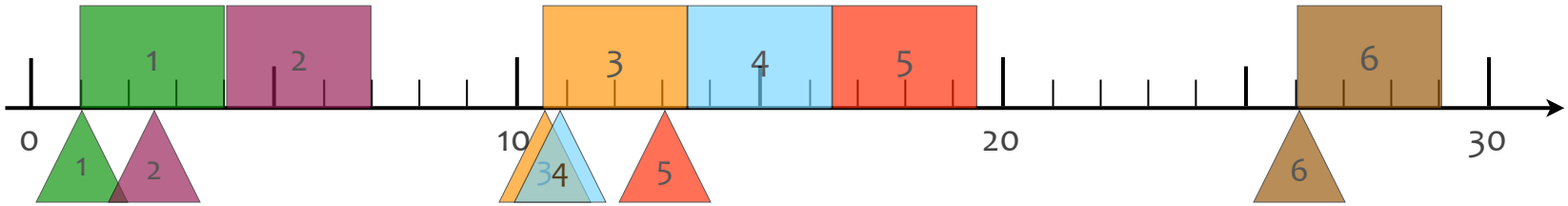
A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec



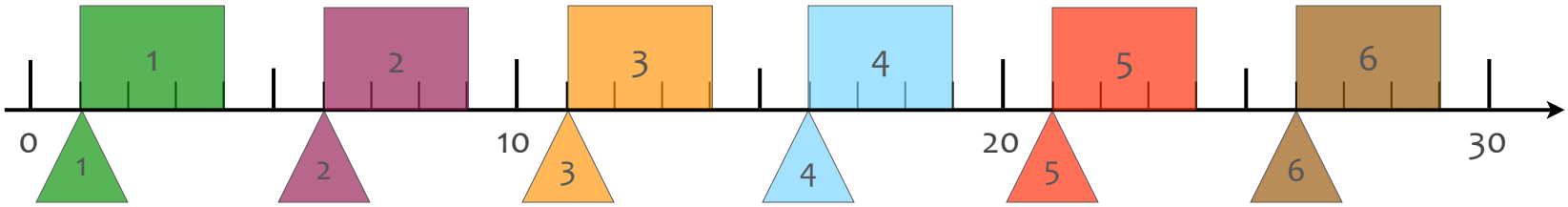
A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec



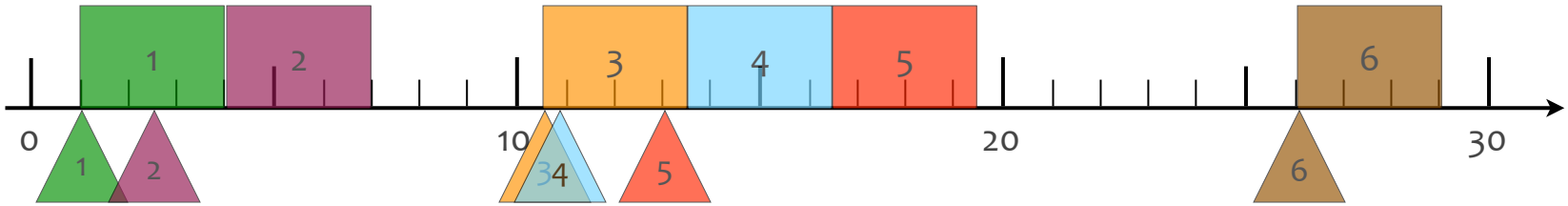
A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec, S = 3 sec, R = 4.267 sec



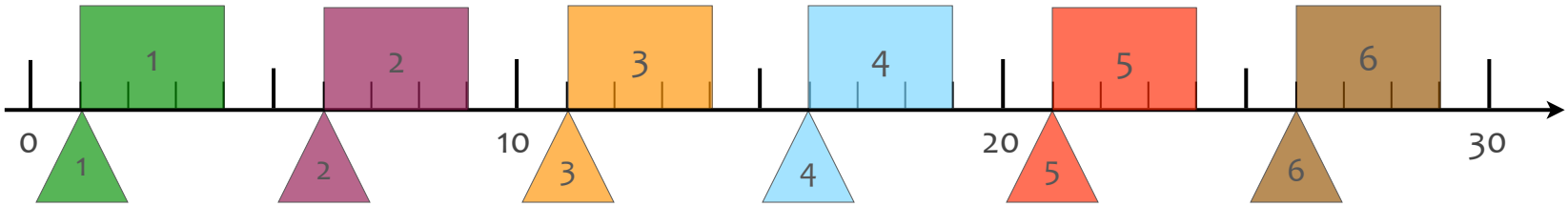
A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec, S = 3 sec, R = 3.000 sec



A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec, S = 3 sec, **R = 4.267 sec**



A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec, S = 3 sec, **R = 3.000 sec**

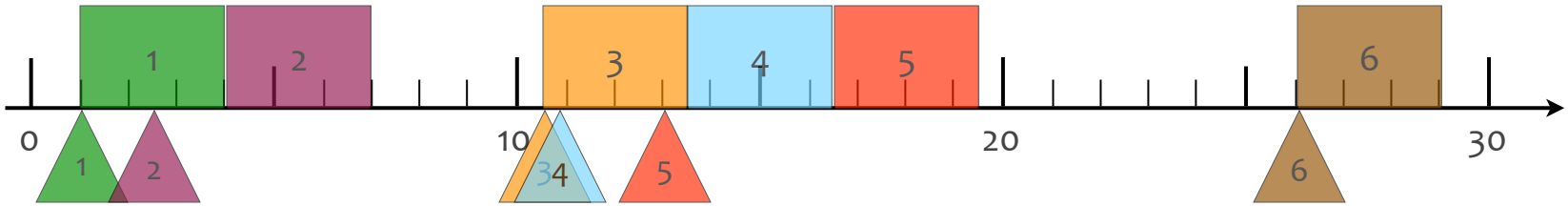


THE FUNDAMENTAL THEOREM OF MEASURING THINGS

When obviously different experiences
yield the same measurement,
you're *measuring the wrong thing*.

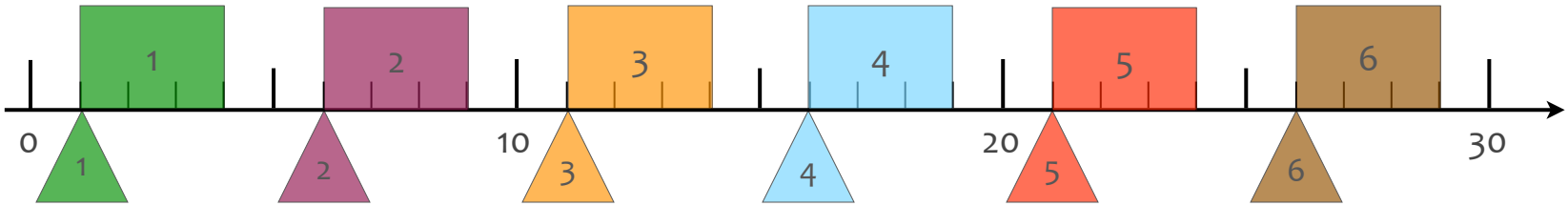
Random arrival process

A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec, S = 3 sec, **R = 4.267 sec**



Deterministic arrival process

A = 6 arrivals, T = 30 sec, λ = .2 arrivals/sec, S = 3 sec, **R = 3.000 sec**



With deterministic arrivals,

you can run up to 100% utilization.

With random arrivals,

you must pay attention to your knees.

19. Coherency Delay

co·her·en·cy de·lay

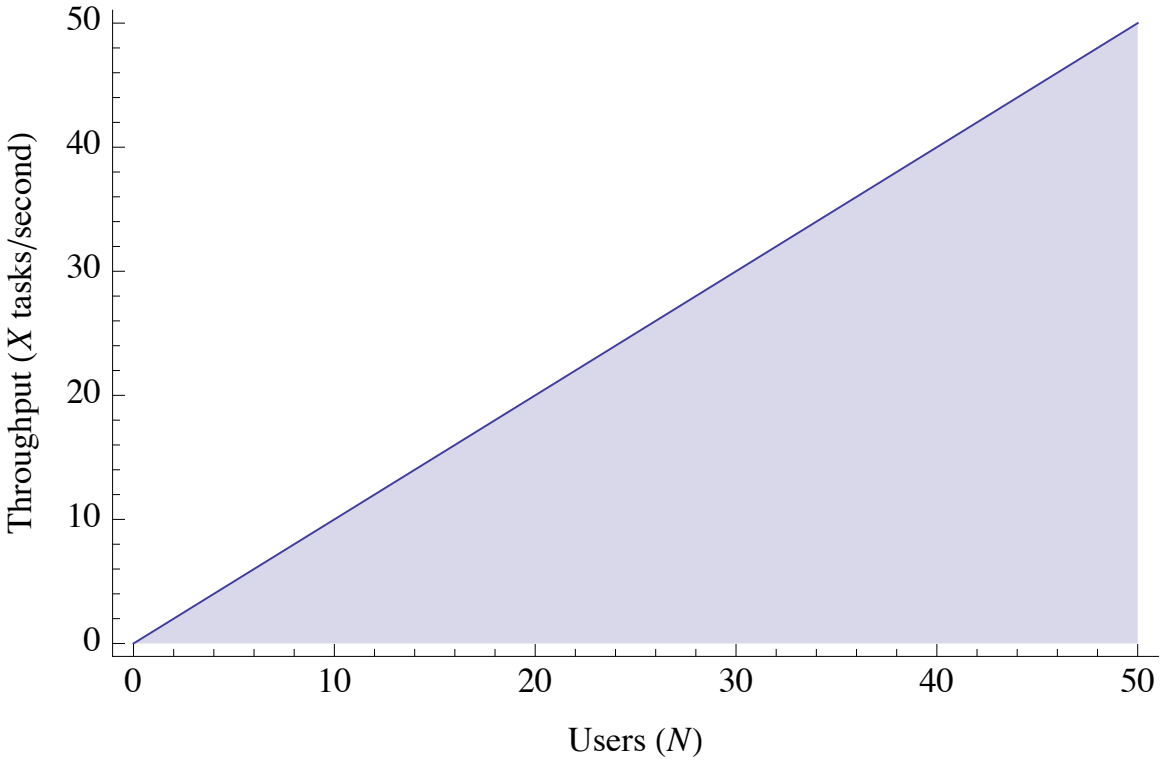
noun

1 time spent communicating and coordinating access to a shared resource.

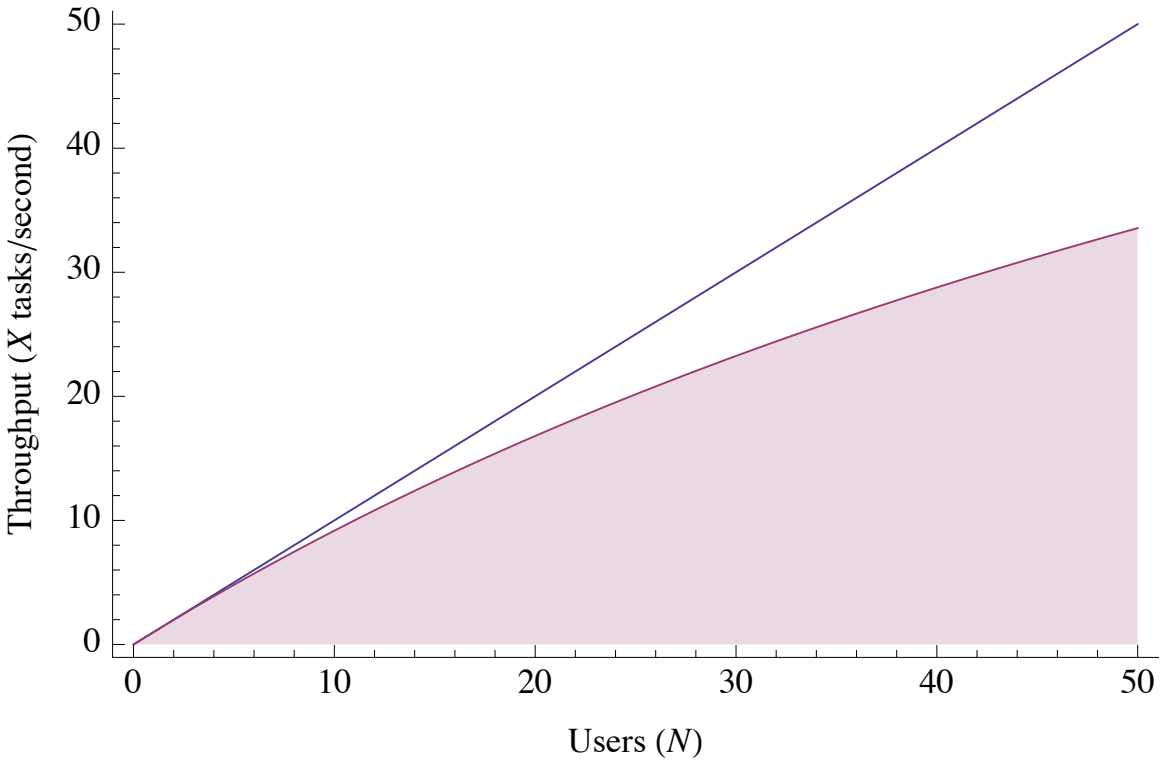
queue·ing de·lay

noun

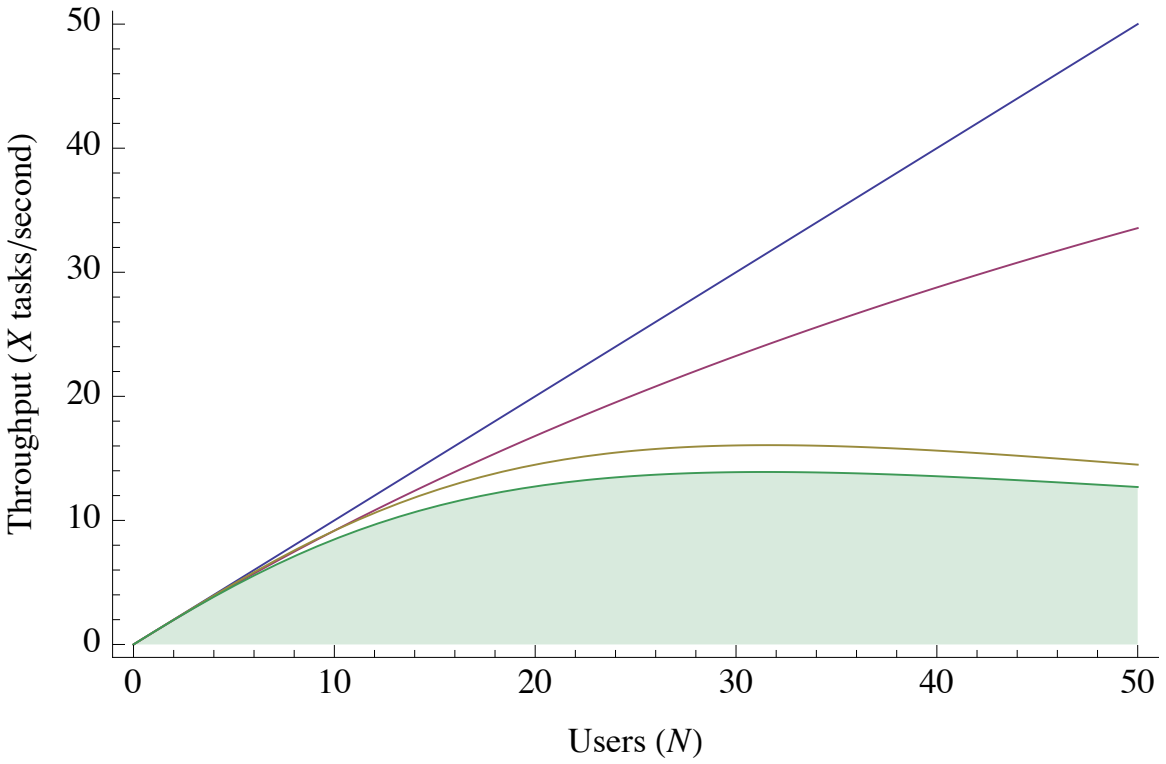
1 time spent **waiting in a queue for** access to a shared resource.



Throughput with perfect scalability



Throughput with queueing delay



Throughput with queueing and coherency delay

In Oracle…

log file sync

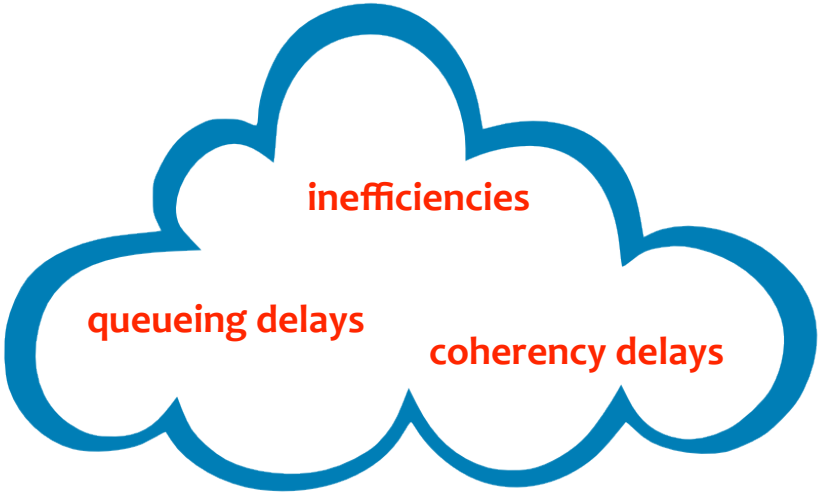
enqueue

latch free

buffer busy

…

20. Performance Testing



How can you **test** for all of them?

1. You will catch **more problems** if you just try.

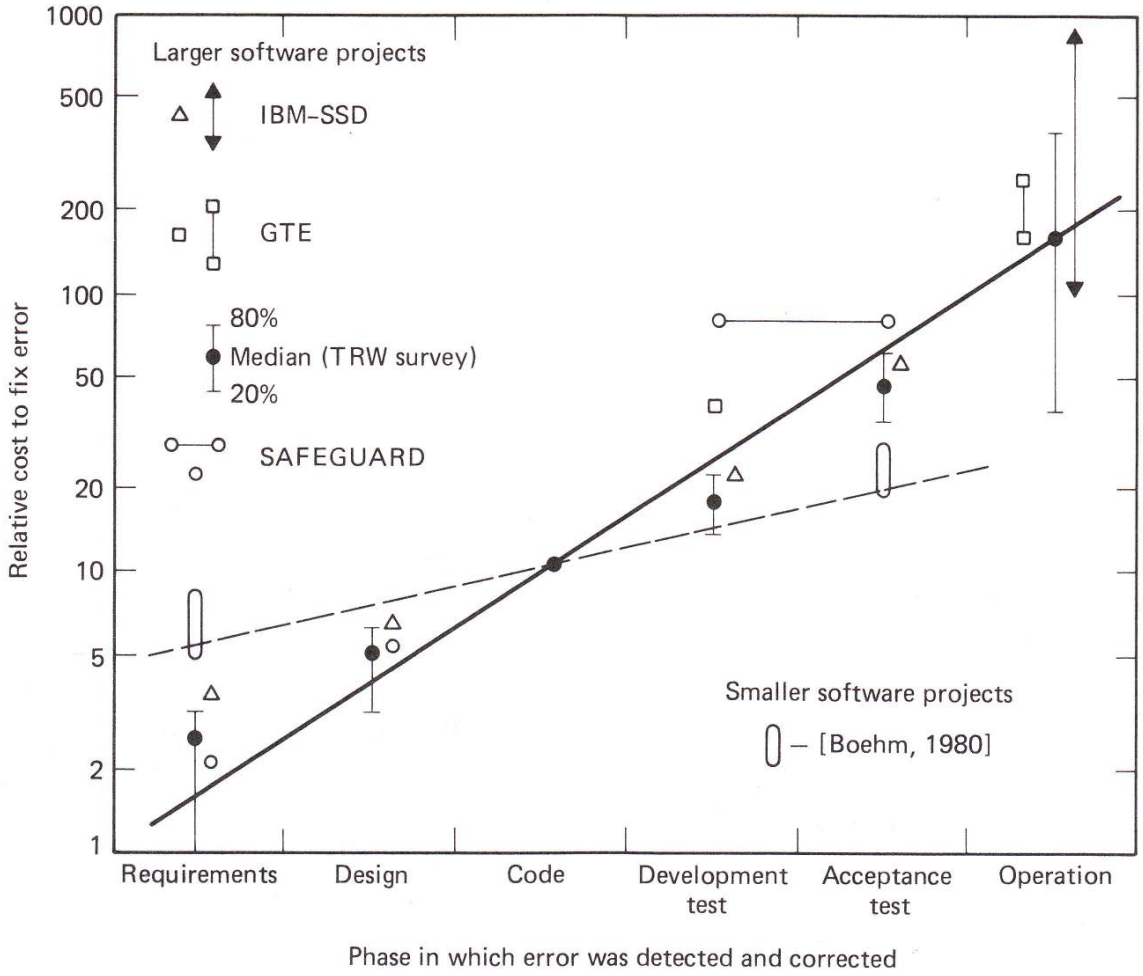


FIGURE 4-2 Increase in cost-to-fix or change software throughout life-cycle

Boehm, B. W. *Software Engineering Economics.* Englewood Cliffs NJ: P T R Prentice Hall, 1981. p40

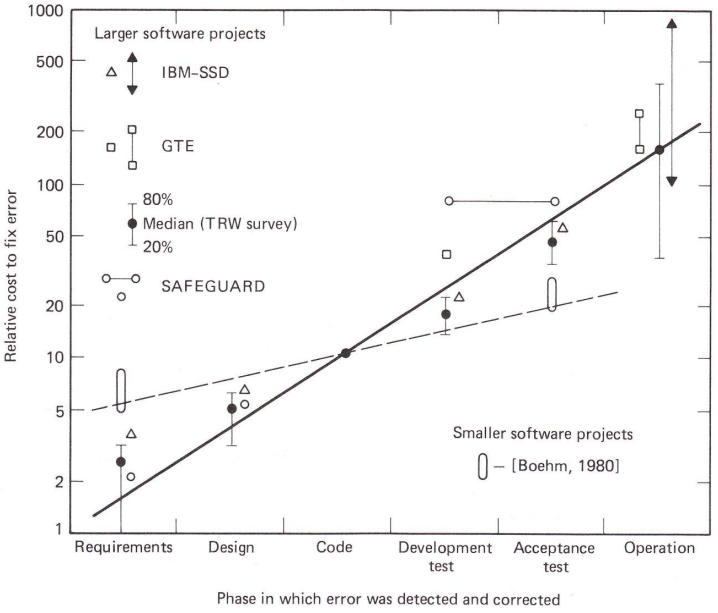


FIGURE 4-2 Increase in cost-to-fix or change software throughout life-cycle

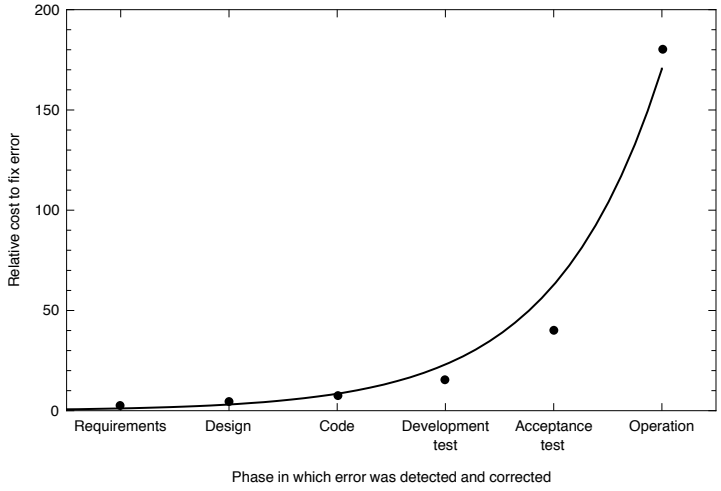


FIGURE 4-2(A) Increase in cost-to-fix or change software (linear scale $y = 0.422778e^{x}$)

Boehm's data on a linear scale

	Bugs fixed per SDLC phase						Relative Cost	Savings
	R 1	D 4	C 10	DT 20	AT 40	O 200		
1						100%	200	0%
2					10%	90%	184	8%
3				10%	10%	80%	166	17%
4			10%	10%	10%	70%	147	27%
5		10%	10%	10%	10%	60%	127	36%
6	10%	10%	10%	10%	10%	50%	108	46%
7	5%	10%	15%	20%	25%	25%	66	67%

You need to try.

1. You will catch **more problems** if you just try.

2. You will never catch them all.

...So you need a plan.

21. Measuring

Performance is not an attribute of a system.

Performance is an attribute of
each individual experience
with a system.

You need to **measure**

individual experiences.

...Remember the percentile conversation.

Surrogate measures aren't good enough.

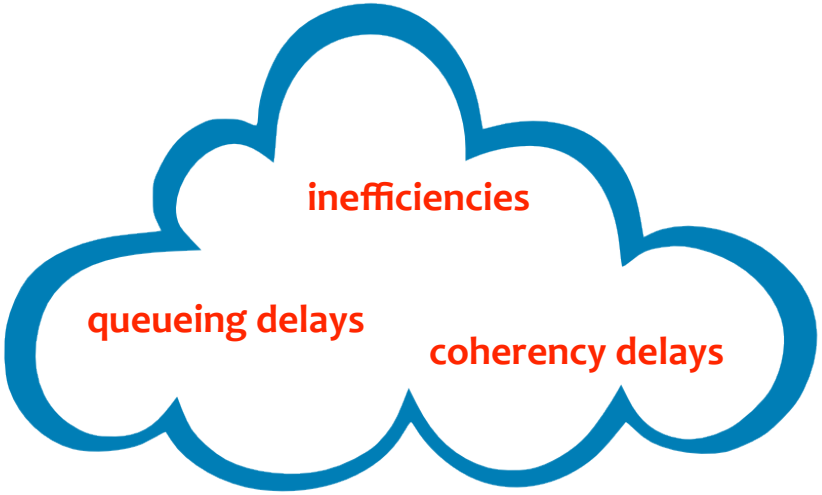
... utilizations latencies hit/miss rates averages ...

Individual experiences are easy to **measure**

when the **application** measures them.

who: user name
what: task name
when: t_0, t_1
where: IP address

22. Performance is a Feature



You **can't know**

how your application will perform until you go live.

You need to **write** your application so it's

easy to **fix** performance

in **production.**

You can.

...Whether you build or buy your software.

tier-specific instrumentation
application instrumentation
Oracle end-to-end tracing
Oracle instrumentation
Method R ILO
triggers

The software designer who integrates performance measurement into his product is much more likely to create a fast application and—more importantly—an application that will become faster over time.

—Cary Millsap
"Thinking Clearly about Performance"
at *http://method-r.com/downloads/doc_view/44-thinking-clearly-about-performance*

@CaryMillsap

method-r.com

Thank you