

Advanced Systems Lab

G. Alonso, D. Kossmann, T. Roscoe

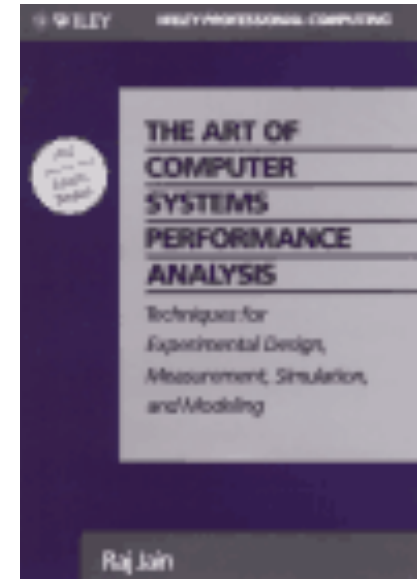
Systems Group

<http://www.systems.ethz.ch>

Overview of the Course

- **Lecture: Tuesdays, 5-7pm, CAB G71**
 - cover material from the text book
 - ~9 sessions (probably done mid November)
- **Project Work (40% of the grade; 100% of the effort)**
 - done in groups of three students
 - Java and C/C++ programming
 - Meetings with supervisors, Thursdays, 5-7pm, CAB E...
- **Exam (60% of the grade)**
 - Sessionsprüfung, details to be announced
- **Web page**
<http://www.systems.ethz.ch/education/courses/hs09/advanced-systems-lab>

Literature



- **RajJain: The Art of Computer Systems Performance Analysis, John Wiley & Sons**
 - (we will not cover Part V „Simulation“)
 - Library has copies. Nevertheless, worth buying.
- **Almost all „Systems“ publications have examples**
 - Networks, operating systems, databases, ...
 - All Master and PhD theses of Systems Group
 - ACM SIGMETRICS: specialized on performance eval.

Lecture Schedule (Planned)

	Lecture	Project
22.9.2009	Introduction	PostGres
29.9.2009	Metrics & Workloads	TPC-H
6.10.2009	Measurements & Logging	Milestone 1
13.10.2009	Distributed Databases	Queues & Middleware
20.10.2009	Statistics	Pencil & Paper
27.10.2009	Experimental Design I	Milestone 2
3.11.2009	Experimental Design II	Distributed Databases
10.11.2009	Queueing Theory I	Pencil & Paper
17.11.2009	Queueing Theory II	Milestone 3
24.11.2009		Pencil & Paper
1.12.2009		Milestone 4
8.12.2009		Pencil & Paper
15.12.2009		Milestone 5

Objective of this course

- Learn how to answer questions of the form...
 - What is better: A or B?
 - Is A good enough?
 - What are the limits of A?
 - How can I improve A?
- Answers are given with the help of experiments
 - learn how to design, implement, and analyze experiments
- A, B, C, ... are computer systems
 - software + hardware
 - often only components of a bigger system
- Almost always the answer is: „It depends!“
 - That is what makes performance eval. interesting.

Examples

- In scope questions (although not well specified)
 - Can we move our web server to the Amazon cloud?
 - Are SSDs better than hard disks?
 - Should I use quick sort instead of merge sort for my online catalogue?
 - Is MySQL fast enough for my online catalogue?
- Out of scope questions
 - Is Gustavo better than Mothy?
 - Those are not computer systems.
 - Should I switch from Oracle to MySQL?
 - Typically, performance not the deciding factor.

How do you answer questions?

- **Real System**

- You implement / install „system(s) under test“ (SUT)
- You run benchmarks and measure observable results

- **Modeling**

- You build a model of the „system(s) under test“
- You calculate results with model

- **Simulation**

- You implement a system that behaves like SUT
- You run benchmarks and measure computed results

How do you answer questions?

- Real System

- You implement / install „system(s) under test“ (SUT)
- You run benchmarks and measure observable results

- Modeling

- You build a model of the „system(s) under test“
- You calculate results with model

- Simulation

- You implement a system that behaves like SUT
- You run benchmarks and measure computed results

Not covered in this class

Real System vs. Modeling

- **Real System**
 - Often expensive to implement
 - Specific to environment (e.g., hardware used)
 - Accurate (quantitative) results
 - Sometimes misleading
- **Modeling**
 - Typically cheap
 - General
 - Qualitative results
 - You always learn something
- **Use modeling whenever you can**
 - Unfortunately, modern systems are too complex

Methodology

1. Ask the right question

- Define the „system(s) under test“
- Define what „better“ means
- Define relevant workloads, understand parameters

2. Make a hypothesis

- „A good scientist predicts the results and explains later why something totally different happened.“

3. Carry out experiment (real system, model)

- Run workloads, measure metrics

4. Report results, analyze results, gotoStep 1

- Give answer to question, possibly refine question

Methodology

1. Ask the right question

- Define the „system(s) under test“
- Def **Difficult** eans
- Define relevant workloads, understand parameters

2. Make a hypothesis

- „A good scientist predicts the results and explains later why something totally different happened.“

3. Carry out experiment (real system, model)

- Run workloads, measure metrics

4. Report results, analyze results, gotoStep 1

- Give answer to question, possibly refine question

Making a Hypothesis

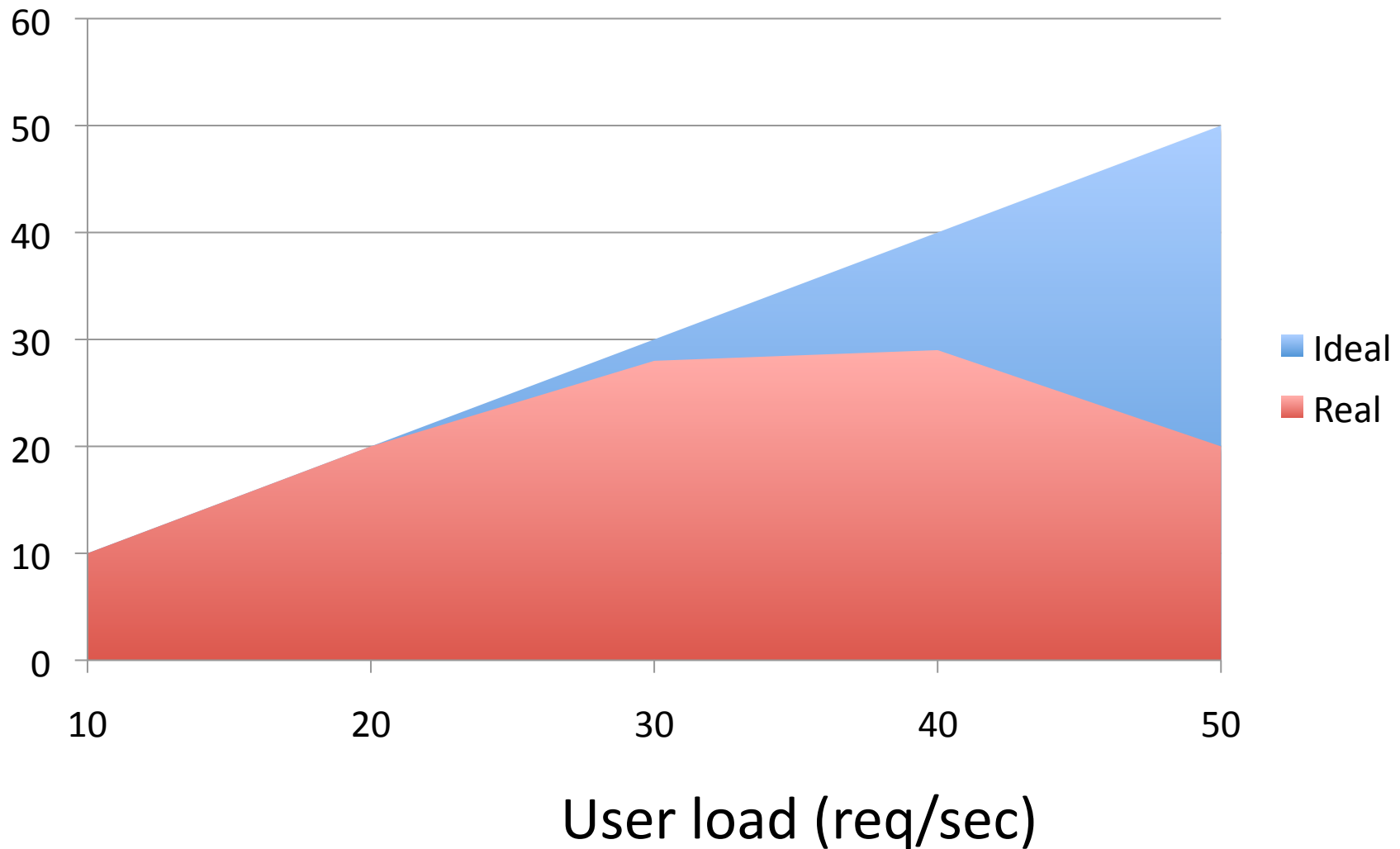
- Use the same format as the final results
 - Draw graphs with expected results
 - Even try to predict variance and statistical properties
 - Make bullet points with explanations
 - Use „modeling“ to make hypothesis
- Share hypothesis with your customer (advisor)
 - Validates whether you are asking the right question
 - i.e., can you make decisions if results turn out like that
- Comparison of expected vs. real results
 - Essential to find bugs in your experiments
 - Essential to understand real results

Example

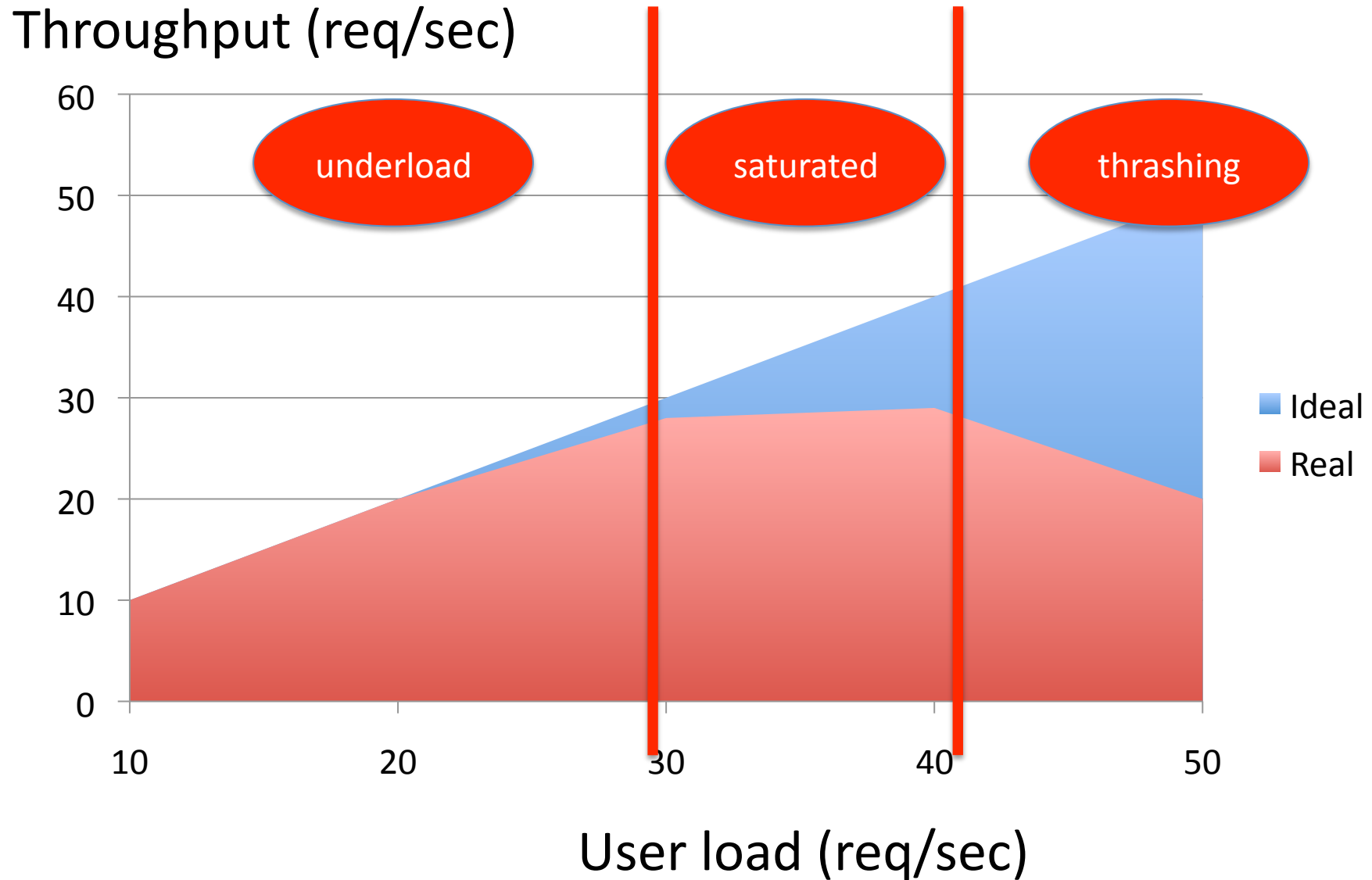
- **Metric 1: Throughput (y-axis of graphs)**
 - requests completed per unit of time (secs)
 - count only “successful” requests (no error, < timeout)
- **Metric 2: Response Time (y-axis of graphs)**
 - max/min/avg time (secs) to complete a request
- **Parameter: User Load (x-axis of graphs)**
 - number of requests arriving per unit of time (secs)

Example: Throughput

Throughput (req/sec)

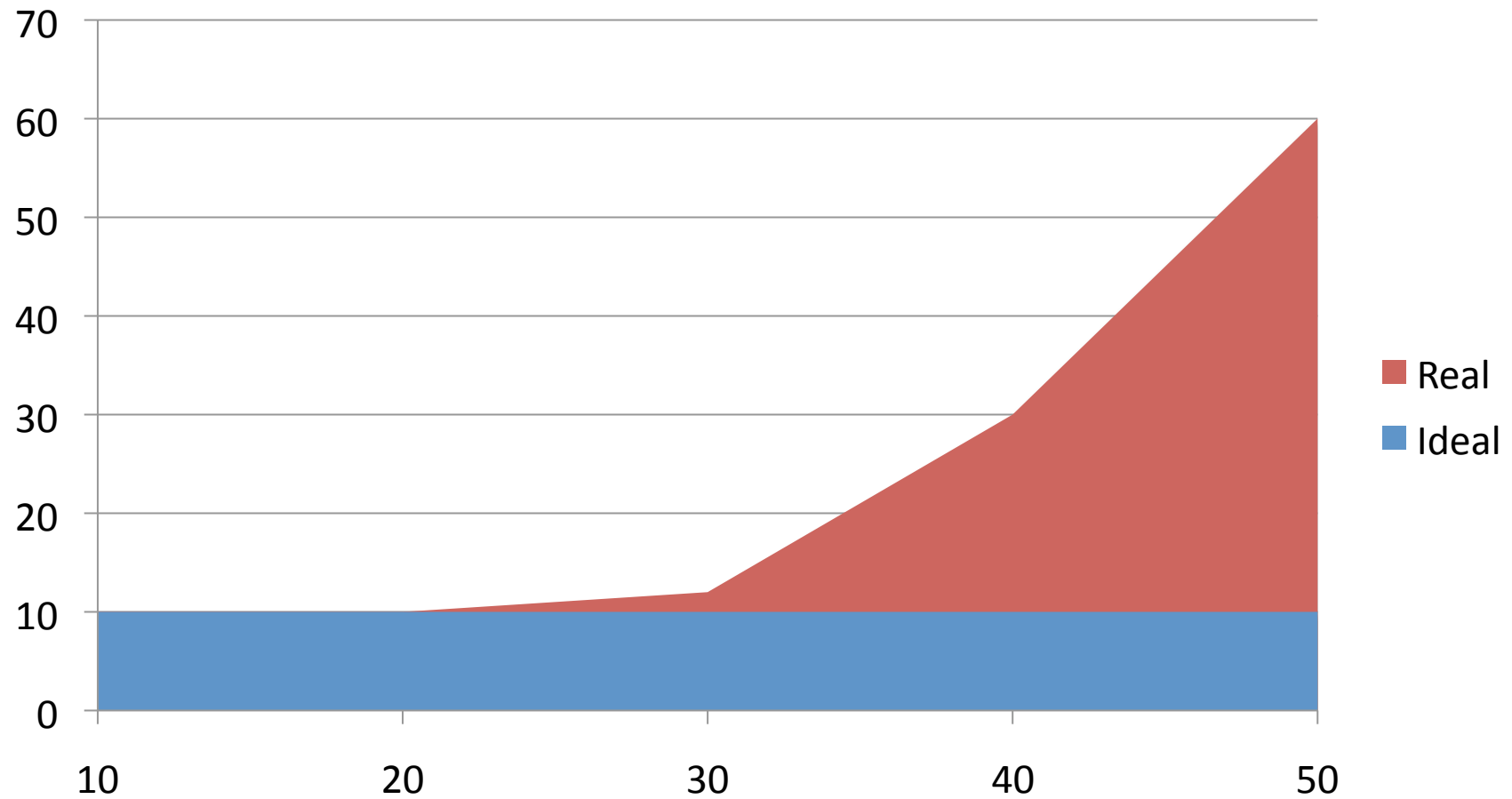


Example: Throughput Analysis



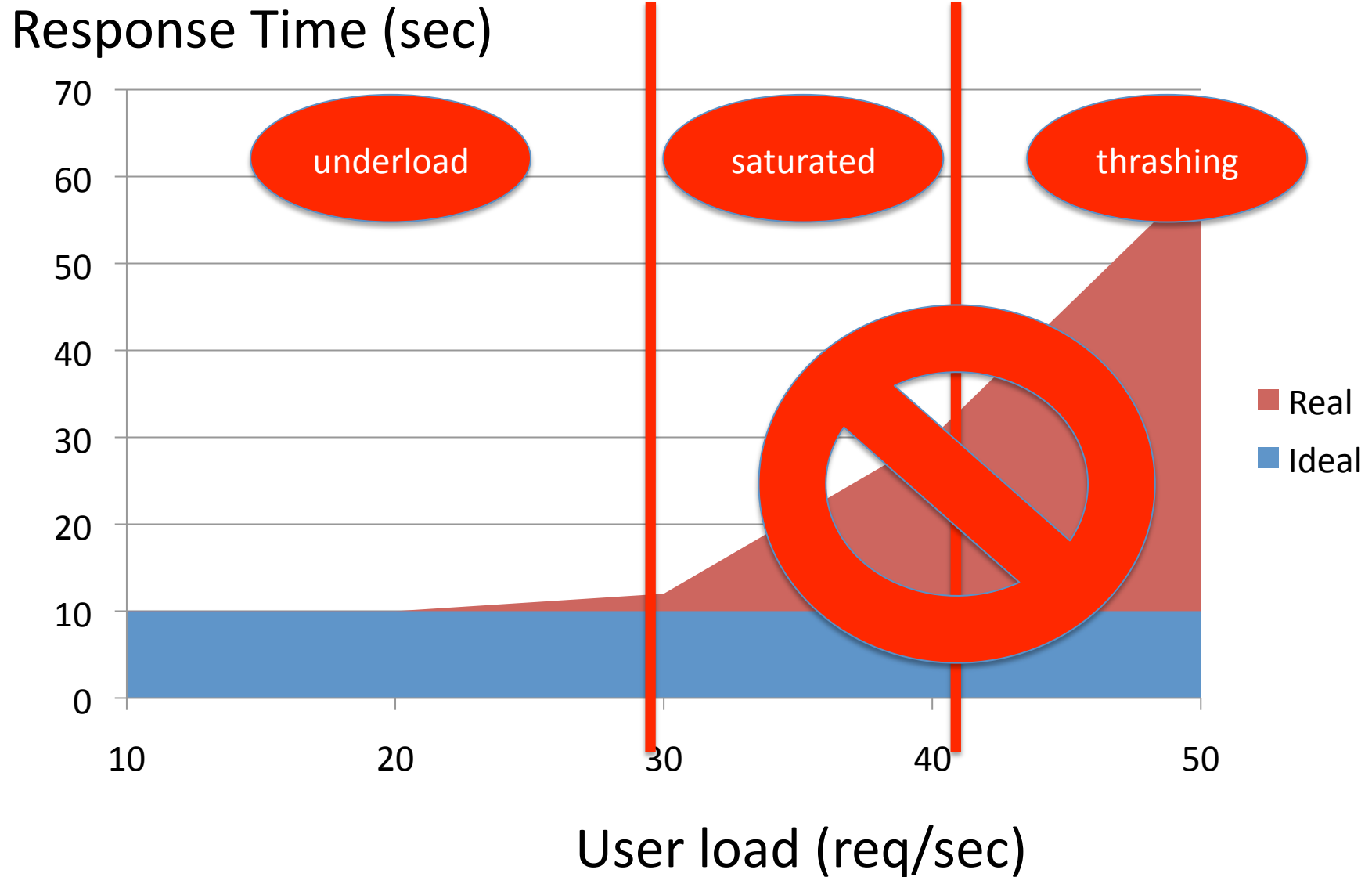
An Example: Avg. Response Time

Response Time (sec)



User load (req/sec)

An Example: Avg. Response Time



Example Summary

- **Throughput <> Response Time**
 - Under certain conditions (closed system; #req is const),
throughput = $1 / \text{avg. response time}$
 - But, in general, this is not true (#req vary)
 - More on that topic later (Queuing Theory)
- **Do not measure Response Time in „overload“**
 - Because of queueing, can become arbitrary high
 - One way of cheating: If A has ϵ higher Tput than B
 - Run system at user load of $\text{Throughput}(B) + \epsilon$
 - Measure response time of A and B 😊

How to lie with statistics

Absolute Throughput

System	Workload 1 (tps)	Workload 2 (tps)	Average (tps)
System A	20	10	15
System B	10	20	15

Relative Throughput: Baseline B

System	Workload 1 (rel.)	Workload 2 (rel.)	Average
System A	2x	0.5x	1.25x
System B	1x	1x	1x

System A looks better although it is not!

How to lie with statistics



Common Mistakes (see textbook)

- No Goal
- Biased Goal
- Unsystematic Approach
- Analysis without understanding problem
- Incorrect metrics
- Unrepresentative workload
- Wrong evaluation technique
- Overlooking params
- Ignoring factors
- Inappropriate design
- Inappropriate level of detail
- No analysis
- Erroneous analysis
- No sensitivity analysis
- Improper treatment of outliers

Common Mistakes (see textbook)

- Assuming no changes over time
- Ignoring variability
- Too complex analysis
- Improper presentation of results
- Ignoring social aspects
- Omitting assumptions and limitations

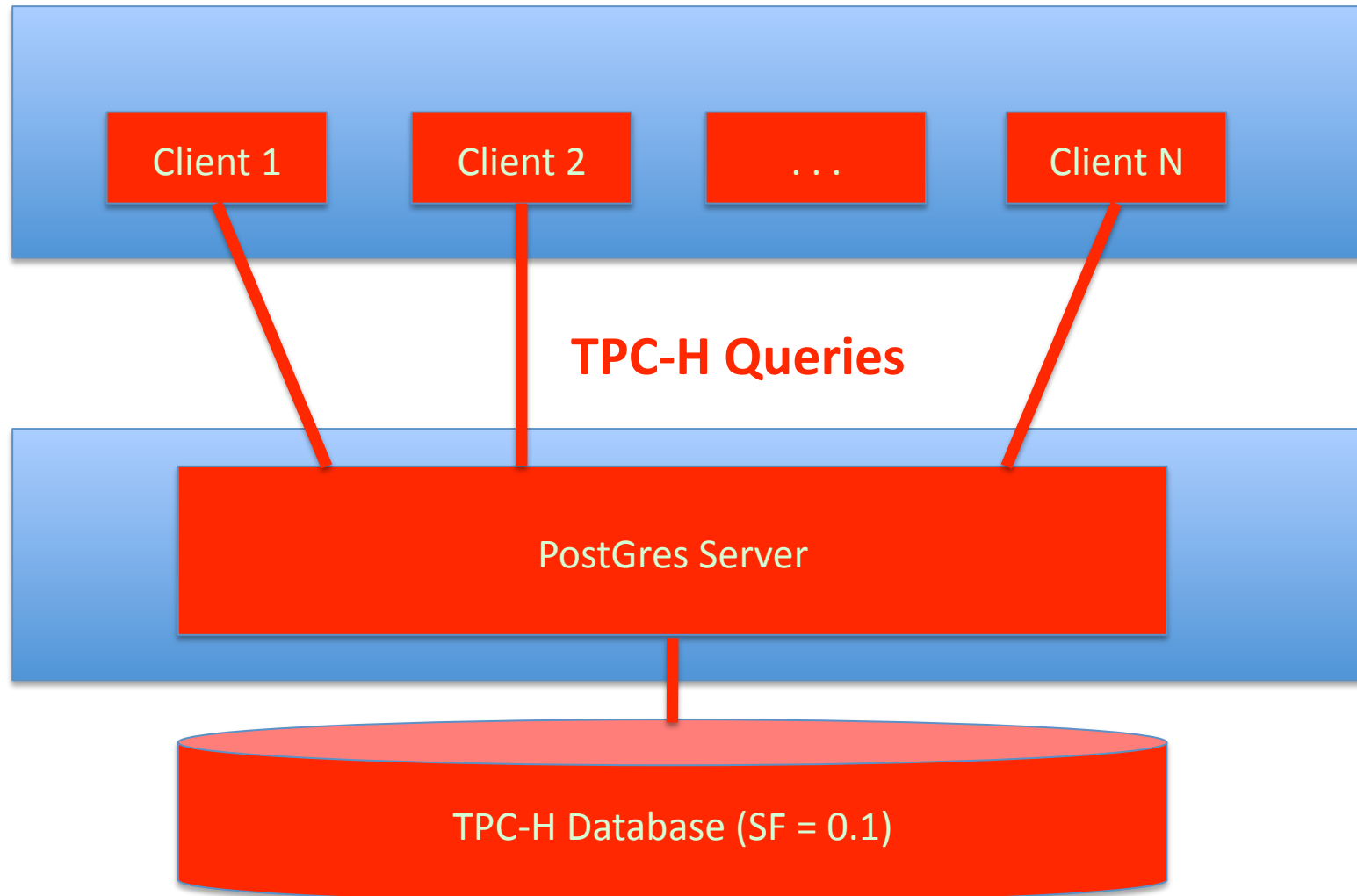
How to become a scientist

- Understand the subject area
 - E.g., lectures on „systems“
- Learn your math (lectures, pencil & paper)
 - Statistics
 - Modeling (Queueing Models)
 - Example designs
- Experience, experience, experience
 - Project as part of this class
 - Lab projects, Master thesis, ...
- Common sense 😊

Project of this course

- Measure standard components (M1 – M3)
 - PostGres database system
- Use standard benchmarks (M1 – M3)
 - TPC-H
- Measure your own code (M2, M3)
 - Workload splitter, queue
 - Bottleneck analysis
- Measure different configurations (M3)
 - Single vs. multiple databases
- Modeling vs. Real System (M4, M5)
 - Limits of modeling, insights of modeling

Milestone 1: Set Up



Milestone Results

- Measure
 - Queries per second (varying user load)
 - Max., min, median response time (varying user load)
- Study different workloads
 - Only Query 1 of TPC-H benchmark
 - All queries of TPC-H benchmark
 - All queries and update functions of TPC-H benchmark
- What questions do these results answer?