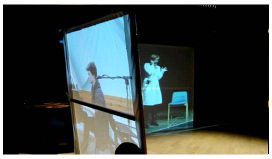
IN5060

Performance in distributed systems

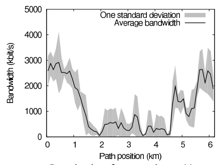
autumn course

UiO : University of Oslo

What is performance?

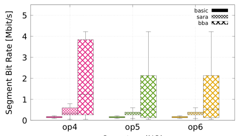


Stage performance
World Opera Production – Dec 2011 @ Tromsø



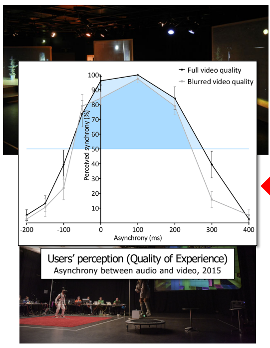
Download performance by position
HTTP Adaptive Streaming measured on Bygdøy Ferry, 2011

Stage performance
Third Life Project@WUK – Oct 2015 @ Vienna



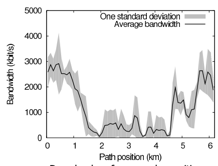
Download performance by operator & algorithm
HTTP Adaptive Streaming, MONROE nodes, 2018

What is performance?

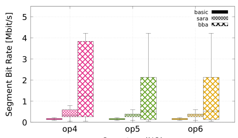


Users' perception (Quality of Experience)
Asynchrony between audio and video, 2015

Stage performance
Third Life Project@WUK – Oct 2015 @ Vienna



Download performance by position
HTTP Adaptive Streaming measured on Bygdøy Ferry, 2011

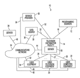


Download performance by operator & algorithm
HTTP Adaptive Streaming, MONROE nodes, 2018

Performance in Distributed Systems

Engineers and researchers solve quantifiable challenges

Idea → Simulation Study → Prototype → Product → Deployment → Feedback

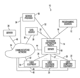


NETFLIX
amazon Prime

Performance in Distributed Systems

Engineers and researchers solve quantifiable challenges

Idea → Simulation Study → Prototype → Product → Deployment → Feedback



NETFLIX
amazon Prime

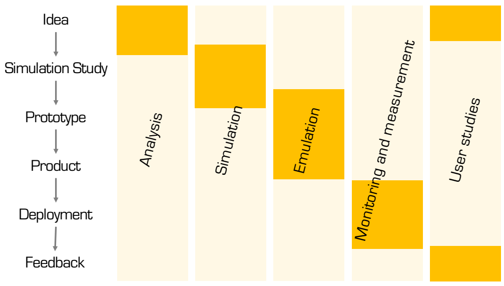
Each step requires a performance assessment

- argue for feasibility
- demonstrate practicality
- study in a context
- measure in the real world
- assess value / success

Performance Evaluation

Performance in Distributed Systems

Engineers and researchers solve quantifiable challenges

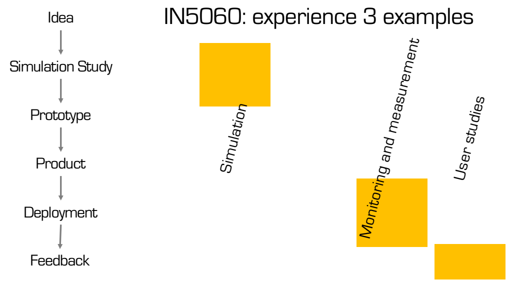


Performance Evaluation

Performance in Distributed Systems

Engineers and researchers solve quantifiable challenges

IN5060: experience 3 examples



Performance Evaluation

Performance in Distributed Systems

Designing and conducting studies

- pre-considerations
- avoiding bias
- measurement points and methods
- data reduction
- drawing conclusions

Presentation and reporting

- formulating a message
- selecting relevant factors
- extracting and interpreting statistics
- dimension reduction
- selecting presentation modes

Specific considerations

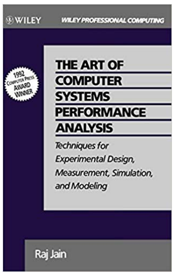
- simulation
- monitoring and measurement
- user studies

Performance in Distributed Systems

- This course is meant to provide you with a taste of the skills needed to become a good system analyst.
- It will provide you with hands-on experience in system evaluation
- It will (to some extent)
  - confront you with the tradeoffs encountered when analysing real systems
  - confront you with the error sources and red herrings encountered when analysing real systems

Performance in Distributed Systems

- The course is based on the book "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling" by Raj Jain



- Reading the book is not mandatory for the course or even necessary to complete, but if you have a chance to read it in full, **do so!**

System performance analysis

Who is interested in system performance analysis?

- The HW designer (company) wants to show that their system is The Best and Greatest system of All Time
- A software provider wants to show that their application is superior to the competition
- The researcher wants to publish her papers, and needs to convince the reviewers that their research improves on the state- of-the-art
- The system administrator or capacity planner needs to choose the system that is best suited for their purpose
- The enthusiast who wants to see if the newest rage from *<insert favourite multinational corporation>* is real, or fake news

System performance analysis

- How do they achieve this?
  - By providing a comparison between their own system and "the competition"
  - The results need to be *(or appear)* convincing to the target audience
  - This comparison is made through proper system performance analysis

- The techniques of models, simulations and measurement are all useful for solving performance problems
  - IN5060 will focus on experimental design, simulation, measurement and analysis
  - For modelling try for instance: MAT-INF3100 - Linear Optimisation

Theory and practice

- Theory / models will provide us with candidates for system optimisations
- Deploying them in reality may in many cases lead to unforeseen results
  - Hardware differences
  - Non-deterministic systems
  - Unexpected workloads
- Key techniques needed
  - Mathematical analysis
  - Simulation
  - Emulation
  - Measurement
  - User studies
  - Measurement techniques (monitors)
  - Data analysis (statistics and presentation)
  - Experimental design

Performance in distributed systems

Key skills of performance analysts

Key skills needed – evaluation techniques

To select appropriate evaluation techniques, performance metrics and workloads for a system

- You must choose which metrics to use for the evaluation
- You must choose which workloads would be representative

What metrics would you choose to compare:

- Two disk drives?
- Two adaptive video streaming algorithms?
- Two IaaS Clouds?

Key skills needed – measurements

Conduct performance measurements correctly

- You must choose how to apply workloads to the system
- You must choose how to measure (monitor) the system

Which type of monitor (or "probe", hardware or software) would be suitable for measuring each of the following:

- Number of instructions executed by a processor?
- Context switch overhead on a multi-user system?
- Response time of packets on a network?

Key skills needed – proper statistical techniques

Use proper statistical techniques to compare several alternatives

- Whenever there are non-deterministic elements in a system, there will be variations in the observed results
- You need to choose from the plethora of available statistical methods in order to correctly filter and interpret the results

Which link is better?

File Size	Packets lost on Link A	Packets lost on Link B
1000	5	10
1200	7	3
1300	3	0
50	0	1

Key skills needed – do not measure for ever

Design measurement and simulation experiments to provide the most information with the least effort

- You must choose the numbers of parameters to investigate
- You must make sure you can draw statistically viable conclusions

The performance of a system depends on the following factors:

- Garbage Collection Technique used: G1, G2, or none
- Type of workload: editing, computing, or machine learning
- Type of CPU: C1, C2, or C3

How many experiments are needed?

How do you estimate the performance impact of each factor?

Performance in distributed systems

Performance is an art

Performance evaluation is an art

Like a work of art, a successful evaluation cannot be produced mechanically

Every evaluation requires an intimate knowledge of the system and a careful selection of methodology, workloads and tools.

Example of the need for knowledge: know your tradeoffs

- "Bufferbloat" is a term used when greedy, loss-based TCP flows probing for bandwidth fill up a large FIFO queue leading to added delay for all flows traversing this bottleneck.
- To mitigate this, aggressively dropping timer-based AQMs or shorter queues are recommended.
- What do you sacrifice by reducing the size of the queue?

Performance evaluation is an art

A major part of the analyst's "art" is:

- defining the real problem from an initial intuition, and
- converting it to a form in which established tools and techniques can be used, and
- where time and other constraints can be met

Two analysts may choose to interpret the same measurements in two different ways, thus reaching different conclusions

Performance evaluation is an art

The throughputs of two systems A and B were measured in transactions per second. The results were as follows:

System	Workload 1	Workload 2
A	20	10
B	10	20

This is called a ratio game. It is not appropriate for objective analysis, but useful for propaganda.

Performance evaluation is an art

The throughputs of two systems A and B were measured in transactions per second.

The results were as follows:

System	Workload 1	Workload 2
A	20	10
B	10	20

System	Workload 1	Workload 2	Average
A	20	10	15
B	10	20	15

Comparing the average throughput

System	Workload 1	Workload 2	Average
A	2	0.5	1.25
B	1	1	1

Throughput with respect to system B

System	Workload 1	Workload 2	Average
A	1	1	1
B	0.5	2	1.25

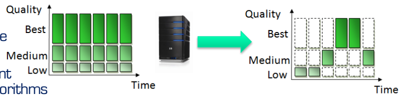
Throughput with respect to system A

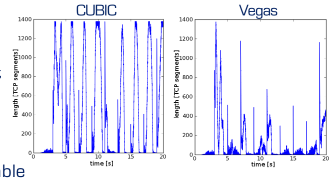
Performance in distributed systems

Real-world examples

Emulation study

Investigation of router queue length development in DASH streaming for different TCP Congestion Control algorithms

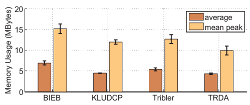




- Simple 2D graph showing an independent parameter X (time) and a dependent Y (queue length)
- does illustrate the unstable queue length for CUBIC, but no actual distribution
- not a quantifiable result, but anecdotal

Simulation study

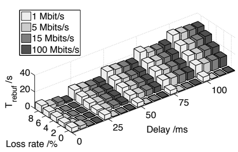
Investigation of memory requirements for several DASH streaming algorithms



- Block diagram does is suitable when X-axis values have no metric relation (no measure of any distance between them)
- block diagram is also better if X-values have an order but no metric relation!
- 2D graph merges 2 questions into 1 graph: average memory use and average peak memory use (average of peaks of several simulation runs) – this does not scale to many questions
- standard deviation is added for each of the averages

Emulation example

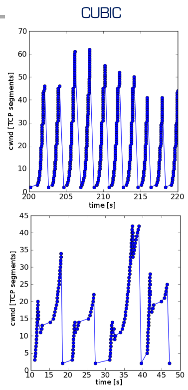
Some HTTP adaptive video streaming strategies can fails when packet loss is high and network delay is high as well. How long are the cumulative waiting times?



- 3D block diagram
- 3 independent variables shown
- 4D information, 3 independent variables (loss rate, delay, network capacity), 1 dependent variable (rebuffering time)
- visually attractive
- tolerances (confidence intervals etc.) cannot be expressed
- absolute height cannot be ascertained by reader for all conditions
- does not scale to many network capacities

Emulation study

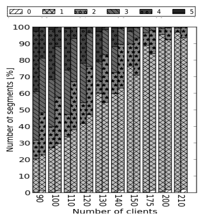
Investigation of sender's congestion window size in the same study.
Video segments have a duration of 2 seconds (top) and 10 seconds (bottom), the algorithm attempts to choose a quality that can be downloaded in 1 second.



- Simple 2D graph showing an independent parameter X (time) and a dependent Y (congestion window size)
- serves to illustrate that CUBIC is incapable of maintaining its congestion window between 2-second DASH segments, but enters TCP slow start
- not a quantifiable result, but anecdotal

Emulation study

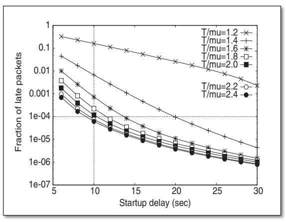
Investigation of the distribution of video quality in the same study.
Segments 2-sec. (left in each column) and 10-sec. (right). Patterns indicate qualities (0 stall, 5 best).
Shows the shares of qualities for entire film.



- Graph with 3 dimensions (X and segment duration independent, Y dependent)
- quite problematic
  - hard to distinguish qualities, patterns are not easily enough recognized
  - quality 1 is dominant, no visual comparison of the others
  - change of order between left and right remains hidden

Analytical performance study

Analytical performance study to discover a relation between streaming (video) over TCP and the likelihood of stalling

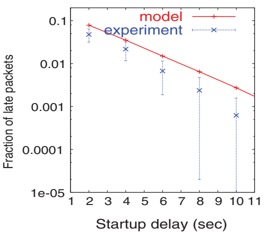


Analytical graph provides deterministic, repeatable results

- symbols distinguish conditions
- Y-axis is logarithmic to expose differences at when very few packets are late
- note that each point is a computation with different parameters

Combined study

Performance study to discover a relation between streaming (video) over TCP and the likelihood of stalling – model validated by ns-2 simulation ("experiment")

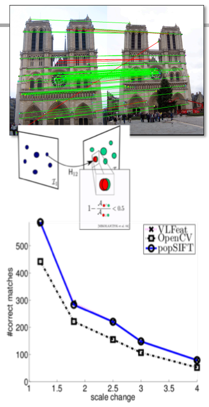


Simulation

- symbols distinguish model and simulation
- Y-axis is logarithmic
- simulation is not deterministic, and error bars show the 95% confidence interval
- for the simulation, the points with error bars are derived from the result of 1000 simulation runs

Emulation example

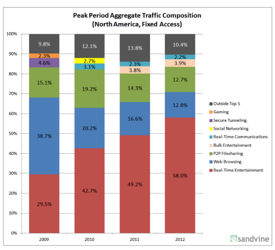
Comparing the performance of the 3 implementations of the algorithm "Scalable Invariation Feature Transform" (SIFT)



- Very simple 2D plot, relating only to a set of very specific image pairs
- 100% deterministic repeatable, no point in expressing errors
- definition ahead of time: boolean condition that defines "match" (adopted from an independent study that developed a good comparison method)

Measurement example

Development of traffic shares over time

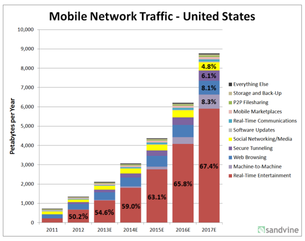


A graph using percentages to express the share of application types on the Internet

- no absolute values, only percentages
- color as well as order allows easy recognition of types, as well as appearance of new types

Measurement example

Development of absolute mobile traffic over time

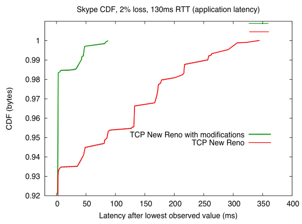


A graph using absolute value to communicate the rapid growth of mobile traffic

- percentages provided as text in graph
- color as well as order allows easy recognition of types, as well as appearance of new types
- note "E" for estimates

Measurement example

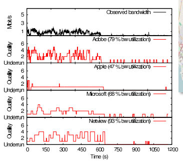
Hypothesis: "Thin-stream" modifications to Linux's implementation of TCP New Reno reduces latency.



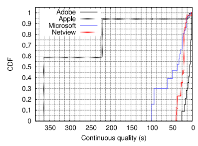
- Cumulative Distribution Function (CDF) provides the percentage of measurement points up to a given X value
- useful if number of samples not identical
- useful if number of samples is quite large

Emulation example

Comparing the bandwidth efficiency and stability of several HTTP Adaptive Streaming methods



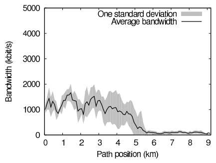
- Upper graph shows quality development over time
- by itself, it has only anecdotal value



- Lower graph shows CDF of quality changes
- Apple HLS is most stable (a desirable property)
- but upper graph exposes that the price for this is nearly very low quality

Emulation example

Documenting the repeatability of bandwidth measurement on a typical commuter path

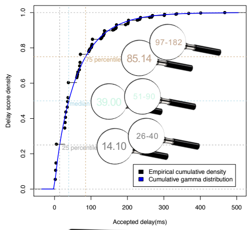


- Map shows the subway route from Stovner to Oslo S

- graph shows the measured bandwidth by distance from Stovner
- figure is does not represent any specific measurement run, the measurements have been collected, and the graph shows both the average bandwidth and 1 standard deviation
- not only anecdotal but valid for predictions

User study example

Hypothesis: users can detect that they are experiencing hand-eye latency below 100ms

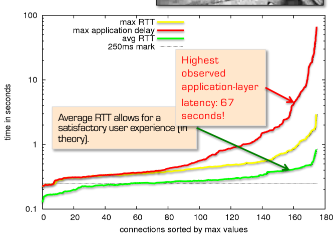


- Cumulative Distribution Function (CDF) provides the percentage of measurement points up to a given X value (using dots in this case)
- matched with a function (here cumulative gamma distribution)
  - better describe the distribution and validate generality
  - create simulations

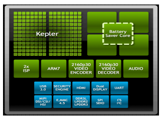
Measurement example

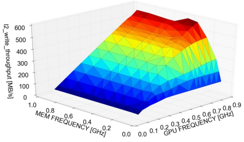
Application-layer behaviour of a popular MMORPG estimated from a single server-sided measurement probe

- simple 2D presentation, both dimensions observed
- data sorting can provide more information than *histograms* or *cumulative distribution functions*



Measurement example



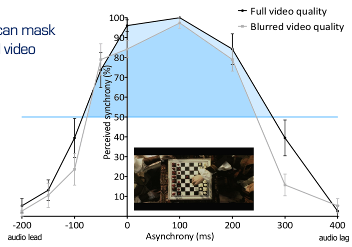


NVidia Tegra K1
impact of frequency on throughput

- note: 4 dimensions in the presentation
- additional dimension can be used to add information or to add expressiveness to one or more of the dimensions

User study example

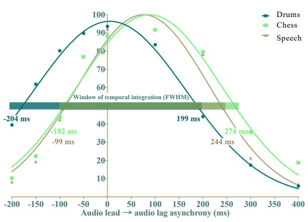
Hypothesis: poor video quality can mask asynchrony between audio and video streams
(note: proven wrong)



- note: 3 dimensions in the presentation
- sample points are plotted with error bars
- highlight color adds meta-information, here highlighting 50 percent of the study population
- also typical to fit a typical behavior function from samples using linear regression (shown on next slide)

User study example

Pre-study: perception of asynchrony for different content



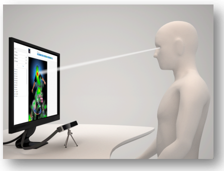
- note: 3 dimensions in the presentation
- curves generated from sample using linear regression
- horizontal bar adds meta-information, here highlighting 50 percent of the study population
- color is used to distinguish items (4th dimension) and to associate measurements with fitted curves

User study example

The influence of semantic relations between visual elements on human attention.

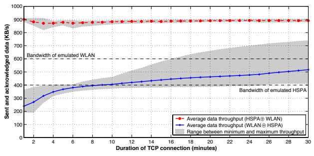
Heatmaps allow a presentation of 2D data accumulated over time.

- 2D input (axes), 1D output (color).
- Can be overlaid over base data.



Measurement example

Linux TCP's ability to recover from out-of-order delivery of packets

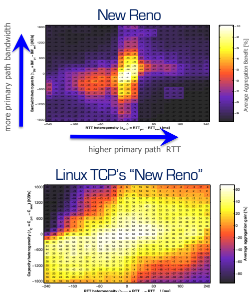


- 2D graph studying values for the average bandwidth of very long-lived TCP flows whose packets are alternately sent over 2 very different paths
- details of short-term TCP behaviour are completely hidden
- smoothness achieved by averaging
- shaded areas illustrate uncertainty (range from min to max average throughput)

Measurement example

TCP's ability to benefit from using the capacity of 2 paths that are heterogeneous in terms of available bandwidth and RTT

- 3D information, 2 independent variables (X and Y), 2 dependent variable (aggregation benefit and detected reordering)
- good memory effect
- highly aggregated data
- concept of certainty (e.g. confidence intervals) gets lost



Performance in distributed systems

Common mistakes

Common mistakes and how to avoid them

No goals:

- Knowing the goal of the performance analysis will guide your choices of techniques, tools, metrics, workloads.

- Without goals, modeling must be identical to reality
  - imagine weather models or models of the universe without specific goals
- There are no general-purpose models. Models are always simplifications of the real world, actively dropping detail.
  - without goals, there is no simplification
  - without simplification, modeling is identical to building
- Defining goals is difficult, especially in combination with bias

Common mistakes and how to avoid them

No goals:

- Knowing the goal of the performance analysis will guide your choices of techniques, tools, metrics, workloads.

Biased goals:

- Avoid implicitly or explicitly bias the goals. The objective should be to perform a fair evaluation of the systems that are compared.
- See also: https://en.wikipedia.org/wiki/List_of_cognitive_biases

Be aware of the risk of **bias** that is present in these interests!

bias [Webster's dictionary]
1.c) deviation of the expected value of a statistical estimate from the quantity it estimates
1.d) systematic error introduced into sampling or testing by selecting or encouraging one outcome or answer over others

Common mistakes and how to avoid them

Unsystematic approach:

- Be systematic when selecting system parameters, metrics, workloads etc. Random choices will provide inaccurate answers.

- Identify a complete set of
  - goals
  - system parameters
  - factors
  - metrics
  - workloads
- then define a goal and select the appropriate subset

Common mistakes and how to avoid them

Unsystematic approach:

- Be systematic when selecting system parameters, metrics, workloads etc. Random choices will provide inaccurate answers.

Analysis without understanding the Problem:

- Make sure that you have done your best to try to understand what is really the problem. This will improve the chances of success by a large factor.

- Identify the real problem
  - this may require a lot of prior work
  - the answer of the preparation may diverge from expectations or common assumptions
- This is not always easy
  - e.g.: for decades, TCP has been improved for throughput
    it was very hard to sell latency as a valid problem

Common mistakes and how to avoid them

Incorrect performance metrics:

- The metrics depends on a range of factors. Avoid choosing easily
- accessible / easy to compute metrics, if they are not the right metrics.

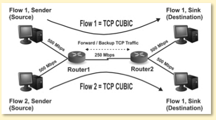
- e.g.: "everybody knows" about TCP that acknowledgement for the same packet that arrives at the sender 3 times triggers a congestion event and a retransmission
  - except that it doesn't happen in Linux TCP
- e.g.: Network performance measurement was all about throughput and fairness. When latency was introduced the whole picture changed.

Common mistakes and how to avoid them

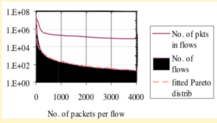
Unrepresentative workload:

- The workload should be representative of the system in the field.

- the usual simulation for TCP research looks like this, and "greedy" streams are sent through the bottleneck



- ignoring that most flows in real networks are extremely short



Common mistakes and how to avoid them

Wrong evaluation technique:

- Choosing between modelling, simulation or measurement can make all the difference.
- In this course, we have made this selection simple for you

The combination of two and more of these techniques add to sellability!

Modelling gives you the best understanding of what's going on, iff the results are confirmed by one of the the other two

Criterion	Analytical Modeling	Simulation	Measurement
Stage	any	any	Post-prototype
Time required	small	medium	varies
Tools	analysts	programs	instrumentation
Accuracy	low	moderate	varies
Trade-off evaluation	easy	moderate	difficult
Cost	low	medium	high
Saleability	low	medium	high
Insight	high	medium	low

Common mistakes and how to avoid them

Overlooking important parameters

- Do your best to make a complete list of the system and workload characteristics that may affect the performance
- After gaining an overview of the parameter list, you may prioritise between parameters to include in the study to allow completion of the experiment set within your lifetime.

Common mistakes and how to avoid them

Ignoring significant factors

- Parameters that are varied in the study are called ***factors***
- Not all parameters have an equal effect on the performance
- Consider which parameters are of significance when choosing which factors to use

- note that a factor is an input parameter
  - there are factors that can usually be ignored because they are mostly constant
  - but these may have huge influence when they do vary – make a pre-study before removing them
- a new challenge has arrived with the prevalence of machine learning:
  - failing to attempt to isolate and understand parameters
  - assuming that you created a machine learning network that will discover them by itself

Common mistakes and how to avoid them

Inappropriate experimental design

- Be careful when selecting the numbers of experiments to run and when
- selecting parameter values.
- If there are dependencies between the effects of some parameters and other parameters in the experiment, a *full factorial experiment* or *fractional factorial experiment* may improve the results.

- design should be simple but not too simply
- e.g.: mathematical analysis must always be extremely simple
  - but it looses detail
  - can you afford that?

Common mistakes and how to avoid them

Inappropriate level of detail

- When modelling, the formulation should not be too broad, nor too narrow.

- very different: high-level model
- compare details: detailed model

No analysis

- After collecting a huge pile of data, make sure to apply analytical skills to ease the new knowledge out of the raw data

- measurement campaigns can frequently end in this problem
  - you have to conduct them when the opportunity arises
  - you have to collect whatever you can think of
  - you cannot go back and collect more
- filtering the right parameters is a major challenges, tools PCA help only for independent Euclidian variables – so you may be in trouble

Common mistakes and how to avoid them

Erroneous analysis

- Be careful to avoid common mistakes when analysing the data

- a very typical danger in analytical approaches is to forget the assumption that parameters are normally distributed before applying a statistical operation

No sensitivity analysis

- The results may be sensitive to workload and system parameters.
- Analyse the outcomes considering such sensitivity.

- a result may not be desirable even if it is best in an example, but it is highly unstable, meaning that performance results change strongly (to the negative) when one or more parameters change slightly
- a result may not be trustworthy if a jhigh-impact parameter is assumed to be constant, but it isn't in reality

Common mistakes and how to avoid them

Ignoring errors in input

- Often the parameters of interest cannot be measured and is estimated using another parameter.
- In such cases, the analyst needs to adjust confidence of the output obtained from such data.

- a recent example
  - assumptions about the presence of an advanced queue management (AQMs) strategy at the network level in a wireless system
  - to design algorithms in wireless systems, it is important to know whether AQM are deployed
  - but time slicing at the link layer level can look like AQM and prevent its correct detetion

Common mistakes and how to avoid them

Improper treatment of outliers

- Deciding which outliers can be ignored and which should be included requires intimate knowledge of the system

- outliers can have a massive impact on averages and consequently on confidence intervals
- but can they be ignored?
- what is an outlier?
- A hugely important question in crowdsourcing! ➔ filtering based on assumptions

Assuming no change in the future

- It is often assumed that the future will be the same as the past
- Consider whether changes in workloads and system behaviour might need to be taken into consideration

Common mistakes and how to avoid them

Ignoring variability

- Determining variability is often difficult, if not impossible, so the mean is often used for analysis.
- You need to apply the system knowledge when determining to which degree variability may end up as misleading results.

- this is a typical sight in paper today
- time-based plots and average as the only applied statistical method
- it makes it impossible to discover and expose instabilities from factors
- it makes it really hard to understand variability in results

Common mistakes and how to avoid them

Too complex analysis

- Occam's razor for analysis. The simpler one and the one easier to explain is usually preferable.
- Convey the results in as simple a way as possible.

- simple questions may have a simple answer
- I saw in a paper
  - use of a Poisson-distribution for packet interarrival time, its average interarrival time E given
  - then, use of a machine learning model to detect average interarrival time
  - Why?

Common mistakes and how to avoid them

Improper presentation of results

- Choose wording/tables/visualisations that communicate the properties of the analysis fairly

- Even if bias was avoided in the study, it can still be in the presentation

Ignoring social aspects

- You will need not only to perform a precise analysis. You will also need to sell the analysis to decision makers.
- Especially when you want to change the opinion of the decision maker(s)

Common mistakes and how to avoid them

Omitting Assumptions and Limitations

- Expose your assumptions and limitations to the audience of your analysis.
- This will help avoid that the analysis will later be used for inappropriate scenarios (for instance as referenced work)

- a study is always limited to some extent
- be aware of your limitations and share them with your audience
- even better, make your study repeatable by sharing code and data

Checklist for avoiding common mistakes

#	What to check
1.	Is the system correctly defined and the goals clearly stated?
2.	Are the goals stated in an unbiased manner?
3.	Have all the steps of the analysis followed systematically?
4.	Is the problem clearly understood before analyzing it?
5.	Are the performance metrics relevant for this problem?
6.	Is the workload correct for this problem?
7.	Is the evaluation technique appropriate?
8.	Is the list of parameters that affect performance complete?
9.	Have all parameters that affect performance been chosen as factors to be varied?
10.	Is the experimental design efficient in terms of time and results?
11.	Is the level of detail proper?
12.	Is the measured data presented with analysis and interpretation?
13.	Is the analysis statistically correct?

Checklist for avoiding common mistakes

#	What to check
14.	Has the sensitivity analysis been done?
15.	Would errors in the input cause an insignificant change in the results?
16.	Have the outliers in the input or output been treated properly?
17.	Have the future changes in the system and workload been modeled?
18.	Has the variance of input been taken into account?
19.	Has the variance of the results been analyzed?
20.	Is the analysis easy to explain?
21.	Is the presentation style suitable for its audience?
22.	Have the results been presented graphically as much as possible?
23.	Are the assumptions and limitations of the analysis clearly documented?

Performance in distributed systems

Systematic approach

A systematic approach to performance evaluation

1) State the goals and define the system
   - What is the goals of the study?
   - What is the boundaries of the system you want to measure?
2) List services and outcomes
   - Each system provides a set of services
   - When a user requests any of these services there are a number of possible outcomes
   - Some of the outcomes are desirable, some are not
   - This list will be useful when selecting the right metrics and workloads

A systematic approach to performance evaluation

3) Select metrics
   - Select the criteria used for comparing the performance
4) List parameters
   - Make a list of all the parameters that affect the performance
   - It might be useful to divide the list into system parameters and workload parameters
   - This list might grow as you learn from the first iterations of experiments and analysis.

A systematic approach to performance evaluation

5) Select factors to study
- The list of parameters can be divided into two parts: those that will be varied in the study and those that will not.
- The parameters that are varied are called ***factors*** and their values are called ***levels***
- An important part of the work is to choose the factors so that the study will be possible to complete with the given resources

6) Select evaluation technique
- Models, simulation or measurement

A systematic approach to performance evaluation

7) Select workload
- The workload consists of a series of service requests to the system
- You need to measure and understand the characteristics of a system in order to build a relevant workload.
- You can build on other people's workload analysis, but beware the future==past trap.

8) Design experiments
- Once you have the list of factors and levels, you need to decide on a sequence of experiments that offer maximum information with minimal effort.
- 2 phases can be useful: 1) Large number of factors, small number of levels to determine the relative effect of factors; 2) fewer factors / more levels for factors with significant impact

A systematic approach to performance evaluation

9) Analyse and interpret data
- Choose appropriate statistical techniques
- Try to make a fair evaluation between the systems

10) Present results
- Visualise the data in a way that fairly and clearly shows the differences in performance
- A good metric for visualisation/presentation is how much effort it takes to read/understand the presentation. Easy = good

A systematic approach to performance evaluation

Steps for a Performance Evaluation Study	
1.	State the goals of the study and define the system boundaries
2.	List system services and possible outcomes
3.	Select performance metrics
4.	List system and workload parameters
5.	Select factors and their values
6.	Select evaluation techniques
7.	Select the workload
8.	Design the experiments
9.	Analyse and interpret the data
10.	Present the results. Start over if necessary.

Performance in distributed systems

Projects

Performance measurement projects

In this course we will give you performance analysis tasks where you will wrestle the tradeoffs, the parameters, the metrics, the methodologies, the analysis and the presentation.

We will

- introduce many of the main concepts of performance analysis
- introduce the topics that form the basis of the graded assignments
- provide example reports of good quality for you to study
- be available on email for guidance and pointers

Performance measurement projects

You must:

- Go to the literature (and the web) for details and resources to help you on the way
- Apply your own skills and judgement in the selection of metrics and methodology
- Justify your choices and try to avoid making random or biased selections
- You will face a lot of tradeoffs and difficult choices. Ask for advice. Communicate!
- This is what researchers and industry professionals are required to do in their practice