Parametrization of Models: Regression, Benchmarking

Budapest University of Technology and Economics
Fault Tolerant Systems Research Group

Performance Analysis Approaches

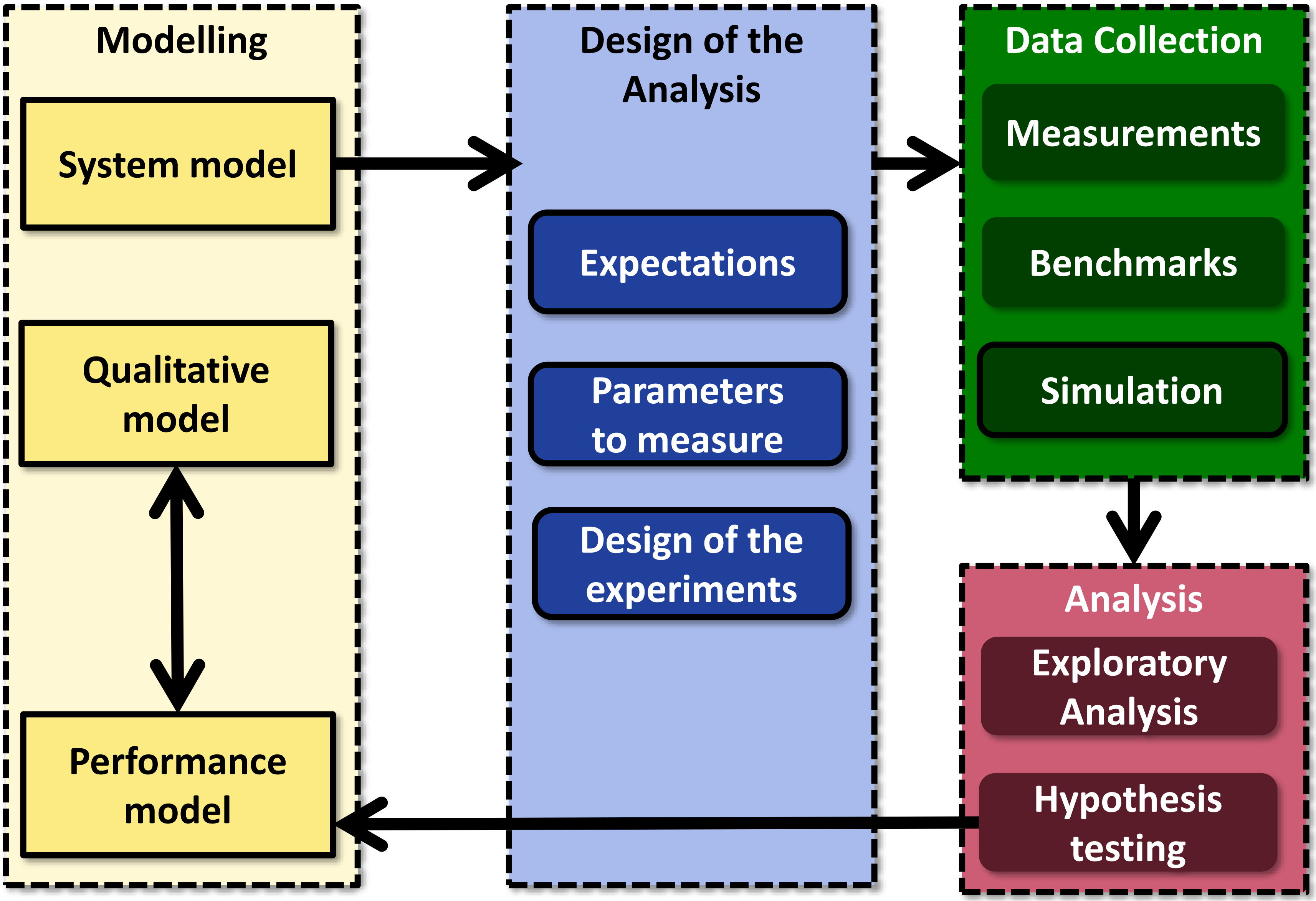
Load Test

- „Synthetic,” simple load
- Exploring maximum throughput
- Comparison of different versions of the same system
- Examining the overloaded state

Benchmarking

- Based on real use cases
- Complex environmental parameters and load
- Objective comparison of different systems
- Examining the stable state

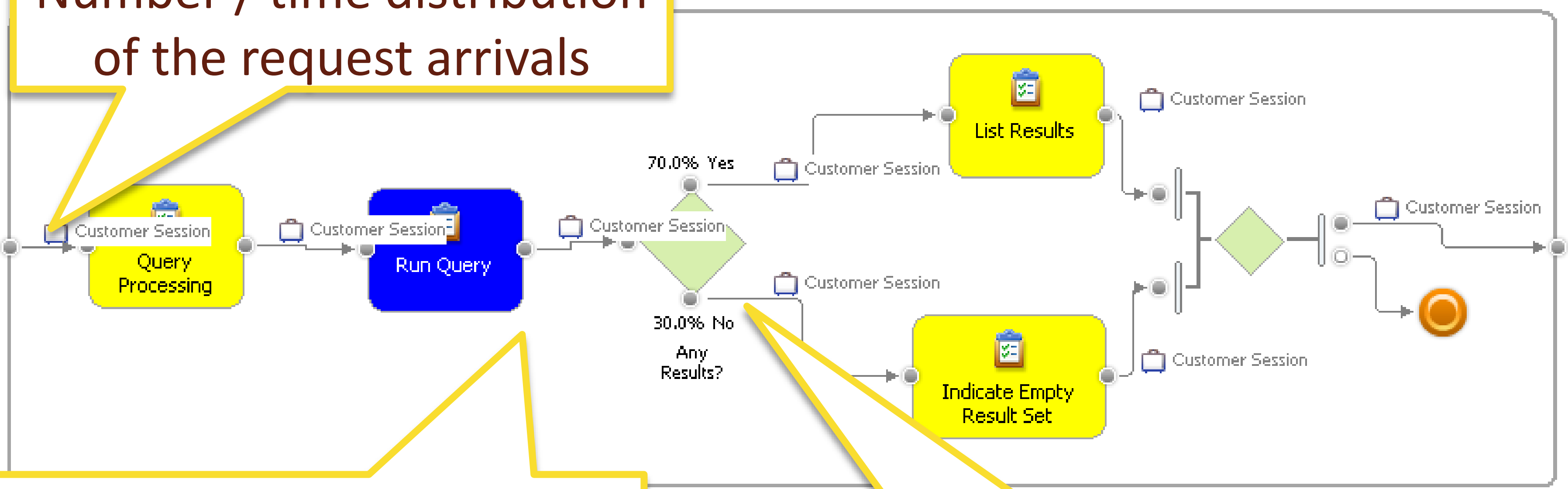
From System Model to Performance Model



The Big Question

- Do we estimate the quantitative paramters well?

Number / time distribution of the request arrivals



Execution time of a given activity on a given resource

Approximated decision probabilities/frequencies (estimated values)

Creditability of Data

- Sensitivity analysis
  - How sensitive the **output** parameters of the model are on the changes of the **input** parameters
    - (number/capacity of resources, decisions of the users) → (response time, throughput of the process)
  - „parameter sweep”: analysing the consequences of the changes of a parameter within a given range
    - → How good our estimation on the parameter has to be?
- Rule of thumb: creditability of data
  - Uncertainty of the measurement (variance) falls with the square of the number of measurements
    - for sufficient amount of data (see Probability Theory)

MATHEMATICAL ESTIMATION: REGRESSION METHODS

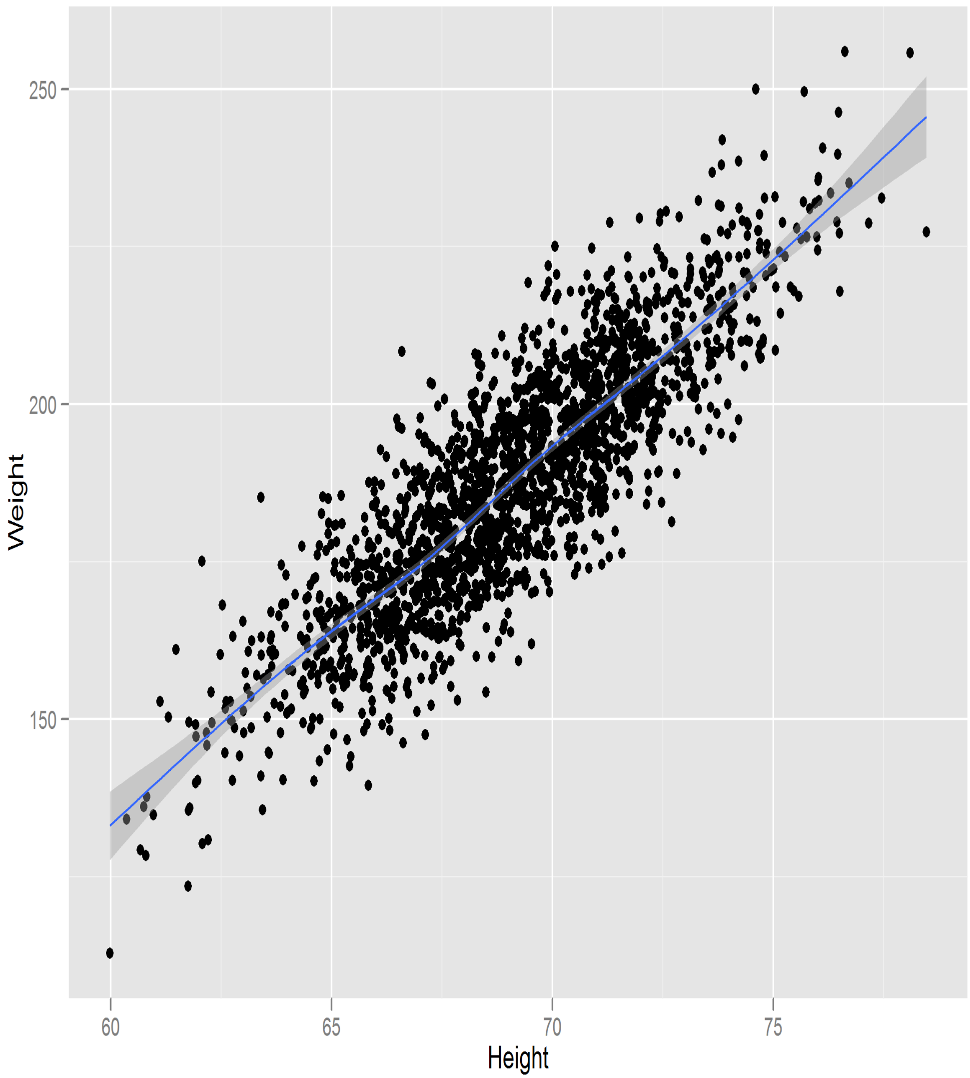
The Problem

- Many variables are given over a longer period of time
- (Some of) The values need to be estimated, because
  - difficult to measure / cannot be measured
- Estimation/Forecast is required
  - Not yet happened, we estimate it as a function of time
  - The corresponding input value (e.g. number of users) cannot be generated
  - The consequences are not yet visible (e.g. response time increases just while waiting for processing the requests)
- How far we can trust the results / conclusions?

Regression

Function *f*,

- input:
  attribute values
- output:
  best approximation of the observations
- „rule of thumb”
- Example:
  the common distribution of height/weight fits on a line



Regression methods

- Principle:

$$Y_t = f(\bullet) + \varepsilon_t$$

Probabilistic variable

Approximation

Error

$$Y = f(X_1, X_2, \ldots, X_n)$$

Estimated result

Observable variables

- Mean error

$$ME = \frac{\sum_{t=1}^{n} (Y_t - F_t)}{n}$$

Estimated value

Measured value

Linear Regression

- Fitting a simple linear function on the data
  - No big changes are expected in the system behaviour

$$Y = a + bX$$

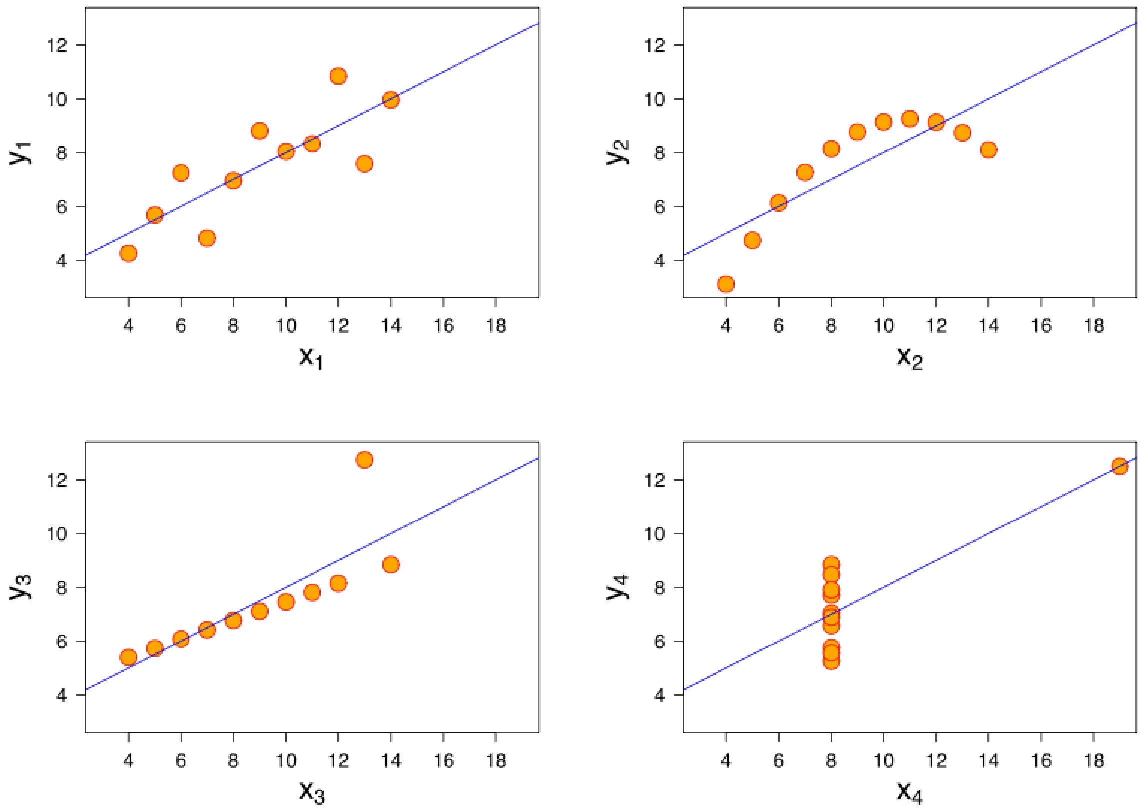
- Method of the smallest squares
  - Looking for parameters *a,b* (here: a – offset, b – rise), for which

$$SSE = \sum_{t=1}^{n} \varepsilon_t^{\,2} = \sum_{t=1}^{n} \left(Y_t - F_t\right)^2$$ minimal (Sum of Squared Errors)

- Goal: $\sum_{t=1}^{n} (Y_t - F_t)^2 = \sum_{t=1}^{n} \left[Y_t - (a + bX_t)\right]^2$

Linear Regression

- Best fitting line
- **But:** Anscombe's quartet
  - Fundamentally different data
  - Same regression line
- Dangerous conclusions for non-linear data



Linear Regression (cont.)

- Correlation coefficient (the square of ~)
  - relation between the expected and actual values of a variable
  - has a value between 0 and 1
  - 0: no relation
  - 1: function like relation
  - R itself between -1 and 1 (direction of the relation)

$$R^2 = \frac{\sum_{t=1}^{n}\left(F_t - \bar{Y}\right)^2}{\sum_{t=1}^{n}\left(Y_t - \bar{Y}\right)^2}$$

- Example: E-mail service, peak load measured for 8 weeks

week	1	2	3	4	5	6	7	8
Max. load (email/minute)	420	410	437	467	448	460	507	514

How can the change of the load approximated?

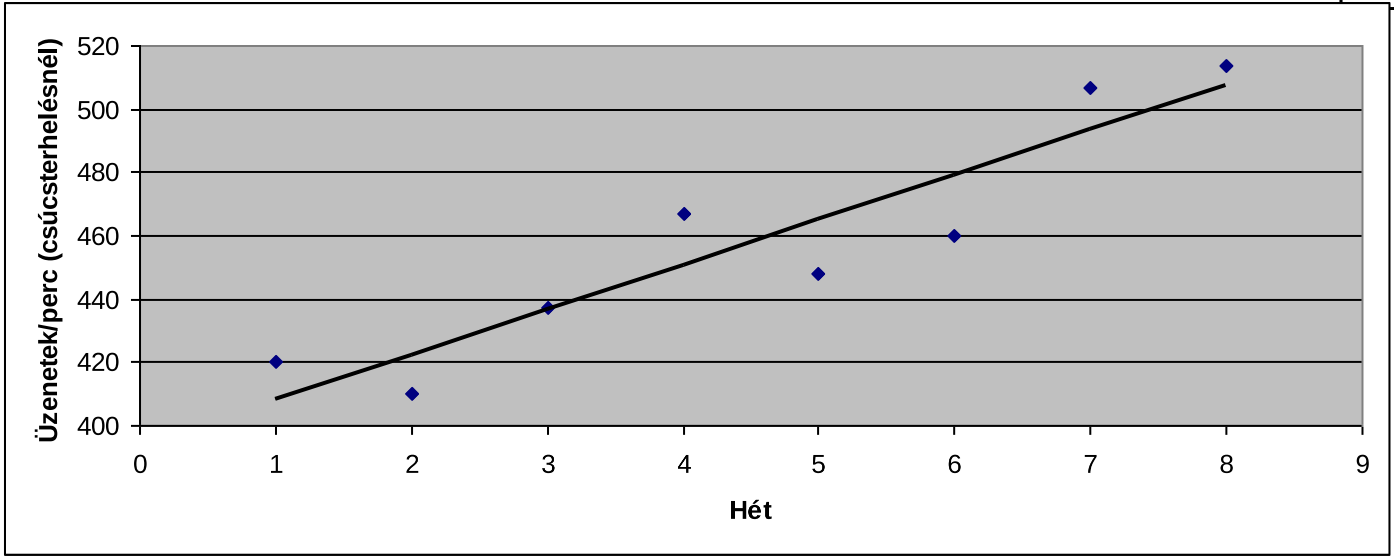
How high is the correlation? (correlation coefficient)

Linear Regression Example

With method of the smallest squares
Y=393.98+14.20X

Correlation coefficient:
R^2=0.855

Measured	Forecasted
420	408,18
410	422,38
437	436,58
467	450,78
448	464,98
460	479,18
507	493,38
514	507,58
	521,78



Studying the Relation of Two Variables

- Let's assume a linear relation between the number of concurrent users and the number of sent mails
(e.g. based on the logs)

Average number of concurrent users (in 1 hour)	2450	2765	2241	2860	3011	2907	3209
Avg. Load (incoming+outgoing mails/hour)	19257	20488	18152	21450	21077	20639	22142

- Linear regression based on the method of smallest squares:

#mails = f(*#users*)

Y=9480.48 + 3.95X R^2=0.937 → strong relation

Non-linear methods

- Exponential approach $Y_t = a \times b^t$
- Fits well to the rise of web traffic
- Transforming the function:

$$\log Y_t = \log a + t \log b$$

$$\log Yt = Y', \log a = a', \log b = b'$$

$$Y' = a' + b't$$

- Method of the smallest squares can be applied
- E.g. the measured values of the highest load are given

What is expected for the end of the year?

Month	1	2	3	4	5	6	7	8	9	10
Max. requests/sec (Y_t)	1035	1100	1160	1250	1350	1555	1770	1950	2210	2630
ln (Y_t)	6,942	7,003	7,056	7,13	7,207	7,349	7,478	7,575	7,7	7,874

Example: Exponential Load

- Estimator function: $Y_t = a \times e^{bt}$
- Method of the smallest squares on the linear function

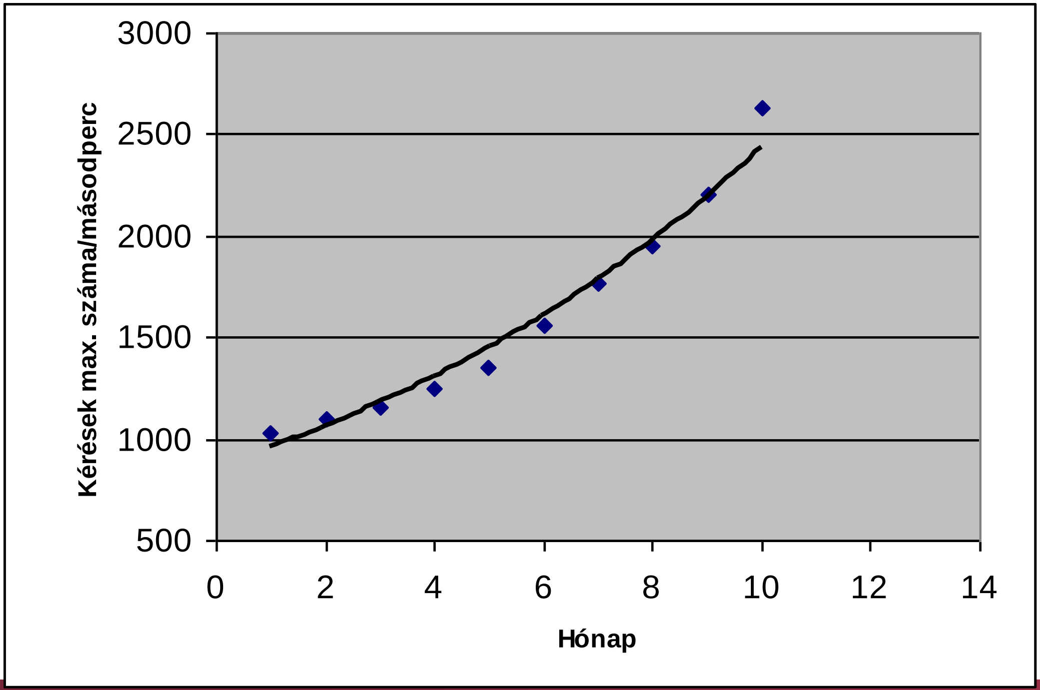
$$Y' = a' + b't, a' = 6.717, b' = 0.110, a = e^{a'}$$

- Result:

$$Y_t = 826.33 \times e^{0.11t}$$

- 12. month:

Y_t=3093.3



Method of the Moving Average

- For short-term forecast only
- Always gives one value at a time only
- The expected value is the average of the last *n* values

$$F_{t+1} = \frac{\sum_{i=t}^{t-n+1} Y_i}{n}$$

where $\boldsymbol{Y_t}$ is the value measured at time t.

$\boldsymbol{F_{t+1}}$	is the expected value
n	is typically between 3 and 10
	(to limit the failure of the estimation)

Exponential Sliding Window

- Always gives one value at a time only, the average of the previous measurements
- The later the measurement, the higher weight
  - Also for the faults
- For short-term forecast only
  - (Why is it called exponential?)

$$F_{t+1} = F_t + \alpha\left(Y_t - F_t\right)$$

Where F_t : the expected value for time t.

Y_t : the value measured at time t.

$Y_t - F_t$: measurement fault at time t.

α : weight $(0 \leq \alpha \leq 1)$

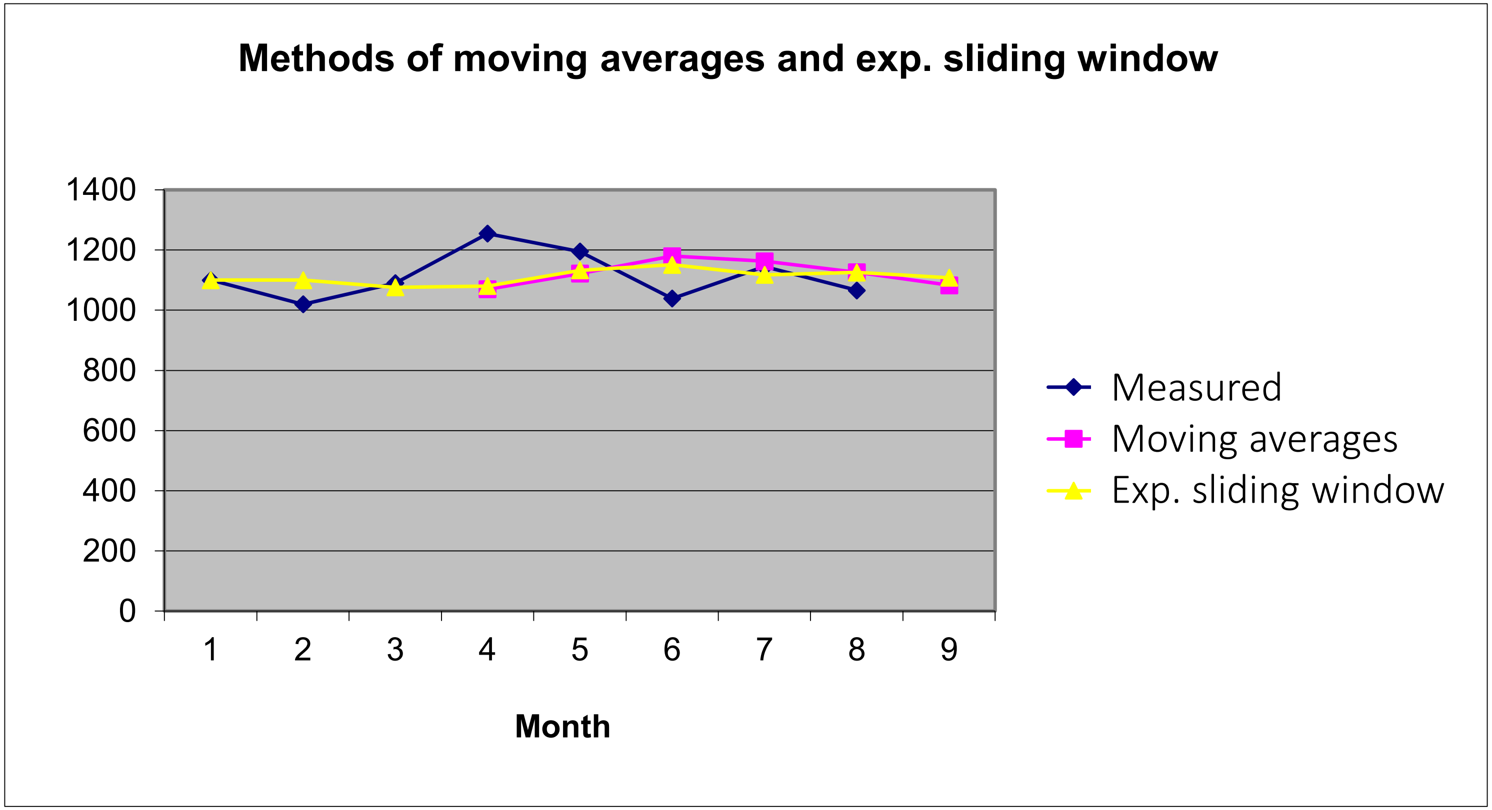
in the practice $0.05 \leq \alpha \leq 0.3$

Comparison of the Two Methods

- The requested bandwidth is given
- Next values are estimated with the two methods

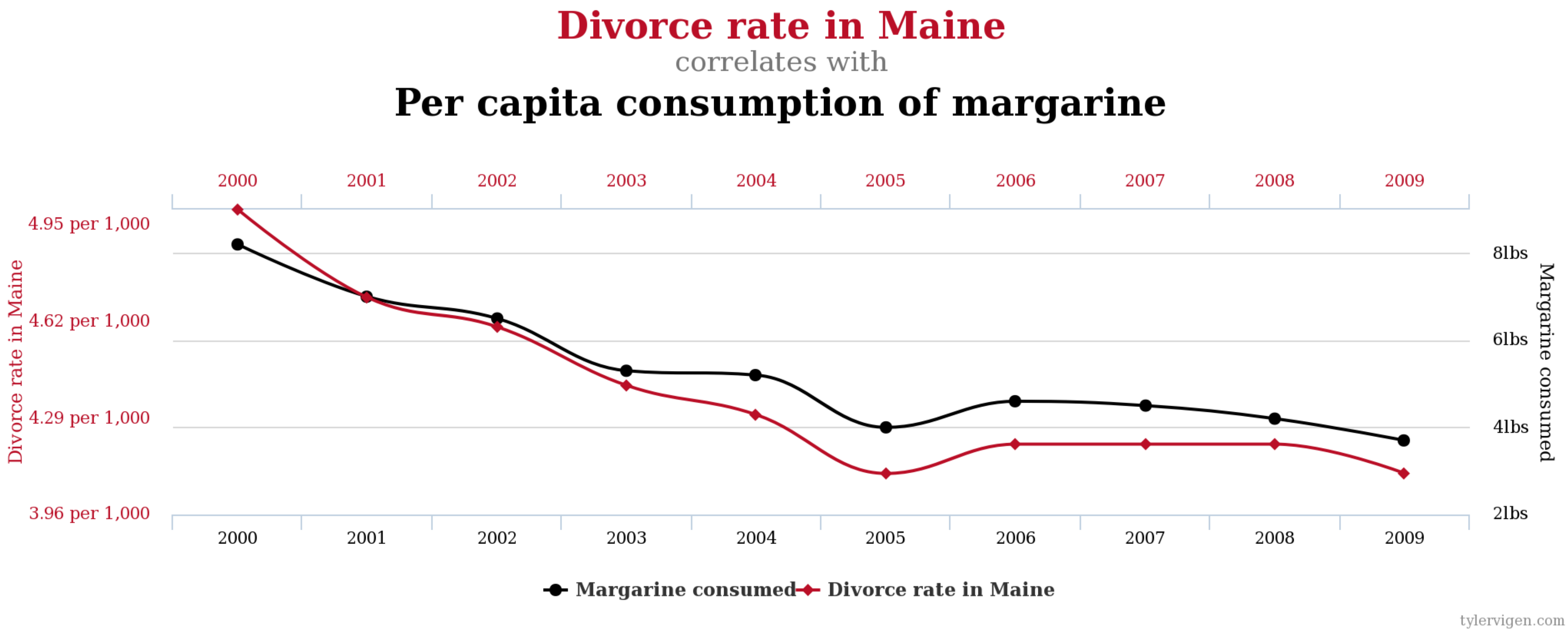
Month	Requested bandwidth	Moving average (n=3)	Exp. sliding window (α = 0.3)
1	1100		1100,00
2	1020		1100,00
3	1090		1076,00
4	1255	1070,0000	1080,20
5	1195	1121,6667	1132,64
6	1039	1180,0000	1151,35
7	1145	1163,0000	1117,64
8	1066	1126,3333	1125,85
9		1083,3333	1107,90

Comparison of the Two Methods



Important note

Causality != Correlation (cause-consequence relation != common occurance)



Example from the IT: many users → high utilization AND long response time

WHY BENCHMARKING?

Why Benchmark?

Benchmarking - Definition

- Wikipedia

*„In **computing**, a benchmark is the **act of running** a computer program, a set of programs, or other operations, in order to **assess the relative performance** of an object, normally by running a number of **standard tests** and trials against it."*

Benchmarking is

- the **execution** of a **program** (of multiple programs or of other operations)
- **with standardised tests** or inputs,
- **to determine the relative performance** of an object.

Benchmarking

- Goals: comparing performance of software/hardware tools
- Decision support
    - Which components should be bought/installed?
    - For what amount of load is the current system sufficient?
    - How powerful are the other vendors?
  - Performance testing
    - Should the performance improved and where? (development phase)
    - Is a specific setting optimal?
    - Does a setting effect the global performance?

Expectations

- Repeatability
  - „Same” results if repeated on the same instance
- Reproducibility
  - Measurement can be reproduced by others
- Relevance
- Complying with standards/agreements
- Generalized use case
  - Result should be intelligible to general user

Benchmark Load Models

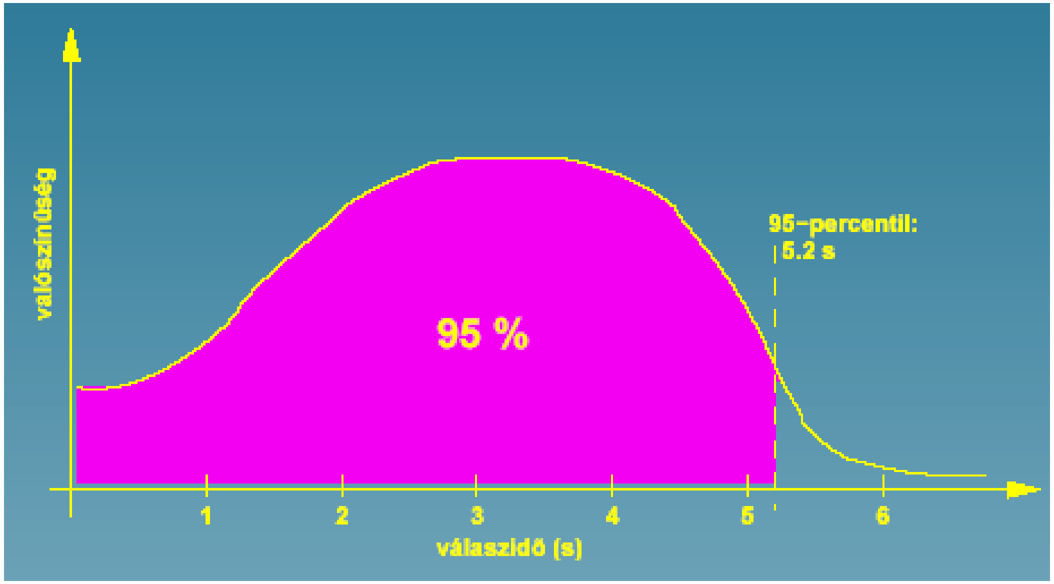
- Scientific/technical systems
  - Processing big amount of data(number crunching)
  - Parallel methods
- Transaction management (OLTP)
  - Client-server environment
  - Multiple quick, parallel transactions
- Batch-type processing
  - Making reports of large amounts of data
- Decision support
  - Few, complex queries
  - Ad hoc operations
  - Lot of data (e.g. OLAP)
- Virtualization

Parameters to be Measured (Metrics)

- Running time
  - Beginning, end?
  - Distribution
  - CPU, I/O, network,…
- Speed of transaction
  - System's reaction time
  - Even nested transactions
- Throughput
  - ***Processed*** data/ running time
  - Depending on load

Metrics (2)

- Response time
  - Depending on load
    - users
    - number of transactions, etc.
- X-Percentil
  - X percent of a set is under this value



Performing Benchmarks

- Ensuring relevance
  - We really measure the application we are supposed to
  - Nature of load generation should approximate to the real load
  - Minimalize confounders

STANDARD BENCHMARKS

SPEC, TPC-C, …

SPEC Benchmarks

- http://www.spec.org/benchmarks.html
  - Standard Performance Evaluation Corp.
- Resource and application level benchmarks
  - CPU
  - Applications
  - Mail servers
  - Web servers, etc.
- Benchmark: a service to order

SPEC CPU2006

- CPU-intensive
- CINT2006
  - Computationally intensive, integer numbers
- CFP2006
  - Floating point numbers

- Results: http://spec.org/cpu2006/results/
  - Test Sponsor (vendor), System Name (product)
  - Processor: enabled cores, enabled chips, cores/chip, threads/core
  - Results: base, peak

CINT2006 and CFP2006 Load Generators

- CINT2006 :

400.perlbench	C	Programming Language
401.bzip2	C	Compression
403.gcc	C	C Compiler
429.mcf	C	Combinatorial Optimization
445.gobmk	C	Artificial Intelligence
456.hmmer	C	Search Gene Sequence
458.sjeng	C	Artificial Intelligence
462.libquantum	C	Physics / Quantum Computing
464.h264ref	C	Video Compression
471.omnetpp	C++	Discrete Event Simulation
473.astar	C++	Path-finding Algorithms
483.xalancbmk	C++	XML Processing

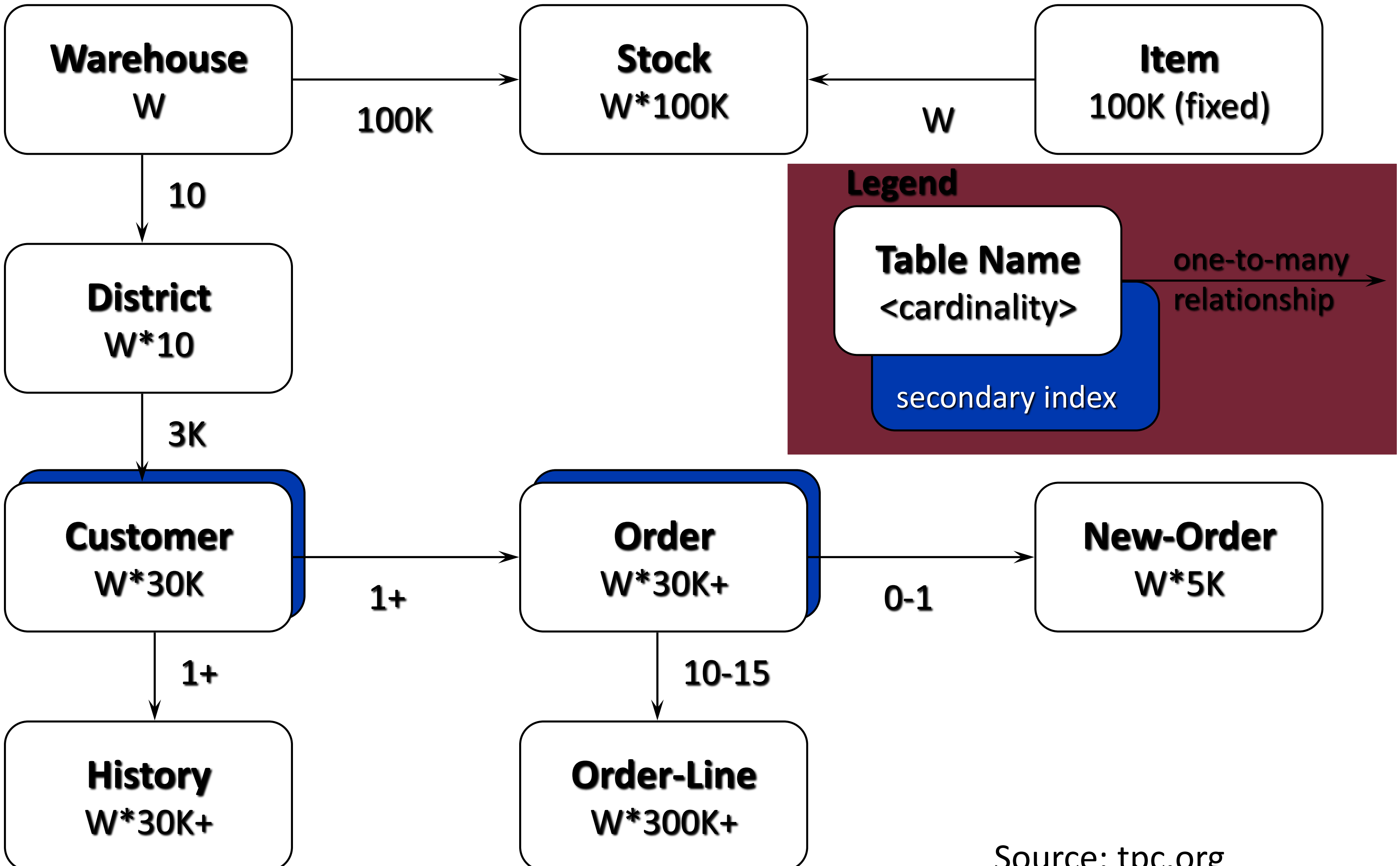
- CFP2006:

410.bwaves	Fortran	Fluid Dynamics
416.gamess	Fortran	Quantum Chemistry
433.milc	C	Quantum Chromodynamics
434.zeusmp	Fortran	Fluid Dynamics
435.gromacs	C, Fortran	Molecular Dynamics
436.cactusADM	C, Fortran	General Relativity
437.leslie3d	Fortran	Fluid Dynamics
444.namd	C++	Molecular Dynamics
447.deall	C++	Finite Element Anal.
450.soplex	C++	Linear Programming
453.povray	C++	Image Ray-tracing
454.calculix	C, Fortran	Structural Mechanics
459.GemsFDTD	Fortran	Electromagnetics
465.tonto	Fortran	Quantum Chemistry
470.lbm	C	Fluid Dynamics
481.wrf	C, Fortran	Weather
482.sphinx3	C	Speech Recognition

TPC Benchmarks

- Benchmarking database management systems
  - RDBMS+OS+HW
- Benchmark environment
  - Sample database: clients and orders
  - 5 transaction types (queries/modifications) mixed
  - Upper limit of running time
  - Real conditions: ACID transactions, users' time to think (atomicity, consistency, isolation, and durability)
- Measured data
  - Throughput (tpmC) (*transaction per minute*)
  - „Efficiency" ($/tpmC)

TPC-C Schema



Source: tpc.org

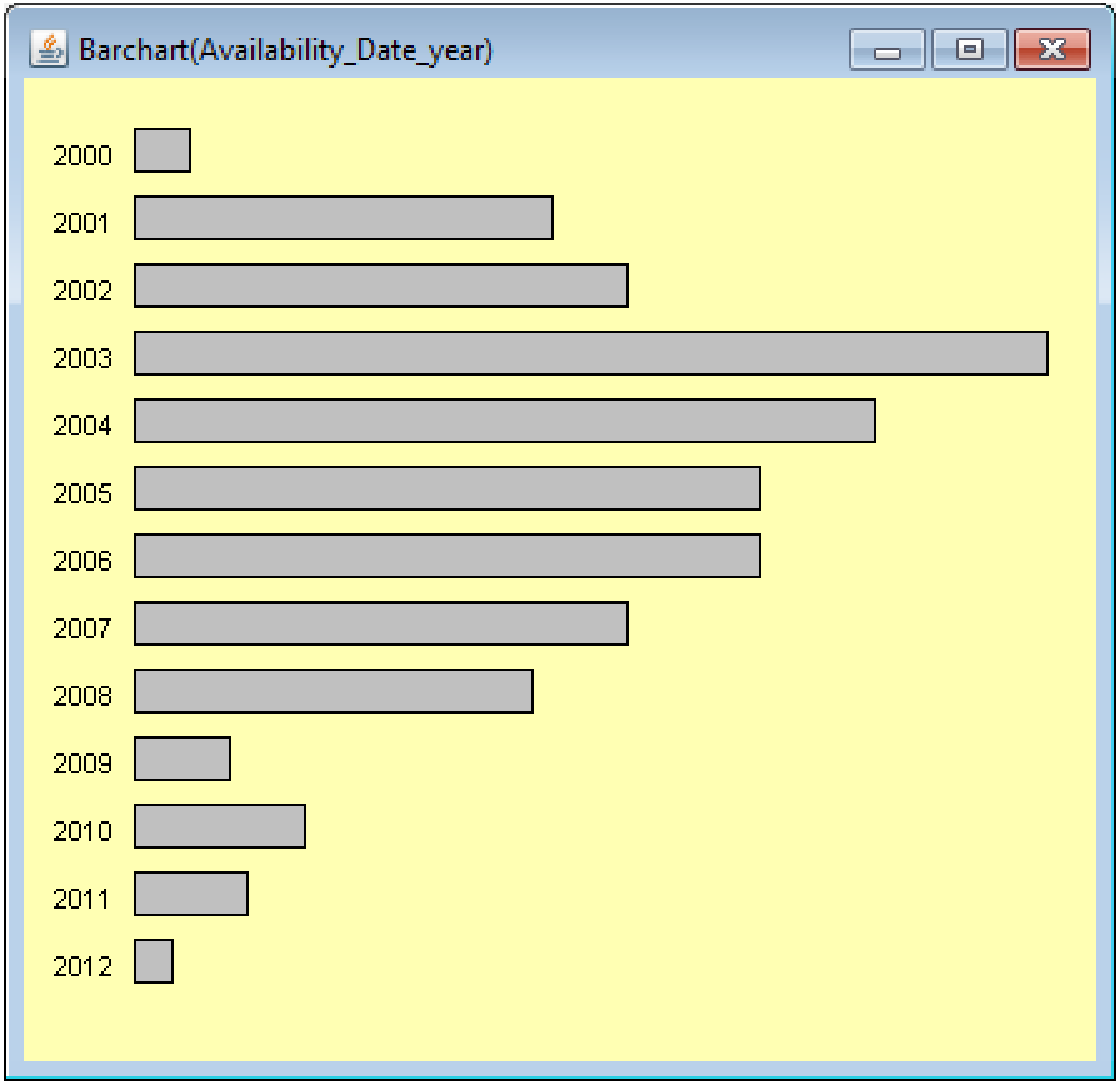
Before Analysing: Cleaning the Data

- Initial data set :

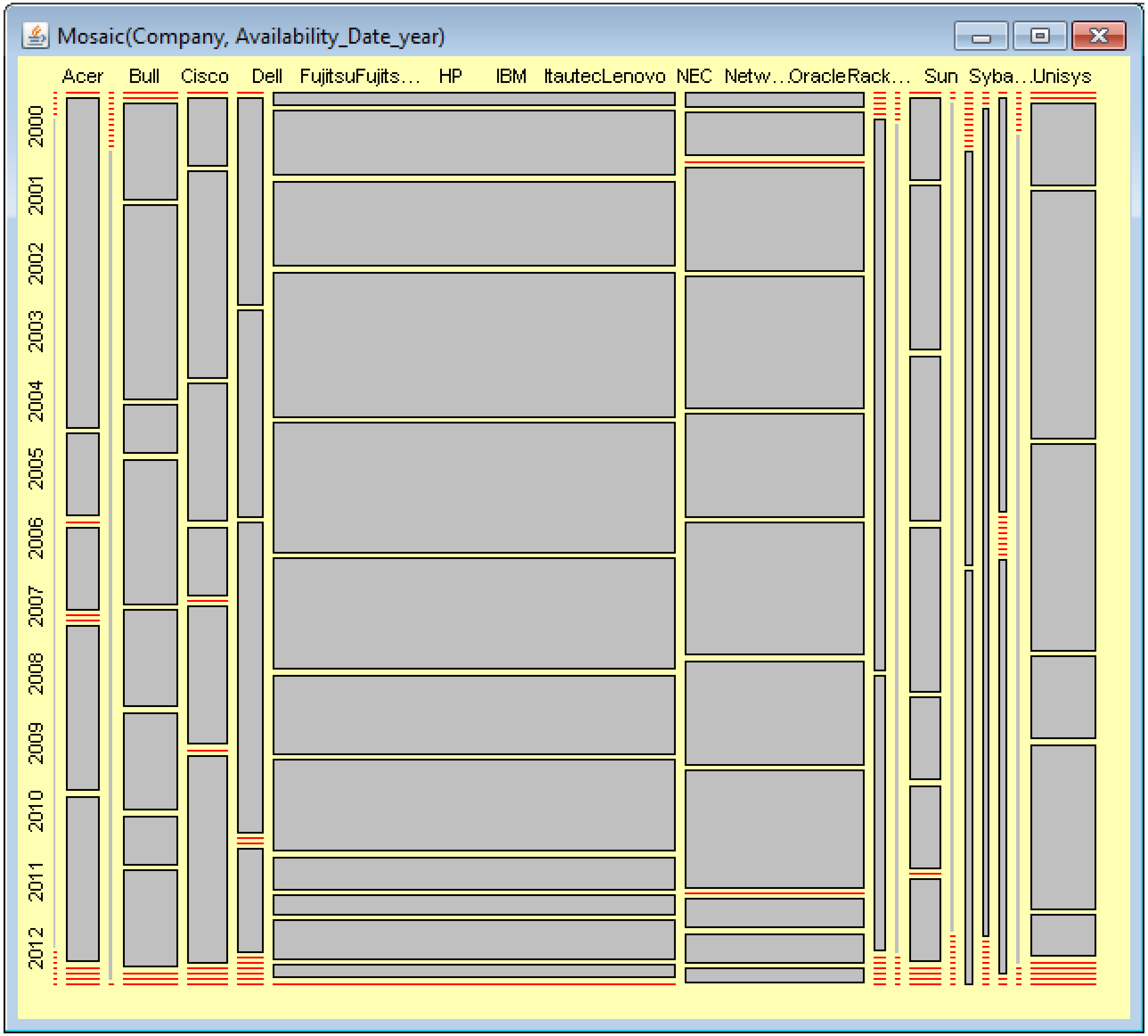
	A	B	C	D	E	F	G	H	I	J	K
1	TPC-C BENCHMARK RESULTS										
2	These results are valid as of date 6/12/2012 10:04:24 PM										
3											
4		TPC-C Results - Revision 5.X									
5											
6	Company	System	Spec. Revision	tpmC	Price/Perf	Total Sys. Cost	Currency	Database Software	Operating System	TP Monitor	Server CPU Type
7	Acer	Altos R710	5.5	66543	12.42	826507.55	AUD	Microsoft SQL Server	Microsoft Windows Serv	Microsoft CO	Intel Xeon - 3.6 GHz
8	Bull	Bull Escal	5.9	6085166	2.81	17127928	USD	IBM DB2 9.5	IBM AIX 5L V5.3	Microsoft CO	IBM POWER6 - 5.0
9	Bull	Bull Escal	5.9	629159	2.49	1566664	USD	IBM DB2 9.5 Enterpri	IBM AIX 5L V5.3	Microsoft CO	IBM POWER6 - 4.2
10	Bull	Bull Escal	5.8	1616162	3.54	5716286	USD	IBM DB2 9.1	IBM AIX 5L V5.3	Microsoft CO	IBM POWER6 - 4.7
11	Bull	Bull Escal	5.8	404462	3.51	1417121	USD	Oracle Database 10g	IBM AIX 5L V5.3	Microsoft CO	IBM POWER6 - 4.7

- Useless data :
  - Rows (e.g. the first and last few rows, not directly connected to the results)
  - Columns (e.g. „Server CPU Type” might not be necessary)
  - E.g. costs in different currencies
  - Decimal comma vs. decimal point
  - *Fujitsu* vs. *Fujitsu-Siemens* (merge it?)

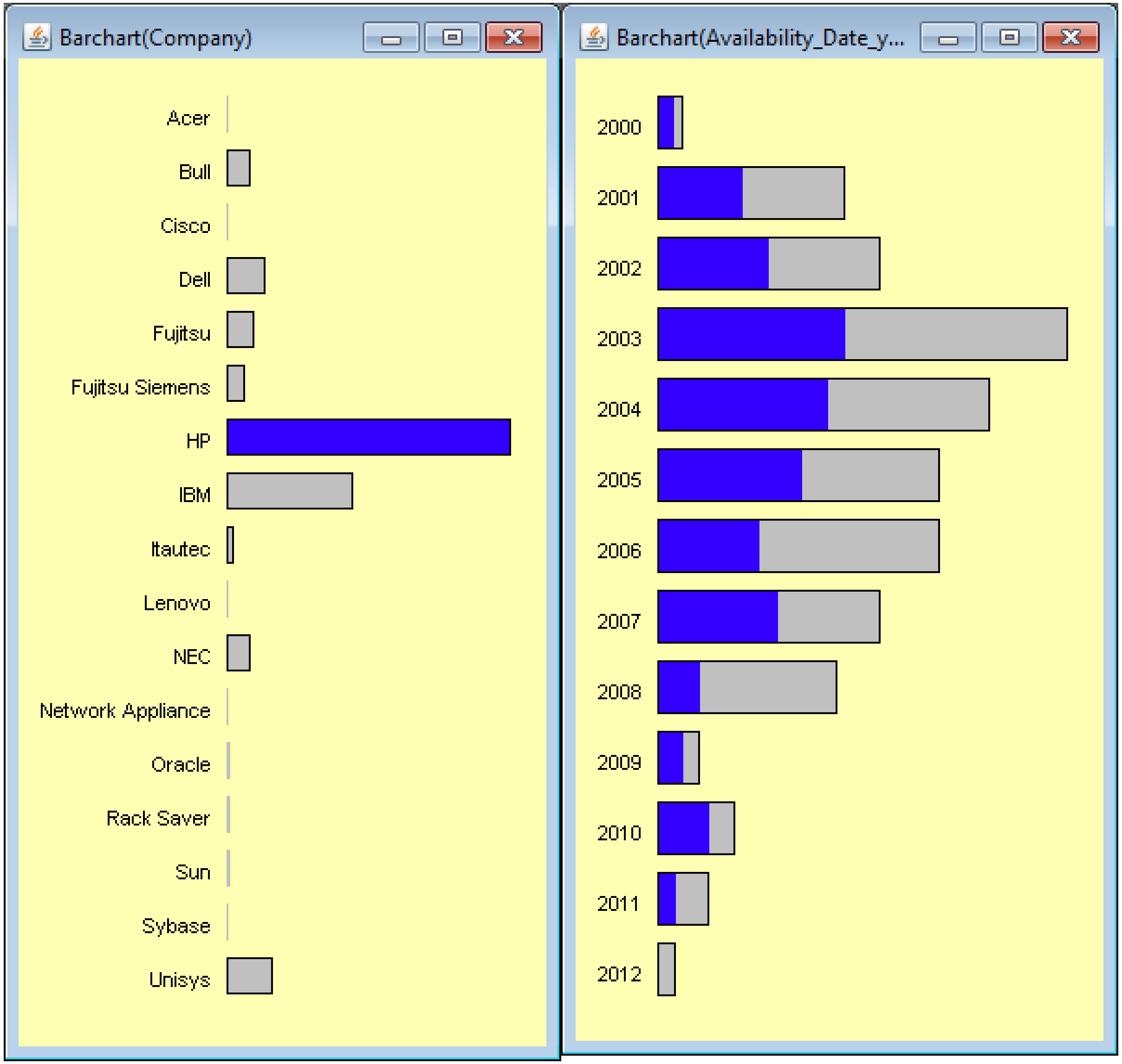
Which Years' Result does the Benchmark Contain?



When Were the Different Suppliers Active?

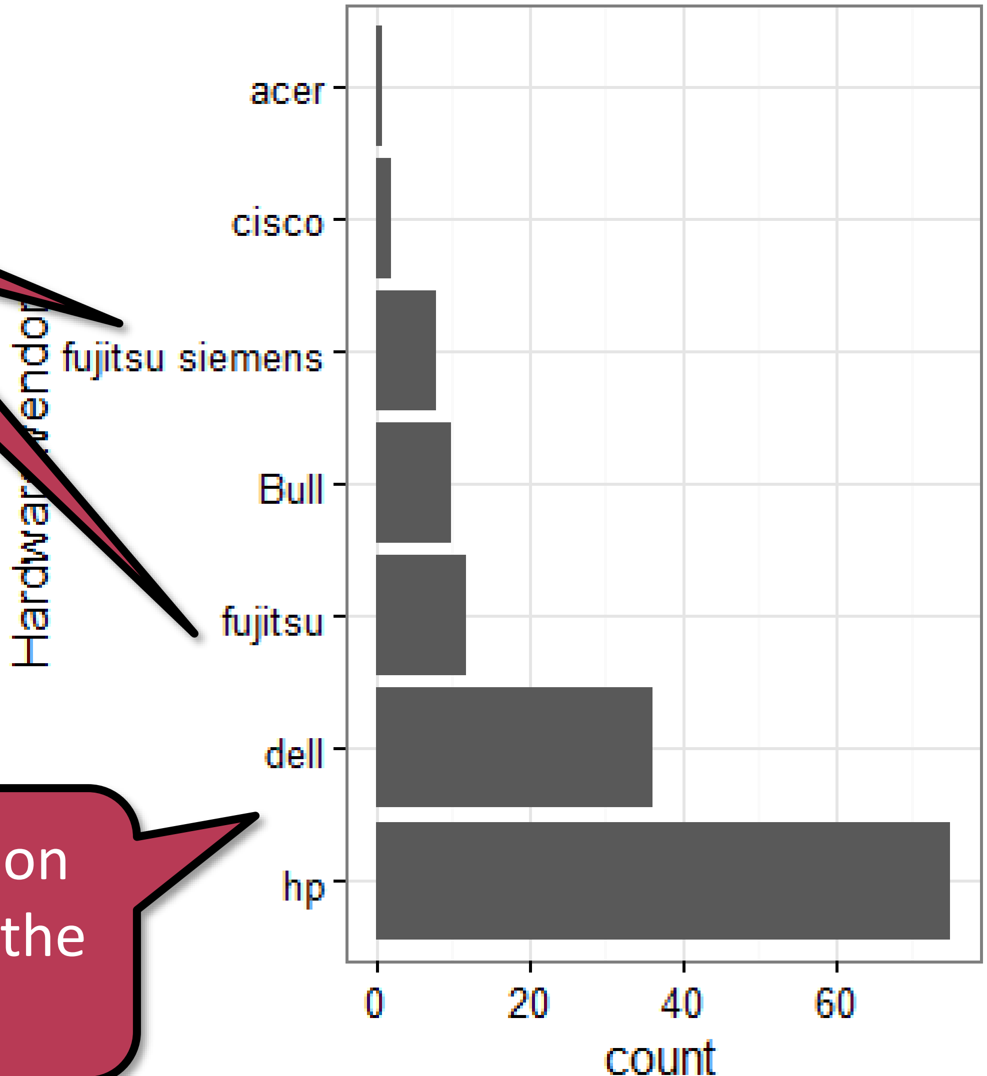


When Were the Different Suppliers Active?



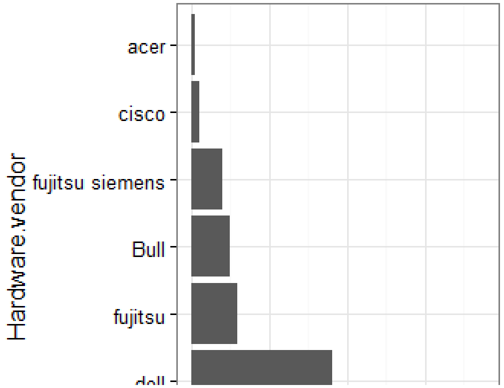
Measured Configurations

Should we merge here?

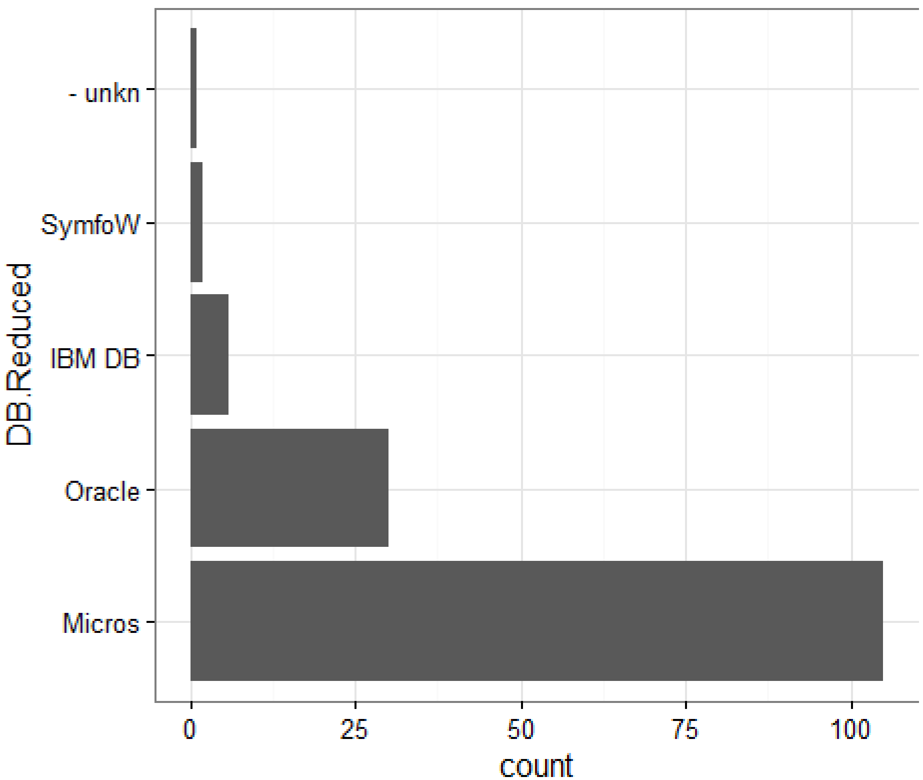


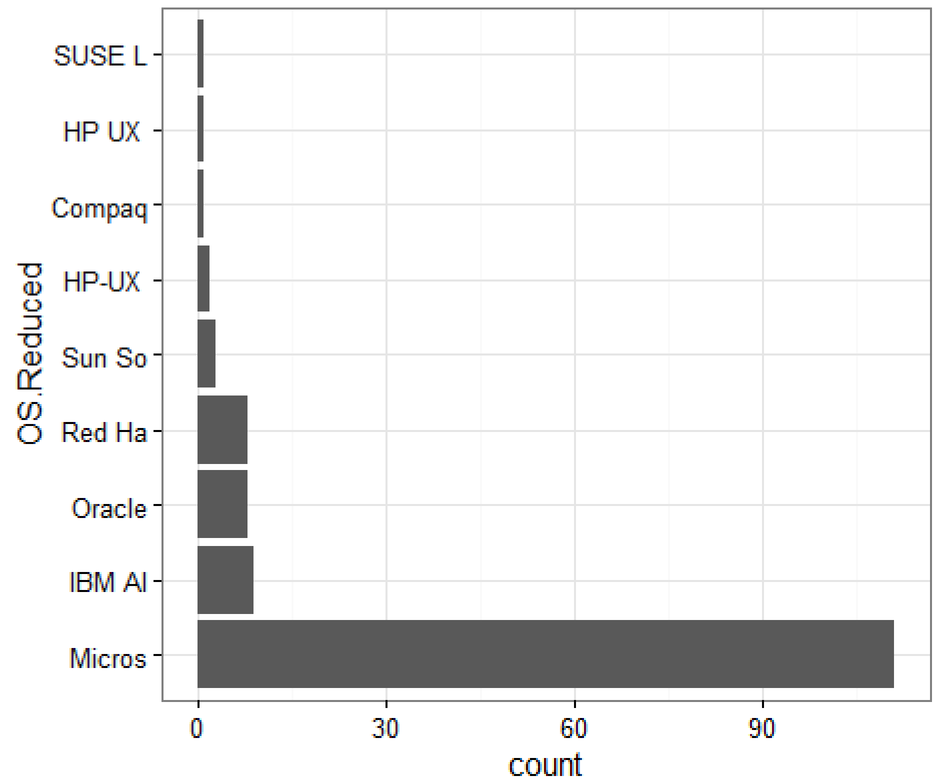
The two most common vendor cover 77% of the cases. Is it OK?

Measured Configurations

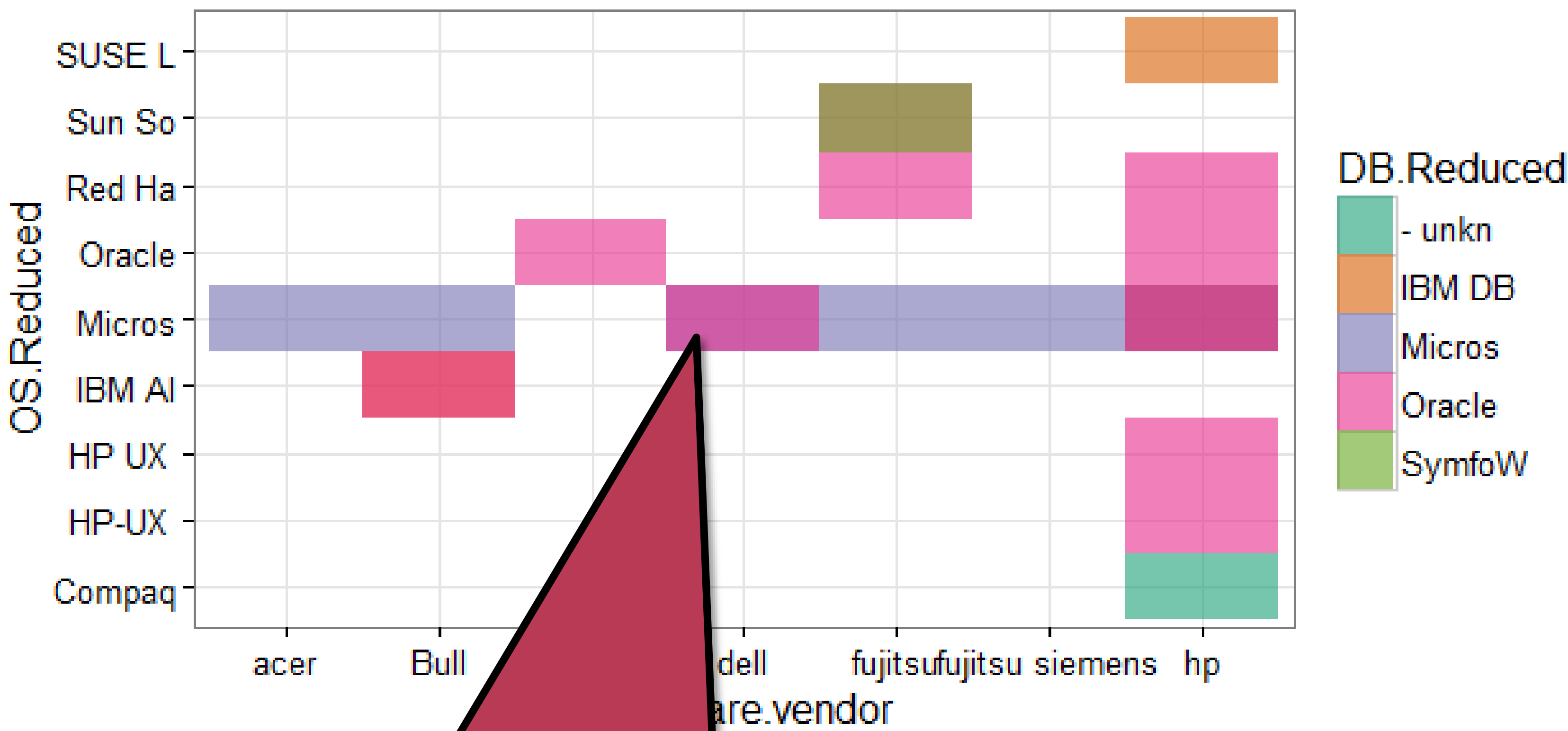


Merging by higher categories?



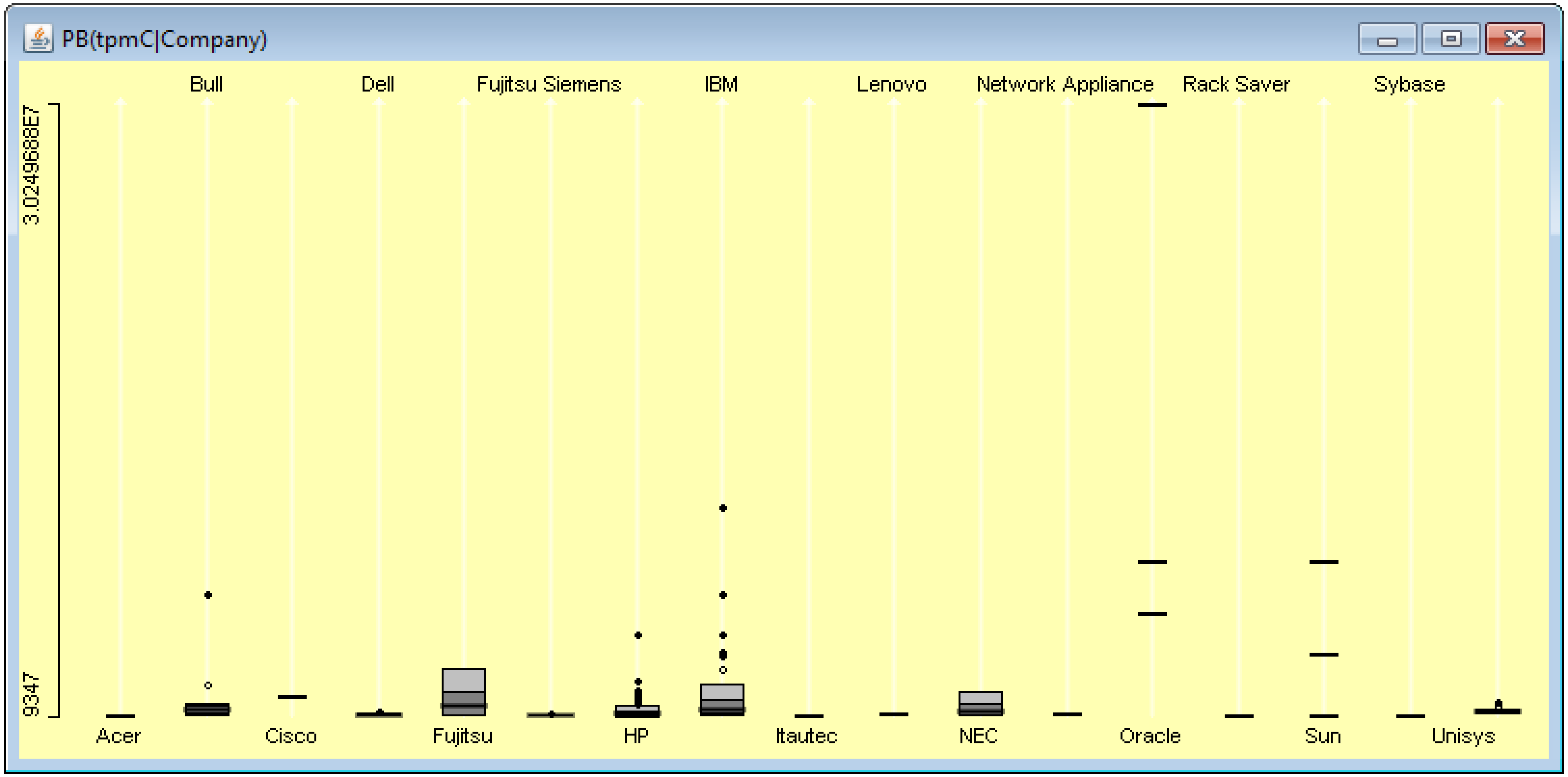


Measured Configuration Variations

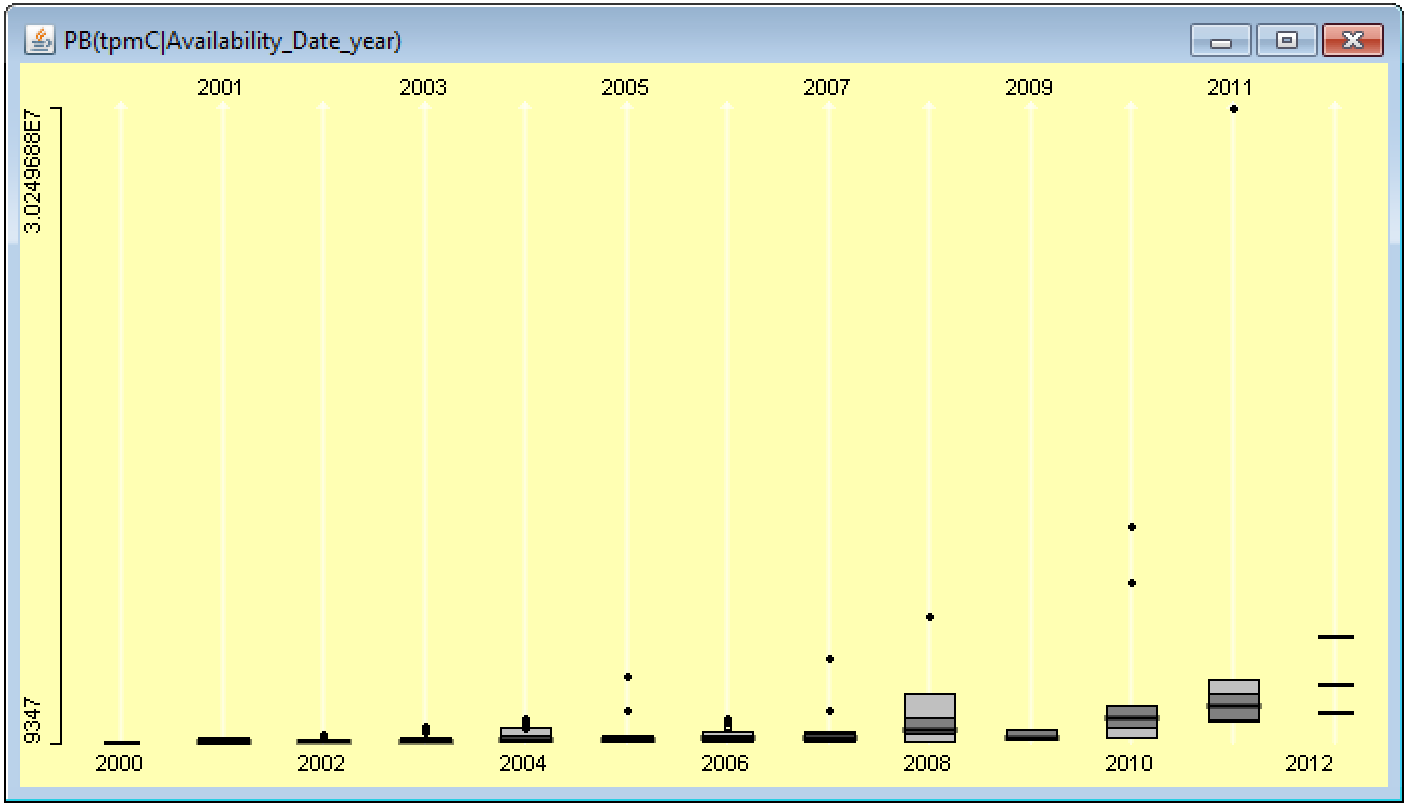


Database systems can run on multiple Oss

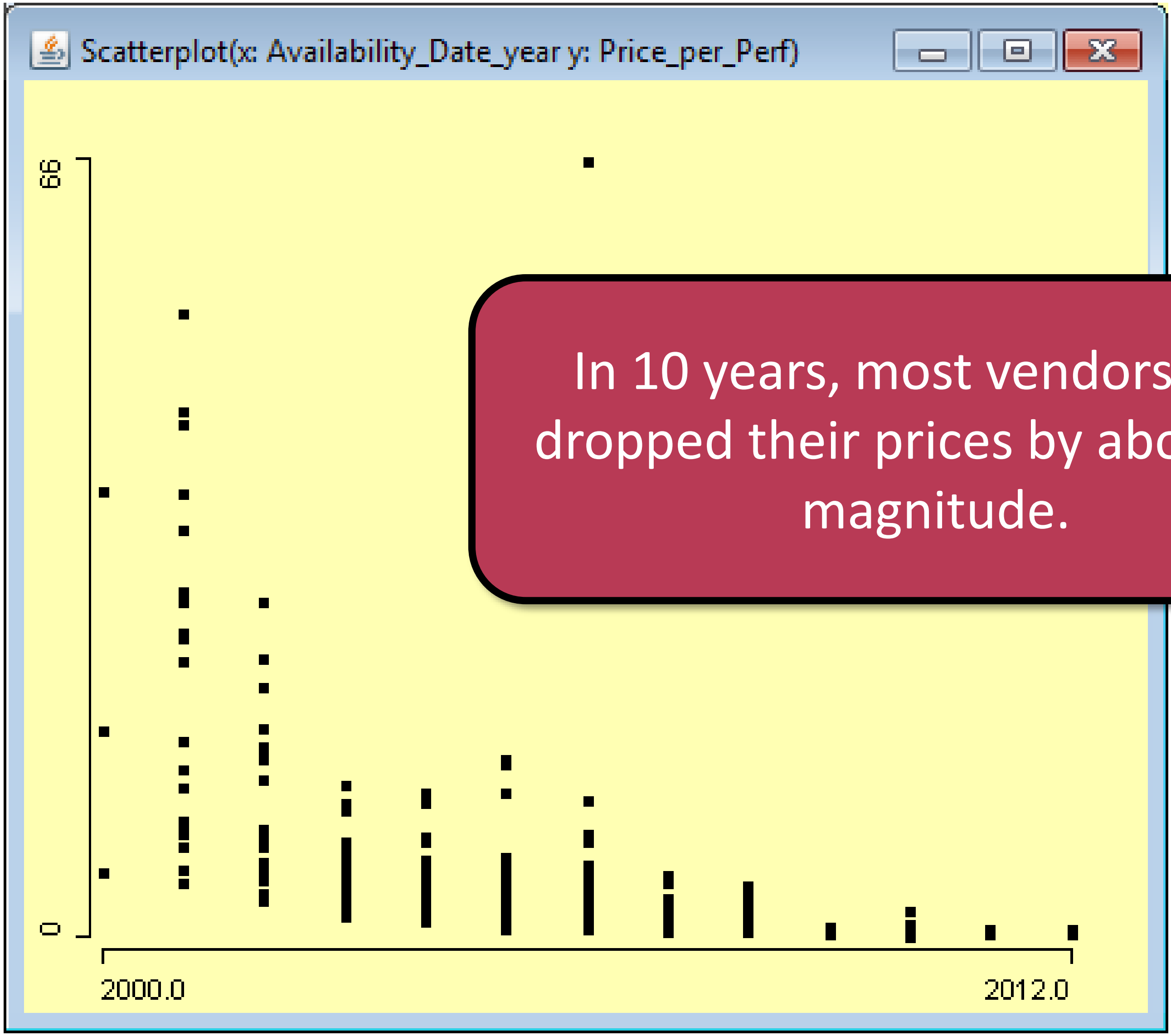
Result of Performance Metrics



How did the Performance Change over Time?

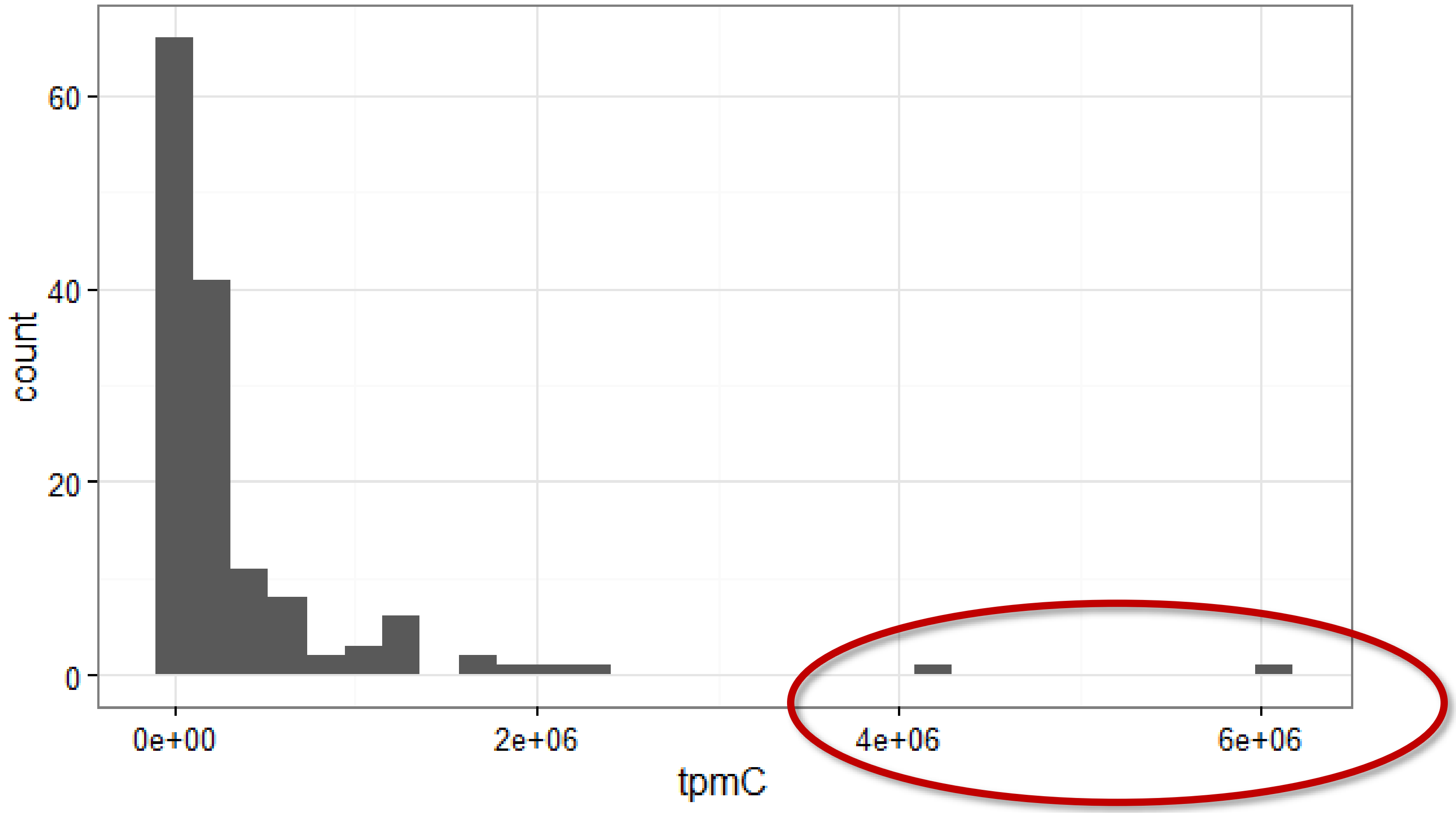


How did the Price Change over Time?



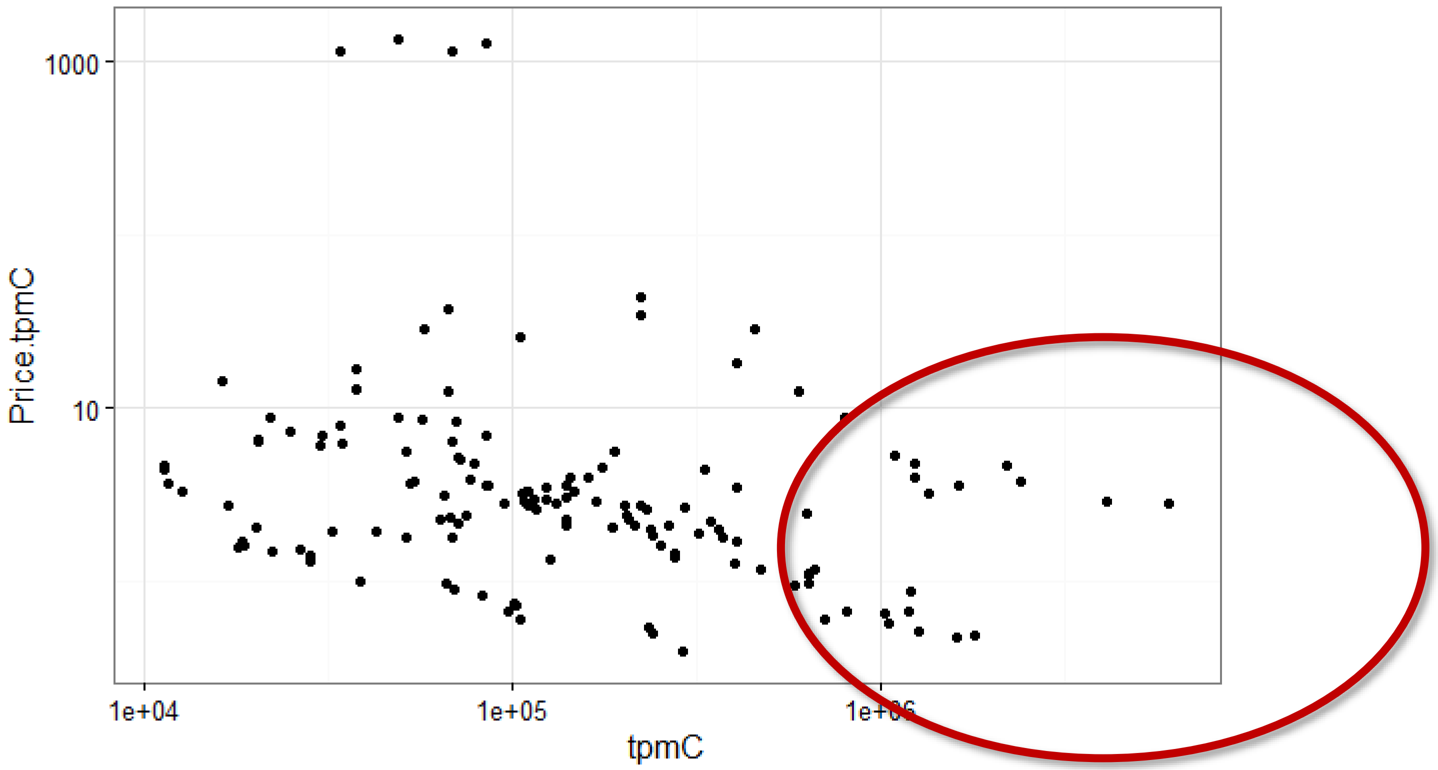
In 10 years, most vendors have dropped their prices by about one magnitude.

Benchmark Results



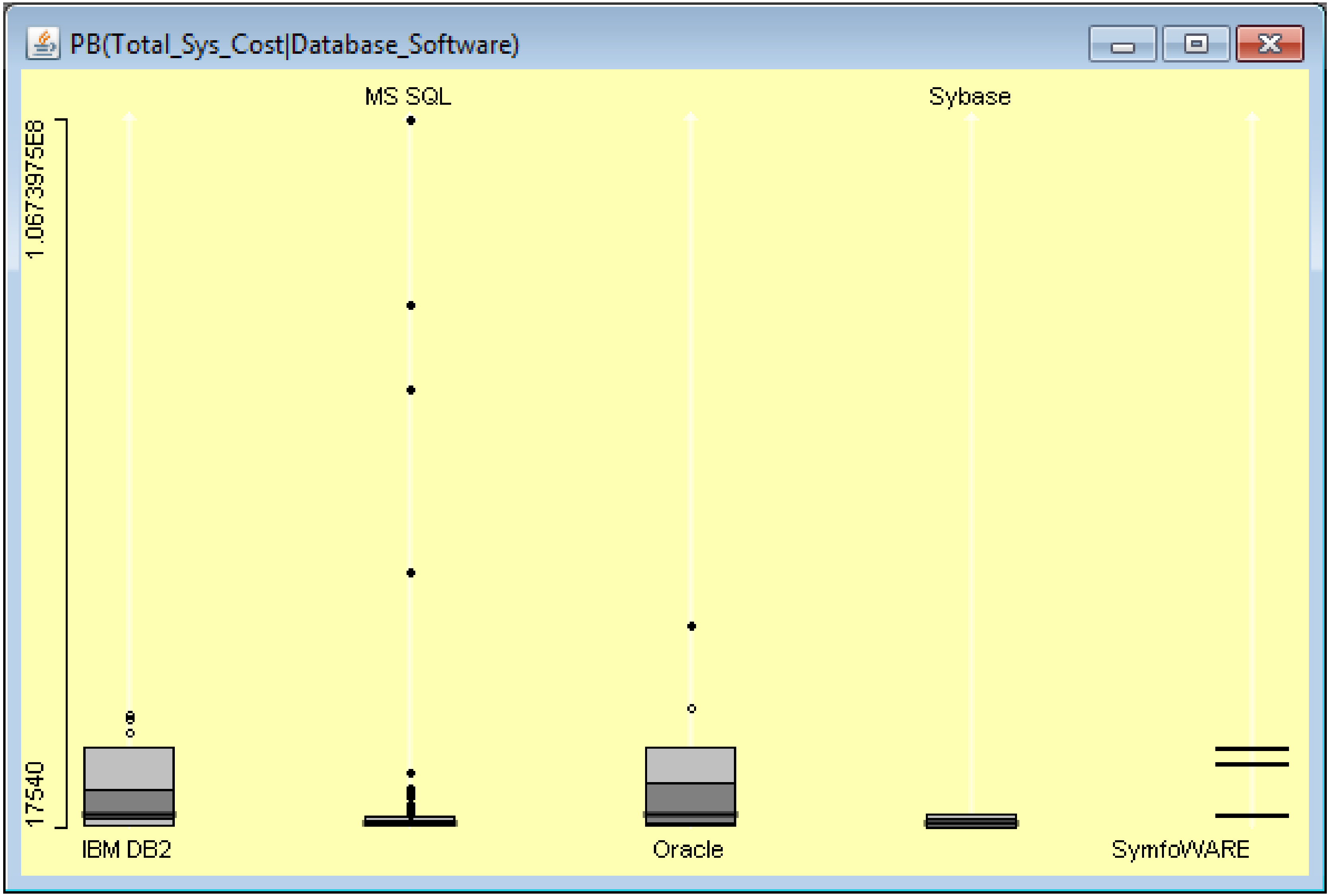
Price/Value ratio?

Benchmark Results

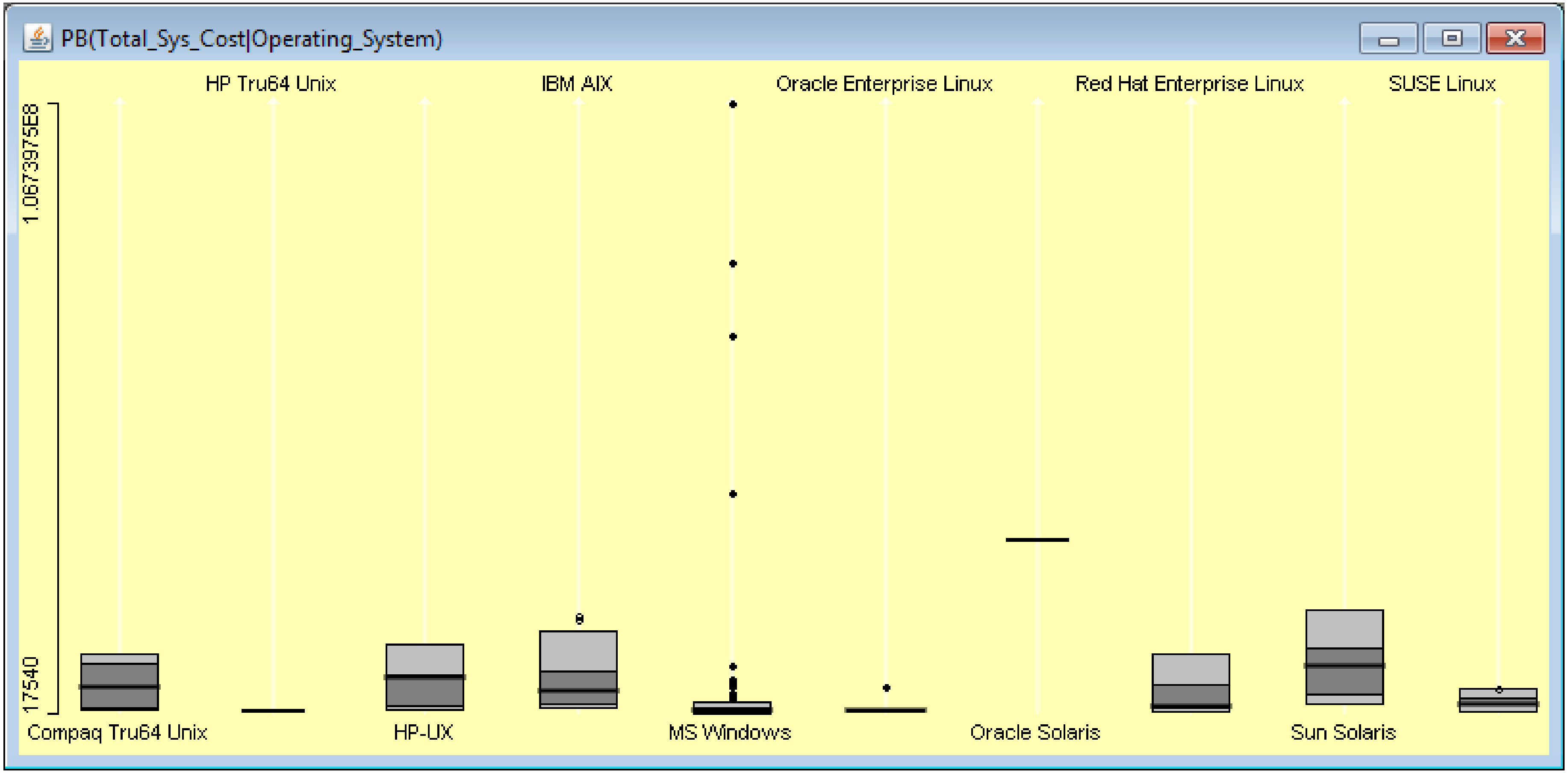


Logarithmic Scale?

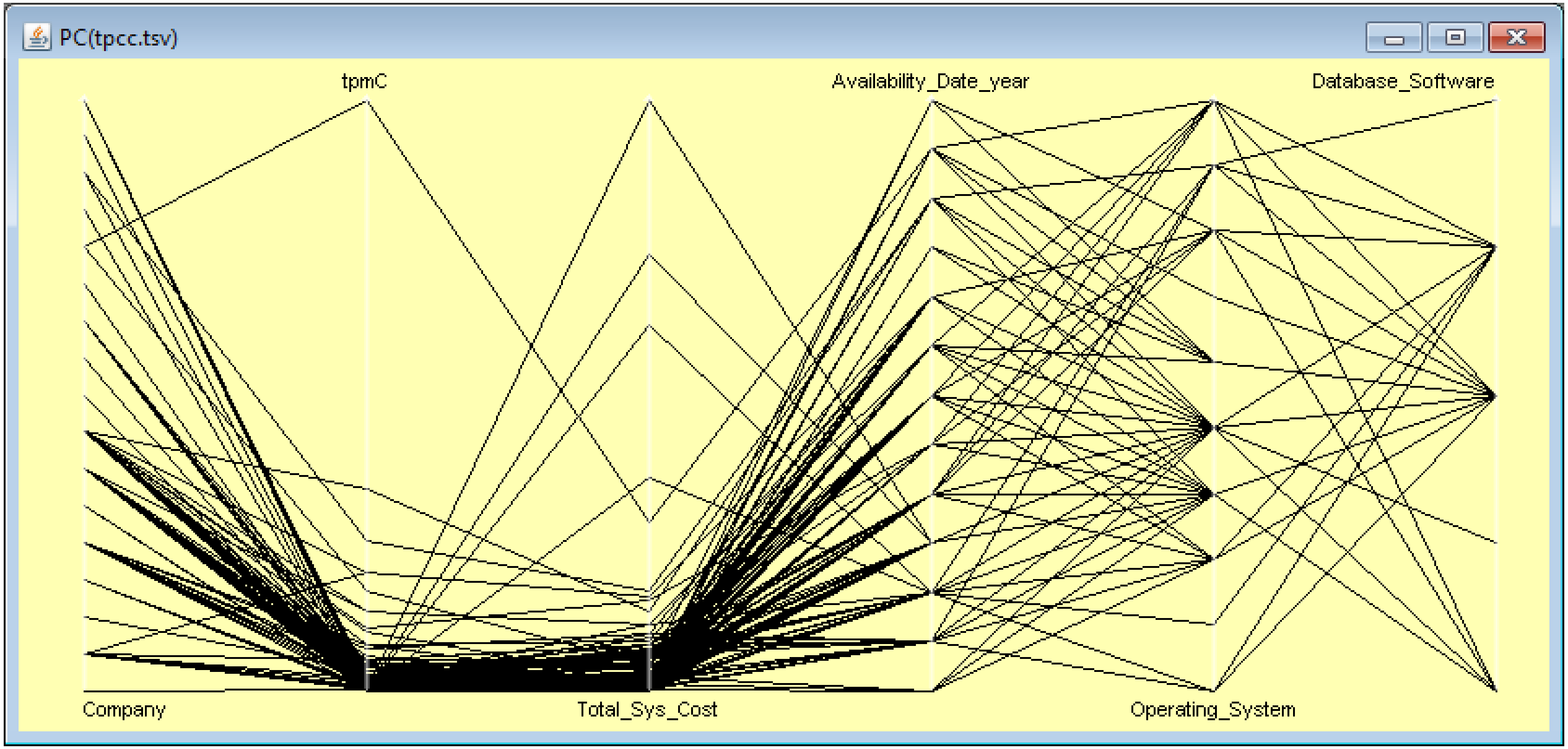
Which DB Manager SW Should We Choose?



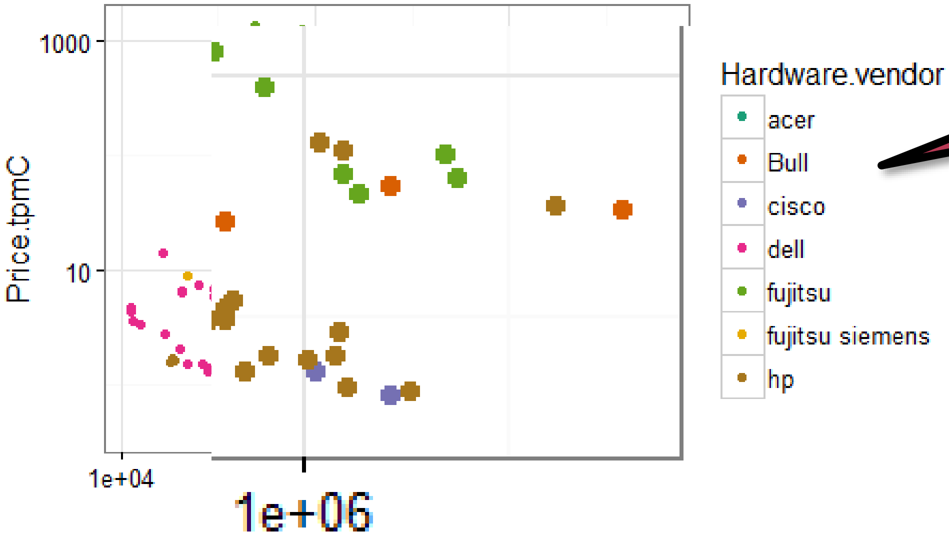
Which OS Should We Choose?



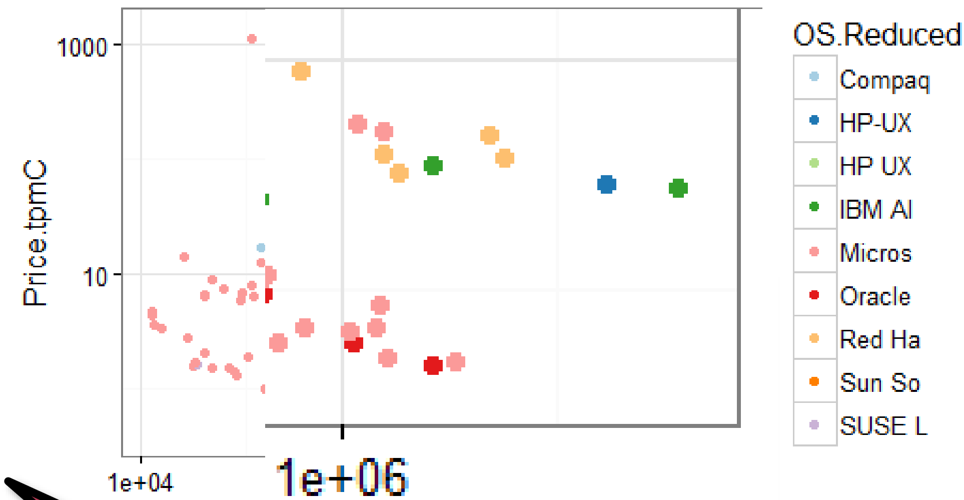
The „big picture”

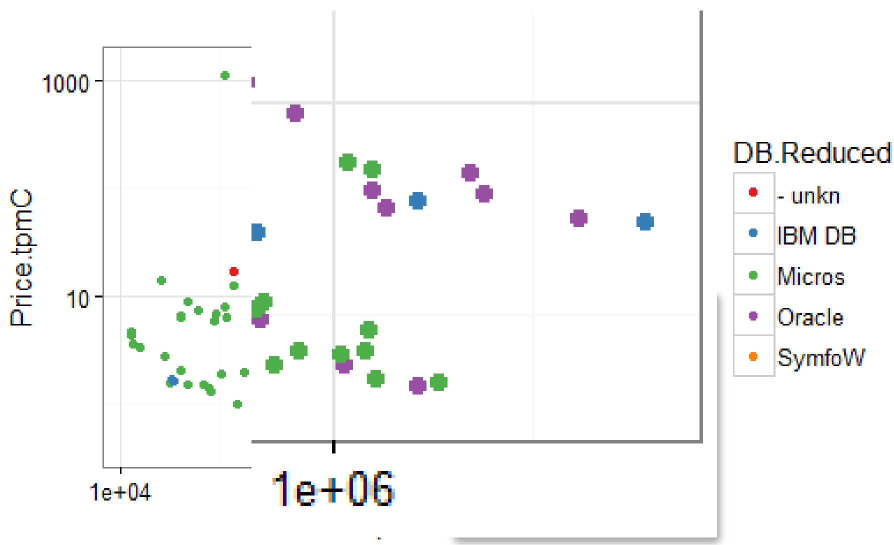


Benchmark Results



The leading group is rather manifold

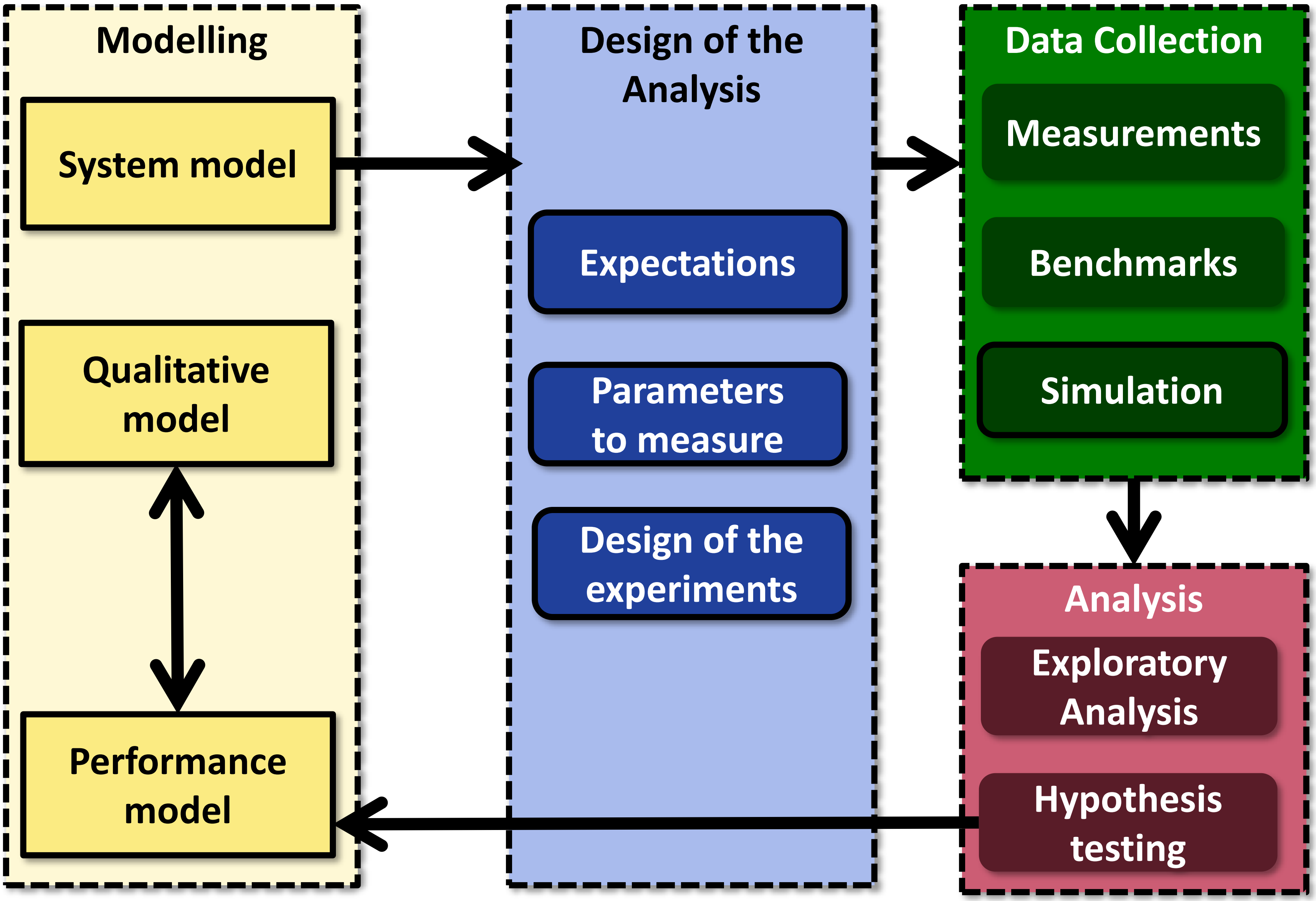




There is neither a best OS, nor a best DB-configuration

SUMMARY

Summary



Modelling
System model
Qualitative model
Performance model
Design of the Analysis
Expectations
Parameters to measure
Design of the experiments
Data Collection
Measurements
Benchmarks
Simulation
Analysis
Exploratory Analysis
Hypothesis testing