Empirical Assessment of Resilience

András Pataricza, Imre Kocsis, Ágnes Salánki, László Gönczy

Budapest University of Technology and Economics, Budapest, Hungary
{pataric|ikocsis|gonczy}@mit.bme.hu,
salanki.agnes@inf.mit.bme.hu

Abstract. Resilience is the ability of a system to return to its normal operation state after a change or disturbance. Frequently, resilience of a system can be only empirically estimated due to the complexity of the underlying mechanisms. While traditional dependability uses quantitative characteristics based on averaging the impacts of faults, resilience requires more focused attributes on the impacts of disturbances. The paper summarizes the main requirements on the statistical background needed for resilience characterization and presents an approach based on Exploratory Data Analysis (EDA) helping to understand disturbance impacts and their respective quantitative characterization.

Keywords: resilience, dependability, exploratory data analysis, statistics, quantitative characterization

1 Introduction

The word 'resilience' (from the Latin *resilire*: to rebound, recoil) as a general term expresses the ability to resist and/or recover from disturbances. It is part of the established terminology of multiple scientific domains; for instance, it is also defined in ecology [1].

Modern computer applications require more and more an analogous property to accommodate to changes in the environment, like e.g. qualitatively and quantitatively different workloads, hard to predict parasitic interactions between different users in a shared infrastructure or changing fault loads. In this sense, resilience as a system or service property can be defined as '*the persistence of service delivery that can justifiable be trusted, when facing changes*' [4] – in other words, the persistence of dependability [2] under changing circumstances.

Dependability is a design time attribute in the sense that it focuses on *anticipated* faults and their effects. In contrast, a resilient system by definition has to maintain its resilience properties if its environmental factors undergo *evolution*. Thus, 'resilience' encompasses the ability to resist and recover from errors, failures, changed environment, operational domains or requirements unknown at design time, as well.

Traditionally, quantitative metrics over a set of well-defined aspects as e.g. MTTF, MTBF or probability of failure on demand characterize the dependability of IT systems. A common attribute of such metrics is that they focus primarily on the *average*

adfa, p. 1, 2011.

impact of disturbances, as e.g. steady-state availability does; the evolution of the measures is typically characterized at most by (distribution) variance.

While ‘dependability services’ granted for the users can indeed be characterized this way, this is potentially insufficient or misguiding from the point of view of discovering and evaluating resilience characteristics, where a) temporal properties of transients and b) ‘worst case’ scenarios are under scrutiny. For instance, in high availability infrastructures rare, but long lasting outages may have little impact on overall availability while seriously violating resilience requirements (speed of recovery).

Quantifying resilience is an actively researched area as there seems to be no clear consensus yet even on the basic descriptive framework. E.g. [5] approaches resilience quantification from the networking perspective; [3] outlines how state-based models of dependability attributes (as for instance availability) can be adapted for resilience evaluation. [8] discusses benchmarking of resilience and introduces the notion of ‘changeload’ (analogously to the established concepts of ‘workload’ and ‘faultload’ [6]). [7] analyses Infrastructure as a Service (IaaS) resilience under capacity and demand changes and introduces the resilience (meta)metrics *settling time*, *peak over/undershoot* and *peak time* for measures that have an associated steady state between changes.

Research on resilience in general seems to employ either analytical or phenomenological statistical modeling approaches similar to dependability analysis.

- Traditional dependability analysis used for decades *closed world model* based prediction of the reactions of the target system to anticipate faults – assuming simultaneously the validity of the underlying fault-error-failure propagation mechanisms estimated and generalized from past observations on similar systems.
- However, resilience mechanisms have to mitigate not only rare, high impact events, but also *unforeseen* ones. It follows that the estimation of resilience necessitates an *open world model*, allowing for unknown faults and error propagation mechanisms.

The drawback of reusing existing analytical dependability models is that they incorporate only specific, already known mechanisms – while future changes may activate hidden ones that can invalidate the model itself. The same holds for the usual empirical (statistical) modeling approaches. Nevertheless, design for resilience necessitates the estimation of the impact of a wide range of potential environmental changes. This calls for a rethinking of the system characterization methodologies employed.

In this paper, we argue that *Exploratory Data Analysis* (EDA) performed on system observations is an invaluable – and in general terms, maybe the only practical – tool for estimating the previously unknown reactions of a system to environmental changes. Our concepts will be illustrated by reanalyzing the data of an independent, carefully executed dependability focused experiment [11] and showing how EDA can provide additional insight into resilience-related properties.

2 Exploratory Data Analysis

Modern statistics usually distinguishes two fundamental modes of data analysis: *Exploratory Data Analysis* (EDA) and *Confirmatory Data Analysis* (CDA). Exploratory data analysis '*is a well-established statistical tradition that provides conceptual and computational tools for discovering patterns to hypothesis development and refinement*.' [9]. Pioneered most famously by the American mathematician John Tukey (see e.g. [10]), EDA can be characterized as the approach of 'looking at data' with the fundamental aim of discovering patterns and building a plausible 'story of the data'. In contrast, CDA deals mainly with formal hypothesis testing and model selection.

In the 'Tukey school' of EDA (the term is generic enough to have slight variations across researchers), the following attributes are characteristic of EDA (based on [1]):

- An emphasis on *understanding the data*
- Graphical representations as the main driver of the 'detective work' adaptively traversing the observed data and making inferences from the phenomenological observations to root causes – thus creating a mental model of the observations
- An iterative process of hypothesis and tentative model specification, testing and respecification
- Flexibility and pragmatism regarding the methods used

Graphical representation techniques and fast, efficient data discovery featuring *interactivity between plots* and *data tours* are key to EDA.

2.1 EDA: core diagram types

In recent years – especially with the appearance of 'Big Data' problems in many domains – data visualization has reached new levels of sophistication and diversity. However, there is a core set of diagram types that is almost invariably present in EDA tools. Most of these, as e.g. scatterplots, histograms or barcharts are widely known. For our purposes we need to introduce *boxplots* and *parallel coordinates*; for an in-depth introduction to the field see e.g. [12].

Boxplots [14] visualize the 'five-number' summary of the distribution of the observations of a single variable: its lower extreme, lower hinge (practically the first quartile), median, upper hinge (third quartile) and upper extreme. A common variation is to define 'outliers' as observations out of the 1.5 IQR (InterQuartile Range; the difference of the third and first quartiles) distance from the first and third quartiles; these are plotted as distinct points. Non-outliers below and above the hinges can be represented by so-called 'whiskers'. It is also customary to use boxplots for examining the interactions between a categorical and a continuous variable in a set of multivariable observations. In this case, each category value is assigned a distinct boxplot. Fig. 1 shows such an example boxplot, based on the data set analyzed later on.

Parallel coordinates [16] is a technique to visualize N-dimensional data in the plane. N equidistant parallel axes are drawn to represent the individual variables; to each observation corresponds a polyline connecting its variable by variable values.

The variables are usually normalized (see e.g. Fig. 6 later). Numerous statistical properties translate to visually easily recognizable patterns in parallel coordinates, as e.g. negative linear correlation into lines crossing in one point between two axes.

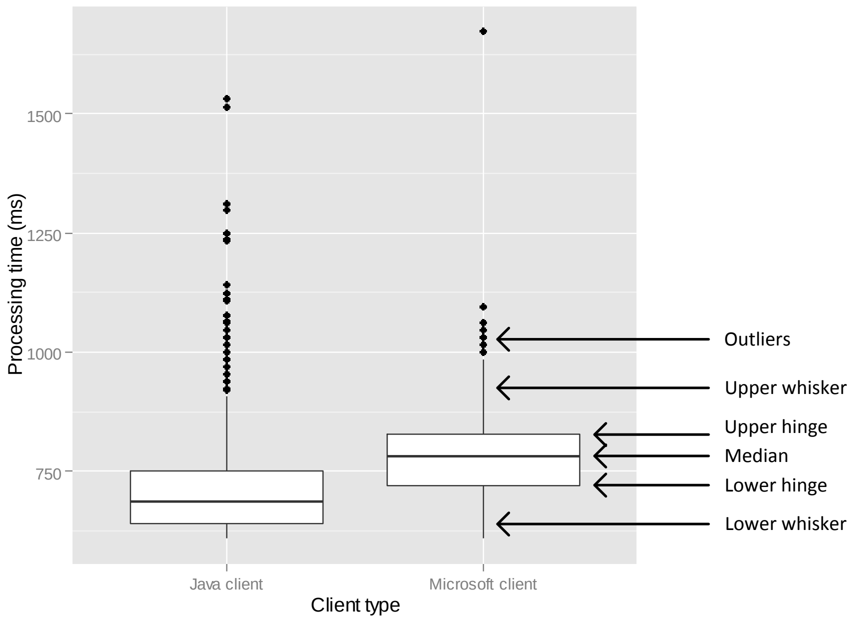

Fig. 1. Boxplot example: processing time by client type in a cloud response time experiment.

2.2 Interactivity

The most important interactive technique in EDA for our purposes is *selection and linked highlighting*. Selection and linked highlighting means that the interactive selection of a subset of observations on a plot will be immediately reflected on all other active plots. Our analysis will include examples of this technique. For other techniques as *querying*, *zooming* or *color brushing* see [12].

2.3 EDA tools

Strictly speaking, EDA can be performed using static diagrams without interactivity or data tour support; in this sense practically all statistical packages and even modern spreadsheet applications can be valid choices. However, many tools provide the full spectrum of capabilities; the most notable open source examples are Mondrian [12], GGobi [17] and the R [15] packages iPlots [18] and iPlots eXtreme [19]. Certain offerings from TIBCO, IBM and SAS are also feature-complete modern EDA tools in the above sense.

2.4 EDA as a process

In practice, the steps performed by a professional data analyst tend to fall into the natural continuum between 'pure' EDA and CDA, the emphasis shifting into the direction of CDA with the progress of the analysis. It is worth to note and is actually regularly emphasized by statisticians that EDA by its nature is a very ad-hoc process; at first glance it may seem to be a random search for 'interesting plots'. While this holds in some cases, there are statistical techniques for suggesting graphical representations that may be worth checking (e.g. so-called *guided tours* [13]). Note that EDA and CDA are complementary: for instance visual clustering (identifying an agglomerate of data) can guide algorithmic clustering by delivering a rough initial model.

Also, our example will show that generic domain specific knowledge and some rules of thumb as 'check marginal distributions first' naturally give rise to a sort of proto-workflow. However, as such questions are out of scope here, we will discuss these aspects in future work.

3 Requirements for statistical methodology

The general definition of resilience poses further requirements on the statistical analysis process in addition to rely on open world models: (i) the observations should drive model building without restrictive assumptions originating in the underlying mathematics, as resiliency has to cope with unexpected behavior and phenomena (e.g. no a priori distribution on the occurrence of faults/intrusions can be assumed), (ii) the estimated normal operation domain model should be highly insensitive to change impacts as no a priori restrictions should apply on the change impacts.

3.1 Non-parametric statistics

Non-parametric statistics addresses data analysis and modeling without taking restrictive assumptions on the data observed and the model structure fitted to them. Using non-parametric statistics results in a high degree of independence of the modeling mechanics.

For instance, distribution free statistics can be generally used over arbitrary data sets without a priori setting the statistical models, inference and statistical tests. Similarly, independence of a pre-specified model type leaves both the structure and size of the model as free parameters to the modeling process.

3.2 Robust statistics

Changes may manifest typically as rare outliers of the normal operation. The requirement of a clear separation of normal and disturbed operation states in resilience analysis means in the terms of statistics that individual disturbances should result in a reasonably small bias in the characteristics of the normal domain; moreover, if normal operation dominates, its characteristics should be asymptotically unbiased. The importance of robustness is illustrated by the following example:

Let assume that the reaction time of a web service lies in the range of [1, 5] ms in the normal operation domain. The traditional characterization of this web service is done by the mean of its reaction time having a value of 3 ms. A single fault in one of 1000 transactions may lead to a reboot lasting for 20 s. This may clearly distort the weighted sum of response time mixing up two essentially different metrics. The impact of an outlier representing the failure may be unlimited, thus the mean is inappropriate to characterize a system with no restrictions on fault impacts.

On the other hand, median, the value cutting the set of ordered observation values into two equal cardinality parts is a robust characteristic. In our example the outlier counts only by its number of occurrence independently of his magnitude being only one sample therefore median remains a little biased characteristics of the normal domain. Naturally, a separate characterization of the faulty domain has to be elaborated after separation.

4 Case study: EDA on cloud performability observations

In this section we present an EDA process performed by us on the data underlying [11]. Two separate analysts worked on the data in a loosely coupled way - one in full knowledge of the previously published findings and one not knowing those - reaching the same conclusions. Omitting overhead factors, the analysis (including understanding the measures) took approximately one day.

4.1 The dataset under analysis

The previous work's basic goal is to compare the performance of Microsoft Azure, GoGrid and an in-house server from the point of view of the clients, using a (remote) web service and taking into account the communication delay. The end-to-end response time (RT) is defined as the sum of the server-side request processing time (RPT) and a network round trip time (RTT). The benchmark web service run at the server side is a compute-intensive sorting task on data sent by the clients.

This tutorial is based on a one-day portion of the dataset of [11] which was recorded by tests on Microsoft Azure instances. The servers ran in the Microsoft Azure Data Centers located in Dublin (Ireland) and Redmond (USA). Clients were set up at 12 locations in the USA, 3 in Europe and 2 in Canada. The client application has a Java as well as a C# implementation. Requests were sent by the clients every minute; the resolution of the observations is one millisecond. The examined data set contained the following attributes:

Timestamp	Client IP	Client location	Client type	Server location	RPT/RTT/RT (ms)

4.2 Reconstructed workflow

Our exploratory data analysis followed the structure shown in Fig. 2. In the first column, the flow of the high-level goals is shown. The next two columns describe the generic EDA approach and its specific manifestations. In the last column we noted the main findings reached. In the following, we will present this process step by step and show how the visual techniques lead to findings as well as to subsequent steps. Note that the workflow is ‘reconstructed’ in the sense that it was not preplanned – the partially ad-hoc process was documented along the way and reassembled at the end.

Throughout the process, we used Mondrian as our EDA tool of choice[1]; Mondrian is completely mouse interaction driven and its usage does not need scripting. This enables rapid data exploration.

EDA findings 1, 2, 4.1 and 5, to our knowledge, were not discovered in the original work. (Note that there the main objective was goal metric driven phenomenological characterization instead of full-scale EDA.)

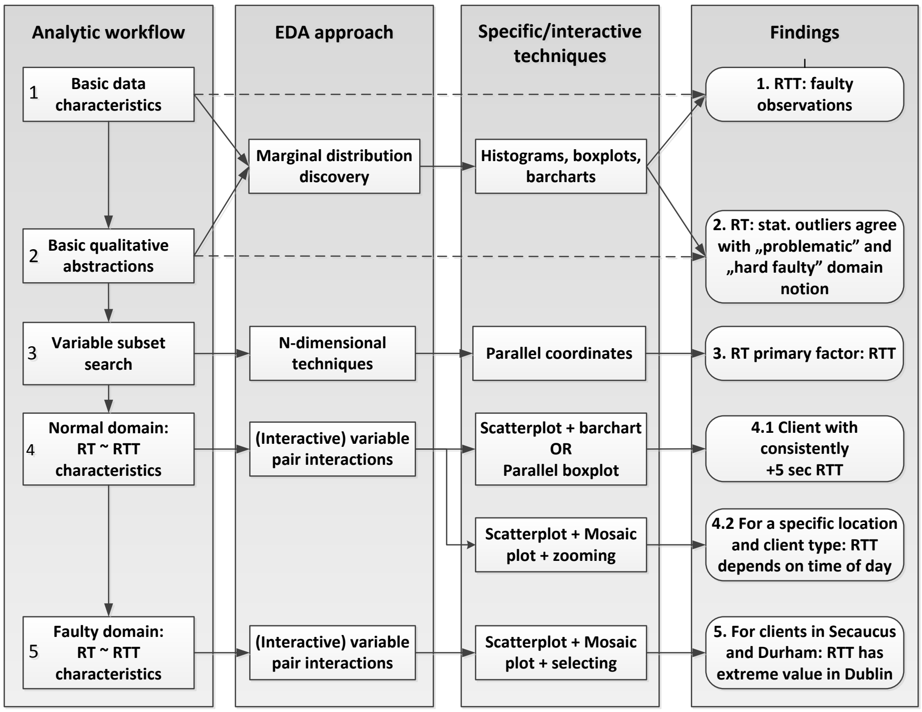

Fig. 2. Overview of the performed exploratory data analysis.

1 Note that some figures in this section were not taken from Mondrian, but reconstructed in R using the static visualization package ggplot2 [20] to improve legibility and figure quality. There are only aesthetic differences between the original plots and the alternatives presented.

4.3 Basic data characteristics

With marginal distribution discovery, one can get a first impression about the characteristics of single variables: whether they contain ‘NA’ (missing) values, what kind of distribution they follow, which values their quartiles take on. Based on our experience, this step is vital for understanding and validating the data to find e.g. inconsistencies between the recorded values of an attribute and its theoretical domain. (A few such minor errors were found and reported to the authors of [11].)

4.4 Basic qualitative abstractions

For resilience assessment, another goal of marginal distribution analysis can be defining an initial qualitative discretization of the ‘goal’ variable (here RT).

A boxplot of RT has shown that there are significant outliers (some are even over 50 s) that visually suppress the non-outliers. Zooming in on that boxplot we remove the points over 20 sec (19 observations), resulting in Fig. 3. Based on the filtered boxplot, it becomes apparent that the majority of the observations (~92%) are roughly in the [500 ms, 3000 ms] interval. Another set (~7%) seems to be tightly grouped in approx. [3000 ms, 8500 ms]; and we have points with even higher values (0.4%). This categorization inspired by basic statistical properties would be even acceptable from the engineering point of view, too. (Note that we test web services, not interactive web pages.)

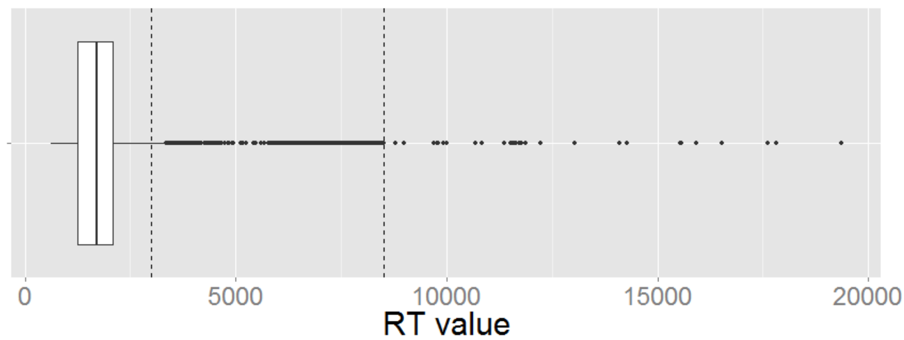

Fig. 3. ‘Zoomed’ boxplot and histogram of RT values with dashed lines at 3000 and 8500 (ms)

Although the boxplot is a very compact and efficient tool, it is also useful to examine the histogram of the variable (see Fig. 4). Based on that, we decided to refine the intervals to [0, 5000], [5000, 7500] and (7500, ∞] – this is a more natural quantization from a statistical as well as engineering point of view.

Based on their plausible engineering interpretation, we term these intervals the ‘ok’, ‘problematic’ and ‘faulty’ domains of the service. In the following, when we speak of a ‘normal domain’, it will mean the union of ‘ok’ and ‘problematic’.

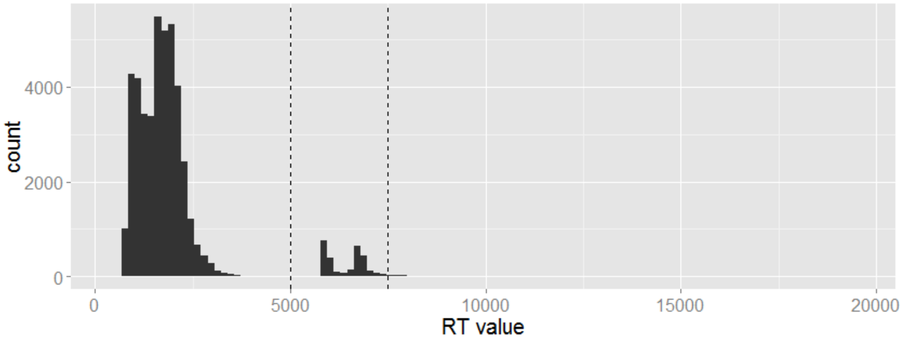

Fig. 4. Histogram of RT values smaller than 20 s with dashed lines at 5000 and 7500 ms.

4.5 Variable subset search: faulty domain

One of the key parts of the analysis is to figure out that which are the variables with the most significant impact on the goal variable. It is advisable to perform feature selection for each discovered ‘operational domain’ as the underlying phenomena may be different.

Analysing the faulty domain with a parallel coordinates diagram, it can be seen that by far the strongest relationship is that the high values of RT coincide with the high values of RTT (see Fig. 5 and Fig. 6). Note that the RTT/RPT ratio has a typical value around 2, in extreme cases reaching 200. This shows that only a small portion of RT is spent with server-side computing. This way, we can formulate the hypothesis that in the faulty domain RT is predominantly defined by RTT.

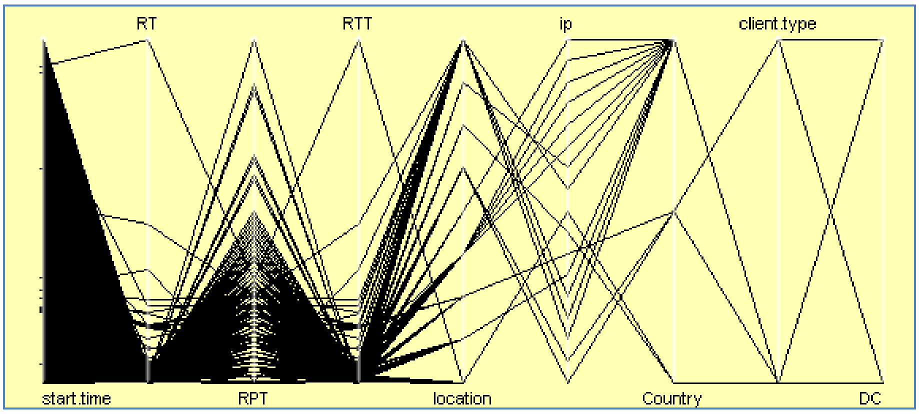

Fig. 5. Parallel coordinates plot of the whole data set (without selection)

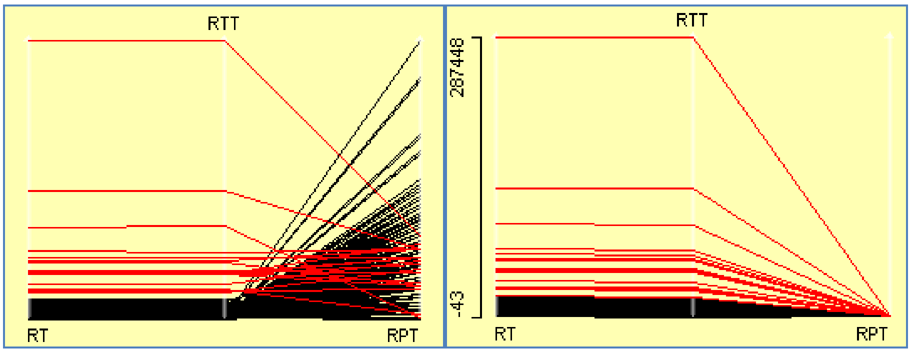

Fig. 6. Normalized and 'common scale' (ms) parallel coordinates; selection: 'faulty' domain

4.6 Variable subset search: normal domain

In the normal domain, one can find a similar strong relationship between the RTT and RT metrics; this can be seen e.g. on a scatterplot where the RT values are plotted against the RTT values (Fig. 7). With Mondrian we can even fit a linear regression line on the scatterplot (with a very good fit in the statistical sense). In practical terms this means that RT almost equals RTT plus a quasi-constant offset.

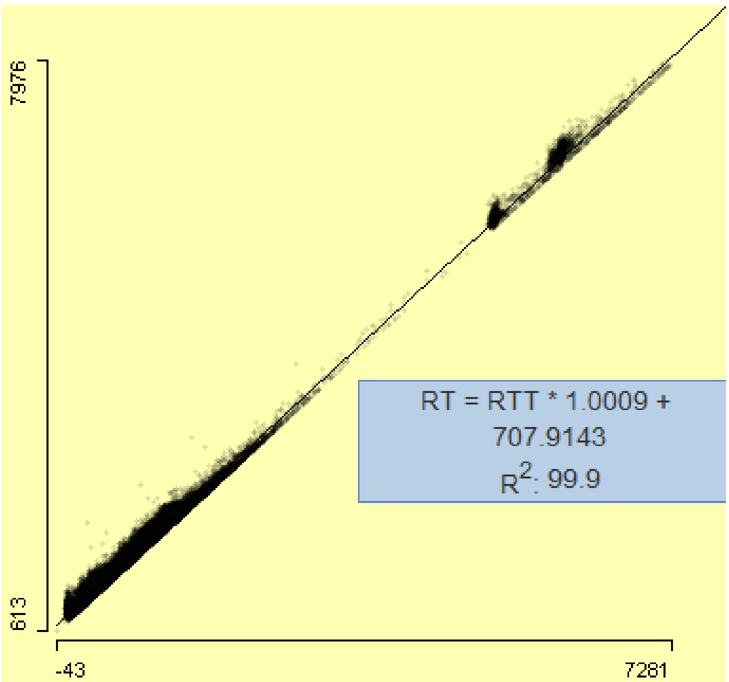

Fig. 7. RTT versus RT scatterplot (ms/ms) in the normal domain with linear regression

4.7 Relationship analysis in the normal domain

In the first step, we defined the 'normal' domain as the union of two suspiciously distinct clusters of values. Following up on that, we investigate the relationship of this phenomenon with our tentative linear model. Fig. 8 shows an ensemble of three plots; the 'problematic' cluster was selected on the scatterplot. In turn, this leads to the following findings: a) from the histogram: unsurprisingly, this indeed covers all 'problematic' RT observations and b) from the mosaic plot[2]: these observations belong almost solely to one specific client in Lansing.
Observing the distributions of the RTT values of different clients from Lansing (Fig. 9), it becomes clear that one of the clients shows a constantly with 5 second higher delay than the others from the same location. What makes the phenomenon interesting is that this "suspicious" client shares one subnet with one of the machines with normal RTTs, so the probability of a subnet-dependent fault is low. The anomaly is most likely caused by different firewall settings. This can be seen obviously on the time series of the clients as well (Fig. 10).

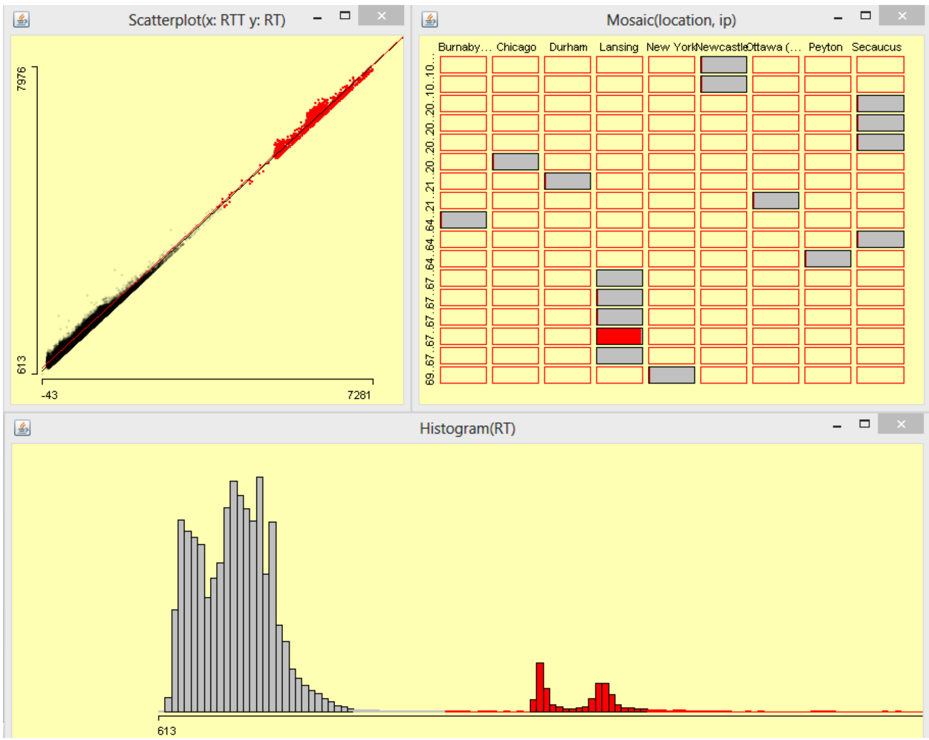

Fig. 8. Linked plots for discovering the client coverage of the 'problematic' category (clockwise: zoomed scatterplot, artificially 'same tile size' mosaic plot, zoomed histogram)

[2] The mosaic plot (see [12]) here serves simply as a multidimensional 'switchboard' representing observation sets with specific IP address – location combinations as 'tiles'.

The explanation for the ‘problematic’ cluster we found with time-independent plots. However, there are phenomena that can be detected only with time series visualization. The result about the dependency of RTT values on the hour-of-the-day presented in the original article is a good example for this.

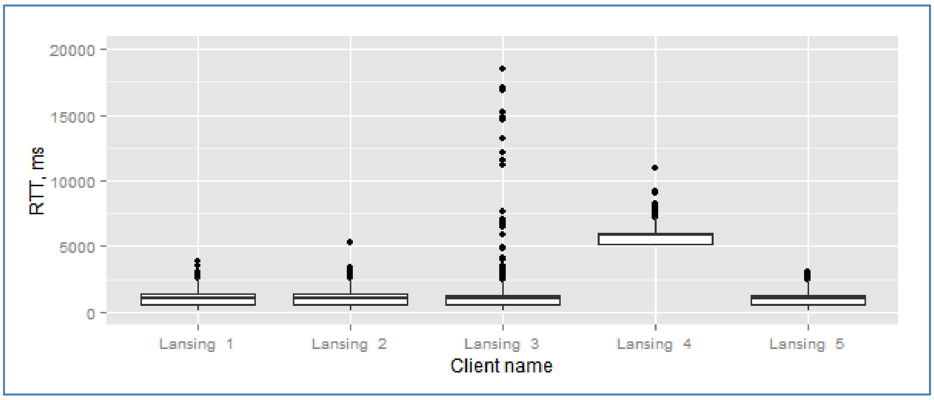

Fig. 9. Lansing client RTT observations as boxplots

Those results were produced by a client in Newcastle which run the Java client; it quickly became clear to us, too, that the RTT time series of different client types diverge from each other. Filtering to the Java clients, the visual detection of the hour-of-the-day dependency needed only a barchart of IP. Clicking through the bars and analyzing the corresponding RTT time series, the ‘interesting’ time series become recognizable (Fig. 11).

We would like to note here that the tentative hypotheses we are reaching should be actually treated as such; for instance, before deciding that there is a fault mode where the RTT to Dublin radically depends on the time of day, we have to control for other factors as e.g. client type, too. (In this case, although the observation set is not balanced with respect to client types, the hypothesis is reasonable.)

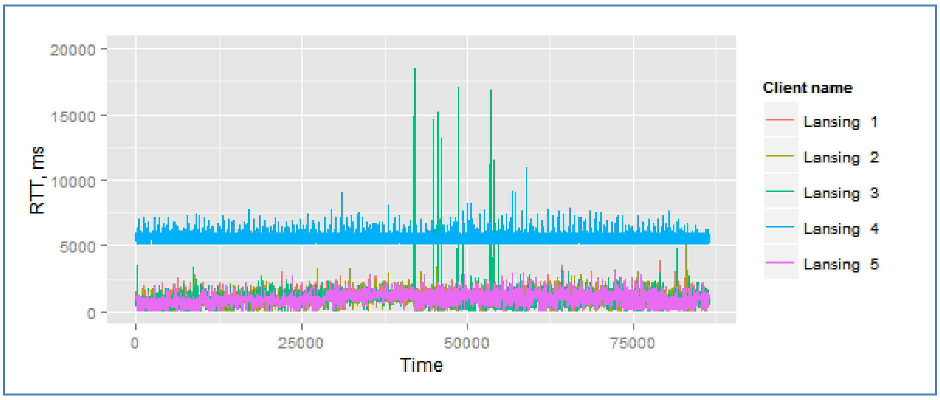

Fig. 10. Lansing client RTT observations as time series

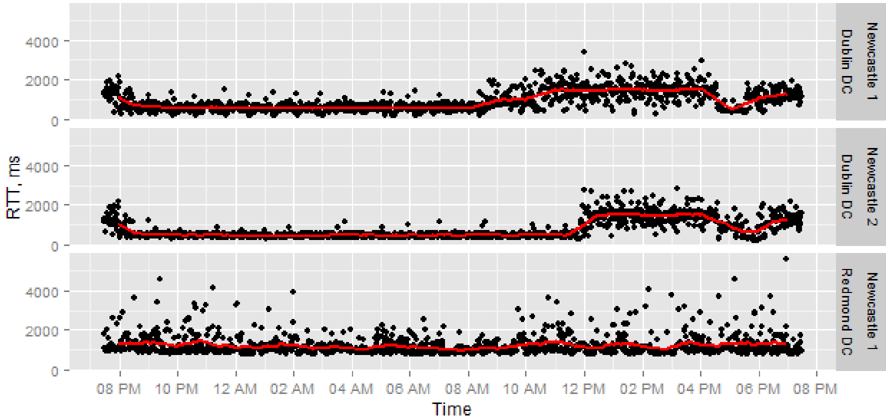

Fig. 11. Newcastle Java client RTT 1 minute moving average (by client and data center)

4.8 Analysis in the faulty domain

Similarly, we should look deeper into the structure of the 'faulty' domain to understand the mechanisms that together result in the 'composite behavior' (linear relationship between RTT and RT). We specifically look for distinct rare events that are suppressed or did not have direct effect on RT, but potentially may have.

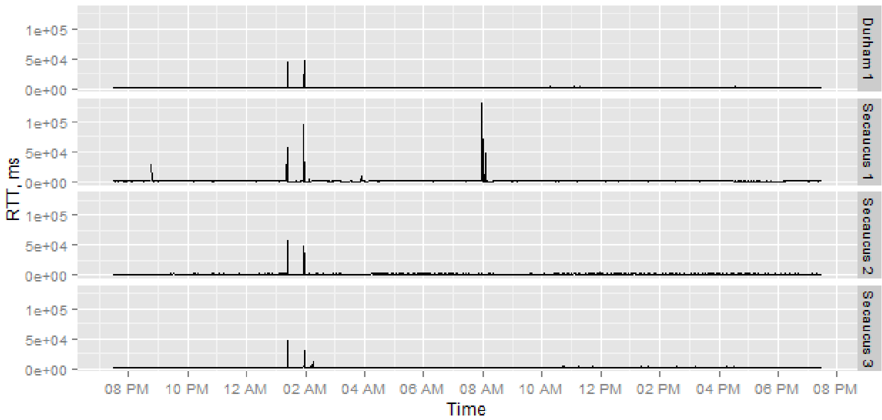

Fig. 12. Secaucus and Durham using Dublin DC RTT time series: correlated hard faults

As in the faulty domain there are only 60 observations clustering into 10-12 groups along the timeline, these places can be examined one by one. Although one cannot recognize any pattern readily – only seemingly random transient errors – the ratio of high RTT delays increased between 1 AM and 2 AM. With barcharts-based linked highlighting it can be detected that clients in Secaucus sending their requests into the Dublin DC produce the majority of these high delays. On the other hand one can observe that not only the 4 clients of Secaucus but also the client in Durham shows similar high delays at the same time (in the same minute), twice within a 35-minute interval (see Fig. 12). We can conclude that the communication error – in contrast with our initial first thoughts – is specific for the Dublin DC, not the Secaucus client group.

5 Interpretation: from EDA to resilience

The EDA process provided valuable and rather deep insight into the observed performability characteristics of the experimental environment. How can we utilize this knowledge for *'as is' resilience characterization* and during *resilience mechanism design*? At a first glance, there is not much to say about as-is resilience. We are aware of only network latency failures and we do not know whether these stemmed from network level faults/overload or configuration changes. Tolerating these faults is a resilience issue only in the sense that the benchmarking 'system' was certainly not equipped to deal with them. The same holds for such true changes as moving the client to a different location.

However, the discovered relationships can advise us about resilience against *certain classes of cloud failures*. Due to virtual machine interferences and scheduling policies, virtual machines running in an IaaS cloud may lose temporarily a significant portion of their 'steady state' CPU time allowance (see e.g. [21]). Based on the EDA findings, we can begin to formulate the expected effects of (unexpected) server CPU slowdowns on RT. Fig. 13 shows the histogram of our observations in the lower part of the normal category – with the RPT component of the RT sum being scaled from the original values to ten times greater. It becomes clear that 'all else being equal' (i.e. no network faults are present), the setup is "resilient" as even a ~5 times 'slowdown' still keeps us in our original 'normal' RT domain. However, 10 times slowdowns evidently lead us out from this category, the maximal frequency shifting to ~8000 ms. This way, we have effectively identified a significant amount of slack in the system against disturbances we have no direct observations for at design time based on the available data. Additionally, the discovery of the time dependent nature of RTT for certain data centers means that at a finer granularity of modeling, the slack, and thus the inherent resilience becomes time-dependent as well. Note that the scope of this paper only allows for presenting these core ideas; future work will investigate the necessary (nonparametric) statistical tooling.

Our findings have important ramifications for design for resilience as well. On the one hand, we have an empirical sample on network time faults that can serve as a basis for their classic dependability modeling. However, maybe more importantly from the point of view of resilience, we have found evidence for 1) single machine deployment problems leading to consistently 'problematic' RTT; and 2) empirical proof for RTT faults characteristic to using a specific data center. Consequently, resilience techniques for systems using the measured resources should be aware of these fault modes – especially in a cloud setting, where the potentially highly dynamic deployment configuration of clients as well as servers can be interpreted as system-internal change.

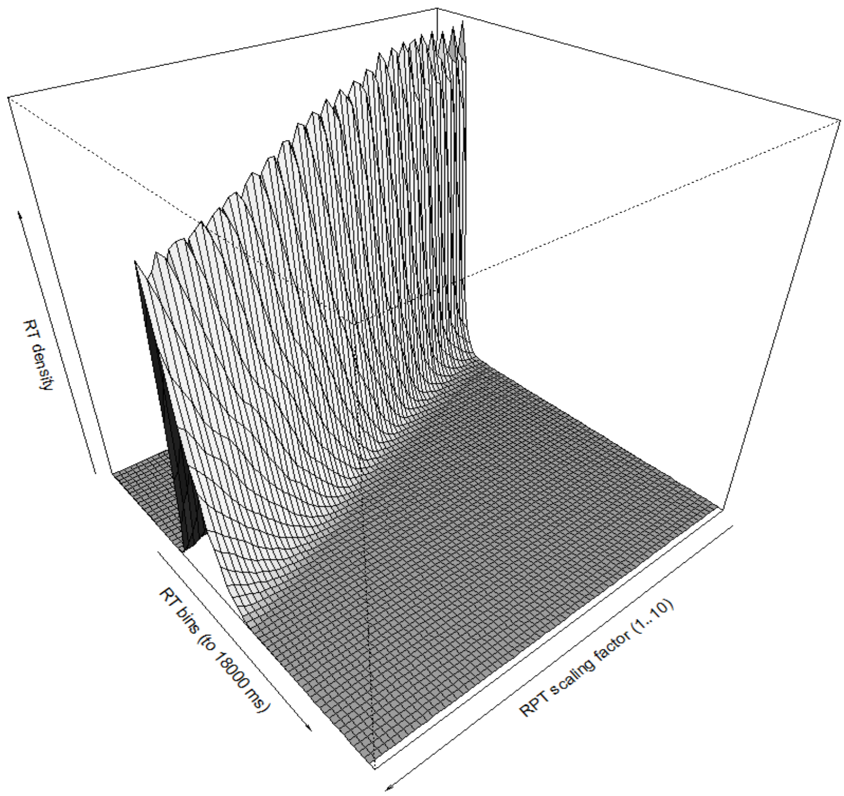

Fig. 13. Histograms of existing ‘ok’ RT observations with RPT component scaling 1..10

Acknowledgements. This comparative study would have been impossible without the help of the excellent dependability analysis paper [11]. We would like to express our gratitude to the authors, especially to Alexander Romanovsky and Anatoly Gorbenko, for the access to their experimental data and the fruitful discussions.

References

1. Holling, C.S.: Resilience and stability of ecological systems. Annual review of ecology and systematics. 4, 1–23 (1973).

2. Avizienis, A., Laprie, J.-C., Randell, B., Landwehr, C.: Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. on Dependable and Secure Computing.. 1, 11–33 (2004).
3. Trivedi, K.S., Kim, D.S., Ghosh, R.: Resilience in computer systems and networks. Proc. of Int. Conf. on Computer-Aided Design - ICCAD '09. p. 74. ACM Press, New York, USA (2009).
4. Laprie, J.-C.: From dependability to resilience. 38th Annual IEEE/IFIP International Conference On Dependable Systems and Networks. Fast abstracts. (2008).
5. Sterbenz, J.P.G., Hutchison, D., Çetinkaya, E.K., Jabbar, A., Rohrer, J.P., Schöller, M., Smith, P.: Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. Computer Networks. 54, 1245–1265 (2010).
6. Kanoun, K., Spainhover, L. (editors): Dependability benchmarking for computer systems. John Wiley & Sons (2008).
7. Ghosh, R., Longo, F., Naik, V.K., Trivedi, K.S.: Quantifying Resiliency of IaaS Cloud. 2010 29th IEEE Symposium on Reliable Distributed Systems. pp. 343–347. IEEE (2010).
8. Almeida, R., Vieira, M.: Benchmarking the resilience of self-adaptive software systems. Proc. of the 6th international symposium on Software engineering for adaptive and self-managing systems - SEAMS'11. p. 190. ACM Press, New York, New York, USA (2011).
9. Behrens, J.T.: Principles and procedures of exploratory data analysis. Psychological Methods. 2, 131–160 (1997).
10. Tukey, J.: We need both exploratory and confirmatory. The American Statistician. 34, 23–25 (1980).
11. Gorbenko, A., Kharchenko, V., Mamutov, S., Tarasyuk, O., Romanovsky, A.: Exploring Uncertainty of Delays as a Factor in End-to-End Cloud Response Time. 2012 Ninth European Dependable Computing Conference. pp. 185–190. IEEE (2012).
12. Theus, M., Urbanek, S.: Interactive graphics for data analysis: principles and examples. CRC Press (2011).
13. Cook, D., Buja, A., Lee, E., Wickham, H.: Grand Tours, Projection Pursuit Guided Tours and Manual Controls. Handbook of Data Visualization. pp. 295–314. Springer (2008).
14. McGill, R., Tukey, J., Larsen, W.: Variations of box plots. The American Statistician. 32, 12–16 (1978).
15. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2013).
16. Inselberg, A.: Parallel Coordinates: Visual Multidimensional Geometry and Its Applications. Springer Science+Business Media, New York, USA (2009).
17. Cook, D., Swayne, D.F.: Interactive and Dynamic Graphics for Data Analysis: With Examples Using R and GGobi. Springer (2007).
18. Urbanek, S., Theus, M.: iPlots: high interaction graphics for R. Proceedings of the 3rd International Workshop on Distributed Statistical Computing (2003).
19. Urbanek, S.: iPlots eXtreme: next-generation interactive graphics design and implementation of modern interactive graphics. Computational Statistics. 26, 381–393 (2011).
20. Wickham, H.: ggplot2: Elegant Graphics for Data Analysis. Springer (2010).
21. Kocsis, I., Pataricza, A., Micskei, Z., Kövi, A., Kocsis, Zs.: Analytics of Resource Transients in Cloud Based Applications. To appear in: International Journal of Cloud Computing, 2 (2/3), 191-212 (2013).