Identification of Dependability Models from Large Observation Sets

Ágnes Salánki
Budapest University of Technology and Economics
Department of Measurement and Information Systems
H-1117 Magyar tudósok krt. 2, Budapest, Hungary
salanki@mit.bme.hu

András Pataricza
Budapest University of Technology and Economics
Department of Measurement and Information Systems
H-1117 Magyar tudósok krt. 2, Budapest, Hungary
pataric@mit.bme.hu

***Abstract*—Dependability analysis of complex infrastructures necessitates a sophisticated statistical analysis in order to estimate the causes of failures. The paper presents a combination of visual analytics and algorithmic methods specialized to rare event analysis. The main outcomes of the proposed analysis process are a statistical model of the system and the estimation of the main factors to be used in supervisory systems for an early detection of potential failures.**

I. Introduction

By design, failures in high-availability systems are extremely rare. Resilience of such systems can be analyzed only empirically, because the diversity of the building components and the complexity of their interaction jointly exclude efficient usage of analytic methods. System models typically lack an exact statistical projection, thus, nonparametric analytics methods are needed.

Visual exploratory data analysis (EDA) performed by domain experts is a promising approach to support empirical construction of dependability model. However, application of analysis techniques can be challenging because of rare occurrence of fault events (*outliers, anomalies*). Nowadays, a variety of tools exist for detection of rare events in a multi-dimensional space. These automated methods without exact parameter tuning may retrieve spurious relationships instead of semantically correct results.

Visualization-based and algorithmic methods have the potential to complement each other [2], [3]. Visual analytics is intuitive, exploits the semantic interpretation skills of an expert typically without a demand for deep statistical knowledge. However, visual analytics can only provide rough estimations of clusters. On the other hand, computation-centric methods are able to select best-fitting boundaries to data clusters, find the best configuration, build models and test hypotheses.

Three types of integration are defined in [3]: a) *computationally enhanced visualization* techniques (e.g. artificial selection of „interesting" scatterplots); b) *visually enhanced mining* (e.g. presentation of hierarchical clustering with dendograms) and c) *fully integrated visualization and mining* (e.g. interactive decision tree building in [2]), where the current approach also belongs to as well.

The goal of this paper is to present an approach for computationally enhanced visual detection and characterization of rare events. We find the synergistic integration of the semantic interpretation capability of an expert used in visual analytics with the precise processing of algorithmic methods very promising.

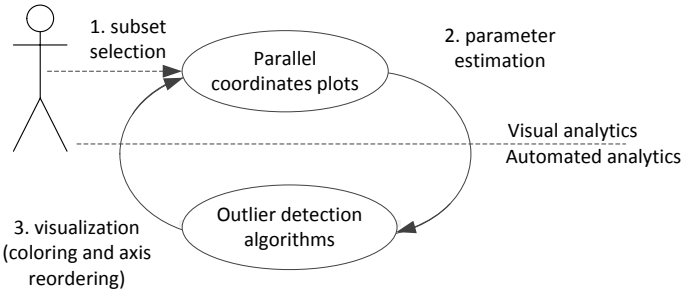

Fig. 1. The visual outlier detection approach

Parallel coordinates plots (PCPs) [8] were selected for visualization as the most sufficient means to present multi-dimensional data sets.

The user can select an initial set of candidate outliers on the PCP interactively, thereafter a dedicated algorithm refines the parametrization of the DB [10] detection algorithm. Thus, the processing of a large data set can be reduced to semi-supervised learning by this outlier-focused relevance filtering. Results from the automated detection are back-annotated on the PCP to allow the user to refine the query.

II. Related work

A. Outlier detection

Outliers are data points deviating significantly from other observations. Accordingly, it is deduced that they were generated potentially by a different mechanism than the bulk of data [7]. Outliers are categorized into three main groups [4]: 1) *point outliers;* 2) *collective outliers* and 3) *contextual outliers*. Point outliers are observations individually differing from the rest of the data set, e.g. an extreme peak in CPU usage. Contextual outliers are observations of a context-dependent deviation, e.g., a large peak in CPU usage may be suspicious only at night. A collective outlier group includes a subset of observations, collectively deviating from the bulk of points, even if the individual observations themselves may not be outliers, e.g., continuously large CPU usage in a system where only short peaks are allowed.

The two main groups of outlier detection algorithms are *distance-based* and *density-based* methods. Distance-based methods assume that outliers are far from the bulk of points or the center of the entire data set. This family of methods contains e.g. DB, which marks a node as outlier if it has only few neighbors around it in a hypersphere. Density-based approaches mark those data points as outliers whose density (e.g., average number of neighbors) is highly different of their *local* neighborhood (LOF or NNDB [12]).

B. Multi-dimensional visualization techniques

General-purpose multi-dimensional visualization techniques, including PCPs, scatterplot matrices or tableplots [13], all focus on a specific aspect of the observation set. Other methods, like MDS (*Multi-Dimensional Scaling*), PCA (*Principal Component Analysis*) plots or biplots project the multi-dimensional space into a subspace of a lower dimension and simultaneously highlight some special characteristics of the observation set. [9] presents specific transparency functions to control the visualization of outliers in PCP.

Interestingness measures and labels can be used as guidance of visual analysis by a preference attribute defined by the user. These metrics and annotations are analyzed simultaneously during the analysis as support in the subsequent steps for search space reduction or result ranking [5] [6].

III. Analysis workflow in IT infrastructure management

This section summarizes the possible steps of a failure-triggered analysis workflow (Fig. 2). Its primary goal is root cause analysis while identification of trends or a fine diagnosis is only of secondary importance.

Usually the assurance of high availability, especially in our focus area of cloud computing, is a main engineering objective. Accordingly, coarse-grained diagnosis is sufficient to take mitigation actions. A detailed analysis serves only the fine-tuning and extension of automated monitoring and control mechanisms of the supervisory system.

Failure-triggered analysis of IT systems consists of two steps: 1) *analysis of phenomenological observations* (failure analysis); 2) *analysis of the monitored infrastructure* (root cause analysis).

The goal of failure analysis is to detect any sudden parasitic degradations in the high-level QoS metrics to avoid permanent SLA violation, assuming that our system works typically correctly and errors happen only very rarely. The corresponding analytics methodology here is outlier detection.

We assume that the number of high-level metrics is around 10 in our domain (e.g. [1]); root cause analysis still has to deal with a possibly large number of low level infrastructure metrics. There are several million observations, according to the typical sampling frequency and error latency, this may represent a day with time-triggered data acquisition methods applied in modern large systems.

Root cause analysis serves identification and elimination of faults in the system causing the observed QoS degradation, e.g. to find the resource acting as a bottleneck. Thereafter,

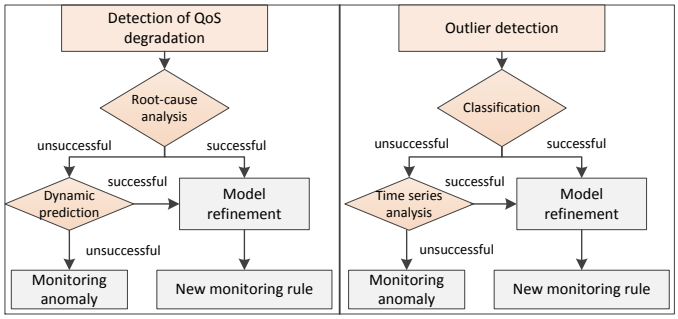

Fig. 2. Analysis workflow of a typical IT environment

new monitoring rules can be added to the supervisory system to detect, diagnose and mitigate such faults. The analytics method here is *classification*, being capable of exploring and localizing the data set subspaces corresponding to erroneous system states.

Root cause analysis sometimes fails to identify the fault. The main reason for this problem is that no complete a priori fault model can be constructed for such large-scale and complex systems. If the failure is caused by an unobserved factor (e.g., some change in the environment), then frequently no significant difference in resource utilization, operational domain, etc. can be found prohibiting a proper diagnosis. The prediction of the QoS degradation based solely on detecting changes in high-level observations before the rare event is still possible at the phenomenological level. *Dynamic time series analysis* methods are the most promising in this case.

IV. Interactive outlier detection with PCPs

Outlier detection is usually performed in the first phase of the analysis process. The domain expert faces two challenges at this point: 1) *very efficient multi-dimensional visualization* is needed requiring no exact knowledge about relevant factors in the operational domain under investigation; 2) *parametrization of automated outlier detection algorithms* is a hard problem, their indirect estimate is sometimes not intuitive.

Our approach supports an efficient offline outlier detection and characterization along the following workflow (Fig 1):

Step 1 the user selects an initial subset of candidate outliers on the PCP plot and estimates the number of outliers in the data set: this number can be influenced by the domain, the time window, etc.;

Step 2 the DB algorithm is parameterized based on the previous user interaction; thereafter the algorithm determines the entire set of outlier candidates;

Step 3 the outliers (and simultaneously the dense clusters as point of reference) are visualized, where colors reflect the decision parameters of the algorithm and axes are reordered to highlight the relevant dimensions separating the outliers from normal points.

Individual steps are detailed in the following subsections.

A. Parameter estimation

An object x in a dataset T is a $DB(p, D)$ outlier if at least a fraction p of the objects in T lies from x greater than distance

D [10]. This is one of the basic, actively used concepts in the outlier detection community. Despite the simplicity of the intuitive definition, it is cumbersome to estimate the parameters p and D in the early exploration phase of the analytics process. However, the domain expert could have an exact vision about the proportion of outliers in the entire data set and, after exploring it, which observations seem most suspicious. Our approach presents a heuristic of estimation of p and D, based on the aforementioned two input parameters.

The parameters of the algorithms (Algorithm 1 and Algorithm 2) contain the data set T, the user-defined candidate subset O, the estimated proportion of outliers in the data set ε and the number of maximal allowed iterations n.

Algorithm 1 computes a (p, D) value pair based on the maximum proportion of neighbors to be considered i. First, the empirical distribution function is computed from the distances around each outlier candidate o (line 1). Secondly, a minimal distance D is defined, assuming that an outlier candidate has at most i neighbors in distance D (line 2). The final estimate for parameter p is computed from this distance to ensure that the selected candidate subset satisfies the DB outlier conditions (line 3). Finally, the size of the final outlier subset is determined (line 4).

Algorithm 1 Parameter estimation for a specific initial i quantile

Input: T: entire data set, O: outlier candidate subset, $\imath$: maximum proportion of neighbors to be considered;
Output: D_i: border distance of close neighbors,
p_i: border proportion of close neighbors,
nn_i: vector containing the number of close neighbors
1: h_x: distance from other points for $\forall x \in T$
2: $D_i := \min_{o \in O} \mathrm{ecdf}_{h_o}^{-1}(i)$
3: $p_i := \max_{o \in O} \mathrm{ecdf}_{h_o}(D)$
4: $nn_i := |x : x \in T, \mathrm{ecdf}_{h_x}(D) \leq p|$
5: **return** p_i, D_i, nn_i

Algorithm 2 computes possible (p, D) values in discrete points in $[0, \varepsilon]$, considering the maximal number of iterations n. The estimated values of p and D are calculated by Algorithm 1 in every iteration, which results in different outlier sets. No convergence can be assumed without a priori knowledge about distributions, thus, we calculate different outlier candidate sets at discrete points. At the end, an ideal parametrization is chosen to minimize the difference between the expected and the calculated outlier set size.

B. Visualization of outliers

This phase of analysis aims at visualization of the input data set; special coloring and axis reordering were chosen to reflect the DB results and highlight the characteristics of outliers.

Coloring: Any visual characteristics of a parallel coordinates line could be used to highlight outliers, e.g. transparency, color, size, texture, etc. Coloring was chosen in our visualization, because this is a non-additive characteristics: dense clusters do not amplify each others' value, as transparency, texture or size do.

The color of each observation is based on its number of neighbors. The breakpoints on the color scale partitions

Algorithm 2 Parameter estimation for DB algorithm

Input: T: entire data set, ε: estimated proportion of outliers in the data set, O: outlier candidate subset, n: maximum number of iterations
Output: p_i: border proportion of close neighbors, D_i: border distance of close neighbors
1: **for** $i := 0$ **to** ε **do**
2: D_i: border distance of close neighbors applying Algorithm 1
3: p_i: border proportion of close neighbors applying Algorithm 1
4: nn_i: vector containing the number of close neighbors applying Algorithm 1
5: $error_i := |nn_i - \varepsilon|$
6: $i+ = \varepsilon/n$
7: **end for**
8: $init := \arg\min_i error_i$
9: **return** p_{init}, D_{init}

the observations into three main groups: outliers (low values) are colored with red, normal points (high values) are blue, intermediate points are yellow. This is another supporting feature to highlight the separation quality of regions.

Axis order: Relevant dimensions to be monitored can be selected differently from one operational domain to another one. Thus, an appropriate order of coordinates axes should be computed to determine the best dimensions separating outliers from normal points. For this, the selected detection algorithm (e.g., DB) is run on every one-dimensional subset. Dimensions are ranked based on the correlation between their value and the final outlier score. The strength of the interdependence of two dimensions is evaluated by metrics, which can be used in the case of a non-linear relation.

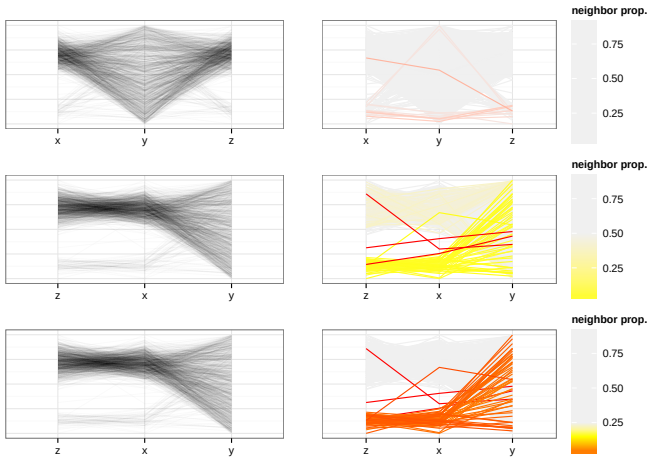

Fig. 3. An initial user interface and results from different initial subsets

Results from two different subset selections are presented in Fig. 3. The typical cluster visualizations with low transparency (left) and outlier parallel coordinates plot (right) are present side-by-side, where the former serves as point of the reference and so supports outlier characterization. The first panel (top) presents the initial user interface, the other two (middle and bottom) show results of different subset selections. Although the set of outliers are different, identical axis reordering was performed.

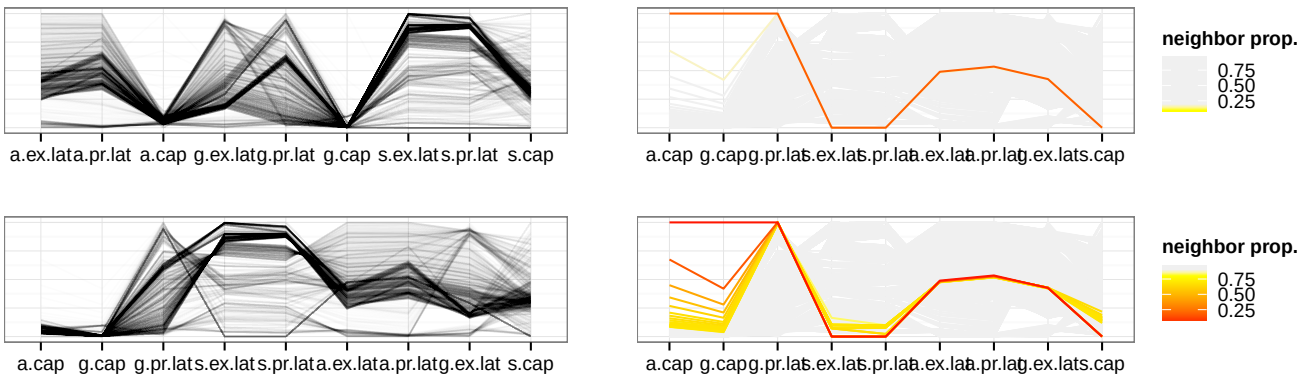

Fig. 4. Outlier detection in QoS metrics of a stream processing environment

V. Case study: outlier detection in a stream processing environment

Our outlier detection approach is demonstrated on a data set collected in the Storm [1] stream processing environment, while running an application computing the average delay of US flights. Data tuples in the experimental environment are being transferred between three types of nodes: *aggregator*, the *gatherer* and the *sweeper*, being responsible for aggregation by date, the computation on the aggregated data and the time handling.

Three metrics are stored about every component: *process latency*, *execution latency* and *capacity*. The first two reflect the time spent between start and end of processing of the incoming tuple, the time spent between start of processing and the formal verification of it, while the capacity is a derived metric in the range of $[0, 1]$, reflecting how strongly the actual component can be considered as a bottleneck in the system. The environment was monitored by a 10 sec sampling time; the data set contained around 1800 observations in 9 dimensions in this small-scale pilot.

The initial user interface and the final result after selection of a tight subset is presented on Fig. 4. Examining the clustering and outlier plots simultaneously, we can notice that a large subset of outliers and intermediate observations show a typical behavior in every dimension but the capacity dimensions of the aggregator and gatherer node (*a.cap* and *g.cap*). These were chosen as two first axes: their high values show indeed a strange behavior. Thus, we can conclude that the most relevant factors to be observed in the future are the capacity of aggregator and the gatherer.

VI. Conclusion and future work

The paper presented an interactive outlier detection and characterization approach for computing enhanced exploration of outliers.

Although PCP is an ideal visualization technique in terms of scalability and information compression, the method is unable to efficiently give to the users an idea about distances between points (contrary, e.g. to the scatterplot matrices). Coloring can partly eliminate this problem. Initial visual analysis approaches in our domain (e.g., [11]) suggest that operational domains are frequently analyzed separately. Thus, use of distance based algorithms seems very promising in our domain.

Extensions of current user interface is planned to support visualization of object relationships typical in our domain: absolute and relative amount of resource metrics on different levels, possible interference between virtual machines, etc.

References

[1] Apache storm project. https://storm.apache.org/.

[2] Mihael Ankerst, Martin Ester, and Hans-Peter Kriegel. Towards an effective cooperation of the user and the computer for classification. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, pages 179–188, New York, New York, USA, 2000. ACM Press.

[3] Enrico Bertini and Denis Lalanne. Surveying the complementary role of automatic data analysis and visualization in knowledge discovery. In *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery: Integrating Automated Analysis with Interactive Exploration*, pages 12–20, 2009.

[4] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.

[5] Tijl De Bie. Subjective interestingness in exploratory data mining. In *Advances in Intelligent Data Analysis XII*, pages 19–31. Springer, 2013.

[6] Liqiang Geng and Howard J Hamilton. Interestingness measures for data mining. *ACM Computing Surveys*, 38(3):9–es, September 2006.

[7] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[8] Alfred Inselberg and Bernard Dimsdale. Parallel coordinates. In *Human-Machine Interactive Systems*, pages 199–233. Springer, 1991.

[9] Jimmy Johansson, Patric Ljung, Mikael Jern, and Matthew Cooper. Revealing structure within clustered parallel coordinates displays. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 125–132. IEEE, 2005.

[10] Edwin M Knorr, Raymond T Ng, and Vladimir Tucakov. Distance-based outliers: algorithms and applications. *The VLDB JournalThe International Journal on Very Large Data Bases*, 8(3-4):237–253, 2000.

[11] András Pataricza, Imre Kocsis, Ágnes Salánki, and László Gönczy. Empirical assessment of resilience. In *Software Engineering for Resilient Systems*, pages 1–16. Springer, 2013.

[12] András Pataricza and Ágnes Salánki. Detection of rare events. In Péter Antal, editor, *Intelligens adatelemzés*, pages 28–45. Typotex Kft., Budapest, 2014.

[13] Martijn Tennekes, Edwin de Jonge, and PJH Daas. Visual profiling of large statistical datasets. In *New Techniques and Technologies for Statistics conference, Brussels, Belgium*, 2011.