# Fingerprinting the Datacenter: Automated Classification of Performance Crises

Peter Bodík
University of California, Berkeley

Moises Goldszmidt
Microsoft Research

Armando Fox
University of California, Berkeley

Hans Andersen
Microsoft

## ABSTRACT

When a performance crisis occurs in a datacenter, rapid recovery requires quickly recognizing whether a similar incident occurred before, in which case a known remedy may apply, or whether the problem is new, in which case new troubleshooting is necessary. To address this issue we propose a new and efficient representation of the datacenter's state, a *fingerprint*, that scales linearly with the number of performance metrics considered and it is not affected by the number of machines. These fingerprints are generated online and then used as unique identifiers of the different types of performance crises so that we can effectively recognize previous occurrences and retrieve repair actions. We evaluate our approach on a production datacenter with hundreds of machines running a 24x7 enterprise-class user-facing application, verifying each identification result with the operators of the datacenter and trouble-shooting tickets. Our approach has $80\%$ identification accuracy in the operations-online setting with time to detection below 10 minutes (our operators stated that even 30 minutes into the crises is desirable), and offline identification on the order of high $90\%$. To the best of our knowledge this is the first time such an approach has been applied to a large-scale production installation with such rigorous verification. We compare our approach and show it is superior to various alternatives to the construction of a fingerprint including an adaptation to the datacenter setting of the signatures work in [6].

## 1. INTRODUCTION

Important user-facing applications are increasingly being run in large datacenters, from search and email to e-commerce and enterprise applications. 24x7 availability is key to the competitiveness of such applications, but as others have pointed out [14], occasional hardware and software failures are inevitable in such installations, making it prudent to develop strategies for rapid recovery when failures occur. One way to speed recovery from a performance crisis is to quickly recognize whether a similar crisis has occurred before, in which case a known remedy may apply. Automating this recognition and retrieval avoids wasting time repeating previous manual analysis, captures the knowledge of previous analysis for future personnel, and avoids escalation of crises that can subsequently be addressed by tier-0 and tier-1 operators.

We present a methodology for automating recognition and identification of known crises, and indicating when a crisis is new (previously unseen), for applications comprising hundreds of machines in a datacenter. Our methodology makes the following contributions:

1. Representation: A representation of the performance state of the datacenter, the *fingerprint*, which a) can be efficiently computed for an installation of thousands of machines and whose size scales linearly with the number of performance metrics being collected per machine (and is independent of the number of machines), and b) captures the most relevant metrics for identifying the different classes of performance crises.
2. Recognition and retrieval: A method for comparing fingerprints and effectively retrieving and identifying known performance crises and instances of new crises with high degree of accuracy.
3. Validation methodology: A rigorous methodology for evaluating the approach itself, and validation using labeled data from a production datacenter.

Specifically, during a four-month period, 19 performance crises were diagnosed and labeled by datacenter operators, ranging from configuration problems to unexpected workloads to backups caused by a connection to another datacenter. These problems required different repair actions, hence the importance of rapid identification. We show that for identifying and discriminating different crises, fingerprints outperform not only simpler "baseline" methods, but also specifically outperform the method of "signatures" proposed by the authors of [6].

Our evaluation of the approach on real data covers various settings isolating the impact of the various parameters and sensitivity to various threshold decisions. We evaluated several alternatives to the fingerprint construc-

tion and our implementation of an approach similar to that of [6] assuming ideal conditions for the model management and model selection. In our experiments we repeatedly simulated permutations of the actual sequence of crises in order to ensure that our results were not due to one lucky series of events. We also use standard techniques of cross-validation, ROC curves, and online analysis to estimate the different evaluation metrics. Our results clearly establish that:

1. Our approach discriminates almost optimally between crises and almost optimally labels the different crises correctly if given perfect future knowledge of the whole dataset we used;
2. Our approach achieves identification accuracy over $80\%$ when used in an online manner, i.e. when the crisis data is presented incrementally, one crisis at a time, as would occur in a real setting;
3. The approach is superior to the most closely related work [6], to approaches based on using all metrics available as the fingerprint, and to approaches that would rely on the key performance indicators used by the human operators to *detect* the crisis;
4. The approach clearly quantifies the trade-offs among false positives, accuracy of identification, and time to identification (some crises take longer than an hour, so having an identification even 30 minutes into the crisis would be useful).

To the best of our knowledge, this is the first time such an approach has been applied to a large-scale production installation with rigorous verification using hand-labeled data.

## 2. RELATED WORK

As early as 2003, the authors of [16] proposed the use of compute-intensive modeling techniques to recognize and diagnose operational problems. Since then, much work has focused on detecting and recognizing operational problems using machine learning [5, 6, 21, 3, 20, 15, 8], identifying unusual or noteworthy sequences of events that might be indicators of unexpected behavior [17, 4], and laying out general methodological challenges in using computers to diagnose computer problems [9, 7]. Other methods rely on manually instrumenting the system [2] or creating an extensive library of all possible faults and their consequences [18].

By far the work closest in spirit to our own is the "signatures" approach to identifying and retrieving the essential system state corresponding to previously-seen crises [6]. The authors propose a methodology for constructing "signatures" of server performance problems by first using machine learning techniques to identify the performance metrics most relevant to a particular crisis; second, using the induced models for online identification; and third, relying on similarity search to recognize a previously recorded instance of a particular incident. They showed their approach to be successful in a small transactional system on a handful of performance problems.

We view our methodology as a direct descendant of the "signatures" approach but with several important improvements. First, our fingerprint representation size scales linearly, rather than exponentially, with the number of metrics considered. Second, the signatures approach maintains multiple models, one for each crisis seen; when a new crisis occurs, the signatures approach computes a particular fitness score (the Brier score) to decide which of the existing model(s) are likely to provide the best identification of the current crisis, and then use those model(s) to construct the signature of (and therefore identify) the crisis. In contrast, we only use our models to determine which metrics are relevant for modeling the crises; we do not use the models themselves to "match" each crisis. This simplification allows our approach to avoid two related sources of potential error and their corresponding free parameters. First, we need no policies to maintain and ensure the validity of multiple models. Second, since we don't maintain multiple models, we need no fitness score to determine which model(s) to apply to a particular problem and no need for a method to combine the output of multiple models.

The authors of [3] observe that the use of regularized logistic regression [19] as a classifier results in a metric selection process that is more robust to noise than other techniques, and specifically more robust than the naïve Bayes classifier used in the signatures approach. In particular, their technique avoids the wrap-around search for the relevant features for each model as was done in the signatures work. However, as the methodology of [3] otherwise follows the signatures methodology closely, it must deal with the problems of model management and online selection. Also, their representation of performance state relies on histograms of signatures, so the size of the representation can grow exponentially with the number of metrics being recorded. We first followed their methodology, but concluded that the proposed signatures have little discriminative power.

## 3. PROBLEM AND APPROACH

A typical datacenter-scale user-facing application runs simultaneously on hundreds or thousands of machines in a datacenter. In order to detect performance problems and perform postmortem analysis after such problems, several *performance metrics* are usually collected on each machine and logged to online or nearline storage. Since large collections of servers execute the same code, under normal load balancing conditions the values of these metrics should come from the same distribution;

as we will show, we use this intuition to capture the state of each metric and identify unusual behavior.

Usually each metric is sampled once per aggregation epoch—typically a few minutes—and the sampled values may represent a simple aggregate over the aggregation epoch, e.g. the mean. Some metrics correspond to low-level hardware measurements, others may correspond to OS-level, application-level or runtime-level metrics, such as the current size of the object heap or number of threads waiting in the run queue. Wide variation exists in what can be collected and at what granularity; packages such as HP OpenView [1], Ganglia [13], and others provide off-the-shelf starting points.

A subset of the collected metrics may be *key performance indicators* (KPI's) whose values form part of the definition of a contractual service-level agreement (SLA) for the application. An SLA typically specifies a threshold value for each KPI and the minimum fraction of machines that have to satisfy the requirement over a particular time interval. For example, an SLA might require that the end-to-end interactive response time be below a certain threshold value for 99.9% of all requests in any 15-minute interval.

A performance *crisis* is defined as a prolonged violation of one or more specified SLA's. Recovery from the crisis involves taking the necessary actions to return the system to an SLA-compliant state. If the operators can recognize that the crisis is of a previously-seen type, a known remedy can be applied, reducing overall recovery time. Conversely, if the operators can quickly determine that the crisis does not correspond to any previously seen incident, they can immediately focus on diagnosis and resolution steps, and record the result in case this crisis recurs in the future.

Our goal is to automate the *crisis identification* process by concisely summarizing and capturing the subset of the collected metrics that do the best job of discriminating among different crises. We next describe our process for doing this, called *fingerprinting* the datacenter, and how we define a similarity metric between two fingerprints to identify recurring problems.

## 3.1 Approach: Quantiles as Fingerprints

A fingerprint is a vector representing the performance state of the system that will uniquely identify a performance crisis. There are four steps to our fingerprint-based recognition and identification technique.

1. We summarize the values of each performance metric across the whole datacenter by computing the *quantiles* of the measured values (such as the median of CPU utilization on all servers). Unlike statistics such as the mean, quantiles are more robust to outliers in the distribution of the metric values. As we discuss in Section 3.2, this summarization scales well with the size of the datacenter.
2. We track each metric quantile over time and identify its *hot*, *cold*, and *normal* regimes—values that are abnormally high, abnormally low, and normal, respectively. We discuss this step and the choice of hot and cold thresholds in Section 3.3. This gives us a *summary vector* vector containing one element per quantile per tracked metric, indicating whether the value of that quantile is cold, normal, or hot during that epoch.
3. We identify the *relevant metrics* whose behavior distinguishes normal performance from the performance crises defined by the SLA's. The metric selection process is described in Section 3.4. This subset of the summary vector for a given epoch is the *epoch fingerprint*.
4. Since most crises span multiple epochs, we show how to combine consecutive epoch fingerprints into a *crisis fingerprint*. Finally, we define a *similarity metric* that is used to compare fingerprints from two crises and determine whether they correspond to the same underlying problem. These two steps are described in Section 3.5.

## 3.2 Tracking Metric Quantiles

It is customary to sample the values of hundreds of performance metrics for each machine during each epoch.We can view this as taking samples of random variables whose distributions are unknown. Rather than trying to model each metric's behavior by fitting a parametric distribution to the metric's observed values over time, we model the metric using quantiles of the observed empirical cumulative distribution over an epoch. Recall that the $p$'th quantile of a distribution is the value $x$ such that if we sample the distribution and observe value $x'$, then $\Pr(x < x') = p$; for example, the 50th quantile is also known as the median. Empirically, we estimate the $p$'th quantile by ordering the measured values of some metric and selecting the $\lceil Np/100 \rceil$'th value. Quantiles are considered robust statistics; it is well known, for example, that the median is less susceptible to outliers than the mean. There is considerable literature on algorithms for estimating quantiles with bounded error using online sampling and/or streams [10]. Thus, as the datacenter grows, the quantiles can be computed efficiently, although in our case study involving several hundred machines, we were able to compute them exactly.

For our experiments and throughout the paper we tracked three quantiles for each metric: 25%, 50% (i.e. the median), and 95%.[1]

[1]In Sections 5 and 6.1 we discuss our empirical choices for free parameters such as the number of quantiles and quantile thresholds, and we examine the sensitivity of our approach to

### 3.3 Hot and Cold Quantile Thresholds

A fingerprint tells us which metrics' quantiles were abnormally high (hot) or abnormally low (cold) during a particular epoch. (We reiterate that we are referring to the values of the *quantiles*, not the raw metric values.) We studied three methods for setting the thresholds used to make the hot/cold determination. In this section we describe the method we actually used, which yields fingerprints that are less sensitive to spurious variations in the quantiles and is also the simplest to compute. In the Appendix we describe two other methods we tried and rejected.

We consider a quantile to be *normal* if its value is between the 2nd and 98th percentile of the values observed during some contiguous period of length $T$ in which *no crisis was indicated*. In other words, we expect that $4\%$ of the time, the quantiles of a metric will be outside this normal range even though no crisis is in progress. Intuitively, we set $4\%$ as the acceptable level of false positives. To compute hot and cold thresholds for metric $m$:

1. For a period $T$, select only the epochs where the key performance indicators were normal (i.e., no crisis condition is indicated).
2. For each of the three quantiles of $m$ (25%, 50%, 95%):
   (a) Compute the 98th percentile value and set this as the *hot* threshold.
   (b) Compute the 2nd percentile and set this as the *cold* threshold.

It is easy to see how we can turn this into an online procedure that for a window of size $T$ updates the thresholds. For the experiments in the paper we evaluated $T$ at $\{240, 120, 60, 30\}$ days.

We can now build a summary vector for one epoch: it is a vector of $3M$ elements ($M$ is the number of metrics being tracked). Each group of 3 elements corresponds to the 25th, 50th and 95th quantiles for a particular metric over that epoch. An element's value is $-1$ if the quantile's value is below the *cold* threshold during the epoch, $+1$ if the quantile's value is above the *hot* threshold, and $0$ otherwise (see Figure 1). We next show how to convert this vector into a fingerprint by selecting only the subset of *relevant metrics* for discriminating among and identifying crises.

### 3.4 Relevant Metrics in Fingerprints

As we will show in our experiments (Sections 5.1.1 and 5.1.2), achieving robust discrimination and identification accuracy requires selecting a subset of the metrics, namely the *relevant* metrics for building the fingerprints. We determine which metrics are relevant in two steps. We first select metrics that correlate well with the occurrence of each individual crisis by performing *feature selection and classification* on data surrounding each crisis, similar to [6]. Second, we use metrics most frequently selected in the first step as the relevant metrics used for building all fingerprints.

Feature selection and classification is a technique from statistical machine learning that first induces a function between a set of features (the metrics in our case) and a class (crisis or no crisis) and concurrently tries to find a small subset of the available features that yields an accurate function. Let $X_{m,t}$ be the vector of metrics collected on machine $m$ at time $t$ and $Y_{m,t}$ be 0 if the performance of $m$ was normal and 1 if it was anomalous, as specified by the performance SLA's. A classifier is a function that predicts the performance state of a machine $Y$ given the collected metrics $X$ as input. The feature selection component picks a subset of $X$ that still renders this prediction accurate. In our approach we use logistic regression with L1 regularization [19] as the statistical machine learning method. The idea behind regularized logistic regression is to augment the maximization in the statistical fitting of the logistic regression with a constraint on the parameters of the model. The constraint is that the sum of the parameters be bounded. This in turn forces irrelevant parameters to go to zero, effectively performing feature selection. It has been (empirically) shown in various settings that this method is effective even in cases where the number of samples is comparable to the number of parameters in the original model [11], as is the case in our scenario in which the number of possible features (over 100 per server for several hundred servers) exceeds the number of classification samples.

We remark that while this approach is inspired by [6], the use of logistic regression (vs. a Bayes classifier) has been shown to result in more robust feature selection [3].[2] This result is consistent with the general direction in the statistical community. Other important differences between our approach and that of [6] are detailed in Section 8; part of our evaluation is a direct comparison to that work, and we will show that these differences result in significant improvement in identification accuracy.

For the second step, in which we select the most frequent metrics from the first step, we use a population of 20 crises which changes as time progresses and new crises are detected. The initial 20 crises were not labeled by the operators of the datacenter. Note that this is irrelevant for the method as it only needs to know that there is a crisis and this is detected automatically through the SLA violation. Even though we did experiment with the total number of metrics necessary for accurate crises iden-

most of these free parameters.

[2]We confirmed this empirically, but we omit the results as they provide no additional insights beyond the results reported in [3].

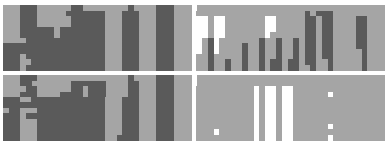


**Figure 1**: Counterclockwise from top left, fingerprints of crises B, B, D and C from Table 1. Each row is an epoch and each column represents the state of a particular metric quantile, with white, gray, black corresponding to cold, normal, hot $(-1, 0, +1)$ respectively in the fingerprint.

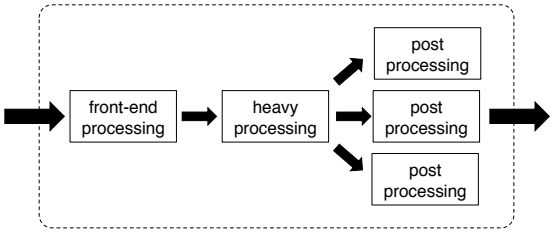



**Figure 2**: Request flow in each machine in the datacenter under study.

tification (see Section 6.1) we did not experiment with changing the number of crises involved which was left constant at 20.

The summary vector is converted into an *epoch fingerprint* by selecting only the relevant metrics.

## 3.5 Identifying Similar Crises

As most crises last for more than one epoch, comparing crises requires comparing several consecutive epochs of fingerprints. We obtained good results with the following simple method, but we note that this step is orthogonal to the rest of our approach and can be modified without altering any of the other computations.

Given a set of epochs during a crisis, we create a crisis fingerprint by averaging the fingerprint vectors across these epochs. Figure 1 shows fingerprints of four crises; each row represents an epoch, each column represents a metric quantile, and white, gray, and black represent the values $-1$, $0$, and $1$, respectively, in the corresponding fingerprint. The three left-most columns of the top-right crisis in the figure would be summarized as $\{\frac{-7}{12}, \frac{-4}{12}, \frac{6}{12}\}$; there are $12$ epochs in the crisis and the column sums are $-7$, $-4$, and $6$. Notice that the quantiles often don't move in the same direction—for example, see the left-most three columns in the top-right crisis—and this is important for identification. If we tracked fewer than 3 quantiles, we wouldn't capture this behavior.

To compute a similarity score between two crises, we compute the L2 distance between the corresponding crisis fingerprints. Two crises are considered identical if the L2 distance between their fingerprint summaries is less than a specified *identification threshold*. In our experiments, we compare the efficacy of setting this threshold based on perfect future knowledge (offline estimation) vs. estimating the threshold online as crises are actually observed.

# 4. EVALUATION

## 4.1 System Under Study

We validate our results on a commercial datacenter running a $24\times7$ enterprise-class user-facing application. It is one of four datacenters worldwide running this application, each containing hundreds of machines, serving several thousand enterprise customers, and processing a few billion transactions per day.[3] Most machines execute the same application, as depicted in Figure 2. The incoming workload is processed on the machine in three stages: light processing in the front-end, core of the execution in the second stage, followed by some back-end processing. The processed requests are then distributed either to the clients or to another datacenter for archival and further processing. We have no visibility to the clients or to machines in the other datacenters.

For each server, we sample about 100 metrics each averaged over a period of 15 minutes. The 15-minute averaging window is established practice in this datacenter, and we had no choice on this matter; similarly, we have no access to any other performance counters or to information allowing us to reconstruct the actual path of each job through the server or processing time at each stage. The metrics include counts of alerts set up by the operators, queue lengths, latencies on intermediate processing steps, summaries of CPU utilization, and various application-specific metrics.

The operators of the site designate three *key performance indicators* (KPI's) corresponding to the average processing time in the front end, the second stage, and one of the post-processing stages. Each KPI has an associated service-level agreement (SLA) threshold determined as a matter of business policy. A performance crisis is declared when $10\%$ of the machines in the datacenter violate *any* KPI SLA's. This definition is set by the operators and we did not tamper with it.

We use a set of 19 performance crises that occurred during a four month period between January and April 2008. These crises were labeled by the application operators according to the determined underlying cause. Somewhat descriptive labels of the crises and their frequency appear in Table 1. We also use additional collected metrics and 20 unlabeled crises that occurred between September and December 2007. We don't use the unlabeled crises to evaluate the identification accuracy of our fingerprints, but we do use the data from that period when simulating online deployment of our method.

## 4.2 Evaluation Methodology

[3]The exact numbers are considered confidential by the company that operates the datacenter.

| ID | # of instances | label |
|---|---|---|
| A | 2 | overloaded front-end |
| B | 9 | overloaded back-end |
| C | 1 | database configuration error |
| D | 1 | configuration error 1 |
| E | 1 | configuration error 2 |
| F | 1 | performance issue |
| G | 1 | middle-tier issue |
| H | 1 | request routing error |
| I | 1 | whole DC turned off and on |
| J | 1 | workload spike |

**Table 1**: List of identified performance crises.

We compare and contrast four different methods for identification of performance crises:

- *Fingerprints* refers to our proposed method as described in Section 3.1.
- *Signatures* refers to the method proposed in [6]. We had to make some changes to the signatures method to adapt it to the setting of a large-scale datacenter and to remove several design choices for comparison, always making such choices favorable to the performance of the signatures approach. These changes are detailed in Appendix 8.
- *Fingerprints (all metrics)* refers to a version of our proposed fingerprinting method that uses *all* the collected metrics to create a fingerprint instead of a smaller set of relevant metrics. The intention is to quantify the level of noise introduced by having irrelevant metrics participate in the identification of the problem.
- *KPI* refers to fingerprint method based just on the three KPI's being currently tracked in the datacenter. For each KPI, the fingerprint contains the number of machines in the datacenter that are violating the performance SLA specified for that KPI.

## 4.3 Evaluation Criteria

**Discrimination.** We first report on the *discriminative power* of our fingerprints: their ability to determine, independently of applying a specific label to a crisis, whether two crises are the same or distinct. Two crises are classified as identical if the distance between their fingerprints is less than a specified *identification threshold*, otherwise they are considered different. To quantify the discriminate power of each method, we check whether each pair of the 19 labeled crises is correctly classified as "identical" or "different" and, as is customary, represent these results in a form of a ROC curve [12]. This curve depicts the tradeoff between the rate of false positives (incorrectly classifying two different crises as identical) and the rate of detection (correctly classifying two identical crises) for different values of the identification threshold. We will use $\alpha$, the false positive rate, as the single parameter to be adjusted in order to select the optimal threshold (also standard practice). We compare ROC curves by computing the area under the curve (AUC); higher AUC implies better discriminative power.

The discriminative power essentially removes issues regarding the determination of clustering boundaries for the different performance problems and clearly establishes the basic capabilities of each of the representation methods for capturing the differences between the different crises.

**Identification stability.** Because all crises in our dataset last longer than a single 15-minute epoch, we perform identification during each epoch. During discussions with the operators of this enterprise application we confirmed that identification information is useful even up to one hour into the crisis. We thus perform five identifications per crisis; starting when the crisis is detected and continuing for the four subsequent epochs. In each epoch, identification emits either the label of a known crisis, or the label `x` for "don't know." An identification sequence is considered *stable* if it consists of zero or more consecutive `x`'s followed by zero or more consecutive *identical* labels. For example, if A and B are labels of known crises, the sequences `xxAAA`, `BBBBB`, and `xxxxx` are all stable, whereas `xxAxA`. `xxAAB`, `AAAAB` are all unstable.

**Identification accuracy.** If an identification sequence is stable, we can evaluate whether it is accurate (i.e. whether the label emitted was correct), and also measure *time to identification* as the time of the first epoch after the start of the crisis during which the correct label was emitted. We subdivide the definition of *identification accuracy* into two categories. When identifying a crisis that had already been seen and diagnosed before, *known accuracy* is the probability of correctly labeling the crisis using a stable identification sequence. If a sequence is unstable, the accuracy of each individual label is irrelevant and we do not count it as a successful identification. When identifying a previously unseen crisis, *unknown accuracy* is the probability if correctly labeling that crisis as "unknown" in all five identification epochs.

We remark that when compared to the evaluation criteria for identification used in [6], our criteria are much more stringent and more realistic. In [6], an identification was considered successful as long as the actual crisis was among the $k$ most similar crises according to a distance metric. In contrast, in our definition of identification, we differentiate between assigning a label of a particular crisis and declaring the crisis unknown. We also count the identification as accurate if it was stable and the assigned label was exactly correct. The reason we can claim that this is more realistic is that the operators will execute a sequence of actions depending on the label assigned to a

crisis. If the crisis is known, but our technique is unable to identify it correctly, there is a cost related to the missed opportunity of executing the repair actions, and possibly re-doing the diagnosis. On the other hand, if an unknown crisis is mislabeled, then the repair actions will be incorrect with possibly severe consequences. We demonstrate in Section 5.3 that this tradeoff can be controlled using the $\alpha$ parameter introduced above.

## 4.4 Setting

Performing online identification using fingerprints depends on estimating the following three sets of parameters based on past data:

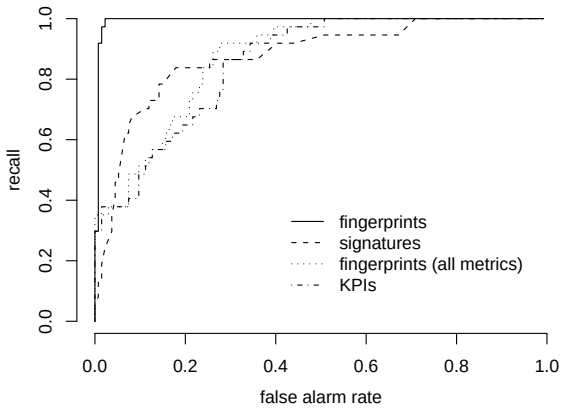
1. **Relevant metrics**. We perform feature selection to narrow down the metrics that are used for crisis identification using fingerprints.
2. **Hot and cold quantile thresholds**. These thresholds change over time due to changes in workload and performance of the application and we periodically recompute them based on new data.
3. **Identification threshold**. A new crisis is labeled as "unknown" if the distance of its fingerprint to fingerprints of all the past crises is greater than a specified identification threshold. Otherwise, the new crisis is labeled as identical to the crisis that is closest. The identification threshold is recomputed after each new crisis.

In order to characterize the effects of these parameters on accuracy and time of identification, we examine three settings: offline, quasi-online, and online.

In the offline setting, we assume perfect future knowledge and set the values of the parameters using data from the whole period under study. We pick the relevant metrics using feature selection on the 19 crises, compute the metric quantile thresholds using the four months of data surrounding the 19 crises, and set the identification threshold using the ROC curve based on pairwise distances of all the crises (see Section 5.3 for details). This is essentially the best-case scenario for all the evaluated techniques and errors are, by and large, the product of numerical instability etc. inherent in the computations and it therefore establishes in a precise sense an empirical optimal performance for the method.

In the quasi-online setting, the relevant metrics and metric quantile thresholds are computed in an online fashion over a moving window of data, while the identification threshold is still estimated assuming full knowledge of all the labeled crises. The decrease of accuracy of approximately 15% is therefore the price we pay for not having the luxury of looking at the whole period for selecting the relevant metrics and setting the hot/cold thresholds.

Finally, in the online setting we estimate all three parameters in an online fashion and thus realistically simulate the actual deployment of our technique as a tool for crisis identification. As reported in Section 5.3, the decrease in accuracy is not significant, only about 3%. This is encouraging as it establishes that the approach is resilient to the online estimation of the identification threshold and therefore the fingerprints are robust, and that the loss in accuracy is mainly due to variations on the conditions of operation (changes in relevant metrics and hot/cold thresholds).




| type of fingerprint | AUC |
|---|---|
| fingerprints | 0.994 |
| fingerprints with all metrics | 0.873 |
| KPIs | 0.854 |
| signatures | 0.876 |

**Figure 3**: Distance ROC curves and the area under the curves (AUC) for different types of fingerprints.

# 5. RESULTS

In the following sections, we describe the details of selecting the relevant metrics, computing the hot and cold quantile thresholds, and estimating the identification threshold for each of the offline, quasi-online, and online settings and summarize the results.

## 5.1 Offline setting

### *5.1.1 Discrimination results*

Recall that by discrimination we mean how accurately can each technique classify two crises as identical or different. We consider two crises identical if the distance between their fingerprints is less than the identification threshold $T$. As in information retrieval, for a given choice of $T$, we can compute *recall*, the fraction of crisis pairs of the same type that are actually identified as being the same, and *false alarm rate* (or false positive rate), the fraction of pairs in which the two crises are distinct but the difference between their fingerprints is less than $T$.

For each evaluated fingerprint technique, we quantify

the discriminative power using the area under the corresponding *distance ROC* curve. The distance ROC shows the tradeoff between recall and false positive rate as a function of a threshold $T$. An optimal distance-ROC curve would have a recall value of 1.0 for all values of the false positive rate (i.e. zero false positives with perfect recall) and would pass through the point $(0, 1)$, indicating that there exists a distance threshold $T$ that perfectly separates pairs of identical crises from distinct ones. An optimal ROC curve would have area under the curve (AUC) of 1.0, and in general, larger values of AUC imply better discrimination of a given set of fingerprints. To generate the ROC, we compute the false alarm rate and recall for all feasible values of the threshold parameter $T$ and connect the resulting points.

The graph in Figure 3 shows the distance ROC curves for the four fingerprinting techniques along with the corresponding AUC. Our fingerprinting method clearly dominates the remaining three and achieves almost perfect separation of the distinct crises.

#### 5.1.2 Offline Identification Results

The goal of the identification experiments in the offline setting is to quantify the identification accuracy of the proposed techniques in the best-case scenario when assuming the perfect future knowledge of the collected metrics and labeled crises. To pick the relevant metrics, we first use feature selection (as described in Section 3.4) to select top ten metrics for each of the 19 labeled crises and then select the 15 most frequent ones as the relevant metrics.

In the offline setting, the hot and cold thresholds are computed as the $2^{\text{nd}}$ and the $98^{\text{th}}$ quantile of each metric across the four months for data.

The selection of the identification threshold $T$ is based on the distance ROC curve of all the labeled crises and is parameterized by the false alarm rate $\alpha$ between 0 and 1, which controls the trade-off between the known and the unknown accuracy. In particular, $T$ is set to the largest threshold such that the corresponding false alarm rate, the x-axis in the ROC plot, is less than $\alpha$. In the offline case, we compute the threshold based on a distance ROC curve generated from all labeled crises.

In the identification experiments, we start with an initial set of five labeled crises, and perform identification on the remaining 14 crises. The initial set of crises contains two randomly selected crises of type "B", one of type "A", and two other crises (see Table 1). If the new crisis A that we want to identify is *known* (i.e., the initial set of crises contains a crisis identical to A), the identification is accurate if it's *stable* and we assign the correct crisis label. If the crisis A is *unknown*, the identification is accurate if we don't assign any label during the five identification epochs. For each identification experiment we compute the *known accuracy*, ratio of accurate known identifications and total number of known identifications, *unknown accuracy*, ratio of accurate unknown identifications and total number of unknown identifications. We also compute the *time of identification* – the average time when a known crisis was correctly identified relative to its start. For each of the evaluated techniques we executed five runs with different initial set of crises and report the average of all the evaluation metrics in Figure 4. In the offline setting we do not update the initial set of crises by adding the newly identified crises.

#### 5.1.3 Summary of offline results

The offline identification results are important because they establish the optimal performance of each technique and serve as a baseline for comparison with results in the online setting.

Based on the ROC curves in Figure 3 we conclude that our fingerprint approach clearly dominates the other approaches and achieves almost perfect AUC score of $1.0$. It also achieves very high known accuracy of $97.5\%$ and unknown accuracy of $93.3\%$[4] in the identification experiments. This confirms, that the proposed fingerprints indeed capture the important state of the datacenter when assuming perfect future knowledge about the metrics and the crises when computing the relevant metrics and the quantile and identification thresholds. This concise representation is achieved by using quantiles to summarize the metric values across the whole datacenter and by further reducing the number of metrics using feature selection.

The fingerprints that use all the collected metrics achieve identification accuracy of only approximately 50%. This shows that the feature selection using regularized logistic regression is a crucial step in generating the fingerprints. On the other hand, using only the three KPI metrics does not provide enough discrimination power as this method also achieves identification accuracy of only about 55%.

Although signature approach performs worse than our fingerprint method, it performs better than the baselines, and it achieves accuracies of $75\%$ and $80\%$. This provides evidence that a) there is value in selecting a subset of metrics even though the final output is not exactly a fingerprint, and b) our implementation of the signatures is valid.

### 5.2 Quasi-Online Setting

In a real-world scenario, the identification of crises would be performed on-line; the relevant metrics, the quantile thresholds, and the identification threshold would be determined from the past crises and the historic values of the metrics. To isolate the effects of the online

[4]We report accuracies for $\alpha$ where the two accuracy curves cross or where their values are closest together.

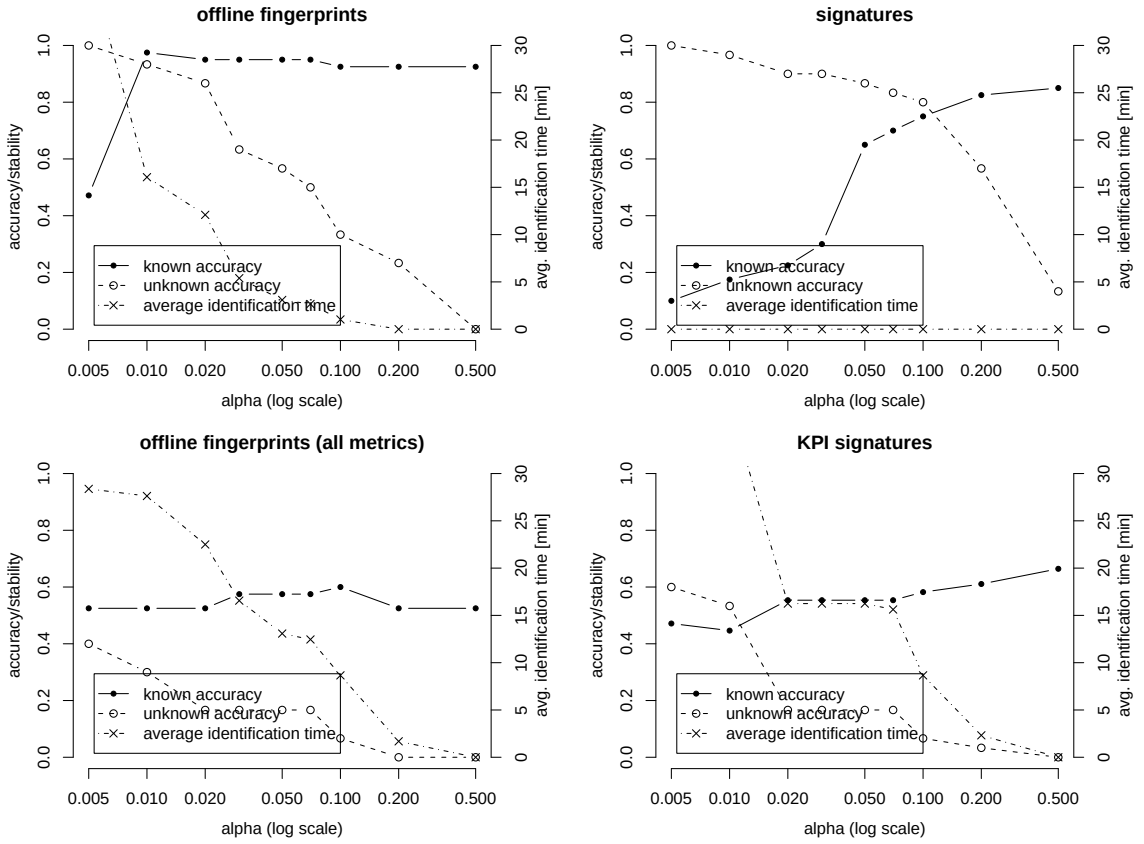



**Figure 4**: Known accuracy, unknown accuracy, and time of identification for different crisis signatures: offline fingerprints (top-left), the SOSP'05 signatures (top-right), KPIs (bottom-left), and offline fingerprint using all the available metrics (bottom-right).

computation of relevant metrics and quantile thresholds on the identification accuracy, in the quasi-online setting we perform these steps in an online fashion, while still estimating the identification threshold offline. The identification of crises is performed as they are presented, i.e. identification of the first crisis cannot exploit any information about later crises.

When identifying a new crisis $A$ at time $t$, we select the relevant metrics based on the 20 crises that occurred before crisis $A$. We first perform feature selection as described in Section 3.4 to pick the top ten metrics for each crisis and then use the more frequent 30 metrics as the relevant metrics. Experiments using different number of total metrics are described in the discussion section. The cold and hot metric thresholds are computed as the $2^{\text{nd}}$ and $98^{\text{th}}$ percentiles of data over a moving window of 240 days that excludes all time intervals that were anomalous according to the specified SLAs. For each past crisis we keep track of the raw values of all collected metric quantiles and we recompute fingerprints of *all the past crises* based on the newly computed metrics and thresholds.

For the quasi-online setting, we evaluate crisis identification using the chronological order of the crises, as well as using 20 random permutations. In each experiment, we assume that we have already seen the first two crises and start the identification process from the third crisis. For each new crisis, we perform the following steps: a) we compute the relevant metrics and metric thresholds, b) recompute the fingerprints of past crises, c) compute the value of the identification threshold assuming that we have seen *all 19 crises* and thus use the full ROC curve, and d) perform identification using the distances between crises and the identification threshold. This scenario thus differs from a real-world deployment of our method only in the usage of the optimal identification threshold. In the experiments with a random permutation of the crises, we present the crises to the algorithm in a new order, but for each crisis we use fingerprints that were computed in the chronological order.

In Figure 5, we present identification results when using 30 relevant metrics and length of the moving window of 240 days. Our fingerprint method achieves known and unknown accuracy of 85%. Thus, the online selection of relevant metrics and quantile thresholds caused a decrease of accuracy of approximately 15%. As we demonstrate in the following section, switching to the fully online setting will not significantly decrease the identification accuracy.

## 5.3 Online setting

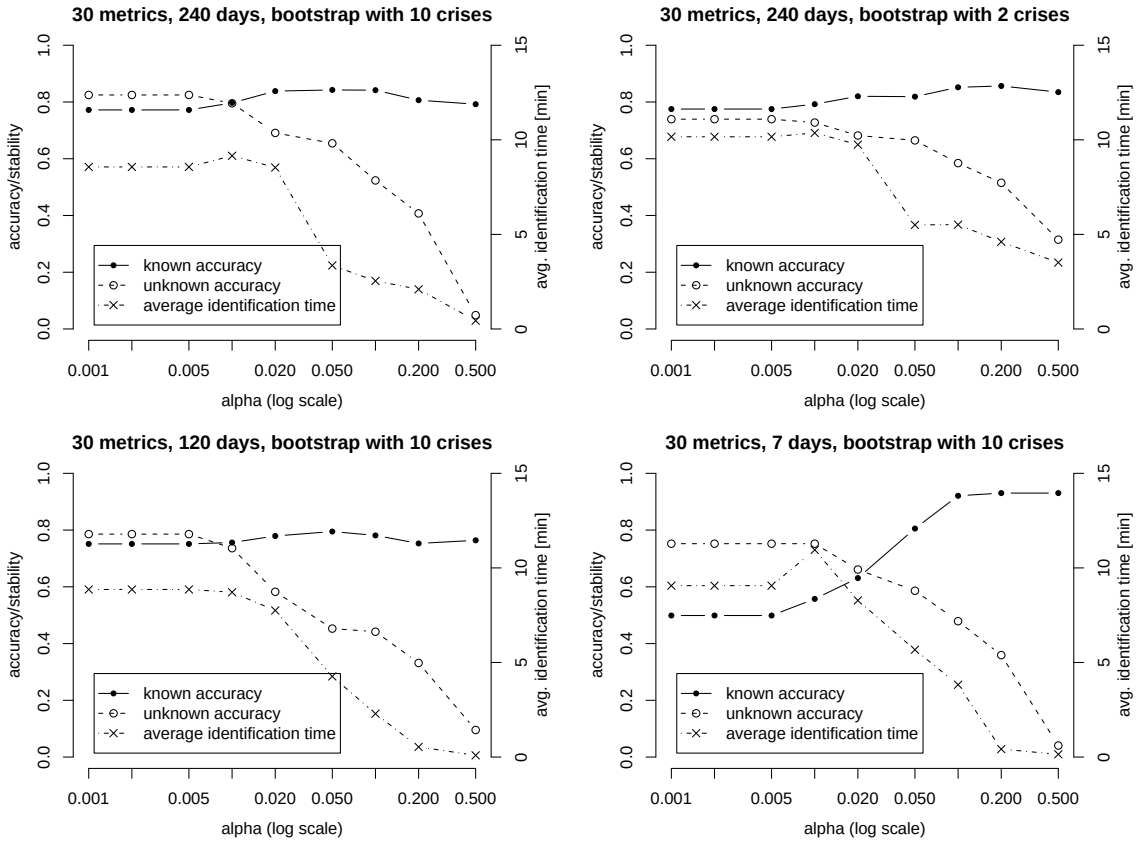



**Figure 6**: Known accuracy, unknown accuracy, and time of identification results when using 30 metrics in fingerprints, 240 days of moving window, and bootstrapping with 10 labeled crises and 2 labeled crises (top). Bottom shows the results for moving window of size 120 and 7 days.

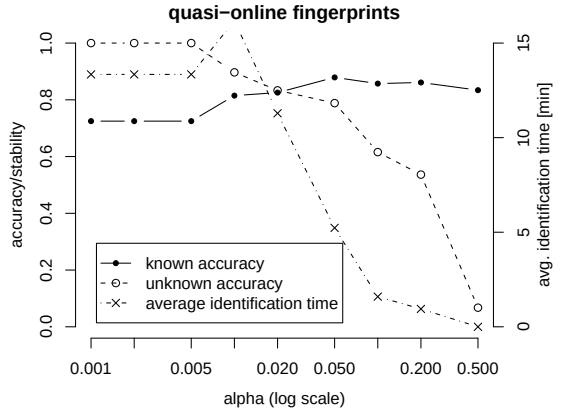



**Figure 5**: Identification results in the quasi-online setting.

In the fully online setting we update the relevant metrics and metric thresholds as described in the previous section, but also update the identification threshold in an online fashion. To compute the identification threshold $T$, we employ a method very similar to the offline case. We use the distance ROC curve computed based on the past crises and the specified $\alpha$ to estimate $T$ as described in Section 5.1.2. However, if the number of past crises is very small, or we observed crises of only the same type or of only different types, we use the following set of intuitive rules to set $T$. First, if we have observed only crises of the same type, we set $T = \max_d(1+\alpha)$, where $\max_d$ is the maximum distance between the crises of the same type; $\alpha$ controls the extra *buffer* we add to $\max_d$. New crises of the same type should thus be correctly classified. Second, if we have observed only crises of different types, we set $T = \min_d(1-\alpha)$, where $\min_d$ is the minimum distance between crises of different types. Finally, if we have observed crises of both the same and different types, but the ROC curve is optimal ($\max_d < \min_d$), any value of $T$ between $\max_d$ and $\min_d$ will result in no expected false alarms. In this case we set $T = \max_d + \alpha(\min_d - \max_d)$.

In the online setting, the identification threshold is first computed based on the ROC curve of the initial set of labeled crises and then recomputed after each new crisis. As we observe more crises, the estimate of the threshold should be more accurate, thus resulting in a more accurate identification. To test this hypothesis, we performed two sets of experiments; in the first one, we seed the identification process with only two crises, just like in

| setting | known acc. | unknown acc. |
|---|---|---|
| offline | 98% | 93% |
| quasi online | 83% | 83% |
| online, bootstrap w/ 10 | 80% | 80% |
| online, bootstrap w/ 2 | 78% | 74% |

**Table 2**: Summary of the results for different settings.

the quasi-online setting, while in the second one we seed the identification with ten crises. Because when starting with ten crises, we only identify the remaining nine, to keep the total number of identifications similar, we perform 41 runs in this setup. The rest of the methodology for the online identification experiments is identical to the quasi-online setting.

The results for 30 relevant metrics and different lengths of the moving window are presented in Figure 6. When starting with two labeled crises, we achieve known and unknown accuracy of $78\%$ and $74\%$. Initializing the identification using ten crises increases both accuracies to $80\%$.

## 5.4 Summary of results

We summarize the main results with respect to the accuracy in the identification of the crises by our methods and the "cost" of performing this identification in an online manner in the real like scenario. Table 2 summarizes these accuracies. As explained above the accuracy is a duplex containing the accuracy of identifying known crises and unknown ones. The operating point of the methods with respect to these measures will depend on the cost of missing a known crises (and repeating the diagnosis again), and the cost of mislabeling a known crises (and performing a possibly erroneous repair action). In our summary we selected the operating point where the measures are closer. In the offline case, our methods achieve very high accuracy. These measures reflect the inherent error in the method which for the accuracy of known crises is just $2\%$. The online adjustment of the quantile threshold plus the online selection of the relevant metrics to be included in the fingerprint, decreases the accuracy by over $15$ points. This decrease is to be expected as otherwise crises identification would be a realitve easy problem. It is reasuring to find out that the online selection of the optimal trheshold is not that costly (only 3 points in accuracy) when the process is bootstraped by the use of 10 labeled metrics. This provides with an $80\%$ accuracy for our method. The last line presents the accuracy when the whole process of identification starts with only 2 known crises.

# 6. DISCUSSION

## 6.1 Sensitivity to free parameters

As the results of Section 5.1 show, when we use all available metrics rather than a subset of relevant metrics as the fingerprint, both the discriminative power and the identification accuracy of the approach are reduced. In the experiments in Section 5 we used 30 metrics for the fingerprint; in additional experiments we observed a steady decrease in the accuracy of the identification as we tried fingerprints of $20$, $10$, and $5$ metrics, holding the period $T$ constant ($240$ days). We also experimented with the length of the moving window $T$ that adjusts the quantile thresholds, observing decreasing accuracy for $T = 120$, $T = 30$, and $T = 7$ days, observing a steady decrease in accuracy especially at operating points where the tradeoff between the known and unknown accuracy is minimal (i.e., both lines intersect – see Figure 6). When both the number of metrics and window length were varied at the same time, we observed that, in general, for small window sizes, fingerprints with fewer metrics have higher accuracy than fingerprints with more metrics. For example, when the moving window was set at 7 days, fingerprints with less than 30 metrics displayed better accuracy than those with 30. This is unsurprising: as the size of the fingerprint decreases, the fingerprint adjusts more nimbly to rapid changes in values, which comes as a consequence of reducing $T$; but as $T$ increases and a greater variety of crises is seen, additional information is needed in the fingerprint to capture the differences among them. We observe that overall the best accuracy was obtained using 30 metrics and a window of $240$ days.

When comparing two crises, we first compute the crisis fingerprints by averaging the corresponding epoch fingerprints. In all the experiments in Section 5, we average across epochs $-30$ minutes, $\ldots$, $60$ minutes, relative to the start of the crisis. Here we compare the discriminative power of fingerprint summaries computed over ranges $\langle t_0, t_1 \rangle$, where $t_0 \in \{-60, \ldots, 0\}$ and $t_1 \in \{t_0 + 15, \ldots, 150\}$. Figure 7 shows that ranges that start at least $30$ minutes before the beginning of the crisis, quickly achieve high levels of discrimination.

## 6.2 Hot/Cold threshold settings

In our experiments, we compute the hot and cold thresholds using the $2^{\text{nd}}$ and the $98^{\text{th}}$ percentile over the past values of each metric. The discriminative power (area under the ROC curve) using these values was 0.99 (1.0 being the maximum). When we varied these values to $\langle 1^{\text{nd}}, 99^{\text{th}} \rangle$, $\langle 5^{\text{nd}}, 95^{\text{th}} \rangle$, and $\langle 10^{\text{nd}}, 90^{\text{th}} \rangle$, the discriminative power decreased to 0.96 or less.

In addition to setting hot/cold quantile thresholds based on the $\langle 2^{\text{nd}}, 98^{\text{th}} \rangle$ percentiles of quantile values, we explored two other methods. The first was based on fitting a non-parametric time series to the quantiles, estimating the standard deviations of the prediction in the time series and then setting the hot and cold thresholds three standard deviations from the mean. The second method

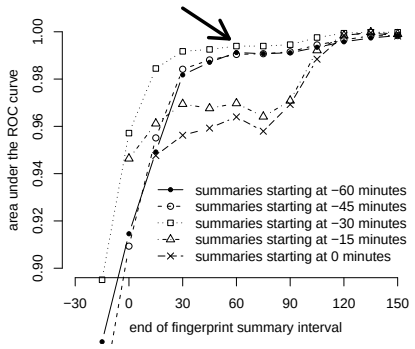



**Figure 7**: Area under the ROC curve (discriminative power) of fingerprints when summarized over different ranges. Each line on the graph represents ranges that start at the same epoch, while the x-axis represents the end of the range. The arrow points to AUC corresponding to the range $\langle -30\text{minutes}, +60\text{minutes}\rangle$ used in all our experiments.

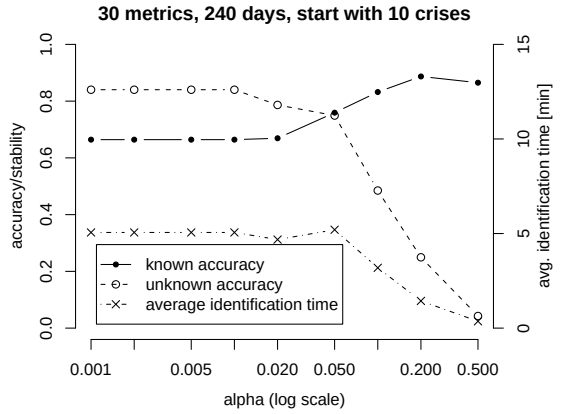



**Figure 8**: Identification results when the fingerprint's hot/cold thresholds are not updated.

fits thresholds by separating various regions on the quantile values, and then correlating to violations on the KPIs. Both displayed discriminative power of $0.95$ or below, inferior to the $0.99$ achieved by simply setting fixed percentiles, so we did not consider either alternative further.

## 6.3 Updating the fingerprints

Currently in our method we store the raw values of the metric quantiles for all the collected metrics, and we update the $\{0, -1, 1\}$ values of the fingerprints according to the updated hot and cold thresholds as they change over time. Assuming total of 100 metrics, 3 quantiles per metric, fingerprint summary range of 7 epochs, and 4 bytes per value, we need to store $100 \times 3 \times 7 \times 4 = 8400$ bytes per crisis (which is not terribly expensive). To find out whether this bookkeeping is necessary we experimented with a method where fingerprints for past crises are computed using metric thresholds that were estimated when those crises occurred. The result is a decrease in the accuracy for identification. The operating point where the know and unknown accuracies cross is $75\%$ or 5 points below our best result (see Figure 8).

## 6.4 Time to Identification

Our results make clear that, as is typical for information retrieval problems, there is a tradeoff between the accuracy of identifying known crises and the accuracy of correctly labeling a new crisis as unknown. However, there is also a third element in the tradeoff, namely the time at which the method makes a decision regarding the identity of the crisis. The dependency among these three elements is made clear in Figures 4, 5, and 6: drawing a vertical line through the three curves captures the tradeoff for any given value of the parameter $\alpha$, which corresponds to the false positive rate for discrimination (Section 4.3). We expect that graphs such as these will be crucial for setting the optimal "operating point" of our method for a given datacenter, as the choice depends on the priorities and associated costs of correct identification of known as well as unknown crises vs. time to identification. We note that our method is able to make the identification with $80\%$ accuracy within 10 minutes of crisis detection, even in a fully-online setting where the parameters and thresholds have not been learned from all data in advance.

# 7. FUTURE WORK

We are currently extending this work on four directions. The first one involves finding in the fingerprints early signs of the crises so that they can be forecasted. Our initial results were encouraging especially in regards to forecasting crises of type "B" (see Table 1). The second one is on modeling the complete evolution of the crisis. As operators apply actions to resolve the problem, they would like to be able to monitor progress and also have knowledge on how long would it take for the crises to be resolved. These models would enable optimal decision making with regards to repair actions, and would enable the road to full automation. The third direction focuses on using information about the crises labels in the task of selecting the relevant metrics for building fingerprints. We have started this investigation by posing this as a classification problem in terms of discrimination and using regularized logistic regression to select the metrics. We expect to have results about this methodology in the near future. Finally, the fingerprints are agnostic on whether the metrics collected belong to a single application, to the hardware platform, etc. In this datacenter we were dealing with one application and mostly with metrics related to this application. We are currently collecting data on different applications and different datacenters and designing the appropriate extensions to the fingerprints.

# 8. CONCLUSIONS

In this paper we propose a datacenter fingerprint that compactly captures the important datacenter state by sum-

marizing metrics across all the machines using quantiles, discretizing the value of each quantile into just three states (hot, cold, or normal), and selecting only metrics relevant for discriminating between crises. As our offline results indicate, in terms of both the discriminative power and the accuracy of the identification, the basic construction of the fingerprint is nearly optimal for the datacenter on which we run our experiments. All the loss in accuracy of identification is caused by the realistic need of making decisions online after getting information about crises one at a time. In this realistic setting our method achieves 80% accuracy and on average identifies an anomaly in under ten minutes. Our evaluation is based on 19 real performance crises ranging from configuration issues, to performance problems due to high workload, to issues with other datacenters that caused back-ups in the system. Because of the excellent results of our fingerprints, we are currently engaged with the owners of the service in designing a pilot program to have this approach work in advisory mode with live data.

We remark that fingerprints are interpretable by human operators and can be used as visualization aids for diagnosis purposes (of new crises) and for monitoring the datacenter after performing a recovery action. When we showed a few of these fingerprints to the application operators, they very quickly recognized most of the crises.

Finally, it is worth relating a piece of anecdotal evidence reflecting the impact that this work has had. A month after one of our briefings with the administrators of the system, there was an accident that threatened the physical security of the datacenter. An operator was dispatched with specific instructions putting priority on backing up the collected performance metrics that we had pointed out as being correlated with the crises.

## Appendix: Comparison to Signatures

As mentioned in Section 4 we needed to modify the approach in [6] for our experiments. In this appendix we detail the modifications. Two factors make "apples to apples" comparison impossible. First, the signatures approach was applied to a black-box system comprising a handful of servers. Since we consider an application running on hundreds of servers, to apply the signature approach we must either build a model for each server and aggregate the models, or else aggregate the data from the hundreds of servers and build the model from the aggregates. The former approach was proposed in [3], but we note that this would result in a representation whose size is exponential in the number of metrics involved in the signature. (That paper did not conduct any experiments on a scale comparable to ours.) We therefore apply the latter approach, which we believe to be more in the spirit of the signatures work anyway, using quantiles as the method of aggregating metrics across servers and then building the models based on the quantiles.

The second difference is that the signatures approach maintains multiple models, one for each crisis seen, and computes a particular fitness score (the Brier score) to decide which model(s) are best to use for constructing the signature of (and therefore identifying) a particular crisis in progress. To simulate this, we assume perfect future knowledge and we generate an optimal model for each particular crisis. This has the effect of assuming that the model-selection machinery in the signatures work always and makes the optimal choice.

Lastly, the third difference is that, as explained in Section 3.4 we use regularized logistic regression as a classifier (for the metric selection part). To complete the building of the signature, once we have the log-linear model generated by the classifier, we set up to generate a threshold for each metric. To accomplish this we use the same classifier on each metric in isolation. This is equivalent in this context to the threshold taken from the Bayesian classifier used in "attribution" for generating the signatures in [6].

By assuming optimal model management and optimal online model selection, and by using a more appropriate classifier for the dataset at hand we claim we are using an optimistic version of [6] in our comparison. Evidence that indeed it is capturing signal about the state of the datacenter is that it did perform better than the baseline approaches consisting of using all the metrics in the fingerprints, and the one relying only on the KPIs.

## Acknowledgments

Many thanks to David Lafaurie for his help in labeling the crises. D. Terry and M. Abadi provided useful comments on an early draft. This work was done while Peter was an intern with Microsoft Research Silicon Valley.

## 9. REFERENCES

[1] HP OpenView, welcome.hp.com/country/us/en/prodserv/software.html.

[2] P. Barham, A. Donnelly, R. Isaacs, and R. Mortier. Using magpie for request extraction and workload modelling. In *OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*, pages 18–18, Berkeley, CA, USA, 2004. USENIX Association.

[3] P. Bodík, M. Goldszmidt, and A. Fox. Hilighter: Automatically building robust signatures of performance behavior for small- and large-scale systems. In A. Fox and S. Basu, editors, *SysML*. USENIX Association, 2008.

[4] M. Y. Chen, E. Kıcıman, A. Accardi, E. A. Brewer, D. Patterson, and A. Fox. Path-based failure and evolution management. In *Proc. 1st USENIX/ACM Symposium on Networked Systems Design and*

*Implementation (NSDI'04)*, San Francisco, CA, March 2004.

[5] I. Cohen, M. Goldszmidt, T. Kelly, J. Symons, and J. S. Chase. Correlating instrumentation data to system states: A building block for automated diagnosis and control. In *Proc. 6th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2004)*, San Francisco, CA, Dec 2004.

[6] I. Cohen, S. Zhang, M. Goldszmidt, J. Symons, T. Kelly, and A. Fox. Capturing, indexing, clustering, and retrieving system history. In A. Herbert and K. P. Birman, editors, *SOSP*, pages 105–118. ACM, 2005.

[7] B. Cook, S. Babu, G. Candea, and S. Duan. Toward Self-Healing Multitier Services. 2007.

[8] S. Duan and S. Babu. Guided problem diagnosis through active learning. In *ICAC '08: Proceedings of the 2008 International Conference on Autonomic Computing*, pages 45–54, Washington, DC, USA, 2008. IEEE Computer Society.

[9] M. Goldszmidt, I. Cohen, S. Zhang, and A. Fox. Three research challenges at the intersection of machine learning, statistical inference, and systems. In *Proc. Tenth Workshop on Hot Topics in Operating Systems (HotOS-X)*, Santa Fe, NM, June 2005.

[10] S. Guha and A. McGregor. Stream order and order statistics: Quantile estimation in random-order streams. *SIAM Journal on Computing*, 38(5):2044–2059, 2009.

[11] K. Koh, S.-J. Kim, and S. Boyd. An interior-point method for large-scale L1-regularized logistic regression. *Journal of Machine Learning Research*, 8:1519–1555, 2007.

[12] N. Lachiche and P. Flach. Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves. In *20th International Conference on Machine Learning (ICML03)*, 2003.

[13] M. Massie. The ganglia distributed monitoring system: design, implementation, and experience. *Parallel Computing*, 30(7):817–840, July 2004.

[14] D. Patterson, A. Brown, P. Broadwell, G. Candea, M. Chen, J. Cutler, P. Enriquez, A. Fox, E. Kiciman, M. Merzbacher, D. Oppenheimer, N. Sastry, W. Tetzlaff, J. Traupamn, and N. Treuhaft. Recovery oriented computing (roc): Motivation, definition, techniques, and case studies. Technical report, UC Berkeley, March 2002.

[15] S. Pertet, R. Gandhi, and P. Narasimhan. Fingerpointing correlated failures in replicated systems. In *SYSML'07: Proceedings of the 2nd USENIX workshop on Tackling computer systems problems with machine learning techniques*, pages 1–6, Berkeley, CA, USA, 2007. USENIX Association.

[16] J. A. Redstone, M. M. Swift, , and B. N. Bershad. Using computers to diagnose computer problems. In *9th Workshop on Hot Topics in Operating Systems (HotOS-IX)*, Elmau, Germany, 2003.

[17] P. Reynolds, J. L. Wiener, J. C. Mogul, M. A. Shah, C. Killian, and A. Vahdat. Experiences with pip: finding unexpected behavior in distributed systems. In *SOSP '05: Proceedings of the twentieth ACM symposium on Operating systems principles*, pages 1–2, New York, NY, USA, 2005. ACM.

[18] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. High speed and robust event correlation. *Communications Magazine, IEEE*, 34(5):82–90, 1996.

[19] M. Young and P. T. Hastie. L1 regularization path algorithm for generalized linear models, 2006.

[20] C. Yuan, N. L. J.-R. Wen, J. Li, Z. Zhang, Y.-M. Wang, and W.-Y. Ma. Automated known problem diagnosis with event traces. In *EuroSys 2006*, Leuven, Belgium, April 2006.

[21] S. Zhang, I. Cohen, M. Goldszmidt, J. Symons, and A. Fox. Ensembles of models for automated diagnosis of system performance problems. In *2005 Intl. Conf. on Dependable Systems and Networks (DSN 2005)*, Yokohama, Japan, June 2005.